Technical Report

UCAM-CL-TR-688
ISSN 1476-2986

Number 688

UNIVERSITY OF CAMBRIDGE
Computer Laboratory

Name-passing process calculi: operational models and structural operational semantics

Sam Staton

June 2007

15 JJ Thomson Avenue
Cambridge CB3 0FD
United Kingdom
phone +44 1223 763500

http://www.cl.cam.ac.uk/

This technical report is based on a dissertation submitted December 2006 by the author for the degree of Doctor of Philosophy to the University of Cambridge, Girton College.

This version of the report incorporates minor changes to the June 2007 original, which were released March 2008.

Technical reports published by the University of Cambridge Computer Laboratory are freely available via the Internet:

http://www.cl.cam.ac.uk/techreports/

ISSN 1476-2986

Summary

This thesis is about the formal semantics of name-passing process calculi. We study operational models by relating various different notions of model, and we analyse structural operational semantics by extracting a congruence rule format from a model theory. All aspects of structural operational semantics are addressed: behaviour, syntax, and rule-based inductive definitions.

A variety of models for name-passing behaviour are considered and developed. We relate classes of indexed labelled transition systems, proposed by Cattani and Sewell, with coalgebraic models proposed by Fiore and Turi. A general notion of structured coalgebra is introduced and developed, and a natural notion of structured bisimulation is related to Sangiorgi's open bisimulation for the π-calculus. At first the state spaces are organised as presheaves, but it is reasonable to constrain the models to sheaves in a category known as the Schanuel topos. This topos is exhibited as equivalent to a category of named-sets proposed by Montanari and Pistore for efficient verification of name-passing systems.

Syntax for name-passing calculi involves variable binding and substitution. Gabbay and Pitts proposed nominal sets as an elegant model for syntax with binding, and we develop a framework for substitution in this context. The category of nominal sets is equivalent to the Schanuel topos, and so syntax and behaviour can be studied within one universe.

An abstract account of structural operational semantics was developed by Turi and Plotkin. They explained the inductive specification of a system by rules in the GSOS format of Bloom *et al.*, in terms of initial algebra recursion for lifting a monad of syntax to a category of behaviour. The congruence properties of bisimilarity can be observed at this level of generality. We study this theory in the general setting of structured coalgebras, and then for the specific case of name-passing systems, based on categories of nominal sets.

At the abstract level of category theory, classes of rules are understood as natural transformations. In the concrete domain, though, rules for name-passing systems are formulae in a suitable logical framework. By imposing a format on rules in Pitts's nominal logic, we characterise a subclass of rules in the abstract domain. Translating the abstract results, we conclude that, for a name-passing process calculus defined by rules in this format, a variant of open bisimilarity is a congruence.

Acknowledgements

This thesis would not have been possible without the guidance of Marcelo Fiore, my Ph.D. supervisor. Much of this research develops material in an article that he wrote with Daniele Turi [Fiore and Turi, 2001].

I am very grateful to Gordon Plotkin and Andrew Pitts for examining my thesis. Specifically, Andy suggested that I include a proof of Prop. 7.3.1, which is less straightforward than I had first thought.

I have enjoyed discussing the material presented here with many people at a variety of occasions. Some of the anonymous referees of our articles gave good advice. There are a few people I will mention explicitly: Chung-Kil Hur, with whom I enjoyed talking about the more category-theoretic aspects of this research; Bartek Klin, who made insightful suggestions about the presentation of this material; Dale Miller and Catuscia Palamidessi, who explained some aspects of their work to me (mentioned briefly in Section 9.2); John Power, who pointed me towards Max Kelly's work for Props. 2.3.1 and 6.1.4; Peter Sewell, with whom I have extensively discussed the material of Chapter 3; and Kidane Yemane, with whom I discussed the ideas of Section 5.2 when they were at a very early stage, as well as the problems mentioned in Section 9.3.5. Glynn Winskel, who has been a second supervisor, deserves a special mention, for his ongoing encouragement and support.

I must express my gratitude to my family and friends: to Meinou, for her continued patience and kindness; to my parents, for encouraging me to study at Cambridge in the first place; to all the people I have lived with over the past four years; to the Simmons family, for their hospitality, and for the memory of Colin, an academic of the highest order; and to my many friends at the laboratory in Cambridge with whom I have enjoyed eating and drinking. I am grateful to David Richerby for first interesting me in theoretical computer science, and Alan Blackwell for supervising my first research project.

I am grateful to Julian Owens of Qubix International for financial help and work experience. This research in this thesis was funded by a Research Studentship from the Engineering and Physical Science Research Council (EPSRC), and for the final part by the EPSRC Project GR/T22049/01, Domain Theory for Concurrency, for which the principle investigators are Glynn Winskel and Marcelo Fiore.

Sometimes he [Pierre] remembered how he had heard that soldiers in war when entrenched under the enemy's fire, if they have nothing to do, try hard to find some occupation the more easily to bear the danger. To Pierre all men seemed like those soldiers, seeking refuge from life: some in ambition, some in cards, some in framing laws, some in women, some in toys, some in horses, some in politics, some in sport, some in wine, and some in governmental affairs. "Nothing is trivial, and nothing is important, it's all the same — only to save oneself from it as best one can," thought Pierre. "Only not to see it, that dreadful it!"

Leo Tolstoy
War and Peace, Book Eight, Chapter I
Translated by Louise and Aylmer Maude

Contents

Chapter 1

Introduction

We survey the background for this thesis, and outline the main contributions.

1.1 Background

We set the background for this thesis by summarising the key developments in theoretical computer science that form its basis. Relevant aspects of name-passing process calculi, and the π-calculus in particular, are considered. We then introduce some models for name-passing calculi, with an emphasis on operational models for bisimulation. This background section concludes with a survey of relevant aspects of the structural approach to operational semantics, with particular attention paid to category-theoretic analysis, and issues of variable binding.

Note. The research projects mentioned here all have a direct impact on the present work. The reader should turn to the concluding chapter of this thesis to find a discussion of more orthogonal problems that go beyond the scope of this thesis.

1.1.1 Rudiments of name-passing process calculi

Name-passing process calculi are calculi that involve the communication of names along named channels. Channels for communication have been named for as long as communication channels have been considered, but the novelty of name-passing is that these names can also be communicated along those channels. As Needham [1989] explains, the concept of *pure name* is fundamental to computer science.

Milner's Calculus of Communicating Systems [CCS: 1980; 1989] supports the communication of values along channels, and even in his first introduction Milner mentioned his interest in the idea of name-passing [Milner, 1980, 11.5(i)]. A first serious attempt at a name-passing calculus was made by Engberg and Nielsen [see 2000], but there can be no doubt that the most notable and pervasive calculus with this facility is the π-calculus as introduced by Milner, Parrow, and Walker [1992]. Full details of the π-calculus are provided in the article of Milner *et al.*, and in Section 3.1 of this thesis, and also in the comprehensive book by Sangiorgi and Walker [2001]. The reader will also find insight in Parrow's introduction [2001] and in Milner's book [1999]. For now, we take a cursory glance at some of the important aspects of this calculus. These aspects may indeed be considered as important characteristics of any reasonable name-passing calculus.

Restriction and scope extrusion. The syntax of the π-calculus includes a restriction operator, ν. When names are considered as channels, this serves to hide a name within a process. For instance, the π-calculus term $P = (\bar{c}d.0 \mid c(d).0)$ can perform an output action, of name d on channel c, to become the process $(0 \mid c(d).0)$; alternatively, the process P can perform a silent action τ during which the two parallel processes communicate to become $(0 \mid 0)$. If the name c is restricted, as in

the process $\nu c.(\bar{c}d.0 \mid c(d).0)$, then the latter transition remains possible, while the former does not, because the channel c is hidden.

The ν operator acts to hide names when they are used as channels, and this is in common with the restriction operator of CCS. But when names are considered as data, the operator ν has a different behaviour. The process $\nu d.\bar{c}d.0$ can perform an output action; here it is helpful to think of the operator ν as describing the generation of a new name d. The output of newly generated names is described as *bound output*. This behaviour gives rise to the phenomenon of *scope extrusion*. For instance, in the π-calculus, we have a silent transition

$$(\nu d.\bar{c}d.Q) \mid c(d).Q' \xrightarrow{\tau} \nu d.(Q \mid Q')$$

during which the scope of d extrudes to include Q'. (Here Q and Q' are arbitrary π-calculus terms.) This phenomenon has been used to describe mobility of communication links.

Equivalences for name-passing systems. No study of a process calculus is complete without a discussion of appropriate notions of behavioural equivalence. Sangiorgi and Walker [2001, Chapters 2 and 4] provide a thorough discussion of behavioural equivalences for the π-calculus.

In this thesis we focus on (strong) bisimulation equivalences. This is primarily because they are elegant from a mathematical perspective, though there are also practical reasons for this approach. Indeed, no matter what equivalence happens to be relevant from the pragmatic viewpoint, one cannot dispute the power of proof techniques based on bisimulation. There are automatic and efficient 'partition refinement' techniques for finite state systems [see *e.g.* Paige and Tarjan, 1987], and, even when these are not relevant, 'bisimulation up-to' techniques are powerful [an account is given by Sangiorgi, 1998]. From this point of view, bisimilarity need not be the most relevant equivalence, for it is sufficient that it is contained in a relevant equivalence, *i.e.* that the bisimulation proof techniques be sound. In most situations this is the case, for bisimilarity is the most refined equivalence that it is ever reasonable to consider. Ideally, though, a proof technique is complete, and there has been a significant amount of interest in characterising important equivalences in terms of bisimulations. For instance, notions of contextual equivalence [*e.g.* Gordon, 1999; Jeffrey and Rathke, 2005] and observational equivalence [*e.g.* Sewell, 2002; Leifer and Milner, 2000] have been characterised in terms of bisimulation.

When defining bisimulation for the π-calculus, some care must be taken over free and bound variables. For instance, the process $P_1 = \nu d.\bar{c}d.0$ can perform a bound output of name e on channel c, whereas the process $P_2 = \nu d.\bar{c}d.[e = e]0$ cannot, because the name e happens to appear free in that term. The processes P_1 and P_2 should not be distinguished for this reason, and so any notion of bisimulation should only consider bound labels where the binder is sufficiently fresh.

A second complication for bisimilarity in the π-calculus is that none of the most natural notions of bisimilarity are congruences; that is, they are not respected by the syntax. To see this, notice that the process $P_1 = [c = d]\bar{c}d.0$ is certainly bisimilar with $P_2 = 0$, because the name c is different from the name d and so neither process can reduce. The context $(\bar{c}d.0 \mid c(c).[-])$ can distinguish these two processes: we have the sequence of transitions

$$\bar{c}d.0 \mid c(c).P_1 \;=\; \bar{c}d.0 \mid c(c).[c = d]\bar{c}d.0 \xrightarrow{\tau} 0 \mid [d = d]\bar{d}d.0 \xrightarrow{\bar{d}d} 0 \mid 0$$

which cannot be matched by $(\bar{c}d.0 \mid c(c).P_2)$.

Congruence of bisimilarity is important for a variety of reasons, depending on the one's motivation for working with bisimilarity. If bisimilarity is a relevant equivalence in its own right, then it must be a congruence if one is to have a compositional understanding of systems. If bisimilarity is used as a characterisation of another specified process equivalence, then the property of bisimilarity being a congruence is often crucial in the characterisation proof. Even if we are only interested in a sound bisimulation proof technique, congruence of bisimilarity permits compositional reasoning within that proof technique.

The above example illustrates that the failure of congruence of bisimilarity for the π-calculus arises because bisimilarity is not closed under arbitrary substitutions, while input contexts are able to force these substitutions. The solution that we take in this thesis is to consider only those bisimulations that are closed under arbitrary substitutions. We say that these bisimulations are *wide open*, and the greatest such bisimulation we call *wide open bisimilarity*. This equivalence is a congruence for the π-calculus.

Wide open bisimilarity has been studied by various authors, although under different names. It is the equivalence used in the models of Cattani and Sewell [2004] (there called open bisimilarity), and in the models of Fiore and Turi [2001] (called early/late congruence). Sangiorgi [1996] studies wide open bisimilarity for a fragment of the π-calculus, and Sangiorgi and Walker [2001, Sec. 4.6] briefly record some properties. Wide open bisimilarity is related to the notion of *hyperbisimilarity* considered for the *update* calculus of Parrow and Victor [1997]. It is a mathematically natural notion: firstly, wide-open bisimilarity is (strong) *dynamic congruence* [in the sense of Montanari and Sassone, 1992]; secondly, the coalgebraic models of wide open bisimilarity of Section 3.4 (of this thesis) arise in a rather elegant way.

It has been argued, however, that wide open bisimilarity is *too* fine, in that it distinguishes processes that could perhaps never be told apart. Some more elaborate notions have been suggested to deal with some anomalies. These include open bisimulation [Sangiorgi, 1996, and Section 9.3.5 of this thesis] and symbolic bisimulation [see *e.g.* Hennessy and Lin, 1995; Boreale and Nicola, 1996; Lin, 2003], but will not concern us here.

1.1.2 Models of name-passing

The presentation of the π-calculus semantics of Milner *et al.* [1992] is in the *structural operational semantics* style of Plotkin [1981]. This style of semantics is concerned with those transitions that are provable, and as such the theory can be developed to a very large extent without paying much attention to notions of model. This proof-theoretic attitude is further developed by Miller [*e.g.* 2006] and others.

In spite of this, model theories play an important role in theoretical computer science. For one thing, they distinguish aspects of behaviour from the syntax of languages, and as such provide an important abstraction mechanism. By studying notions of model, important properties of behaviour can be isolated.

The first explicit work on models for the π-calculus appeared in 1996, with the domain-theoretic models of Fiore, Moggi, and Sangiorgi [2002] and Stark [1996]. A central concern in giving denotational semantics for name-passing systems is that the free names of a term must be captured abstractly in the model: firstly, this is convenient when specifying a denotational semantics; secondly, the notions of bisimulation for name-passing explicitly involve the free names of terms. The two domain-theoretic models both address this concern in the same way, by indexing their category of domains by the category $\mathbf{I}$ of finite sets of names and injective functions between them, or equivalently the skeletal category $\mathbb{I}$ of natural numbers (considered as sets) and injective functions between them. Thus one seeks a solution P of a domain equation in a functor category $\mathbf{Dom}^{\mathbf{I}}$ (here $\mathbf{Dom}$ is an appropriate category of domains). For each set of names $C \in \mathbf{I}$ there is thus a domain $P(C)$ of denotations of name-passing processes with free names C, and for each injection $\iota : C \rightarrowtail D$ in $\mathbf{I}$ there is a continuous map $P(C) \to P(D)$ converting denotations of processes involving names C into denotations of processes involving names D; after all, behaviour should be stable under injective substitution.

It seems that fully abstract models could still have been found if the indexing category $\mathbf{I}$ was replaced with the lattice of finite sets of names — that is, if the non-identity bijection maps were stripped from $\mathbf{I}$. The bijective morphisms in $\mathbf{I}$ are not redundant, though; they imbue a certain uniformity into the model. One might argue that the best model is the one that fits tightest around

the intended semantics while remaining independent from syntax. This will be a guiding principle in the development of models for name-passing in this thesis.

Operational models of name-passing. We mention here three different approaches to modelling name-passing calculi. The first approach has its roots in the denotational semantics of CCS originally offered by Milner [1980]. There, the models are somewhat 'operational': *synchronisation trees* explictly describe the stepwise evolution of processes, and are not derived from abstract mathematical constructions. However, as Aczel [1988, Chapter 8] observed, quotiented synchronisation trees can be equivalently seen as elements of the final coalgebra for the endofunctor $\mathscr{P}(Lab \times (-))$ on the category of classes. (Here, $\mathscr{P}$ is a covariant powerset functor, and Lab is a suitable set of labels.) Indeed, Abramsky [1991, Sec. 7] has noted that an operational semantics, *i.e.* a labelled transition system, can be understood as a coalgebra for the above-mentioned endofunctor, explictly by giving a set X of states together with a 'next-step' function $X \to \mathscr{P}(Lab \times X)$; on the other hand, as Abramsky notes, a *denotational* semantics for such a system can be given in a solution for a domain equation of the form $D \cong \mathscr{P}(Lab \times D)$, where $\mathscr{P}$ is now considered as a powerdomain construction. Thus coalgebraic principles became recognised from the domain-theoretic perspective. These principles are made explicit in the abstract development of Freyd [1991] and in the techniques of Pitts [1994], and in other related work.

In this way, using notions of algebraic compactness [Freyd, 1991, 1992], the solutions of domain equations for name-passing systems proposed by Fiore *et al.* [2002] and Stark [1996] are not only initial solutions but also final solutions. The resulting coinductive reasoning technique is essential, and allows the domains to be considered from an operational stance. Thus the order structures are not necessary: one can provide fully abstract semantics in the final coalgebra of a certain endofunctor on $\mathbf{Set}^{\mathbf{I}}$. This idea is made explicit in the work of Fiore and Turi [2001, Sec. 2.2].

From the operational perspective, finality is not especially important. All the coalgebras provide a general notion of transition system, and notions of bisimulation can be studied there. This viewpoint was developed in the work of Rutten [*e.g.* 2000] and others throughout the 1990s. Thus ideas from operational semantics become relevant at the level of model theory.

A rather different approach to operational models of name-passing calculi is provided by Cattani and Sewell [2004] in their provision of a denotational semantics of the π-calculus. Their models are based on a kind of labelled transition system. Let $\mathbf{F}$ be the full subcategory of $\mathbf{Set}$ whose objects are finite sets of names, and, instead of a set of states, a functor $P : \mathbf{F} \to \mathbf{Set}$ (a 'presheaf') is given. The intention is that, for any set C of names, there is a set $P(C)$ of states that use some of the names in C, and that the functorial action of P describes a substitution of names in states. The labelled transition systems that they consider are over the sets of *elements* of the presheaf of states; an element (C, p), with $p \in P(C)$, represents the state p in name-context C.

The structuring of the states in this way allows, firstly, for appropriate notions of bisimulation to be defined at the level of the model theory. Secondly, it allows the class of models to be restricted by imposing conditions on how the transition system interacts with the indexing by name contexts. A most basic example is that if there is an output transition

$$(C, p) \xrightarrow{c!d} (D, q)$$

then the channel c on which the output occurs must be known by the process — so c must be in the set C. (The idea of presenting the semantics of the π-calculus with explicit name contexts appears elsewhere in less model-theoretic presentations: see *e.g.* Sewell [2001], Bruni *et al.* [2004].)

The final operational model that we mention here is the model of *history dependent automata* due to Montanari and Pistore [1997]. The main idea in this work is to regard models of the π-calculus as automata with extra structure. This approach is especially amenable to efficient verification techniques. A problem with model-checking for the π-calculus is that an infinity of

names must be considered when checking input transitions, and to solve this problem one must recognise that all these names are treated uniformly. History dependent automata deliver this solution in a particularly efficient way: the states of history dependent automata provide canonical representations of π-calculus terms up-to renaming of *free* variables, and so the state space can be significantly reduced.

1.1.3 Structural operational semantics

Rule formats. The Aarhus notes of Plotkin [1981] introduced the structural approach to operational semantics (SOS). Soon after, the work of de Simone [1985] demonstrated that whole classes of systems can be simultaneously studied in a formal way. The idea of de Simone was that instead of studying the particular semantics of a particular system, one could consider an arbitrary SOS specification, and that by imposing constraints on the kinds of rules under consideration, results could be established for arbitrary systems in general. An overview of such rule formats is provided by Aceto *et al.* [2001, Sec. 5]. The rule format with which we are most interested is a positive version of the GSOS format of Bloom, Istrail, and Meyer [1995]. To paraphrase Bloom *et al.*: a rule for defining a labelled transition system is in the *positive GSOS format* if it has the form

$$\frac{\bigcup_{i=1}^{l} \left\{ X_i \xrightarrow{a_{ij}} Y_{ij} \;\middle|\; 1 \leq j \leq m_i \right\}}{\mathsf{op}(X_1, \ldots, X_l) \xrightarrow{c} C\left[\vec{X}, \vec{Y}\right]}$$

where all variables are distinct, $l \geq 0$ is the arity of op, $m_i \geq 0$, and $C\left[\vec{X}, \vec{Y}\right]$ is a context with free variables including at most the X's and Y's. Bloom *et al.* [1995, Sec. 5] establish a variety of results about systems defined by such rules. Importantly, bisimilarity is a congruence for such systems.

Mathematical Structural Operational Semantics. The rule formats of Bloom *et al.* [1995] are general and profound. To understand what is going on at an abstract level, though, one can study the structures and procedures in terms of category theory. Structural operational semantics is concerned with the interplay between behaviour and syntax, by means of recursion on the structure of the syntax. In this Introduction, we have already discussed how operational models can be considered in a general category-theoretic way using coalgebras and bisimulation. For a long time, the concepts of syntax, recursion and induction have been understood in terms of the category-theoretic machinery of initial algebras and free monads; an early introduction for computer science is provided by Goguen, Thatcher, and Wagner [1978]. Thus structural operational semantics can be studied in the setting of category theory.

The concepts of structural operational semantics were first understood at this more abstract level by Turi and Plotkin [1997]. (The ideas that we now overview are made precise in that paper and also in Chapter 6 of this thesis.) It is helpful to think of two universes. The first is a category $\mathscr{S}$ which is a domain for syntax, and as such is equipped with a monad T whose algebras are models of that syntax. The second universe is a category $\mathscr{B}$ of behaviour, whose objects represent denotations of systems. These categories are typically related by a forgetful functor $\mathscr{B} \to \mathscr{S}$, mapping a system to its set of states. For an example, we consider the case where $\mathscr{S} = \mathbf{Set}$; the monad T is the monad of terms of Milner's CCS [1980]; $\mathscr{B}$ is the category of coalgebras for the endofunctor $\mathscr{P}(Lab \times (-))$, or equivalently, $\mathscr{B}$ is a category of labelled transition systems with functional simulations between them. An important guiding principle that holds at the abstract level is that

> a lifting of a monad T on $\mathscr{S}$ along the forgetful functor $\mathscr{B} \to \mathscr{S}$ is a specification of well-behaved semantics.

For the situation of CCS, just described, this means that the initial algebra for a lifting of the monad T (of CCS terms) to the category of $\mathscr{P}(Lab \times (-))$-coalgebras provides a labelled transition system semantics over CCS terms, for which bisimilarity is a congruence.

Under certain conditions, rule-based inductive definitions for operational semantics can be seen as the definition of a monad lifting by initial algebra recursion. For a specific example, consider one of the CCS rules for synchronisation

$$\frac{P \xrightarrow{a} Q \qquad P' \xrightarrow{\bar{a}} Q'}{P \,|\, P' \xrightarrow{\tau} Q \,|\, Q'}$$

which, informally, says that if P and P' can perform complementary actions then they will synchronise in parallel and only a silent (τ) action will occur. As Plotkin [2004] observes, the rules will typically be read only in a clockwise manner, so that the following alternative presentation is sensible.

$$P \left\{ \xrightarrow{a} Q \right\} \;\Big|\; P' \left\{ \xrightarrow{\bar{a}} Q' \right\} \;\xrightarrow{\tau}\; Q \;\Big|\; Q'$$

This latter presentation is to be thought of as saying that if P, which can perform an action a to become Q, is in parallel with P', which can perform action $\bar{a}$ to become Q', then P and P' will synchronise to become Q in parallel with Q'. Here, the source (left-hand side) of this generic transition is a basic expression of the language that additionally involves behaviours. Thus the source is an element of the set $\Sigma(X \times BX)$, where: X is a set of process variables, including P, P', Q and Q'; the endofunctor Σ on **Set** defines the signature of CCS syntax, and is a sum including a term $((-) \times (-))$ for the operator of parallel composition; and B is the endofunctor $\mathscr{P}(Lab \times (-))$ on **Set** whose coalgebras are labelled transition systems. On the other hand, the transition label and the target (right-hand side) together comprise a behaviour involving a term of the language; that is, an element of the set BTX. In this way we can see GSOS rules as functions from $\Sigma(X \times BX)$ to BTX. Natural families of such functions, *i.e.* natural transformations

$$\Sigma((-) \times B(-)) \to BT(-) \tag{1.1.1}$$

between endofunctors on **Set**, provide the data for using initial algebra recursion to lift the monad T to the category of T-coalgebras. Thus the congruence of bisimilarity for GSOS systems can be understood at an abstract level.

Syntax with variable binding. An inadequacy of traditional rule formats, such as the GSOS format, with respect to modern semantics research, is that they are not relevant when syntax involves variable binding. Variable binding is an essential part of the syntax for value-input in any value-passing process calculus, and in name-passing calculi in particular. Moreover, in the π-calculus, the binding behaviour of the restriction operator plays an essential role.

Variable binding can be a tricky issue in operational semantics. For a simple example, suppose that an early semantics for input in the π-calculus was specified by the following axiom.

$$\frac{}{c(c).P \xrightarrow{cd} [d/c]P'} \tag{1.1.2}$$

One might argue informally that c is binding in P, and so the axiom can be equivalently written as follows.

$$\frac{}{c(z).P \xrightarrow{cd} [d/z]P'}$$

However, let us take the axiom (1.1.2) *literally* for a moment. Then the process $P_1 = c(z).0$ is α-equivalent to $c(c).0$, and from this we can use the axiom (1.1.2) to derive a transition $c(c).0 \xrightarrow{cd} 0$. On the other hand, though, the process $P_2 = c(z).[c = c]\,0$ cannot be α-converted

into a form relevant to the axiom (1.1.2), for certainly P_2 is not α-equivalent to $c(c).[c = c]\,0$. So we cannot derive any input transitions for P_2. This is obscure: the test $[c = c]$ in process P_2 is surely redundant and in any sensible semantics, P_1 and P_2 should have the same behaviour.

Fokkink and Verhoef [1998], Middelburg [2001] and others have studied properties and meanings of transition system specifications over syntax with variable binding. Their work provides guidance for any subsequent study of operational semantics over syntax with binding. But these developments cannot be immediately studied in a mathematical way, at the level of category theory, because the model of syntax developed by Goguen *et al.* [1978] is not immediately relevant in the variable-binding setting.

Variable binding became something of a 'hot-topic' in the late 1990s and the early 2000s, when a myriad of formal frameworks, both model-theoretic and proof-theoretic, were suggested. A suitable framework should be rigorous, and yet relevant to the informal intuitions one has when reasoning about binding. For instance, the convention of Barendregt [1981, 2.1.13] is too informal for some purposes, while the notation of de Bruijn [1972] is rigorous and efficient, yet difficult to read.

In this thesis, we focus on the models of Fiore, Plotkin, and Turi [1999] and of Gabbay and Pitts [*e.g.* 2001]. The approach of Fiore *et al.* is as follows. Instead of working with sets of terms, one works with variable sets, *i.e.* functors $X : \mathbb{F} \to \mathbf{Set}$; here $\mathbb{F}$ is the category of natural numbers (considered as sets) and functions between them, so it is a skeleton of the category $\mathbf{F}$ mentioned above. The intention is that, for any number n, a set $X(n)$ of terms with at most n free variables is given, and for each function $f : n \to m$ between numbers, a renaming function $X(f) : X(n) \to X(m)$ is specified. The 'set' of all variables can be seen as the inclusion V of $\mathbb{F}$ in $\mathbf{Set}$. The presheaf category $\mathbf{Set}^{\mathbb{F}}$ is Cartesian closed, as usual, and exponentiation by V is an abstract form of variable binding, as can be seen by the equation $X^V(n) = X(n+1)$.

Substitution can be formulated within this framework. There is a monoidal structure on $\mathbf{Set}^{\mathbb{F}}$, for which the tensor defines a variable set of substitutable pairs. A substitution structure can then be defined as a monoid over this monoidal structure.

This framework provides an abstract account of the principles of variable binding and substitution. One can use this framework to study principles of recursion and induction for syntax with variable binding, in terms of initial algebras and free monads.

These techniques and constructions, however elegant, remain somewhat removed from the informal pen-on-paper arguments that mathematicians and computer scientists use every day. It is in this aspect that the framework of nominal sets, developed by Gabbay and Pitts [2001], excels. The basic idea of nominal sets is to consider sets that are equipped with actions of the symmetric group on names. In other words, we consider sets X that are equipped with an assignment from permutations of variables to functions $X \to X$ describing how the permutations 'rename' the elements of X (all subject to the requirement that the identity permutation and composition of permutations are suitably respected). Notions of 'free variable' can be considered at this abstract level using the concept of *support*. Notions of α-equivalence can be defined in terms of the permutation action structure.

An important attraction of nominal sets, in contrast with the functor category approach, is that one can work with a set of terms, as in traditional studies of syntax, and consider the permutation action as an additional structure. Construction and manipulation of nominal sets can then be seen in terms of the traditional construction and manipulation of sets, provided that the additional group action structure is respected.

Abstract notions of substitution have not been extensively studied in the setting of nominal sets [although first steps are made by Gabbay and Mathijssen, 2006]. It is well-known, however, that the category of nominal sets is equivalent to a sheaf topos, and in this way the theory of nominal sets can be connected with the framework of Fiore *et al.*.

1.2 Contributions

The pinnacle of this thesis is the rule format for name-passing that is presented in Chapter 8: the important result is that for any system defined using rules in this format, wide open bisimilarity is a congruence. This format is derived from a model based on mathematical structural operational semantics, and the development and analysis of this model forms the body of this thesis.

We now collect and summarise the contributions of this thesis. The contributions are split into strands that span the length of the thesis. A chapter-by-chapter synopsis is provided in Section 1.3.

Indexed labelled transition systems and coalgebras. The coalgebraic model for the early semantics of name-passing proposed by Fiore and Turi [2001] involves an endofunctor on the presheaf category $\mathbf{Set}^{\mathbb{I}}$. (Here, as earlier, $\mathbb{I}$ is the category of natural numbers and injections between them.) Here, instead of $\mathbb{I}$, we work here with the equivalent category $\mathbf{I}$ of sets of names and injections between them. An additional novelty of our presentation is that we consider both an early semantics and a ground semantics. (The ground semantics is concerned only with input of fresh names, and it is not interesting in itself, but in the context of wide open bisimulation.)

The notion of behaviour suggested by the coalgebraic model of Fiore and Turi incorporates the complex constructions involved in the behaviour endofunctor. Moreover, morphisms in $\mathbf{Set}^{\mathbb{I}}$ are families of functions subject to a naturality requirement, and this requirement plays a central role. In this thesis we explore this notion of behaviour by making all the constructions and requirements explicit. To do this we require a more bland, permissive notion of model for name-passing.

The more permissive models that we study are *indexed labelled transition systems*, loosely following the analysis of Cattani and Sewell [2004]. An indexed labelled transition system over a presheaf $P \in \mathbf{Set}^{\mathbf{I}}$ is a labelled transition system in which the states are pairs (C, p) with C an object of $\mathbf{I}$, and $p \in P(C)$. Our intuitions about the coalgebraic model of Fiore and Turi suggest a straightforward translation from coalgebras to indexed labelled transition systems.

Indexed labelled transition systems thus provide a foundation within which the requirements of the coalgebraic notion of model can be studied explicitly. For instance, the naturality requirements on the coalgebra structures ensure that transitions are invariant under bijective renamings. We can write this as a property of indexed labelled transition systems: for any bijection $\beta : C \xrightarrow{\sim} D$ in $\mathbf{I}$,

$$\text{If } (C,p) \xrightarrow{l} (C,p') \text{ then } (D,[\beta]p) \xrightarrow{[\beta]l} (D,[\beta]p') \quad .$$

By axiomatising exactly those indexed labelled transition systems that are induced by coalgebras, we arrive at a characterisation of the coalgebraic model in which the requirements on behaviour are made explicit.

From another point of view, indexed labelled transition systems provide a basic model theory within which many lemmas about the transition systems of the π-calculus can be phrased without reference to syntax. The axiom about bijective renaming, mentioned above, is an important example of an abstract form of such a lemma. A notion of model of name-passing can thus be found by treating lemmas of the π-calculus as axioms on indexed labelled transition systems. As mentioned in the background discussion, this is the approach taken by Cattani and Sewell [2004]. A contribution of this thesis, then, is that this approach is related with the coalgebraic approach of Fiore and Turi.

Sheaf conditions on state spaces. The axioms imposed on indexed labelled transition systems help to characterise notions of behaviour for name-passing systems. However, no requirements are imposed on the presheaves of states, other than functoriality. In this thesis we assert that it is appropriate to impose a sheaf condition on the presheaves of states, and demonstrate that all the constructions and developments can be carried out for this restricted notion of state space. (This is an idea that was first considered in this context by Pitts and Stark [1993, Example 4.3(i),(iii)].)

The first step in this direction is to recognise that the presheaves that arise in models of the π-calculus have the property that the actions of the injections in **I** are themselves injective. This property of presheaves is not imposed by functoriality, but corresponds to separatedness for a certain topology on **I**. A further, more sophisticated observation is that for a presheaf $P \in \mathbf{Set}^{\mathbf{I}}$, if $p \in P(C)$ and $p \in P(D)$, we should imagine that the 'free names' of p are included both in C and in D, and so we can expect that $p \in P(C \cap D)$. This informal description, when formalised, constitutes the sheaf condition for that topology on **I**.

One can write down these requirements on presheaves with no knowledge of sheaf theory, but the sheaf-theoretic characterisations connect the work with a well-developed mathematical theory. Thus some basic constructions, such as limits and function spaces, can be immediately understood, and adjoint connections between models can be observed by using results of a general kind.

Indeed, the sheaf category that arises in this way, which has been called the *Schanuel topos*, is well known to be equivalent to the category of nominal sets of Gabbay and Pitts [*e.g.* 2001]. Thus the model using coalgebras over sheaves can be seen as a model of name-passing in nominal sets. Nominal sets have been championed as an elegant domain for models of syntax, and we are now able to assess their elegance in the context of behaviour. When developing a theory of semantics for name-passing, it is helpful to have syntax and semantics considered within the same universe.

The restriction to the sheaf subcategory permits a simpler characterisation of behaviour. Fiore [2001] has observed that sheaves in the Schanuel topos are freely determined by their action on bijections. Using this idea, we consider indexed labelled transition systems over presheaves in $\mathbf{Set}^{\mathbf{B}}$: here, **B** is the category (actually, groupoid) of sets of names and bijections between them. We use the constructions of Fiore [2001] to relate indexed labelled transition systems over $\mathbf{Set}^{\mathbf{B}}$ with those over sheaves in the Schanuel topos. The resulting axiomatisation of indexed labelled transition systems over $\mathbf{Set}^{\mathbf{B}}$ is simpler than that over $\mathbf{Set}^{\mathbf{I}}$, and thus a new model of name-passing is designed.

Another important property of the Schanuel topos is that it is equivalent to a category of named-sets used by Ferrari, Montanari, and Pistore [2002] in their article about model-checking with history dependent automata. This correspondence is seen to arise from a categorical version of the orbit-stabiliser theorem. Thus, by requiring extra properties of the state spaces, the models become amenable to efficient verification.

Name-for-name substitution. Arbitrary name-for-name substitutions play an important role in theory of name-passing. Consider the axiom for input in the early treatment, and one of the rules for communication in the late treatment. Both involve potentially non-injective substitutions in the right-hand-side of the conclusion.

$$\frac{}{c(z).P \xrightarrow{cd} [d/z]P} \text{ (input-early)} \qquad \frac{P \xrightarrow{c(z)} P' \quad Q \xrightarrow{\bar{c}d} Q'}{P \mid Q \xrightarrow{\tau} [d/z]P' \mid Q'} \text{ (com-l-late)}$$

Here we have a problem: the presheaves and sheaves over **I** that we have used for the models so far only support injective substitutions. To consider arbitrary substitutions within the abstract framework, we must move to presheaves over **F**, the category of finite sets of names and functions between them.

It is straightforward to compare the presheaf categories $\mathbf{Set}^{\mathbf{I}}$ and $\mathbf{Set}^{\mathbf{F}}$. However, if we are to restrict to a sheaf subcategory of $\mathbf{Set}^{\mathbf{I}}$, it makes sense to also restrict to a sheaf subcategory of $\mathbf{Set}^{\mathbf{F}}$. In this thesis we introduce an appropriate topology on **F**, and observe that the sheaf condition for this topology has a very simple meaning.

The sheaf subcategory of $\mathbf{Set}^{\mathbf{I}}$ is equivalent to the category of nominal sets, and this is an important equivalence to exploit. Thus we are led to develop a 'nominal' view of the sheaf subcategory of $\mathbf{Set}^{\mathbf{F}}$. We do this by defining a 'nominal algebraic theory' of *nominal substitutions*, whose

models in the category of nominal sets correspond to sheaves over **F**, and whose model homomorphisms correspond to natural transformations between sheaves. Axioms of the theory provide an abstract axiomatisation of the concept of substitution: we enforce the identity law of substitution, $[a/a]x = x$; we require that if a is fresh for x, then $[b/a]x = x$; and we also enforce a kind of substitution lemma stating when the order of substitutions is irrelevant.

Structured coalgebras and wide open bisimulation. It is common in operational semantics for the states of a system to have structure, some of which is not respected by the behaviour. The central example of this phenomenon for this thesis involves the arbitrary name-for-name substitutions that play an important role in name-passing calculi. The behaviour is not respected by such substitutions: one does not expect a theorem stating that for any substitution f,

$$x \xrightarrow{l} y \quad \text{if and only if} \quad [f]x \xrightarrow{[f]l} [f]y \quad .$$

(The left-to-right implication is an important property of some fragments of the π-calculus, but the right-to-left implication is certainly obscure.)

To cope with situations such as this we introduce a new and general concept of *structured coalgebra*. Consider a functor $U : \mathscr{D} \to \mathscr{C}$ together with an endofunctor B on $\mathscr{C}$. Then a *U-structured B-coalgebra* is an object X of $\mathscr{D}$ together with a morphism $UX \to BUX$ in $\mathscr{C}$ — *i.e.* a B-coalgebra structure for UX.

For modelling the π-calculus, the forgetful functor $U : \mathscr{D} \to \mathscr{C}$ is the obvious forgetful functor $\mathbf{Set}^{\mathbf{F}} \to \mathbf{Set}^{\mathbf{I}}$ (or a version thereof restricted to sheaf subcategories): "if you know what to do with all functions, you know what to do with injections". There is a natural notion of *structured bisimulation*, which, for the π-calculus, coincides with wide open bisimulation.

The correspondence between coalgebras and indexed labelled transition systems is important. To handle arbitrary substitutions in indexed labelled transition systems one can require that presheaves of states are over **F** rather than **I**, and the axioms on transition systems developed before still make sense. However, when arbitrary substitutions are in the model, it is possible to consider an extra axiom, about uniformity of input: informally,

$$\text{if } P \xrightarrow{cz} Q \text{ then } P \xrightarrow{cd} [d/z]Q \quad .$$

(Notice that the substitution in the consequent need not be injective.) Indexed labelled transition systems that satisfy this additional axiom are shown to correspond to indexed labelled transition systems with ground labels (with only fresh input data). Thus the ground semantics is relevant from the perspective of wide open bisimulation.

A rule format for name-passing calculi. The abstract developments of Turi and Plotkin [1997] can be reworked in the setting of structured coalgebras. We have models of both syntax and semantics in nominal sets and nominal substitutions, and thus an abstract GSOS-like rule format arises for name-passing calculi. Structured coalgebras defined using this format have the property that wide open bisimulation is a congruence.

A problem with this result is that rules in this format are not logical statements, but rather natural transformations between functors. So, while this development is illuminating at the conceptual level, the result is not much use to the working operational semanticist. The final contribution of this thesis is the extraction of a concrete rule format from this abstract result. This concrete rule format provides a notation, at a logical level, for a class of natural transformations from the abstract domain.

A first challenge is in identifying a logical formalism that is sufficiently expressive so as to support appropriate notions of rule. The format that we extract must be generous enough to

support the specification of the π-calculus. For an example of the kinds of feature that are required, consider the π-calculus rule for scope opening, as presented by Milner *et al.* [1992, II, Table 2].

$$\frac{P \xrightarrow{\bar{x}y} P'}{\nu y.P \xrightarrow{\bar{x}(w)} [w/y]P'} \begin{array}{l} y \neq x \\ w \notin \mathrm{fn}(\nu y.P) \end{array}$$

This exhibits several aspects of rules for name-passing that are not catered for in the GSOS format of Bloom *et al.* [1995]: (i) there is universal quantification over both processes (P,P') and names (w, x, y); (ii) there are binding operators, such as ν; (iii) there are explicit substitutions in the conclusion target; (iv) there are side conditions about the freshness and distinctness of name variables.

Given the role of nominal sets in our model theory, a suitable candidate for a logical framework is Pitts's nominal logic [2003]. We identify the following GSOS-like template for rules in nominal logic that we will focus on.

$$\frac{P_1 \xrightarrow{l_1} P_1' \quad P_2 \xrightarrow{l_2} P_2' \quad \ldots}{\mathsf{op}(\ldots) \xrightarrow{l} t} \tag{1.2.1}$$

Here, op is an operator from a special kind of binding signature: the parameters of this operator can be name or process variables, and may include binders. The target t of the conclusion is a complex expression built from name and process variables using the binding signature, augmented with the additional operator sub for substitution. We refer to structures of this shape as 'rule structures'.

Note that nominal logic is merely a first order theory, and therefore α-equivalence is *not* included in syntactic equality. Thus, when reasoning about rule structures at this metalogical level, we are concerned with terms of *raw* syntax, without α-equivalence. It is only possible to reason up-to α-equivalence once one is reasoning within nominal logic, or working in a suitable model.

The astute reader will have noticed that the rules used to define the π-calculus include side conditions about freshness and distinctness of certain variables, and yet there is no provision for such side conditions in the shape of rule structures in (1.2.1). It is a little messy to reason about these side conditions in a general way, and so here we adopt a convention that makes these side conditions implicit: whenever two different name variables are written in the rule, there is an implicit side condition requiring that the name variables are distinct; there is also an implicit side condition requiring that the bound names of the conclusion label are fresh for the conclusion source. While these conventions are technically convenient, they make presentations of calculi slightly clumsy — for instance, the π-calculus rule for output must be rewritten as the following two rules.

$$\frac{}{\bar{c}d.P \xrightarrow{\bar{c}d} P} \qquad \frac{}{\bar{c}c.P \xrightarrow{\bar{c}c} P}$$

(The first rule here has an implicit side condition: $c \neq d$.) In general, though, this rewriting is rather mechanical.

Presupposing some GSOS-like conditions on a rule structure of the form (1.2.1) we can derive a family of functions of the form (roughly) $\{\Sigma(X \times BX) \to BTX\}$; here X ranges over nominal substitutions. (Since our model of behaviour involves *structured* coalgebras, some additional structure operators are required in the type of these functions, which, for brevity, are omitted in this overview.) This family of functions is almost of the form (1.1.1) required to recursively define a lifting of the monad of syntax to the category of behaviour. In general, though, the functions in this family are not equivariant — they do not preserve the renaming structure of nominal sets. Moreover, the family need not be natural. Thus an important question is:

> *What conditions must be placed on a rule structure to guarantee equivariance and naturality?*

Viewing the rule structures of the form (1.2.1) as nominal logic formulas, we can consider the least labelled transition system over process terms that satisfies the rule. We would like the induced transition system to reside in the model theory developed for name-passing behaviour, and moreover we are interested in the congruence of wide open bisimilarity. None of these properties hold for systems defined by rule structures in general. From this perspective, the above question can be reformulated as:

> *What conditions must be placed on a rule structure to guarantee a well-behaved transition system and congruence of wide open bisimilarity?*

To answer these questions we suggest conditions for well-behaved rule structures. These conditions include GSOS-like conditions about the appearance of process variables, as well as conditions about name variables. For instance, it is necessary to forbid the anomalous semantics for input described by the axiom

$$\frac{}{c(c).P \xrightarrow{cd} [d/c]P}$$

discussed earlier. We forbid this axiom by requiring that, in the conclusion source, names that appear bound cannot also appear free.

To prove that the two questions are answered, we prove that the induced family of maps is equivariant and natural.

Final coalgebraic bisimulations. To conclude this discussion, we mention an attitude to the theory of coalgebras which pervades this thesis. Researchers in the theory of coalgebras often make use of final coalgebras. For an endofunctor B on a category $\mathscr{C}$, the carrier of the final B-coalgebra is a kind of universal domain of the behaviour described by B. Often, however, the final coalgebra for an endofunctor does not exist. As a basic cardinality argument will show, this problem arises even for the basic endofunctor $\mathscr{P}(Lab \times (-))$ on **Set**, whose coalgebras are Lab-labelled transition systems. In other words: there is no universal labelled transition system. When the final coalgebra does not exist, I identify three possible ways to progress.

1. One can usually find a restricted form of the endofunctor for which the final coalgebra does exist. For the case of labelled transition systems, non-determinism can be bounded. Many authors (including Rutten [2000]) have considered the endofunctor $\mathscr{P}_{\mathrm{f}}(Lab \times (-))$ on **Set**; here $\mathscr{P}_{\mathrm{f}}$ is the covariant *finite* powerset endofunctor. Coalgebras for this endofunctor are finitely branching Lab-labelled transition systems, and the final coalgebra is the set of finitely branching synchronisation trees, quotiented by bisimilarity.

   Although such restrictions are often computationally natural, this approach might be considered distasteful because the notion of behaviour is restricted, not primarily to arrive at an improved notion of behaviour, but to activate some mathematical machinery.

2. One can add enough objects to the category so that the final coalgebra exists. For the case of labelled transition systems, one can work in the category of classes, instead of in the category of sets: the final coalgebra can then be found, and its carrier is a proper class. This is the approach taken in the non-well-founded set theory of Aczel [1988], and is advocated for more general categories by Adámek, Milius, and Velebil [2005]. This approach is an interesting one, and warrants further investigation for base categories other than **Set**.

3. One can reason about behaviour without mentioning final coalgebras. One can instead work with the universal properties of *final bisimulations*. For the case of labelled transition systems, this amounts to reasoning about bisimulations and bisimilarity, rather than about synchronisation trees.

The last approach is the one taken in this thesis. By working with bisimulations rather than final coalgebras, one is freer to work with more general notions of equivalence, such as final *structured* bisimulations, and, most generally, what we call *final lifting spans*.

In our development we establish basic results in this direction: final bisimulations are typically relations; right adjoints between categories of coalgebras preserve final bisimulations; final bisimulations exist in a very wide variety of situations. Final bisimulations can often be constructed in a manner related to the terminal sequence [studied by *e.g.* Worrell, 2005]; this construction is related to partition refinement techniques [of *e.g.* Paige and Tarjan, 1987]. We provide a termination proof for this construction, at an abstract level, using properties of finite presentability. Most importantly, the results of the GSOS format can be understood without requiring any conditions of the categories or constructions concerned: a lifting of a monad of syntax to a category of coalgebras defines a semantics for which the final bisimulation, if it exists, is a congruence.

1.3 Synopsis

This thesis is split into two parts. In the first part, we relate and develop notions of model for name-passing. In the second part, we analyse structural operational semantics for name-passing caluli.

Each part begins with a chapter dedicated to recalling and developing some abstract notions, and illustrating these notions with basic examples from CCS-like systems.

Part I. Operational models.

Chapter 2: Transition systems, coalgebras, and bisimulation. This chapter is concerned with miscellaneous general notions from the theory of coalgebras that will be useful throughout the thesis. We introduce coalgebras as an abstract form of transition system, and, through examples from CCS with value-passing, we explain how restrictions to the model of behaviour can be described by modifying the behaviour endofunctor. We introduce the general notions of structured coalgebra and structured bisimulation.

We introduce a convenient way of comparing categories of coalgebras, by working with morphisms between endofunctors, and we establish basic results about bisimulations.

Chapter 3: Transition systems and coalgebras for name-passing. In this chapter we compare and develop models for name-passing, where state spaces are considered as presheaves on the categories $\mathbf{I}$ and $\mathbf{F}$. We begin this chapter by recalling definitions and properties for the π-calculus. We then survey coalgebraic models and transition system models for name-passing. An explicit characterisation of the coalgebraic models is established by axiomatising a class of transition systems.

Chapter 4: Models for name-passing, refined. In this chapter we consider the implications of imposing a sheaf condition on the state spaces of the models of name-passing. We redevelop the theory of Chapter 3 in this context. Inspired by a result of Fiore [2001], we study a class of transition systems over presheaves on the category $\mathbf{B}$.

Chapter 5: Practicality. The final chapter of this part is concerned with model checking problems for the models introduced in Chapter 4. In the first half of this chapter, we establish a correspondence between the Schanuel topos and a category of named sets. In the second half, we investigate conditions implying the existence of final bisimulations, and study a procedure for constructing these final bisimulations.

Part II. Operational semantics.

Chapter 6: Rule induction and mathematical operational semantics. The first chapter in this second part recalls and introduces some developments in abstract syntax and the analysis of rule induction, providing a thorough account of the analysis of Turi and Plotkin [1997], taylored to our needs. We recall basic notions for the categorical treatment of abstract syntax, considering fundamental notions of monad morphism and free monad. A mathematical theory of structural operational semantics is introduced and made relevant to the case of structured coalgebras. We illustrate this theory by explaining the positive GSOS rule format.

Chapter 7: Nominal sets for syntax and behaviour. In this chapter we recall some aspects of the framework of nominal sets of Gabbay and Pitts [2001], and develop some notions for this thesis. Basic definitions and properties for nominal sets and nominal logic are summarised. A theory of *nominal substitutions* is introduced, and an equivalence is established between the category of nominal substitutions and the category of sheaves on $\mathbf{F}$ used in Chapter 4.

A notion of *nominal algebraic signature* is proposed, as a generalisation of the notion of algebraic signature to handle syntax with binding. We explain how functors between different models of syntax arise as morphisms between models of such signatures. Finally, we develop a nominal logic theory of nominal transition systems. A correspondence is established between nominal transition systems and a model of Chapter 4.

Chapter 8: Operational semantics for name-passing. In the final chapter of this part we introduce the rule format for name-passing. We introduce a notion of rule structure, and consider conditions on rule structures that guarantee good behaviour. We then explain how rule structures satisfying these conditions give rise to natural transformations, supplying recursion data for lifting a monad to the category of structured coalgebras. Thus the main theorem is established: for a name-passing system defined by a set of rules in our rule format, wide open bisimilarity is a congruence.

The thesis concludes, in Chapter 9, with a brief summary and a discussion of related ideas and developments.

1.3.1 Relation with previously published work of the author

To a large extent, the substance of this thesis derives from and builds on work presented in two articles written by the author with M. Fiore. We tend not to refer to these articles in the body of the thesis, because the connections are so pervasive and because the most developed form of the material is here, in this thesis. We now briefly summarise the connections between these articles and the work presented here.

Comparing operational models of name-passing process calculi [Fiore and Staton, 2006a]. This article forms the basis for Part I of this thesis. Sections 2 and 3 of that article correspond to Chapter 3 here, and Section 4 of that article corresponds to Chapter 4 here. The internal transition systems presented in Section 5 of that article are similar in spirit to the nominal transition systems of Section 7.5 here. In Section 5.1 of that article, we explored the relationships between categories of named-sets and the Schanuel topos, which is also done in Section 5.1 here.

A congruence rule format for name-passing process calculi from mathematical structural operational semantics [Fiore and Staton, 2006b]. This article forms the basis for Part II of this thesis. In Section 1 of that article we develop a general theory of mathematical operational semantics for structured coalgebras; this theory is developed in Chapters 2 and 6 of this thesis. Section 2 of that article is concerned with nominal sets for modelling abstract syntax with variable

binding and substitution, and thus involves the concepts covered here in Chapter 7. Section 3 of that article introduces models for name-passing behaviour in the setting of nominal sets. As such, that section summarises the developments of Chapters 3 and 4 of this thesis, following the discussion in Section 7.1.5 here. Section 4 of that article introduces the notion of rule structure. That section corresponds to Chapter 8 of this thesis.

Part I

Operational Models

Chapter 2

Transition Systems, Coalgebras, and Bisimulation

Labelled transition systems provide a model of the stepwise evolution of systems, from which bisimulation arises as a natural notion of behavioural equivalence. The theory of coalgebras provides a more abstract approach to modelling such systems, and bisimulation is a natural notion to study in this general context, too. The first purpose of this section is to recall and motivate these ideas. A second purpose, which is equally important, is to introduce some novel concepts and observations that will be important in this thesis.

We begin in Section 2.1 by recalling basic notions of coalgebras, with the specific example of the coalgebraic representation of labelled transition systems.

In Section 2.2, we consider the particular case of value-passing. Labelled transition systems provide a suitable model of this paradigm, but we argue that this model is too generous. We assert that the possibility of input should be independent of the value being input. To resolve this problem, we axiomatise those labelled transition systems that respect this property. A theme of the subsequent chapters will be the capturing of such properties within behaviour endofunctors. By way of introduction, then, we provide an endofunctor for value-passing behaviour.

Having motivated the importance of considering different behaviour endofunctors, we consider, in Section 2.3, morphisms between endofunctors. In this way we are able to compare behaviour endofunctors. Indeed, we consider a 2-category of morphisms between behaviour endofunctors. By using this 2-category it is straightforward to induce functors and also adjunctions between categories of coalgebras.

Following this abstract development, we take a different tack, and consider a more general notion of coalgebra. When working with models of systems, the state space often has structure that is important for defining the model and for reasoning about it, but this structure need not be respected by the evolving behaviour of the system. To describe such systems we introduce the notion of *structured coalgebra*, in Section 2.4.

We conclude this chapter in Section 2.5 with two fundamental theorems about bisimulation. The first result gives conditions under which bisimulation spans (*i.e.* intensional bisimulations) factor through bisimulation relations (that is, extensional relations). The second result explains when two different behaviour endofunctors give rise to the same notion of final bisimulation. Using this result, the techniques for comparing behaviour endofunctors introduced in Section 2.3 can be used to relate notions of bisimulation.

2.1 Transition systems and coalgebras for evolving systems

Transition systems provide a formal description of the stepwise atomic evolution of a system. It is often helpful to label the evolution steps by the actions that are observed. Let Lab be a set of labels,

and recall that a *labelled transition system* is a set X, thought of as a set of states, together with a ternary relation

$$\longrightarrow \subseteq X \times \mathrm{Lab} \times X$$

thought of as describing the transitions that are permitted. As usual, we will use infix notation for the transition relation. So $x \xrightarrow{l} y$ indicates that from state x the system can evolve to state y, in a manner described by the label l.

To give such a transition relation on the set X is to give a "next step" function

$$h : X \to \mathscr{P}(\mathrm{Lab} \times X) \quad , \tag{2.1.1}$$

writing $\mathscr{P}$ for powerset operator. For each state $x \in X$, we have a set $h(x)$ of label-result pairs to which a system in state x is capable of evolving.

We will use the symbol B_{lts} for the operator $\mathscr{P}(\mathrm{Lab} \times (-))$, which takes sets to sets. Structures of the form (2.1.1) can then be called *coalgebras* for the operator B_{lts}: they are maps $X \to B_{\mathrm{lts}}X$. Indeed, the operator B_{lts} can be given an action on functions between sets, by considering a canonical covariant action of $\mathscr{P}$ and the universal characterisation of products. Thus B_{lts} is an endofunctor on the category of sets.

It is convenient to allow the state space and the behaviour endofunctor to take a general form, and we are led to consider coalgebras for arbitrary endofunctors on arbitrary categories.

Definition 2.1.2. Consider an endofunctor B on a category $\mathscr{C}$. A *B-coalgebra* is an object $X \in \mathscr{C}$ equipped with a morphism $X \to BX$.

A *B-coalgebra homomorphism* between B-coalgebras, (X,h) and (Y,k), is a morphism $f : X \to Y$ in $\mathscr{C}$ that makes the following diagram commute.

$$\begin{array}{ccc} X & \xrightarrow{f} & Y \\ {\scriptstyle h}\downarrow & & \downarrow{\scriptstyle k} \\ BX & \xrightarrow[Bf]{} & BY \end{array}$$

B-coalgebra homomorphisms are composed according to the composition of the underlying morphisms in $\mathscr{C}$. Thus we have a category, B-Coalg, of B-coalgebras and homomorphisms between them. We will often use the forgetful functor B-Coalg $\to \mathscr{C}$ that maps a B-coalgebra, (X,h), to its underlying 'carrier' object, X.

Bisimulation. Central to the theory of coalgebras is the notion of behaviour-respecting relation, or *bisimulation*. At times, it will be helpful to view bisimulations as instances of the following general notion.

Definition 2.1.3. Consider a functor $V : \mathscr{B} \to \mathscr{C}$ between categories. A *V-lifting span* between objects X and Y of $\mathscr{B}$ is a span $(VX \xleftarrow{r_1} R \xrightarrow{r_2} VY)$ in $\mathscr{C}$ for which there exists a span in $\mathscr{B}$, $(X \xleftarrow{s_1} S \xrightarrow{s_2} Y)$, such that $(VS, Vs_1, Vs_2) = (R, r_1, r_2)$.

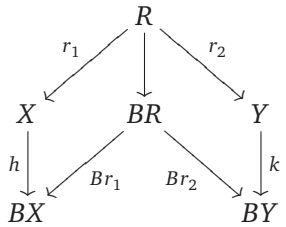
Now bisimulation can be defined as follows.

Definition 2.1.4. Consider an endofunctor B on a category $\mathscr{C}$, and consider the forgetful functor $V : B$-Coalg $\to \mathscr{C}$. Let (X,h) and (Y,k) be B-coalgebras. A *B-bisimulation* between (X,h) and (Y,k) is a V-lifting span between (X,h) and (Y,k).

Thus a B-bisimulation between B-coalgebras (X,h) and (Y,k) is a span $(X \xleftarrow{r_1} R \xrightarrow{r_2} Y)$ in $\mathscr{C}$ for which there exists a coalgebra structure $r : R \to BR$ making a span

$$(X,h) \xleftarrow{r_1} (R,r) \xrightarrow{r_2} (Y,k)$$

in the category of B-coalgebras. To say that $[(X,h) \xleftarrow{r_1} (R,r) \xrightarrow{r_2} (Y,k)]$ is a span of coalgebras is to say that the following diagram commutes.



Consider objects X and Y of a category $\mathscr{C}$. We say that a span between these objects, $(X \xleftarrow{r_1} R \xrightarrow{r_2} Y)$, is a *relation* if it is jointly monic — that is, if, for any object $A \in \mathscr{C}$ and any morphisms $f, g : A \to R$ such that $r_1 f = r_1 g$ and $r_2 f = r_2 g$, we have $f = g$. When $\mathscr{C}$ has products, this amounts to requiring that the universal morphism $(r_1, r_2) : R \to X \times Y$ is monic.

In the process algebra literature, it is conventional to focus on bisimulations that are relations. Importantly we have a correspondence between B_{lts}-bisimulation relations and the traditional bisimulations of labelled transition systems.

By working without this restriction, though, we are working with *intensional bisimulations*: bisimulations in which the relationship may be witnessed — and indeed where there may be several witnesses for each related pair. *Extensional bisimulations*, by contrast, which are jointly monic, record only the existence of a relationship.

In this thesis, it will be convenient to work with intensional bisimulations in our analysis at the abstract level, since many of the bisimulations that will arise from categorical constructions are, by nature, intensional. We return to this issue in Theorem 2.5.5, where we prove that, under certain basic assumptions, every intensional bisimulation factors through a extensional one.

Further reading. Rutten [2000] provides a detailed exposition of the theory of coalgebras over **Set**. Sections I.3 and III.12 of Turi's thesis [1996] deal with some of the basics of the theory of coalgebras. For category theory, the standard text is that by Mac Lane [1998].

2.2 Transition systems and coalgebras for value-passing systems

In this section we shall be concerned with systems where the observable actions correspond to the communication of values along named channels. We fix a set $\mathbb{C}$ of channels, and a set $\mathbb{V}$ of values. The values in $\mathbb{V}$ are to be thought of as basic data, and we will not expect these values to have any particular structure. A simple class of models for value-passing involves taking labelled transition relations or coalgebras as above, with label set

$$\mathrm{Lab}_{\mathbb{V}} \;=\; \mathbb{C} \times \mathbb{V} \;+\; \mathbb{C} \times \mathbb{V} \;+\; 1.$$

Here, the '+' operator denotes the coproduct of sets, taken to be disjoint union. We write '1' to denote a chosen initial set, that is, a set with one element. We consider the components of this sum as labels describing *input* actions (written $c?v$), *output* actions (written $c!v$) and *silent* actions (written τ).

Of course, it remains the case that to give a transition system over these labels is to give a coalgebra as in (2.1.1), of the form

$$X \to \mathscr{P}(\mathrm{Lab}_{\mathbb{V}} \times X) \quad .$$

$\mathbb{V}$-1 If one value can be input then so can any other:

$$x \xrightarrow{c?v} x' \implies \forall w \in \mathbb{V}.\ \exists x'' \in X.\ x \xrightarrow{c?w} x''$$

Figure 2.1: Axiom $\mathbb{V}$-1 on a value passing transition relation

The situation is made clearer, though, by pushing the structure from the labels into the behaviour endofunctor. Using the isomorphisms

$$\mathscr{P}(A+B) \cong \mathscr{P}(A) \times \mathscr{P}(B) \tag{2.2.1a}$$
$$\mathscr{P}(A \times B) \cong [A \Rightarrow \mathscr{P}B] \tag{2.2.1b}$$

(writing $[A \Rightarrow \mathscr{P}B]$ is the set of functions $A \to \mathscr{P}B$) we can consider coalgebras as in (2.1.1), assuming labels $\mathrm{Lab}_{\mathbb{V}}$, as coalgebras

$$\begin{aligned} X \to \quad & [\mathbb{C} \Rightarrow [\mathbb{V} \Rightarrow \mathscr{P}(X)]] \\ & \times [\mathbb{C} \times \mathbb{V} \Rightarrow \mathscr{P}(X)] \\ & \times [1 \Rightarrow \mathscr{P}(X)]. \end{aligned} \tag{2.2.2}$$

Such a coalgebra can be read as follows. The behaviour of each state $x \in X$ is described by a triple of functions

$$(i : \mathbb{C} \to [\mathbb{V} \Rightarrow \mathscr{P}(X)],\ o : \mathbb{C} \times \mathbb{V} \to \mathscr{P}(X),\ t : 1 \to \mathscr{P}(X)). \tag{2.2.3}$$

The function i assigns to each channel $c \in \mathbb{C}$ a function, which in turn assigns to each value $v \in \mathbb{V}$ the set of possible result states following the input of v on c; the function o assigns to each channel-value pair (c, v) the set of all possible result states following the output of v on channel c; the function t returns the set of possible result states following the silent action.

This notion of model for value-passing is perhaps too liberal, as the following example illustrates. Pick a channel $c \in \mathbb{C}$, and two distinct values, $v, w \in \mathbb{V}$. We consider a singleton state space, $\{*\}$, together with the transition relation

$$\longrightarrow = \{(*, c?v, *)\} \quad .$$

In state $*$, the system can perform action $c?v$ but cannot perform $c?w$. This is unnatural: the system is choosing which values can be input before the input event has taken place. In other words, the state is able to selectively refuse values without knowing what they are. Thus we are led to impose the uniformity Axiom $\mathbb{V}$-1 on transition relations, shown in Figure 2.2.

It is clear that we could impose a similar condition on the coalgebras that we consider. This, however, would not be in the spirit of the theory of coalgebras! Instead, we will refine our behaviour endofunctor to more accurately describe the nature of value-passing.

Recall from (2.2.3) that the behaviour of a state is described by a triple of functions (i, o, t). Axiom $\mathbb{V}$-1 can be thought of as having the following effect on component i: if, for any $c \in \mathbb{C}$, the function $i(c) : \mathbb{V} \to \mathscr{P}(X)$ *ever* returns the empty set then it *always* returns the empty set. This can be enforced in the behaviour functor as follows.

Let $\mathscr{P}_{\mathrm{ne}}$ be the covariant non-empty powerset operator on **Set**; so $\mathscr{P} \cong 1 + \mathscr{P}_{\mathrm{ne}}$. Thus each $i(c)$ is equivalent to a function of type $\mathbb{V} \to (1 + \mathscr{P}_{\mathrm{ne}}(X))$. We will refine this to the type $1 + (\mathbb{V} \to \mathscr{P}_{\mathrm{ne}}(X))$, indicating that the decision about whether or not to input is made before the value is seen. So: let $B_{\mathbb{V}}$: **Set** $\to$ **Set** be the following endofunctor

$$\begin{aligned} B_{\mathbb{V}} = \quad & [\mathbb{C} \Rightarrow (1 + [\mathbb{V} \Rightarrow \mathscr{P}_{\mathrm{ne}}(-)])] \\ & \times [\mathbb{C} \times \mathbb{V} \Rightarrow (1 + \mathscr{P}_{\mathrm{ne}}(-))] \\ & \times [1 \Rightarrow (1 + (\mathscr{P}_{\mathrm{ne}}(-)))] \quad . \end{aligned} \tag{2.2.4}$$

This endofunctor for value-passing was proposed by Fiore and Turi [2001, eqn. 23]. We have the following theorem.

Theorem 2.2.5. *There is a bijective correspondence between labelled transition systems with labels in* $\mathrm{Lab}_{\mathbb{V}}$ *that satisfy Axiom* $\mathbb{V}$*-1, and* $B_{\mathbb{V}}$*-coalgebras.* □

Importantly, coalgebraic $B_{\mathbb{V}}$-bisimulation corresponds to bisimulation for these labelled transition systems.

2.3 Morphisms between endofunctors

It is often useful to relate different notions of behaviour. To this end, we describe how the construction of coalgebras for an endofunctor can be given a functorial action.

We will consider the following 2-category coEndo of endofunctors. The name coEndo is used because this category is suited to working with *co*algebras. In Section 6.1.1 we will introduce an analogous category for working with algebras.

- Objects of coEndo are pairs $(\mathscr{C}, B)$ of a category $\mathscr{C}$ and an endofunctor B on $\mathscr{C}$.
- A morphism from $(\mathscr{C}, B)$ to $(\mathscr{C}', B')$ is given by a functor $F : \mathscr{C} \to \mathscr{C}'$ together with a natural transformation $\alpha : FB \to B'F$.
- The identity morphism is given by the identity functor together with the identity natural transformation. Composition of two morphisms

  $$(\mathscr{C}, B) \xrightarrow{(F,\alpha)} (\mathscr{C}', B') \xrightarrow{(F',\alpha')} (\mathscr{C}'', B'')$$

  is given by the composite functor $(F' \circ F)$ together with the natural transformation

  $$F'FB \xrightarrow{F'\alpha} F'B'F \xrightarrow{\alpha' F} B''F'F \quad .$$

- Let (F, α) and (G, β) be two morphisms from $(\mathscr{C}, B)$ to $(\mathscr{C}', B')$. A 2-cell from (F, α) to (G, β) is given by a natural transformation $\gamma : F \to G$ such that the following diagram commutes.

  $$\begin{array}{ccc} FB & \xrightarrow{\alpha} & B'F \\ {\scriptstyle \gamma B}\downarrow & & \downarrow{\scriptstyle B'\gamma} \\ GB & \xrightarrow[\beta]{} & B'G \end{array}$$

  Composition and identities of 2-cells are as in the category **CAT** of categories.

2.3.1 From endofunctors to coalgebras

There is a forgetful 2-functor coEndo → **CAT**, sending an endofunctor $(\mathscr{C}, B)$ to the underlying category $\mathscr{C}$. Another 2-functor coEndo → **CAT** sends an endofunctor $(\mathscr{C}, B)$ to the category B-Coalg of B-coalgebras. A morphism of endofunctors $(F, \alpha) : (\mathscr{C}, B) \to (\mathscr{C}', B')$ induces a functor

$$(F, \alpha)\text{-Coalg} : B\text{-Coalg} \to B'\text{-Coalg}$$

sending a B-coalgebra $(X, h : X \to BX)$ to the B'-coalgebra

$$\left(FX, FX \xrightarrow{Fh} F(BX) \xrightarrow{\alpha X} B'(FX)\right) \quad .$$

A 2-cell $\gamma : (F,\alpha) \to (G,\beta)$ between morphisms of endofunctors induces a natural transformation γ-Coalg : (F,α)-Coalg $\to (G,\beta)$-Coalg given as follows. For any B-coalgebra (X,h), the morphism $\gamma_X : FX \to GX$ describes a B-coalgebra homomorphism

$$(\gamma\text{-Coalg})_{(X,h)} : \big((F,\alpha)\text{-Coalg}\big)(X,h) \longrightarrow \big((G,\beta)\text{-Coalg}\big)(X,h)$$

since the following diagram commutes.

$$\begin{array}{ccccc} FX & \xrightarrow{Fh} & FBX & \xrightarrow{\alpha X} & BFX \\ {\scriptstyle \gamma X}\downarrow & (1) & \downarrow{\scriptstyle \gamma BX} & (2) & \downarrow{\scriptstyle B\gamma X} \\ GX & \xrightarrow[Gh]{} & GBX & \xrightarrow[\beta X]{} & BGX \end{array}$$

Using: 1: nat. of γ; 2: since γ is a 2-cell in coEndo.

This construction has been described by Lenisa, Power, and Watanabe [2000, Sec. 2.3], who write Endo*(**CAT**) for coEndo. Like them, our 2-category coEndo is inspired by the 2-category of comonads **Mnd**(**CAT**$_*$)$_*$ introduced by Street [1972].

The reader may be interested to note that the 2-functor $(-)$-Coalg : coEndo $\to$ **CAT** is representable as the hom-functor coEndo$((1,\mathsf{id}_1),-)$; here $(1,\mathsf{id}_1)$ denotes the identity endofunctor on a terminal category.

2.3.2 Adjunctions in coEndo

As coEndo is a 2-category, one can consider adjunctions in it. By making use of the 2-functor $(-)$-Coalg : coEndo $\to$ **CAT** introduced in the previous section, we see that adjunctions between endofunctors in coEndo induce adjunctions between the corresponding categories of coalgebras. Such adjunctions will be studied closely in Theorem 2.5.7.

The following characterisation result is fundamental. It follows immediately from a basic result of 2-dimensional monad theory [see *e.g.* Kelly, 1972, Sec. 1.3], since coEndo is the category of algebras and oplax morphisms for the identity 2-monad on **CAT**.

Proposition 2.3.1. *A morphism of endofunctors $(G,\beta) : (\mathscr{C}',B') \to (\mathscr{C},B)$ has a left adjoint in* coEndo *if and only if the underlying functor G has a left adjoint in* **CAT** *and β is an isomorphism.* □

2.4 Structured coalgebras

A recurring theme in this thesis is that of coalgebras whose carriers are equipped with extra structure. Examples abound throughout operational semantics, as the following examples illustrate.

Examples 2.4.1. 1. When an operational semantics for a language is given in terms of the language structure, it is important that the states of the resulting transition system are terms built out of the operators of the language. The transition system need not respect the operators precisely, though. For example, if the language has a unary operator op, then one would not necessarily expect a theorem

$$x \longrightarrow y \quad \text{if and only if} \quad \mathsf{op}(x) \longrightarrow \mathsf{op}(y) \quad .$$

2. In value-passing calculi, such as value-passing CCS [Milner, 1980], the transition system is typically defined over ground terms, *i.e.* terms with no free variables. But to make the definition, and for reasoning about the system, it is crucial to consider open terms.

3. The transition system for the π-calculus is defined over open terms, and that transition relation respects injective substitution of names. Arbitrary substitution of names need not be respected, though: one does not expect a theorem stating that for any substitution f,

$$x \xrightarrow{l} y \quad \text{if and only if} \quad [f]x \xrightarrow{[f]l} [f]y \quad .$$

It does make sense, though, to perform arbitrary substitution of names on the terms of the π-calculus, and indeed this is important when giving semantics.

Only this last example, of the π-calculus, will be studied in detail in this thesis. All the examples are catered for by the following general notion.

Definition 2.4.2. Consider a functor $U : \mathscr{D} \to \mathscr{C}$, and let B be an endofunctor on $\mathscr{C}$.

A *U-structured B-coalgebra* is an object X of $\mathscr{D}$ together with a morphism $UX \to BUX$ in $\mathscr{C}$ — *i.e.* a B-coalgebra structure for UX.

A *U-structured B-coalgebra homomorphism* between U-structured B-coalgebras, $(X, h : UX \to BUX)$ and $(Y, k : UY \to BUY)$, is a morphism $f : X \to Y$ in $\mathscr{D}$ making the following commute.

$$\begin{array}{ccc} UX & \xrightarrow{Uf} & UY \\ {\scriptstyle h}\downarrow & & \downarrow{\scriptstyle k} \\ BUX & \xrightarrow[BUf]{} & BUY \end{array}$$

Identity and composite homomorphisms are the identities and composites of the underlying morphisms in $\mathscr{D}$. U-structured B-coalgebras and homomorphisms between them form a category (U,B)-Coalg.

For such $U : \mathscr{D} \to \mathscr{C}$ and $B : \mathscr{C} \to \mathscr{C}$, we always have the following commuting diagram

$$\begin{array}{ccc} (U,B)\text{-Coalg} & \longrightarrow & B\text{-Coalg} \\ \downarrow & & \downarrow \\ \mathscr{D} & \xrightarrow[U]{} & \mathscr{C} \end{array}$$

where the unlabelled arrows denote the forgetful functors.

Structured coalgebras include coalgebras in the conventional sense: for any endofunctor B on a category $\mathscr{C}$, an B-coalgebra is the same thing as an $\mathsf{id}_{\mathscr{C}}$-structured B-coalgebra, writing $\mathsf{id}_{\mathscr{C}}$ for the identity functor on $\mathscr{C}$. Indeed, the categories $(\mathsf{id}_{\mathscr{C}}, B)$-Coalg and B-Coalg are isomorphic.

We illustrate Definition 2.4.2 by considering the scenarios from Example 2.4.1 above.

Examples 2.4.3.

1. To treat Example 2.4.1(1), we let T be a monad on a category $\mathscr{C}$, and B an endofunctor on $\mathscr{C}$. We let $U : T\text{-Alg} \to \mathscr{C}$ be the forgetful functor from the category of T-algebras. A U-structured B-coalgebra is a B-coalgebra whose carrier is equipped with a T-algebra structure. This might be called a bialgebra, following Turi and Plotkin [1997], although their notion is more symmetric, and involves a distributive law.

2. For the case of value-passing CCS we suppose that $\mathscr{D}$ is a category whose objects denote sets of open terms that are equipped with a notion of substitution. An suitable choice would be the category of monoid actions proposed by Fiore and Turi [2001, Sec. 4]. There is a functor $|\_|_0 : \mathscr{D} \to \mathbf{Set}$ mapping a set including open terms to the set of only ground terms. A $|\_|_0$-structured $B_{\mathbb{V}}$-coalgebra is a transition system over ground terms, in which the sets of states also include the open terms, and descriptions of substitution.

3. To model the π-calculus, we suppose that $\mathscr{D}$ is a category whose objects denote sets of states that are equipped with a notion of arbitrary substitution, and that $\mathscr{C}$ is a category whose objects denote sets of states that are equipped only with a notion of injective substitution. Arbitrary substitution subsumes injective substitution, so there is a forgetful functor $U : \mathscr{D} \to \mathscr{C}$. Choosing a suitable behaviour endofunctor B on $\mathscr{C}$, we have that a U-structured B-coalgebra is a transition system, where transitions are invariant under injective substitution, but where the sets of states also admit arbitrary substitution.

It is also interesting to consider some more general situations.

4. Suppose that a functor $U : \mathscr{D} \to \mathscr{C}$ has a right adjoint, say $G : \mathscr{C} \to \mathscr{D}$. For any endofunctor B on $\mathscr{C}$, to give a morphism $UX \to BUX$ is to give a morphism $X \to GBUX$. In this way, the category of U-structured B-coalgebras is isomorphic to the category of (GBU)-coalgebras.

5. Consider a full and faithful functor $i : \mathscr{C}' \to \mathscr{C}$ together with endofunctors B, B' on $\mathscr{C}$, $\mathscr{C}'$ respectively, and together with a natural isomorphism $\beta : iB' \xrightarrow{\sim} Bi$. (Such a situation arises, for instance, from a full reflection in coEndo.)

   The category (i,B)-Coalg is isomorphic to the category B'-Coalg. To see this, consider the following chain of natural correspondences.

$$\frac{\dfrac{iX \to BiX}{iX \to iB'X}\ (\text{using } \beta)}{X \to B'X}\ (i \text{ is f\&f})$$

Structured bisimulation. Since every U-structured B-coalgebra is also a B-coalgebra, the usual notion of coalgebraic bisimulation remains relevant here. It is also interesting, though, to consider the following notion of bisimulation, that involves the additional structure.

Definition 2.4.4. Consider a functor $U : \mathscr{D} \to \mathscr{C}$, and let B be an endofunctor on $\mathscr{C}$. Let V be the forgetful functor (U,B)-Coalg $\to \mathscr{D}$. A *U-structured B-bisimulation* between U-structured B-coalgebras, (X,h) and (Y,k), is a V-lifting span between (X,h) and (Y,k).

Thus a U-structured B-bisimulation between U-structured B-coalgebras (X,h) and (Y,k) is a span $(X \xleftarrow{r_1} R \xrightarrow{r_2} Y)$ in $\mathscr{D}$ for which there exists a coalgebra structure $r : UR \to BUR$ (*i.e.* a morphism in $\mathscr{C}$) making a span $[(X,h) \xleftarrow{r_1} (R,r) \xrightarrow{r_2} (Y,k)]$ in the category of U-structured B-coalgebras.

In other words, a U-structured B-bisimulation is a B-bisimulation that is in the image of the functor $U : \mathscr{D} \to \mathscr{C}$. Indeed, if $\mathscr{C} = \mathscr{D}$ and U is the identity functor, then U-structured B-bisimulation is the same thing as B-bisimulation.

We illustrate this notion of bisimulation in terms of the previous examples.

Examples 2.4.5. 1. First, we return to the case when we have a monad T on a category $\mathscr{C}$, and also an endofunctor B on $\mathscr{C}$. Again, we let $U : T\text{-Alg} \to \mathscr{C}$ be the forgetful functor from the category of T-algebras. Now, a U-structured B-coalgebra homomorphism is a morphism in $\mathscr{C}$ that is both a B-coalgebra homomorphism and a T-algebra homomorphism. Thus a U-structured B-bisimulation is a T-congruence that underlies a B-bisimulation. This kind of span might be called a *bicongruence*, following Turi and Plotkin [1997].

2. For the case of value-passing CCS, we return to the situation involving $|\_|_0$-structured $B_{\mathbb{V}}$-coalgebras. The relation involved in a $|\_|_0$-structured $B_{\mathbb{V}}$-bisimulation relation is defined over all (potentially open) terms. Thus this structured notion provides a way of reasoning about the 'open extension' of bisimilarity, as considered by Milner [1980, Sec. 5.8].

3. For the proposed model of the π-calculus, a U-structured B-bisimulation is a B-bisimulation that is closed under arbitrary substitution. Thus U-structured B-bisimulations are a kind of *open bisimulation*, in the sense of Sangiorgi [1996].

4. Consider the case of a functor $U : \mathscr{D} \to \mathscr{C}$ that has a right adjoint, say $G : \mathscr{C} \to \mathscr{D}$, and let B be an endofunctor on $\mathscr{C}$. A U-structured B-bisimulation between two U-structured B-coalgebras is a (GBU)-bisimulation between the corresponding (GBU)-coalgebras.

5. We will return to the case of Example 2.4.3(5) following Theorem 2.5.7.

2.5 Notions of bisimulation

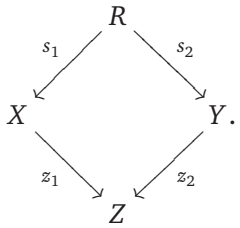
We now develop the theory of bisimulations by providing two fundamental results. The first result provides conditions under which bisimulation spans factor through bisimulation relations. For the second result, we introduce *final bisimulations* as an abstract formulation of the notion of bisimilarity. The result is concerned with how functors between categories of coalgebras preserve final bisimulations. We illustrate this by showing that certain kinds of morphism between endofunctors exhibit the endofunctors as describing the same notion of bisimulation.

In most of the work in the literature, it is assumed that, for the endofunctors under consideration, a final coalgebra exists. The unique final morphisms provide a kind of denotational semantics, which is necessarily 'fully abstract' with respect to bisimulation. The reader should note that neither of the endofunctors considered in this chapter have final coalgebras, essentially because they involve the unbounded powerset functor. Indeed, throughout this thesis, we take a purely operational approach: we will never assume the existence of final coalgebras. With this attitude, the notion of bisimulation, as opposed to the notion of denotational coincidence, takes a prominent role.

2.5.1 Bisimulation relations

In Section 2.1 we defined relations as jointly-monic spans. The aim of this subsection is to provide (in Theorem 2.5.5) conditions under which all bisimulations factor through bisimulation relations.

Regular relations. Note that any span that arises as a pullback, as in the following diagram, is jointly monic.



(2.5.1)

This can be seen by modifying the familiar argument that equalisers are monic.

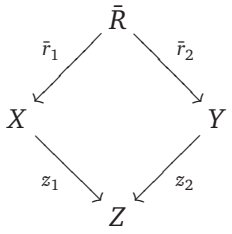
We now introduce a technical lemma, which establishes a result at the generality of lifting spans.

Lemma 2.5.2. Consider a functor $V : \mathscr{B} \to \mathscr{C}$ between categories $\mathscr{B}$ and $\mathscr{C}$. Suppose that every span in $\mathscr{B}$ completes to a square, and that for every cospan in $\mathscr{B}$, the pullback of the image under V exists in $\mathscr{C}$, and lies in the image of V.

Then every V-lifting span factors through a V-lifting relation.

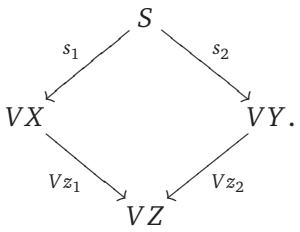
Proof. Consider objects X and Y of $\mathscr{B}$, and a V-lifting span (R, r_1, r_2) between them. By definition, we have a span $(X \xleftarrow{\bar{r}_1} \bar{R} \xrightarrow{\bar{r}_2} Y)$ in $\mathscr{B}$ such that $(R, r_1, r_2) = (V\bar{R}, V\bar{r}_1, V\bar{r}_2)$. We complete this span to a square in $\mathscr{B}$,

(2.5.3)



and then form the pullback in $\mathscr{C}$ of the image of the resulting cospan:

(2.5.4)



Since diagram 2.5.4 is a pullback, we know that S is jointly monic, and that R factors uniquely through S.

Moreover, by assumption, the relation (S, s_1, s_2) is in the image of V — that is, it is a V-lifting span. □

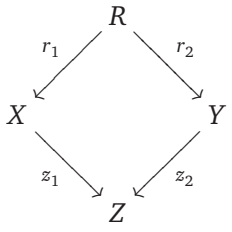
We now use this lemma with respect to structured bisimulations. Recall that a weak pullback (resp. pushout) is a pullback (resp. pushout), but for which the mediating maps need not be unique.

Theorem 2.5.5. *Let $U : \mathscr{D} \to \mathscr{C}$ be a functor, and let B be an endofunctor on $\mathscr{C}$. If $\mathscr{D}$ has pullbacks and weak pushouts, U preserves weak pullbacks and weak pushouts, and B preserves weak pullbacks, then every U-structured B-bisimulation factors through a U-structured B-bisimulation relation.*

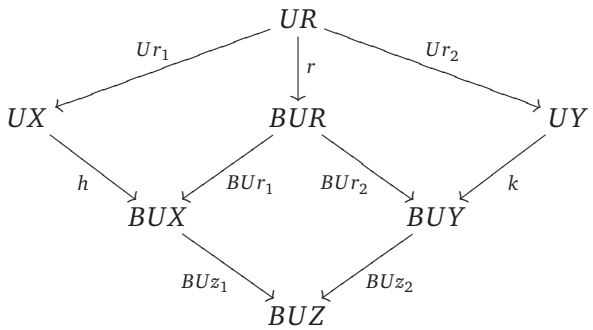
Proof. We show that the forgetful functor $V : (U,B)\text{-Coalg} \to \mathscr{D}$ satisfies the conditions of Lemma 2.5.2.

We first show that every span in (U,B)-Coalg completes to a square, as follows. Consider a span $[(X,h) \xleftarrow{r_1} (R,r) \xrightarrow{r_2} (Y,k)]$ in (U,B)-Coalg. We take a weak pushout in $\mathscr{D}$ of the underlying span in $\mathscr{D}$.

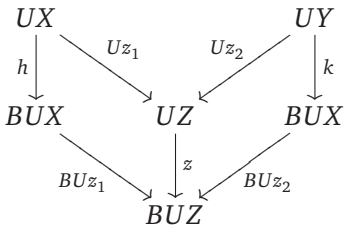
(2.5.6)



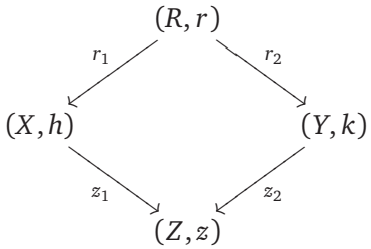
Observe that the following diagram commutes in $\mathscr{C}$.



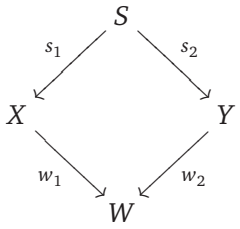
Thus we have a cocone over the span (UR, Ur_1, Ur_2). Now diagram 2.5.6 is a weak pushout in $\mathscr{D}$, so the image under U is a weak pushout in $\mathscr{C}$, because U preserves weak pushouts. Thus we have a morphism $z : UZ \to BUZ$ in $\mathscr{C}$ making the following diagram commute.



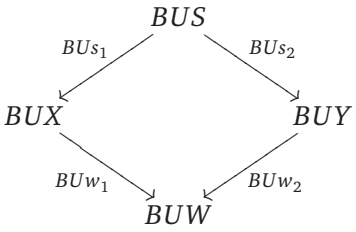
So we have the following commuting square in (U,B)-Coalg, as required.



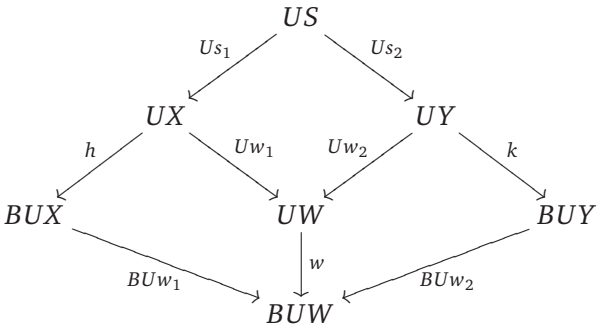
To show that the second condition of Lemma 2.5.2 is satisfied, we consider an arbitrary cospan $[(X,h) \xrightarrow{w_1} (W,w) \xleftarrow{w_2} (Y,k)]$ in (U,B)-Coalg. We construct the following pullback in $\mathscr{D}$.



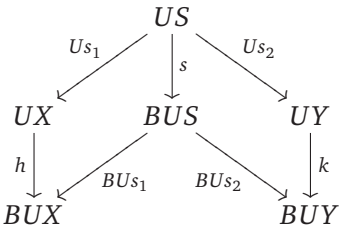
We must show that the span (S, s_1, s_2) lifts to a span of U-structured B-coalgebras. Since B and U preserve weak pullbacks, we have the following weak pullback in $\mathscr{C}$.



There is another cone for this cospan, since the following diagram commutes.



Thus we have a mediating morphism $s : US \to BUS$ making the following diagram commute in $\mathscr{C}$.



Thus we see that the pullback span in $\mathscr{D}$ lifts to (U,B)-Coalg.

So the conditions of Lemma 2.5.2 are satisfied, and so every U-structured B-bisimulation factors through a U-structured B-bisimulation relation. □

2.5.2 Relating notions of bisimulation

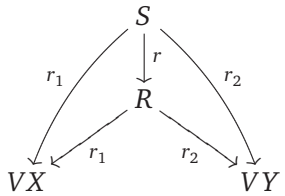
The purpose of this section is to investigate conditions under which functors between categories of coalgebras preserve final bisimulations. We begin with Theorem 2.5.7, which works at the generality of lifting spans. The conditions for this theorem are a little obscure and technical.

Following the proof we illustrate the theorem with some important examples. In Section 2.3 we introduced morphisms between behaviour endofunctors, that induce functors between categories of coalgebras. The two example scenarios that we consider involve a full reflection in the category coEndo, and the case where a behaviour endofunctor is a split subfunctor of another endofunctor on the same category. In both cases, we are able to say that the two endofunctors that are involved 'describe the same notion of bisimulation'.

Final lifting spans and final bisimulations. The greatest bisimulation plays an important role in operational semantics. Greatest bisimulations can be expressed in a universal way, as follows.

We have defined bisimulations and structured bisimulations in terms of the general notion of lifting span. We now introduce a notion of universal span.

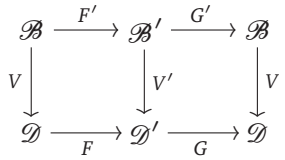
Consider a functor $V : \mathscr{B} \to \mathscr{C}$. We say that a V-lifting span (R, r_1, r_2) between objects X and Y in $\mathscr{B}$ is a *final V-lifting span* if every other V-lifting span (S, s_1, s_2) between X and Y factors uniquely through it, that is, if there is a unique map $r : S \to R$ in $\mathscr{C}$ making the following diagram commute in $\mathscr{C}$.



Note that, under the conditions of Theorem 2.5.5, every final U-structured B-bisimulation is a relation. Indeed, for the situation introduced in Section 2.1 involving the endofunctor B_{lts} on **Set**, the final B_{lts}-bisimulation is the usual bisimilarity relation on the corresponding transition systems.

We address the issue of existence of final bisimulations later, in Section 5.2. Now, we establish a theorem that compares final lifting spans over different categories.

Theorem 2.5.7. *Consider the following situation between functors and categories.*



Suppose that there are natural transformations $\varepsilon : F'G' \to 1_{\mathscr{B}'}$, $\eta : 1_{\mathscr{D}} \to GF$, such that the following diagram of natural transformations (a 'triangle identity') commutes.

$$\begin{array}{ccccc} VG' & \xrightarrow{\eta VG'} & GFVG' & \xrightarrow{=} & VG'F'G' \\ & \searrow^{\mathsf{id}} & & & \downarrow{\scriptstyle VG'\varepsilon} \\ & & & & VG' \end{array} \tag{2.5.8}$$

Let X and Y be objects of $\mathscr{B}'$. If (R, r_1, r_2) is a final V'-lifting span between X and Y then (GR, Gr_1, Gr_2) is a final V-lifting span between $G'X$ and $G'Y$.

Proof. [Warning: throughout this proof we elide the equalities $FV = V'F'$ and $GV' = VG'$ — hopefully the proof remains readable.]

Suppose that (R, r_1, r_2) is a final V'-lifting span between objects X and Y of $\mathscr{B}'$. It is immediate that (GR, Gr_1, Gr_2) is a V-lifting span between $G'X$ and $G'Y$; it remains for us to show that it is final.

Consider another V-lifting span (S, s_1, s_2) between $G'X$ and $G'Y$. We must have a span $(G'X \xleftarrow{\bar{s}_1} \bar{S} \xrightarrow{\bar{s}_2} G'Y)$ in $\mathscr{B}$ such that $(V\bar{S}, V\bar{s}_1, V\bar{s}_2) = (S, s_1, s_2)$. From this, we derive the following span in $\mathscr{B}'$.

$$X \xleftarrow{\varepsilon X} F'G'X \xleftarrow{F'\bar{s}_1} F'\bar{S} \xrightarrow{F'\bar{s}_2} F'G'Y \xrightarrow{\varepsilon Y} Y$$

Applying V', we see that the following span in $\mathscr{D}'$ is a V'-lifting span.

$$V'X \xleftarrow{V'\varepsilon X} FVG'X \xleftarrow{Fs_1} FS \xrightarrow{Fs_2} FVG'Y \xrightarrow{V'\varepsilon Y} V'Y$$

Thus, since (R, r_1, r_2) is a final V'-lifting span, we have a unique map $m : FS \to R$ in $\mathscr{D}'$, making the following diagram commute.

$$\begin{array}{ccccccc} V'X & \xleftarrow{V'\varepsilon X} & FVG'X & \xleftarrow{Fs_1} & FS & \xrightarrow{Fs_2} & FVG'Y & \xrightarrow{V'\varepsilon Y} & V'Y \\ & {\scriptstyle r_1}\searrow & & & \downarrow{\scriptstyle m} & & & \swarrow{\scriptstyle r_2} & \\ & & & & R & & & & \end{array} \tag{2.5.9}$$

Observe that the following diagram commutes in $\mathscr{D}$.

$$\begin{array}{ccccc} VG'X & \xleftarrow{s_1} & S & \xrightarrow{s_2} & VG'Y \\ \downarrow{\scriptstyle \eta VG'X} & (1) & \downarrow{\scriptstyle \eta S} & (2) & \downarrow{\scriptstyle \eta VG'Y} \\ GFVG'X & \xleftarrow{GFs_1} & GFS & \xrightarrow{GFs_2} & GFVG'Y \\ \downarrow{\scriptstyle GV'\varepsilon X} & (3) & \downarrow{\scriptstyle Gm} & (4) & \downarrow{\scriptstyle GV'\varepsilon Y} \\ GV'X & \xleftarrow[Gr_1]{} & GR & \xrightarrow[Gr_2]{} & GV'Y \end{array}$$

Using: 1,2: nat. of η; 3,4: dgm. 2.5.9.

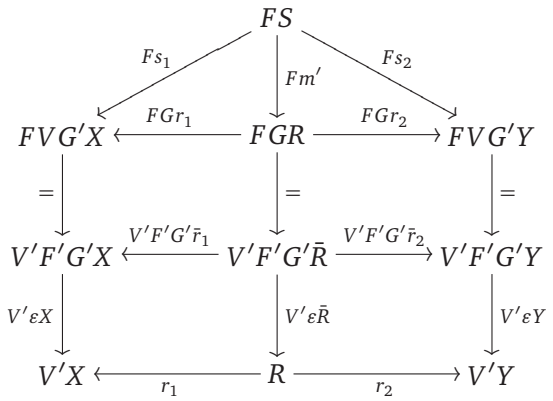
It follows, from diagram 2.5.8, that (S, s_1, s_2) factors, via $(Gm \circ \eta S)$, through (GR, Gr_1, Gr_2).

It remains for us to show that this factorisation is unique. Suppose that there is a morphism $m' : S \to GR$ such that the following diagram commutes.

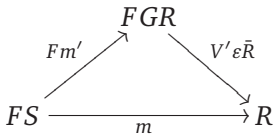
$$\begin{array}{ccccc} & & S & & \\ & {\scriptstyle s_1}\swarrow & \downarrow{\scriptstyle m'} & \searrow{\scriptstyle s_2} & \\ VG'X & \xleftarrow[Gr_1]{} & GR & \xrightarrow[Gr_2]{} & VG'Y \end{array}$$

We must show that $m' = Gm \circ \eta S$.

First, note that since R is a V'-lifting span between X and Y, the span (R, r_1, r_2) in $\mathscr{D}'$ lifts to a span $(X \xleftarrow{\bar{r}_1} \bar{R} \xrightarrow{\bar{r}_2} Y)$ in $\mathscr{B}'$ with $(V'\bar{R}, V'\bar{r}_1, V'\bar{r}_2) = (R, r_1, r_2)$. Next, observe that the following diagram commutes, in $\mathscr{D}'$.

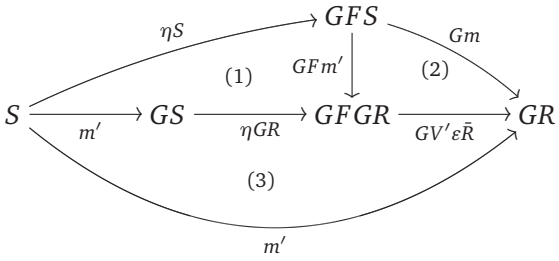


But $m : FS \to R$ was introduced as the unique mediator, in diagram 2.5.9. Thus we know that the following triangle commutes in $\mathscr{D}'$.



(2.5.10)

We conclude by putting all this together, to see that the following diagram commutes in $\mathscr{D}$.



Using: 1: nat. of η; 2: dgm. 2.5.10; 3: dgm. 2.5.8.

Thus $m' = Gm \circ \eta S$; one might say that $m' : S \to GR$ is a right adjunct of $m : FS \to R$. So the factorisation is unique. □

The situation considered in Theorem 2.5.7 is curious. The functors F', G' are almost required to be adjoint, but not quite: firstly, the 'unit' is associated with the underlying functors F, G, and secondly, only one of the triangle identities is considered.

Adjunctions. The case when F', G' and F, G are adjoint is particularly interesting. Specifically, we consider the situation induced by a full reflection

$$(\mathscr{C}', B') \underset{(F,\alpha)}{\overset{(G,\beta)}{\rightleftarrows}} (\mathscr{C}, B) \quad (\top)$$

in coEndo, as in Example 2.4.3(5). In this circumstance, Theorem 2.5.7 says that, for B'-coalgebras (that is, G-structured B-coalgebras), final B'-bisimulation (*i.e.* final G-structured B-bisimulation) is the same thing as final B-bisimulation.

Note that G' preserves limits, since it has a left adjoint. Thus if the final B'-coalgebra exists, and $\mathscr{C}, \mathscr{C}'$ have pullbacks, and B, B' preserve weak pullbacks, then the result in the previous paragraph is obvious — G' must send the final B'-coalgebra to the final B-coalgebra, and final bisimulations are determined by the final coalgebras. In this thesis, though, we will often work with endofunctors that do not have final coalgebras.

Split subfunctors. Because we only require one of the two triangle identities, Theorem 2.5.7 is also relevant in other situations. For instance, given two endofunctors B, B' on a category $\mathscr{C}$, if B' is a split subfunctor of B then Theorem 2.5.7 says that, for B'-coalgebras, final B'-bisimulation is the same thing as final B-bisimulation.

For a first example of this scenario, we return to the case of value-passing systems. The endofunctor $B_{\mathbb{V}}$ of (2.2.4) is a split subfunctor of B_{lts}. Hence we see that $B_{\mathbb{V}}$-bisimulation is the same thing as bisimulation for the induced labelled transition systems.

Chapter 3

Transition Systems and Coalgebras for Name-Passing

This chapter is concerned with models of name-passing process calculi. We study models involving labelled transition systems, and models involving coalgebras.

In the theory of name-passing calculi, notions of 'free names' and of substitution play a central role. In this chapter, we treat these aspects from a model theoretic perspective by considering the 'state spaces' of the models as presheaves. Thus a state space is given by specifying, for each set C of names, a set $P(C)$ of states with free names contained in C. Moreover, for every appropriate function $f : C \to D$ between sets of names, a substitution function between sets of states $Pf : P(C) \to P(D)$ must be specified. Thus all states are associated with a name context, and substitution in states is described by presheaf action.

We begin the chapter, in Section 3.1, with an overview of the π-calculus, which is our principle example of a name-passing calculus. We emphasise some results for the π-calculus that will be considered in the abstract setting as axioms for transition systems for name-passing.

In Section 3.2, we consider coalgebraic models for name-passing, based on the models introduced by Fiore and Turi [2001]. To do this, it is important to consider some of the sophisticated type constructions that arise from working in a presheaf category. For instance, the function space of the presheaf category is convenient for describing input behaviour. The actions of the presheaves of states account for injective substitution, and the coalgebra structures are natural transformations between presheaves. As a result, the invariance of transitions under injective substitution is built into the model.

We consider labelled transition system models of name-passing in Section 3.3. The approach we take is loosely related to one of Cattani and Sewell [2004]. As with the coalgebraic model, the state spaces are specified by presheaves. The transition systems are built over elements of the presheaves. Thus we are concerned with transitions of the form

$$C \vdash p \xrightarrow{l} C' \vdash p'$$

to be read as "process p, in name context C, can perform the transition labelled by l to become process p' in name context C'". The name contexts grow during evolution to record that new names are learnt through communication.

To achieve a bijective correspondence between the coalgebraic model and the labelled transition system model, we axiomatise a class of labelled transition systems. The axioms that arise are abstract forms of properties of the π-calculus. For instance, we enforce that transitions must be invariant under injective substitution.

The models considered in Sections 3.2 and 3.3 only allow for injective substitution, and so the notions of early and ground bisimulation can be defined for the models. In Section 3.4 we consider models that incorporate arbitrary substitution. We endow the states with this additional structure.

Symbol	Meaning	Example of usage
$\xrightarrow[\pi]{}$	Early labelled transition system for the π-calculus.	Fig. 3.2
$\xrightarrow[\pi]{}$	Ground labelled transition system for the π-calculus.	Fig. 3.3
$\longrightarrow$	Arbitrary labelled transition system.	(3.1.5), (3.1.6).
$\longrightarrow$	Arbitrary indexed early labelled transition system.	Defn. 3.3.2, defn. 3.4.4
$\longrightarrow$	Arbitrary indexed ground labelled transition system.	Defn. 3.3.2, defn. 3.4.4
${}_{g}\!\longrightarrow$	**I**-indexed ground labelled transition system derived from the **I**-indexed early labelled transition system, $\longrightarrow$.	(3.3.21)
$\xrightarrow[P]{}$	Indexed early labelled transition system over presheaf P.	pp. 61–62
$\xrightarrow[P]{}$	Indexed ground labelled transition system over presheaf P.	pp. 71–62
$\longrightarrow_h$	**I**-indexed early labelled transition system derived from a B_{e}-coalgebra with structure h.	p. 62
$\longrightarrow_h$	**I**-indexed ground labelled transition system derived from a B_{g}-coalgebra with structure h.	p. 71
${}_{\mathrm{e}}\!\longrightarrow$	**F**-indexed early labelled transition system derived from the **F**-indexed ground labelled transition system, $\longrightarrow$.	(3.4.10)
${}_{\mathrm{eg}}\!\longrightarrow$	**F**-indexed early labelled transition system $\longrightarrow$ is first converted to an **F**-indexed ground labelled transition system, and then back to a **F**-indexed early labelled transition system.	Thm. 3.4.12

Figure 3.1: Some symbols for relations that are used in this chapter.

From the coalgebraic perspective this amounts to working with structured coalgebras. The natural notion of bisimulation that arises is a form of open bisimulation.

Once all substitutions are considered, it is possible to impose an additional axiom on transition systems, asserting that input behaviour is suitably uniform. Thus we arrive at the labelled transition system model proposed by Cattani and Sewell [2004]. In this way the transition systems of Cattani and Sewell are given a coalgebraic foundation.

A variety of symbols are used for transition relations in this chapter. A summary is provided in Figure 3.1.

3.1 A name-passing process calculus: the π-calculus

We begin this chapter with an overview of the π-calculus of Milner *et al.* [1992]. We introduce some basic notions such as the early and ground labelled transition systems (Sections 3.1.1 and 3.1.2 respectively), and some forms of bisimulation (Section 3.1.3).

Further discussion, and proofs, of the results mentioned in this section can be found in the book by Sangiorgi and Walker [2001], hereafter referred to as SW01.

Syntax. We will assume a class $\mathcal{N}$ of names, ranged over by $c, d, \ldots$ and z. The syntax of the π-calculus is given by the following grammar.

$$\begin{array}{rcll} p & ::= & \mathbf{0} & \text{deadlock} \\ & | & p \mid p & \text{parallel composition} \\ & | & p + p & \text{non-deterministic sum} \\ & | & c(z).p & \text{input prefix} \\ & | & \bar{c}d.p & \text{output prefix} \\ & | & \tau.p & \text{silent prefix} \\ & | & [c = d]p & \text{match prefix} \\ & | & [c \neq d]p & \text{mismatch prefix} \\ & | & \boldsymbol{\nu} z.p & \text{restriction} \end{array} \tag{3.1.1}$$

Syntax up-to-α-equivalence. In the expressions $\boldsymbol{\nu} z.p$ and $c(z).p$ the name z is binding in p. Thus we have a notion of α-equivalence, that is, equality up-to renaming of binders. Here, we will consider α-equivalent terms as equal.

In their discussion, Milner *et al.* [1992, Part II, Sec. 3.1] seem *not* to equate α-equivalent terms. This is a little confusing since it follows from their Defn. 6 that a variant of α-equivalence *is* contained in syntactic identity.

Other presentations [*e.g.* Milner *et al.*, 1993] work with more sophisticated equivalence classes of terms, involving structural congruences. Although the models that we consider in the first part of this thesis appear to support the equating of structurally congruent terms, we will not explicitly consider notions of structural congruence in this thesis.

Substitution. We write $[c/z]p$ for the term p with the name c substituted for z. We assume that the substitution operation respects the binders, and hence respects the α-equivalence classes of terms. Notions of α-equivalence and substitution will be treated formally within an abstract framework in Chapter 7. Alternatively, the reader will find an account in SW01 (Defns. 1.1.5 and 1.1.8). Our conventions differ from those of SW01 only in that we write substitution as a prefix (rather than postfix) operator.

Notation. For any π-calculus term p, we write $\mathsf{fn}(p)$ for the set of variables that occur free (*i.e.* not bound) in p. A precise explanation of the fn function is found in Defn. 1.1.2 of SW01, or alternatively in the supp function of Section 7.1.

Remark: finiteness. Many presentations of the π-calculus provide language features such as recursive definitions or replication, for describing processes with infinitary behaviour. No such constructs have been included in the grammar presented here (3.1.1), but we remark the models introduced in this thesis do work very well with infinitary behaviour.

3.1.1 Early semantics

We recall the early operational semantics of the π-calculus.

Labels. For the early semantics we consider four sorts of labels: input labels (written cd for any $c, d \in \mathcal{N}$); output labels ($\bar{c}d$ for any $c, d \in \mathcal{N}$); bound output labels ($\bar{c}(z)$ for any $c, z \in \mathcal{N}$); and silent labels (τ).

Silent $\dfrac{}{\tau.p \xrightarrow[\pi]{\tau} p}$	**Input** $\dfrac{}{c(z).p \xrightarrow[\pi]{cd} [d/z]p}$
Output $\dfrac{}{\bar{c}d.p \xrightarrow[\pi]{\bar{c}d} p}$	
Match $\dfrac{p \xrightarrow[\pi]{\ell} q}{[c=c]p \xrightarrow[\pi]{\ell} q}\ (c \notin \mathsf{bn}(\ell))$	**Mismatch** $\dfrac{p \xrightarrow[\pi]{\ell} q}{[c\neq d]p \xrightarrow[\pi]{\ell} q}\ (c \neq d,\ c \notin \mathsf{bn}(\ell) \not\ni d)$
Sum left choice $\dfrac{p \xrightarrow[\pi]{\ell} q}{p+p' \xrightarrow[\pi]{\ell} q}\ (\mathsf{bn}(\ell)\cap\mathsf{fn}(p')=\emptyset)$	**Parallel** on right $\dfrac{p' \xrightarrow[\pi]{\ell} q'}{p\,|\,p' \xrightarrow[\pi]{\ell} p\,|\,q'}\ (\mathsf{bn}(\ell)\cap\mathsf{fn}(p)=\emptyset)$
Communication output on left $\dfrac{p \xrightarrow[\pi]{\bar{c}d} q \qquad p' \xrightarrow[\pi]{cd} q'}{p\,|\,p' \xrightarrow[\pi]{\tau} q\,|\,q'}$	**Restriction** $\dfrac{p \xrightarrow[\pi]{\ell} q}{\boldsymbol{\nu} z.p \xrightarrow[\pi]{\ell} \boldsymbol{\nu} z.q}\ (z \notin \mathsf{n}(\ell))$
Scope closure output on left $\dfrac{p \xrightarrow[\pi]{\bar{c}(z)} q \qquad p' \xrightarrow[\pi]{cz} q'}{p\,|\,p' \xrightarrow[\pi]{\tau} \boldsymbol{\nu} z.\,(q\,|\,q')}\ (z \notin \mathsf{fn}(p'))$	**Scope opening** $\dfrac{p \xrightarrow[\pi]{\bar{c}z} q}{\boldsymbol{\nu} z.p \xrightarrow[\pi]{\bar{c}(z)} q}\ (z \neq c)$

Figure 3.2: The early semantics of the π-calculus. Symmetric versions of the rules for sum, parallel transitions, communication, and scope closure, are elided. [*C.f.* SW01, Table 1.5.]

Notation. For any label ℓ, we write $\mathsf{n}(\ell)$ for the set of all names that appear in the label. We write $\mathsf{bn}(\ell)$ for the names in the label that appear in binding position — so

$$\mathsf{bn}(\ell) = \begin{cases} \{z\} & \text{if } \exists c \in \mathcal{N}.\ \ell = \bar{c}(z) \\ \emptyset & \text{otherwise} \quad . \end{cases}$$

The early semantics of the π-calculus is the least labelled transition relation $\xrightarrow[\pi]{}$ over the set of terms that satisfies the rules in Figure 3.2. The presentation in Figure 3.2 is not quite standard; we return to this point in the discussion preceeding Prop. 3.1.7.

We now collect some results about the early semantics. These results are important in our axiomatisation of transition systems for name-passing in Section 3.3.

Proposition 3.1.2 (*c.f.* SW01, Lemma 1.4.1)**.**

1. *If* $p \xrightarrow[\pi]{cd} p'$ *then* $c \in \mathsf{fn}(p)$ *and* $\mathsf{fn}(p') \subseteq \mathsf{fn}(p) \cup \{d\}$.
2. *If* $p \xrightarrow[\pi]{\bar{c}d} p'$ *then* $c, d \in \mathsf{fn}(p)$ *and* $\mathsf{fn}(p') \subseteq \mathsf{fn}(p)$.
3. *If* $p \xrightarrow[\pi]{\bar{c}(z)} p'$ *then* $c \in \mathsf{fn}(p)$ *and* $\mathsf{fn}(p') \subseteq \mathsf{fn}(p) \cup \{z\}$.
4. *If* $p \xrightarrow[\pi]{\tau} p'$ *then and* $\mathsf{fn}(p') \subseteq \mathsf{fn}(p)$. □

Regarding input behaviour:

Proposition 3.1.3 (*c.f.* SW01, Lemma 1.4.4(i,ii))**.**

1. *If* $p \xrightarrow{cz}_{\pi} p'$ *and* $z \notin \mathsf{fn}(p)$ *then for any* $d \in \mathcal{N}$, $p \xrightarrow{cd}_{\pi} [d/z]p'$.
2. *If* $p \xrightarrow{cd}_{\pi} p'$ *and* $z \notin \mathsf{fn}(p)$ *then there is* p'' *such that* $p \xrightarrow{cz}_{\pi} p''$ *and* $[d/z]p'' = p'$. □

The following weaker property of input behaviour will be relevant in Sections 3.2 and 3.3. Unlike Prop. 3.1.3, this property does not make use of non-injective substitutions in π-calculus terms.

Corollary 3.1.4.

1. *If* $p \xrightarrow{cd}_{\pi} p'$ *then for all* d' *there is* p'' *such that* $p \xrightarrow{cd'}_{\pi} p''$.
2. *If* $p \xrightarrow{cz}_{\pi} p'$ *and* $z \notin \mathsf{fn}(p)$ *then for any other* $z' \notin \mathsf{fn}(p)$ *we have* $p \xrightarrow{cz'}_{\pi} [z'/z]p'$. □

Fresh versus bound outputs. In the π-calculus semantics suggested by Milner *et al.* [1992, Part II], care is taken to ensure that the following statement holds of an induced transition system:

$$\text{If } p \xrightarrow{\bar{c}(z)} p' \text{ and } z' \notin \mathsf{fn}(p') \text{ then } p \xrightarrow{\bar{c}(z')} [z'/z]p'. \tag{3.1.5}$$

Thus the 'binding' name on the label appears to be binding in the right hand side of a transition. For example, the following transition is allowed in the system of Milner *et al.* [1992, Part II].

$$\boldsymbol{\nu} z.\bar{c}z.\mathbf{0} \xrightarrow{\bar{c}(c)} \mathbf{0} \tag{3.1.6}$$

In subsequent work, authors have been less careful with regard to property 3.1.5, instead focusing attention on bound output transitions where the data is *fresh*, *i.e.* transitions of the form

$$p \xrightarrow{\bar{c}(z)} p' \quad \text{with } z \notin \mathsf{fn}(p) \quad .$$

After all, it is clear that (i) no reasonable properties or constructions make specific use of non-fresh bound outputs, and (ii) name generation is an important and natural notion.

Here, we take this one step further, by modifying the usual semantics so that all bound output data is fresh. This is sensible because (i) non-fresh bound output data is not relevant for any reasonable properties or constructions, and (ii) it is conceptually natural to think of bound output transitions as transitions describing the output of fresh names. Thus the following statement holds of the semantics presented in Figure 3.2. (We obtain this result by adding side conditions to the usual presentations of rules for sum, match and mismatch.)

Proposition 3.1.7 (*c.f.* SW01, Exercise 1.4.2)**.** *If* $p \xrightarrow{\bar{c}(z)}_{\pi} p'$ *then* $z \notin \mathsf{fn}(p)$. □

Hence, in particular, the transition in (3.1.6) is not allowed in the semantics presented here.

In view of Prop. 3.1.2(2), we could omit the parentheses on bound output labels; instead, an output can be considered to be bound if it is not in the free names of the transition source. Thus the early semantics of the π-calculus can be described more simply in terms of input, output and silent labels. This approach will be taken in the labelled transition system models that we consider later in this chapter. For the remainder of Section 3.1, though, we will continue to mark bound outputs explicitly so that connections with the more common presentation can be seen.

It remains for us to record a weaker form of property 3.1.5 that applies to our transition system. A version of Corollary 3.1.4(2) holds for bound output transitions:

Proposition 3.1.8 (*c.f.* SW01, Lemma 1.4.9)**.** *If* $p \xrightarrow{\bar{c}(z)}_{\pi} p'$ *then for any* $z' \notin \mathsf{fn}(p)$ *we have that* $p \xrightarrow{\bar{c}(z')}_{\pi} [z'/z]p'$. □

3.1.2 Ground semantics

The ground semantics of the π-calculus is an alternative semantics in which the data of input labels is considered to be binding.

Labels for ground transitions. The ground semantics uses the same output, bound output and silent labels as the early semantics. In place of input labels we have *bound input* labels, written $c(z)$, for any names c, z. For any ground label we let

$$\mathsf{bn}(\ell) = \begin{cases} \{z\} & \text{if } \exists c \in \mathcal{N}.\ \ell = c(z) \text{ or } \ell = \bar{c}(z) \\ \emptyset & \text{otherwise} \quad . \end{cases}$$

The intention is that a bound input label $c(z)$ describes the action of input on channel c, and the name z is a placeholder for the name that is actually received.

In Figure 3.3 we present rules defining the ground transition relation $\xrightarrow[\pi]{}$. (The ground transition relation is distinguished from the early one by the use of a 'harpoon' arrow.) Because the bound input data is only a placeholder, the substitution of the input data does not appear in the axiom for input transitions, as it did in the early semantics. Instead, this substitution is described in the rules for communication and scope closure.

Many authors, including Milner *et al.* [1992, Part II], refer to the semantics that we have introduced in Figure 3.3 as *late semantics*. As will be seen in Definition 3.1.10, the ground semantics as presented here is more closely related to ground bisimulation than to late bisimulation. The phrase 'late semantics' is perhaps best reserved for *abstraction/concretion* presentations of the π-calculus [see *e.g.* SW01, Sec. 4.3.1].

By contrast with most presentations of the ground semantics, we have designed the operational semantics so that all bound inputs have fresh data. This matches the behaviour of bound outputs, as discussed in the previous subsection. It is sensible modification because, as for bound outputs, no reasonable property or construction will make use of non-fresh bound input parameters.

The following proposition makes precise the connection between the early and ground semantics. It exhibits the ground semantics as a variation of the early semantics where only fresh input data is allowed.

Proposition 3.1.9 (*c.f.* SW01, Lemma 4.3.2)**.**

1. *If* $p \xrightarrow[\pi]{cz} q$ *and* $z \notin \mathsf{fn}(p)$ *then* $p \xrightarrow[\pi]{c(z)} q$.
2. *If* $p \xrightarrow[\pi]{c(z)} q$ *then for all* d, $p \xrightarrow[\pi]{cd} [d/z]q$.
3. $p \xrightarrow[\pi]{\bar{c}d} q$ *if and only if* $p \xrightarrow[\pi]{\bar{c}d} q$.
4. $p \xrightarrow[\pi]{\bar{c}(d)} q$ *if and only if* $p \xrightarrow[\pi]{\bar{c}(d)} q$.
5. $p \xrightarrow[\pi]{\tau} q$ *if and only if* $p \xrightarrow[\pi]{\tau} q$. □

3.1.3 Bisimulations: early, ground, late, and wide open

We introduce four kinds of bisimulation. We begin with early, ground, and late bisimulation. The first of these, early bisimulation, is arguably the most elegant notion. Ground bisimulation on its own is a rather useless notion, since, for instance, it is not a congruence for the parallel composition operator. A definition of late bisimulation is included here only for quick comparison with the other notions. Late bisimulation will not be studied in this thesis. Following these three basic kinds of bisimulation, we introduce wide open bisimulation, which is a more refined kind of bisimulation that gives rise to a congruence.

Silent $\dfrac{}{\tau.p \xrightarrow[\pi]{\tau} p}$	**Input** $(*)$ $\dfrac{}{c(z).p \xrightarrow[\pi]{c(z)} p}\ (c \neq z)$
Output $\dfrac{}{\bar{c}d.p \xrightarrow[\pi]{\bar{c}d} p}$	
Match $\dfrac{p \xrightarrow[\pi]{\ell} q}{[c=c]p \xrightarrow[\pi]{\ell} q}\ (c \notin \mathsf{bn}(\ell))$	**Mismatch** $\dfrac{p \xrightarrow[\pi]{\ell} q}{[c \neq d]p \xrightarrow[\pi]{\ell} q}\ (c \neq d,\ c \notin \mathsf{bn}(\ell) \not\ni d)$
Sum left choice $\dfrac{p \xrightarrow[\pi]{\ell} q}{p + p' \xrightarrow[\pi]{\ell} q}\ (\mathsf{bn}(\ell) \cap \mathsf{fn}(p') = \emptyset)$	**Parallel** on right $\dfrac{p' \xrightarrow[\pi]{\ell} q'}{p \mid p' \xrightarrow[\pi]{\ell} p \mid q'}\ (\mathsf{bn}(\ell) \cap \mathsf{fn}(p) = \emptyset)$
Communication output on left $(*)$ $\dfrac{p \xrightarrow[\pi]{\bar{c}d} q \qquad p' \xrightarrow[\pi]{c(z)} q'}{p \mid p' \xrightarrow[\pi]{\tau} q \mid [d/z]q'}$	**Restriction** $\dfrac{p \xrightarrow[\pi]{\ell} q}{\boldsymbol{\nu} z.p \xrightarrow[\pi]{\ell} \boldsymbol{\nu} z.q}\ (z \notin \mathsf{n}(\ell))$
Scope closure output on left $(*)$ $\dfrac{p \xrightarrow[\pi]{\bar{c}(z)} q \qquad p' \xrightarrow[\pi]{c(z)} q'}{p \mid p' \xrightarrow[\pi]{\tau} \boldsymbol{\nu} z.\,(q \mid q')}$	**Scope opening** $\dfrac{p \xrightarrow[\pi]{\bar{c}z} q}{\boldsymbol{\nu} z.p \xrightarrow[\pi]{\bar{c}(z)} q}\ (z \neq c)$

Figure 3.3: The ground semantics of the π-calculus. Symmetric versions of the rules for sum, parallel transitions, communication, and scope closure, are elided. The rules marked with $(*)$ are the only rules that are different from the presentation in Figure 3.2. [*C.f.* SW01, Table 4.1.]

Definition 3.1.10 (*c.f.* SW01, Defns. 2.2.1, 4.4.1, 4.5.2)**.** Let R be a binary relation on π-calculus terms. Let p, p', q be arbitrary π-calculus terms, and let ℓ be an arbitrary early label, and ℓ' an arbitrary ground label.

- R is an *early simulation* if whenever $p\,R\,q$ and $p \xrightarrow[\pi]{\ell} p'$ with $\mathsf{bn}(\ell) \cap \mathsf{fn}(p,q) = \emptyset$ we have q' with $q \xrightarrow[\pi]{\ell} q'$ and $p'\,R\,q'$.
- R is a *ground simulation* if whenever $p\,R\,q$ and $p \xrightarrow[\pi]{\ell'} p'$ with $\mathsf{bn}(\ell) \cap \mathsf{fn}(p,q) = \emptyset$ we have q' with $q \xrightarrow[\pi]{\ell'} q'$ and $p'\,R\,q'$.
- R is a *late simulation* if whenever $p\,R\,q$ and $p \xrightarrow[\pi]{\ell'} p'$ with $\mathsf{bn}(\ell) \cap \mathsf{fn}(p,q) = \emptyset$, then
  1. if $\ell' = c(z)$ then there exists q' such that $q \xrightarrow[\pi]{\ell'} q'$ and for all names d we have $[d/z]p'\ R\ [d/z]q'$, and
  2. if ℓ is not an input label then we have q' with $q \xrightarrow[\pi]{\ell'} q'$ and $p'\,R\,q'$.

A binary relation R on π-calculus terms is an *(early/ground/late) bisimulation* if both R and its inverse are (early/ground/late) simulations.

Two π-calculus terms p, p' are *(early/ground/late) bisimilar* (written $p \sim_{e/g/l} p'$) if they are related by some (early/ground/late) bisimulation.

The three notions of bisimilarity are all bisimulations:

Proposition 3.1.11 (*c.f.* SW01, Convention 2.1.6)**.** *The relations* $(\sim_e)$, $(\sim_g)$, *and* $(\sim_l)$ *are respectively early, ground, and late bisimulations.* □

Invariance under injective substitutions. Since injective substitutions do not join names they do not affect transitions.

Notation. For a function $f : A \to B$ and a subset $A' \subseteq A$, we let $f|_{A'} : A' \twoheadrightarrow f(A')$ be the surjection found by restricting the domain of f to the subset A' and restricting the codomain of f to its image on A'.

Proposition 3.1.12 (*c.f.* SW01, Lems. 1.4.8, 2.2.3; Ex. 4.4.2)**.** *For any endofunction f on names, any π-calculus terms p, p', q and any π-calculus early label ℓ and ground label ℓ':*

1. *If $p \xrightarrow[\pi]{\ell} p'$ and $f|_{\mathsf{fn}(p)\cup\mathsf{fn}(p')\cup\mathsf{fn}(\ell)}$ is bijective then $[f]p \xrightarrow[\pi]{[f]\ell} [f]q$.*
2. *If $p \sim_e q$ and $f|_{\mathsf{fn}(p)\cup\mathsf{fn}(q)}$ is bijective then $[f]p \sim_e [f]q$.*
3. *If $p \xrightarrow[\pi]{\ell'} p'$ and $f|_{\mathsf{fn}(p)\cup\mathsf{fn}(p')\cup\mathsf{fn}(\ell')}$ is bijective then $[f]p \xrightarrow[\pi]{[f]\ell'} [f]q$.*
4. *If $p \sim_g q$ and $f|_{\mathsf{fn}(p)\cup\mathsf{fn}(q)}$ is bijective then $[f]p \sim_g [f]q$.* □

(Similar results hold for late bisimilarity but do not concern us here.)

Invariance under *all* substitutions. Neither early, ground nor late bisimilarity is invariant under *all* substitutions. For an example, consider the π-calculus terms

$$p_1 = [c = d]\bar{c}c.\mathbf{0} \qquad p_2 = \mathbf{0} \quad .$$

Now p_1 and p_2 are related by all three forms of bisimilarity, while all three forms distinguish $[c/d]p_1$ from $[c/d]p_2$.

For a similar reason, neither early, ground nor late bisimilarity is a congruence for the π-calculus. To see this, consider the bisimilar terms p_1, p_2 as before, but now in the context $((c(d).[-]) \mid (\bar{c}c.\mathbf{0}))$. The following transitions are derivable in both the early and ground semantics.

$$(c(d).p_1) \mid (\bar{c}c.\mathbf{0}) \xrightarrow{\tau} [c/d]p_1 \qquad (c(d).p_2) \mid (\bar{c}c.\mathbf{0}) \xrightarrow{\tau} [c/d]p_2$$

So the context $((c(d).[-]) \mid (\bar{c}c.\mathbf{0}))$ distinguishes bisimilar terms.

Wide open bisimulation. In the introduction to this thesis we argued that a process equivalence that is not a congruence is of little practical use. For instance, if a process equivalence is not a congruence then one cannot reason in an equational way. We now introduce a notion of bisimilarity that is a congruence.

Definition 3.1.13 (*c.f.* Sangiorgi [1996, Defn. 3.6, Prop. 3.9])**.** A binary relation R on π-calculus terms is *wide open* if for all endofunctions f on names and all π-calculus terms p, p',

$$p\,R\,p' \implies [f]p\,R\,[f]p' \quad .$$

We introduce the term 'wide open' (suggested by Peter Sewell) because the term 'open bisimulation' is perhaps best saved for the less discerning notion proposed by Sangiorgi [1996], which is studied briefly in Section 9.3.5 of this thesis.

If only wide open relations are considered, the early, ground and late bisimilarities are identified. This result follows straightforwardly from Prop. 3.1.9.

Proposition 3.1.14. *Let R be a binary relation on π-calculus terms. If R is wide open then the following are equivalent.*

1. *R is an early bisimulation.*
2. *R is a ground bisimulation.*
3. *R is a late bisimulation.* □

(A related result will be established at the model theoretic level, in Theorem 3.4.9.)

We will say that a relation satisfying these three equivalent conditions is a *wide open bisimulation*. Two π-calculus terms p, p' are *wide open bisimilar* (written $p \sim_{\mathrm{wo}} p'$) if they are are related by some wide open bisimulation. Following Prop. 3.1.11, wide open bisimilarity is itself a wide open bisimulation. Indeed, wide open bisimilarity is the greatest wide open bisimulation. Note that it is *not* the substitution closure of early, ground or late bisimilarity. These latter relations have been called strong equivalences [Milner *et al.*, 1992, Defn. II.10], strong congruences [Parrow, 2001, Defn. 6.2], and full bisimilarities [Sangiorgi and Walker, 2001, Defn. 2.2.2], but will not concern us here.

The following important result will be seen to arise from the abstract framework presented in Chapter 8.

Proposition 3.1.15 (*c.f.* SW01, Exercise 4.6.1(2))**.** *Wide open bisimilarity is a congruence.* □

Comparing bisimilarities. The different forms of bisimilarity introduced so far are related as follows.

Proposition 3.1.16 (*c.f.* SW01, Lems. 4.4.4, 4.5.3)**.** $(\sim_{\mathrm{wo}}) \subsetneq (\sim_{\mathrm{l}}) \subsetneq (\sim_{\mathrm{e}}) \subsetneq (\sim_{\mathrm{g}})$. □

3.2 Coalgebras for name-passing

We now present the theory of name-passing from a coalgebraic point of view. An aim of this section is to introduce behaviour endofunctors for coalgebras for name-passing. The endofunctor that we use for early behaviour is essentially the one introduced by Fiore and Turi [2001, eq. (26)].

The definitions of bisimulations for the π-calculus (Definition 3.1.10) are coinductive, but are different from the usual notions of bisimulation on transition systems because of the requirement that binding names in the transition labels be fresh for the transition sources. To cast such bisimulations in an abstract light, we need the states of the system to be tagged with the names that they may use. The solution that we consider here is to work with coalgebras in presheaf categories; the indexing of the presheaf serves to tag the states with the names that are available.

We begin this section with a discussion of the presheaf category that is under consideration. In Section 3.2.1, we formulate some constructions in this presheaf category, and in Section 3.2.2 we use these constructions to specify endofunctors for both early and ground behaviour.

Presheaves. According to Prop. 3.1.12, both transtions and bisimilarity are stable under injective renaming. This will be taken as an important abstract characteristic of the systems that we consider. For this reason, notions of *free names of a state* and *injective substitution* are crucial. One way to embed these notions into a model is to index the states by the names that they may use. To this end we let $\mathbf{I}$ be the category with objects given by finite subsets of $\mathcal{N}$, the class of all names, and with morphisms injective functions between them. A covariant presheaf $P : \mathbf{I} \to \mathbf{Set}$ can be thought of as associating to each finite set of names C, a set $P(C)$ of states that may use these names; to each injection $\iota : C \rightarrowtail D$ is associated a function $P\iota : P(C) \to P(D)$ intended to model substitution in the states in $P(C)$ according to ι.

Notation. We write $[\iota]p$ for $P\iota(p)$, when the presheaf P is clear from the context. Whenever C is a subset of D, we have the inclusion morphism $C \to D$ in $\mathbf{I}$, which we denote $(C \hookrightarrow D)$.

Throughout this thesis, we write "$X \subseteq_{\mathrm{f}} Y$" to mean that X is a subset of Y, and moreover that X is finite.

For any $C \subseteq_{\mathrm{f}} \mathcal{N}$, and any $z, z' \notin C$, we have an injection $[z'/z] : C \cup \{z\} \rightarrowtail C \cup \{z'\}$ acting as identity on C, and mapping z to z'.

More generally, given an injection $\iota : C \rightarrowtail D$ in $\mathbf{I}$, and $z' \in (D - \mathrm{im}(\iota))$, and $z \notin C$, we let $[\iota, z'/z] : C \cup \{z\} \to D$ be the injection given as follows.

$$[\iota, z'/z](c) = \begin{cases} \iota(c) & \text{if } c \in C \\ z' & \text{if } c = z \end{cases}$$

A trivial but important fact is that for any finite set of names C, we can always find a name $z \in \mathcal{N}$ that is not in C.

Natural transformations between presheaves. A natural transformation α between presheaves $P, Q : \mathbf{I} \to \mathbf{Set}$ is a family of functions between sets of states

$$\{\alpha_C : P(C) \to Q(C)\}_{C \in \mathbf{I}}$$

that respects injective substitutions in the sense that for any injection $\iota : C \rightarrowtail D$ in $\mathbf{I}$ the following square commutes.

$$\begin{array}{ccc} P(C) & \xrightarrow{\alpha_C} & Q(C) \\ {\scriptstyle P\iota}\downarrow & & \downarrow{\scriptstyle Q\iota} \\ P(D) & \xrightarrow[\alpha_D]{} & Q(D) \end{array}$$

Example: presheaf for the π-calculus. The state space of the π-calculus is represented by a presheaf $P_\pi \in \mathbf{Set}^{\mathbf{I}}$ defined as follows. For $C \in \mathbf{I}$,

$$P_\pi(C) = \left\{ p \;\middle|\; p \text{ is a } \pi\text{-calculus term and } \mathrm{fn}(p) \subseteq C \right\} \tag{3.2.1a}$$

while for any $C, D \in \mathbf{I}$ and injection $\iota : C \rightarrowtail D$, and any term $p \in P_\pi(C)$:

$$P_\pi\iota(p) = [\iota]p \qquad \text{— } \textit{i.e.} \text{ the substitution } \iota \text{ applied to } p. \tag{3.2.1b}$$

Sets of elements of presheaves. It will be useful to consider the states at all the stages of a presheaf, to yield a set of all the states. For any small category $\mathbf{C}$ we have a faithful functor $\int : \mathbf{Set}^{\mathbf{C}} \to \mathbf{Set}$ mapping a presheaf $P \in \mathbf{Set}^{\mathbf{C}}$ to its set of *elements*, *viz.*

$$\int P = \sum_{C \in \mathbf{C}} P(C) \quad . \tag{3.2.2a}$$

We write $(C \vdash p)$ for an element $p \in P(C)$ injected into $\int P$. In case $\mathbf{C} = \mathbf{I}$, the element $(C \vdash p)$ denotes a state p in the context of names C. To each natural transformation $\alpha : P \to Q$ we associate a function of sets $(\int \alpha) : \int P \to \int Q$, given as follows.

$$(\int \alpha)(C \vdash p) = C \vdash (\alpha_C(p)) \tag{3.2.2b}$$

A category with a faithful functor to **Set** has been called *concrete*. From the point of view of presheaves as structured state spaces, the concretion functor $\int$ serves to 'forget' the substitution structure and to give a 'global' view of the state space.

For the presheaf of π-calculus terms, $P_\pi \in \mathbf{Set}^{\mathbf{I}}$ as defined above in (3.2.1), we have

$$\int P_\pi = \left\{ (C \vdash p) \;\middle|\; p \text{ is a } \pi\text{-calculus term and } \mathrm{fn}(p) \subseteq C \right\} \quad .$$

So $\int P_\pi$ is the set of π-calculus terms in name contexts.

3.2.1 Constructions in $\mathbf{Set}^{\mathbf{I}}$

By considering presheaves in $\mathbf{Set}^{\mathbf{I}}$ we have at our disposal a rich collection of types and type constructions. We now recall the constructions that Fiore and Turi [2001] found useful in their models of name-passing. (Fiore and Turi work with a skeleton $\mathbb{I}$ of **I**, by representing names by numbers. Here, by working with **I**, we treat names as names, but a disadvantage is that the 'plus' tensor product of $\mathbb{I}$ is unnatural in **I**, and so the descriptions of some of the constructions are more involved here.)

It is perhaps worth remarking that the main motivation behind all these constructions is their utility in modelling name-passing systems. Many of the constructions that we introduce here could be defined by universal properties, but instead we take a concrete approach because this will be more useful in the remainder of this chapter. A disadvantage of this pragmatism is that, when we move to categories other $\mathbf{Set}^{\mathbf{I}}$ in Chapter 4 there is some extra work involved in lifting the constructions to the different categories.

A different approach to take when modelling name-passing would be to simply reinterpret the constructions of Section 2.2 internally within the presheaf topos $\mathbf{Set}^{\mathbf{I}}$. We return to this idea in Section 9.3.2. At this stage, though, it is hard to motivate this approach from a pragmatic point of view. For now, then, the structure of $\mathbf{Set}^{\mathbf{I}}$ is used simply to capture the essence of invariance under injective substitutions. The constructions that we consider in this subsection are suggested more by the nature of name-passing than by mathematical elegance.

Products, coproducts. Products and coproducts are computed 'pointwise' in $\mathbf{Set}^{\mathbf{I}}$ (as are all limits and colimits in any functor category). That is, for any presheaves $P, Q \in \mathbf{Set}^{\mathbf{I}}$, and $C \in \mathbf{I}$,

$$(P \times Q)(C) = P(C) \times Q(C) \qquad \text{and} \qquad (P + Q)(C) = P(C) + Q(C) \quad .$$

The pointwise projections/injections and mediating morphisms are natural.

Type of names. Let the presheaf of names $N \in \mathbf{Set}^{\mathbf{I}}$ be given as follows. For any name context $C \subseteq_{\mathrm{f}} \mathcal{N}$,

$$N(C) = C$$

while for any injection $\iota : C \rightarrowtail D$ in **I**, and any $c \in C$,

$$N\iota(c) = \iota(c)$$

(This presheaf will play a role analogous to that played by the set $\mathbb{V}$ of values in Section 2.2.)

Function space. For each state space $P \in \mathbf{Set}^{\mathbf{I}}$, we have a space $[N \Rightarrow P]$ of 'functions' from N to P. For any $C \subseteq_{\mathrm{f}} \mathcal{N}$,

$$[N \Rightarrow P](C) = \left\{ \phi \in \prod_{d \in \mathcal{N}} P(C \cup \{d\}) \;\middle|\; \forall z, z' \notin C.\ P[z'/z](\phi(z)) = \phi(z') \right\} . \tag{3.2.3a}$$

Thus an element ϕ of the function space $[N \Rightarrow P]$ at name-context C is an assignment from each name $c \in C$ in the name-context to an element $\phi(c) \in P(C)$, together with a *uniform* assignment of each name $z \in (\mathcal{N} - C)$ *not* in the name-context to an element $\phi(z) \in P(C \cup \{z\})$. This captures some of the nature of the input behaviour described in Corollary 3.1.4, as will be seen in Section 3.3.

The functorial action of $[N \Rightarrow P]$ is given as follows. For any $D \subseteq_{\mathrm{f}} \mathcal{N}$, $\iota : C \to D$ in $\mathbf{I}$, and $\phi \in [N \Rightarrow P](C)$, and $z, z' \in \mathcal{N}$, we let

$$\left([N \Rightarrow P]\iota(\phi)\right)(z) = \begin{cases} P\iota(\phi(\iota^{-1}(z))) & \text{if } z \in \mathsf{im}(\iota) \\ P[\iota, z/z'](\phi(z')) & \text{if } z \notin \mathsf{im}(\iota) \text{ and } z' \notin C \end{cases} \tag{3.2.3b}$$

The definition in equation 3.2.3b is unambiguous because if $z \notin \mathsf{im}(\iota)$ and $z', z'' \notin C$ then $P[\iota, z/z''](\phi(z'')) = P[\iota, z/z'](\phi(z'))$.

If we consider inclusion maps and bijections seperately, we have the following identities. For any $C, D \subseteq_{\mathrm{f}} \mathcal{N}$ with $C \subseteq D$, and any $\phi \in [N \Rightarrow P](C)$ and $z \in \mathcal{N}$, we have

$$\left([N \Rightarrow P][C \hookrightarrow D](\phi)\right)(z) = P[C \cup \{z\} \hookrightarrow D \cup \{z\}](\phi(z)) \quad . \tag{3.2.3c}$$

For any $C, C' \subseteq_{\mathrm{f}} \mathcal{N}$ and any $z \in \mathcal{N}$, we have that for any bijection $\beta : C \cup \{z\} \xrightarrow{\sim} C'$ and any $\phi \in [N \Rightarrow P](C)$,

$$\left([N \Rightarrow P](\beta|_C)(\phi)\right)(\beta(z)) = P(\beta)(\phi(z)) \quad . \tag{3.2.3d}$$

For any other presheaf Q, and any natural transformation $\alpha : P \to Q$ in $\mathbf{Set}^{\mathbf{I}}$, we let $[N \Rightarrow \alpha]$ be the natural transformation $\alpha : [N \Rightarrow P] \to [N \Rightarrow Q]$ in $\mathbf{Set}^{\mathbf{I}}$ given as follows. For each $C \subseteq_{\mathrm{f}} \mathcal{N}$ and $\phi \in [N \Rightarrow P]$, and each $z \in \mathcal{N}$, let

$$[N \Rightarrow \alpha]_C(\phi)(z) = \alpha_C(\phi(z)) \quad . \tag{3.2.3e}$$

Name generation/abstraction. The operator δ of name generation in $\mathbf{Set}^{\mathbf{I}}$ is defined on a presheaf $P \in \mathbf{Set}^{\mathbf{I}}$ as follows. For $C \in \mathbf{I}$, we let

$$\delta P(C) = \left\{ \phi \in \prod_{z \in (\mathcal{N} - C)} P(C \cup \{z\}) \;\middle|\; \begin{array}{l} \forall z, z' \in (\mathcal{N} - C). \\ \quad \phi(z') = P[z'/z](\phi(z)) \end{array} \right\} . \tag{3.2.4a}$$

Thus an element ϕ of the presheaf δP at name-context C describes a uniform treatment of fresh names, *i.e.* names not in C. This captures some of the nature of the bound output behaviour described in Prop. 3.1.8, and also of the bound input behaviour used in the ground semantics.

The name generation operator can also be seen as a name *abstraction* operator, by considering its 'colimit' form. We have the following isomorphism, natural in $C \in \mathbf{I}$:

$$\delta P(C) \;\cong\; \sum_{z \in (\mathcal{N} - C)} P(C \cup \{z\}) \Big/ {\sim_{PC}}$$

where $\mathsf{inj}_z(p) \sim_{PC} \mathsf{inj}_{z'}(P[z'/z]p)$. Thus an element $[\mathsf{inj}_z(p)]_{\sim PC}$ provides an equivalence class of treatments of fresh names. This interpretation will be important in Chapter 7.

The functorial action of δP is as follows. For any name sets $C, D \subseteq_{\mathrm{f}} \mathcal{N}$, any injection $\iota : C \rightarrowtail D$, and any function $\phi \in \delta P(C)$, and names $z \in (\mathcal{N} - D)$, $z' \in (\mathcal{N} - C)$, we let

$$\big(\delta P\iota(\phi)\big)(z) = P[\iota, z/z'](\phi(z')) \quad . \tag{3.2.4b}$$

To each natural transformation $\alpha : P \to Q$ between presheaves in $\mathbf{Set}^{\mathbf{I}}$, we associate a natural transformation $\delta\alpha : \delta P \to \delta Q$ as follows. For each name set $C \subseteq_{\mathrm{f}} \mathcal{N}$, and each function $\phi \in \delta P(C)$ and name $z \in (\mathcal{N} - C)$, let

$$\big((\delta\alpha)_C(\phi)\big)(z) = \alpha_{C\cup\{z\}}(\phi(z)) \quad . \tag{3.2.4c}$$

Notice the similarity between the definition of $[N \Rightarrow (-)]$ in equations 3.2.3 and the definition of δ in equations 3.2.4. Indeed we have a natural transformation

$$r : [N \Rightarrow (-)] \to \delta(-) \tag{3.2.5}$$

that acts by restricting the domains of functions.

Pointwise powerset. Fiore and Turi [2001] introduce non-determinism in $\mathbf{Set}^{\mathbf{I}}$ using variations of a 'pointwise powerset' functor $\mathscr{P}$ on $\mathbf{Set}^{\mathbf{I}}$, given as follows. For any name context $C \subseteq_{\mathrm{f}} \mathcal{N}$, we let

$$(\mathscr{P}P)(C) = \mathscr{P}(P(C)) \quad . \tag{3.2.6a}$$

Thus an element of the powerset of P at a name context C is a set of elements of P at the same name context C.

The functorial action of $\mathscr{P}P$ is given by direct image. So for any injection $\iota : C \rightarrowtail D$ in $\mathbf{I}$, and any subset $S \subseteq P(C)$,

$$(\mathscr{P}P)\iota(S) = \big\{ q \in P(D) \;\big|\; \exists p \in S.\ P\iota(p) = q \big\} \quad . \tag{3.2.6b}$$

For any natural transformation $\alpha : P \to Q$ between presheaves in $\mathbf{Set}^{\mathbf{I}}$, we have a natural transformation $\mathscr{P}\alpha : P \to Q$ given by pointwise direct image. So, for any name context C and any subset $S \subseteq P(C)$ we have

$$(\mathscr{P}\alpha)_C(S) = \big\{ q \in Q(C) \;\big|\; \exists p \in S.\ \alpha_C(p) = q \big\} \quad . \tag{3.2.6c}$$

The isomorphism (2.2.1a) is natural and so we have a natural isomorphism

$$\mathscr{P}(P) \times \mathscr{P}(Q) \cong \mathscr{P}(P + Q) \tag{3.2.7}$$

between presheaves in $\mathbf{Set}^{\mathbf{I}}$. We also have a natural isomorphism

$$i : \mathscr{P}(\delta(-)) \xrightarrow{\sim} \delta(\mathscr{P}(-)) \tag{3.2.8}$$

given as follows, for any presheaf $P \in \mathbf{Set}^{\mathbf{I}}$, any name context $C \subseteq_{\mathrm{f}} \mathcal{N}$ and any set $S \in (\mathscr{P}(\delta P))(C)$:

$$i_{P,C}(S) = \boldsymbol{\lambda} z \in (\mathcal{N} - C).\ \big\{ p \in P(C \cup \{z\}) \;\big|\; \exists \phi \in S.\ \phi(z) = p \big\} \quad .$$

It is straightforward to verify that such $i_{P,C}(S)$ satisfies the uniformity conditions imposed in the definition of $(\delta(\mathscr{P}P))(C)$, and that the resulting family $\big\{ i_{P,C} \big\}_{P \in \mathbf{Set}^{\mathbf{I}},\, C \in \mathbf{I}}$ is natural in C and P.

The *pointwise non-empty powerset* functor $\mathscr{P}_{\mathrm{ne}}$ is also of interest. It is the subfunctor $\mathscr{P}$ given by

$$\mathscr{P}_{\mathrm{ne}}P(C) = \{ S \in \mathscr{P}P(C) \mid S \neq \emptyset \} \quad .$$

Pointwise partial functions. Sets of partial functions play an important role in describing input behaviour in Section 2.2. We now recall the related notions suggested by Fiore and Turi for the context of name-passing.

Consider a presheaf $Q \in \mathbf{Set}^{\mathbf{I}}$ that has injective action, *i.e.* such that for every injection $\iota : C \rightarrowtail D$ in $\mathbf{I}$, the function $Q\iota : QC \to QD$ is injective. We define an operator $[Q \rightharpoonup (-)]$ on $\mathbf{Set}^{\mathbf{I}}$.

For any presheaf $P \in \mathbf{Set}^{\mathbf{I}}$, let $[Q \rightharpoonup P]$ be given as follows, for any $C \subseteq_{\mathrm{f}} \mathcal{N}$:

$$[Q \rightharpoonup P](C) = \{\phi : Q(C) \rightharpoonup P(C)\} \quad . \tag{3.2.9a}$$

(Here, we write $(Q(C) \rightharpoonup P(C))$ for the type of partial functions between sets $Q(C)$ and $P(C)$.) For any other name context $D \subseteq_{\mathrm{f}} \mathcal{N}$, any injection $\iota : C \rightarrowtail D$, and any partial function $\phi : Q(C) \rightharpoonup P(C)$, we let

$$[Q \rightharpoonup P]\iota(\phi) = \left(Q(D) \xrightarrow{(Q\iota)^{-1}} Q(C) \xrightarrow{\phi} P(C) \xrightarrow{P\iota} P(D) \right) \tag{3.2.9b}$$

where $(Q\iota)^{-1} : D \rightharpoonup C$ is the partial injective function given by, for any $q \in Q(D)$:

$$(Q\iota)^{-1}(q) = \begin{cases} q' & \text{if } Q\iota(q') = q \\ \text{undefined} & \text{if } q \notin \mathsf{im}(Q\iota) \end{cases} \quad .$$

For any natural transformation $\alpha : P \to P'$ between presheaves in $\mathbf{Set}^{\mathbf{I}}$, we define a natural transformation $[N \rightharpoonup \alpha] : [Q \rightharpoonup P] \to [Q \rightharpoonup P']$ as follows. For any name context $C \subseteq_{\mathrm{f}} \mathcal{N}$, and any partial function $\phi : Q(C) \rightharpoonup P(C)$, we let

$$[Q \rightharpoonup \alpha]_C(\phi) = \left(Q(C) \xrightarrow{\phi} P(C) \xrightarrow{\alpha_C} P'(C) \right) \quad .$$

A class of presheaves that have injective actions will play an important role in the next chapter. For now it is sufficient to note that the presheaves 1, N and indeed $(N \times N)$ all have injective actions. Indeed, to give an element of $[N \rightharpoonup P]$ is to give an element of the full function space $[N \Rightarrow (P + \{\text{undefined}\})]$, as specified in (3.2.3), that is always undefined on fresh names.

One explanation as to why the $[Q \rightharpoonup (-)]$ construction is appropriate is that we have the following natural isomorphism (*c.f.* (2.2.1b)).

$$[Q \rightharpoonup \mathscr{P}_{\mathrm{ne}}(-)] \cong \mathscr{P}(Q \times (-)) \tag{3.2.10}$$

Notation. For any partial function $f : A \rightharpoonup B$ between sets A and B, if a is an element of A then we write $(f \downarrow a)$ to mean that f is defined at a.

3.2.2 Behaviour endofunctors for early and ground bisimulation

We now describe two endofunctors on $\mathbf{Set}^{\mathbf{I}}$, using the constructions that we introduced in the previous section. In Section 3.3 we will see that coalgebraic bisimulation for these endofunctors respectively corresponds to the early and ground bisimulations introduced for the π-calculus in Section 3.1.

Fiore and Turi [2001, eqn. (24)] have also introduced an endofunctor that captures late bisimulation, but this will not be considered here.

Early behaviour endofunctor. Fiore and Turi [2001, eqn. (26)] have suggested the following behaviour endofunctor B_{e} on $\mathbf{Set}^{\mathbf{I}}$ for capturing the early semantics of name-passing systems. The endofunctor takes a similar shape to that for value-passing introduced in equation 2.2.4. We now

have two forms of output, though, and we use the restricted partial exponential and powerset functors. In the definition here we explicitly name the components of the product.

$$\begin{array}{rlll} B_{\mathrm{e}}(-) = & \mathsf{inp}: & [N \rightharpoonup [N \Rightarrow \mathscr{P}_{\mathrm{ne}}(-)]] & \textit{Input} \\ \times & \mathsf{out}: & [(N \times N) \rightharpoonup \mathscr{P}_{\mathrm{ne}}(-)] & \textit{Free output} \\ \times & \mathsf{bout}: & [N \rightharpoonup \delta(\mathscr{P}_{\mathrm{ne}}(-))] & \textit{Bound output} \\ \times & \mathsf{tau}: & [1 \rightharpoonup \mathscr{P}_{\mathrm{ne}}(-)] & \textit{Silent}. \end{array} \tag{3.2.11}$$

As indicated, we will write π_{inp}, π_{out}, π_{bout}, π_{tau} for the projection functions.

A B_{e}-coalgebra is given by a presheaf $P \in \mathbf{Set}^{\mathbf{I}}$ together with a natural family of functions

$$\begin{array}{rl} \Big\{ h_C : P(C) \to & [C \rightharpoonup \big([N \Rightarrow \mathscr{P}_{\mathrm{ne}}P](C)\big)] \\ & \times\ [(C \times C) \rightharpoonup \big((\mathscr{P}_{\mathrm{ne}}P)(C)\big)] \\ & \times\ [C \rightharpoonup \big(\delta(\mathscr{P}_{\mathrm{ne}}P)(C)\big)] \\ & \times\ [1 \rightharpoonup \big((\mathscr{P}_{\mathrm{ne}}P)(C)\big)] \Big\}_{C \in \mathbf{I}} \quad . \end{array} \tag{3.2.12}$$

For each $p \in P(C)$ (thought of as a state p whose free names are included in C), the 4-tuple $h_C(p)$ is to be thought of as follows. Each component is a partial function. If the first component, $\pi_{\mathsf{inp}}(h_C(p))$, is defined at some $c \in C$, then we understand that the state p is capable of input on channel c; in this case the value of $\pi_{\mathsf{inp}}(h_C(p))(c)$ is a function associating to each input value $d \in \mathscr{N}$ a non-empty set S of states with free names included in $C \cup \{d\}$; the intended interpretation is that the state p might input d on channel c to become any state in S. The second component $\pi_{\mathsf{out}}(h_C(p))$ is defined at $(c,d) \in C \times C$ if p can output data d on channel c; in this case the value of $\pi_{\mathsf{out}}(h_C(p))(c,d)$ is to be thought of as a set of resumptions. The third component $\pi_{\mathsf{bout}}(h_C(p))$ is defined at $c \in C$ if p can perform a bound output on channel c; then we have $p' \in \big(\pi_{\mathsf{bout}}(h_C(p))(c)\big)(z)$ if p can output the fresh name z on channel c to become p'. Finally, the fourth component $\pi_{\mathsf{tau}}(h_C(p))$ is defined if p can perform a silent action, in which case $\pi_{\mathsf{tau}}(h_C(p))(*)$ is to be thought of as the set of states that can be reached from p following a silent action.

Following the above discussion, the interested reader can perhaps imagine how to define a B_{e}-coalgebra that models the π-calculus. We will introduce a B_{e}-coalgebra for the π-calculus in (3.3.10), by developing a formal correspondence with a class of labelled transition systems.

Ground behaviour endofunctor. We now introduce an endofunctor B_{g} on $\mathbf{Set}^{\mathbf{I}}$ for which coalgebras are related to the presentation of ground transition systems introduced in Figure 3.3. In that presentation bound input and bound output are treated the same. Thus we define B_{g} as follows.

$$\begin{array}{rlll} B_{\mathrm{g}}(-) = & \mathsf{binp}: & [N \rightharpoonup \delta(\mathscr{P}_{\mathrm{ne}}(-))] & \textit{Bound input} \\ \times & \mathsf{out}: & [(N \times N) \rightharpoonup \mathscr{P}_{\mathrm{ne}}(-)] & \textit{Free output} \\ \times & \mathsf{bout}: & [N \rightharpoonup \delta(\mathscr{P}_{\mathrm{ne}}(-))] & \textit{Bound output} \\ \times & \mathsf{tau}: & [1 \rightharpoonup \mathscr{P}_{\mathrm{ne}}(-)] & \textit{Silent}. \end{array} \tag{3.2.13}$$

A B_{g}-coalgebra for the π-calculus is given below, in (3.3.11), by taking advantage of a correspondence with a class of labelled transition systems.

The only difference between the endofunctors B_{e} and B_{g} is in their description of input. In (3.2.5) we introduced a natural transformation

$$r : [N \Rightarrow (-)] \to \delta(-)$$

and this induces a morphism of endofunctors $(\mathbf{Set}^{\mathbf{I}}, B_{\mathrm{e}}) \to (\mathbf{Set}^{\mathbf{I}}, B_{\mathrm{g}})$, in the sense of Section 2.3, and hence a functor

$$B_{\mathrm{e}}\text{-Coalg} \to B_{\mathrm{g}}\text{-Coalg} \quad . \tag{3.2.14}$$

This provides an abstract account of the derivation of a ground transition system from an early one (*i.e.* Prop. 3.1.9(1,3,4,5)).

Another description of B_{g} will be more useful in Chapters 7 and 8. First, we define an endofunctor L_{g} on $\mathbf{Set}^{\mathbf{I}}$ for *deterministic* name-passing systems.

$$\begin{array}{rcll} L_{\mathrm{g}}(-) & = & \mathsf{binp}: & N \times \delta(-) & \textit{Bound input} \\ & + & \mathsf{out}: & N \times N \times (-) & \textit{Free output} \\ & + & \mathsf{bout}: & N \times \delta(-) & \textit{Bound output} \\ & + & \mathsf{tau}: & (-) & \textit{Silent}. \end{array} \tag{3.2.15}$$

Via the isomorphisms (3.2.7) and (3.2.10) we have the following isomorphism, which exhibits B_{g} in a simpler form.

$$B_{\mathrm{g}}(-) \;\cong\; \mathscr{P}(L_{\mathrm{g}}(-)) \tag{3.2.16}$$

Several authors, including Power and Turi [1999] and Hasuo, Jacobs, and Sokolova [2006], have found that endofunctors of the form (3.2.16) are particularly amenable to the study of linear time (trace) semantics. This decomposition will prove useful in Chapter 8, for essentially the same reason.

3.3 Transition systems for name-passing

We now turn from the coalgebraic models of the previous section to transition system models of name-passing. We begin this section with a notion of **I**-*indexed labelled transition system*, which is a labelled transition system whose states are elements of presheaves. This notion of model is inspired by the models of Cattani and Sewell [2004]. We are able to consider the transition systems of the π-calculus in this framework, and we can frame the kinds of bisimulation that arise in the π-calculus within the model.

We begin, in Section 3.3.1, by developing a theory of **I**-*indexed early labelled transition systems* (**I**-IL$_{\mathrm{e}}$TSs) which are intended to model the early semantics of input. We define a notion of bisimulation for these transition systems. In Section 3.3.2, we investigate the connections between **I**-IL$_{\mathrm{e}}$TSs and the classes of coalgebras considered in the previous section. Every B_{e}-coalgebra gives rise to an **I**-IL$_{\mathrm{e}}$TS, and we axiomatise those **I**-IL$_{\mathrm{e}}$TSs that are induced by coalgebras in this way, with a result analogous to Theorem 2.2.5.

Given the tight correspondence that is achieved between the transition system and coalgebra models of name-passing, one might expect a tight correspondence between the notions of bisimulation arising in the transition system and coalgebra models. We investigate this in Section 3.3.3.

Section 3.3.4 is concerned with reformulating the basic definitions of Section 3.3.1 for ground transition systems. We explain that every **I**-IL$_{\mathrm{e}}$TS gives rise to an **I**-indexed ground labelled transition system.

3.3.1 Indexed early labelled transition systems

Labels. We define a set of labels for early transitions by

$$Lab_{\mathrm{e}} = \mathscr{N} \times \mathscr{N} \;+\; \mathscr{N} \times \mathscr{N} \;+\; 1 \tag{3.3.1}$$

writing $c?d$, $c!d$, τ for elements of the first (input), second (output) and third (silent) summands respectively.

As discussed in Section 3.1, we do not include a distinguished bound output label for the early transition systems. Bound outputs can be identified as outputs where the data is not in the context of the source state.

We define functions $\mathsf{ch}, \mathsf{dat} : Lab_{\mathrm{e}} \to \mathscr{P}(\mathcal{N})$ that assign to each label the set of names involved respectively in the channel and data.

$\ell \in Lab_{\mathrm{e}}$	$\mathsf{ch}(\ell)$	$\mathsf{dat}(\ell)$
$c?d$	$\{c\}$	$\{d\}$
$c!d$	$\{c\}$	$\{d\}$
τ	$\emptyset$	$\emptyset$

Transition systems. We define **I**-indexed early labelled transition systems as labelled transition relations, with labels from Lab_{e}, and with state sets being sets of elements of presheaves.

Definition 3.3.2. An **I**-*indexed early labelled transition system* (**I**-IL$_{\mathrm{e}}$TS) is a presheaf $P \in \mathbf{Set}^{\mathbf{I}}$ together with a transition relation $\longrightarrow \subseteq \int P \times Lab_{\mathrm{e}} \times \int P$.

(Here, $\int P$ is the set of elements of P, introduced at the beginning of Section 3.2.)

Example: the π-calculus. The π-calculus can be given an early semantics in terms of an **I**-IL$_{\mathrm{e}}$TS. The carrier presheaf is P_π, introduced in equation 3.2.1, while the transition relation is defined using the relation $\xrightarrow[\pi]{}$ on π-calculus terms introduced in Figure 3.2, as follows. We let $\xrightarrow[P_\pi]{}$ be the least relation

$$\xrightarrow[P_\pi]{} \subseteq \int P_\pi \times Lab_{\mathrm{e}} \times \int P_\pi$$

satisfying the following implications. We make use of Prop. 3.1.2 in determining name contexts for right-hand sides of the transitions. The definition is essentially the body of Lemma 2.3 of Cattani and Sewell [2004].

$$\begin{aligned}
&\text{If } p \xrightarrow[\pi]{cd} p' \text{ and } \mathsf{fn}(p) \subseteq C \text{ then } C \vdash p \xrightarrow[P_\pi]{c?d} C \cup \{d\} \vdash p';\\
&\text{If } p \xrightarrow[\pi]{\bar{c}d} p' \text{ and } \mathsf{fn}(p) \subseteq C \text{ then } C \vdash p \xrightarrow[P_\pi]{c!d} C \vdash p';\\
&\text{If } p \xrightarrow[\pi]{\bar{c}(z)} p' \text{ and } \mathsf{fn}(p) \subseteq C \not\ni z \text{ then } C \vdash p \xrightarrow[P_\pi]{c!z} C \cup \{z\} \vdash p';\\
&\text{If } p \xrightarrow[\pi]{\tau} p' \text{ and } \mathsf{fn}(p) \subseteq C \text{ then } C \vdash p \xrightarrow[P_\pi]{\tau} C \vdash p'.
\end{aligned} \tag{3.3.3}$$

Indexed bisimulations. We now introduce notions of bisimulation for these indexed labelled transition systems.

Definition 3.3.4.

1. Consider presheaves $P, Q \in \mathbf{Set}^{\mathbf{I}}$. An **I**-*indexed binary relation* R between P and Q is a presheaf $R \in \mathbf{Set}^{\mathbf{I}}$ that is a subobject of $P \times Q$. Concretely, for each name context $C \subseteq_{\mathrm{f}} \mathcal{N}$ a relation $R(C) \subseteq P(C) \times Q(C)$ must be given, subject to the constraint that for any injection $\iota : C \rightarrowtail D$, and any $(p, q) \in R(C)$, we have $([\iota]p, [\iota]q) \in R(D)$.

2. Consider two **I**-IL$_{\mathrm{e}}$TSs with carriers $P, Q \in \mathbf{Set}^{\mathbf{I}}$ and relations

$$\xrightarrow[P]{} \subseteq \int P \times Lab_{\mathrm{e}} \times \int P \qquad \xrightarrow[Q]{} \subseteq \int Q \times Lab_{\mathrm{e}} \times \int Q \quad .$$

An **I**-*indexed early simulation* between $(P, \xrightarrow[P]{})$ and $(Q, \xrightarrow[Q]{})$ is an **I**-indexed binary relation $R \subseteq P \times Q$ such that

$$\forall C \subseteq_{\mathrm{f}} \mathcal{N},\ (p,q) \in R(C),\ \ell \in Lab_{\mathrm{e}},\ (C' \vdash p') \in \int P.$$
$$C \vdash p \xrightarrow[P]{\ell} C' \vdash p' \implies \exists q' \in Q(C').\ C \vdash q \xrightarrow[Q]{\ell} C' \vdash q' \text{and } (p', q') \in R(C') \quad .$$

3. An **I**-*indexed early bisimulation* between **I**-IL$_e$TSs $(P, \xrightarrow[P]{})$ and $(Q, \xrightarrow[Q]{})$ is an **I**-indexed binary relation R between P and Q such that R is an **I**-indexed early simulation between $(P, \xrightarrow[P]{})$ and $(Q, \xrightarrow[Q]{})$ and also R^{op} is an **I**-indexed early simulation between $(Q, \xrightarrow[Q]{})$ and $(P, \xrightarrow[P]{})$.

   (Here, R^{op} is the **I**-indexed binary relation between Q and P given by, for each $C \in \mathbf{I}$, $R^{\mathrm{op}}(C) = (R(C))^{\mathrm{op}}$.)

In this chapter and the next, when we speak of relations between presheaves we will mean spans that are not only jointly monic, but which are also componentwise subsets, as in Definition 3.3.4(1).

The notion of bisimulation is justified by the following result.

Proposition 3.3.5 (*c.f.* Cattani and Sewell 2004, Thm. 2.4)**.** *Consider two π-calculus terms, p, q, with $\mathsf{fn}(p) \cup \mathsf{fn}(q) \subseteq C$. The term p is early bisimilar with q (according to Definition 3.1.10(1)) if and only if there is an* **I**-*indexed early bisimulation R on $(P_\pi, \xrightarrow[P_\pi]{})$ such that $(p, q) \in R(C)$.* □

Remark. It is clear that, using the terminology of Section 2.4, an **I**-indexed early labelled transition system is the same thing as a $\int$-structured $\mathscr{P}(Lab_e \times -)$-coalgebra, and an **I**-indexed early bisimulation is the same thing as a $\int$-structured $\mathscr{P}(Lab_e \times -)$-bisimulation. We will not, however, make use of these descriptions in this thesis.

3.3.2 Relating I-indexed early labelled transition systems with coalgebras

We now relate the **I**-IL$_e$TSs introduced in this section with the coalgebraic models introduced in the previous one. We begin by explaining how every B_e-coalgebra gives rise to an **I**-IL$_e$TS.

We axiomatise those **I**-IL$_e$TSs that are induced by B_e-coalgebras in Figure 3.4. This provides a complete concrete characterisation of the B_e-coalgebras.

I-indexed early labelled transition systems from B_e-coalgebras. The discussion after equation 3.2.12 has informally described how a B_e-coalgebra can be thought of as a transition function. We now make this connection precise. Given a B_e-coalgebra $(P, h : P \to B_e P)$ we let $\longrightarrow_h \subseteq \int P \times Lab_e \times \int P$ be the least relation satisfying the following statements $(B_e{\to}\mathbf{I})$-1 – 4.

$(B_e{\to}\mathbf{I})$-1. For any $C \subseteq_f \mathcal{N}$, $p \in P(C)$, $c \in C$, $d \in \mathcal{N}$, and $p' \in P(C \cup \{d\})$:
if $\pi_{\mathsf{inp}}(h_C(p)) \downarrow c$ and $p' \in (\pi_{\mathsf{inp}}(h_C(p))(c))(d)$
then $C \vdash p \xrightarrow{c?d}_h C \cup \{d\} \vdash p'$.

$(B_e{\to}\mathbf{I})$-2. For any $C \subseteq_f \mathcal{N}$, $p \in P(C)$, $c, d \in C$, and $p' \in P(C)$:
if $\pi_{\mathsf{out}}(h_C(p)) \downarrow (c, d)$ and $p' \in \pi_{\mathsf{out}}(h_C(p))(c, d)$
then $C \vdash p \xrightarrow{c!d}_h C \vdash p'$.

$(B_e{\to}\mathbf{I})$-3. For any $C \subseteq_f \mathcal{N}$, $p \in P(C)$, $c \in C$, $z \in (\mathcal{N} - C)$, and $p' \in P(C \cup \{z\})$:
if $\pi_{\mathsf{bout}}(h_C(p)) \downarrow c$ and $p' \in (\pi_{\mathsf{bout}}(h_C(p))(c))(z)$
then $C \vdash p \xrightarrow{c!z}_h C \cup \{z\} \vdash p$.

$(B_e{\to}\mathbf{I})$-4. For any $C \subseteq_f \mathcal{N}$, $p \in P(C)$, and $p' \in P(C)$:
if $\pi_{\mathsf{tau}}(h_C(p)) \downarrow *$ and $p' \in \pi_{\mathsf{tau}}(h_C(p))(*)$
then $C \vdash p \xrightarrow{\tau}_h C \vdash p'$.

I-IL$_e$TSs that are induced by B_e-coalgebras. Not every **I**-indexed early labelled transition system is induced by a B_e-coalgebra according to $(B_e{\to}\mathbf{I})$-1–4. We can, however, axiomatise those that are. Axioms **I**1–**I**6 in Figure 3.4 are intended to capture those **I**-indexed early labelled transition systems that are induced by coalgebras; this is a result that will be shown in Theorem 3.3.8.

I1. Channel is known and at most transmitted data is learnt:

$$C \vdash p \xrightarrow{\ell} C' \vdash p' \implies \mathsf{ch}(\ell) \subseteq C \ \wedge\ C' = C \cup \mathsf{dat}(\ell)$$

I2. If one name can be input, then so can any other: for all $d' \in \mathcal{N}$:

$$C \vdash p \xrightarrow{c?d} C \cup \{d\} \vdash p'$$
$$\implies \exists p'' \in P(C \cup \{d'\}).\quad C \vdash p \xrightarrow{c?d'} C \cup \{d'\} \vdash p''$$

I3. Bijective maps preserve transitions: for all $D \in \mathbf{I}$, with $C \cup \mathsf{dat}(\ell) \overset{\beta}{\cong} D$:

$$C \vdash p \xrightarrow{\ell} C \cup \mathsf{dat}(\ell) \vdash p' \ \wedge\ \mathsf{ch}(\ell) \subseteq C$$
$$\implies \beta(C) \vdash [\beta|_C]p \xrightarrow{[\beta]\ell} D \vdash [\beta]p'$$

I4a. Knowing/forgetting input data preserves transitions:

$$C \vdash p \xrightarrow{c?z} C \cup \{z\} \vdash p'$$
$$\iff\ C \cup \{z\} \vdash [C \hookrightarrow C \cup \{z\}]p \xrightarrow{c?z} C \cup \{z\} \vdash p'$$

I4b. Known output data must really be known:

$$C \cup \{d\} \vdash [C \hookrightarrow C \cup \{d\}]p \xrightarrow{c!d} C \cup \{d\} \vdash p' \implies d \in C$$

I5. Inclusion maps preserve transitions:

$$C \vdash p \xrightarrow{\ell} C \cup \mathsf{dat}(\ell) \vdash p' \ \wedge\ (D \setminus C) \cap \mathsf{dat}(\ell) = \emptyset$$
$$\implies D \vdash [C \hookrightarrow D]p \xrightarrow{\ell} D \cup \mathsf{dat}(\ell) \vdash [C \cup \mathsf{dat}(\ell) \hookrightarrow D \cup \mathsf{dat}(\ell)]p'$$

I6. Inclusion maps reflect transitions:

$$D \vdash [C \hookrightarrow D]p \xrightarrow{\ell} D \cup \mathsf{dat}(\ell) \vdash p' \ \wedge\ (D \setminus C) \cap \mathsf{dat}(\ell) = \emptyset$$
$$\implies \exists p'' \in P(C \cup \mathsf{dat}(\ell)).$$
$$[C \cup \mathsf{dat}(\ell) \hookrightarrow D \cup \mathsf{dat}(\ell)]p'' = p' \ \wedge\ C \vdash p \xrightarrow{\ell} C \cup \mathsf{dat}(\ell) \vdash p''$$

Figure 3.4: Requirements on an **I**-indexed labelled transition system over $P \in \mathbf{Set}^{\mathbf{I}}$.

Axiom **I**1 is an abstract form of to Prop. 3.1.2. Axiom **I**2 mirrors Axiom $\mathbb{V}$-1 of Figure 2.2: this axiom, building on Corollary 3.1.4, requires that the eligibility of an input transition is not dependent on the data involved. Axiom **I**3 is related to Prop. 3.1.12(1), but it also gives rise to analogues of Corollary 3.1.4(2) and Prop. 3.1.8. For if fresh data z (*i.e.* with $z \notin C$) can be input (or output), we have that any other fresh name z' could just have well been input (output), by considering the bijection $[z'/z] : C \cup \{z\} \xrightarrow{\sim} C \cup \{z'\}$.

Axioms **I**4–**I**6 relate to changes in size of name-context. Axiom **I**4a describes a dichotomy for input transitions: the eligibility of an input transition is not dependent on the size of the context of names. Axiom **I**4b enforces a separation between output of known names and output of fresh names, allowing us to code bound outputs as outputs of fresh data. Finally, Axioms **I**5 and **I**6 capture a dichotomy for transitions — in general, transitions cannot depend on learning or forgetting names, as long as the data is kept well out of the way.

The **I**-indexed early labelled transition system $\xrightarrow[P_\pi]{}$ for the π-calculus, as introduced in (3.3.3), satisfies Axioms **I**1–**I**6. Indeed, Axioms **I**1–**I**3 appeared as propositions in Section 3.1, while Axioms **I**4–**I**6 arise from the definition (3.3.3).

We now continue to explain the correspondence between B_{e}-coalgebras and **I**-ILTSs.

Theorem 3.3.6. *For every B_e-coalgebra (P,h), the induced* **I**-IL$_e$TS $(P,\longrightarrow_h)$ *satisfies Axioms* **I**1–**I**6.

Our proof of this theorem is rather laborious and is postponed to Appendix 3.A.

B_e-coalgebras from I-indexed early labelled transition systems. To each **I**-IL$_e$TS $(P,\longrightarrow)$ that satisfies Axioms **I**2 and **I**3 is assigned the family of functions

$$\left\{\overrightarrow{h}_C : P(C) \to B_e P(C)\right\}_{C\in\mathbf{I}}$$

defined as follows.

$(\mathbf{I}{\to}B_e)$-1. For any $C \subseteq_f \mathcal{N}$, $p \in P(C)$, $c \in C$:
$\pi_{\mathsf{inp}}(\overrightarrow{h}_C(p)) \downarrow c$ if and only if
there exist $d \in \mathcal{N}$ and $p' \in P(C \cup \{d\})$
such that $C \vdash p \xrightarrow{c?d} C \cup \{d\} \vdash p'$;
In this case, for every $d \in \mathcal{N}$
$$\left(\pi_{\mathsf{inp}}(\overrightarrow{h}_C(p))(c)\right)(d) = \left\{p' \;\middle|\; C \vdash p \xrightarrow{c?d} C \cup \{d\} \vdash p'\right\}.$$

$(\mathbf{I}{\to}B_e)$-2. For any $C \subseteq_f \mathcal{N}$, $p \in P(C)$, $c,d \in C$:
$\pi_{\mathsf{out}}(\overrightarrow{h}_C(p)) \downarrow (c,d)$ if and only if
there is $p' \in P(C)$ such that $C \vdash p \xrightarrow{c!d} C \vdash p'$;
In this case,
$$\pi_{\mathsf{out}}(\overrightarrow{h}_C(p))(c,d) = \left\{p' \;\middle|\; C \vdash p \xrightarrow{c!d} C \vdash p'\right\}.$$

$(\mathbf{I}{\to}B_e)$-3. For any $C \subseteq_f \mathcal{N}$, $p \in P(C)$, $c \in C$:
$\pi_{\mathsf{bout}}(\overrightarrow{h}_C(p)) \downarrow c$ if and only if
there exist $d \in (\mathcal{N} - C)$ and $p' \in P(C \cup \{d\})$
such that $C \vdash p \xrightarrow{c!d} C \cup \{d\} \vdash p'$;
In this case, for every $d \in (\mathcal{N} - C)$,
$$\left(\pi_{\mathsf{bout}}(\overrightarrow{h}_C(p))(c)\right)(d) = \left\{p' \;\middle|\; C \vdash p \xrightarrow{c!d} C \cup \{d\} \vdash p'\right\}.$$

$(\mathbf{I}{\to}B_e)$-4. For any $C \subseteq_f \mathcal{N}$, $p \in P(C)$:
$\pi_{\mathsf{tau}}(\overrightarrow{h}_C(p)) \downarrow *$ if and only if
there is $p' \in P(C)$ such that $C \vdash p \xrightarrow{\tau} C \vdash p'$
In this case,
$$\pi_{\mathsf{tau}}(\overrightarrow{h}_C(p))(*) = \left\{p' \;\middle|\; C \vdash p \xrightarrow{\tau} C \vdash p'\right\}.$$

We must show that the sets constructed by comprehension in the definition are not empty, and also that $(\mathbf{I}{\to}B_e)$-1 and $(\mathbf{I}{\to}B_e)$-3 respect the uniformity conditions imposed in the definitions of $[N \Rightarrow -]$ (3.2.3a) and of δ (3.2.4).

When $\pi_{\mathsf{out}}(\overrightarrow{h}_C(p))$ is defined at (c,d), it follows from the definition ($(\mathbf{I}{\to}B_e)$-2) that $\pi_{\mathsf{out}}(\overrightarrow{h}_C(p))(c,d)$ is inhabited. Similarly, $\pi_{\mathsf{tau}}(\overrightarrow{h}_C(p))(*)$ is always inhabited whenever it is defined. The set constructed in $(\mathbf{I}{\to}B_e)$-1 is never empty as a result of Axiom **I**2. As for the set involved in $(\mathbf{I}{\to}B_e)$-3, notice that if $\pi_{\mathsf{bout}}(\overrightarrow{h}_C(p)) \downarrow c$ then there must be $d \in (\mathcal{N} - C)$ such that $C \vdash p \xrightarrow{c!d} C \cup \{d\} \vdash p'$. Consider another fresh name $d' \in (\mathcal{N} - C)$. Axiom **I**3 together with the substitution $[d'/d] : C \cup \{d\} \xrightarrow{\sim} C \cup \{d'\}$ gives a transition $C \vdash p \xrightarrow{c!d'} C \cup \{d'\} \vdash [d'/d]p'$. So we have $[d'/d]p' \in \left(\pi_{\mathsf{bout}}(\overrightarrow{h}_C(p))(c)\right)(d')$, *i.e.*, $\left(\pi_{\mathsf{bout}}(\overrightarrow{h}_C(p))(c)\right)(d')$ is inhabited.

We now explain why the uniformity constraints hold of the constructions introduced in $(\mathbf{I}{\to}B_e)$-1 and $(\mathbf{I}{\to}B_e)$-3. As regards $(\mathbf{I}{\to}B_e)$-1, we must show that for any $d,d' \in (\mathcal{N} - C)$,

$$\left(\pi_{\mathsf{inp}}(\overrightarrow{h}_C(p))(c)\right)(d') = (\mathscr{P}_{\mathrm{ne}}P)[d'/d]\left(\pi_{\mathsf{inp}}(\overrightarrow{h}_C(p))(c)\right)(d) \quad .$$

This amounts to showing that for every $p' \in P(C \cup \{d\})$ such that

$$C \vdash p \xrightarrow{c?d} C \cup \{d\} \vdash p'$$

we have $C \vdash p \xrightarrow{c?d'} C \cup \{d'\} \vdash [d'/d]p'$. Since $d, d' \notin C$, this arises immediately from Axiom **I**3.

Validity of $(\mathbf{I}{\rightarrow}B_{\mathrm{e}})$-3 is derived from Axiom **I**3 in a similar manner. Thus for each **I**-IL$_{\mathrm{e}}$TS $\longrightarrow$ that satisfies Axioms **I**2 and **I**3, we have a family of maps

$$\left\{\overrightarrow{h}_C \colon P(C) \to (B_{\mathrm{e}}P)(C)\right\}_{C \in \mathbf{I}} \quad .$$

Theorem 3.3.7. *For any* **I**-IL$_{\mathrm{e}}$TS$\longrightarrow$ *that satisfies Axioms* **I**1–**I**6*, the induced family* $\left\{\overrightarrow{h}_C\right\}_C$ *is natural in* C*.*

The proof of this theorem is postponed to Appendix 3.A.

Thus we have a procedure for extracting a B_{e}-coalgebra from an **I**-IL$_{\mathrm{e}}$TS that satisfies Axioms **I**1–**I**6. In fact, the class of B_{e}-coalgebras is in bijection with the class of **I**-IL$_{\mathrm{e}}$TSs satisfying Axioms **I**1–**I**6.

Theorem 3.3.8. *The procedure of* $(\mathbf{I}{\rightarrow}B_{\mathrm{e}})$*-1–4 is left and right inverse to* $(B_{\mathrm{e}}{\rightarrow}\mathbf{I})$*-1–4.*

Proof. We must show that for any presheaf $P \in \mathbf{Set}^{\mathbf{I}}$, and any B_{e}-coalgebra $h : P \to B_{\mathrm{e}}P$, we have

$$h = {}^{(\overrightarrow{h})} \tag{3.3.9a}$$

and that for any **I**-IL$_{\mathrm{e}}$TS $\longrightarrow \subseteq \int P \times Lab_{\mathrm{e}} \times \int P$ satisfying Axioms **I**1–**I**6, we have

$$\longrightarrow \; = \longrightarrow_{(\overrightarrow{h})} \quad . \tag{3.3.9b}$$

(Here (3.3.9a) is equality of natural transformations and (3.3.9b) is equality of labelled transition relations.)

Equation 3.3.9a follows straightforwardly from the definitions of $(\mathbf{I}{\rightarrow}B_{\mathrm{e}})$-1–4 and $(B_{\mathrm{e}}{\rightarrow}\mathbf{I})$-1–4. As regards equation 3.3.9b, the inclusion

$$\longrightarrow \supseteq \longrightarrow_{(\overrightarrow{h})}$$

is equally straightforward. To show the inclusion

$$\longrightarrow \subseteq \longrightarrow_{(\overrightarrow{h})}$$

we make use of Axiom **I**1 on $\longrightarrow$. For instance, suppose that we have $C, C' \subseteq_{\mathrm{f}} \mathcal{N}$ and $p \in P(C)$ and $p' \in P(C')$ together with $c, d \in \mathcal{N}$ such that

$$C \vdash p \xrightarrow{c?d} C' \vdash p' \quad .$$

Axiom **I**1 ensures that $c \in C$ and that $C' = C \cup \{d\}$. So, by $(\mathbf{I}{\rightarrow}B_{\mathrm{e}})$-1, we have that $\pi_{\mathsf{inp}}(\overrightarrow{h}_C(p)) \downarrow c$ and

$$p' \in \left(\pi_{\mathsf{inp}}(\overrightarrow{h}_C(p))\right)(c))(d) \quad .$$

So, by $(B_{\mathrm{e}}{\rightarrow}\mathbf{I})$-1, we have

$$C \vdash p \xrightarrow{c?d}_{(\overrightarrow{h})} C' \vdash p' \quad .$$

The other modes of communication are treated similarly. □

Coalgebras for the π-calculus. Theorem 3.3.8 allows us to give coalgebraic interpretations of the π-calculus. The **I**-IL$_e$TS $(P_\pi, \xrightarrow[P_\pi]{})$ defined in (3.3.3) induces a B_e-coalgebra structure

$$P_\pi \to B_e P_\pi \quad . \tag{3.3.10}$$

Using the mapping described in (3.2.14), we also have a B_g-coalgebra structure

$$P_\pi \to B_g P_\pi \quad . \tag{3.3.11}$$

3.3.3 Relating notions of bisimulation

The induction of transition systems from coalgebras is most credible if the bisimulation on the induced transition systems corresponds to coalgebraic bisimulation. For the name-passing case, the reader may have anticipated that B_e-bisimulation between B_e-coalgebras coincided with **I**-indexed early bisimulation for the corresponding **I**-indexed early labelled transition systems (induced according to $(B_e{\to}\mathbf{I})$-1 – 4). Unfortunately, and contrary to the claims of Fiore and Turi [2001, Prop. 2.4], this is not quite the case.

This disparity is only a minor anomaly. The equivalence relation with which we are ultimately concerned is bisimilarity, the greatest bisimulation. As will be shown, the greatest coalgebraic bisimulation does coincide with the greatest transition system bisimulation.

We begin this subsection by noting that every coalgebraic bisimulation is a bisimulation for the labelled transition system; it is the converse that is the problem. An illustration of this problem is provided, and we then provide a weaker form for the converse of the result, by imposing a closure condition on the relations under consideration. Thus we can conclude that a final B_e-bisimulation is a final **I**-indexed early bisimulation.

In the next chapter, in Theorem 4.2.5, we show that, for B_e-coalgebras whose carriers satisfy a sheaf condition, there is no problem and the B_e-bisimulations are exactly the **I**-indexed early bisimulations.

Coalgebraic bisimulations are labelled transition system bisimulations. We first establish a general lemma, and then note that every coalgebraic bisimulation is a bisimulation for the labelled transition system.

Lemma 3.3.12. Consider B_e-coalgebras (P,h) and (Q,k) and an **I**-indexed binary relation R between P and Q. Suppose we have a B_e-coalgebra structure $r : R \to B_e R$ on R.

The structure r lifts R to a span of coalgebras if and only if the following three properties hold of the induced **I**-IL$_e$TSs.

1. If $C \vdash (p,q) \xrightarrow{\ell}_r C' \vdash (p',q')$ then $C \vdash p \xrightarrow{\ell}_h C' \vdash p'$ and $C \vdash q \xrightarrow{\ell}_k C' \vdash q'$.
2. If $(p,q) \in R(C)$ and $C \vdash p \xrightarrow{\ell}_h C' \vdash p'$ then there is $q' \in Q(C')$ such that $(p',q') \in R(C')$ and $C \vdash (p,q) \xrightarrow{\ell}_r C' \vdash (p',q')$.
3. If $(p,q) \in R(C)$ and $C \vdash q \xrightarrow{\ell}_k C' \vdash q'$ then there is $p' \in P(C')$ such that $(p',q') \in R(C')$ and $C \vdash (p,q) \xrightarrow{\ell}_r C' \vdash (p',q')$.

Proof notes. This lemma follows straightforwardly from the definitions. For instance, suppose that R is a B_e-bisimulation. We will show item (2) for the case of bound output labels $\ell = c!(z)$.

If $(p,q) \in R(C)$ and $C \vdash p \xrightarrow{\ell}_h C' \vdash p'$, then we know, from $(B_e{\to}\mathbf{I})$-3, that $C' = C \cup \{z\}$, and that $\pi_{\mathsf{bout}}(h_C(p))$ is defined at c, and that the set $(\pi_{\mathsf{bout}}(h_C(p))(c))(z)$ contains p'. Since the left projection $r_1 : R \to P$ is a coalgebra homomorphism, we know that $(B_e r_1)_C(r_C(p,q)) = h_C(p)$, and so, from the action of B_e, we know that $\pi_{\mathsf{bout}}(r_C(p,q))$ is defined at c, and that there is

an element $q' \in Q(C \cup \{z\})$ such that the pair (p', q') is in $R(C \cup \{z\})$, and moreover that the pair (p', q') is in the set $(\pi_{\mathsf{bout}}(r_C(p,q))(c))(z)$. Thus, by $(B_{\mathrm{e}}{\to}\mathbf{I})$-3, we know that the transition $C \vdash (p,q) \xrightarrow{\ell}_r C' \vdash (p', q')$ is induced. □

Proposition 3.3.13. *Every B_{e}-bisimulation relation R between B_{e}-coalgebras (P,h) and (Q,k) is also an* **I***-indexed early bisimulation between the induced* **I**-IL$_{\mathrm{e}}$TS*s $\longrightarrow_h$ and $\longrightarrow_k$.*

Proof. We consider B_{e}-coalgebras (P,h) and (Q,k), and a B_{e}-bisimulation relation R between them. We write the projections of the relation as $r_1 : R \to P$ and $r_2 : R \to Q$, and we have a natural transformation $r : R \to B_{\mathrm{e}}R$ lifting R to a span of coalgebra homomorphisms. It follows that properties 1–3 of Lemma 3.3.12 are satisfied, and from these conditions it follows immediately that R is an **I**-indexed early bisimulation between $(P, \longrightarrow_h)$ and $(Q, \longrightarrow_k)$. □

Anomaly. We now explain why the converse of Prop. 3.3.13 does not hold. We invent three distinct states, p, p_1, p_2, and consider the presheaf $P \in \mathbf{Set}^{\mathbf{I}}$ given by

$$\begin{aligned} P(\emptyset) &= \{\mathrm{p}, \mathrm{p}_1, \mathrm{p}_2\} + \{\mathrm{p}, \mathrm{p}_1, \mathrm{p}_2\} \\ P(C) &= \{\mathrm{p}, \mathrm{p}_1, \mathrm{p}_2\} \qquad\qquad \text{for } C \neq \emptyset \quad . \end{aligned}$$

The action of P on injections $C \rightarrowtail D$ is identity, except when $C = \emptyset$, in which case the action is the codiagonal.

We define an **I**-IL$_{\mathrm{e}}$TS structure $\longrightarrow$ on P to be the least admitting the following transitions.

$$\begin{array}{lll} \emptyset \vdash \mathsf{inl}\,\mathrm{p} \xrightarrow{\tau} \emptyset \vdash \mathsf{inl}\,\mathrm{p}_1 & & \emptyset \vdash \mathsf{inl}\,\mathrm{p} \xrightarrow{\tau} \emptyset \vdash \mathsf{inl}\,\mathrm{p}_2 \\ \emptyset \vdash \mathsf{inr}\,\mathrm{p} \xrightarrow{\tau} \emptyset \vdash \mathsf{inr}\,\mathrm{p}_1 & & \emptyset \vdash \mathsf{inr}\,\mathrm{p} \xrightarrow{\tau} \emptyset \vdash \mathsf{inr}\,\mathrm{p}_2 \\ C \vdash \mathrm{p} \xrightarrow{\tau} C \vdash \mathrm{p}_1 & & C \vdash \mathrm{p} \xrightarrow{\tau} C \vdash \mathrm{p}_2 \qquad \text{for } C \neq \emptyset \end{array}$$

It is clear that this **I**-IL$_{\mathrm{e}}$TS satsifies Axioms **I**1–**I**6, and so, by Theorem 3.3.7, we have a B_{e}-coalgebra structure $P \to B_{\mathrm{e}}P$.

We define an **I**-indexed binary relation $R \subseteq P \times P$ as follows.

$$\begin{aligned} R(\emptyset) &= \left\{ \begin{array}{l} (\mathsf{inl}\,\mathrm{p}, \mathsf{inl}\,\mathrm{p}), (\mathsf{inl}\,\mathrm{p}_1, \mathsf{inl}\,\mathrm{p}_1), (\mathsf{inl}\,\mathrm{p}_2, \mathsf{inl}\,\mathrm{p}_2), \\ (\mathsf{inr}\,\mathrm{p}, \mathsf{inr}\,\mathrm{p}), (\mathsf{inr}\,\mathrm{p}_1, \mathsf{inr}\,\mathrm{p}_2), (\mathsf{inr}\,\mathrm{p}_2, \mathsf{inr}\,\mathrm{p}_1) \end{array} \right\} \\ R(C) &= \{(\mathrm{p},\mathrm{p}), (\mathrm{p}_1,\mathrm{p}_1), (\mathrm{p}_1,\mathrm{p}_2), (\mathrm{p}_2,\mathrm{p}_1), (\mathrm{p}_2,\mathrm{p}_2)\} \qquad \text{for } C \neq \emptyset \end{aligned}$$

It is straightforward to check that this is a **I**-indexed early bisimulation on $(P, \longrightarrow)$. This relation, however, is not a B_{e}-bisimulation on the induced B_{e}-coalgebra, since there is no appropriate natural transformation $r : R \to B_{\mathrm{e}}R$. To show this, we suppose that there is such a natural transformation, and derive a contradiction. The natural transformation $r : R \to B_{\mathrm{e}}R$ gives rise to an **I**-IL$_{\mathrm{e}}$TS $(R, \longrightarrow_r)$ satisfying Axioms **I**1–**I**6, and also satisfying properties 1–3 of Lemma 3.3.12. Property 2 ensures that

$$\emptyset \vdash (\mathsf{inl}\,\mathrm{p}, \mathsf{inl}\,\mathrm{p}) \xrightarrow{\tau}_r \emptyset \vdash (\mathsf{inl}\,\mathrm{p}_1, \mathsf{inl}\,\mathrm{p}_1) \quad .$$

By Axiom **I**5, we must have

$$C \vdash (\mathrm{p},\mathrm{p}) \xrightarrow{\tau}_r C \vdash (\mathrm{p}_1,\mathrm{p}_1)$$

for any non-empty set C, and by Axiom **I**6, we must have $z \in R(\emptyset)$ such that

$$[\emptyset \hookrightarrow C]z = (\mathrm{p}_1,\mathrm{p}_1) \qquad \text{and} \qquad \emptyset \vdash (\mathsf{inr}\,\mathrm{p}, \mathsf{inr}\,\mathrm{p}) \xrightarrow{\tau}_r \emptyset \vdash z \quad .$$

From the definition of R, we we can only have $z = (\mathsf{inl}\,\mathrm{p}_1, \mathsf{inl}\,\mathrm{p}_1)$, and so

$$\emptyset \vdash (\mathsf{inr}\,\mathrm{p}, \mathsf{inr}\,\mathrm{p}) \xrightarrow{\tau}_r \emptyset \vdash (\mathsf{inl}\,\mathrm{p}_1, \mathsf{inl}\,\mathrm{p}_1) \quad .$$

But this violates property 1 of Lemma 3.3.12 — a contradiction.

The counterexample that we have presented here is a little artificial. Central to the example is the property that, in the presheaf of states that is considered, the action of injections is not always injective. We will argue in the next chapter that this is an unnatural property of a state space.

A closure operator. A peculiar aspect in the above example is that the relation R contains the pair $(\mathrm{p}_1, \mathrm{p}_1)$ at all non-empty contexts, but does not contain $(\mathsf{inr}\,\mathrm{p}_1, \mathsf{inr}\,\mathrm{p}_1)$ at the empty context. We now consider the implications of *closing* the relation in this way.

Given any presheaves $P, Q \in \mathbf{Set}^{\mathbf{I}}$, such that $Q \subseteq P$, we let $\bar{Q} \in \mathbf{Set}^{\mathbf{I}}$ be the subpresheaf of P that is given on objects $C \in \mathbf{I}$ by

$$\bar{Q}(C) = \left\{ p \in P(C) \;\middle|\; \exists D \supseteq C \text{ in } \mathbf{I}.\ P[C \hookrightarrow D](p) \in Q(D) \right\} \quad . \tag{3.3.14}$$

To see that $\bar{Q}$ is indeed a subfunctor of P, notice that for every span in $\mathbf{I}$ of the form $(C' \xleftarrow{\iota} C \hookrightarrow D)$ there is a superset D' of C' and an injection $\jmath : D \rightarrowtail D'$ such that the following diagram commutes in $\mathbf{I}$.

$$\begin{array}{ccc} C & \hookrightarrow & D \\ {\scriptstyle\iota}\downarrow & & \downarrow{\scriptstyle\jmath} \\ C' & \hookrightarrow & D' \end{array}$$

Note that the $\mathbf{I}$-indexed binary relation for early bisimilarity for the π-calculus, considered in Prop. 3.3.5, is closed in this way.

For the $\mathbf{I}$-indexed binary relation R considered in the example above, we have

$$\bar{R}(\emptyset) = \left\{ \begin{array}{l} (\mathsf{inl}\,\mathrm{p}, \mathsf{inl}\,\mathrm{p}), (\mathsf{inl}\,\mathrm{p}_1, \mathsf{inl}\,\mathrm{p}_1), (\mathsf{inl}\,\mathrm{p}_1, \mathsf{inl}\,\mathrm{p}_2), (\mathsf{inl}\,\mathrm{p}_2, \mathsf{inl}\,\mathrm{p}_1), (\mathsf{inl}\,\mathrm{p}_2, \mathsf{inl}\,\mathrm{p}_2), \\ (\mathsf{inl}\,\mathrm{p}, \mathsf{inr}\,\mathrm{p}), (\mathsf{inl}\,\mathrm{p}_1, \mathsf{inr}\,\mathrm{p}_1), (\mathsf{inl}\,\mathrm{p}_1, \mathsf{inr}\,\mathrm{p}_2), (\mathsf{inl}\,\mathrm{p}_2, \mathsf{inr}\,\mathrm{p}_1), (\mathsf{inl}\,\mathrm{p}_2, \mathsf{inr}\,\mathrm{p}_2) \\ (\mathsf{inr}\,\mathrm{p}, \mathsf{inl}\,\mathrm{p}), (\mathsf{inr}\,\mathrm{p}_1, \mathsf{inl}\,\mathrm{p}_1), (\mathsf{inr}\,\mathrm{p}_1, \mathsf{inl}\,\mathrm{p}_2), (\mathsf{inr}\,\mathrm{p}_2, \mathsf{inl}\,\mathrm{p}_1), (\mathsf{inr}\,\mathrm{p}_2, \mathsf{inl}\,\mathrm{p}_2), \\ (\mathsf{inr}\,\mathrm{p}, \mathsf{inr}\,\mathrm{p}), (\mathsf{inr}\,\mathrm{p}_1, \mathsf{inr}\,\mathrm{p}_1), (\mathsf{inr}\,\mathrm{p}_1, \mathsf{inr}\,\mathrm{p}_2), (\mathsf{inr}\,\mathrm{p}_2, \mathsf{inr}\,\mathrm{p}_1), (\mathsf{inr}\,\mathrm{p}_2, \mathsf{inr}\,\mathrm{p}_2) \end{array} \right\}$$

$$\bar{R}(C) = R(C) = \{(\mathrm{p}, \mathrm{p}), (\mathrm{p}_1, \mathrm{p}_1), (\mathrm{p}_1, \mathrm{p}_2), (\mathrm{p}_2, \mathrm{p}_1), (\mathrm{p}_2, \mathrm{p}_2)\} \qquad \text{for } C \neq \emptyset.$$

Applying the closure operator to an $\mathbf{I}$-indexed early bisimulation results in a new $\mathbf{I}$-indexed early bisimulation, as we now show.

Proposition 3.3.15. *Consider two* $\mathbf{I}$-IL$_\mathrm{e}$TS*s,* $(P, \xrightarrow[P]{})$ *and* $(Q, \xrightarrow[Q]{})$*, that both satisfy Axioms* **I**1–**I**6*. Let* R *be an* $\mathbf{I}$*-indexed binary relation between* P *and* Q*. (So* $R \subseteq P \times Q$*.) If* R *is an* $\mathbf{I}$*-indexed early bisimulation between* $(P, \xrightarrow[P]{})$ *and* $(Q, \xrightarrow[Q]{})$*, then so is* $\bar{R}$*.*

Proof. We assume that R is an $\mathbf{I}$-indexed early bisimulation between induced $\mathbf{I}$-IL$_\mathrm{e}$TSs $\xrightarrow[P]{}$ and $\xrightarrow[Q]{}$ that satisfy Axioms **I**1–**I**6. We will show that $\bar{R}$ is also an $\mathbf{I}$-indexed early bisimulation. For brevity, we will show that $\bar{R}$ is an $\mathbf{I}$-indexed early simulation; the opposite direction is symmetric.

Suppose that $(p, q) \in \bar{R}(C)$ and that we have a transition

$$C \vdash p \xrightarrow[P]{\ell} C' \vdash p' \quad .$$

We must show that there is a transition $C \vdash q \xrightarrow[Q]{\ell} C' \vdash q'$ such that $(p', q') \in \bar{R}(C')$.

For the case when $\ell = c!z$, with $z \notin C$, we proceed as follows. Axiom **I**1 ensures that $C' = C \cup \{z\}$. The closure operator ensures that there is $D \in \mathbf{I}$ such that the pair $(P \times Q)[C \hookrightarrow D](p, q)$ is in $R(D)$. We let $p_0 = P[C \hookrightarrow D](p)$ and let $q_0 = Q[C \hookrightarrow D](q)$. We pick a fresh name $z' \in (\mathcal{N} - D)$, and, by Axiom **I**3, we know that

$$C \vdash p \xrightarrow[P]{c!z'} C \cup \{z'\} \vdash [z'/z]p' \quad .$$

By Axiom **I**5 we have

$$D \vdash p_0 \xrightarrow[P]{c!z'} D \cup \{z'\} \vdash [C \cup \{z'\} \hookrightarrow D \cup \{z'\}][z'/z]p' \quad .$$

Since R is an **I**-indexed early bisimulation, we have an element $q_0' \in Q(D \cup \{z'\})$ such that $([C \cup \{z'\} \hookrightarrow D \cup \{z'\}][$ is in $R(D \cup \{z'\})$ and

$$D \vdash q_0 \xrightarrow[Q]{c!z'} D \cup \{z'\} \vdash q_0' \quad .$$

Axiom **I**6 provides $q' \in Q(C \cup \{z'\})$ such that $[C \cup \{z'\} \hookrightarrow D \cup \{z'\}]q' = q_0'$ and such that

$$C \vdash q \xrightarrow[Q]{c!z'} C \cup \{z'\} \vdash q' \quad .$$

By definition of $\bar{R}$, we have that $([z'/z]p', q') \in \bar{R}(C \cup \{z'\})$. Functoriality of $\bar{R}$ ensures that $(p', [z/z']q') \in \bar{R}(C \cup \{z\})$, and Axiom **I**3 gives

$$C \vdash q \xrightarrow[Q]{c!z} C \cup \{z\} \vdash [z/z']q'$$

as required.

The cases for known output, input, and silent labels are similar. For known output, Axiom **I**4b is required, and for the input labels, Axiom **I**4a is required. Thus we can conclude that $\bar{R}$ is an **I**-indexed early simulation, and through a symmetric argument, an **I**-indexed early bisimulation. □

We now show that, if we restrict attention to those **I**-indexed binary relations that are closed, then the B_e-bisimulations are exactly the **I**-indexed early bisimulations. The reader should note that we do *not* assert that every B_e-bisimulation is closed, and so the problem of characterising coalgebraic bisimulations remains open.

Theorem 3.3.16. *Let (P,h) and (Q,k) be B_e-coalgebras, and consider an* **I***-indexed binary relation R between P and Q. If the closed relation $\bar{R}$ is an* **I***-indexed early bisimulation between the induced* **I**-IL$_e$TS*s $(P, \longrightarrow_h)$ and $(Q, \longrightarrow_k)$, then it is a B_e-bisimulation between B_e-coalgebras (P,h) and (Q,k).*

Proof. We suppose that $\bar{R}$ is an **I**-indexed early bisimulation between $(P, \longrightarrow_h)$ and $(Q, \longrightarrow_k)$, and show that it is a B_e-bisimulation between B_e-coalgebras (P,h) and (Q,k). To do this, we define an **I**-IL$_e$TS with carrier $\bar{R}$. The transition relation $\longrightarrow \subseteq \int \bar{R} \times Lab_e \times \int \bar{R}$ is the least such that for any $C, C' \in \mathbf{I}$, and any label $\ell \in Lab_e$, and any $(p,q) \in \bar{R}(C)$ and $(p',q') \in \bar{R}(C')$: if

$$C \vdash p \xrightarrow{\ell}_h C' \vdash p' \quad \text{and} \quad C \vdash q \xrightarrow{\ell}_k C' \vdash q'$$

then

$$C \vdash (p,q) \xrightarrow{\ell} C' \vdash (p',q') \quad .$$

We now show that this **I**-IL$_e$TS $(\bar{R}, \longrightarrow)$ satisfies Axioms **I**1–**I**6. Axiom **I**1 holds of $(\bar{R}, \longrightarrow)$ because it holds of $(P, \longrightarrow_h)$. Axiom **I**2 holds because it holds of $(P, \longrightarrow_h)$ and because $\bar{R}$ is an **I**-indexed early bisimulation.

Axioms **I**3 and **I**5 hold of $(\bar{R}, \longrightarrow)$ because R is functorial, and because Axioms **I**3 and **I**5 hold of $(P, \longrightarrow_h)$ and $(Q, \longrightarrow_k)$. Axioms **I**4a and **I**4b hold of $(\bar{R}, \longrightarrow)$ because they hold of $(P, \longrightarrow_h)$ and $(Q, \longrightarrow_k)$.

To see that Axiom **I**6 holds of $(\bar{R}, \longrightarrow)$: note that **I**6 holds of $(P, \longrightarrow_h)$ and $(Q, \longrightarrow_k)$, and **I**6 for $(\bar{R}, \longrightarrow)$ follows from the closure properties of $\bar{R}$. For instance, if we have a transition

$$D \vdash [C \hookrightarrow D](p,q) \xrightarrow{\tau} D \vdash (p',q')$$

then we must have

$$D \vdash [C \hookrightarrow D]p \xrightarrow{\tau}_h D \vdash p' \quad \text{and} \quad D \vdash [C \hookrightarrow D]q \xrightarrow{\tau}_k D \vdash q' \quad .$$

Axiom **I**6 for $(P, \longrightarrow_h)$ and $(Q, \longrightarrow_k)$ supplies elements $p'' \in P(C)$ and $q'' \in Q(C)$ such that $P[C \hookrightarrow D](p'') = p'$ and $Q[C \hookrightarrow D](q'') = q'$, and

$$C \vdash p \xrightarrow{\tau}_h D \vdash p'' \quad \text{and} \quad C \vdash q \xrightarrow{\tau}_k D \vdash q'' \quad .$$

By definition of the closure construction, $(p'', q'') \in \bar{R}(C)$, and so

$$C \vdash (p, q) \xrightarrow{\tau} C \vdash (p'', q'') \quad .$$

Thus the **I**-IL$_e$TS $(\bar{R}, \longrightarrow)$ satisfies Axioms **I**1–**I**6. By Theorem 3.3.7, we have a coalgebra structure $\bar{r} : \bar{R} \to B_e\bar{R}$ that induces the **I**-IL$_e$TS $(\bar{R}, \longrightarrow)$. In other words, $\longrightarrow = \longrightarrow_{\bar{r}}$.

To conclude that $\bar{R}$ is a B_e-bisimulation we appeal to Lemma 3.3.12. Property 1 of that lemma holds by definition, while Properties 2 and 3 hold since $\bar{R}$ is an **I**-indexed early bisimulation. □

The closure operation $(R \mapsto \bar{R})$ is inflationary, and so, by considering Theorem 3.3.16 together with Props. 3.3.13 and 3.3.15, we obtain the following result.

Corollary 3.3.17. *The final B_e-bisimulation between B_e-coalgebras (P, h) and (Q, k) is the greatest* **I**-*indexed early bisimulation between the* **I**-IL$_e$TS*s $(P, \longrightarrow_h)$ and $(Q, \longrightarrow_k)$.* □

3.3.4 Indexed ground labelled transition systems

Having developed a theory of indexed early labelled transition systems over the past three subsections, we now consider models of ground behaviour. For ground transition systems we will use the set of labels given by

$$Lab_g = \mathcal{N} \times \mathcal{N} \;+\; \mathcal{N} \times \mathcal{N} \;+\; \mathcal{N} \times \mathcal{N} \;+\; 1 \tag{3.3.18}$$

writing $c?(d)$, $c!d$, $c!(d)$, τ for elements of the first (bound input), second (output), third (bound output) and fourth (silent) summands respectively.

Notice that we distinguish bound and free outputs in ground labels. In this way our approach to ground labels differs from the approach to early labels (equation 3.3.1). One reason for this is that, with the ground systems, bound input and bound output are treated in the same way, and the distinction in the labels may help to clarify this.

We define channel and data functions $\mathsf{ch}, \mathsf{dat} : Lab_g \to \mathscr{P}(\mathcal{N})$ in much the same way as for early labels.

$\ell \in Lab_g$	$\mathsf{ch}(\ell)$	$\mathsf{dat}(\ell)$
$c?(d)$	$\{c\}$	$\{d\}$
$c!d$	$\{c\}$	$\{d\}$
$c!(d)$	$\{c\}$	$\{d\}$
τ	$\emptyset$	$\emptyset$

Transition systems. **I**-indexed ground labelled transition systems differ from **I**-IL$_e$TSs (Definition 3.3.2) only in that they use ground labels rather than early ones.

Definition 3.3.19. An **I**-*indexed ground labelled transition system* (**I**-IL$_g$TS) is a presheaf $P \in \mathbf{Set}^{\mathbf{I}}$ together with a transition relation $\longrightarrow \subseteq \int P \times Lab_g \times \int P$.

As in Section 3.1, we distinguish ground transitions from early transitions by using 'harpoon' arrows.

In (3.3.3) we described how a semantics for the π-calculus can be given in terms of an **I**-IL$_e$TS. A ground semantics for the π-calculus is provided by an **I**-IL$_g$TS in a similar manner, but this time using the ground transition relation $\xrightarrow[\pi]{}$ on π-calculus terms that was introduced in Figure 3.3. Again, the carrier is P_π, while the transition relation is the least relation

$$\xrightarrow[P_\pi]{} \subseteq \int P_\pi \times Lab_g \times \int P_\pi$$

satisfying the following implications.

$$\begin{array}{l}
\text{If } p \xrightarrow[\pi]{c(z)} p' \text{ and } \mathrm{fn}(p) \subseteq C \not\ni z \text{ then } C \vdash p \xrightarrow[P_\pi]{c?(z)} C \cup \{z\} \vdash p';\\
\text{If } p \xrightarrow[\pi]{\bar{c}d} p' \text{ and } \mathrm{fn}(p) \subseteq C \text{ then } C \vdash p \xrightarrow[P_\pi]{c!d} C \vdash p';\\
\text{If } p \xrightarrow[\pi]{\bar{c}(z)} p' \text{ and } \mathrm{fn}(p) \subseteq C \not\ni z \text{ then } C \vdash p \xrightarrow[P_\pi]{c!(z)} C \cup \{z\} \vdash p';\\
\text{If } p \xrightarrow[\pi]{\tau} p' \text{ and } \mathrm{fn}(p) \subseteq C \text{ then } C \vdash p \xrightarrow[P_\pi]{\tau} C \vdash p'.
\end{array} \tag{3.3.20}$$

Ground bisimulation. **I**-indexed ground bisimulation between **I**-IL$_g$TSs is defined in much the same way as **I**-indexed early bisimulation was defined between **I**-IL$_e$TSs in Definition 3.3.4; the only difference is that labels come from Lab_g instead of from Lab_e.

It is straightforward to prove a version of Prop. 3.3.5 for the ground case: a π-calculus term p is ground bisimilar with a term q (according to Definition 3.1.10(2)) if and only if there is an **I**-indexed ground bisimulation R on $(P_\pi, \xrightarrow[P_\pi]{})$ such that $(p,q) \in R(C)$.

From early to ground labelled transition systems. An abstract form of part of Prop. 3.1.9 is useful. For every **I**-IL$_e$TS $(P, \longrightarrow)$ We let $\overset{g}{\longrightarrow} \subseteq \int P \times Lab_g \times \int P$ be the least **I**-IL$_g$TS over P satisfying the following predicates.

$$\begin{array}{l}
\text{If } C \vdash p \xrightarrow{c?z} C' \vdash p' \text{ and } z \notin C \text{ then } C \vdash p \overset{c?(z)}{\longrightarrow} C' \vdash p'\\
\text{If } C \vdash p \xrightarrow{c!d} C' \vdash p' \text{ and } d \in C \text{ then } C \vdash p \overset{c!d}{\longrightarrow} C' \vdash p'\\
\text{If } C \vdash p \xrightarrow{c!z} C' \vdash p' \text{ and } z \notin C \text{ then } C \vdash p \overset{c!(z)}{\longrightarrow} C' \vdash p'\\
\text{If } C \vdash p \xrightarrow{\tau} C' \vdash p' \text{ then } C \vdash p \overset{\tau}{\longrightarrow} C' \vdash p'
\end{array} \tag{3.3.21}$$

The **I**-IL$_g$TS model of the π-calculus $(P_\pi, \xrightarrow[P_\pi]{})$, defined in (3.3.20), is precisely the **I**-IL$_g$TS induced via (3.3.21) from the **I**-IL$_e$TS model $(P_\pi, \xrightarrow[P_\pi]{})$, of (3.3.10).

I-indexed ground labelled transition systems from B_g-coalgebras. An **I**-IL$_g$TS is induced by a B_g-coalgebra in a similar way to the induction of an **I**-IL$_e$TS from a B_g-coalgebra (see $(B_e{\to}\mathbf{I})$-1 – 4 on page 62). Given a B_g-coalgebra $(P, h : P \to B_g P)$ we define $\longrightarrow_h \subseteq \int P \times Lab_g \times \int P$ to be the least relation satisfying the following statements $(B_g{\to}\mathbf{I})$-1 – 4.

$(B_g{\to}\mathbf{I})$-1. For any $C \subseteq_f \mathcal{N}$, $p \in P(C)$, $c \in C$, $z \in (\mathcal{N} - C)$, and $p' \in P(C \cup \{z\})$:
if $\pi_{\mathsf{binp}}(h_C(p)) \downarrow c$ and $p' \in \big(\pi_{\mathsf{binp}}(h_C(p))(c)\big)(z)$
then $C \vdash p \xrightarrow{c?(z)}_h C \cup \{z\} \vdash p'$.

$(B_g{\to}\mathbf{I})$-2. For any $C \subseteq_f \mathcal{N}$, $p \in P(C)$, $c, d \in C$, and $p' \in P(C)$:
if $\pi_{\mathsf{out}}(h_C(p)) \downarrow (c,d)$ and $p' \in \pi_{\mathsf{out}}(h_C(p))(c,d)$
then $C \vdash p \xrightarrow{c!d}_h C \vdash p'$.

$(B_g{\to}\mathbf{I})$-3. For any $C \subseteq_f \mathcal{N}$, $p \in P(C)$, $c \in C$, $z \in (\mathcal{N} - C)$, and $p' \in P(C \cup \{z\})$:
if $\pi_{\mathsf{bout}}(h_C(p)) \downarrow c$ and $p' \in (\pi_{\mathsf{bout}}(h_C(p))(c))(z)$
then $C \vdash p \xrightarrow{c!(z)}_h C \cup \{z\} \vdash p$.

$(B_g{\to}\mathbf{I})$-4. For any $C \subseteq_f \mathcal{N}$, $p \in P(C)$, and $p' \in P(C)$:
if $\pi_{\mathsf{tau}}(h_C(p)) \downarrow *$ and $p' \in \pi_{\mathsf{tau}}(h_C(p))(*)$
then $C \vdash p \xrightarrow{\tau}_h C \vdash p'$.

In summary, we have the following situation. (Arrows are labelled with references to where the maps are defined.)

$$\begin{array}{ccc} B_e\text{-coalgebras} & \xrightarrow{(3.2.14)} & B_g\text{-coalgebras} \\ {\scriptstyle (B_e{\to}\mathbf{I})\text{-}1\text{–}4}\downarrow & & \downarrow{\scriptstyle (B_g{\to}\mathbf{I})\text{-}1\text{–}4} \\ \mathbf{I}\text{-IL}_e\text{TSs} & \xrightarrow[(3.3.21)]{} & \mathbf{I}\text{-IL}_g\text{TSs} \end{array} \tag{3.3.22}$$

It is straightforward to verify that the diagram commutes.

Coalgebraic bisimulation versus transition system bisimulation. It appears that the anomaly that was discussed in Section 3.3.3 (with reference to early bisimulation) will also arise in the context of ground bisimulation. It seems that the treatment presented there, involving the closure operator, will be appropriate in the ground setting too.

3.4 Arbitrary substitutions and uniform input

For certain aspects of the theory of name-passing it is essential to consider arbitrary substitutions, and not just the injective ones. To this end it is necessary to require more structure in our state spaces than was allowed in the previous section. Indeed, this is always reasonable if the states are associated with some syntax, as we will see in (3.4.1) and in Section 7.4.

We begin, in Section 3.4.1, by revisiting the labelled transition system and coalgebraic models for name-passing, in the context of all substitutions. The early (resp. ground) labelled transition systems in this context we call $\mathbf{F}$-IL$_e$TSs (resp. $\mathbf{F}$-IL$_g$TSs). For the coalgebraic models, the notion of structured coalgebra, introduced in Section 2.4, comes into play.

In Section 3.4.2, we investigate the notions of bisimulation that arise in these classes of model. For the model of the π-calculus, these notions are the wide open bisimulations considered in Definition 3.1.13. In Prop. 3.1.14 we asserted that, for wide open relations, the notions of early and ground bisimulation all coincide; we conclude Section 3.4.2 by rephrasing this result in the coalgebraic setting.

Inspired by this result, we proceed in Section 3.4.3 to derive $\mathbf{F}$-IL$_e$TSs from $\mathbf{F}$-IL$_g$TSs. The $\mathbf{F}$-IL$_e$TSs that are so derivable are precisely those that satisfy a uniformity condition on input behaviour. We conclude this section by showing that the $\mathbf{F}$-IL$_e$TSs that satisfy this uniformity condition are precisely the $\mathcal{N}$-LTSs introduced by Cattani and Sewell [2004]. Thus the $\mathcal{N}$-LTSs of Cattani and Sewell are given a coalgebraic foundation.

3.4.1 Coalgebras and indexed labelled transition systems

Presheaves. Let $\mathbf{F}$ be the category with objects given by finite subsets of $\mathcal{N}$, and morphisms by all functions between them. A covariant presheaf $X : \mathbf{F} \to \mathbf{Set}$ can be thought of as associating to each set of names C, a set $X(C)$ of states that may use these names. To each function $f : C \to D$ between name contexts is associated a function $Xf : X(C) \to X(D)$ intended to model substitution in states according to the function f.

Notation. The following notation generalises the notation for injections introduced in Section 3.2. For every set $C \subseteq_{\mathrm{f}} \mathcal{N}$, and all names $c, d \in \mathcal{N}$, the surjection $[d/c] : (C \cup \{c\}) \to ((C - \{c\}) \cup \{d\})$ acts as identity on $(C - \{c\})$ and maps c to d.

Example: presheaf over F for the π-calculus. The state space of the π-calculus was introduced in equation 3.2.1 as a presheaf over **I**. It makes sense to substitute in π-calculus terms using non-injective functions, so we are led to consider the following presheaf X_π over **F**. For $C \subseteq_{\mathrm{f}} \mathcal{N}$, let

$$X_\pi(C) = \left\{ p \;\middle|\; p \text{ is a } \pi\text{-calculus term and } \mathrm{fn}(p) \subseteq C \right\} \tag{3.4.1a}$$

while for any $C, D \subseteq_{\mathrm{f}} \mathcal{N}$, any function $f : C \to D$, and any term $p \in X_\pi(C)$:

$$X_\pi f(p) = [f]p \qquad \text{— } i.e. \text{ the substitution } f \text{ applied to } p. \tag{3.4.1b}$$

[The presheaf X_π is precisely the functor π introduced by Cattani and Sewell, 2004, Defn. 3.10, and is the initial algebra $T0$ of Fiore and Turi, 2001, eqn. 16.]

Relating presheaves over F with presheaves over I. Any presheaf on **F** can be considered as a presheaf on **I**: the obvious faithful identity-on-objects functor $j^{\mathbf{I}}_{\mathbf{F}} : \mathbf{I} \to \mathbf{F}$ induces by precomposition a faithful functor

$$U^{\mathbf{I}}_{\mathbf{F}} = (j^{\mathbf{I}}_{\mathbf{F}})^* : \mathbf{Set}^{\mathbf{F}} \to \mathbf{Set}^{\mathbf{I}}$$

given by forgetting non-injective actions. For instance, the presheaf P_π on **I** for the π-calculus (3.2.1) is related with the presheaf X_π on **F** (3.4.1) by the identity

$$P_\pi = U^{\mathbf{I}}_{\mathbf{F}} X_\pi \quad .$$

Via this equation, we can consider the early and ground coalgebras for the π-calculus (as introduced in (3.3.2)) as natural transformations

$$h_{\pi \mathrm{e}} : U^{\mathbf{I}}_{\mathbf{F}} X_\pi \to B_{\mathrm{e}} U^{\mathbf{I}}_{\mathbf{F}} X_\pi \tag{3.4.2a}$$
$$h_{\pi \mathrm{g}} : U^{\mathbf{I}}_{\mathbf{F}} X_\pi \to B_{\mathrm{g}} U^{\mathbf{I}}_{\mathbf{F}} X_\pi \quad . \tag{3.4.2b}$$

Thus we are led to consider $U^{\mathbf{I}}_{\mathbf{F}}$-structured $B_{\mathrm{e/g}}$-coalgebras.

An adjunction. The forgetful functor $U^{\mathbf{I}}_{\mathbf{F}} : \mathbf{Set}^{\mathbf{F}} \to \mathbf{Set}^{\mathbf{I}}$ is the inverse image of a geometric morphism $\mathbf{Set}^{\mathbf{I}} \to \mathbf{Set}^{\mathbf{F}}$, induced by the inclusion $j^{\mathbf{I}}_{\mathbf{F}} : \mathbf{I} \to \mathbf{F}$. So the direct image of this geometric morphism, $(j^{\mathbf{I}}_{\mathbf{F}})_* : \mathbf{Set}^{\mathbf{I}} \to \mathbf{Set}^{\mathbf{F}}$ provides $U^{\mathbf{I}}_{\mathbf{F}}$ with a right adjoint. We do not need an explicit description here. [A more detailed discussion is provided by Fiore and Turi, 2001, Sec. 1.3. They denote the direct image by $\langle N, - \rangle$.]

Notice that we have the situation described in Example 2.4.3(2), and as such there is an isomorphism of categories,

$$(U^{\mathbf{I}}_{\mathbf{F}}, B_{\mathrm{e}})\text{-Coalg} \cong ((j^{\mathbf{I}}_{\mathbf{F}})_* B_{\mathrm{e}} U^{\mathbf{I}}_{\mathbf{F}})\text{-Coalg} \quad . \tag{3.4.3}$$

So our study of $U^{\mathbf{I}}_{\mathbf{F}}$-structured $B_{\mathrm{e/g}}$-coalgebras can equivalently be seen as a study of the kinds of coalgebras considered by Fiore and Turi [2001, Sec. 3].

Indexed labelled transition systems. We now introduce notions of labelled transition system over elements of presheaves over **F**. The notions are very closely related to the **I**-indexed labelled transition systems of Definition 3.3.2.

Definition 3.4.4. An **F**-*indexed early labelled transition system* (**F**-IL$_e$TS) is a presheaf $X \in \mathbf{Set}^{\mathbf{F}}$ together with a transition relation $\longrightarrow \subseteq \int X \times Lab_e \times \int X$.

An **F**-*indexed ground labelled transition system* (**F**-IL$_g$TS) is a presheaf $X \in \mathbf{Set}^{\mathbf{F}}$ together with a transition relation $\longrightarrow \subseteq \int X \times Lab_g \times \int X$.

For any presheaf $X \in \mathbf{Set}^{\mathbf{F}}$ we have $\int X = \int U_{\mathbf{F}}^{\mathbf{I}} X$. So an **F**-IL$_e$TS (resp. **F**-IL$_g$TS) with carrier X is the same thing as an **I**-IL$_e$TS (resp. **I**-IL$_g$TS) with carrier $U_{\mathbf{F}}^{\mathbf{I}} X$. For instance, the **I**-IL$_e$TS introduced for the π-calculus in (3.3.3) can be considered as an **F**-IL$_e$TSs with carrier X_π. In this way Axioms **I**1–**I**6 (Figure 3.4) on **I**-IL$_e$TSs can be considered as axioms on **F**-IL$_e$TSs. Recalling Theorem 3.3.8, we note that

> The class of $U_{\mathbf{F}}^{\mathbf{I}}$-structured B_e-coalgebras is in bijective correspondence with the class of those **F**-IL$_e$TSs that satisfy Axioms **I**1–**I**6.

3.4.2 Bisimulation

We now introduce notions of bisimulation on **F**-IL$_e$TSs and **F**-IL$_g$TSs, and compare these notions with $U_{\mathbf{F}}^{\mathbf{I}}$-structured B_e-bisimulation and $U_{\mathbf{F}}^{\mathbf{I}}$-structured B_g-bisimulation.

Indexed bisimulation.

Definition 3.4.5.

1. Consider presheaves $X, Y \in \mathbf{Set}^{\mathbf{F}}$. An **F**-*indexed binary relation* R between X and Y is a presheaf $R \in \mathbf{Set}^{\mathbf{F}}$ that is a subobject of $X \times Y$. Thus an **F**-indexed binary relation R is an **I**-indexed binary relation between $U_{\mathbf{F}}^{\mathbf{I}} X$ and $U_{\mathbf{F}}^{\mathbf{I}} Y$ such that for any function $f : C \to D$ in **F**, and any $(x, y) \in R(C)$, we have $([f]x, [f]y) \in R(D)$.

2. An **F**-*indexed early bisimulation* between **F**-IL$_e$TSs $(X, \xrightarrow[X]{})$ and $(Y, \xrightarrow[Y]{})$ is an **F**-indexed binary relation R between X and Y such that $U_{\mathbf{F}}^{\mathbf{I}} R$ is an **I**-indexed early bisimulation between $(U_{\mathbf{F}}^{\mathbf{I}} X, \xrightarrow[X]{})$ and $(U_{\mathbf{F}}^{\mathbf{I}} Y, \xrightarrow[Y]{})$.

3. An **F**-*indexed ground bisimulation* between **F**-IL$_g$TSs $(X, \xrightarrow[X]{})$ and $(Y, \xrightarrow[Y]{})$ is an **F**-indexed binary relation R between X and Y such that $U_{\mathbf{F}}^{\mathbf{I}} R$ is an **I**-indexed ground bisimulation between $(U_{\mathbf{F}}^{\mathbf{I}} X, \xrightarrow[X]{})$ and $(U_{\mathbf{F}}^{\mathbf{I}} Y, \xrightarrow[Y]{})$.

Wide open bisimulation for the π-calculus. The notion of **F**-indexed binary relation can be thought of as an abstract form of the notion of wide open relation on π-calculus terms introduced in Definition 3.1.13: an **F**-indexed binary relation is an indexed relation that is closed under all substitutions. The following result follows straightforwardly from Prop. 3.1.14 and Prop. 3.3.5.

Proposition 3.4.6.
Consider two π-calculus terms, p, q, with $\mathsf{fn}(p) \cup \mathsf{fn}(q) \subseteq C$. The following are equivalent.

1. *p is wide open bisimilar with q.*
2. *There is an **F**-indexed early bisimulation R on $(X_\pi, \xrightarrow[X_\pi]{})$ such that $(p, q) \in R(C)$.*
3. *There is an **F**-indexed ground bisimulation R on $(X_\pi, \xrightarrow[X_\pi]{})$ such that $(p, q) \in R(C)$.* □

The equivalence of items 1 and 2 has also been established by Cattani and Sewell [2004, Thm. 4.5].

Coalgebraic bisimulation. We now relate the notions of **F**-indexed bisimulation with the coalgebraic notions of structured bisimulation. We begin by defining a closure operator on subpresheaves in $\mathbf{Set}^{\mathbf{F}}$, analogous to that defined in equation 3.3.14 for subpresheaves in $\mathbf{Set}^{\mathbf{I}}$. Given any presheaves $X, Y \in \mathbf{Set}^{\mathbf{F}}$ such that $Y \subseteq X$, we let $\bar{Y} \in \mathbf{Set}^{\mathbf{F}}$ be the subpresheaf of X that is given on objects $C \in \mathbf{F}$ by

$$\bar{Y}(C) = \{x \in X(C) \mid \exists D \supseteq C \text{ in } \mathbf{F}.\ X[C \hookrightarrow D](x) \in Y(D)\} \quad . \tag{3.4.7}$$

To see that $\bar{Y}$ is indeed a subfunctor of X, the important thing to note is that for every span in **F** of the form $(C' \xleftarrow{f} C \hookrightarrow D)$ there is a superset D' of C' and a function $g : D \to D'$ such that the following diagram commutes in **F**.

$$\begin{array}{ccc} C & \hookrightarrow & D \\ {\scriptstyle f}\downarrow & & \downarrow{\scriptstyle g} \\ C' & \hookrightarrow & D' \end{array}$$

Note that for any subpresheaf of a presheaf in $\mathbf{Set}^{\mathbf{F}}$, if one first applies the closure operator of (3.4.7), and then the functor $U^{\mathbf{I}}_{\mathbf{F}} : \mathbf{Set}^{\mathbf{F}} \to \mathbf{Set}^{\mathbf{I}}$, the result is the same as if one first applies the functor $U^{\mathbf{I}}_{\mathbf{F}} : \mathbf{Set}^{\mathbf{F}} \to \mathbf{Set}^{\mathbf{I}}$ and then the closure operator of (3.3.14). That is, whenever we have $Y \subseteq X$ in $\mathbf{Set}^{\mathbf{F}}$, then $U^{\mathbf{I}}_{\mathbf{F}}\overline{Y} = \overline{U^{\mathbf{I}}_{\mathbf{F}}Y}$.

We highlight the following result, which follows immediately from Prop. 3.3.13 and Theorem 3.3.16. It is related to an assertion of Fiore and Turi [2001, Prop. 3.1].

Proposition 3.4.8. *Consider two $U^{\mathbf{I}}_{\mathbf{F}}$-structured B_{e}-coalgebras, (X,h) and (Y,k). Let R be an* **F***-indexed binary relation between X and Y.*

1. *If R is a $U^{\mathbf{I}}_{\mathbf{F}}$-structured B_{e}-bisimulation between (X,h) and (Y,k) then it is also an* **F***-indexed early bisimulation between induced* **F**-IL$_{\mathrm{e}}$TS*s $(X, \longrightarrow_h)$ and $(Y, \longrightarrow_k)$.*
2. *If R is an* **F***-indexed early bisimulation between induced* **F**-IL$_{\mathrm{e}}$TS*s $(X, \longrightarrow_h)$ and $(Y, \longrightarrow_k)$ then $\bar{R}$ is a $U^{\mathbf{I}}_{\mathbf{F}}$-structured B_{e}-bisimulation between (X,h) and (Y,k).* □

Wide open early = wide open ground, revisited. In Prop. 3.1.14 we asserted that, for wide open relations on π-calculus terms, all notions of bisimulation coincide. We now provide an abstract form of this result.

Theorem 3.4.9. *An* **F***-indexed binary relation is a final $U^{\mathbf{I}}_{\mathbf{F}}$-structured B_{g}-bisimulation if and only if it is a final $U^{\mathbf{I}}_{\mathbf{F}}$-structured B_{e}-bisimulation.*

Proof. To prove this result, we will appeal to Theorem 2.5.7. Because of the isomorphism (3.4.3), it is sufficient to exhibit a retraction $((j^{\mathbf{I}}_{\mathbf{F}})_* B_{\mathrm{e}} U^{\mathbf{I}}_{\mathbf{F}})$-Coalg $\to$ $((j^{\mathbf{I}}_{\mathbf{F}})_* B_{\mathrm{g}} U^{\mathbf{I}}_{\mathbf{F}})$-Coalg.

Indeed, as discussed in Section 3.2.2, the natural transformation

$$r : [N \Rightarrow -] \to \delta(-)$$

of (3.2.5) induces a natural transformation $B_{\mathrm{e}} \to B_{\mathrm{g}}$ between endofunctors on $\mathbf{Set}^{\mathbf{I}}$, and hence also a natural transformation $((j^{\mathbf{I}}_{\mathbf{F}})_* B_{\mathrm{e}} U^{\mathbf{I}}_{\mathbf{F}}) \to ((j^{\mathbf{I}}_{\mathbf{F}})_* B_{\mathrm{g}} U^{\mathbf{I}}_{\mathbf{F}})$ between endofunctors on $\mathbf{Set}^{\mathbf{F}}$. Using the ideas of Section 2.3.1, this natural transformation can be seen to induce a functor $((j^{\mathbf{I}}_{\mathbf{F}})_* B_{\mathrm{e}} U^{\mathbf{I}}_{\mathbf{F}})$-Coalg $\to$ $((j^{\mathbf{I}}_{\mathbf{F}})_* B_{\mathrm{g}} U^{\mathbf{I}}_{\mathbf{F}})$-Coalg between categories of coalgebras. It is this functor that we will show to be a retraction.

We will first describe a natural transformation $s : \delta U^{\mathbf{I}}_{\mathbf{F}}(-) \to [N \Rightarrow U^{\mathbf{I}}_{\mathbf{F}}(-)]$ between functors $\mathbf{Set}^{\mathbf{F}} \to \mathbf{Set}^{\mathbf{I}}$; then we will show that s is a section of

$$r U^{\mathbf{I}}_{\mathbf{F}} : [N \Rightarrow U^{\mathbf{I}}_{\mathbf{F}}(-)] \to \delta U^{\mathbf{I}}_{\mathbf{F}}(-) \quad .$$

For each presheaf $X \in \mathbf{Set}^{\mathbf{F}}$ and each $C \subseteq_{\mathrm{f}} \mathcal{N}$ we define $s_{X,C} : \delta U_{\mathbf{F}}^{\mathbf{I}} X(C) \to [N \Rightarrow U_{\mathbf{F}}^{\mathbf{I}} X](C)$ to be such that for any $\phi \in \delta U_{\mathbf{F}}^{\mathbf{I}} X(C)$, $z \in (\mathcal{N} - C)$, and any $c \in \mathcal{N}$,

$$s_{X,C}(\phi)(c) = X[c/z]\,(\phi(z)) \quad .$$

The uniformity condition in the definition of $\delta(-)$ ensures that this equation is independent of the choice of z, and so can be taken as a definition of s. It is routine to check that for each presheaf $X \in \mathbf{Set}^{\mathbf{F}}$ the family $\{s_{X,C}\}_{C \in \mathbf{I}}$ is natural, and then that the family $\{s_X\}_{X \in \mathbf{Set}^{\mathbf{F}}}$ of natural transformations is natural. It is equally straightforward to see that the composite

$$\delta U_{\mathbf{F}}^{\mathbf{I}}(-) \xrightarrow{s} [N \Rightarrow U_{\mathbf{F}}^{\mathbf{I}}(-)] \xrightarrow{rU_{\mathbf{F}}^{\mathbf{I}}} \delta U_{\mathbf{F}}^{\mathbf{I}}(-)$$

is the identity on $\delta U_{\mathbf{F}}^{\mathbf{I}} : \mathbf{Set}^{\mathbf{F}} \to \mathbf{Set}^{\mathbf{I}}$.

(As an aside, we note that the natural transformation $s : \delta U_{\mathbf{F}}^{\mathbf{I}}(-) \to [N \Rightarrow U_{\mathbf{F}}^{\mathbf{I}}(-)]$ has been used by Fiore and Turi [2001, Sec. 3] to give a coalgebraic semantics to the input operator of the π-calculus.)

Now, the pointwise non-empty powerset endofunctor $\mathscr{P}_{\mathrm{ne}}$ on $\mathbf{Set}^{\mathbf{I}}$ lifts along the forgetful functor $U_{\mathbf{F}}^{\mathbf{I}} : \mathbf{Set}^{\mathbf{F}} \to \mathbf{Set}^{\mathbf{I}}$. That is, we have an endofunctor $\mathscr{P}_{\mathrm{ne}}^{(\mathbf{F})}$ on $\mathbf{Set}^{\mathbf{F}}$ such that $U_{\mathbf{F}}^{\mathbf{I}} \mathscr{P}_{\mathrm{ne}}^{(\mathbf{F})} = \mathscr{P}_{\mathrm{ne}} U_{\mathbf{F}}^{\mathbf{I}}$. Thus the natural transformation

$$r\mathscr{P}_{\mathrm{ne}} U_{\mathbf{F}}^{\mathbf{I}} : [N \Rightarrow \mathscr{P}_{\mathrm{ne}} U_{\mathbf{F}}^{\mathbf{I}}] \to \delta \mathscr{P}_{\mathrm{ne}} U_{\mathbf{F}}^{\mathbf{I}}$$

also has a section, and so we have a retraction $B_{\mathrm{e}} U_{\mathbf{F}}^{\mathbf{I}} \twoheadrightarrow B_{\mathrm{g}} U_{\mathbf{F}}^{\mathbf{I}}$ in the functor category $[\mathbf{Set}^{\mathbf{F}}, \mathbf{Set}^{\mathbf{I}}]$. Applying the right adjoint $(j_{\mathbf{F}}^{\mathbf{I}})_* : \mathbf{Set}^{\mathbf{I}} \to \mathbf{Set}^{\mathbf{F}}$, we arrive at a retraction $((j_{\mathbf{F}}^{\mathbf{I}})_* B_{\mathrm{e}} U_{\mathbf{F}}^{\mathbf{I}}) \twoheadrightarrow ((j_{\mathbf{F}}^{\mathbf{I}})_* B_{\mathrm{g}} U_{\mathbf{F}}^{\mathbf{I}})$ in the category of endofunctors on $\mathbf{Set}^{\mathbf{F}}$.

Using the constructions of Section 2.3.1, we obtain a retraction between categories of coalgebras, $((j_{\mathbf{F}}^{\mathbf{I}})_* B_{\mathrm{e}} U_{\mathbf{F}}^{\mathbf{I}})$-Coalg $\twoheadrightarrow$ $((j_{\mathbf{F}}^{\mathbf{I}})_* B_{\mathrm{g}} U_{\mathbf{F}}^{\mathbf{I}})$-Coalg, and the result follows, using Theorem 2.5.7. □

3.4.3 Uniform input behaviour

In this subsection, we describe how $\mathbf{F}$-IL$_{\mathrm{g}}$TSs give rise to $\mathbf{F}$-IL$_{\mathrm{e}}$TSs. The $\mathbf{F}$-IL$_{\mathrm{e}}$TSs that arise in this way are shown to be those satisfying an additional axiom, Axiom **F2'** (Figure 3.5). In this way we arrive at a labelled transition system characterisation of the $U_{\mathbf{F}}^{\mathbf{I}}$-structured B_{g}-coalgebras. This axiom ensures that input behaviour is uniform across known and unknown values. We study the strength of Axiom **F2'**, showing that it implies certain of the Axiom **I1**–**I6** of Figure 3.4. Finally, we show that the $\mathbf{F}$-IL$_{\mathrm{e}}$TSs that satisfy **I1**–**I6** and **F2'** are precisely the $\mathcal{N}$-LTSs of Cattani and Sewell [2004]. Thus, combining the results of this section, we derive a coalgebraic foundation for $\mathcal{N}$-LTSs.

$\mathbf{F}$-IL$_{\mathrm{g}}$TSs form a subclass of the $\mathbf{F}$-IL$_{\mathrm{e}}$TSs. In our proof of Theorem 3.4.9, it is crucial that the category of $((j_{\mathbf{F}}^{\mathbf{I}})_* B_{\mathrm{g}} U_{\mathbf{F}}^{\mathbf{I}})$-coalgebras is essentially a split subcategory of the category of $((j_{\mathbf{F}}^{\mathbf{I}})_* B_{\mathrm{e}} U_{\mathbf{F}}^{\mathbf{I}})$-coalgebras. We now consider this relationship in a concrete way. In (3.3.21) it was explained how an $\mathbf{I}$-IL$_{\mathrm{e}}$TS induces an $\mathbf{I}$-IL$_{\mathrm{g}}$TS. Now, to each $\mathbf{F}$-IL$_{\mathrm{g}}$TS $(X, \longrightarrow)$ we associate an $\mathbf{F}$-IL$_{\mathrm{e}}$TS with the same carrier X and with transition relation

$$\overset{\mathrm{e}}{\longrightarrow} \subseteq \int X \times Lab_{\mathrm{e}} \times \int X$$

the least satisfying the following predicates.

$$\begin{aligned}
&\text{If } C \vdash p \xrightarrow{c?(z)} C' \cup \{z\} \vdash p' \text{ and } z \notin (C \cup C') \\
&\quad \text{then } C \vdash p \overset{\mathrm{e}}{\xrightarrow{c?d}} C' \cup \{d\} \vdash [d/z]p'. \\
&\text{If } C \vdash p \xrightarrow{c!d} C' \vdash p' \text{ then } C \vdash p \overset{\mathrm{e}}{\xrightarrow{c!d}} C' \vdash p'. \\
&\text{If } C \vdash p \xrightarrow{c!(z)} C' \vdash p' \text{ then } C \vdash p \overset{\mathrm{e}}{\xrightarrow{c!z}} C' \vdash p'. \\
&\text{If } C \vdash p \xrightarrow{\tau} C' \vdash p' \text{ then } C \vdash p \overset{\mathrm{e}}{\xrightarrow{\tau}} C' \vdash p'.
\end{aligned} \tag{3.4.10}$$

Thus we move between **F**-IL$_g$TSs and **F**-IL$_e$TSs. This can be seen as an abstract form of Prop. 3.1.9.

Observe that we have the following situation. (Compare with diagram 3.3.22.)

$$\begin{array}{ccc} U_{\mathbf{F}}^{\mathbf{I}}\text{-structured} \\ B_g\text{-coalgebras} & \xrightarrow{\text{thm. 3.4.9}} & U_{\mathbf{F}}^{\mathbf{I}}\text{-structured} \\ & & B_e\text{-coalgebras} \\ \Big\downarrow{\scriptstyle (B_g \to \mathbf{I})\text{-1–4}} & & \Big\downarrow{\scriptstyle (B_e \to \mathbf{I})\text{-1–4}} \\ \mathbf{F}\text{-IL}_g\text{TSs} & \xrightarrow[(3.4.10)]{} & \mathbf{F}\text{-IL}_e\text{TSs} \end{array} \tag{3.4.11}$$

(We do not claim that the mapping of (3.4.10) is inverse to the mapping of (3.3.21).)

Which F-IL$_e$TSs are F-IL$_g$TSs?. We now investigate the extent to which the **F**-IL$_g$TSs provide a generous model of name-passing. In Figure 3.5 we introduce Axiom **F**2' on **F**-IL$_e$TSs. It says that all input data is treated in a uniform way.

F2'. Input is determined by the input of fresh names:

$$C \vdash x \xrightarrow{c?d} C \cup \{d\} \vdash x'$$
$$\iff \exists z \in (\mathcal{N} - C),\ x'' \in X(C \cup \{z\}).\ [d/z]x'' = x' \ \wedge\ C \vdash x \xrightarrow{c?z} C \cup \{z\} \vdash x''$$

Figure 3.5: Axiom **F**2' on an **F**-indexed early labelled transition system over $X \in \mathbf{Set}^{\mathbf{F}}$, expressing that input behaviour treats all data in a uniform way.

We now justify this axiom by showing that, in the presence of Axiom **I**1, an **F**-IL$_e$TS satisfying Axiom **F**2' is the same thing as an **F**-IL$_e$TS that has been been induced from an **F**-IL$_g$TS.

Theorem 3.4.12.

1. *Every* **F**-IL$_e$TS *satisfying* **I**1 *and* **F**2' *is induced from an* **F**-IL$_g$TS. *Indeed, for any* **F**-IL$_e$TS $(X, \longrightarrow)$ *satisfying* **I**1 *and* **F**2'*, we have* $\longrightarrow\ =\ {}_{\text{eg}}\!\!\longrightarrow$.

2. *If the* **F**-IL$_e$TS *induced from an* **F**-IL$_g$TS *satisfies* **I**1 *then it also satisfies* **F**2'.

Proof. To show item (1), we consider an **F**-IL$_e$TS $(X, \longrightarrow)$ that satisfies Axioms **I**1 and **F**2'; we will show that $\longrightarrow\ =\ {}_{\text{eg}}\!\!\longrightarrow$.

Quick inspection of (3.3.21) and (3.4.10) leads us to the observation that ${}_{\text{eg}}\!\!\longrightarrow$ is the least relation satisfying

$$\begin{array}{l} \text{If } C \vdash x \xrightarrow{c?z} C' \cup \{z\} \vdash x' \text{ and } z \notin (C \cup C') \\ \quad \text{then } C \vdash x \ {}_{\text{eg}}\!\!\xrightarrow{c?d} C' \cup \{d\} \vdash [d/z]x' \\ \text{If } C \vdash x \xrightarrow{c!d} C' \vdash x' \text{ then } C \vdash x \ {}_{\text{eg}}\!\!\xrightarrow{c!d} C' \vdash x' \\ \text{If } C \vdash x \xrightarrow{\tau} C' \vdash x' \text{ then } C \vdash x \ {}_{\text{eg}}\!\!\xrightarrow{\tau} C' \vdash x'. \end{array} \tag{3.4.13}$$

It is clear that for output and silent labels $\ell \in Lab_e$,

$$C \vdash x \xrightarrow{\ell} C' \vdash x' \quad \text{iff} \quad C \vdash x \ {}_{\text{eg}}\!\!\xrightarrow{\ell} C' \vdash x' \qquad .$$

So it remains for us to show this correspondence for input labels. To see that $\longrightarrow \subseteq {}_{\text{eg}}\!\!\longrightarrow$, suppose that $C \vdash x \xrightarrow{c?d} C' \vdash x'$. By Axiom **I**1, $C' = C \cup \{d\}$. By Axiom **F**2', we have $z \in (\mathcal{N} - C)$

and $x'' \in X(C \cup \{z\})$ such that $[d/z]x'' = x'$ and $C \vdash x \xrightarrow{c?z} C \cup \{z\} \vdash x''$. So by (3.4.13), a transition $C \vdash x \xrightarrow{c?d}_{\text{eg}} C' \vdash x'$ is induced, as required.

Now, to see that $\xrightarrow{\text{eg}} \subseteq \longrightarrow$, suppose that $C \vdash x \xrightarrow{c?d}_{\text{eg}} C' \vdash x'$ is induced. Then, by (3.4.13) we must have $C'' \subseteq_{\text{f}} \mathcal{N}$, $z \in (\mathcal{N} - (C \cup C''))$ and $x'' \in X(C'' \cup \{z\})$ such that $C \vdash x \xrightarrow{c?z} C'' \cup \{z\} \vdash x''$ and $C' = C'' \cup \{d\}$ and $x' = [d/z]x''$. By Axiom **F**2' (right-to-left), $C \vdash x \xrightarrow{c?d} C' \vdash x'$ as required.

To show item (2), we consider an **F**-IL$_{\text{g}}$TS $(X, \longrightarrow)$ that induces, according to (3.4.10), an **F**-IL$_{\text{e}}$TS $(X, \xrightarrow{\text{e}})$ that satisfies Axiom **I**1. We will show that $(X, \xrightarrow{\text{e}})$ also satisfies Axiom **F**2'.

We show Axiom **F**2' from left to right. Suppose that

$$C \vdash x \xrightarrow{c?d}_{\text{e}} C \cup \{d\} \vdash x' \quad .$$

Then, following (3.4.10), we must have $C' \subseteq_{\text{f}} \mathcal{N}$, $z \in (\mathcal{N} - (C \cup C'))$ and $x'' \in X(C \cup \{z\})$ such that $C \vdash x \xrightarrow{c?(z)} C' \cup \{z\} \vdash x''$ and $(C' \cup \{d\}) = (C \cup \{d\})$ and $x' = [d/z]x''$. Indeed, (3.4.10) also induces $C \vdash x \xrightarrow{c?z}_{\text{e}} C' \cup \{z\} \vdash x''$, and, by Axiom **I**1, $(C' \cup \{z\}) = (C \cup \{z\})$.

The other direction of Axiom **F**2' is proved in a similar manner. Thus Theorem 3.4.12 is proved. □

From this result we have a labelled transition system characterisation of the U^{I}_{F}-structured B_{g}-coalgebras.

Corollary 3.4.14. *The following data are equivalent.*

1. *An* **F**-IL$_{\text{e}}$TS *that satisfies Axioms* **I**1–**I**6 *and* **F**2'.
2. *A U^{I}_{F}-structured B_{g}-coalgebra.*

Proof. We move from data (1) to data (2) by the process

$$\begin{array}{c} \textbf{F}\text{-IL}_{\text{e}}\text{TS with} \\ \textbf{I}1\text{–}\textbf{I}6,\ \textbf{F}2' \end{array} \xrightarrow{(\textbf{I}\to B_{\text{e}})\text{-}1\text{–}4} \begin{array}{c} U^{\text{I}}_{\text{F}}\text{-structured} \\ B_{\text{e}}\text{-coalgebra} \end{array} \xrightarrow{(3.2.14)} \begin{array}{c} U^{\text{I}}_{\text{F}}\text{-structured} \\ B_{\text{g}}\text{-coalgebra} \end{array} .$$

We move from data (2) to data (1) by the process

$$\begin{array}{c} U^{\text{I}}_{\text{F}}\text{-structured} \\ B_{\text{g}}\text{-coalgebra} \end{array} \xrightarrow{\text{thm. 3.4.9}} \begin{array}{c} U^{\text{I}}_{\text{F}}\text{-structured} \\ B_{\text{e}}\text{-coalgebra} \end{array} \xrightarrow{(B_{\text{e}}\to\textbf{I})\text{-}1\text{–}4} \begin{array}{c} \textbf{F}\text{-IL}_{\text{e}}\text{TS with} \\ \textbf{I}1\text{–}\textbf{I}6,\ \textbf{F}2' \end{array} .$$

To see that **I**1–**I**6 hold of the **F**-IL$_{\text{e}}$TS that results from the second process, consult Theorem 3.3.6; as for **F**2', consult diagram 3.4.11 with reference to Theorem 3.4.12.

We now show that moving from (1) to (2) to (1) again yields the original **F**-IL$_{\text{e}}$TS. It follows from diagrams 3.3.22 and 3.4.11 that the process (1)→(2)→(1) is the same as the process

$$\begin{array}{c} \textbf{F}\text{-IL}_{\text{e}}\text{TS} \\ \text{with } \textbf{I}1\text{–}\textbf{I}6,\ \textbf{F}2' \end{array} \xrightarrow{(3.3.21)} \textbf{F}\text{-IL}_{\text{g}}\text{TS} \xrightarrow{(3.4.10)} \textbf{F}\text{-IL}_{\text{e}}\text{TS}$$

that is, the process sending an **F**-IL$_{\text{e}}$TS $(X, \longrightarrow)$ that satisfies **I**1–**I**6, **F**2' to the **F**-IL$_{\text{e}}$TS$(X, \xrightarrow{\text{eg}})$. By Theorem 3.4.12, $\longrightarrow = \xrightarrow{\text{eg}}$, and we are done.

To see how moving from (2) to (1) to (2) again yields the original U^{I}_{F}-structured B_{g}-coalgebra, consider the composite mapping, and apply Theorem 3.3.8 followed by Theorem 3.4.9. □

Strength of Axiom F2'. Axiom **F**2' relates with the other axioms as follows.

Proposition 3.4.15. *For any* **F**-IL$_e$TS*:*

1. *Axiom* **F**2' *implies Axiom* **I**2*.*
2. *Axioms* **F**2'*,* **I**3*,* **I**5 *and* **I**6 *together imply Axiom* **I**4*a.*

Proof. We begin with item (1). Consider an **F**-IL$_e$TS $(X, \longrightarrow)$ of which Axiom **F**2' holds. Suppose that the premise of Axiom **I**2 holds, *i.e.*

$$C \vdash x \xrightarrow{c?d} C \cup \{d\} \vdash x' \quad .$$

Then, by Axiom **F**2', we have $z \in (\mathcal{N} - C)$ and $x'' \in X(C \cup \{z\})$ such that $[d/z]x'' = x'$ and $C \vdash x \xrightarrow{c?z} C \cup \{z\} \vdash x''$. Consider some other name $d' \in \mathcal{N}$. Applying Axiom **F**2' again, this time from right-to-left, we have

$$C \vdash x \xrightarrow{c?z} C \cup \{d'\} \vdash [d'/z]x''$$

and so x'' provides a witness for Axiom **I**2.

We turn now to item (2). Consider an **F**-IL$_e$TS $(X, \longrightarrow)$ of which Axioms **F**2', **I**3, **I**5 and **I**6 hold. We will prove **I**4a from left-to-right. Suppose that the left-hand-side holds, *i.e.*

$$C \vdash x \xrightarrow{c?z} C \cup \{z\} \vdash x' \quad .$$

We concentrate on the case $z \notin C$, otherwise the result is trivial. Pick some $z' \in (\mathcal{N} - (C \cup \{z\}))$. By **I**3, considering the bijection $[z'/z] : (C \cup z) \xrightarrow{\sim} (C \cup z')$, we have $C \vdash x \xrightarrow{c?z'} C \cup \{z'\} \vdash [z'/z]x'$. Now, by **I**5,

$$C \cup \{z\} \vdash [C \hookrightarrow C \cup \{z\}]x \xrightarrow{c?z'} C \cup \{z, z'\} \vdash [C \cup \{z'\} \hookrightarrow C \cup \{z, z'\}][z'/z]x'.$$

Finally, by **F**2',

$$C \cup \{z\} \vdash [C \hookrightarrow C \cup \{z\}]x \xrightarrow{c?z} C \cup \{z\} \vdash [z/z'][C \cup \{z'\} \hookrightarrow C \cup \{z, z'\}][z'/z]x'.$$

But the composite

$$C \cup \{z\} \xrightarrow{[z'/z]} C \cup \{z'\} \hookrightarrow C \cup \{z, z'\} \xrightarrow{[z/z']} C \cup \{z\}$$

is the identity map, and so the right-hand-side of **I**4a follows.

The converse of **I**4a is proved in a similar way, by applying Axiom **F**2' followed by Axiom **I**6 and Axiom **I**3. □

The transition systems of Cattani and Sewell. We now consider the model of name-passing suggested by Cattani and Sewell. According to Cattani and Sewell [2004, Defn. 3.4], an $\mathcal{N}$-LTS is an **F**-IL$_e$TS that satisfies Axioms $\mathcal{N}1$–$\mathcal{N}4$ in Figure 3.6. Axioms $\mathcal{N}1$, $\mathcal{N}3$ and $\mathcal{N}4$ only mention injective substitutions, and so Cattani and Sewell (in Sec. 7) define an $\mathcal{N}_{\text{inj}}$-LTS to be an **I**-IL$_e$TS that satisfies Axioms $\mathcal{N}1$, $\mathcal{N}3$ and $\mathcal{N}4$. (Cattani and Sewell work with transition systems that have distinguished initial states; we neglect this aspect here.) $\mathcal{N}$-LTSs and $\mathcal{N}_{\text{inj}}$-LTSs are related with the systems that we have introduced, as follows.

Theorem 3.4.16.

1. *An* **I**-IL$_e$TS *is an* $\mathcal{N}_{\text{inj}}$-LTS *if and only if it satisfies Axioms* **I**1 *and* **I**3–**I**6*.*
2. *An* **F**-IL$_e$TS *is an* $\mathcal{N}$-LTS *if and only if it satisfies Axioms* **I**1–**I**6 *and Axiom* **F**2'*.*

$\mathcal{N}1$. Naming:

$$C \vdash x \xrightarrow{\ell} C' \vdash x' \implies \mathsf{ch}(\ell) \subseteq C \wedge C' = C \cup \mathsf{n}(\ell)$$

$\mathcal{N}2$a. Input – new, *where* $z \notin C$:

$$C \vdash x \xrightarrow{c?z} C \cup \{z\} \vdash x' \implies \forall d \in C.\ C \vdash x \xrightarrow{c?d} C \vdash [d/z]x''$$

$\mathcal{N}2$b. Input – old:

$$C \vdash x \xrightarrow{c?d} C \vdash x' \implies \forall z \in (\mathcal{N} - C).\ \exists x''.\ C \vdash x \xrightarrow{c?z} C \cup \{z\} \vdash x'' \ \wedge\ x' = [d/z]x''$$

$\mathcal{N}3$a. Injective substitution, *for* $\iota : C \rightarrowtail D$, $\beta : (\mathsf{n}(\ell) - C) \xrightarrow{\sim} D'$ *with* $D \cap D' = \emptyset$:

$$C \vdash x \xrightarrow{\ell} C' \vdash x' \implies \exists D'', x''.\ D \vdash [\iota]x \xrightarrow{[\iota+\beta]\ell} D'' \vdash x''$$

$\mathcal{N}3$b. Shifting, $z \notin C$:

$$C \vdash x \xrightarrow{c?z} C \cup \{z\} \vdash x' \implies C \cup \{z\} \vdash [C \hookrightarrow C \cup \{z\}]x \xrightarrow{c?z} C \cup \{z\} \vdash x'$$

$\mathcal{N}4$. *For any injection* $\iota : C \rightarrowtail D$:

If $D \vdash [\iota]x \xrightarrow{\ell'} D' \vdash x''$
Then $D' = D \cup \mathsf{n}(\ell')$
and either: there exist ℓ, x'', $\beta : (\mathsf{n}(\ell) - C) \xrightarrow{\sim} (\mathsf{n}(\ell') - D)$
such that $\ell' = [\iota + \beta]\ell$
and $C \vdash x \xrightarrow{\ell} C \cup \mathsf{n}(\ell) \vdash x'$ and $x'' = [\iota + \beta]x'$
or: there exist $c \in C$, $d \in (D - \mathsf{im}(\iota))$, $z \in (\mathcal{N} - C)$ and x'
such that $\ell' = (\iota c)?d$ and $C \vdash x \xrightarrow{c?z} C \cup \{z\} \vdash x'$
and $x'' = [\iota, [d/z]]x'$

Figure 3.6: Requirements on an **F**-indexed labelled transition system as suggested by Cattani and Sewell [2004, Defn. 3.4], rewritten with our notation.

Proof. We begin by proving item (1). It is clear that Axiom $\mathcal{N}1$ is equivalent to Axiom **I**1. In the presence of Axioms $\mathcal{N}1$ and **I**1, Axiom $\mathcal{N}3$a is equivalent to Axioms **I**3 and **I**5 together. To see this, consider the decomposition of each injection into its image bijection and an inclusion map of the image into the codomain.

It remains for us to show how Axioms $\mathcal{N}1$–$\mathcal{N}4$ imply Axioms **I**4 and **I**6 and how Axioms **I**1–**I**6 imply Axioms $\mathcal{N}3$b and $\mathcal{N}4$.

First, we assume Axioms $\mathcal{N}1$, $\mathcal{N}3$ and $\mathcal{N}4$, and prove Axioms **I**4 and **I**6. First, Axiom **I**4a follows

immediately from $\mathcal{N}3$b and Cattani and Sewell's Proposition 3.8.

For Axiom **I**4b, we proceed as follows. Suppose that the premise of Axiom **I**4b holds, *i.e.* that

$$C \cup \{d\} \vdash [C \hookrightarrow C \cup \{d\}]x \xrightarrow{c!d} C' \vdash x' \quad .$$

We now regard Axiom $\mathcal{N}4$ with respect to the inclusion map $[C \hookrightarrow C \cup \{d\}]$. The first conclusion must hold for this case since the label is not an input label. So in particular we have a label ℓ together with a bijection

$$\beta : (\mathsf{n}(\ell) - C) \xrightarrow{\sim} (\{c, d\} - (C \cup \{d\}))$$

such that $\big[[C \hookrightarrow C \cup \{d\}], \beta\big]\,\ell = c!d$. By Axiom $\mathcal{N}1$, $c \in C \cup \{d\}$, so we know that $(\mathsf{n}(\ell) - C)$ is empty. So we have $\ell = c!d$ and consequently that $(\{c, d\} - C) = (\mathsf{n}(\ell) - C) = \emptyset$, and hence that $d \in C$, as required.

Axiom **I**6 is essentially the same as Cattani and Sewell's Proposition 3.7.

Finally, we assume that Axioms **I**1 and **I**3–**I**6 hold of an **I**-IL$_e$TS $(X, \longrightarrow)$ and prove that Axioms $\mathcal{N}3$b and $\mathcal{N}4$ hold of $(X, \longrightarrow)$. Axiom $\mathcal{N}3$b is part of Axiom **I**4a. As for Axiom $\mathcal{N}4$, we proceed as follows. Suppose that the premise of $\mathcal{N}4$ holds — *i.e.*, that we have some injection $\iota : C \to D$ and a name-context $D' \subseteq_{\mathrm{f}} \mathcal{N}$ together with $x \in X(C)$, $x'' \in X(D')$, such that

$$D \vdash [\iota]x \xrightarrow{\ell'} D' \vdash x'' \quad .$$

Then by Axiom **I**1 we have $D' = D \cup \{\mathsf{dat}\ell'\}$. We divide the proof according to the following cases.

1. $\mathsf{dat}(\ell) \subseteq \mathsf{im}(\iota)$.
2. $\mathsf{dat}(\ell) = \{z'\}$ for some $z' \notin D$.
3. $\mathsf{dat}(\ell) = \{d\}$ for some $d \in (D - \mathsf{im}(\iota))$.

In case (1) we will derive the first conclusion of Axiom $\mathcal{N}4$. Axiom **I**6 gives $x''' \in X(\iota(C))$ such that $[\iota(C) \hookrightarrow D]x''' = x''$ and

$$\iota(C) \vdash [\iota|_C]x \xrightarrow{\ell'} \iota(C) \vdash x'''$$

We let $\ell = \iota^{-1}(\ell')$ and we let $x' = [\iota|_C{}^{-1}]x'''$. By Axiom **I**3 we have

$$C \vdash x \xrightarrow{\ell} C \vdash x'$$

and we know that $[\iota]x = x''$ and $[\iota]\ell = \ell'$, as required.

In case (2) we will again derive the first conclusion of Axiom $\mathcal{N}4$. Axiom **I**6 ensures that there is $x''' \in X(\iota(C) \cup \{z'\})$ such that $[\iota(C) \cup \{z'\} \hookrightarrow D \cup \{z'\}]x''' = x''$ and

$$\iota(C) \vdash [\iota|_C]x \xrightarrow{\ell'} \iota(C) \cup \{z'\} \vdash x'''$$

Now, we pick $z \in (\mathcal{N} - C)$. We let $\beta : \{z\} \xrightarrow{\sim} \{z'\}$ be the unique such bijection, and we let $\ell = [\iota|_C{}^{-1} + \beta^{-1}](\ell')$ and let $x' = [\iota|_C{}^{-1} + \beta^{-1}]x'''$. By Axiom **I**3 we have

$$C \vdash x \xrightarrow{\ell} C \cup \{z\} \vdash x'$$

and, moreover, $[\iota + \beta]x' = x''$ and $[\iota + \beta]\ell = \ell'$, as required.

For case (3), we will derive the second conclusion of Axiom $\mathcal{N}4$. First, observe that, because of **I**4b, ℓ' cannot be an output label, and so ℓ' must be an input label — so we have $c' \in \mathcal{N}$ such that $\ell' = c'?d$. By Axiom **I**4a, we have

$$(D - \{d\}) \vdash [\iota(C) \hookrightarrow (D - \{d\})][\iota|_C]x \xrightarrow{c'?d} D \vdash x'' \quad .$$

Now, by Axiom **I**6, we have $x''' \in X(\iota(C) \cup \{d\})$ such that

$$\iota(C) \vdash [\iota|_C]x \xrightarrow{c'?d} \iota(C) \cup \{d\} \vdash x''' \quad .$$

By Axiom **I**1, we have that $c' \in \iota(C)$. We let $c = \iota^{-1}c'$, and we pick some $z \in (\mathcal{N} - C)$. Let $x' = [\iota|_C{}^{-1} + z/d]x'''$. By Axiom **I**3, we have

$$C \vdash x \xrightarrow{c?z} C \cup \{z\} \vdash x'$$

and, moreover, $[\iota, d/z]x' = x''$. Thus Axiom $\mathcal{N}4$ is satisfied, and so item (1) is proved.

Item (2) follows straightforwardly because it is clear that Axioms $\mathcal{N}2$a, $\mathcal{N}2$b are together equivalent to Axiom **F**2', while, by Prop. 3.4.15, Axiom **F**2' implies **I**2. □

Combining this result with Corollary 3.4.14, we can view $\mathcal{N}$-LTSs from a coalgebraic perspective.

Corollary 3.4.17. *The following data are equivalent.*

1. *An $\mathcal{N}$-LTS.*
2. *A $U^{\mathbf{I}}_{\mathbf{F}}$-structured B_g-coalgebra.* □

3.A Appendix to Chapter 3: Proofs of results in Section 3.3

Here, we give proofs for Theorems 3.3.6 and 3.3.7.

3.A.1 Preliminary result

The naturality of h plays a very important and rather subtle role in ensuring that behaviour is invariant under injective renamings. We record here the following properties.

Proposition 3.A.1. *Consider a presheaf $P \in \mathbf{Set}^{\mathbf{I}}$ and a family*

$$\{h_C : P(C) \to B_e P(C)\}_{C \in \mathbf{I}} \quad .$$

The family h is natural if and only if the following properties hold, for every $C \subseteq_f \mathcal{N}$, and any $p \in P(C)$:

1. *For any $D \subseteq_f \mathcal{N}$ such that $C \subseteq D$, and any $c, d \in D$,*

$$\begin{array}{lll}
\pi_{\mathsf{inp}}(h_D([C \hookrightarrow D]p)) \downarrow c & \textit{iff} & c \in C \textit{ and } \pi_{\mathsf{inp}}(h_C(p)) \downarrow c \\
\pi_{\mathsf{out}}(h_D([C \hookrightarrow D]p)) \downarrow (c,d) & \textit{iff} & c,d \in C \textit{ and } \pi_{\mathsf{out}}(h_C(p)) \downarrow (c,d) \\
\pi_{\mathsf{bout}}(h_D([C \hookrightarrow D]p)) \downarrow c & \textit{iff} & c \in C \textit{ and } \pi_{\mathsf{bout}}(h_C(p)) \downarrow c \\
\pi_{\mathsf{tau}}(h_D([C \hookrightarrow D]p)) \downarrow * & \textit{iff} & \pi_{\mathsf{tau}}(h_C(p)) \downarrow * \quad .
\end{array}$$

   When both sides are defined,

$$\begin{array}{rl}
\big(\pi_{\mathsf{inp}}(h_D([C \hookrightarrow D]p))\big)(c) &= [N \Rightarrow \mathscr{P}_{\mathrm{ne}}P][C \hookrightarrow D]\Big(\big(\pi_{\mathsf{inp}}(h_C(p))\big)(c)\Big) \\
\big(\pi_{\mathsf{out}}(h_D([C \hookrightarrow D]p))\big)(c,d) &= \mathscr{P}_{\mathrm{ne}}P[C \hookrightarrow D]\,\big(\big(\pi_{\mathsf{out}}(h_C(p))\big)(c,d)\big) \\
\big(\pi_{\mathsf{bout}}(h_D([C \hookrightarrow D]p))\big)(c) &= \delta\mathscr{P}_{\mathrm{ne}}P[C \hookrightarrow D]\,\big(\big(\pi_{\mathsf{bout}}(h_C(p))\big)(c)\big) \\
\big(\pi_{\mathsf{tau}}(h_D([C \hookrightarrow D]p))\big)(*) &= \mathscr{P}_{\mathrm{ne}}P[C \hookrightarrow D]\,\big(\big(\pi_{\mathsf{tau}}(h_C(p))\big)(*)\big) \quad .
\end{array}$$

2. *For any bijection $\beta : C \xrightarrow{\sim} D$, and any $c, d \in C$,*

$$\begin{array}{lll}
\pi_{\mathsf{inp}}(h_D([\beta]p)) \downarrow \beta(c) & \textit{iff} & \pi_{\mathsf{inp}}(h_C(p)) \downarrow c \\
\pi_{\mathsf{out}}(h_D([\beta]p)) \downarrow (\beta(c), \beta(d)) & \textit{iff} & \pi_{\mathsf{out}}(h_C(p)) \downarrow (c,d) \\
\pi_{\mathsf{bout}}(h_D([\beta]p)) \downarrow \beta(c) & \textit{iff} & \pi_{\mathsf{bout}}(h_C(p)) \downarrow c \\
\pi_{\mathsf{tau}}(h_D([\beta]p)) \downarrow * & \textit{iff} & \pi_{\mathsf{tau}}(h_C(p)) \downarrow * \quad .
\end{array}$$

   When both sides are defined,

$$\begin{array}{rl}
\big(\pi_{\mathsf{inp}}(h_D([\beta]p))\big)(\beta(c)) &= [N \Rightarrow \mathscr{P}_{\mathrm{ne}}P]\beta\,\Big(\big(\pi_{\mathsf{inp}}(h_C(p))\big)(c)\Big) \\
\big(\pi_{\mathsf{out}}(h_D([\beta]p))\big)(\beta(c),\beta(d)) &= \mathscr{P}_{\mathrm{ne}}P\beta\,\big(\big(\pi_{\mathsf{out}}(h_C(p))\big)(c,d)\big) \\
\big(\pi_{\mathsf{bout}}(h_D([\beta]p))\big)(\beta(c)) &= \delta\mathscr{P}_{\mathrm{ne}}P\beta\,\big(\big(\pi_{\mathsf{bout}}(h_C(p))\big)(c)\big) \\
\big(\pi_{\mathsf{tau}}(h_D([\beta]p))\big)(*) &= \mathscr{P}_{\mathrm{ne}}P\beta\,\big(\big(\pi_{\mathsf{tau}}(h_C(p))\big)(*)\big) \quad .
\end{array}$$

Proof notes. These results follow immediately from the actions of products and of partial exponentials. □

3.A.2 Proof of Theorem 3.3.6

We now provide a proof of Theorem 3.3.6.

Theorem 3.3.6. *For every B_e-coalgebra (P,h), the induced* **I**-IL$_e$TS $(P, \longrightarrow_h)$ *satisfies Axioms* **I**1–**I**6.

Proof. We must show that for any B_e-coalgebra (P,h), the transition system $(P, \longrightarrow_h)$ induced by $(B_e{\to}\mathbf{I})$-1–$(B_e{\to}\mathbf{I})$-4 satisfies Axioms **I**1–**I**6.

Axiom **I**1 follows directly from the definition.

Axiom **I**2 arises from the use of the exponential for the input component. Suppose that $C \vdash p \xrightarrow{c?d}_h C \vdash p'$ is induced. This transition must have been induced by $(B_e{\to}\mathbf{I})$-1. Thus we must have $\pi_{\mathsf{inp}}(h_C(p)) \downarrow c$ and $p' \in (\pi_{\mathsf{inp}}(h_C(p))(c))(d)$. Now, we know $(\pi_{\mathsf{inp}}(h_C(p))(c)) \in [N \Rightarrow \mathscr{P}_{\mathrm{ne}}P](C)$, while

$$[N \Rightarrow \mathscr{P}_{\mathrm{ne}}P](C) = \left\{ \phi \in \prod_{d\in\mathcal{N}} \mathscr{P}_{\mathrm{ne}}P(C\cup\{d\}) \;\middle|\; \forall z,z' \notin C.\ P[z'/z](\phi(z)) = \phi(z') \right\}.$$

So, for any other name $d' \in \mathcal{N}$, we have a state $p'' \in (\pi_{\mathsf{inp}}(h_C(p))(c))(d')$. By $(B_e{\to}\mathbf{I})$-1, the transition

$$C \vdash p \xrightarrow{c?d'}_h C \cup \{d'\} \vdash p''$$

is induced. So Axiom **I**2 is satisfied.

Axiom **I**3 holds because of the naturality of the coalgebra map; the uniform treatment of fresh names by the exponential and name generation structures is also important here. For instance, suppose that $C \vdash p \xrightarrow{c!z}_h C \cup \{z\} \vdash p'$ is induced, with $z \notin C$. This must have been induced by $(B_e{\to}\mathbf{I})$-3, so we must have

$$\pi_{\mathsf{bout}}(h_C(p)) \downarrow c \qquad \text{and} \qquad p' \in \big(\pi_{\mathsf{bout}}(h_C(p))(c)\big)(z).$$

Consider some bijection $\beta : C \cup \{z\} \xrightarrow{\sim} D$. Prop. 3.A.1, with regard to the bijection $\beta|_C : C \xrightarrow{\sim} \beta(C)$, gives

$$\pi_{\mathsf{bout}}(h_{\beta(C)}(P\beta|_C(p))) \downarrow (\beta(c)) \quad \text{and} \quad \pi_{\mathsf{bout}}(h_{\beta(C)}(P\beta|_C(p)))(\beta(c)) = (\delta\mathscr{P}_{\mathrm{ne}}(P))(\beta|_C)\big(\pi_{\mathsf{bout}}(h_C(p))(c)\big) \quad .$$

We know that $\beta(z) \notin \beta(C)$, so the action of $\delta(\mathscr{P}_{\mathrm{ne}}P))$ gives

$$P\beta(p') \in \big(\pi_{\mathsf{bout}}(h_{\beta(C)}(P\beta|_C(p)))(\beta(c))\big)(\beta(z)) \quad .$$

Thus, by $(B_e{\to}\mathbf{I})$-3, $\beta(C) \vdash [\beta|_C]p \xrightarrow{[\beta](c!z)}_h D \vdash [\beta]p'$ is induced. Other modes of communication are treated similarly; thus Axiom **I**3 is proved.

We turn now to Axiom **I**4a. We will prove the left-to-right part of this axiom; a proof of the converse is very similar. Suppose that

$$C \vdash p \xrightarrow{c?z}_h C \cup \{z\} \vdash p'$$

is induced. If $z \in C$, the axiom is trivial, so suppose that $z \notin C$. The transition must have been induced by $(B_e{\to}\mathbf{I})$-1, so we must have

$$\pi_{\mathsf{inp}}(h_C(p)) \downarrow c \qquad \text{and} \qquad p' \in \big(\pi_{\mathsf{inp}}(h_C(p))(c)\big)(z).$$

Since h is natural, Prop. 3.A.1 gives

$$\pi_{\mathsf{inp}}(h_{C\cup\{z\}}([C \hookrightarrow C\cup\{z\}]p)) \downarrow c$$

and

$$\pi_{\mathsf{inp}}(h_{C\cup\{z\}}([C \hookrightarrow C\cup\{z\}]p))(c) \;=\; [N \Rightarrow P][C \hookrightarrow C\cup\{z\}]\big(\pi_{\mathsf{inp}}(h_C(p))(c)\big) \quad .$$

From the action of the exponential and powerset we have

$$p' \in \big(\pi_{\mathsf{inp}}(h_{C\cup\{z\}}([C \hookrightarrow C\cup\{z\}]p))(c)\big)(z)$$

and so, by $(B_{\mathrm{e}}{\to}\mathbf{I})$-1, we have $C \vdash p \xrightarrow{c?z}_h C\cup\{z\} \vdash p'$ as required. Thus Axiom **I**4a is satisfied.

To prove Axiom **I**4b we proceed as follows. Suppose that the premise of the axiom holds, *i.e.*, $C\cup\{d\} \vdash [C \hookrightarrow C\cup\{d\}]p \xrightarrow{c!d}_h C\cup\{d\} \vdash p'$. This must have been induced by $(B_{\mathrm{e}}{\to}\mathbf{I})$-2. So we must have that

$$\pi_{\mathsf{out}}(h_{C\cup\{d\}}([C \hookrightarrow C\cup\{d\}]p)) \downarrow (c,d) \quad .$$

Prop. 3.A.1(1) asserts that $d \in C$, as required. So Axiom **I**4b holds.

We now turn to Axiom **I**5. We will concentrate on the case of input transitions. Suppose that $C \vdash p \xrightarrow{c?d}_h C\cup\{d\} \vdash p'$, and consider $D \subseteq_{\mathrm{f}} \mathcal{N}$ such that $C \subseteq D$ and with $d \notin (D - C)$. The transition must have been induced by $(B_{\mathrm{e}}{\to}\mathbf{I})$-1, so we must have that

$$\pi_{\mathsf{inp}}(h_C(p)) \downarrow c \qquad \text{and} \qquad p' \in \big(\pi_{\mathsf{inp}}(h_C(p))(c)\big)(d).$$

Prop. 3.A.1(1) gives us that $\pi_{\mathsf{inp}}\big(h_D([C \hookrightarrow D]p)\big) \downarrow c$ and

$$\pi_{\mathsf{inp}}(h_D([C \hookrightarrow D]p))(c) \;=\; [N \Rightarrow \mathscr{P}_{\mathrm{ne}}(P)][C \hookrightarrow D]\big(\pi_{\mathsf{inp}}(h_C(p))(c)\big) \quad .$$

Either $d \in C$, or $d \notin D$. For both these cases, the action of the exponential gives us

$$\big(\pi_{\mathsf{inp}}(h_D([C \hookrightarrow D]p))(c)\big)(d) \;=\; (\mathscr{P}_{\mathrm{ne}}P)[C\cup\{d\} \hookrightarrow D\cup\{d\}]\big(\big(\pi_{\mathsf{inp}}(h_C(p))(c)\big)(d)\big) \quad .$$

We know that $p' \in \big(\pi_{\mathsf{inp}}(h_C(p))(c)\big)(d)$. The action of the powerset is pointwise, so

$$[C\cup\{d\} \hookrightarrow D\cup\{d\}]p' \in \big(\pi_{\mathsf{inp}}(h_D([C \hookrightarrow D]p))(c)\big)(d) \quad .$$

So, by $(B_{\mathrm{e}}{\to}\mathbf{I})$-1, a transition

$$D \vdash [C \hookrightarrow D]p \xrightarrow{c?d}_h D\cup\{d\} \vdash [C\cup\{d\} \hookrightarrow D\cup\{d\}]p'$$

is induced, as required. Other modes of communication are treated similarly; thus Axiom **I**5 is proved.

Finally, we show that Axiom **I**6 holds of the induced transition system $\longrightarrow_h$. We will focus on the case of output transitions. Suppose, then, that

$$D \vdash [C \hookrightarrow D]p \xrightarrow{c!d}_h D\cup\{d\} \vdash p'$$

with $d \notin (D - C)$, as in the premise of Axiom **I**6.

For the case $d \in C$, the transition must have been induced by $(B_{\mathrm{e}}{\to}\mathbf{I})$-2, so we have

$$\pi_{\mathsf{out}}(h_D([C \hookrightarrow D]p)) \downarrow (c,d) \qquad \text{and} \qquad p' \in \pi_{\mathsf{out}}(h_D([C \hookrightarrow D]p))(c,d).$$

By Prop. 3.A.1(1), we have $c \in C$, $\pi_{\mathsf{out}}(h_C(p)) \downarrow (c,d)$, and

$$\big(\pi_{\mathsf{out}}(h_D([C \hookrightarrow D]p))\big)(c,d) \;=\; \mathscr{P}_{\mathrm{ne}}P[C \hookrightarrow D]\big(\pi_{\mathsf{out}}(h_C(p))(c,d)\big) \quad .$$

Now, since

$$p' \in \mathscr{P}_{\mathrm{ne}}P[C \hookrightarrow D]\big(\pi_{\mathsf{out}}(h_C(p))(c,d)\big)$$

the action of the powerset ensures that there exists a state $p'' \in \pi_{\mathsf{out}}(h_C(p))(c,d)$ with $[C \hookrightarrow D]p'' = p'$. By $(B_{\mathrm{e}}{\to}\mathbf{I})$-2, $C \vdash p \xrightarrow{c!d}_h C \vdash p''$ as required.

For the case of output where $d \notin D$, the transition must have been induced by $(B_e{\to}\mathbf{I})$-3, so we have

$$\pi_{\mathsf{bout}}(h_D([C \hookrightarrow D]p)) \downarrow c \qquad \text{and} \qquad p' \in \big(\pi_{\mathsf{bout}}(h_D([C \hookrightarrow D]p))(c)\big)(d).$$

By Prop. 3.A.1(1), we have $c \in C$, $\pi_{\mathsf{bout}}(h_C(p)) \downarrow c$, and

$$\big(\pi_{\mathsf{bout}}(h_D([C \hookrightarrow D]p))\big)(c) \;=\; (\delta\mathscr{P}_{\mathrm{ne}}P)[C \hookrightarrow D]\big(\pi_{\mathsf{bout}}(h_C(p))(c)\big) \quad .$$

Since $d \notin D$ we have $d \notin C$ and the action of $\delta\mathscr{P}_{\mathrm{ne}}P$ gives

$$\big(\pi_{\mathsf{bout}}(h_D([C \hookrightarrow D]p))(c)\big)(d) \;=\; (\mathscr{P}_{\mathrm{ne}}P)[C \cup \{d\} \hookrightarrow D \cup \{d\}]\Big(\big(\pi_{\mathsf{bout}}(h_C(p))(c)\big)(d)\Big) \quad .$$

So we have

$$p' \in (\mathscr{P}_{\mathrm{ne}}P)[C \cup \{d\} \hookrightarrow D \cup \{d\}]\Big(\big(\pi_{\mathsf{bout}}(h_C(p))(c)\big)(d)\Big) \quad .$$

The action of the powerset ensures that there exists $p'' \in \big(\pi_{\mathsf{bout}}(h_C(p))(c)\big)(d)$ which is such that $[C \cup \{d\} \hookrightarrow D \cup \{d\}]p'' = p'$. By $(B_e{\to}\mathbf{I})$-3, $C \vdash p \xrightarrow{c!d}_h C \cup \{d\} \vdash p''$ as required.

Other modes of communication are treated similarly; thus Axiom **I**6 is proved. Thus we have shown that the **I**-IL$_e$TS induced by a coalgebra satisfies Axioms **I**1 – **I**6, and Theorem 3.3.6 is proved. □

3.A.3 Proof of Theorem 3.3.7

We now provide a proof of Theorem 3.3.7.

Theorem 3.3.7. *For any* **I**-IL$_e$TS$\longrightarrow$ *that satisfies Axioms* **I**1–**I**6*, the induced family* $\big\{\overrightarrow{h}_C\big\}_C$ *is natural in* C.

Proof. We consider an **I**-IL$_e$TS that satisfies Axioms **I**1–**I**6, and show that the family

$$\big\{\overrightarrow{h}_C \colon P(C) \to B_e P(C)\big\}_{C \in \mathbf{I}}$$

induced according to $(\mathbf{I}{\to}B_e)$-1–4 is natural in C. For this, we use Prop. 3.A.1. We begin by proving naturality with respect to inclusion maps — that is, property (1) of Prop. 3.A.1. Here, we will focus on the input component of the coalgebras. We fix some $C, D \subseteq_f \mathscr{N}$ such that $C \subseteq D$, and consider $p \in P(C)$.

We begin by showing that if $\pi_{\mathsf{inp}}(\overrightarrow{h}_D([C \hookrightarrow D]p)) \downarrow c$ then $c \in C$ and $\pi_{\mathsf{inp}}(\overrightarrow{h}_C(p)) \downarrow c$, and also that for each $d \in \mathscr{N}$,

$$\Big(\pi_{\mathsf{inp}}(\overrightarrow{h}_D([C \hookrightarrow D]p))(c)\Big)(d) \;\subseteq\; (\mathscr{P}_{\mathrm{ne}}P)[C \hookrightarrow D]\Big(\Big(\pi_{\mathsf{inp}}(\overrightarrow{h}_C(p))(c)\Big)(d)\Big) \quad .$$

Suppose that $\pi_{\mathsf{inp}}(\overrightarrow{h}_D([C \hookrightarrow D]p)) \downarrow c$, and that for some $d \in \mathscr{N}$ there is $p' \in P(C \cup \{d\})$ such that $p' \in \Big(\pi_{\mathsf{inp}}(\overrightarrow{h}_D([C \hookrightarrow D]p))(c)\Big)(d)$. This must all have been induced by $(\mathbf{I}{\to}B_e)$-1, so we must have

$$D \vdash [C \hookrightarrow D]p \xrightarrow{c?d} D \cup \{d\} \vdash p' \quad .$$

Either $d \in C$, or $d \in (\mathscr{N} - D)$, or $d \in (D - C)$. If $d \in C$ or $d \in (\mathscr{N} - D)$, then by Axiom **I**6 we have $p'' \in P(C \cup \{d\})$ such that $p' = [C \cup \{d\} \hookrightarrow D \cup \{d\}]p''$ and

$$C \vdash p \xrightarrow{c?d} C \cup \{d\} \vdash p'' \quad .$$

If $d \in (D - C)$, then by Axiom **I**6 we have

$$C \cup \{d\} \vdash [C \hookrightarrow C \cup \{d\}]p \xrightarrow{c?d} C \cup \{d\} \vdash p''$$

and, by Axiom **I**4a,

$$C \vdash p \xrightarrow{c?d} C \cup \{d\} \vdash p'' \quad .$$

Now, no matter which subset of $\mathscr{N}$ contains d, by Axiom **I**1 we have $c \in C$, and $(\mathbf{I}{\rightarrow}B_{\mathrm{e}})$-1 ensures that $\pi_{\mathsf{inp}}(\overrightarrow{h}_C(p)) \downarrow c$, and that $p'' \in \left(\pi_{\mathsf{inp}}(\overrightarrow{h}_C(p))(c)\right)(d)$. So

$$\left(\pi_{\mathsf{inp}}(\overrightarrow{h}_D([C \hookrightarrow D]p))(c)\right)(d) \;\subseteq\; (\mathscr{P}_{\mathrm{ne}}P)[C \hookrightarrow D]\left(\left(\pi_{\mathsf{inp}}(\overrightarrow{h}_C(p))(c)\right)(d)\right)$$

as required.

We now consider the converse: we show that if $c \in C$ and $\pi_{\mathsf{inp}}(\overrightarrow{h}_C(p)) \downarrow c$ then we have $\pi_{\mathsf{inp}}(\overrightarrow{h}_D([C \hookrightarrow D]p)) \downarrow c$, and also that for each $d \in \mathscr{N}$,

$$(\mathscr{P}_{\mathrm{ne}}P)[C \hookrightarrow D]\left(\left(\pi_{\mathsf{inp}}(\overrightarrow{h}_C(p))(c)\right)(d)\right) \;\subseteq\; \left(\pi_{\mathsf{inp}}(\overrightarrow{h}_D([C \hookrightarrow D]p))(c)\right)(d) \quad .$$

Indeed suppose that $c \in C$ and $\pi_{\mathsf{inp}}(\overrightarrow{h}_C(p)) \downarrow c$ and suppose that for some $d \in \mathscr{N}$ there is $p' \in P(D \cup \{d\})$ such that

$$p' \in (\mathscr{P}_{\mathrm{ne}}P)[C \hookrightarrow D]\left(\left(\pi_{\mathsf{inp}}(\overrightarrow{h}_C(p))(c)\right)(d)\right) \quad .$$

Then we must have $p'' \in P(C \cup \{d\})$ such that $p' = [C \hookrightarrow D]p''$ and

$$p'' \in \left(\pi_{\mathsf{inp}}(\overrightarrow{h}_C(p)(c)\right)(d) \quad .$$

All this must have been induced by $(\mathbf{I}{\rightarrow}B_{\mathrm{e}})$-1 so we must have

$$C \vdash p \xrightarrow{c?d} C \cup \{d\} \vdash p'' \quad .$$

Now, either $d \in C$, or $d \in (\mathscr{N} - D)$, or $d \in (D - C)$. If $d \in C$ or $d \in (\mathscr{N} - D)$ then Axiom **I**5 gives

$$D \vdash [C \hookrightarrow D]p \xrightarrow{c?d} D \cup \{d\} \vdash p' \quad .$$

If $d \in (D - C)$ then Axiom **I**4a ensures that

$$C \cup \{d\} \vdash [C \hookrightarrow C \cup \{d\}]p \xrightarrow{c?d} C \cup \{d\} \vdash p''$$

and by Axiom **I**5 we have

$$D \vdash [C \hookrightarrow D]p \xrightarrow{c?d} D \vdash p' \quad .$$

So, no matter which subset of $\mathscr{N}$ contains d, $(\mathbf{I}{\rightarrow}B_{\mathrm{e}})$-1 ensures that $\pi_{\mathsf{inp}}(\overrightarrow{h}_D([C \hookrightarrow D]p)) \downarrow c$ and also that

$$(\mathscr{P}_{\mathrm{ne}}P)[C \hookrightarrow D]\left(\left(\pi_{\mathsf{inp}}(\overrightarrow{h}_C(p)(c)\right)(d)\right) \subseteq \left(\pi_{\mathsf{inp}}(\overrightarrow{h}_D([C \hookrightarrow D]p))(c)\right)(d) \quad .$$

The components for other modes of communication are seen to be natural with respect to inclusion maps in a similar manner. For output and bound output, Axiom **I**4b is required.

We now turn to show that the family $\left\{\overrightarrow{h}_C\right\}_C$ is natural in C with respect to bijections — this is property (2) of Prop. 3.A.1. This property involves only Axiom **I**3. For an example we concentrate on the bound output component. Consider some $C, D \subseteq_{\mathrm{f}} \mathscr{N}$, and some $p \in P(C)$, and a bijection $\beta : C \xrightarrow{\sim} D$. Because bijections are invertible, to prove the bound output aspect of property (2) of Prop. 3.A.1 it is sufficient to show that for any $c \in C$, whenever $\pi_{\mathsf{bout}}(\overrightarrow{h}_C(p)) \downarrow c$ we have that $\pi_{\mathsf{bout}}(\overrightarrow{h}_D([\beta]p)) \downarrow (\beta(c))$ and that for all $z \notin D$,

$$\left(\delta\mathscr{P}_{\mathrm{ne}}P\beta\left(\left(\pi_{\mathsf{bout}}(\overrightarrow{h}_C(p))\right)(c)\right)\right)(z) \subseteq \left(\pi_{\mathsf{bout}}(\overrightarrow{h}_D([\beta]p))\right)(c))(z)$$

Indeed, suppose that $\pi_{\mathsf{bout}}(\overrightarrow{h}_C(p)) \downarrow c$ and that

$$p' \in \left(\delta \mathscr{P}_{\mathrm{ne}} P \beta \left((\pi_{\mathsf{bout}}(\overrightarrow{h}_C(p)))(c)\right)\right)(z) \quad .$$

Pick some $z' \in (\mathcal{N} - C)$. The actions of δ and $\mathscr{P}_{\mathrm{ne}}$ ensure that there is $p'' \in \left((\pi_{\mathsf{bout}}(\overrightarrow{h}_C(p)))(c)\right)(z')$ such that $p' = [\beta, z/z']p''$. This must have been induced by $(\mathbf{I}{\rightarrow}B_{\mathrm{e}})$-3, so we must have that

$$C \vdash p \xrightarrow{c!z'} C \cup \{z'\} \vdash p'' \quad .$$

By Axiom **I**3, considering the bijection $[\beta, z/z'] : C \cup \{z'\} \xrightarrow{\sim} D \cup \{z\}$,

$$D \vdash [\beta]p \xrightarrow{\beta(c)!z} D \cup \{z\} \vdash p' \quad .$$

So, by $(\mathbf{I}{\rightarrow}B_{\mathrm{e}})$-3, $\pi_{\mathsf{bout}}(\overrightarrow{h}_D([\beta]p)) \downarrow (\beta(c))$ and

$$p' \in (\pi_{\mathsf{bout}}(\overrightarrow{h}_D([\beta]p)))(\beta(c)))(z) \quad .$$

Thus $\overrightarrow{h} \colon P \to B_{\mathrm{e}}P$ is seen to be natural, and Theorem 3.3.7 is proved. □

Chapter 4

Models for Name-Passing, Refined

In this chapter we investigate refinements of the models of Chapter 3 by imposing a sheaf condition on the presheaves that we consider as state spaces. We consider a coverage of the category $\mathbf{I}$ for which the separatedness condition for a presheaf $P \in \mathbf{Set}^{\mathbf{I}}$ enforces that the action of P on the injections of $\mathbf{I}$ is injective. The sheaf condition for this coverage further enforces that a presheaf P maps pullbacks in $\mathbf{I}$ to pullbacks of sets. We assert that this sheaf condition is reasonable, given our intuitions about presheaves as state spaces.

We recall some basic and general aspects of sheaf theory in Section 4.1. In Section 4.2 we introduce a coverage on the category $\mathbf{I}$ for which the category of sheaves is known as the Schanuel topos. We show that the problems with coalgebraic bisimulation (observed in 3.3.3) disappear when the state spaces are sheaves. We then establish that the behaviour endofunctors on $\mathbf{Set}^{\mathbf{I}}$ restrict to endofunctors on this sheaf subcategory.

In Section 4.3 we account for state spaces that admit all substitutions by identifying an appropriate sheaf subcategory of $\mathbf{Set}^{\mathbf{F}}$. In Section 4.4 we explore some benefits of restricting the model in this way, by working with certain minimal elements of presheaves. In doing this, we arrive at a simple axiomatisation of transition systems.

4.1 Preliminaries: Coverages on categories

We begin with some definitions from Grothendieck's sheaf theory. It is usual in sheaf theory to deal with contravariant functors, but for the purposes of this thesis it is sensible to work with covariant functors. For instance, the presheaves considered in the previous chapter were all presented as covariant functors. We take this opportunity to rewrite the usual definitions from this covariant perspective.

The exposition here is terse as it is intended merely to set notation and collect some results that will be important in our development. The interested reader should turn to the literature for examples and discussion, perhaps beginning with the books by Mac Lane and Moerdijk [1992] and by Johnstone [2002]. The terminology here is based on Johnstone's book [2002].

4.1.1 Coverages, sheaves, and separated presheaves

We begin by recalling the notion of coverage on a category. We follow this by providing notions of separated presheaves and sheaves for coverages. We conclude this subsection by considering coverages made of singleton families.

Coverages. Let $\mathbf{C}$ be a small category. A *coverage on* $\mathbf{C}$ is an assignment $\mathscr{T}$ from objects $C \in \mathbf{C}$ to collections $\mathscr{T}C$ of families of morphisms with domain C. The families in $\mathscr{T}C$ are called *$\mathscr{T}$-covers* of C. (The prefix $\mathscr{T}$ will be omitted when it is obvious.)

A coverage is *stable* if for every $\mathscr{T}$-cover T of C, and every morphism $f : C \to D$ in $\mathbf{C}$, there is a $\mathscr{T}$-cover T_f of D such that for every $g : D \to E$ in T_f, the composite $gf : C \to E$ factors through a morphism in T.

Sieves. For any object C of a small category $\mathbf{C}$, a *sieve on* C is a collection T of morphisms in $\mathbf{C}$ with domain C, such that if $f : C \to D$ is in T then for any $E \in \mathbf{C}$, and any morphism $g : D \to E$ in $\mathbf{C}$, the composite $gf : C \to E$ is in T.

Every collection T of morphisms in $\mathbf{C}$ with domain C generates a sieve on C by closure under postcomposition.

Presheaves that are separated or sheaves for a coverage. Let $\mathbf{C}$ be a small category. Consider a covariant presheaf $P : \mathbf{C} \to \mathbf{Set}$, and consider a coverage $\mathscr{T}$ on $\mathbf{C}$. We recall what it means for P to be $\mathscr{T}$-separated, and for P to be a $\mathscr{T}$-sheaf.

Definition 4.1.1. Let $\mathscr{T}$ be a coverage on a small category $\mathbf{C}$.

1. Consider an object $C \in \mathbf{C}$, and let T be a collection of morphisms in $\mathbf{C}$, all with domain C. A *compatible family* for T is given by assigning to each morphism $f : C \to D$ in T an element $p_f \in P(D)$, such that, for any two morphisms $f : C \to D$, $f' : C \to D'$ in T, and any object $E \in \mathbf{C}$, and any morphisms $g : D \to E$, $g' : D' \to E$, if $gf = g'f'$ then we have that $Pg(p_f) = Pg'(p_{f'})$.

2. For any object $C \in \mathbf{C}$, and any collection T of morphisms in $\mathbf{C}$ with domain C, an *amalgamation* of a compatible family $(p_f)_{f \in T}$ is an element $p \in P(C)$ such that for every morphism $f : C \to D$ in T we have $Pf(p) = p_f$.

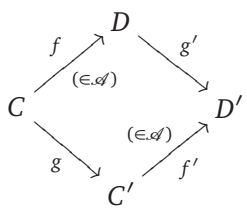
3. A presheaf is *$\mathscr{T}$-separated* if for every $C \in \mathbf{C}$, every compatible family for every $\mathscr{T}$-cover of C has at most one amalgamation.

4. A presheaf is a *$\mathscr{T}$-sheaf* if for every $C \in \mathbf{C}$, every compatible family for every $\mathscr{T}$-cover of C has a unique amalgamation.

Sites. A site $(\mathbf{C}, \mathscr{T})$ is a small category $\mathbf{C}$ equipped with a coverage $\mathscr{T}$. A sheaf for a site $(\mathbf{C}, \mathscr{T})$ is a presheaf on $\mathbf{C}$ that is a $\mathscr{T}$-sheaf. We write $\mathbf{Sep}(\mathbf{C}, \mathscr{T})$ for the full subcategory of $\mathbf{Set}^{\mathbf{C}}$ whose objects are $\mathscr{T}$-separated; we write $\mathbf{Sh}(\mathbf{C}, \mathscr{T})$ for the full subcategory of $\mathbf{Set}^{\mathbf{C}}$ whose objects are $\mathscr{T}$-sheaves. If $\mathscr{T}$ is stable then the category $\mathbf{Sh}(\mathbf{C}, \mathscr{T})$ is a Grothendieck topos [see *e.g.* Johnstone, 2002, Prop. C2.1.9 and Ex. C2.1.13].

Coverages made of singleton families. In this thesis the coverages that we will consider will either consist of singleton families of morphisms, or will be sifted coverages that are induced by such coverages. Fixing a category $\mathbf{C}$, we let $\mathscr{A}$ be a class of morphisms in $\mathbf{C}$; the morphisms in $\mathscr{A}$ will be called *$\mathscr{A}$-morphisms*. This class $\mathscr{A}$ generates a coverage $\mathscr{T}_{\mathscr{A}}$ on $\mathbf{C}$, for which the $\mathscr{T}_{\mathscr{A}}$-covers of an object $C \in \mathbf{C}$ are the singleton sets containing $\mathscr{A}$-morphisms with domain C.

It is straightforward to establish that the induced coverage $\mathscr{T}_{\mathscr{A}}$ is stable if and only if for every $\mathscr{A}$-morphism $f : C \to D$ and every morphism $g : C \to C'$ in $\mathbf{C}$, there is an $\mathscr{A}$-morphism $f' : C' \to D'$ and a morphism $g' : D \to D'$ in $\mathbf{C}$ such that the following diagram commutes.



4.1.2 Morphisms between sites

Recall that every functor $F : \mathbf{C} \to \mathbf{D}$ induces an essential geometric morphism $\mathbf{Set}^{\mathbf{C}} \to \mathbf{Set}^{\mathbf{D}}$. That is, we have a functor $F^* : \mathbf{Set}^{\mathbf{D}} \to \mathbf{Set}^{\mathbf{C}}$ given by precomposition with F, and this functor F^* always has a left adjoint $F_! : \mathbf{Set}^{\mathbf{C}} \to \mathbf{Set}^{\mathbf{D}}$ and a right adjoint $F_* : \mathbf{Set}^{\mathbf{C}} \to \mathbf{Set}^{\mathbf{D}}$.

When $\mathbf{C}$ and $\mathbf{D}$ are equipped with coverages, these functors do not always restrict to functors between categories of sheaves, or of separated presheaves. However, by imposing some conditions on the way that F respects the coverages, one can achieve some results in this direction.

Definition 4.1.2. Let $(\mathbf{C}, \mathscr{S})$ and $(\mathbf{D}, \mathscr{T})$ be sites, and let $F : \mathbf{C} \to \mathbf{D}$ be a functor.

1. We say that F *preserves covers* if, for every object $C \in \mathbf{C}$ and every $\mathscr{S}$-cover S of C, the sieve generated by the family $\left\{ Ff \;\middle|\; f \in S \right\}$ contains a $\mathscr{T}$-cover of FC.
2. We say that F *reflects covers* if, for every object $C \in \mathbf{C}$ and every $\mathscr{T}$-cover T of FC, there is an $\mathscr{S}$-cover S of C such that the family $\left\{ Ff \;\middle|\; f \in S \right\}$ is contained in T.

Inverse images. We record some properties of the inverse image functor $F^* : \mathbf{Set}^{\mathbf{D}} \to \mathbf{Set}^{\mathbf{C}}$. These three results follow straightforwardly from expanding the definitions.

Proposition 4.1.3. *Let $(\mathbf{C}, \mathscr{S})$ and $(\mathbf{D}, \mathscr{T})$ be sites, and let $F : \mathbf{C} \to \mathbf{D}$ be a functor. Consider a presheaf $P \in \mathbf{Set}^{\mathbf{D}}$.*

1. *If F preserves covers, and P is $\mathscr{T}$-separated, then F^*P is $\mathscr{S}$-separated.*
2. *If F reflects covers, and F^*P is $\mathscr{S}$-separated, then P is $\mathscr{T}$-separated.*
3. *If F reflects covers, and F^*P is an $\mathscr{S}$-sheaf, then P is $\mathscr{T}$-sheaf.* □

Note that Prop. 4.1.3 does *not* assert that the inverse image functor F^* maps $\mathscr{T}$-sheaves to $\mathscr{S}$-sheaves.

Direct images. We record a property of the direct image functor $F_* : \mathbf{Set}^{\mathbf{D}} \to \mathbf{Set}^{\mathbf{C}}$. This result is given by [Johnstone, 2002, Prop. C2.3.18], although he assumes some additional properties of the sites and the proof there is not direct. For these reasons we include a proof here.

Proposition 4.1.4. *Let $(\mathbf{C}, \mathscr{S})$ and $(\mathbf{D}, \mathscr{T})$ be sites, with $\mathscr{T}$ stable. For any cover reflecting functor $F : \mathbf{C} \to \mathbf{D}$, the direct image functor $F_* : \mathbf{Set}^{\mathbf{C}} \to \mathbf{Set}^{\mathbf{D}}$ restricts to a functor $F_* : \mathbf{Sh}(\mathbf{C}, \mathscr{S}) \to \mathbf{Sh}(\mathbf{D}, \mathscr{T})$.*

Proof. Let P be a presheaf on $\mathbf{C}$.

We begin by recalling that, by the Yoneda lemma, we must have for each $C \in \mathbf{D}$ that

$$F_*P(C) \cong \mathbf{Set}^{\mathbf{C}}(\mathbf{D}(C, F-), P)$$

and that this natural isomorphism is natural in $C \in \mathbf{D}$. Here, we will take this isomorphism as a *definition* of F_*P.

We will show that if P is an $\mathscr{S}$-sheaf then the presheaf F_*P is a $\mathscr{T}$-sheaf. To this end we let T be a $\mathscr{T}$-cover of an object D of $\mathbf{D}$, and consider a compatible family in F_*P

$$\left(\alpha_f : \mathbf{D}(E, F-) \to P \right)_{f : D \to E \in T} \tag{4.1.5}$$

for this cover. We define an amalgamation

$$\alpha : \mathbf{D}(D, F-) \to P$$

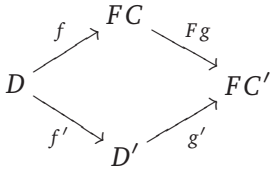
for this family as follows.

Consider an object $C \in \mathbf{C}$, and a morphism $f : D \to FC$ in $\mathbf{D}$. Since $\mathscr{T}$ is stable we have a $\mathscr{T}$-cover T_f of FC such that for every morphism $g : FC \to E$ the composite $gf : D \to E$ factors through a morphism in T. Because F reflects covers, and T_f is a $\mathscr{T}$-cover of FC, we must have an $\mathscr{S}$-cover S_f of C such that the family FS_f is contained in T_f.

We now create a compatible family for S_f in P,

$$\left(p_g^{(f)} \in P(C')\right)_{g:C\to C'\in S_f} \quad . \tag{4.1.6}$$

We do this as follows. Since F reflects covers, we know that for every morphism $g : C \to C'$ in S_f, there is a morphism $f' : D \to D'$ in T and a morphism $g' : D' \to FC'$ in $\mathbf{D}$ such that the following diagram commutes in $\mathbf{D}$.



Therefore we have a member $\alpha_{f'} : \mathbf{D}(D', F-) \to P$ of the compatible family (4.1.5), and we let $p_g^{(f)} = \alpha_{f'C'}(g')$. It follows from the compatibility of (4.1.5) that this definition of $p_g^{(f)}$ is independent of the choice of f' and g'. The resulting family $\left(p_g^{(f)} \in P(C')\right)_{g:C\to C'\in S_f}$ is compatible since (4.1.5) is compatible.

We define $\alpha_C(f) \in P(C)$ to be the amalgamation of this family (4.1.6). This exists, since P is a sheaf. It is routine to verify that the family of functions $\{\alpha_C\}_{C\in\mathbf{C}}$ defined in this way is natural. Thus we have an amalgamation for the family (4.1.5).

The uniqueness of this amalgamation α follows from the uniqueness of the amalgamations of families (of the form (4.1.6)) used to define α. □

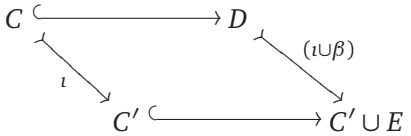
4.2 Models of name-passing in the Schanuel topos

In Sections 3.2 and 3.3 we studied models of name-passing for which the state spaces were organised as presheaves over the category $\mathbf{I}$. We now introduce a coverage on $\mathbf{I}$, and argue that the only state spaces of interest are in fact sheaves for this coverage.

In Section 4.2.1 we investigate some basic properties of this sheaf category. Following this, in Section 4.2.2 we consider bisimulations for sheaf state spaces, and then prove that the behaviour endofunctors introduced in equations 3.2.11 and 3.2.13 lift along the inclusion of the sheaf category into the presheaf category.

4.2.1 The Schanuel topos

We will consider the singleton coverage $\mathscr{T}_\mathbf{I}$ on $\mathbf{I}$ generated by the collection of all inclusion maps $(C \hookrightarrow D)$, for $C \subseteq D$ in $\mathbf{I}$. To see that $\mathscr{T}_\mathbf{I}$ is stable, suppose that $C \subseteq D$, and consider an injection $\iota : C \rightarrowtail C'$. Pick an object E with a bijection $\beta : D \to E$ such that E is disjoint from C'. We have the following situation.



A similar situation arose in Section 3.3.3, in showing that the closure operator presented in (3.3.14) was well-defined. In fact, the operator $(Q \mapsto \bar{Q})$ presented there is the closure operator on

subobjects of presheaves in $\mathbf{Set}^{\mathbf{I}}$ as derived from the coverage $\mathscr{T}_{\mathbf{I}}$ [see *e.g.* Johnstone, 2002, Example A4.3.5].

Supports and seeds. It is convenient to reformulate the notion of compatible family for the Schanuel topos. There is a way of determining the names used by a state by observing the action of the injections on the state. We say that a set of names obtained in this way supports the state; formally:

> A set $D \in \mathbf{I}$ *supports* a state $p \in P(C)$ if $D \subseteq C$ and, for any $\iota, \jmath : C \rightarrowtail C'$ in $\mathbf{I}$ with $\iota|_D = \jmath|_D$, we have $P\iota(p) = P\jmath(p)$.

Thus, if D supports $p \in P(C)$ then the action of an injection on the state is determined by the action of the injection on D.

For an example, consider the presheaf P_π of π-calculus processes, introduced in (3.2.1). A set $D \subseteq C$ supports a process $p \in P_\pi(C)$ if and only if $\mathsf{fn}(p) \subseteq D$.

If D does support $p \in P(C)$, and there is an element $p' \in P(D)$ such that $[D \hookrightarrow C]p' = p$, then we say that the element p' is a *seed of p at D*.

In our sheaf-theoretic terminology, the statement "D supports $p \in P(C)$" means that the singleton set $\{p\}$ is compatible with the singleton family $\{(D \hookrightarrow C)\}$. The statement "$p'$ is a seed of p at D" means that p' is an amalgamation of that compatible family. Thus, to require that a presheaf P is $\mathscr{T}_{\mathbf{I}}$-separated is to require that whenever D supports $p \in P(C)$, then p has at most one seed at D. To require that a presheaf P is a $\mathscr{T}_{\mathbf{I}}$-sheaf is to require that whenever D supports $p \in P(C)$, then p has a unique seed at D.

For a sheaf P, when D supports $p \in P(C)$ we will write $\mathsf{seed}(p@D)$ for the unique seed of p at D.

Properties of supports. We collect some useful basic properties of supports.

Proposition 4.2.1. *Let P be a presheaf in $\mathbf{Set}^{\mathbf{I}}$. Consider name contexts $C, D \in \mathbf{I}$ such that $D \subseteq C$.*

1. *Consider $p \in P(C)$ supported by D. For any natural transformation $\alpha : P \to Q$ to a presheaf Q we have that D supports $\alpha_C(p)$.*
2. *Consider $p \in P(C)$. For any monic natural transformation $\alpha : P \rightarrowtail Q$ to a presheaf Q, if D supports $\alpha_C(p)$, then D also supports p.*
3. *If $p \in P(D)$ then D supports $[D \hookrightarrow C]p$.*

Proof. Consider injections $\iota, \jmath : C \to C'$ such that $\iota|_D = \jmath|_D$. For item (1),

$$\begin{aligned} [\iota](\alpha_C(p)) &= \alpha_{C'}([\iota]p) && \text{(naturality of } \alpha) \\ &= \alpha_{C'}([\jmath]p) && \text{(since } D \text{ supports } p) \\ &= [\jmath](\alpha_C(p)) && \text{(naturality of } \alpha) \end{aligned}$$

as required.

For item (2), we note that

$$\begin{aligned} \alpha_{C'}([\iota]p) &= [\iota](\alpha_C(p) && \text{(naturality of } \alpha) \\ &= [\jmath](\alpha_C(p) && \text{(since } D \text{ supports } \alpha_C) \\ &= \alpha_{C'}([\jmath]p) && \text{(naturality of } \alpha). \end{aligned}$$

Since α is monic, the component $\alpha_{C'}$ is an injection between sets. Thus $[\iota]p = [\jmath]p$, as required.

For item (3), $[\iota][D \hookrightarrow C] = [\jmath][D \hookrightarrow C]$ and so $[\iota][D \hookrightarrow C]p = [\jmath][D \hookrightarrow C]p$, as required. □

Separated presheaves.

Proposition 4.2.2. *A presheaf $P \in \mathbf{Set}^{\mathbf{I}}$ is $\mathcal{T}_{\mathbf{I}}$-separated if and only if for each $C, D \in \mathbf{I}$ with $C \subseteq D$, the function $P(C \hookrightarrow D) : PC \to PD$ is injective.*

Proof. First, we prove from left to right. We suppose that a presheaf $P \in \mathbf{Set}^{\mathbf{I}}$ is $\mathcal{T}_{\mathbf{I}}$-separated, and show that, for any $C, D \in \mathbf{I}$ with $C \subseteq D$, the function $P(C \hookrightarrow D) : PC \to PD$ is injective.

Consider some $p, p' \in PC$ such that $[C \hookrightarrow D]p = [C \hookrightarrow D]p'$. By Prop. 4.2.1, we have that p and p' are two seeds of the element $[C \hookrightarrow D]p \in P(D)$ that is supported by C. Since P is $\mathcal{T}_{\mathbf{I}}$-separated, we must have at most one such seed, so $p = p'$.

To prove from right to left, we suppose that C supports $p \in P(D)$. Suppose that we have two seeds $p, p' \in PC$, for p at C — so $[C \hookrightarrow D]p = [C \hookrightarrow D]p'$. Then, because $P(C \hookrightarrow D)$ is injective, we have that $p = p'$, and the two seeds are the same. □

Every injection in $\mathbf{I}$ factors as a bijection followed by an inclusion. Bijections are invertible so the action of P on bijections is necessarily bijective; thus a presheaf $P \in \mathbf{Set}^{\mathbf{I}}$ is $\mathcal{T}_{\mathbf{I}}$-separated if and only if P preserves injections.

We assert that $\mathcal{T}_{\mathbf{I}}$-separation is a reasonable property to expect of presheaves. Suppose that $C \subseteq C' \in \mathbf{I}$. If we follow the intuition that has been introduced, the action of a presheaf $P \in \mathbf{Set}^{\mathbf{I}}$ on the inclusion map $(C \hookrightarrow C')$ will map states in name context C to states in name context C'. Our argument is that if two states are distinct in one name context then it is strange if they become the same state when more names are considered.

Sheaves. We assert that the sheaf condition for $\mathcal{T}_{\mathbf{I}}$ is a reasonable one to expect of presheaves. Our intuition is that a state that appears to be only using names in D should exist in name context D. It is clear that the presheaf $P_\pi \in \mathbf{Set}^{\mathbf{I}}$ introduced in (3.2.1) is a sheaf.

Since we will not consider any other coverage on $\mathbf{I}$, we will hereafter refer to $\mathcal{T}_{\mathbf{I}}$-separated functors as simply *separated* and we will refer to $\mathcal{T}_{\mathbf{I}}$-sheaves as sheaves. We write $\mathbf{Sh}(\mathbf{I})$ for the full subcategory of $\mathbf{Set}^{\mathbf{I}}$ whose objects are sheaves.

Pullback-preserving presheaves, and intersections of supports. Since pullbacks of monos are monic, we know that $\mathbf{I}$ has pullbacks and that the inclusion functor $\mathbf{I} \to \mathbf{Set}$ preserves them. We recall a characterisation result.

Proposition 4.2.3. *A presheaf in $\mathbf{Set}^{\mathbf{I}}$ is a sheaf if and only if it preserves pullbacks.* □

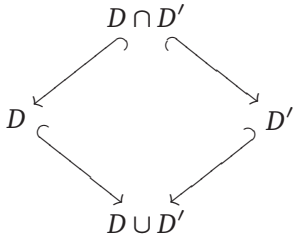
This result is Exercise III.13 of Mac Lane and Moerdijk [1992]. A solution is given by Johnstone [2002, Example A2.1.11(h)], and involves an elegant combinatorial argument.

According to Johnstone [1989, Remark 1.4], Prop. 4.2.3 was first observed in unpublished work by S. H. Schanuel; the category $\mathbf{Sh}(\mathbf{I})$ has been called the *Schanuel topos*.

Prop. 4.2.3 has the following consequence.

Proposition 4.2.4. *Let $P \in \mathbf{Set}^{\mathbf{I}}$ be a $\mathcal{T}_{\mathbf{I}}$-separated presheaf. If both D and D' support $p \in P(C)$, then $D \cap D'$ also supports p.*

Proof. For the case where P is a sheaf, the result follows from Prop. 4.2.3 by considering the following pullback in $\mathbf{I}$.



Now, for the case where P is separated, we know that the map from P into its associated sheaf is monic [see *e.g.* Mac Lane and Moerdijk, 1992, Lem.III.5.2], and so the result follows from Prop. 4.2.1(2). □

4.2.2 Coalgebras in the Schanuel topos

We now develop coalgebraic models of name-passing in the Schanuel topos. We begin with an improvement on the analysis of bisimulation provided in Section 3.3.2, which provides a strong motivation for restricting attention to coalgebras whose carriers are sheaves.

We then show that all the operators on $\mathbf{Set}^{\mathbf{I}}$ that were introduced in Section 3.2.1 restrict to constructions on $\mathbf{Sh}(\mathbf{I})$, and hence we are able to consider B_{e} as an endofunctor on $\mathbf{Sh}(\mathbf{I})$.

Bisimulation for coalgebras over sheaves. We now revisit the discussion of Section 3.3.2. When the carriers of coalgebras are sheaves then a direct correspondence between the coalgebraic and transition system notions of bisimulation is possible.

Theorem 4.2.5. *Let (P,h) and (Q,k) be B_{e}-coalgebras. Suppose that P and Q are both sheaves. Every* **I***-indexed early bisimulation R between induced* **I**-IL$_{\mathrm{e}}$TS*s $\longrightarrow_h$ and $\longrightarrow_k$ is also a B_{e}-bisimulation between B_{e}-coalgebras (P,h) and (Q,k).*

A proof of this result is provided in Appendix 4.A. I further conjecture that the statement of Theorem 4.2.5 holds if P and Q are only required to be separated.

Sheaf-structured coalgebras. Following Section 3.2, we are led to sheaf-based coalgebraic models of name-passing, namely the classes of $J_{\mathbf{I}}^{(\mathbf{I},\mathscr{T}_{\mathbf{I}})}$-structured B_{e}-coalgebras and $J_{\mathbf{I}}^{(\mathbf{I},\mathscr{T}_{\mathbf{I}})}$-structured B_{g}-coalgebras, where we write $J_{\mathbf{I}}^{(\mathbf{I},\mathscr{T}_{\mathbf{I}})}$ for the embedding of $\mathbf{Sh}(\mathbf{I})$ into $\mathbf{Set}^{\mathbf{I}}$.

Constructions of sheaves. Structuring of coalgebras is not necessary, since, as we now explain, the constructions of Section 3.2.1 all restrict to constructions on sheaves, and so we can define a behaviour endofunctor directly on the category $\mathbf{Sh}(\mathbf{I})$.

Theorem 4.2.6.

1. *Products of sheaves are calculated pointwise, as in* $\mathbf{Set}^{\mathbf{I}}$.
2. *The function space $[N \Rightarrow P]$ in* $\mathbf{Set}^{\mathbf{I}}$ *is a sheaf whenever P is a sheaf.*
3. *Coproducts of sheaves are calculated pointwise, as in* $\mathbf{Set}^{\mathbf{I}}$.
4. *The object of names $N \in \mathbf{Set}^{\mathbf{I}}$ is a sheaf.*
5. *If $P \in \mathbf{Set}^{\mathbf{I}}$ is a sheaf, the space δP of names generated into P is a sheaf.*
6. *If P is a sheaf then so are $\mathscr{P}P$ and $\mathscr{P}_{\mathrm{ne}}P$.*
7. *If P and Q are sheaves then $[Q \rightharpoonup P]$ is a sheaf.*

Proof. Item (1) is standard [see *e.g.* Mac Lane and Moerdijk, 1992, Prop. III.4.4]. Item (2) can also be derived from standard results [*e.g.* Mac Lane and Moerdijk, 1992, Prop. III.6.1] by showing that $[N \Rightarrow P]$ is the exponential in $\mathbf{Set}^{\mathbf{I}}$. However, instead of doing this, we will provide an explicit proof that $[N \Rightarrow P]$ is a sheaf when P is a sheaf.

We begin by proving that if P is separated then so is $[N \Rightarrow P]$. So we suppose that P is separated, and, following Prop. 4.2.2, we show that for each $C, D \in \mathbf{I}$ with $C \subseteq D$, the action $[N \Rightarrow P](C \hookrightarrow D)$ is injective.

To this end, consider $\phi, \psi \in [N \Rightarrow P](C)$, and suppose that

$$[N \Rightarrow P](C \hookrightarrow D)(\phi) = [N \Rightarrow P](C \hookrightarrow D)(\psi) \quad .$$

We know from equation 3.2.3c that, for all $z \in \mathcal{N}$,

$$P(C \cup \{z\} \hookrightarrow D \cup \{z\})(\phi(z)) = P(C \cup \{z\} \hookrightarrow D \cup \{z\})(\phi(z)) \quad .$$

It follows that $\phi = \psi$, because for each $z \in \mathcal{N}$, the action $P(C \cup \{z\} \hookrightarrow D \cup \{z\})$ is injective, by Prop. 4.2.2. Thus we know that $[N \Rightarrow P]$ is separated if P is.

To finish proving item (2), we now suppose that P is a sheaf, and show that $[N \Rightarrow P]$ is also a sheaf. Consider $C \in \mathbf{I}$ together with $\phi \in [N \Rightarrow P](C)$. Suppose that we have that $D \subseteq C$ and that D supports ϕ.

We will first prove that for each $z \in \mathcal{N}$ we have that $(D \cup \{z\})$ supports $\phi(z) \in P(C \cup \{z\})$. To this end, consider a name context $E \in \mathbf{I}$, and injections $\iota, \jmath : C \cup \{z\} \to E$ which are such that $\iota|_{D \cup \{z\}} = \jmath|_{D \cup \{z\}}$. We must show that $P\iota(\phi(z)) = P\jmath(\phi(z))$. We do this differently depending on whether z is in C.

For the case where $z \in C$, consider the following sequence of identities.

$$\begin{aligned}
P\iota(\phi(z)) &= \big([N \Rightarrow P]\iota(\phi)\big)(\iota(z)) && (1)\\
&= \big([N \Rightarrow P]\jmath(\phi)\big)(\iota(z)) && (2)\\
&= \big([N \Rightarrow P]\jmath(\phi)\big)(\jmath(z)) && (3)\\
&= P\jmath(\phi(z)) && (4)
\end{aligned}$$

Using: (1) defn. of action of $[N \Rightarrow P]$; (2) since D supports ϕ, and $\iota|_D = \jmath|_D$; (3) since $\iota(z) = \jmath(z)$; (4) action of $[N \Rightarrow P]$.

For the case where $z \in (\mathcal{N} - C)$, we let $\iota', \jmath' : C \to E$ act as the injections $\iota, \jmath$ on the restricted domain. Then we have the following sequence of equations.

$$\begin{aligned}
P\iota(\phi(z)) &= P[\iota', \iota(z)/z](\phi(z)) && (1)\\
&= \big([N \Rightarrow P]\iota'(\phi)\big)(\iota(z)) && (2)\\
&= \big([N \Rightarrow P]\jmath'(\phi)\big)(\iota(z)) && (3)\\
&= \big([N \Rightarrow P]\jmath'(\phi)\big)(\jmath(z)) && (4)\\
&= P[\jmath', \jmath(z)/z](\phi(z)) && (5)\\
&= P\jmath(\phi(z)) && (6)
\end{aligned}$$

Using: (1) defn. of $[\iota', \iota(z)/z]$; (2) action of $[N \Rightarrow P]$; (3) since D supports ϕ, and $\iota'|_D = \jmath'|_D$; (4) since $\iota(z) = \jmath(z)$; (5) action of $[N \Rightarrow P]$; (6) defn. of $[\jmath', \jmath(z)/z]$.

So we know that for each $z \in \mathcal{N}$ we have that $(D \cup \{z\})$ supports $\phi(z)$. Using this fact, and the assumption that P is a sheaf, we define the seed of ϕ at D,

$$\mathsf{seed}(\phi @ D) \in \prod_{z \in \mathcal{N}} P(C \cup \{z\})$$

as follows: for each $z \in \mathcal{N}$, let

$$\mathsf{seed}(\phi @ D)(z) = \mathsf{seed}((\phi(z)) @ (D \cup \{z\})) \quad .$$

It remains for us to check that $\mathsf{seed}(\phi @ D)$ is in $[N \Rightarrow P](D)$, *i.e.*, that it satisfies the uniformity condition. Consider $z, z' \in (\mathcal{N} - D)$. If neither of z, z' are in C then the uniformity condition follows from the uniformity condition for ϕ. We now consider the case where $z \in C$ and $z' \notin C$. We must show that

$$P[z'/z](\mathsf{seed}((\phi(z)) @ (D \cup \{z\}))) \;=\; \mathsf{seed}((\phi(z')) @ (D \cup \{z'\})) \quad .$$

By the universal property of seeds, it suffices to show that

$$P((D \cup \{z'\} \hookrightarrow C \cup \{z'\}) \circ [z'/z])(\mathsf{seed}((\phi(z))@(D \cup \{z\}))) \;=\; \phi(z') \quad .$$

This is established by the following sequence of identities.

$$\begin{aligned}
&P\left((D \cup \{z'\} \hookrightarrow C \cup \{z'\}) \circ [z'/z]\right)\left(\mathsf{seed}((\phi(z))@(D \cup \{z\}))\right) \\
&\quad = P\left(((C - \{z\}) \cup \{z'\} \hookrightarrow C \cup \{z'\}) \circ [z'/z] \circ (D \cup \{z\} \hookrightarrow C)\right) \\
&\qquad\qquad \left(\mathsf{seed}((\phi(z))@(D \cup \{z\}))\right) && (1)\\
&\quad = P\left(((C - \{z\}) \cup \{z'\} \hookrightarrow C \cup \{z'\}) \circ [z'/z]\right)\ (\phi(z)) && (2)\\
&\quad = \left([N \Rightarrow P]\left(((C - \{z\}) \cup \{z'\} \hookrightarrow C \cup \{z'\}) \circ [z'/z]\right)(\phi)\right)\ (z') && (3)\\
&\quad = \left([N \Rightarrow P](C \hookrightarrow C \cup \{z'\})(\phi)\right)\ (z') && (4)\\
&\quad = \phi(z') && (5)
\end{aligned}$$

Using: (1) factorisation; (2) defn. of seeds; (3) action of $[N \Rightarrow P]$; (4) since D supports ϕ, and since $(C \hookrightarrow C \cup \{z'\})|_D = (((C - \{z\}) \cup \{z'\} \hookrightarrow C \cup \{z'\}) \circ [z'/z])|_D$; (5) eqn. 3.2.3c.

So we have the uniformity condition when at most one of $\{z, z'\}$ are in C. For the case when both z and z' are in C, we pick another name $z'' \in \mathcal{N}$ that is not in C, and use the former result twice. Thus $\mathsf{seed}(\phi@D)$ satisfies the uniformity condition, and so is indeed an element of $[N \Rightarrow P](D)$. So $[N \Rightarrow P]$ is a sheaf.

Item (3) is true in any sheaf topos with a coverage generated by singleton families. In the present case, consider an I-indexed family of sheaves $\{P_i\}_{i \in I}$ and the coproduct $\coprod_{i \in I} P_i$ in $\mathbf{Set}^{\mathbf{I}}$. For any element

$$\mathsf{inj}_i(p) \in \left(\coprod_{i \in I} P_i\right)(C)$$

if D supports $\mathsf{inj}_i(p)$ then D supports $p \in P_i(C)$. Hence we have a seed $\mathsf{seed}(p@D) \in P_i(D)$ and also a seed

$$\mathsf{inj}_i(\mathsf{seed}(p@D)) \in \left(\coprod_{i \in I} P_i\right)(D) \quad .$$

Item (4) is straightforward, because if D supports $c \in N(C)$ then $c \in D$. To put it another way: N is the inclusion functor $\mathbf{I} \to \mathbf{Set}$, and pullbacks of monos are monic, so N preserves pullbacks and item (4) follows from Prop. 4.2.3. (Indeed, the coverage $\mathcal{T}_{\mathbf{I}}$ is subcanonical, and N is representable — it represents a singleton name context.)

Item (5) is established in a manner similar to item (2). Turning to item (6), the simplest proof that I know uses Prop. 4.2.3. We concentrate on the powerset functor $\mathscr{P}$; the non-empty variant is treated similarly. The presheaf $\mathscr{P}P$ is the composite

$$\mathbf{I} \xrightarrow{P} \mathbf{Set} \xrightarrow{\mathscr{P}} \mathbf{Set} \qquad (4.2.7)$$

where $\mathscr{P}$ is the powerset endofunctor on **Set**.

Suppose that P is separated. Then, by Prop. 4.2.2, we have that the presheaf P preserves monos. It is well-known that the endofunctor $\mathscr{P}$ on **Set** preserves monos; thus we know that the composite of (4.2.7) preserves monos, *i.e.* is separated.

Now suppose that P is a sheaf; by Prop. 4.2.3, it preserves pullbacks. It remains for us to show that the endofunctor $\mathscr{P}$ on **Set** preserves pullbacks of monos. It is well-known that $\mathscr{P}$ on **Set** preserves weak pullbacks (*i.e.* pullbacks but where the mediating morphisms need not be unique). As mentioned above, the endofunctor $\mathscr{P}$ also preserves monos. (In fact, preservation of monos follows from preservation of weak pullbacks.) A weak pullback of monos is a strong pullback, and so the composite of (4.2.7) also preserves pullbacks, and so, by Prop. 4.2.2, it is a sheaf.

For item (7), we consider a partial function $\phi \in [Q \rightharpoonup P](C)$ that is supported by $D \subseteq_{\mathrm{f}} C$ and we prove that (i) if $\phi \downarrow q$ then D supports q; (ii) that if $\phi \downarrow q$ and D supports q then D supports $\phi(q)$. Given these two results we can define $\mathsf{seed}(\phi@D) \in [Q \rightharpoonup P](D)$ by (for $q \in Q(D)$):

$$\mathsf{seed}(\phi@D)\ (q) = \begin{cases} \mathsf{seed}(\phi([D \hookrightarrow C]q)@D) & \text{if } \phi \downarrow [D \hookrightarrow C]q \\ \text{undefined} & \text{otherwise.} \end{cases}$$

For item (i): note that, by Prop. 4.2.1(1), D also supports

$$\phi' = [Q \rightharpoonup (! : P \to 1)]_C(\phi) \quad \in [Q \rightharpoonup 1](C)$$

It is easy to see that $[Q \rightharpoonup 1]$ is isomorphic to $\mathscr{P}(Q)$. Now, by item (6), $\mathscr{P}(Q)$ is a sheaf, and so $[Q \rightharpoonup 1]$ is a sheaf. Hence we have a seed $\mathsf{seed}(\phi'@D) \in [Q \rightharpoonup 1](D)$. Consider some $q \in \mathsf{dom}(\phi)$. By definition of $[Q \rightharpoonup (! : P \to 1)]$ we have $q \in \mathsf{dom}(\phi')$. So, looking at the action of ϕ', we have q' in $Q(D)$ such that $[D \hookrightarrow C]q' = q$. So, by Prop. 4.2.1(3) we know that D supports q.

For item (ii): we consider injections $\iota, \jmath : C \to C'$ such that $\iota|_D = \jmath|_D$. We suppose that D supports $q \in \mathsf{dom}(\phi)$; we must show that $[\iota](\phi(q)) = [\jmath](\phi(q))$. Now

$$\begin{aligned} [\iota](\phi(q)) &= ([\iota]\phi)([\iota]q) && \text{(action of } \phi) \\ &= ([\jmath]\phi)([\jmath]q) && \text{(since } D \text{ supports } \phi \text{ and } q) \\ &= [\jmath](\phi(q)) && \text{(action of } \phi) \end{aligned}$$

as required. □

Coalgebras over sheaves. From Theorem 4.2.6 we know that the behaviour endofunctors B_{e} and B_{g} on $\mathbf{Set}^{\mathbf{I}}$ both lift, along the embedding $J_{\mathbf{I}}^{(\mathbf{I},\mathscr{T}_{\mathbf{I}})} : \mathbf{Sh}(\mathbf{I}) \to \mathbf{Set}^{\mathbf{I}}$, to endofunctors $B_{\mathrm{e}}^{\mathrm{Sh}}$ and $B_{\mathrm{g}}^{\mathrm{Sh}}$ on $\mathbf{Sh}(\mathbf{I})$. (We will omit the superscript (Sh) when it is clear from the context.) We have the situation discussed in Example 2.4.3(3), and so there are isomorphisms of categories

$$(J_{\mathbf{I}}^{(\mathbf{I},\mathscr{T}_{\mathbf{I}})}, B_{\mathrm{e}})\text{-Coalg} \cong B_{\mathrm{e}}^{\mathrm{Sh}}\text{-Coalg} \qquad (J_{\mathbf{I}}^{(\mathbf{I},\mathscr{T}_{\mathbf{I}})}, B_{\mathrm{g}})\text{-Coalg} \cong B_{\mathrm{g}}^{\mathrm{Sh}}\text{-Coalg} \quad .$$

The embedding $J_{\mathbf{I}}^{(\mathbf{I},\mathscr{T}_{\mathbf{I}})} : \mathbf{Sh}(\mathbf{I}) \to \mathbf{Set}^{\mathbf{I}}$ has a left adjoint [the *associated sheaf* functor; see *e.g.* Johnstone, 2002, Example A4.1.8]. Thus we can conclude, using Prop. 2.3.1, that the morphisms of endofunctors

$$(\mathbf{Sh}(\mathbf{I}), B_{\mathrm{e}}^{\mathrm{Sh}}) \to (\mathbf{Set}^{\mathbf{I}}, B_{\mathrm{e}}) \qquad (\mathbf{Sh}(\mathbf{I}), B_{\mathrm{g}}^{\mathrm{Sh}}) \to (\mathbf{Set}^{\mathbf{I}}, B_{\mathrm{g}})$$

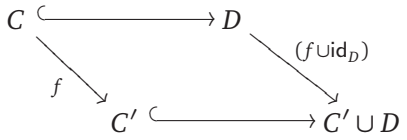
both have left adjoints in coEndo. Thus, by Theorem 2.5.7, the final B_{e}-bisimulation between a pair of B_{e}-coalgebras whose carriers are sheaves, is itself a sheaf; a similar result holds for the ground case. (An alternative way to establish this result is to combine Prop. 3.3.13 with Theorem 3.3.16.)

4.3 Sheaves and all substitutions

In Section 3.4.3 we considered models for name-passing for which the state spaces admitted all substitutions, not only injective renamings. These models captured the uniformity of input behaviour, and also provided a framework for studying wide open bisimulation.

We will now bring these models in line with the sheaf-based approach of the previous section. To do this, it is necessary to consider a coverage on the category $\mathbf{F}$. After studying the corresponding category of sheaves, we show that the sheaf requirement for this coverage on $\mathbf{F}$ is in fact very simple.

Coverages for F. Unfortunately, the forgetful functor $U^{\mathbf{I}}_{\mathbf{F}} : \mathbf{Set}^{\mathbf{F}} \to \mathbf{Set}^{\mathbf{I}}$ does not factor through the subcategory $\mathbf{Sh}(\mathbf{I})$. It is necessary to impose a sheaf condition on the presheaves in $\mathbf{Set}^{\mathbf{F}}$. We will consider the coverage $\mathscr{T}_{\mathbf{F}}$ of singletons, generated by the collection of all inclusion maps in $\mathbf{F}$. To see that $\mathscr{T}_{\mathbf{F}}$ is stable, consider $C, D \in \mathbf{F}$ such that $C \subseteq D$, and consider a function $f : C \to C'$ in $\mathbf{F}$. Then the following diagram commutes in $\mathbf{F}$.



A similar situation arose in Section 3.4.2, in showing that the closure operator of (3.4.7) was well-defined. Indeed, that closure operator on subobjects of presheaves in $\mathbf{Set}^{\mathbf{F}}$ derives from the coverage $\mathscr{T}_{\mathbf{F}}$, in the same way that the closure operator on subobjects of presheaves in $\mathbf{Set}^{\mathbf{I}}$ (of (3.3.14)) derives from the coverage $\mathscr{T}_{\mathbf{I}}$.

As we now prove, this coverage is exactly the one needed to restrict the forgetful functor to a functor $\mathbf{Sh}(\mathbf{F}, \mathscr{T}_{\mathbf{F}}) \to \mathbf{Sh}(\mathbf{I}, \mathscr{T}_{\mathbf{I}})$.

Proposition 4.3.1. *Consider a presheaf $X \in \mathbf{Set}^{\mathbf{F}}$.*

1. *X is $\mathscr{T}_{\mathbf{F}}$-separated if and only if $U^{\mathbf{I}}_{\mathbf{F}}X$ is $\mathscr{T}_{\mathbf{I}}$-separated.*
2. *X is a $\mathscr{T}_{\mathbf{F}}$-sheaf if and only if $U^{\mathbf{I}}_{\mathbf{F}}X$ is a $\mathscr{T}_{\mathbf{I}}$-sheaf.*

Proof. To begin, we note that the inclusion functor $j^{\mathbf{I}}_{\mathbf{F}} : \mathbf{I} \to \mathbf{F}$ preserves and reflects covers. Thus item (1), and also the right-to-left part of item (2), follow from Prop. 4.1.3.

It remains for us to prove item (2) from left to right. Suppose that X is a $\mathscr{T}_{\mathbf{F}}$-sheaf. We will show that $U^{\mathbf{I}}_{\mathbf{F}}X$ is a $\mathscr{T}_{\mathbf{I}}$-sheaf. Consider a compatible family $\left\{x \in U^{\mathbf{I}}_{\mathbf{F}}X(D)\right\}$ for some $\mathscr{T}_{\mathbf{I}}$-cover, $\{(C \hookrightarrow D)\}$. This $\mathscr{T}_{\mathbf{I}}$-cover is also a $\mathscr{T}_{\mathbf{F}}$-cover. We will show that the family $\{x \in X(D)\}$ is compatible for this $\mathscr{T}_{\mathbf{F}}$-cover.

The category $\mathbf{F}$ has finite coproducts. We pick such a binary coproduct structure, as usual writing '+' for the binary coproduct, and inl and inr for the left and right binary coproduct injections respectively.

Consider $E \in \mathbf{F}$, and two functions $f, g : D \to E$ in $\mathbf{F}$, which are such that $f|_C = g|_C$. We define two injections $\imath, \jmath : D \to (D + D)$ as follows.

$$\imath = D \xrightarrow{\mathsf{inl}} (D + D)$$
$$\jmath = D \xrightarrow{=} (C \cup (D - C)) \xrightarrow{(\mathsf{inl} \circ (C \hookrightarrow D)) \cup (\mathsf{inr} \circ ((D-C) \hookrightarrow D))} (D + D)$$

So $\imath|_C = \jmath|_C$. Also, since $f|_C = g|_C$, we have $(f + \mathsf{id}_D) \circ \jmath = (g + \mathsf{id}_D) \circ \jmath : D \to (E + D)$.

Note that the following diagrams commute.

Thus

$$[\mathsf{inl}_{(E+D)}][f]x = [f + \mathsf{id}_D][\imath]x \tag{4.3.2a}$$
$$= [f + \mathsf{id}_D][\jmath]x \tag{4.3.2b}$$
$$= [g + \mathsf{id}_D][\jmath]x \tag{4.3.2c}$$
$$= [g + \mathsf{id}_D][\imath]x \tag{4.3.2d}$$
$$= [\mathsf{inl}_{(E+D)}][g]x \tag{4.3.2e}$$

Using: (4.3.2a): since $(\mathsf{inl}_{(E+D)} \circ f) = (f + \mathsf{id}_D \circ \iota)$; (4.3.2b): since $U^{\mathbf{I}}_{\mathbf{F}}X$ is a $\mathscr{T}_{\mathbf{I}}$-sheaf, and C supports x, and $\iota|_C = \jmath|_C$; (4.3.2c): since $((f + \mathsf{id}_D) \circ \jmath) = (g + \mathsf{id}_D \circ \jmath)$; (4.3.2d): since $U^{\mathbf{I}}_{\mathbf{F}}X$ is a $\mathscr{T}_{\mathbf{I}}$-sheaf, and C supports x, and $\iota|_C = \jmath|_C$; (4.3.2e): since $(\mathsf{inl}_{(E+D)} \circ g) = (g + \mathsf{id}_D \circ \iota)$.

Since X is a $\mathscr{T}_{\mathbf{F}}$-sheaf, it is $\mathscr{T}_{\mathbf{F}}$-separated. So, by item (1) of this proposition, we have that $U^{\mathbf{I}}_{\mathbf{F}}X$ is $\mathscr{T}_{\mathbf{I}}$-separated, and so the action of the injection $\mathsf{inl} : E \to (E + D)$ is injective. Thus we have that $[f]x = [g]x$. So the singleton $\{x\}$ is a compatible family for the $\mathscr{T}_{\mathbf{F}}$-cover $\{(C \hookrightarrow D)\}$. Since X is a $\mathscr{T}_{\mathbf{F}}$-sheaf, there is an amalgamation, $x' \in X(C)$. This element also serves as an amalgamation for the $\mathscr{T}_{\mathbf{I}}$-cover $\{(C \hookrightarrow D)\}$. Thus $U^{\mathbf{I}}_{\mathbf{F}}X$ is a $\mathscr{T}_{\mathbf{I}}$-sheaf. □

We write $\mathbf{Sh}(\mathbf{F})$ for the full subcategory of $\mathbf{Set}^{\mathbf{F}}$ with objects $\mathscr{T}_{\mathbf{F}}$-sheaves.

A simple check for $\mathscr{T}_{\mathbf{F}}$-separated presheaves and $\mathscr{T}_{\mathbf{F}}$-sheaves. Let $1 \in \mathbf{F}$ be a terminal object of $\mathbf{F}$; it is a singleton set. It is interesting to consider the coverage $\mathscr{T}_{\mathbf{F},1}$ on $\mathbf{F}$ for which the only cover is the initial/terminal map $(0 \xrightarrow{!} 1)$. The following result explains that the requirements for $\mathscr{T}_{\mathbf{F}}$-separatedness and $\mathscr{T}_{\mathbf{F}}$-sheaves are very simple.

Proposition 4.3.3. *Consider a presheaf $X \in \mathbf{Set}^{\mathbf{F}}$.*

1. *X is $\mathscr{T}_{\mathbf{F}}$-separated if and only if it is $\mathscr{T}_{\mathbf{F},1}$-separated.*

2. *X is a $\mathscr{T}_{\mathbf{F}}$-sheaf if and only if it is a $\mathscr{T}_{\mathbf{F},1}$-sheaf.*

Proof. The left-to-right direction of both items is trivial. They follow from Prop. 4.1.3 together with the fact that the identity functor on $\mathbf{F}$ is cover-reflecting between sites $(\mathbf{F}, \mathscr{T}_{\mathbf{F}}) \to (\mathbf{F}, \mathscr{T}_{\mathbf{F},1})$.

To prove item (1) from right to left, we suppose that X is $\mathscr{T}_{\mathbf{F},1}$-separated. We will show that it is $\mathscr{T}_{\mathbf{F}}$-separated. By Prop. 4.3.1, this is equivalent to showing that for every injection $\iota : C \rightarrowtail D$ then the action $X\iota : XC \to XD$ is injective.

For the case when $C \neq 0$, we can find a retraction $D \to C$ of the injection ι; thus the action $X\iota$ is forced to be injective.

For the case when $C = 0$, we proceed as follows. Since $(0 \xrightarrow{!} 1)$ is a cover, we have that the action $X(0 \xrightarrow{!} 1)$ is injective. Now, every function $0 \xrightarrow{!} D$ is either the identity on 0, in which case the result is trivial, or it factors as the unique map $(0 \xrightarrow{!} 1)$ followed by an injection $1 \to D$. In this case, the actions of both these parts have been shown to be injective, thus the action $X(0 \xrightarrow{!} D)$ is injective.

To prove item (2) from right to left, we suppose that X is a $\mathscr{T}_{\mathbf{F},1}$-sheaf. We will show that it is $\mathscr{T}_{\mathbf{F}}$-sheaf.

Consider a singleton family $\{(C \hookrightarrow D)\}$ that is a $\mathscr{T}_{\mathbf{F}}$-cover, and let $\{x \in X(D)\}$ be a compatible family for this cover.

For the case $C \neq 0$, we pick a retraction $r : D \to C$ of the inclusion map $(C \hookrightarrow D)$. So we have that $((C \hookrightarrow D) \circ r \circ (C \hookrightarrow D)) = (C \hookrightarrow D)$. Thus, since x is compatible for $(C \in D)$, we have that $[C \hookrightarrow D][r]x = x$. In other words, $[r]x$ is an amalgamation for x.

For the case $C = 0$, we proceed as follows. If also $D = 0$, then the result is trivial, so we concentrate on the case when $D \neq 0$. In this case, we pick a function $\iota : 1 \to D$. For any such choice, the cover factors as

$$(0 \hookrightarrow D) = 0 \xrightarrow{!} 1 \xrightarrow{\iota} D \quad .$$

Moreover, the unique function $r : D \to 1$ is a retraction of ι.

It is easy to see that, since $\{x\}$ is a compatible family for the $\mathscr{T}_{\mathbf{F}}$-cover $(0 \hookrightarrow D)$, then $\{[r]x\}$ is a compatible family for the $\mathscr{T}_{\mathbf{F},1}$-cover $(0 \xrightarrow{!} 1)$. Since X is a $\mathscr{T}_{\mathbf{F},1}$-sheaf, we have an amalgamation $x' \in X(0)$ of $\{[r]x\}$. In fact, x' is an amalgamation of the compatible family $\{x\}$, as we now show.

$$\begin{aligned}[0 \hookrightarrow D]x' &= [\iota][0 \xrightarrow{!} 1]x' \\ &= [\iota][r]x && \text{(4.3.4a)} \\ &= [\mathsf{id}_D]x && \text{(4.3.4b)} \\ &= x\end{aligned}$$

Using: (4.3.4a): because x' is an amalgamation of $\{[r]x\}$ for $(0 \xrightarrow{!} 1)$; (4.3.4b): because $(\iota \circ r \circ (0 \hookrightarrow D)) = (\mathsf{id}_D \circ (0 \hookrightarrow D))$, and $\{x\}$ is compatible for $\{(0 \hookrightarrow D)\}$. □

From $\mathscr{T}_{\mathbf{I}}$-sheaves to $\mathscr{T}_{\mathbf{F}}$-sheaves. The development so far leads us to consider the model for name-passing given by the classes of $U_{\mathbf{F}}^{\mathbf{I}}$-structured $B_{\mathrm{e}}^{\mathrm{Sh}}$-coalgebras.

We note, however, that the structuring here is not strictly necessary. The inclusion functor $\mathbf{I} \to \mathbf{F}$ reflects covers, and so, by Prop. 4.1.4, the direct image functor $(j_{\mathbf{F}}^{\mathbf{I}})_* : \mathbf{Set}^{\mathbf{I}} \to \mathbf{Set}^{\mathbf{F}}$ considered in Section 3.4.1 restricts to a functor $\mathbf{Sh}(\mathbf{I}) \to \mathbf{Sh}(\mathbf{F})$. Thus the behaviour endofunctor $((j_{\mathbf{F}}^{\mathbf{I}})_* B_{\mathrm{e}} U_{\mathbf{F}}^{\mathbf{I}})$ on $\mathbf{Set}^{\mathbf{F}}$, considered in Section 3.4.1, restricts to an endofunctor on $\mathbf{Sh}(\mathbf{F})$.

4.4 Transition systems simplified

We now return to the models of name-passing over sheaves in $\mathbf{Sh}(\mathbf{I})$, as considered in Section 4.2. In Section 4.4.1 we recall some of the analysis of Fiore [2001], who has shown that a sheaf in $\mathbf{Sh}(\mathbf{I})$ can be specified in terms of its minimal seeds. We follow this, in Section 4.4.2, by studying indexed labelled transition systems (in the spirit of Section 3.3) where the only states that are considered are those that are minimal seeds. We thus arrive at a simpler axiomatisation of transition systems for name-passing.

4.4.1 Least supports and minimal seeds

We begin this section by observing that every separated presheaf has a least support. Thus we are led to consider presheaves of minimal seeds, where we consider only the action of bijective renamings. We find a functor from the category of such presheaves into the Schanuel topos, which we show to be essentially surjective.

Least supports and minimal seeds. For any $\mathscr{T}_{\mathbf{I}}$-separated presheaf $P \in \mathbf{Set}^{\mathbf{I}}$ we can associate to each element $p \in P(C)$ a *least support*

$$\mathsf{supp}(p) = \bigcap \{D \subseteq C \mid D \text{ supports } p\}.$$

The set $\mathsf{supp}(p)$ is a subset of C, because C trivially supports p. All the sets of names under consideration are finite; thus, by Prop. 4.2.4, we know that $\mathsf{supp}(p)$ supports the element p. One might think of $\mathsf{supp}(p)$ as the precise set of names that a state p uses. For the sheaf P_π of π-calculus processes, introduced in (3.2.1), the least support $\mathsf{supp}(p)$ of a process $p \in P_\pi(C)$ is precisely the set of free names of that process, $\mathsf{fn}(p)$.

For a sheaf $P \in \mathbf{Sh}(\mathbf{I})$, we write $\mathsf{seed}(p)$ for the minimal seed of p, *i.e.*, we let

$$\mathsf{seed}(p) = \mathsf{seed}(p@\mathsf{supp}(p)) \quad .$$

Presheaves on B. We have been working with the intuition that the elements of a sheaf P at C are the states whose names are contained in C. Following this, one might expect that the entire sheaf can be described simply in terms of the minimal seeds. This is true, as we now describe.

Let $\mathbf{B}$ be the category with objects finite subsets of $\mathcal{N}$, and morphisms *bijections* between them. A presheaf on $\mathbf{B}$ is to be thought of as assigning to each name context C a collection of states for which the set of names that they use is precisely C. The action of a bijection $\beta : C \xrightarrow{\sim} D$ is to be thought of as mapping states that use precisely the names in C to states that use precisely the names in D.

Presheaves of seeds. Examples of presheaves in $\mathbf{Set}^{\mathbf{B}}$ are found as follows. To every sheaf $P \in \mathbf{Sh}(\mathbf{I})$ we can associate a "presheaf of seeds" $\langle P \rangle \in \mathbf{Set}^{\mathbf{B}}$, given by

$$\langle P \rangle(C) = \{p \in P(C) \mid p = \mathsf{seed}(p)\} \quad ; \tag{4.4.1a}$$

for any $\beta : C \xrightarrow{\sim} C'$ we have a map $\langle \beta \rangle : \langle P \rangle(C) \xrightarrow{\sim} \langle P \rangle(C')$ given by

$$\langle P \rangle \beta(p) = P\beta(p) \quad . \tag{4.4.1b}$$

The construction $\langle P \rangle : \mathbf{B} \to \mathbf{Set}$ respects composition and identities, since P does.

(Note that $\langle - \rangle$ does *not* extend to a functor $\mathbf{Sh}(\mathbf{I}) \to \mathbf{Set}^{\mathbf{B}}$. For there is a map $N \to 1$ in $\mathbf{Sh}(\mathbf{I})$, but, for any name $c \in \mathcal{N}$,

$$\langle N \rangle(\{c\}) = \{c\} \qquad \text{and} \qquad \langle 1 \rangle(\{c\}) = \emptyset$$

and so there are no maps $\langle N \rangle \to \langle 1 \rangle$ in $\mathbf{Set}^{\mathbf{B}}$.)

Presheaf of seeds for the π-calculus. For a basic example, recall the sheaf $P_\pi \in \mathbf{Set}^{\mathbf{I}}$ of π-calculus processes introduced in (3.2.1). The presheaf $\langle P_\pi \rangle \in \mathbf{Set}^{\mathbf{B}}$ is given by, for $C \in \mathbf{B}$,

$$\langle P_\pi \rangle(C) = \left\{ p \;\middle|\; p \text{ is a } \pi\text{-calculus process, and } \mathsf{fn}(p) = C \right\} \quad .$$

Relating $\mathbf{Set}^{\mathbf{I}}$ with $\mathbf{Set}^{\mathbf{B}}$. The $\langle - \rangle$ construction is sufficient for us to develop and analyse a model of name-passing based on presheaves in $\mathbf{Set}^{\mathbf{B}}$ (as is done in Section 4.4.2). As a further motivation, however, we explore some properties of this construction.

The inclusion $j_{\mathbf{I}}^{\mathbf{B}} : \mathbf{B} \to \mathbf{I}$ induces (by precomposition) an inverse image functor

$$U_{\mathbf{B}}^{\mathbf{I}} = (j_{\mathbf{I}}^{\mathbf{B}})^* : \mathbf{Set}^{\mathbf{I}} \to \mathbf{Set}^{\mathbf{B}}$$

that forgets what to do with non-bijective renamings. For $P \in \mathbf{Set}^{\mathbf{I}}$,

$$U_{\mathbf{B}}^{\mathbf{I}} P \;=\; \mathbf{B} \hookrightarrow \mathbf{I} \xrightarrow{P} \mathbf{Set}. \tag{4.4.2}$$

For $\alpha : P \to P'$ in $\mathbf{Set}^{\mathbf{I}}$ we have $U_{\mathbf{B}}^{\mathbf{I}}\alpha : U_{\mathbf{B}}^{\mathbf{I}}P \to U_{\mathbf{B}}^{\mathbf{I}}P'$ in $\mathbf{Set}^{\mathbf{B}}$ given by $U_{\mathbf{B}}^{\mathbf{I}}\alpha = \alpha$; we know $U_{\mathbf{B}}^{\mathbf{I}}\alpha$ is natural with respect to bijections since α is natural with respect to all injections.

As usual, this forgetful functor $U_{\mathbf{B}}^{\mathbf{I}}$ has a left adjoint $F_{\mathbf{I}}^{\mathbf{B}} = (j_{\mathbf{I}}^{\mathbf{B}})_! : \mathbf{Set}^{\mathbf{B}} \to \mathbf{Set}^{\mathbf{I}}$. This is given on objects as follows. For $Q \in \mathbf{Set}^{\mathbf{B}}$ and $C \in \mathbf{B}$,

$$F_{\mathbf{I}}^{\mathbf{B}} Q(C) = \coprod_{D \subseteq C} Q(D) \quad . \tag{4.4.3a}$$

So the elements of the set $F_{\mathbf{I}}^{\mathbf{B}} Q(C)$ are of the form $\mathsf{inj}_D q$, where $D \subseteq C$ and $q \in Q(D)$.

The presheaf $F_{\mathbf{I}}^{\mathbf{B}} Q$ is defined to act on morphisms in $\mathbf{B}$ as follows. For $\iota : C \rightarrowtail C'$ and $\mathsf{inj}_D q \in F_{\mathbf{I}}^{\mathbf{B}} Q(C)$ we have $F_{\mathbf{I}}^{\mathbf{B}} Q\iota : F_{\mathbf{I}}^{\mathbf{B}} Q(C) \to F_{\mathbf{I}}^{\mathbf{B}} Q(C')$ given by

$$F_{\mathbf{I}}^{\mathbf{B}} Q\iota(\mathsf{inj}_D q) = \mathsf{inj}_{\iota(D)}(Q(\iota|_D)(q)) \quad . \tag{4.4.3b}$$

Intuitively: $F_{\mathbf{I}}^{\mathbf{B}}Q$ is the presheaf in $\mathbf{Set}^{\mathbf{I}}$ found by growing the minimal seeds described by $Q \in \mathbf{Set}^{\mathbf{B}}$. Recall that every injection $\iota : C \to D$ in $\mathbf{I}$ factors as an inclusion $(\iota(C) \hookrightarrow D)$ precomposed with a bijection $\iota|_C$. The action of $F_{\mathbf{I}}^{\mathbf{B}}Q$ on bijections is inherited from Q; the action on inclusions is as the identity. This matches the slogan "if the names of a state lie in C and $C \subseteq D$ then they also lie in D".

Finally, we define the action of the left adjoint $F_{\mathbf{I}}^{\mathbf{B}} : \mathbf{Set}^{\mathbf{B}} \to \mathbf{Set}^{\mathbf{I}}$ on natural transformations in $\mathbf{Set}^{\mathbf{B}}$. For $\alpha : Q \to Q'$ in $\mathbf{Set}^{\mathbf{B}}$ and $\mathsf{inj}_D q \in F_{\mathbf{I}}^{\mathbf{B}}Q(C)$ we have $F_{\mathbf{I}}^{\mathbf{B}}\alpha : F_{\mathbf{I}}^{\mathbf{B}}Q \to F_{\mathbf{I}}^{\mathbf{B}}Q'$ given by

$$(F_{\mathbf{I}}^{\mathbf{B}}\alpha)_C(\mathsf{inj}_D q) = \mathsf{inj}_D(\alpha_D(q)) \quad . \tag{4.4.3c}$$

Essential surjectivity. The functor $F_{\mathbf{I}}^{\mathbf{B}}$ can be thought of as "growing a sheaf $F_{\mathbf{I}}^{\mathbf{B}}Q$ from the seeds in Q". This construction is inverse to the "finding seeds" construction $\langle - \rangle$ introduced above, in that we have the following properties.

Proposition 4.4.4.

1. *Let Q be a presheaf in $\mathbf{Set}^{\mathbf{B}}$. We have an isomorphism $Q \xrightarrow{\sim} \langle F_{\mathbf{I}}^{\mathbf{B}}Q \rangle$.*
2. *Let P be a sheaf in $\mathbf{Sh}(\mathbf{I})$. We have an isomorphism $P \xrightarrow{\sim} F_{\mathbf{I}}^{\mathbf{B}}\langle P \rangle$.* □

Prop. 4.4.4 says that the functor $F_{\mathbf{I}}^{\mathbf{B}} : \mathbf{Set}^{\mathbf{B}} \to \mathbf{Sh}(\mathbf{I})$ is essentially surjective: every sheaf in $\mathbf{Sh}(\mathbf{I})$ is isomorphic to a sheaf of the form $F_{\mathbf{I}}^{\mathbf{B}}Q$, for some $Q \in \mathbf{Set}^{\mathbf{B}}$.

$\mathbf{Sh}(\mathbf{I})$ is a Kleisli category. Prop. 4.4.4 is an important step towards establishing the following result; Fiore [2001], and Fiore and Menni [2005, Example 1.14], provide some details and discussion.

Theorem 4.4.5 (Fiore [2001, Thm. 6.2])**.** *$\mathbf{Sh}(\mathbf{I})$ is equivalent to the Kleisli category of the monad on $\mathbf{Set}^{\mathbf{B}}$ induced by the adjunction $F_{\mathbf{I}}^{\mathbf{B}} \dashv U_{\mathbf{B}}^{\mathbf{I}}$.* □

4.4.2 B-indexed early labelled transition systems

We introduce transition systems on presheaves in $\mathbf{Set}^{\mathbf{B}}$. By doing so we are able to significantly simplify the axiomatisation of transition systems given in Figure 3.4. Notice that Axioms **I**4–**I**6 are concerned only with inclusion maps. By considering transition systems over minimal seeds, we are thus able to eliminate these axioms.

We now introduce a class of indexed labelled transition systems whose carriers are sets of elements of presheaves over $\mathbf{B}$.

Definition 4.4.6. A $\mathbf{B}$-indexed early labelled transition system ($\mathbf{B}$-IL$_e$TS) over $Q \in \mathbf{Set}^{\mathbf{B}}$, is a labelled transition relation

$$\dashrightarrow \subseteq \int Q \times Lab_e \times \int Q \quad .$$

Figure 4.1 introduces Axioms **B**1–**B**3 on $\mathbf{B}$-IL$_e$TSs.

From B-IL$_e$TSs to I-IL$_e$TSs. Consider $P \in \mathbf{Sh}(\mathbf{I})$. Let $\dashrightarrow$ be a $\mathbf{B}$-IL$_e$TS over $\langle P \rangle$. We induce an $\mathbf{I}$-IL$_e$TS with carrier P, and with transition relation

$$\dashrightarrow_! \subseteq \int P \times Lab_e \times \int P$$

the least satisfying the following.

$$
\begin{array}{l}
\text{If } C_0 \vdash p \xrightarrow{c?d} C_0' \vdash p' \text{ and } C_0 \subseteq C \text{ and } C_0' \subseteq C \cup \{d\},\\
\quad \text{then } C \vdash [C_0 \hookrightarrow C]p \xrightarrow{c?d}_! C \cup \{d\} \vdash [C_0' \hookrightarrow (C \cup \{d\})]p'.\\
\text{If } C_0 \vdash p \xrightarrow{c!d} C_0' \vdash p' \text{ and } d \in C_0 \subseteq C \text{ and } C_0' \subseteq C,\\
\quad \text{then } C \vdash [C_0 \hookrightarrow C]p \xrightarrow{c!d}_! C \vdash [C_0' \hookrightarrow C]p'.\\
\text{If } C_0 \vdash p \xrightarrow{c!d} C_0' \vdash p' \text{ and } C_0 \subseteq C \text{ and } C_0' \subseteq C \cup \{d\} \text{ and } d \notin C\\
\quad \text{then } C \vdash [C_0 \hookrightarrow C]p \xrightarrow{c!d}_! C \cup \{d\} \vdash [C_0' \hookrightarrow (C \cup \{d\})]p'.\\
\text{If } C_0 \vdash p \xrightarrow{\tau} C_0' \vdash p' \text{ and } C_0 \subseteq C \text{ and } C_0' \subseteq C\\
\quad \text{then } C \vdash [C_0 \hookrightarrow C]p \xrightarrow{c!d}_! C \vdash [C_0' \hookrightarrow C]p'.
\end{array}
\tag{4.4.7}
$$

From I-IL$_e$TSs to B-IL$_e$TSs. Consider $P \in \mathbf{Sh}(\mathbf{I})$. Let $\longrightarrow$ be an **I**-IL$_e$TS over P. We induce a **B**-IL$_e$TS with carrier $\langle P \rangle$, and with transition relation

$$\langle\!\langle \longrightarrow \rangle\!\rangle \subseteq \int \langle P \rangle \times Lab_e \times \int \langle P \rangle$$

the least satisfying the following.

$$\text{If } C \vdash p \xrightarrow{\ell} C' \vdash p' \text{ then } \mathsf{supp}(p) \vdash \mathsf{seed}(p)\ \langle\!\langle \xrightarrow{\ell} \rangle\!\rangle\ \mathsf{supp}(p') \vdash \mathsf{seed}(p') \quad . \tag{4.4.8}$$

Theorem 4.4.9. *Consider a sheaf $P \in \mathbf{Sh}(\mathbf{I})$.*

1. *The above mappings (4.4.7, 4.4.8) describe a bijective correspondence between* **B**-IL$_e$TS*s over $\langle P \rangle$ that satisfy Axiom* **B**1*, and* **I**-IL$_e$TS*s over P that satisfy Axioms* **I**1*, and* **I**4*–*6*.*
2. *A relation $\dashrightarrow$ satisfies Axiom* **B**2 *(resp. Axiom* **B**3*) if and only if the induced relation $\dashrightarrow_!$ satisfies Axiom* **I**2 *(resp. Axiom* **I**3*).*

A proof of this result is provided in Appendix 4.B.

B1. Channel is known and at most transmitted data is learnt:

$$C \vdash q \xrightarrow{\ell} C' \vdash q' \implies \mathsf{ch}(\ell) \subseteq C \ \wedge\ C' \subseteq C \cup \mathsf{dat}(\ell)$$

B2. If one name can be input, then so can any other: for all $d' \in \mathcal{N}$:

$$\begin{array}{l} C \vdash q \xrightarrow{c?d} C' \vdash q' \\ \quad \implies \exists C'',\ q'' \in Q(C'').\ \ C \vdash q \xrightarrow{c?d'} C'' \vdash q'' \end{array}$$

B3. Bijective maps preserve transitions: for all $D \in \mathbf{I}$, with $C \cup \mathsf{dat}(\ell) \overset{\beta}{\cong} D$:

$$\begin{array}{l} C \vdash q \xrightarrow{\ell} C' \vdash q' \ \wedge\ \mathsf{ch}(\ell) \subseteq C \ \wedge\ C' \subseteq C \cup \mathsf{dat}(\ell) \\ \quad \implies \beta(C) \vdash [\beta|_C]q \xrightarrow{[\beta]\ell} \beta(C') \vdash [\beta|_{C'}]q' \end{array}$$

Figure 4.1: Requirements on a **B**-indexed labelled transition system.

4.A Appendix to Chapter 4: Proof of Theorem 4.2.5

We now provide a proof of Theorem 4.2.5.

Theorem 4.2.5. *Let (P,h) and (Q,k) be B_e-coalgebras. Suppose that P and Q both sheaves. Every* **I***-indexed early bisimulation R between induced* **I**-IL$_e$TS*s $\longrightarrow_h$ and $\longrightarrow_k$ is also a B_e-bisimulation between B_e-coalgebras (P,h) and (Q,k).*

Proof. We consider B_e-coalgebras (P,h) and (Q,k), and an **I**-indexed early bisimulation between the induced **I**-IL$_e$TSs, $\longrightarrow_h$ and $\longrightarrow_k$.

We define a new **I**-IL$_e$TS, with carrier R, and transition relation $\longrightarrow$ the least satisfying the following implication, for all $C \in \mathbf{I}$, and $(p,q) \in R(C)$, every label $\ell \in Lab_e$, and every pair $(p',q') \in R(C \cup \mathsf{dat}(\ell))$:

$$\begin{array}{l}\text{If } \forall C_0 \subseteq C,\ (p_0,q_0) \in R(C_0).\\ \quad (C - C_0) \cap \mathsf{dat}(\ell) = \emptyset \text{ and } [C_0 \hookrightarrow C](p_0,q_0) = (p,q) \implies\\ \qquad \exists (p_0',q_0') \in R(C_0 \cup \mathsf{dat}(\ell)).\quad [C_0 \cup \mathsf{dat}(\ell) \hookrightarrow C \cup \mathsf{dat}(\ell)](p_0',q_0') = (p',q')\\ \qquad\qquad \text{and } C_0 \vdash p_0 \xrightarrow{\ell}_h C_0 \cup \mathsf{dat}(\ell) \vdash p_0'\\ \qquad\qquad \text{and } C_0 \vdash q_0 \xrightarrow{\ell}_k C_0 \cup \mathsf{dat}(\ell) \vdash q_0'\\ \text{Then } C \vdash (p,q) \xrightarrow{\ell} C \cup \mathsf{dat}(\ell) \vdash (p',q') \qquad .\end{array}$$

Recall that, by Theorem 3.3.6, the **I**-IL$_e$TSs $(P, \longrightarrow_h)$ and $(Q, \longrightarrow_k)$ satisfy Axiom **I**1–**I**6. It follows straightforwardly from this, and from the definition, that the **I**-IL$_e$TS $(R, \longrightarrow)$ satisfies Axioms **I**1, **I**3, **I**4a, **I**4b and **I**6. Axioms **I**2 and **I**5 are more subtle.

We will use Axiom **I**5 to prove Axiom **I**2, and so we prove Axiom **I**5 first. Suppose that the premise of Axiom **I**5 holds, *i.e.*

$$C \vdash (p,q) \xrightarrow{\ell} C \cup \mathsf{dat}(\ell) \vdash (p',q')$$

and consider $D \in \mathbf{I}$ with $C \subseteq D$ and $(D - C) \cap \mathsf{dat}(\ell) = \emptyset$. We must show that the transition

$$D \vdash [C \hookrightarrow D](p,q) \xrightarrow{\ell} D \cup \mathsf{dat}(\ell) \vdash [C \cup \mathsf{dat}(\ell) \hookrightarrow D \cup \mathsf{dat}(\ell)](p',q')$$

is permitted. So suppose that we have $C_0 \subseteq D$ and $(p_0,q_0) \in R(C_0)$, all such that $(D - C_0) \cap \mathsf{dat}(\ell) = \emptyset$ and $[C_0 \hookrightarrow D](p_0,q_0) = [C \hookrightarrow D](p,q)$. Since both C_0 and C support $[C \hookrightarrow D]p \in P(D)$, we know, by Prop. 4.2.4, that $(C_0 \cap C)$ supports $[C \hookrightarrow D]p \in P(D)$; since P is a sheaf, we have $\mathsf{seed}(p@(C_0 \cap C)) \in P(C_0 \cap C)$ and

$$[(C_0 \cap C) \hookrightarrow C]\left(\mathsf{seed}(p@(C_0 \cap C))\right) = p \quad \text{and} \quad [(C_0 \cap C) \hookrightarrow C_0]\left(\mathsf{seed}(p@(C_0 \cap C))\right) = p_0 \quad .$$

By Axiom **I**6 for $(P, \longrightarrow_h)$, we know that $((C_0 \cap C) \cup \mathsf{dat}(\ell))$ supports p' and that

$$(C_0 \cap C) \vdash \mathsf{seed}(p@(C_0 \cap C)) \xrightarrow{\ell}_h (C_0 \cap C) \cup \mathsf{dat}(\ell) \vdash \mathsf{seed}(p'@(((C_0 \cap C) \cup \mathsf{dat}(\ell)))) \quad .$$

Let $p_0' = [(C_0 \cap C) \cup \mathsf{dat}(\ell) \hookrightarrow C_0 \cup \mathsf{dat}(\ell)]\ \left(\mathsf{seed}(p'@(C_0 \cap C) \cup \mathsf{dat}(\ell))\right)$. By Axiom **I**5, we have

$$C_0 \vdash p_0 \xrightarrow{\ell}_h C_0 \cup \mathsf{dat}(\ell) \vdash p_0' \quad .$$

In the same way, since Q is a sheaf, we have an element $q_0' \in Q(C_0 \cup \mathsf{dat}(\ell))$ such that $[C_0 \cup \mathsf{dat}(\ell) \hookrightarrow D \cup \mathsf{dat}(\ell)]q' = [C \cup \mathsf{dat}(\ell) \hookrightarrow D \cup \mathsf{dat}(\ell)]q$, and

$$C_0 \vdash q_0 \xrightarrow{\ell}_k C_0 \cup \mathsf{dat}(\ell) \vdash q_0' \quad .$$

It remains for us to show that $(p_0', q_0') \in R(C_0 \cup \mathsf{dat}(\ell))$. We show this differently depending on whether or not C_0 is smaller in size than C.

If C_0 is smaller than C then we can construct an injection $\iota : C_0 \cup \mathsf{dat}(\ell) \rightarrowtail C \cup \mathsf{dat}(\ell)$ and a bijection $\beta : D \cup \mathsf{dat}(\ell) \xrightarrow{\sim} D \cup \mathsf{dat}(\ell)$ such that $\beta|_{(C_0 \cap C) \cup \mathsf{dat}(\ell)} = \mathsf{id}_{(C_0 \cap C) \cup \mathsf{dat}(\ell)}$, and such that the following diagram commutes.

$$\begin{array}{ccc} C_0 \cup \mathsf{dat}(\ell) & \hookrightarrow & D \cup \mathsf{dat}(\ell) \\ \downarrow{\scriptstyle \iota} & & \downarrow{\scriptstyle \beta} \\ C \cup \mathsf{dat}(\ell) & \hookrightarrow & D \cup \mathsf{dat}(\ell) \end{array}$$

Consider the following sequence of identities between elements of $(P \times Q)(D \cup \mathsf{dat}(\ell))$.

$$\begin{aligned} [C \cup \mathsf{dat}(\ell) \hookrightarrow D \cup \mathsf{dat}(\ell)](p', q') &= [C_0 \cup \mathsf{dat}(\ell) \hookrightarrow D \cup \mathsf{dat}(\ell)](p_0', q_0') \\ &= [\beta][C_0 \cup \mathsf{dat}(\ell) \hookrightarrow D \cup \mathsf{dat}(\ell)](p_0', q_0') \qquad (*) \\ &= [C \cup \mathsf{dat}(\ell) \hookrightarrow D \cup \mathsf{dat}(\ell)][\iota](p_0', q_0') \end{aligned}$$

(Here, line $(*)$ holds because $((C_0 \cap C) \cup \mathsf{dat}(\ell))$ supports (p_0', q_0'), and because β fixes the set $((C_0 \cap C) \cup \mathsf{dat}(\ell))$.) The sheaves P and Q must both act injectively, and hence we can cancel $[C \cup \mathsf{dat}(\ell) \hookrightarrow D \cup \mathsf{dat}(\ell)]$ from both sides of the resulting equation to deduce that $(p', q') = [\iota](p_0', q_0')$. Since, by assumption, the transition

$$C \vdash (p, q) \xrightarrow{\ell} C \cup \mathsf{dat}(\ell) \vdash (p', q')$$

is allowed, we can conclude from the closure condition in the definition of $\longrightarrow$ that

$$[\iota|_{C_0 \cup \mathsf{dat}(\ell)}](p_0', q_0') \in R(\iota(C_0 \cup \mathsf{dat}(\ell)))$$

and so, since R is a functor, that

$$(p_0', q_0') = [(\iota|_{C_0 \cup \mathsf{dat}(\ell)})^{-1}][\iota|_{C_0 \cup \mathsf{dat}(\ell)}](p_0', q_0') \in R(C_0 \cup \mathsf{dat}(\ell))$$

as required.

If, on the other hand, C is smaller in size than C_0, then we can construct an injection $\iota : C \cup \mathsf{dat}(\ell) \rightarrowtail C_0 \cup \mathsf{dat}(\ell)$ and a bijection $\beta : D \cup \mathsf{dat}(\ell) \xrightarrow{\sim} D \cup \mathsf{dat}(\ell)$ such that we have $\beta|_{(C_0 \cap C) \cup \mathsf{dat}(\ell)} = \mathsf{id}_{(C_0 \cap C) \cup \mathsf{dat}(\ell)}$, and such that the following diagram commutes.

$$\begin{array}{ccc} C \cup \mathsf{dat}(\ell) & \hookrightarrow & D \cup \mathsf{dat}(\ell) \\ \downarrow{\scriptstyle \iota} & & \downarrow{\scriptstyle \beta} \\ C_0 \cup \mathsf{dat}(\ell) & \hookrightarrow & D \cup \mathsf{dat}(\ell) \end{array}$$

This time we consider the following sequence of identities between elements of the set $(P \times Q)(D \cup \mathsf{dat}(\ell))$.

$$\begin{aligned} [C_0 \cup \mathsf{dat}(\ell) \hookrightarrow D \cup \mathsf{dat}(\ell)](p_0', q_0') &= [C \cup \mathsf{dat}(\ell) \hookrightarrow D \cup \mathsf{dat}(\ell)](p', q') \\ &= [\beta][C \cup \mathsf{dat}(\ell) \hookrightarrow D \cup \mathsf{dat}(\ell)](p', q') \\ &= [C_0 \cup \mathsf{dat}(\ell) \hookrightarrow D \cup \mathsf{dat}(\ell)][\iota](p', q') \end{aligned}$$

We cancel $[C_0 \cup \mathsf{dat}(\ell) \hookrightarrow D \cup \mathsf{dat}(\ell)]$ from both sides of the resulting equation to deduce that $(p_0', q_0') = [\iota](p', q')$. Since $(p', q') \in R(C \cup \mathsf{dat}(\ell))$ we conclude that we have $(p_0', q_0') \in R(C_0 \cup \mathsf{dat}(\ell))$, as required. Thus Axiom **I5** is proved.

For Axiom **I2**, we suppose that we have a transition

$$C \vdash (p, q) \xrightarrow{c?d} C \cup \{d\} \vdash (p', q') \tag{4.A.1}$$

and we will show that, for any name $d' \in \mathcal{N}$, we have $(p'', q'') \in R(C \cup \{d'\})$ such that

$$C \vdash (p,q) \xrightarrow{c?d'} C \cup \{d'\} \vdash (p'', q'') \quad .$$

It follows from (4.A.1) that we must have the transition

$$C \vdash p \xrightarrow{c?d}_h C \cup \{d\} \vdash p' \quad .$$

By Axiom **I**4a, we have

$$C \cup \{d\} \vdash [C \hookrightarrow C \cup \{d\}]p \xrightarrow{c?d}_h C \cup \{d\} \vdash p' \quad .$$

Let n_0 be the smallest number for which there exists a subset C_0 of C of size n_0 that supports $(p,q) \in (P \times Q)(C)$ and such that

$$\big(\mathsf{seed}(p@C_0), \mathsf{seed}(q@C_0)\big) \in R(C_0) \quad .$$

We pick some such C_0 of this minimal size. By Axiom **I**6, we know that $C_0 \cup \{d\}$ supports p', and that

$$C_0 \cup \{d\} \vdash [C_0 \hookrightarrow C_0 \cup \{d\}]\,\big(\mathsf{seed}(p@C_0)\big) \xrightarrow{c?d}_h C_0 \cup \{d\} \vdash \mathsf{seed}(p'@(C_0 \cup \mathsf{dat}(\ell))) \quad .$$

Using Axiom **I**4a again, we deduce that

$$C_0 \vdash \mathsf{seed}(p@C_0) \xrightarrow{c?d}_h C_0 \cup \{d\} \vdash \mathsf{seed}(p'@(C_0 \cup \mathsf{dat}(\ell))) \quad .$$

Axiom **I**2 provides $p'_0 \in P(C_0 \cup \{d'\})$ such that

$$C_0 \vdash \mathsf{seed}(p@C_0) \xrightarrow{c?d'}_h C_0 \cup \{d'\} \vdash p'_0 \quad .$$

Since R is an **I**-indexed early bisimulation we have an element $q'_0 \in Q(C_0 \cup \{d'\})$ such that $(p'_0, q'_0) \in R(C_0 \cup \{d'\})$ and

$$C_0 \vdash \mathsf{seed}(q@C_0) \xrightarrow{c?d'}_h C_0 \cup \{d'\} \vdash q'_0 \quad .$$

It follows from the minimality of C_0 that the transition

$$C_0 \vdash \big(\mathsf{seed}(p@C_0), \mathsf{seed}(q@C_0)\big) \xrightarrow{c?d'} C_0 \cup \{d'\} \vdash (p'_0, q'_0)$$

is permitted. By Axioms **I**4a, **I**5, and **I**4a again, we conclude that

$$C \vdash (p,q) \xrightarrow{c?d'} C \cup \{d'\} \vdash [C_0 \cup \{d'\} \hookrightarrow C \cup \{d'\}](p'_0, q'_0) \quad .$$

Thus Axiom **I**2 is proved. So $(R, \longrightarrow)$ satisfies Axioms **I**1–**I**6, and by Theorem 3.3.7 we have an B_e-coalgebra structure $r : R \to B_e R$ that induces the relation $\longrightarrow$. In other words, we have $\longrightarrow_r = \longrightarrow$.

To conclude our proof of this theorem we must show that the B_e-coalgebra structure $r : R \to B_e R$ lifts the relation R to a span of coalgebras. We do this using the characterisation of Lemma 3.3.12: we will show that the properties of that lemma (properties 1–3) are satisfied. Property 1 is immediate from the definition of $\longrightarrow$. For property 2, we proceed as follows. Suppose that we have $(p,q) \in R(C)$ such that $C \vdash p \xrightarrow{\ell}_h C' \vdash p'$. We must exhibit $q' \in Q(C')$ such that $(p', q') \in R(C')$ and $C \vdash (p,q) \xrightarrow{\ell} C' \vdash (p', q')$. We will focus on the case where $\ell = \tau$. By Axiom **I**1, we know that $C' = C$. As in our proof of Axiom **I**2, we let n_0 be the smallest number

for which there exists a subset C_0 of C of size n_0 that supports $(p,q) \in (P \times Q)(C)$ and is such that $\big(\mathsf{seed}(p@C_0), \mathsf{seed}(q@C_0)\big) \in R(C_0)$. We pick some such minimal set C_0. By Axiom **I**6, C_0 supports p', and

$$C_0 \vdash \mathsf{seed}(p@C_0) \xrightarrow{\ell}_h C_0 \vdash \mathsf{seed}(p'@C_0) \quad .$$

Since R is an **I**-indexed early bisimulation we have an element $q'_0 \in Q(C_0)$ such that $\Big(\mathsf{seed}(p'@C_0), q'_0\Big) \in R(C_0)$ and

$$C_0 \vdash \mathsf{seed}(q@C_0) \xrightarrow{\ell}_k C_0 \vdash q'_0 \quad .$$

The transition

$$C_0 \vdash \big(\mathsf{seed}(p@C_0), \mathsf{seed}(q@C_0)\big) \xrightarrow{\ell} C_0 \vdash \Big(\mathsf{seed}(p'@C_0), q'_0\Big)$$

must be permitted, due to the minimality of C_0. Axiom **I**5 now ensures that we have a transition

$$C \vdash (p,q) \xrightarrow{\ell} C \vdash (p', [C_0 \hookrightarrow C]q'_0)$$

as required. Input and output labels are treated similarly, requiring Axioms **I**4a and **I**4b respectively. Thus property 2 of Lemma 3.3.12 is satisfied. Property 3 of Lemma 3.3.12 is established in a symmetric manner, and thus we can conclude that R is a B_e-bisimulation. □

4.B Appendix to Chapter 4: Proof of Theorem 4.4.9

We now provide a proof of Theorem 4.4.9.

Theorem 4.4.9. *Consider a sheaf $P \in \mathbf{Sh}(\mathbf{I})$.*

1. *The above mappings (4.4.7, 4.4.8) describe a bijective correspondence between* **B**-IL$_e$TS*s over* $\langle P \rangle$ *that satisfy Axiom* **B**1*, and* **I**-IL$_e$TS*s over P that satisfy Axioms* **I**1*, and* **I**4*–*I6*.*
2. *A relation $\dashrightarrow$ satisfies Axiom* **B**2 *(resp. Axiom* **B**3*) if and only if the induced relation $\dashrightarrow_!$ satisfies Axiom* **I**2 *(resp. Axiom* **I**3*).*

Proof. We begin by proving item (1). Let $\dashrightarrow$ be a **B**-IL$_e$TS over $\langle P \rangle$. We will show that

$$\dashrightarrow \;=\; \langle \dashrightarrow_! \rangle \quad . \tag{4.B.1}$$

(None of the axioms are needed for this part.)

First, we show that $\dashrightarrow \subseteq \langle \dashrightarrow_! \rangle$. We will consider here the case of output transitions, which is the most complex; other kinds of label are treated similarly. Suppose that

$$C_0 \vdash p \xdashrightarrow{c!d} C'_0 \vdash p' \quad .$$

Either $d \in C_0$, or $d \notin C_0$. In first case, $d \in C_0$, we let $C = C_0 \cup C'_0$. In the second case, $d \notin C_0$, we let $C = (C_0 \cup C'_0) - \{d\}$. In both cases, we have induced

$$C \vdash [C_0 \hookrightarrow C]p \xdashrightarrow{c!d}_! C \cup \{d\} \vdash [C'_0 \hookrightarrow (C \cup \{d\})]p' \quad .$$

Now, by definition of $\langle P \rangle$, $\mathsf{supp}([C_0 \hookrightarrow C]p) = C_0$, and $\mathsf{supp}([C'_0 \hookrightarrow (C \cup \{d\})]p') = C'_0$. So we have induced

$$C_0 \vdash p \;\langle \xdashrightarrow{c!d}_! \rangle\; C'_0 \vdash p'$$

as required.

Next, we show that $\langle \dashrightarrow_! \rangle \subseteq \dashrightarrow$. This time we focus on input transitions, although other kinds of label are treated similarly. Suppose that

$$C_0 \vdash p \;\langle \xdashrightarrow{c?d}_! \rangle\; C'_0 \vdash p' \quad .$$

Then we must have $C, C' \in \mathbf{I}$, and $q \in P(C)$, $q' \in P(C)$ such that

$$C_0 = \mathsf{supp}(q),\ p = \mathsf{seed}(q),\ C_0' = \mathsf{supp}(q'),\ p' = \mathsf{seed}(q')$$

and that

$$D \vdash q \xrightarrow{c?d}_! D' \vdash q' \quad .$$

We must have $E_0, E_0' \in \mathbf{B}$, and $r \in \langle P \rangle(E_0)$, and $r' \in \langle P \rangle(E_0')$, such that

$$[E_0 \hookrightarrow D]r = q \qquad \text{and} \qquad [E_0' \hookrightarrow D']r' = q' \qquad \text{and} \qquad E_0 \vdash r \xrightarrow{c?d} E_0' \vdash r' \quad .$$

Now, we must have that $E_0 = \mathsf{supp}(q)$, but we already have that $C_0 = \mathsf{supp}(q)$; so we know that $E_0 = C_0$. Similarly, $E_0' = C_0'$. Since P is separated, the map $[C_0 \hookrightarrow D]$ is injective, and so $p = r$. Similarly, $p' = r'$. So we must have

$$C_0 \vdash p \xrightarrow{c?d} C_0' \vdash p'$$

as required. Thus we establish equation 4.B.1.

Next we show that for every $\mathbf{B}$-IL$_e$TS $\dashrightarrow$ that satisfies Axiom **B**1, the relation $\dashrightarrow_!$ satisfies Axiom **I**1 and Axioms **I**4–**I**6. It is easy to see that Axiom **I**1 follows from Axiom **B**1 and the definition of $\dashrightarrow_!$, and that Axioms **I**4a, **I**4b and **I**5 all follow from the the definition of $\dashrightarrow_!$.

We now explain why Axiom **I**6 holds of $\dashrightarrow_!$. Suppose that we have, as in the premise of Axiom **I**6,

$$D \vdash [C \hookrightarrow D]p \xrightarrow{\ell}_! D \cup \{d\} \vdash p'$$

and, for now, we assume that $\ell = c?d$. Then, using equation 4.B.1, we must have that

$$\mathsf{supp}(p) \vdash \mathsf{seed}(p) \xrightarrow{\ell} \mathsf{supp}(p') \vdash \mathsf{seed}(p')$$

We know that $\mathsf{supp}(p) \subseteq C$, and, by Axiom **B**1, $\mathsf{supp}(p') \subseteq \mathsf{supp}(p) \cup \{d\}$. So we know that $\mathsf{supp}(p) \subseteq C \cup \{d\}$. Thus we have

$$C \vdash p \xrightarrow{\ell}_! C \cup \{d\} \vdash [\mathsf{supp}(p') \hookrightarrow C \cup \{d\}]\mathsf{seed}(p')$$

as required. Thus Axiom **I**6 holds of $\dashrightarrow_!$.

Let $\longrightarrow$ be a $\mathbf{I}$-IL$_e$TS over P that satisfies Axiom **I**1 and Axioms **I**4–**I**6. We will now show that

$$\longrightarrow \ = \ \langle \longrightarrow \rangle_! \quad . \tag{4.B.2}$$

We first show that $\longrightarrow \subseteq \langle \longrightarrow \rangle_!$. Suppose that

$$C \vdash p \xrightarrow{\ell} C' \vdash p' \quad .$$

We will concentrate on the case $\ell = c!d$ where $d \in C$. From Axiom **I**1, we know that $C' = C$. The following transition is induced:

$$\mathsf{supp}(p) \vdash \mathsf{seed}(p)\ \langle \xrightarrow{\ell} \rangle\ \mathsf{supp}(p') \vdash \mathsf{seed}(p') \quad .$$

By Axiom **I**4b, we know that $d \in \mathsf{supp}(p)$. Thus the transition

$$C \vdash p\ \langle \xrightarrow{\ell} \rangle_!\ C' \vdash p'$$

is induced. Other modes of communication are considered in a similar manner.

We now show that $\langle \longrightarrow \rangle_! \subseteq \longrightarrow$. Suppose that the transition

$$C \vdash p\ \langle \xrightarrow{\ell} \rangle_!\ C' \vdash p'$$

is induced. Suppose that $\ell = c?d$. We must have $C' = C \cup \{d\}$ and $C_0, C_0' \in \mathbf{B}$ together with $q \in \langle P \rangle(C_0)$, $q' \in \langle P \rangle(C_0')$ such that $C_0 \subseteq C$, $C_0' \subseteq (C \cup \{d\})$ and $[C_0 \hookrightarrow C]q = p$, $[C_0' \hookrightarrow C']q' = p'$ and

$$C_0 \vdash q \ \langle \xrightarrow{c?d} \rangle \ C_0' \vdash q' \quad .$$

This transition, in turn, must have been induced, so we must have $D, D' \in \mathbf{I}$ and $r \in P(D)$, $r' \in P(D')$ for which $\mathsf{supp}(r) = C_0$, $\mathsf{supp}(r') = C_0'$ and $\mathsf{seed}(r) = q$, $\mathsf{seed}(r') = q'$ and

$$D \vdash r \xrightarrow{c?d} D' \vdash r' \quad .$$

By Axiom **I**1 we have that $D' = D \cup \{d\}$. Using Axiom **I**4a we have

$$D \cup \{d\} \vdash [D \hookrightarrow (D \cup \{d\})]r \xrightarrow{c?d} D' \vdash r' \quad .$$

Using Axiom **I**6 we have

$$C_0 \cup \{d\} \vdash [C_0 \hookrightarrow (C_0 \cup \{d\})]q \xrightarrow{c?d} C_0 \cup \{d\} \vdash [C_0' \hookrightarrow (C_0 \cup \{d\})]q' \quad .$$

Using Axiom **I**5, we have

$$C \cup \{d\} \vdash [C \hookrightarrow (C \cup \{d\})]p \xrightarrow{c?d} C \cup \{d\} \vdash p' \quad .$$

We use Axiom **I**4a to conclude that

$$C \vdash p \xrightarrow{c?d} C \cup \{d\} \vdash p' \quad .$$

The other modes of communication are treated in a similar manner. Thus we establish equation 4.B.2.

Finally, we show that, provided Axioms **I**1 and **I**4–**I**6 hold of an **I**-IL$_e$TS $\longrightarrow$, then Axiom **B**1 holds of $\langle \longrightarrow \rangle$. Suppose that

$$C_0 \vdash p \ \langle \xrightarrow{\ell} \rangle \ C_0' \vdash p' \quad .$$

We will focus on the case where $\ell = c?d$; other modes of communication are considered in a similar manner.

We must have $C, C' \in \mathbf{I}$ together with $q \in P(C)$, $q' \in P(C')$ such that

$$C \vdash q \xrightarrow{\ell} C' \vdash q'$$

and so that $C_0 = \mathsf{supp}(q)$, $C_0' = \mathsf{supp}(q')$ and $p = \mathsf{seed}(q)$, $p' = \mathsf{seed}(q')$. By Axiom **I**1, we know that $C' = C \cup \{d\}$. Using Axiom **I**4a, we can assume, without loss of generality, that if $d \in C_0$ then $d \in C$, and that if $d \notin C_0$ then $d \notin C$. By Axiom **I**6, we have $q'' \in P(C_0 \cup \mathsf{dat}(\ell))$ such that

$$C_0 \vdash p \xrightarrow{\ell} C_0 \cup \{d\} \vdash q''$$

and $[(C_0 \cup \{d\}) \hookrightarrow C']q'' = q'$. Thus we know that $C_0' \subseteq C_0 \cup \{d\}$. By Axiom **I**1, we have that $c \in C_0$. Thus Axiom **B**1 holds of $\langle \longrightarrow \rangle$.

Item (2) follows straightforwardly from the definition (4.4.7). □

Chapter 5

Practicality

In Chapters 3 and 4 we introduced models for name-passing process calculi. One reason for pursuing a model theory is to develop efficient model checking procedures. In this chapter we take some steps in this direction.

An important idea in model-checking is to take advantage of any symmetry in a model. Clarke, Grumberg, and Peled [2000, Chapter 14] provide a general overview. For model-checking name-passing systems, *named-sets* have been proposed by Montanari and Pistore [*e.g.* 1997] and others in their work on *history dependent automata*. This work has led to efficient verification tools for name-passing systems [see *e.g.* Ferrari, Gnesi, Montanari, and Pistore, 2003]. The essential idea of named-sets is that the size of the state space can be drastically reduced by representing states up-to renaming of free variables.

In Section 5.1 we explain how a certain flavour of named-set provides an efficient presentation of the sheaves in the Schanuel topos, used in the models of Section 4.2.2, by taking advantage of the inherent symmetry there. We do this by establishing an equivalence between the Schanuel topos and a category of named-sets.

In Section 5.2 we move away from name-passing systems to consider final bisimulations in the general setting. We introduce a description of final bisimulations as greatest fixed points, and discuss criteria for the existence of final bisimulations. A transfinite procedure is introduced for finding such final bisimulations by repeatedly refining the coarsest relation. We relate this procedure with the well-known terminal sequence for an endofunctor. Furthermore, we provide conditions under which the procedure will find the final bisimulation on a coalgebra within a finite number of steps. One of these conditions is that the carrier of the coalgebra is finitely presentable: for sheaves in the Schanuel topos, finite presentability amounts to finiteness of the correponding named-set presentation.

Thus the two sections of this chapter together provide first steps towards an analysis of a partition refinement technique for checking bisimulation in the π-calculus, such as the algorithm suggested by Ferrari, Montanari, and Pistore [2002], perhaps eventually leading to the algorithm suggested by Pistore and Sangiorgi [2001].

In this chapter, and particularly in Section 5.1, we will make frequent reference to the theory of locally presentable categories, for which a reference is provided by Adámek and Rosický [1994]. For basic algebra, my reference is that of Cohn [2003].

5.1 Presentations of sheaves

For the transition systems that were studied in Chapters 3 and 4, the state spaces were either empty or infinite. (For instance, let 1 be a terminal presheaf in $\mathbf{Set}^{\mathbf{I}}$; then the set of elements $\int 1$ is in bijection with the set of objects of $\mathbf{I}$.) As such, the presheaf description of state spaces is not immediately useful for implementation.

In this section we recall some notions of named-set from the work of Montanari and Pistore [1997] and others. We find that named-sets provide an efficient description of sheaves in the Schanuel topos.

We begin this section with a discussion of some basic notions: categories of group actions, and coproduct completions. In Section 5.1.2 we use these notions to study named-sets with symmetries, concentrating on a category that we prove to be equivalent to the presheaf category $\mathbf{Set}^{\mathbf{B}}$ used in Section 4.4. Next, in Section 5.1.3, we explain that named-sets can be viewed as *presentations* of presheaves in $\mathbf{Set}^{\mathbf{B}}$, in the same way as generators and relations are presentations of algebraic structures. Finally, in Section 5.1.4, we introduce a category of named-sets that is equivalent to the Schanuel topos.

5.1.1 Preliminaries: Categories of coset actions, and coproduct completions

In this subsection we develop some basic notions. We begin by introducing categories of coset actions, which are equivalently categories of transitive group actions. We then discuss free coproduct completions, and the 'families' construction as an explicit description of this. In Theorem 5.1.3 we present the orbit-stabliser theorem as the statement that the category of group actions is a free coproduct completion of its subcategory of transitive actions.

Categories of coset actions. For any group G, the category **Coset-Act**(G) of left coset G-actions is defined as follows.

- Objects of **Coset-Act**(G) are subgroups of G.
- Morphisms in **Coset-Act**(G) from H to H' are given by left H'-cosets gH' which are such that $H \subseteq gH'g^{-1}$.
- The identity morphism on H is given by the coset H. The composition of a morphism $gH' : H \to H'$ with morphism $g'H'' : H' \to H''$ is given by $gg'H'' : H \to H''$.

Transitive group actions. The category **Coset-Act**(G) embeds into of the category $\mathbf{Set}^G$ of left G-sets, as (essentially) the full subcategory of transitive G-sets. We define a functor $e : \mathbf{Coset\text{-}Act}(G) \to \mathbf{Set}^G$ as follows. For any object of **Coset-Act**(G), *i.e.* a subgroup H of G, we let eH be the G-set of left cosets of H in G. A morphism $gH' : H \to H'$ in **Coset-Act**(G) determines a G-set homomorphism $e(gH) : eH \to eH'$, mapping a left coset $g'H$ of H to the left coset $g^{-1}g'gH'$ of H'.

It is routine to confirm that this action on morphisms respects the coset equivalence classes. Notice that the condition $H \subseteq gH'g^{-1}$ states that the stabiliser of the coset action $\mathrm{id}_G H$ must also stabilise gH'.

The use of cosets as morphisms in **Coset-Act**(G) ensures that the functor e is faithful. In fact, $e : \mathbf{Coset\text{-}Act}(G) \to \mathbf{Set}^G$ is also full. Consider subgroups H and H' of G. Every G-set homomorphism $f : eH \to eH'$ determines a coset gH' of H' found by applying f to the coset $\mathrm{id}_G H$. The property $H \subseteq gH'g^{-1}$ follows because the stabiliser of $\mathrm{id}_G H$ must be contained in gH', since f is a G-set homomorphism.

Free coproduct completions and the families construction. We write **COPROD** for the 2-category whose objects are categories with all small coproducts, morphisms are coproduct preserving functors, and 2-cells are natural transformations.

Recall [*e.g.* from Kock, 1995] that a functor $F : \mathbb{C} \to \mathscr{C}$ is a free coproduct completion if $\mathscr{C}$ has small coproducts, and if, for any other category $\mathscr{D}$ with coproducts, precomposition with F induces an equivalence of categories between $\mathbf{COPROD}(\mathscr{C}, \mathscr{D})$ and $\mathbf{CAT}(\mathbb{C}, \mathscr{D})$.

We now recall an explicit description of a free coproduct completion. For any small category $\mathbb{C}$ we have a category $\mathbf{Fam}(\mathbb{C})$ given as follows. Objects of $\mathbf{Fam}(\mathbb{C})$ are given by a set I together with an I-indexed family of objects from $\mathbb{C}$. A morphism in $\mathbf{Fam}(\mathbb{C})$ from $(I, \{X_i\}_{i\in I})$ to $(J, \{Y_j\}_{j\in J})$ is given by a function $f : I \to J$ together with, for each $i \in I$, a morphism $X_i \to Y_{f(i)}$ in $\mathbb{C}$.

Proposition 5.1.1. *Let* 1 *be a one element set. The functor* $\mathbb{C} \to \mathbf{Fam}(\mathbb{C})$ *which maps an object* $X \in \mathbb{C}$ *to the family* $(1, \{X\})$ *exhibits* $\mathbf{Fam}(\mathbb{C})$ *as a free coproduct completion of* $\mathbb{C}$. □

There is another, more intrinsic, characterisation of free coproduct completions. Recall that an object X of a locally small category $\mathscr{C}$ is said to be *indecomposable* if the hom-functor $\mathscr{C}(X, -) : \mathscr{C} \to \mathbf{Set}$ preserves coproducts.

Proposition 5.1.2 (see *e.g.* Carboni and Vitale, 1998, Lemma 42)**.** *Let* $\mathbb{C}$ *be a small category. A functor* $F : \mathbb{C} \to \mathscr{C}$ *is a free coproduct completion if and only if* $\mathscr{C}$ *has small coproducts,* F *is an embedding, the objects in the image of* F *are indecomposable, and every object in* $\mathscr{C}$ *is a coproduct of objects from* $\mathbb{C}$. □

Orbit-stabiliser theorem and free coproduct completions. Having exhibited the category $\mathbf{Coset}\text{-}\mathbf{Act}(G)$ as the full category of transitive G-sets, we can give a categorical account of the orbit-stabiliser theorem.

Theorem 5.1.3. *The embedding* $e : \mathbf{Coset}\text{-}\mathbf{Act}(G) \to \mathbf{Set}^G$ *exhibits the category* $\mathbf{Set}^G$ *of left* G*-sets as a free coproduct completion of* $\mathbf{Coset}\text{-}\mathbf{Act}(G)$.

Proof. We will prove this result by reference to Prop. 5.1.2. It is standard that $\mathbf{Set}^G$ has coproducts, and we have just explained that $e : \mathbf{Coset}\text{-}\mathbf{Act}(G) \to \mathbf{Set}^G$ is an embedding.

The coset actions are indecomposable for the following reason. Let $\{X_i\}_{i\in I}$ be a family of G-sets, and consider a subgroup H of G, together with a homomorphism $f : eH \to \coprod_{i\in I} X_i$. Suppose that $f(H) = \mathsf{inj}_i(x)$; then it is clear that f factors through the coproduct injection $\mathsf{inj}_i : X_i \to \coprod_{i\in I} X_i$.

The fact that every G-set is a coproduct of coset actions is precisely the orbit-stabiliser theorem. Consider a G-set $X \in \mathbf{Set}^G$. We write $\mathsf{Orb}(X)$ for a set of canonical representatives of the orbits of X, *i.e.* canonical representatives of equivalence classes for the relation $\sim$ on X given by

$$x \sim y \qquad \text{if and only if} \qquad \exists g \in G.\ x = g \bullet y \quad .$$

The stabiliser $\mathsf{Stab}(x)$ of $x \in X$ is the subgroup of G containing those elements that fix x. A standard result of group theory is the following:

$$X \;\cong\; \coprod_{x\in \mathsf{Orb}(X)} e(\mathsf{Stab}(x)) \quad .$$

Thus Theorem 5.1.3 is proved. □

5.1.2 Named-sets with symmetries

Montanari and Pistore [*e.g.* 1997] have developed models of name-passing systems using *history dependent automata*, *viz.* automata internal to a categories of so-called *named-sets*. A variety of categories of named-sets have been considered; among them are the categories of named-sets *with symmetries* which provide efficient representations of states by taking into account the symmetry involved in name-passing systems.

We begin this subsection with some remarks about the category of named-sets with symmetries that is studied in Pistore's thesis [1999]. As we explain, this category is equivalent to a category of group actions. We then go on to consider a more practically orientated version of named-sets with symmetries, inspired by the work of Ferrari, Montanari, and Pistore [2002]. We show that this latter category is equivalent to the presheaf category $\mathbf{Set}^{\mathbf{B}}$ that was considered in Section 4.4.

Named-sets with symmetries as in Pistore's thesis. A first version of named-sets with symmetries was studied by Pistore [1999, Chapter 7 and Defn. 9.3]. Let $\mathcal{N}$ be a countably infinite set, and let $\mathsf{Sym}(\mathcal{N})$ is the group of permutations on that set. It is straightforward to see that category **Sym** introduced in Pistore's Defn. 9.3 is the category **Coset-Act**$(\mathsf{Sym}(\mathcal{N}))$ that we introduced above. (Here we are assuming that the sets involved in Pistore's morphisms must not be empty; this point is not clear in his thesis.)

Pistore's category **SymSet** of named-sets with symmetries, introduced in his Defn. 9.3, is the category

$$\mathbf{Fam}(\mathbf{Coset\text{-}Act}(\mathsf{Sym}(\mathcal{N}))) \quad .$$

Using Theorem 5.1.3, we can conclude that Pistore's category **SymSet** of named-sets with symmetries is equivalent to the category $\mathbf{Set}^{\mathsf{Sym}(\mathcal{N})}$ of $\mathsf{Sym}(\mathcal{N})$-sets.

This category will appear again in Section 7.1, where we will see that it contains the Schanuel topos as a subcategory.

Named-sets with symmetries of Ferrari, Montanari, and Pistore. The named-sets with symmetries of Pistore's thesis have a major practical disadvantage: the group involved in their definition is infinite. In implementing a verification procedure for name-passing calculi, Ferrari, Montanari, and Pistore [2002] have used an alternative definition. For the category that they consider, we will eventually achieve an equivalence with the Schanuel topos.

First, though, we define the category $\mathbf{NSet}_{\mathbb{B}}$ by

$$\mathbf{NSet}_{\mathbb{B}} = \mathbf{Fam}\left(\coprod_{n\in\mathbb{N}} \mathbf{Coset\text{-}Act}(\mathsf{Sym}(n))\right).$$

(Here, for any number $n \in \mathbb{N}$, we write $\mathsf{Sym}(n)$ for the symmetric group on n symbols.) The objects of $\mathbf{NSet}_{\mathbb{B}}$ can be understood as tuples

$$\left(I, \{m_i\}_{i\in I}, \{H_i\}_{i\in I}\right),$$

where I is a set, and for all $i \in I$, m_i is a natural number and H_i is a subgroup of $\mathsf{Sym}(m_i)$. A morphism

$$\left(I, \{m_i\}_{i\in I}, \{H_i\}_{i\in I}\right) \to \left(J, \left\{n_j\right\}_{j\in J}, \left\{K_j\right\}_{j\in J}\right)$$

in $\mathbf{NSet}_{\mathbb{B}}$ is given by a pair $(f, \left\{\sigma_i K_{f(i)}\right\}_i)$ where f is a function $I \to J$, and for each $i \in I$, σ_i is a permutation in $\mathsf{Sym}(n_{f(i)})$, all such that, for each $i \in I$, $m_i = n_{f(i)}$ and $H_i \subseteq \sigma_i K_{f(i)} \sigma_i^{-1}$.

The named-sets considered by Ferrari *et al.* [2002] are defined in the same way, except that they require a total order on the indexing set. The morphisms in $\mathbf{NSet}_{\mathbb{B}}$ are different from the morphisms of Ferrari *et al.*, and this matter will be addressed in Section 5.1.4.

$\mathbf{NSet}_{\mathbb{B}}$ is a presheaf category. We now consider a skeleton category $\mathbb{B}$ of the category **B** (introduced in Section 4.4). Here, $\mathbb{B}$ is the groupoid whose objects are natural numbers, considered as sets, and whose morphisms are permutations on those sets. So $\mathbb{B}$ is the coproduct category $\coprod_{n\in\mathbb{N}} \mathsf{Sym}(n)$.

We now establish an equivalence between $\mathbf{NSet}_{\mathbb{B}}$ and $\mathbf{Set}^{\mathbb{B}}$.

Theorem 5.1.4. *The category* $\mathbf{NSet}_{\mathbb{B}}$ *is equivalent to* $\mathbf{Set}^{\mathbb{B}}$.

Proof. We prove the result by the following chain of equivalences.

$$\mathbf{NSet}_{\mathbb{B}} = \mathbf{Fam}\left(\coprod_{n\in\mathbb{N}} \mathbf{Coset\text{-}Act}(\mathsf{Sym}(n))\right) \tag{5.1.5a}$$

$$\cong \prod_{n\in\mathbb{N}} (\mathbf{Fam}\,(\mathbf{Coset\text{-}Act}(\mathsf{Sym}(n)))) \tag{5.1.5b}$$

$$\simeq \prod_{n\in\mathbb{N}} \left(\mathbf{Set}^{\mathsf{Sym}(n)}\right) \tag{5.1.5c}$$

$$\cong \mathbf{Set}^{(\coprod_n \mathsf{Sym}(n))} \tag{5.1.5d}$$

$$\cong \mathbf{Set}^{\mathbb{B}} \tag{5.1.5e}$$

Equation 5.1.5a is the definition of the category $\mathbf{NSet}_{\mathbb{B}}$. The next step (5.1.5b) relies on the fact that the **Fam** construction transforms coproducts of categories into products of categories. To see this, consider a set J and a J-indexed family of categories, $\left\{\mathscr{C}_j\right\}_{j\in J}$. We claim that the category $\mathbf{Fam}\left(\coprod_{j\in J}\mathscr{C}_j\right)$ is a product of the family $\left\{\mathbf{Fam}(\mathscr{C}_j)\right\}_{j\in J}$ of categories. We have a cone

$$\left\{\pi_j : \mathbf{Fam}\left(\coprod_{j\in J}\mathscr{C}_j\right) \to \mathbf{Fam}(\mathscr{C}_j)\right\}_{j\in J} .$$

For each $j\in J$, the functor π_j maps a family $(I,\{X_i\}_{i\in I})$ to the family

$$\left(I|_j,\ \{X_i\}_{i\in J}\right)$$

where $I|_j = \left\{i\in I \;\middle|\; X_i\in\mathscr{C}_j\right\}$. Each π_j acts by restriction on morphisms. It is straightforward to show that this cone is limiting.

Step (5.1.5c) follows immediately from Theorem 5.1.3. Step (5.1.5d) is a basic property of bicartesian closed categories (and **CAT** in particular, modulo size), and finally, (5.1.5d) is the definition of $\mathbb{B}$. □

We note here an explicit description of the functor $\mathbf{NSet}_{\mathbb{B}} \to \mathbf{Set}^{\mathbb{B}}$ suggested by (5.1.5). This functor takes a named-set $\left(I,\{m_i\}_{i\in I},\{H_i\}_{i\in I}\right)\in\mathbf{NSet}_{\mathbb{B}}$ to the presheaf $Q\in\mathbf{Set}^{\mathbb{B}}$ given as follows. For any number $n\in\mathbb{N}$, we let

$$Q(n) \;=\; \coprod_{i\in I}\left\{\sigma H_i \;\middle|\; n=m_i \text{ and } \sigma\in\mathsf{Sym}(n)\right\} \quad .$$

For any permutation τ on n, and any element $\mathsf{inj}_i(\sigma H_i)\in Q(n)$, we let

$$Q\tau(\mathsf{inj}_i(\sigma H_i)) \;=\; \mathsf{inj}_i(\tau\sigma H_i) \quad .$$

Suppose that we have two named-sets,

$$\left(I,\{m_i\}_{i\in I},\{H_i\}_{i\in I}\right) \quad \text{and} \quad \left(J,\left\{n_j\right\}_{j\in J},\left\{K_j\right\}_{j\in J}\right) \quad \text{in } \mathbf{NSet}_{\mathbb{B}}$$

with corresponding presheaves $Q,R\in\mathbf{Set}^{\mathbb{B}}$. A morphism

$$\left(f,\left\{\sigma_i K_{f(i)}\right\}_{i\in I}\right) : \left(I,\{m_i\}_{i\in I},\{H_i\}_{i\in I}\right) \to \left(J,\left\{n_j\right\}_{j\in J},\left\{K_j\right\}_{j\in J}\right) \quad \text{in } \mathbf{NSet}_{\mathbb{B}}$$

gives rise to a natural transformation $\alpha : Q \to R$ which is given by, for each number $n\in\mathbb{N}$, $\alpha_n(\mathsf{inj}_i(\sigma H_i)) = \mathsf{inj}_{f(i)}(\sigma_i\sigma\sigma_i^{-1}K_{f(i)})$.

5.1.3 Presentable named-sets

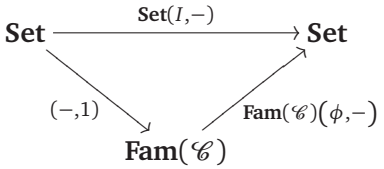
In algebra, structures are often specified by presentations; for example, a group can be specified by a set of generators and a set of relations. As we now explain, named-sets with symmetries can be viewed as presentations for presheaves in $\mathbf{Set}^{\mathbf{B}}$.

Presentable families. We begin with an elementary remark about presentable families in general.

Theorem 5.1.6. *Let λ be a regular cardinal, and let $\mathscr{C}$ be an arbitrary category with a terminal object. A family $(I, \{X_i\}_{i\in I})$ in $\mathbf{Fam}(\mathscr{C})$ is λ-presentable if and only if I has cardinality smaller than λ and for each $i \in I$ the object X_i is λ-presentable in $\mathscr{C}$.*

Proof. We begin by proving this statement from left to right. So we assume that a family $\phi = (I, \{X_i\}_{i\in I})$ in $\mathbf{Fam}(\mathscr{C})$ is λ-presentable.

The forgetful functor $\pi : \mathbf{Fam}(\mathscr{C}) \to \mathbf{Set}$, sending a family to its indexing set, has a right adjoint $(-,1) : \mathbf{Set} \to \mathbf{Fam}(\mathscr{C})$, sending a set I to the I-indexed family of terminal objects. In consequence, we see that the following diagram of categories commutes (up to isomorphism).



The functor $(-,1) : \mathbf{Set} \to \mathbf{Fam}(\mathscr{C})$ has itself a right adjoint, sending a family $(I, \{X_i\}_{i\in I})$ to the set of points of the family, *i.e.* the set

$$\coprod_{i\in I} \mathscr{C}(1, X_i) \quad .$$

Since ϕ is λ-presentable, the hom-functor $\mathbf{Fam}(\mathscr{C})(\phi, -) : \mathbf{Fam}(\mathscr{C}) \to \mathbf{Set}$ preserves λ-filtered colimits. Since $(-,1) : \mathbf{Set} \to \mathbf{Fam}(\mathscr{C})$ has a right adjoint, it preserves colimits, and so the hom-functor $\mathbf{Set}(I, -) : \mathbf{Set} \to \mathbf{Set}$ preserves λ-filtered colimits. So I is λ-presentable, *i.e.* has cardinality smaller than λ.

To complete the left-to-right direction, it remains to show that for each $i \in I$ the object X_i is λ-presentable in $\mathscr{C}$. Indeed, suppose that we have a λ-filtered diagram $D : \mathbf{C} \to \mathscr{C}$ that has a colimit L, and consider a morphism $f : X_i \to L$ in $\mathscr{C}$. Then we can construct another λ-filtered diagram $D' : \mathbf{C} \to \mathbf{Fam}(\mathscr{C})$ of the same shape, mapping an object $c \in \mathbf{C}$ to the family $D'c = (I, \{\text{if } j = i \text{ then } Dc \text{ else } 1\}_{j\in I})$. The colimit of this diagram D' is $(I, \{\text{if } j = i \text{ then } L \text{ else } 1\}_{j\in I})$. We also have a morphism $(\mathrm{id}_I, \{\text{if } j = i \text{ then } f : X_i \to Dc \text{ else } !_1 : X_j \to 1\}_{j\in I})$ from ϕ to that colimit. Since ϕ is λ-presentable, there is an indexing object $c \in \mathbf{C}$ and we can derive a morphism $f : X_i \to Dc$ in $\mathscr{C}$ making appropriate diagrams commute. Thus X_i is λ-presentable in $\mathscr{C}$.

Thus the statement of the theorem is proved from left to right. Before we prove from right to left, we will show that the embedding $\mathscr{C} \to \mathbf{Fam}(\mathscr{C})$ maps λ-presentable objects in $\mathscr{C}$ to λ-presentable objects in $\mathbf{Fam}(\mathscr{C})$. To this end, we assume that we have a λ-presentable X in $\mathscr{C}$, and show that the singleton family $(1, \{X\})$ in $\mathbf{Fam}(\mathscr{C})$ is λ-presentable in $\mathbf{Fam}(\mathscr{C})$.

Consider a λ-filtered diagram $D : \mathbf{C} \to \mathbf{Fam}(\mathscr{C})$ with a colimit $(I, \{L_i\}_{i\in I})$ in $\mathbf{Fam}(\mathscr{C})$, and consider a morphism $f : (1, \{X\}) \to (I, \{L_i\}_{i\in I})$ in $\mathbf{Fam}(\mathscr{C})$. This morphism f picks out an element $f(*)$ of I as well as a morphism $f_1 : X \to L_{f(*)}$.

We now form a new category $\mathbf{C}'$, which is a full subcategory of the category of elements

$$\int \left(\mathbf{C} \xrightarrow{D} \mathbf{Fam}(\mathscr{C}) \xrightarrow{\pi} \mathbf{Set} \right) \quad .$$

The objects in $\mathbf{C}'$ are those elements $(\mathsf{inj}_c(i))$ for which the leg $Dc \to (I, \{L_i\}_{i\in I})$ of the colimiting cone maps i to the chosen element $f(*)$. To see that $\mathbf{C}'$ is λ-filtered, note that $\pi : \mathbf{Fam}(\mathscr{C}) \to \mathbf{Set}$ preserves filtered colimits, and then use properties of λ-filtered colimits in **Set**.

We define a diagram $D' : \mathbf{C}' \to \mathscr{C}$ as follows: for any object $(\mathsf{inj}_c(i))$ in $\mathbf{C}'$, the object $D'(\mathsf{inj}_c(i))$ in $\mathscr{C}$ is defined to be the i-th component of the family Dc in $\mathbf{Fam}(\mathscr{C})$.

It is now straightforward to see that $L_{f(*)}$ is a colimit of this diagram $D' : \mathbf{C}' \to \mathscr{C}$. We have a morphism $f_1 : X \to L_{f(*)}$, and so, since X_i is λ-presentable, we have an object $(\mathsf{inj}_c(i))$ in $\mathbf{C}'$ together with a morphism f' from X to the i-th component of Dc, which is essentially unique in making appropriate diagrams commute in $\mathscr{C}$.

From this we derive a morphism $(i, \{f'\}) : (1, \{X\}) \to Dc$ in $\mathbf{Fam}(\mathscr{C})$ making appropriate diagrams commute there. The required uniqueness property of this morphism follows from the essential uniqueness of f', and thus we can conclude that $(1, \{X\})$ is λ-presentable in $\mathbf{Fam}(\mathscr{C})$.

We are now ready to prove the statement of the theorem from right to left. So we consider a family $\phi = (I, \{X_i\}_{i\in I})$ in $\mathbf{Fam}(\mathscr{C})$ for which I is smaller than λ, and for each $i \in I$ the object X_i is λ-presentable in $\mathscr{C}$.

This family $(I, \{X_i\}_{i\in I})$ is a coproduct of a diagram of λ-presentable objects, and that diagram λ-small— *i.e.*, the cardinality of the set of objects and morphisms of the diagram is smaller than λ. It is well-known that a λ-small colimit of λ-presentables is itself λ-presentable, and thus we can conclude that $(I, \{X_i\}_{i\in I})$ is indeed λ-presentable. □

Presentable named-sets.
Corollary 5.1.7. *Let λ be a regular cardinal. A named-set $\left(I, \{m_i\}_{i\in I}, \{H_i\}_{i\in I}\right) \in \mathbf{NSet}_{\mathbb{B}}$ is λ-presentable if and only if I is smaller than λ.*

Proof. Recall that the category $\mathbf{NSet}_{\mathbb{B}}$ is defined as

$$\mathbf{NSet}_{\mathbb{B}} = \mathbf{Fam}\left(\coprod_{n\in\mathbb{N}} \mathbf{Coset\text{-}Act}(\mathsf{Sym}(n))\right).$$

Thus this corollary follows from Theorem 5.1.6, provided we show that every object of the category $\left(\coprod_{n\in\mathbb{N}} \mathbf{Coset\text{-}Act}(\mathsf{Sym}(n))\right)$ is λ-presentable. Indeed, we will show that every object of $\left(\coprod_{n\in\mathbb{N}} \mathbf{Coset\text{-}Act}(\mathsf{Sym}(n))\right)$ is finitely presentable.

We begin by showing that, for every finite group G, all the objects of $\mathbf{Coset\text{-}Act}(G)$ are finitely presentable. Let H be a subgroup of G. Recall that we have an embedding $e : \mathbf{Coset\text{-}Act}(G) \to \mathbf{Set}^G$, and hence that eH is a model of the theory of G-sets: it is presented by one generator H, and for each element $g \in H$, an equation

$$gH = H \quad .$$

Thus eH is finitely presentable in $\mathbf{Set}^G$. By Theorem 5.1.3, the embedding $e : \mathbf{Coset\text{-}Act}(G) \to \mathbf{Set}^G$ is the free coproduct completion of eH, and so it follows from Theorem 5.1.6 that H is finitely presentable in $\mathbf{Coset\text{-}Act}(G)$.

For each $n \in \mathbb{N}$ the group $\mathsf{Sym}(n)$ is certainly finite, and so we know that every object of $\mathbf{Coset\text{-}Act}(\mathsf{Sym}(n))$ is finitely presentable. We are now ready to show that every object of the coproduct category $\left(\coprod_{n\in\mathbb{N}} \mathbf{Coset\text{-}Act}(\mathsf{Sym}(n))\right)$ is finitely presentable. To this end, consider some $m \in \mathbb{N}$ and a subgroup H of $\mathsf{Sym}(m)$.

We first observe that the following two functors are equal; this follows from the nature of coproducts of categories.

$$\mathbf{Coset\text{-}Act}(\mathsf{Sym}(m)) + \left(\coprod_{n\in\mathbb{N},n\neq m} \mathbf{Coset\text{-}Act}(\mathsf{Sym}(n))\right) \;\rightrightarrows\; \mathbf{Set}$$

$(\,\mathrm{Hom}(H,-)\,,\,\emptyset\,)$ and $\mathrm{Hom}(\mathsf{inl}(H),-)$

(Here, the left-hand functor maps an object K of $\mathbf{Coset\text{-}Act}(\mathsf{Sym}(m))$ to the hom-set $\mathrm{Hom}(H,K)$, and an object of $\left(\coprod_{n\in\mathbb{N},n\neq m} \mathbf{Coset\text{-}Act}(\mathsf{Sym}(n))\right)$ to the empty set. The right-hand functor maps any object X in the domain to the hom-set $\mathrm{Hom}(\mathsf{inl}(H),X)$.)

The object $H \in \mathbf{Coset\text{-}Act}(\mathsf{Sym}(m))$ is finitely presentable; in other words, the hom-functor $(\mathbf{Coset\text{-}Act}(\mathsf{Sym}(m)))(H,-) : \mathbf{Coset\text{-}Act}(\mathsf{Sym}(m)) \to \mathbf{Set}$ preserves filtered colimits. So too does the constant functor

$$\emptyset : \left(\coprod_{n\in\mathbb{N},n\neq m} \mathbf{Coset\text{-}Act}(\mathsf{Sym}(n))\right) \to \mathbf{Set} \quad .$$

Filtered diagrams are connected, and so we know that the copairing

$$(\,\mathrm{Hom}(H,-)\,,\,\emptyset\,) : \; \mathbf{Coset\text{-}Act}(\mathsf{Sym}(m)) + \left(\coprod_{n\in\mathbb{N},n\neq m} \mathbf{Coset\text{-}Act}(\mathsf{Sym}(n))\right) \to \mathbf{Set}$$

preserves filtered colimits. Thus the hom-functor

$$\mathrm{Hom}(\mathsf{inj}_m(H),-) : \left(\coprod_{n\in\mathbb{N}} \mathbf{Coset\text{-}Act}(\mathsf{Sym}(n))\right) \to \mathbf{Set}$$

preserves filtered colimits, and so the object $\mathsf{inj}_m(H)$ is finitely presentable.

We thus conclude, using Theorem 5.1.6, that a named-set in $\mathbf{NSet}_{\mathbb{B}}$ is λ-presentable if and only if its indexing set is smaller than λ. □

5.1.4 Named-sets with symmetries and the Schanuel topos

We now modify the morphisms between named-sets to match those suggested by Ferrari, Montanari, and Pistore [2002]. In doing so, we arrive at a category equivalent to the Schanuel topos.

In Theorem 4.4.5 we stated that the Schanuel topos is the Kleisli category of the monad on $\mathbf{Set}^{\mathbf{B}}$ that arises from the forgetful functor $U^{\mathbf{I}}_{\mathbf{B}} : \mathbf{Set}^{\mathbf{I}} \to \mathbf{Set}^{\mathbf{B}}$ and its left adjoint.

We begin this subsection by considering the Schanuel topos as a Kleisli category over the category of presheaves on the skeleton $\mathbb{B}$. We then introduce the modified morphisms between named-sets, and, in Theorem 5.1.9, we establish an equivalence between the resulting category of named-sets and the Schanuel topos. We conclude by discussing notions of presentability for the sheaves in the Schanuel topos.

A monad on $\mathbf{Set}^{\mathbb{B}}$. Let $\mathbb{I}$ be the category of natural numbers, considered as sets, with injections between them. Notice that $\mathbb{I}$ is a skeleton of $\mathbf{I}$.

The monad $U^{\mathbf{I}}_{\mathbf{B}}F^{\mathbf{B}}_{\mathbf{I}}$ on $\mathbf{Set}^{\mathbf{B}}$ lifts to a monad $U^{\mathbb{I}}_{\mathbb{B}}F^{\mathbb{B}}_{\mathbb{I}}$ on $\mathbf{Set}^{\mathbb{B}}$. Fiore [2001, Defn. 1.1] provides a description of this lifted monad: it is given on objects as follows. For $Q \in \mathbf{Set}^{\mathbb{B}}$ and $m \in \mathbb{N}$, let

$$U^{\mathbb{I}}_{\mathbb{B}}F^{\mathbb{B}}_{\mathbb{I}}Q(m) = \sum_{n\in\mathbb{N}} \big(\mathbb{I}(n,m)\cdot Qn\big)/_{\sim_n}$$

with $(\iota\sigma, p) \sim_n (\iota, Q\sigma(p))$ for any $\sigma \in \mathsf{Sym}(n)$. For $\sigma \in \mathsf{Sym}(m)$, let

$$U^{\mathbb{I}}_{\mathbb{B}} F^{\mathbb{B}}_{\mathbb{I}} Q(\sigma)(\mathsf{inj}_n \left[\iota, p\right]_{\sim_n}) = \mathsf{inj}_n \left[\sigma\iota, p\right]_{\sim_n}.$$

It follows from Theorem 4.4.5 that the Schanuel topos is equivalent to the Kleisli category for the monad $U^{\mathbb{I}}_{\mathbb{B}} F^{\mathbb{B}}_{\mathbb{I}}$ on $\mathbf{Set}^{\mathbb{B}}$.

Notation. Consider sets X, Y, Z. Let F be a set of functions $X \to Y$, and let G be a set of functions $Y \to Z$. We write GF for the set of all functions $X \to Z$ found by composing a function in F with one in G. When F has only one element, say f, we will write Gf for GF; and we adopt a similar convention when G has only one element.

Modified morphisms. We define another category of named-sets $\mathbf{NSet}_{\mathbb{I}}$ as follows. The objects of $\mathbf{NSet}_{\mathbb{I}}$ are the objects of $\mathbf{NSet}_{\mathbb{B}}$. A morphism from $\left(I, \{m_i\}_{i\in I}, \{H_i\}_{i\in I}\right)$ to $\left(J, \left\{n_j\right\}_{j\in J}, \left\{K_j\right\}_{j\in J}\right)$ in $\mathbf{NSet}_{\mathbb{I}}$ is given by a function $f : I \to J$ together with, for each $i \in I$, a non-empty set $\boldsymbol{l}_i$ of injections from $n_{f(i)}$ to m_i such that for each $\iota \in \boldsymbol{l}_i$,

$$\text{(a) } \boldsymbol{l}_i K_{f(i)} = \boldsymbol{l}_i \qquad \text{and} \qquad \text{(b) } H_i \boldsymbol{l}_i \subseteq \boldsymbol{l}_i. \tag{5.1.8}$$

The identity morphism on a named-set $\left(I, \{m_i\}_{i\in I}, \{H_i\}_{i\in I}\right)$ is the identity function $\mathsf{id} : I \to I$ together with, for each $i \in I$, the group H_i.

Composition of morphisms is as follows. Given two composable morphisms,

$$\begin{array}{c} \left(I, \{m_i\}_{i\in I}, \{H_i\}_{i\in I}\right) \\ \Big\downarrow {\scriptstyle (f, \{\iota_i\}_{i\in I})} \\ \left(I', \{m'_{i'}\}_{i'\in I'}, \{H'_{i'}\}_{i'\in I'}\right) \\ \Big\downarrow {\scriptstyle (g, \{J_{i'}\}_{i'\in I'})} \\ \left(I'', \{m''_{i''}\}_{i''\in I''}, \{H''_{i''}\}_{i''\in I''}\right) \end{array}$$

we let the composite be the morphism

$$\left(gf,\ \left\{\boldsymbol{l}_i \boldsymbol{J}_{f(i)}\right\}_{i\in I}\right) : \left(I, \{m_i\}_{i\in I}, \{H_i\}_{i\in I}\right) \to \left(I'', \{m''_{i''}\}_{i''\in I''}, \{H''_{i''}\}_{i''\in I''}\right) \quad .$$

This category $\mathbf{NSet}_{\mathbb{I}}$ of named-sets is essentially the category considered by Ferrari, Montanari, and Pistore [2002, Sec. 3.1]. We note two differences. Firstly, as mentioned above, the named-sets of Ferrari *et al.* are equipped with a total-order on the indexing set. This order structure, though, is not required to be respected by their morphisms, and so (assuming the axiom of choice) a category without this order structure is equivalent to one with it. The second difference is as follows: with the morphisms considered by Ferrari *et al.*, the sets of injections (here denoted $\boldsymbol{l}_i$) are not required to be non-empty. However, it seems that the development of their paper would work equally well with this restriction imposed.

$\mathbf{NSet}_{\mathbb{I}}$ is the Schanuel topos. Notice that any morphism in $\mathbf{NSet}_{\mathbb{B}}$

$$\left(f, \left\{\sigma_i K_{f(i)}\right\}_{i\in I}\right) : \left(I, \{m_i\}_{i\in I}, \{H_i\}_{i\in I}\right) \to \left(J, \left\{n_j\right\}_{j\in J}, \left\{K_j\right\}_{j\in J}\right)$$

is also a morphism between the same objects in $\mathbf{NSet}_{\mathbb{I}}$; for each $i \in I$, condition (a) of (5.1.8) is immediate, while condition (b) amounts to the condition $H_i \subseteq \sigma_i K_{f(i)} \sigma_i^{-1}$ that is imposed on morphisms in $\mathbf{NSet}_{\mathbb{B}}$. Thus we have a faithful functor $\mathbf{NSet}_{\mathbb{B}} \to \mathbf{NSet}_{\mathbb{I}}$.

We also have a functor $\mathbf{NSet}_{\mathbb{I}} \to \mathbf{Kl}(U^{\mathbb{I}}_{\mathbb{B}} F^{\mathbb{B}}_{\mathbb{I}})$ into the Schanuel topos. This functor acts on objects as the equivalence $\mathbf{NSet}_{\mathbb{B}} \simeq \mathbf{Set}^{\mathbb{B}}$ suggested by Theorem 5.1.4: a named-set $\left(I, \{m_i\}_{i\in I}, \{H_i\}_{i\in I}\right)$ in $\mathbf{NSet}_{\mathbb{B}}$ is mapped to the presheaf $Q \in \mathbf{Set}^{\mathbb{B}}$ given as follows. For any number $n \in \mathbb{N}$, we let

$$Q(n) \;=\; \coprod_{i\in I} \left\{ \sigma H_i \;\middle|\; n = m_i \text{ and } \sigma \in \mathsf{Sym}(n) \right\} \quad .$$

For any permutation τ on n, and any element $\mathsf{inj}_i(\sigma H_i) \in Q(n)$, we let

$$Q\tau(\mathsf{inj}_i(\sigma H_i)) \;=\; \mathsf{inj}_i(\tau\sigma H_i) \quad .$$

Suppose that we have two named-sets,

$$\left(I, \{m_i\}_{i\in I}, \{H_i\}_{i\in I}\right) \quad \text{and} \quad \left(J, \left\{n_j\right\}_{j\in J}, \left\{K_j\right\}_{j\in J}\right) \qquad \text{in } \mathbf{NSet}_{\mathbb{B}}$$

with corresponding presheaves $Q, R \in \mathbf{Set}^{\mathbb{B}}$. A morphism

$$(f, \{\boldsymbol{l}_i\}_{i\in I}) : \left(I, \{m_i\}_{i\in I}, \{H_i\}_{i\in I}\right) \to \left(J, \left\{n_j\right\}_{j\in J}, \left\{K_j\right\}_{j\in J}\right) \qquad \text{in } \mathbf{NSet}_{\mathbb{I}}$$

gives rise to a natural transformation $\alpha : Q \to U^{\mathbb{I}}_{\mathbb{B}} F^{\mathbb{B}}_{\mathbb{I}} R$ which is given by, for each number $n \in \mathbb{N}$,

$$\alpha_n(\mathsf{inj}_i(\sigma H_i)) = \mathsf{inj}_{n_{f(i)}} \left[\sigma\iota, \mathsf{inj}_{f(i)}(K_{f(i)})\right]_{\sim_{n_{f(i)}}}$$

for some $\iota \in \boldsymbol{l}_i$. Condition (a) in equation 5.1.8 ensures that it doesn't matter which injection is chosen.

It clear that the following diagram commutes.

$$\begin{array}{ccc} \mathbf{NSet}_{\mathbb{B}} & \longrightarrow & \mathbf{Set}^{\mathbb{B}} \\ \downarrow & & \downarrow \\ \mathbf{NSet}_{\mathbb{I}} & \longrightarrow & \mathbf{Kl}(U^{\mathbb{I}}_{\mathbb{B}} F^{\mathbb{B}}_{\mathbb{I}}) \end{array}$$

Theorem 5.1.9. *The functor* $\mathbf{NSet}_{\mathbb{I}} \to \mathbf{Kl}(U^{\mathbb{I}}_{\mathbb{B}} F^{\mathbb{B}}_{\mathbb{I}})$ *is an equivalence of categories.*

Proof. This functor is essentially surjective, since, by Theorem 5.1.4, the restriction $\mathbf{NSet}_{\mathbb{B}} \to \mathbf{Set}^{\mathbb{B}}$ is essentially surjective. So it remains for us to show that the functor is full and faithful. To this end, we consider named-sets $\left(I, \{m_i\}_{i\in I}, \{H_i\}_{i\in I}\right)$ and $\left(J, \left\{n_j\right\}_{j\in J}, \left\{K_j\right\}_{j\in J}\right)$ in $\mathbf{NSet}_{\mathbb{I}}$, with corresponding presheaves $Q, R \in \mathbf{Set}^{\mathbb{B}}$.

To see that the functor in question is full, consider a natural transformation $\alpha : Q \to U^{\mathbb{I}}_{\mathbb{B}} F^{\mathbb{B}}_{\mathbb{I}} R$ between presheaves on $\mathbb{B}$. We define a morphism of named sets as follows. For each $i \in I$, we must have an element $j \in J$, an injection $\iota : n_j \to m_i$, and a permutation $\sigma \in \mathsf{Sym}(n_j)$ such that

$$\alpha_{m_i}(\mathsf{inj}_i(\mathsf{id}_{H_i})) = \mathsf{inj}_{n_j} \left[\iota, \mathsf{inj}_j(\sigma K_j)\right]_{\sim_{n_j}} \quad .$$

We let $f(i) = j$, and define $\boldsymbol{l}_i$ to be the set $\iota\sigma K_j$ of injections $n_j \rightarrowtail m_i$. This definition is unambiguous, by definition of the equivalence relation $\sim_{n_j}$.

Condition (a) of (5.1.8) is satisfied straightforwardly. As for condition (b): consider an element $\tau \in H_i$. Since $\tau H_i = H_i$, we know that

$$\alpha_{m_i}(\mathsf{inj}_i(\tau H_i)) = \mathsf{inj}_{n_j} \left[\iota, \mathsf{inj}_j(\sigma K_j)\right]_{\sim_{n_j}} \quad .$$

In other words, $(\tau\iota, \mathsf{inj}_j(\sigma K_j)) \sim_{n_j} (\iota, \mathsf{inj}_j(\sigma K_j))$. So we know that there is $\tau' \in \mathsf{Sym}(K_j)$ such that $\tau\iota = \iota\tau'$, and with $\tau'\sigma K_j = \sigma K_j$. So

$$\tau\iota\sigma K_j \;=\; \iota\tau'\sigma K_j \;=\; \iota\sigma K_j$$

and we can conclude that $H_i \boldsymbol{l}_i \subseteq \boldsymbol{l}_i$.

In this way, we derive a morphism

$$(f, \{\boldsymbol{l}_i\}_{i\in I}) : \left(I, \{m_i\}_{i\in I}, \{H_i\}_{i\in I}\right) \to \left(J, \left\{n_j\right\}_{j\in J}, \left\{K_j\right\}_{j\in J}\right)$$

in $\mathbf{NSet}_{\mathbb{I}}$. It is not hard to see that the functor $\mathbf{NSet}_{\mathbb{I}} \to \mathbf{Kl}(U^{\mathbb{I}}_{\mathbb{B}} F^{\mathbb{B}}_{\mathbb{I}})$ maps this morphism to the natural transformation $\alpha : Q \to U^{\mathbb{I}}_{\mathbb{B}} F^{\mathbb{B}}_{\mathbb{I}} R$. Thus this functor is full.

To see that the functor $\mathbf{NSet}_{\mathbb{I}} \to \mathbf{Kl}(U^{\mathbb{I}}_{\mathbb{B}} F^{\mathbb{B}}_{\mathbb{I}})$ is faithful, consider two morphisms

$$(f, \{\boldsymbol{l}_i\}_{i\in I}),\ (g, \left\{\boldsymbol{J}_i\right\}_{i\in I}) \;:\; \left(I, \{m_i\}_{i\in I}, \{H_i\}_{i\in I}\right) \to \left(J, \left\{n_j\right\}_{j\in J}, \left\{K_j\right\}_{j\in J}\right)$$

in $\mathbf{NSet}_{\mathbb{I}}$ with common domain and codomain. Suppose that the functor maps these morphisms respectively to natural transformations $\alpha, \beta : Q \to U^{\mathbb{I}}_{\mathbb{B}} F^{\mathbb{B}}_{\mathbb{I}} R$ between the corresponding presheaves on $\mathbb{B}$, and suppose that $\alpha = \beta$. We must show that $(f, \{\boldsymbol{l}_i\}_{i\in I}) = (g, \left\{\boldsymbol{J}_i\right\}_{i\in I})$.

Consider some $i \in I$; we will show that $f(i) = g(i)$ and that $\boldsymbol{l}_i = \boldsymbol{J}_i$. Since $\alpha = \beta$, we know that for every $\iota \in \boldsymbol{l}_i$, we have

$$\mathsf{inj}_{n_{f(i)}}\left[\iota, \mathsf{inj}_{f(i)}(K_{f(i)})\right]_{\sim_{n_{f(i)}}} \;=\; \mathsf{inj}_{n_{g(i)}}\left[\jmath, \mathsf{inj}_{g(i)}(K_{g(i)})\right]_{\sim_{n_{g(i)}}}$$

for some $\jmath \in \boldsymbol{l}_i$. So we know that $n_{f(i)} = n_{g(i)}$, and that

$$(\iota, \mathsf{inj}_{f(i)}(K_{f(i)})) \sim_{n_{f(i)}} (\jmath, \mathsf{inj}_{g(i)}(K_{g(i)})) \quad .$$

So, by definition of the equivalence relation $(\sim_{n_{f(i)}})$, we know that $f(i) = g(i)$, and that there is a permutation $\sigma \in \mathsf{Sym}(n_{f(i)})$ such that $\iota\sigma = \jmath$ and $\sigma \in K_{f(i)}$. Using condition (a) of (5.1.8), we can conclude that $\iota \in \boldsymbol{J}_i$. Thus we have shown that, for each $i \in I$, $f(i) = g(i)$ and $\boldsymbol{l}_i \subseteq \boldsymbol{J}_i$; in a symmetric way one shows that also $\boldsymbol{J}_i \subseteq \boldsymbol{l}_i$. Hence we can conclude that the two named-sets morphisms are equal:

$$(f, \{\boldsymbol{l}_i\}_{i\in I}) = (g, \left\{\boldsymbol{J}_i\right\}_{i\in I}) \quad .$$

Thus the functor $\mathbf{NSet}_{\mathbb{I}} \to \mathbf{Kl}(U^{\mathbb{I}}_{\mathbb{B}} F^{\mathbb{B}}_{\mathbb{I}})$ is an equivalence of categories. □

Theorem 5.1.9 has also been established (independently) by Gadducci, Miculan, and Montanari [2006]. (Combine Prop. 29, Thm. 36, and Corollary 37 of their article.)

Presentability. Corollary 5.1.7 characterises presentable objects in $\mathbf{NSet}_{\mathbb{B}}$. To conclude this section, we now show that this characterisation extends to presentable objects for the category $\mathbf{NSet}_{\mathbb{I}}$ of named-sets with modified morphisms.

To begin, we provide a general result. We consider a condition on monads that is a weak form of the Cartesianness studied by Burroni [1971]. Under this condition, the canonical functor into the Kleisli category reflects presentable objects.

Proposition 5.1.10. *Let λ be a regular cardinal. Let T be a monad on a category $\mathscr{C}$, for which the naturality squares of the unit of T are pullbacks. If an object $X \in \mathscr{C}$ is λ-presentable in the Kleisli category $\mathbf{Kl}(T)$ then X is also λ-presentable in $\mathscr{C}$.*

Proof. Suppose that $X \in \mathscr{C}$ is λ-presentable in $\mathbf{Kl}(T)$, and consider a λ-filtered diagram $D : \mathbf{C} \to \mathscr{C}$ that has a colimit $L \in \mathscr{C}$, together with a morphism $f : X \to L$. The functor $\mathscr{C} \to \mathbf{Kl}(T)$ has a right adjoint, and so preserves colimits; hence L is a colimit of the diagram $\mathbf{C} \xrightarrow{D} \mathscr{C} \longrightarrow \mathbf{Kl}(T)$. By considering the morphism $f : X \to L$ as a morphism in $\mathbf{Kl}(T)$, we have an object $c \in \mathbf{C}$ and a sufficiently unique morphism $f' : X \to Dc$ in $\mathbf{Kl}(T)$, so that the following triangle commutes in $\mathbf{Kl}(T)$.

$$\begin{array}{ccc} & & Dc \\ & \overset{f'}{\nearrow} & \downarrow \\ X & \xrightarrow[f]{} & L \end{array}$$

(Here, the vertical arrow is part of the colimiting cocone.) So the following diagram commutes in $\mathscr{C}$.

$$\begin{array}{ccccc} & & & & T(Dc) \\ & & \overset{f'}{\nearrow} & & \downarrow \\ X & \xrightarrow[f]{} & L & \xrightarrow[\eta L]{} & T(L) \end{array} \tag{5.1.11}$$

Now, by assumption, the naturality square (in $\mathscr{C}$)

$$\begin{array}{ccc} Dc & \xrightarrow{\eta(Dc)} & T(Dc) \\ \downarrow & & \downarrow \\ L & \xrightarrow[\eta L]{} & TL \end{array}$$

is a pullback, while diagram 5.1.11 is a cone for the same diagram. Thus we have a morphism $f'' : X \to Dc$ in $\mathscr{C}$ making the following diagram commute (in $\mathscr{C}$).

$$\begin{array}{ccccc} & & X & & \\ & \swarrow^{f} & \downarrow{\scriptstyle f''} & \searrow^{f'} & \\ L & \longleftarrow & Dc & \xrightarrow[\eta(Dc)]{} & T(Dc) \end{array}$$

To conclude that X is λ-presentable in $\mathscr{C}$, one must show that the morphism $f'' : X \to Dc$ is appropriately unique. This follows from the fact that that f' is appropriately unique in $\mathbf{Kl}(T)$. Here it is helpful to note that η is monic: this is true since for any object Y in $\mathscr{C}$ the naturality square (in $\mathscr{C}$)

$$\begin{array}{ccc} Y & \xrightarrow{\eta Y} & TY \\ {\scriptstyle \eta Y}\downarrow & & \downarrow{\scriptstyle T\eta Y} \\ TY & \xrightarrow[\eta TY]{} & TTY \end{array}$$

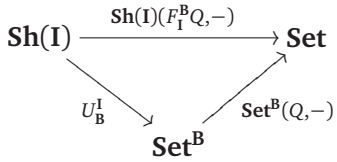
is a pullback. □

Corollary 5.1.12. *Let λ be a regular cardinal. A named-set $\left(I, \{m_i\}_{i\in I}, \{H_i\}_{i\in I}\right)$ is λ-presentable in $\mathbf{NSet}_{\mathbb{I}}$ if and only if the index I is smaller than λ.*

Proof. We already know, from Corollary 5.1.7, that a named-set $\left(I, \{m_i\}_{i\in I}, \{H_i\}_{i\in I}\right)$ is λ-presentable in $\mathbf{NSet}_{\mathbb{B}}$ (note the subscript) if and only if the index I is smaller than λ. To conclude the corollary, we will work across the equivalences introduced in Theorems 5.1.4 and 5.1.9, and furthermore work with the corresponding non-skeletal categories, to show that

$$\text{a presheaf } Q \in \mathbf{Set}^{\mathbf{B}} \text{ is } \lambda\text{-presentable if and only if } F_{\mathbf{I}}^{\mathbf{B}} P \text{ is } \lambda\text{-presentable in } \mathbf{Sh}(\mathbf{I}). \qquad (*)$$

To prove statement $(*)$ from left to right, we proceed as follows. Consider a presheaf $Q \in \mathbf{Set}^{\mathbf{B}}$. Notice that, since $F_{\mathbf{I}}^{\mathbf{B}}$ is left adjoint to $U_{\mathbf{B}}^{\mathbf{I}}$, the following diagram commutes.



If Q is λ-presentable, then the hom-functor $\mathbf{Set}^{\mathbf{B}}(Q,-) : \mathbf{Set}^{\mathbf{B}} \to \mathbf{Set}$ preserves λ-filtered colimits. The inclusion functor $j_{\mathbf{I}}^{\mathbf{B}} : \mathbf{B} \to \mathbf{I}$ reflects covers (in the sense of Definition 4.1.2), when $\mathbf{B}$ is considered with the coverage generated by all identity morphisms, and $\mathbf{I}$ is considered with the coverage $\mathscr{T}_{\mathbf{I}}$. Hence, by Prop. 4.1.4, the forgetful functor $U_{\mathbf{B}}^{\mathbf{I}} : \mathbf{Sh}(\mathbf{I}) \to \mathbf{Set}^{\mathbf{B}}$ has a right adjoint, namely $(j_{\mathbf{I}}^{\mathbf{B}})_* : \mathbf{Set}^{\mathbf{B}} \to \mathbf{Sh}(\mathbf{I})$. Thus $U_{\mathbf{B}}^{\mathbf{I}}$ preserves colimits, and we can conclude that the hom-functor $\mathbf{Sh}(\mathbf{I})(F_{\mathbf{I}}^{\mathbf{B}}Q,-) : \mathbf{Sh}(\mathbf{I}) \to \mathbf{Set}$ preserves λ-filtered colimits, and so that $F_{\mathbf{I}}^{\mathbf{B}}Q$ is λ-presentable.

To prove statement $(*)$ from right to left, we will appeal to Prop. 5.1.10, so it remains for us to show that the naturality diagrams of the unit of the monad $U_{\mathbf{B}}^{\mathbf{I}}F_{\mathbf{I}}^{\mathbf{B}}$ on $\mathbf{Set}^{\mathbf{B}}$ are pullbacks. (In fact, the monad $U_{\mathbf{B}}^{\mathbf{I}}F_{\mathbf{I}}^{\mathbf{B}}$ on $\mathbf{Set}^{\mathbf{B}}$ is Cartesian, but we do not need this here.)

First, we give an explicit description of the unit $\eta : 1 \to U_{\mathbf{B}}^{\mathbf{I}}F_{\mathbf{I}}^{\mathbf{B}}$. For any $Q \in \mathbf{Set}^{\mathbf{B}}$, the unit $\eta_Q : Q \to U_{\mathbf{B}}^{\mathbf{I}}F_{\mathbf{I}}^{\mathbf{B}}$ is the natural transformation with components

$$\left\{ (\eta_Q)_C : Q(C) \to \coprod_{D \subseteq C} Q(D) \right\}_{C \in \mathbf{B}}$$

given by, for each $C \in \mathbf{B}$ and each $q \in QC$,

$$(\eta_Q)_C(q) \;=\; \mathsf{inj}_C(q) \quad .$$

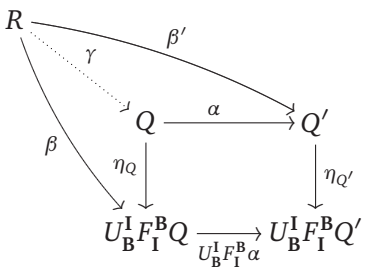
Consider presheaves $Q, Q' \in \mathbf{Set}^{\mathbf{B}}$, and a natural transformation $\alpha : Q \to Q'$. We will show that the naturality square

$$\begin{array}{ccc} Q & \xrightarrow{\alpha} & Q' \\ {\scriptstyle \eta_Q}\downarrow & & \downarrow{\scriptstyle \eta_{Q'}} \\ U_{\mathbf{B}}^{\mathbf{I}}F_{\mathbf{I}}^{\mathbf{B}}Q & \xrightarrow[U_{\mathbf{B}}^{\mathbf{I}}F_{\mathbf{I}}^{\mathbf{B}}\alpha]{} & U_{\mathbf{B}}^{\mathbf{I}}F_{\mathbf{I}}^{\mathbf{B}}Q' \end{array}$$

is a pullback. To this end, suppose that we have $R \in \mathbf{Set}^{\mathbf{B}}$ together with natural transformations, $\beta : R \to Q$ and $\beta' : R \to Q'$, making the following diagram commute.

$$\begin{array}{ccc} R & \xrightarrow{\beta'} & Q' \\ {\scriptstyle \beta}\downarrow & & \downarrow{\scriptstyle \eta_{Q'}} \\ U_{\mathbf{B}}^{\mathbf{I}}F_{\mathbf{I}}^{\mathbf{B}}Q & \xrightarrow[U_{\mathbf{B}}^{\mathbf{I}}F_{\mathbf{I}}^{\mathbf{B}}\alpha]{} & U_{\mathbf{B}}^{\mathbf{I}}F_{\mathbf{I}}^{\mathbf{B}}Q' \end{array} \tag{5.1.13a}$$

We must show that there is a unique natural transformation $\gamma : R \to Q$ making the following diagram commute.



(5.1.13b)

Since diagram 5.1.13a commutes, we know for each $C \in \mathbf{B}$ and every $r \in R(C)$, that if $\beta_C(r) = \mathsf{inj}_D q$, then $\mathsf{inj}_C(\beta'_C(r)) = \mathsf{inj}_D(\alpha_D(q))$. Thus we know that $C = D$, and so that $q \in Q(C)$. So we let $\gamma_C(r) = q$.

Clearly, a family $\{\gamma_C\}_{C\in\mathbf{B}}$ defined in this way will be natural. It is the unique natural transformation making diagram 5.1.13b commute, since the unit η_Q is manifestly monic.

Thus statement $(*)$ is proved. □

5.2 Final bisimulations

This section is comprised of two parts. In the first, Section 5.2.1, we discuss the existence of final bisimulations. In the second, Section 5.2.2, we introduce a procedure for finding them.

Section 5.2.1 is devoted to the redevelopment, within a general coalgebraic setting, of the usual greatest-fixed-point characterisation of bisimulation [for a conventional presentation, see *e.g.* Milner, 1989, Sec. 4.6]. We define a monotone operator on the preorder of relations for which bisimulations are exactly the prefixed points. Under certain assumptions, the preorder of relations forms a complete lattice, and hence the existence of final bisimulations can be concluded from Tarski's fixed point theorem.

Section 5.2.2 is concerned with an analysis of the construction of fixed points. We introduce a transfinite relation refinement sequence that relates to the well-known terminal sequence. We provide conditions under which the relation refinement sequence will converge in a finite number of steps.

Throughout this section we fix a category $\mathscr{C}$ with finite limits and images [see *e.g.* Johnstone, 2002, Sec. A1.3], and consider an arbitrary endofunctor B on $\mathscr{C}$. We let $h : X \to BX$, $k : Y \to BY$ be two B-coalgebras.

We will consider the full subcategory $\mathbf{Rel}(X, Y)$ of $\mathscr{C}/(X \times Y)$ whose objects are pairs of an object R of $\mathscr{C}$ and a monomorphism $R \rightarrowtail X \times Y$. As is common, we will often elide the monomorphism, denoting objects simply by their object parts. As usual, this full subcategory is a preorder; we write $(\subsetneq)$ for the preorder relation.

5.2.1 Final bisimulations

We begin this subsection by defining an operator Φ on the preorder $\mathbf{Rel}(X, Y)$. We then explore some properties of Φ. We conclude this subsection by explaining how Tarski's fixed point theorem can be used to derive final bisimulations.

For any $R \in \mathbf{Rel}(X, Y)$, we define $\Phi R \in \mathbf{Rel}(X, Y)$ as follows. First, let LR be the following pullback in $\mathscr{C}$.

$$\begin{array}{ccc} LR & \longrightarrow & BR \\ \downarrow & & \downarrow \\ & & B(X \times Y) \\ & & \downarrow{\scriptstyle (B\pi_1, B\pi_2)} \\ X \times Y & \xrightarrow[h\times k]{} & BX \times BY \end{array} \tag{5.2.1}$$

We let $\Phi R \rightarrowtail X \times Y$ be the image of the pullback map $LR \to X \times Y$. (So the pullback map factors through ΦR, and moreover ΦR is the least such relation, under the inclusion preorder $\subsetneq$.)

It follows from the universal properties of pullbacks and images that the operator Φ on $\mathbf{Rel}(X, Y)$ is monotone.

Suppose that $\mathscr{C}$ is the category **Set** of sets, and that the endofunctor B is the endofunctor $B_{\mathrm{lts}} = \mathscr{P}(\mathrm{Lab} \times -)$ for labelled transitions systems suggested in Section 2.1. For any rela-

tion $R \subseteq X \times Y$, the refined relation $\Phi R \subseteq X \times Y$ is the following set.

$$\Phi R = \left\{ (x,y) \in X \times Y \;\middle|\; \begin{array}{l} \forall (l,x') \in h(x).\ \exists y' \in Y.\ (l,y') \in k(y) \text{ and } (x',y') \in R \\ \forall (l,y') \in k(y).\ \exists x' \in X.\ (l,x') \in h(x) \text{ and } (x',y') \in R \end{array} \right\}$$

Thus the operator Φ is the construction $\mathcal{F}$ defined by Milner [1989, Defn. 4.7].

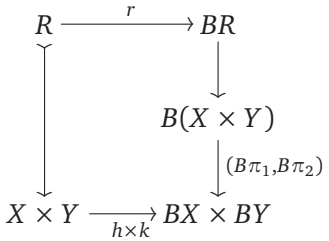
Regular relations and bisimulations. We will say that a relation $R \in \mathbf{Rel}(X,Y)$ is *regular* if the corresponding span $X \leftarrow R \rightarrow Y$ arises as a pullback.

We have the following properties. For the specific case of coalgebras for CCS, items 1 and 2 of Prop. 5.2.2 amount to Prop. 15(2) of Milner [1989]: they correspond post-fixed points of Φ with bisimulations.

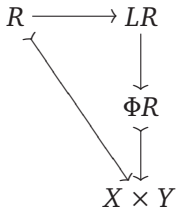
Proposition 5.2.2. *Consider a relation R in $\mathbf{Rel}(X,Y)$.*

1. *If R is a B-bisimulation between (X,h) and (Y,k) then $R \subsetneq \Phi R$.*
2. *If $R \subsetneq \Phi R$ and R is regular, and if B preserves weak pullbacks, then R is a B-bisimulation between (X,h) and (Y,k).*
3. *If B preserves weak pullbacks and R is regular, then so is ΦR. Indeed, if R is a pullback of a span $(X \rightarrow Z \leftarrow Y)$ then ΦR is a pullback of $(X \xrightarrow{h} BX \rightarrow BZ \leftarrow BY \xleftarrow{k} Y)$.*

Proof. Item (1) follows from straightforwardly from the definitions. For if R is a B-bisimulation between (X,h) and (Y,k) then there is a morphism $r : R \rightarrow BR$ making the following diagram commute.

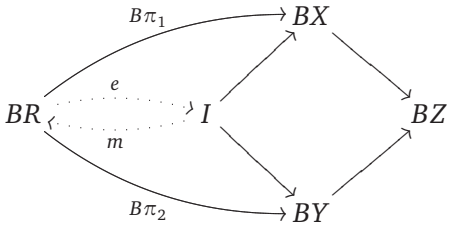


By the universality of LR, we have a morphism $R \rightarrow LR$ making the following diagram commute in particular.

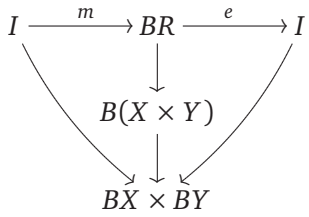


Before we prove items 2 and 3 we provide some general background discussion. Suppose that R is a pullback of a cospan $(X \rightarrow Z \leftarrow Y)$, and that B preserves weak pullbacks. We let I be a strong pullback of the cospan $(BX \rightarrow BZ \leftarrow BY)$. We will prove that ΦR is a pullback of the cospan $(X \times Y \xrightarrow{h \times k} BX \times BY \longleftarrow I)$.

We have the following situation.



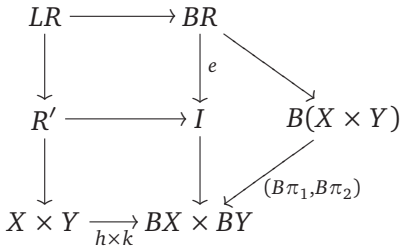
The universal property of I and the weak universal property of BR give us maps $m : I \to BR$ and $e : BR \to I$ such that m is a section of e, and such that the following diagram commutes.



We (temporarily) write R' for a pullback of the cospan

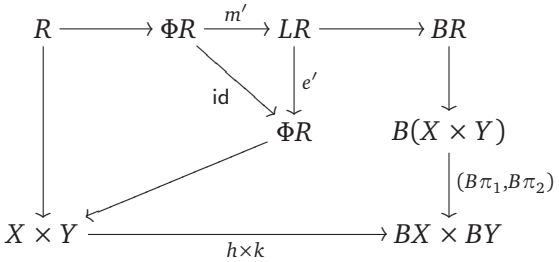
$$(X \times Y \xrightarrow{h \times k} BX \times BY \longleftarrow I) \quad .$$

We know that LR is a pullback of the cospan $(R' \to I \xleftarrow{e} BR)$ since we have the following situation, and pullback squares compose.



The morphism $e : BR \to I$ is split epi, and the morphism $I \to X \times Y$ is monic, since it is an equaliser. Both split epis and monos are stable under pullback, and so the decomposition $LR \to R' \to X \times Y$ is a split-epi/mono factorisation. Such factorisations are images, and so R' is ΦR (up-to isomorphism). We will write $e' : LR \to \Phi R$ for the retraction, and $m' : \Phi R \to LR$ for its section.

We are now ready to prove item (2). Suppose that we have $R \subsetneq \Phi R$. Then there is a morphism $R \to BR$ given by the composite $(R \to \Phi R \xrightarrow{m'} LR \to BR)$. To see that this structure lifts R to a span of coalgebras, consider the following subdivision.



For item (3), notice that ΦR is a pullback of the cospan $(X \xrightarrow{h} BX \to BZ \leftarrow BY \xleftarrow{k} Y)$. Indeed, cones over this cospan correspond bijectively with cones over the cospan $(X \times Y \xrightarrow{h \times k} BX \times BY \longleftarrow I)$. □

Using Tarski's fixed point theorem. Recall that if $\mathscr{C}$ is complete then so is the preorder $\mathbf{Rel}(X, Y)$.

Proposition 5.2.3. *If $\mathscr{C}$ is complete then the property of regularity is meet-closed in* $\mathbf{Rel}(X, Y)$.

Proof notes. For any set I, the meet of an I-indexed family of regular relations $\{R_i\}_{i \in I}$ is computed as follows. For each $i \in I$ there is a cospan $(X \to Z_i \leftarrow Y)$ in $\mathscr{C}$ of which R_i is a pullback. A meet $\bigwedge_{i \in I} R_i$ of $\{R_i\}_{i \in I}$ is a pullback of the cospan

$$\left(X \xrightarrow{\Delta} \prod_{i \in I} X \longrightarrow \prod_{i \in I} Z_i \longleftarrow \prod_{i \in I} Y \xleftarrow{\Delta} Y \right) \quad . \tag{5.2.4}$$

(Here, the arrows labelled Δ are the diagonals.)

This relation $\bigwedge_{i\in I} R_i$ factors through R_i, for each $i \in I$, since a cone over the cospan (5.2.4) is also a cone over the cospan $(X \to Z_i \leftarrow Y)$. □

Suppose now that B preserves weak pullbacks, and that $\mathscr{C}$ is complete and well-powered. The latter assumption ensures that the partial order determined by $\mathbf{Rel}(X, Y)$ is a complete lattice. As mentioned above, the operator Φ is certainly monotone, and so, in this setting, Tarski's fixed point theorem says that the operator Φ has a greatest fixed point. By fixed-point induction, with Prop. 5.2.2(3) and Prop. 5.2.3, we conclude that the greatest fixed point is regular. Hence, by Prop. 5.2.2(1, 2), the greatest fixed point of Φ is a final B-bisimulation.

5.2.2 Constructing final bisimulations

In this subsection we introduce two sequences (precisely: ordinal indexed cochains). The first sequence that we consider is the *terminal sequence*. This sequence is often used to construct a final coalgebra for an endofunctor. We are not especially concerned with final coalgebras here, but the reader will find a recent overview by Worrell [2005, Sec. 2]. The idea of the terminal sequence is to begin with the terminal object, and then successively find *B-algebra* structures, so that if the sequence converges, *i.e.* the B-algebra structure is an isomorphism, then we have a *B-coalgebra* structure, and indeed the final such.

The second sequence that we consider we call the *relation refinement sequence*. This is a sequence of relations in $\mathbf{Rel}(X, Y)$ that arises when trying to construct a greatest fixed point for the operator Φ introduced in the previous section. Thus we begin with the universal relation on the carriers of the coalgebras, successively refining this relation.

Having introduced these two sequences, we show that they are very closely connected. We then provide a basic condition for the termination of the relation refinement sequence.

In this subsection we strengthen the previous assumptions about the ambient category $\mathscr{C}$; we now assume that $\mathscr{C}$ is complete, as well as having images. We continue to consider an arbitrary endofunctor B on $\mathscr{C}$, and B-coalgebras $h : X \to BX$ and $k : Y \to BY$.

The terminal sequence. The terminal sequence of B is the cochain $(z_{\beta,\alpha} : Z_\beta \to Z_\alpha)_{\alpha\le\beta}$ defined by transfinite induction, as follows.

- The limiting step is as follows. For any limit ordinal α, we suppose that we have already defined the α-cochain $(z_{\gamma,\beta} : Z_\gamma \to Z_\beta)_{\beta\le\gamma<\alpha}$. We define Z_β to be the limit of this cochain; then we let $z_{\beta,\alpha}$ be the arrows of the limiting cone.

  In particular, Z_0 is the terminal object of $\mathscr{C}$.

- The inductive step is as follows. If, for some ordinal α, the object Z_α has been defined, then we let $Z_{\alpha+1} = B(Z_\alpha)$.

  We define the morphism $z_{\alpha+1,\alpha} : Z_{\alpha+1} \to Z_\alpha$ differently depending on whether α is limiting.

  – If α is limiting, we already have an α-cochain $(z_{\gamma,\beta} : Z_\gamma \to Z_\beta)_{\beta\le\gamma<\alpha}$ and we know that the cone $(z_{\alpha,\beta} : Z_\alpha \to Z_\beta)_{\beta<\alpha}$ is a limit for the cochain. There is another cone for this cochain, with apex $Z_{\alpha+1}$ and with edges

  $$\left(Z_{\alpha+1} \xrightarrow{=} B(Z_\alpha) \xrightarrow{B(z_{\alpha,\beta})} B(Z_\beta) \xrightarrow{=} Z_{\beta+1} \xrightarrow{z_{\beta+1,\beta}} Z_\beta \right)_{\beta<\alpha} .$$

  Hence we have a mediating morphism $Z_{\alpha+1} \to Z_\alpha$. In this case then we define $z_{\alpha+1,\alpha}$ to be this morphism.

– Otherwise, if $\alpha = \beta + 1$, let $z_{\alpha+1,\alpha}$ be the composite

$$Z_{\alpha+1} \xrightarrow{=} B(Z_{\beta+1}) \xrightarrow{B(z_{\beta+1,\beta})} BZ_\beta \xrightarrow{=} Z_\alpha \ .$$

In either case, for any $\beta \leq \alpha$, we then define $z_{\alpha+1,\beta} : Z_{\alpha+1} \to Z_\beta$ by letting $z_{\alpha+1,\beta} = z_{\alpha+1,\alpha} \circ z_{\alpha,\beta}$.

(It is straightforward to verify that the ordinal indexed family $(z_{\beta,\alpha} : Z_\beta \to Z_\alpha)_{\alpha\leq\beta}$ is indeed a cochain.)

The relation refinement sequence. We define an ordinal indexed cochain $(R_\beta \lneq R_\alpha)_{\alpha\leq\beta}$ in $\mathbf{Rel}(X, Y)$ by transfinite induction as follows.

- For a limiting ordinal α, let R_α be the limit of the α-cochain $(R_\gamma \lneq R_\beta)_{\beta\leq\gamma<\alpha}$ in $\mathbf{Rel}(X, Y)$.

  In particular, $R_0 = X \times Y$.

- We define $R_{\alpha+1}$ to be ΦR_α.

  If α is a limit ordinal, then, since Φ is monotone, we know that for each ordinal $\beta < \alpha$ we have $R_{\alpha+1} \lneq R_\beta$. Thus we can conclude that $R_{\alpha+1} \lneq R_\alpha$. On the other hand, if $\alpha = \beta + 1$ then we can conclude that $R_{\alpha+1} \lneq R_\alpha$ since Φ is monotone.

Coalgebras determine cones over the terminal sequence. For each ordinal α, we have a morphism $x_\alpha : X \to Z_\alpha$ given as follows.

- If α is a limit ordinal, we have an α-cochain $(z_{\gamma,\beta} : Z_\gamma \to Z_\beta)_{\beta\leq\gamma<\alpha}$; the cone $(z_{\alpha,\beta} : Z_\alpha \to Z_\beta)_{\beta<\alpha}$ is by definition limiting for the cochain.

  The morphisms $x_\alpha : X \to Z_\alpha$ for $\alpha < \beta$ form a cocone over this cochain, with apex X. We let $x_\alpha : X \to Z_\alpha$ be the unique mediating morphism.

  For instance, if $\alpha = 0$, then $x_\alpha : X \to Z_\alpha$ is the terminal map $X \to 1$.

- Otherwise, if $\alpha = \beta + 1$, then we let x_β be the composite

$$X \xrightarrow{h} BX \xrightarrow{Bx_\beta} BZ_\beta \xrightarrow{=} Z_\alpha \quad .$$

In the same way, we determine a cone $(y_\alpha : Y \to Z_\alpha)_\alpha$ over the terminal sequence with apex Y.

Relating the relation sequence with the terminal sequence. We are now in a position to relate the sequence of relations $(R_\alpha)_\alpha$ with the terminal sequence $(Z_\alpha)_\alpha$.

Proposition 5.2.5. *Suppose that B preserves weak pullbacks, and consider an ordinal α. The following square is a pullback.*

$$\begin{array}{ccccc} R_\alpha & \xrightarrow{r_{\alpha,0}} & X \times Y & \xrightarrow{\pi_2} & Y \\ {\scriptstyle r_{\alpha,0}}\downarrow & & & & \\ X \times Y & & & & \downarrow{\scriptstyle y_\alpha} \\ {\scriptstyle \pi_1}\downarrow & & & & \\ X & & \xrightarrow[x_\alpha]{} & & Z_\alpha \end{array}$$

Proof notes. This statement is proved by transfinite induction on α. The case when α is a limiting ordinal holds because limits commute with limits. The inductive step follows immediately from Prop. 5.2.2(3). □

Bounds for convergence. We now provide conditions under which the relation refinement sequence will converge. In the category of sets, it is clear that for any finite set, the subset relation on its subsets is well-founded. We now establish this property in a slightly more general setting. (Here we say that a diagram of monos is *bounded* if there is a cocone of monos over it.)

Theorem 5.2.6. *Let λ be a limit ordinal for which* $\mathrm{card}(\lambda)$ *is regular. Let $\mathscr{C}$ be a Boolean topos with colimits of bounded λ-chains of monos. Consider a λ-cochain $(m_{\beta,\alpha} : S_\beta \rightarrowtail S_\alpha)_{\alpha\leq\beta<\lambda}$ of monomorphisms such that S_0 is* $(\mathrm{card}(\lambda))$*-presentable. Then there is an ordinal $\alpha < \lambda$ for which $m_{\alpha+1,\alpha} : S_{\alpha+1} \to S_\alpha$ is an isomorphism.*

Proof. Consider a pair of ordinals α, β such that $\alpha \leq \beta < \lambda$. Since the category $\mathscr{C}$ is Boolean, the morphism $m_{\beta,\alpha} : S_\beta \rightarrowtail S_\alpha$ is complemented, and so we have an object $(S_\alpha - S_\beta)$ such that the object S_α is a coproduct of S_β and $(S_\alpha - S_\beta)$, and the monomorphism $m_{\beta,\alpha}$ is the coproduct injection.

We now define a λ-chain

$$\Big(m'_{\alpha,\beta} : (S_0 - S_\alpha) \rightarrowtail (S_0 - S_\beta)\Big)_{\alpha\leq\beta<\lambda} \tag{5.2.7}$$

with morphisms $m'_{\alpha,\beta}$ given as follows. First, observe that whenever $\alpha \leq \beta$ then we have

$$\begin{aligned}(S_0 - S_\beta) + S_\beta &\cong S_0\\ &\cong (S_0 - S_\alpha) + S_\alpha\\ &\cong (S_0 - S_\alpha) + (S_\alpha - S_\beta) + S_\beta \quad .\end{aligned}$$

Since coproducts in the topos $\mathscr{C}$ are universal we can cancel the S_β from the resulting equation to conclude that

$$(S_0 - S_\beta) \cong (S_0 - S_\alpha) + (S_\alpha - S_\beta) \quad .$$

We define $m'_{\alpha,\beta} : (S_0 - S_\alpha) \rightarrowtail (S_0 - S_\beta)$ to be the evident coproduct injection. It is monic since coproducts in the topos $\mathscr{C}$ are disjoint.

There is a cone of monomorphisms over (5.2.7) with apex S_0. Being bounded, the chain must have a colimit, which we denote $(S_0 - S_\lambda)$; we denote the colimiting cone by

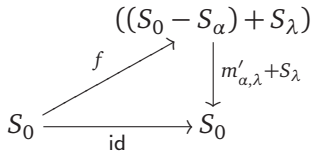
$$\Big(m'_{\alpha,\lambda} : (S_0 - S_\alpha) \rightarrowtail (S_0 - S_\lambda)\Big)_{\alpha<\lambda} \quad . \tag{5.2.8}$$

We have a mediating morphism $(S_0 - S_\lambda) \to S_0$ from this colimit to the cone of monos, and it follows that every morphism of the colimiting cone (5.2.8) is monic. Moreover, the mediating morphism $(S_0 - S_\lambda) \to S_0$ is monic because pullbacks commute with colimits in toposes — so $(S_0 - S_\lambda)$ is a union of (5.2.7). Since $\mathscr{C}$ is Boolean there is an object S_λ such that S_0 is a coproduct $(S_0 - S_\lambda) + S_\lambda$.

Now, consider the λ-chain

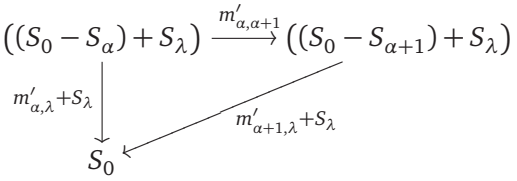
$$\Big((m'_{\alpha,\beta} + S_\lambda) : (S_0 - S_\alpha) + S_\lambda \to (S_0 - S_\beta) + S_\lambda\Big)_{\alpha\leq\beta<\lambda}$$

of which $(S_0 - S_\lambda) + S_\lambda \cong S_0$ must be a colimit. By assumption, S_0 is $(\mathrm{card}(\lambda))$-presentable, and the λ-chain $\Big(m'_{\alpha,\beta} + S_\lambda\Big)_{\alpha\leq\beta<\lambda}$ is certainly $(\mathrm{card}(\lambda))$-filtered; hence there is an ordinal α and a morphism $f : S_0 \to ((S_0 - S_\alpha) + S_\lambda)$ such that the following diagram commutes.



Thus the morphism $(m'_{\alpha,\lambda} + S_\lambda) : (S_0 - S_\alpha) + S_\lambda \to S_0$ is split epic. It is also monic, since sums are universal, and so we can conclude that it is an isomorphism.

By considering the following fragment of the colimiting cone, one can conclude that the morphism $m'_{\alpha,\alpha+1} : (S_0 - S_\alpha) + S_\lambda \to (S_0 - S_{\alpha+1}) + S_\lambda$ is an isomorphism.



Thus we have

$$\begin{aligned}(S_0 - S_{\alpha+1}) &\cong (S_0 - S_\alpha) \\ &\cong (S_0 - S_{\alpha+1}) + (S_\alpha - S_{\alpha+1}) \quad .\end{aligned}$$

Since coproducts are universal, we can cancel the common factor $(S_0 - S_{\alpha+1})$, leaving $(S_\alpha - S_{\alpha+1}) = 0$. So we know that the morphism $m_{\alpha+1,\alpha} : S_{\alpha+1} \rightarrowtail S_\alpha$ (part of the original λ-cochain) is an isomorphism, and the theorem is proved. □

According to this result, the relation refinement process will terminate when $\mathscr{C}$ is a Boolean Grothendieck topos and the carrier object $X \times Y$ is finitely presentable.

The Schanuel topos is Boolean, and so, by combining Theorem 5.2.6 with Corollary 5.1.12, we can conclude that the relation refinement algorithm for the endofunctor B_{e} on $\mathbf{Sh}(\mathbf{I})$ for early bisimulation will always terminate when run on a sheaf that is presented by a named-set with finite index.

Part II

Structural Operational Semantics

Chapter 6

Rule Induction and Mathematical Operational Semantics

We begin this chapter with an overview of some important elements of an abstract theory of syntax. The development is largely standard, although the characterisation of free monads in Theorem 6.1.5 seems to be novel.

In Section 6.2, we survey some aspects of the Mathematical Operational Semantics begun by Turi and Plotkin [1997]. The idea behind this field of research is to put some important results of operational semantics in a mathematical framework. In doing so we can abstract away from the low level details that so often dominate results in operational semantics. Here, we explain how the usual way of defining transition systems by rule induction can be understood as using initial algebra induction to lift a monad to a category of coalgebras. We explain how the main results are valid from the point of view of structured coalgebras.

Finally, in Section 6.3, we illustrate the theory of Section 6.2. We recall a positive version of the GSOS rule format proposed by Bloom, Istrail, and Meyer [1995], and explain the process of monad lifting for rules in this format. This process has been discussed before; our main reason for discussing it here is to introduce the ideas and techniques involved, since in Section 8.4 we will follow a similar process in a more complex setting.

6.1 Rudiments of abstract syntax

We overview some important elements of the theory of abstract syntax. We begin by providing a notion of algebraic signature, and defining a notion of model for such a signature. Such a model can often be seen as an algebra for an endofunctor; this leads us to study categories of algebras for endofunctors, and for monads, in some detail. To do this we introduce, in Section 6.1.2, categories of endofunctors and of monads. We conclude this section with a discussion about free monads on endofunctors, in Section 6.1.3.

6.1.1 Algebraic signatures and their models

We recall a notion of algebraic signature, and explain how a category of models for the signature is often the same thing as a category of algebras for an endofunctor.

Algebraic signatures.

Definition 6.1.1. An *algebraic signature* $\mathbb{S}$ is a set $\mathsf{Op}_{\mathbb{S}}$ of operators op each equipped with an arity $\mathrm{ar}(\mathsf{op}) \in \mathbb{N}$.

As a first example, we consider a signature $\mathbb{CCS}$ for a finite fragment of the pure CCS of Milner [1989], which has operators

$$\mathsf{Op}_{\mathbb{CCS}} = \{\mathsf{par}, \mathsf{sum}, \mathsf{nil}, \mathsf{tau}\} \cup \{a.(-) \mid a \in \mathrm{Act}\} \cup \{\bar{a}.(-) \mid a \in \mathrm{Act}\} \tag{6.1.2}$$

while arities are given according to the following table.

$\mathsf{op} \in \mathsf{Op}_{\mathbb{CCS}}$	$\mathrm{ar}(\mathsf{op})$
par	2
sum	2
nil	0
tau	1
$a.(-)$	1
$\bar{a}.(-)$	1

(Here, Act is a chosen set of CCS actions.)

Models of algebraic signatures. Let $\mathbb{S}$ be a signature. In any category $\mathscr{C}$ with finite products, an $\mathbb{S}$-*structure* is an object $X \in \mathscr{C}$ together with, for each $\mathsf{op} \in \mathsf{Op}_{\mathbb{S}}$, a morphism $[\![\mathsf{op}]\!]_X : X^{\mathrm{ar}(\mathsf{op})} \to X$. (Here, $X^{\mathrm{ar}(\mathsf{op})}$ denotes the $\mathrm{ar}(\mathsf{op})$-fold product of X.)

For our example of CCS: a $\mathbb{CCS}$-structure X in $\mathscr{C} = \mathbf{Set}$ is given by a set X, to be thought of as a set of processes, together with functions denoting the operations of CCS, including, for instance,

- a function $[\![\mathsf{par}]\!]_X : X \times X \to X$ representing parallel composition — to be thought of as mapping a pair of processes to the process that behaves as the two in parallel;
- an element $[\![\mathsf{nil}]\!]_X$ in X representing the deadlocked process.

Morphisms between models. A morphism $X \to Y$ between two $\mathbb{S}$-structures in $\mathscr{C}$ is given by a morphism $f : X \to Y$ in $\mathscr{C}$ such that, for each $\mathsf{op} \in \mathsf{Op}_{\mathbb{S}}$, the following diagram commutes.

$$\begin{array}{ccc} X^{\mathrm{ar}(\mathsf{op})} & \xrightarrow{[\![\mathsf{op}]\!]_X} & X \\ {\scriptstyle f^{\mathrm{ar}(\mathsf{op})}}\downarrow & & \downarrow{\scriptstyle f} \\ Y^{\mathrm{ar}(\mathsf{op})} & \xrightarrow{[\![\mathsf{op}]\!]_Y} & Y \end{array}$$

These objects and morphisms form a category; identities and composition are as in $\mathscr{C}$.

Models as algebras for an endofunctor. If $\mathscr{C}$ also has $\mathsf{Op}_{\mathbb{S}}$-indexed coproducts then we can associate the signature $\mathbb{S}$ with an endofunctor $\Sigma_{\mathbb{S},\mathscr{C}}$ on $\mathscr{C}$:

$$\Sigma_{\mathbb{S},\mathscr{C}} = \coprod_{\mathsf{op} \in \mathsf{Op}_{\mathbb{S}}} \left(\mathsf{id}_{\mathscr{C}}{}^{\mathrm{ar}(\mathsf{op})}\right). \tag{6.1.3}$$

In this situation, to give an $\mathbb{S}$-structure in $\mathscr{C}$ is to give an object X of $\mathscr{C}$ together with a morphism $\coprod_{\mathsf{op} \in \mathsf{Op}_{\mathbb{S}}} \left(X^{\mathrm{ar}(\mathsf{op})}\right) \to X$; in other words, an $\mathbb{S}$-structure in $\mathscr{C}$ is the same thing as an algebra for the endofunctor $\Sigma_{\mathbb{S},\mathscr{C}}$. Indeed, the category of $\mathbb{S}$-structures in $\mathscr{C}$ is isomorphic to the category of algebras for the endofunctor $\Sigma_{\mathbb{S},\mathscr{C}}$.

6.1.2 Categories of endofunctors and of monads

The concept of algebras for an endofunctor is a basic one. We now turn to consider endofunctors and their algebras in a general setting. The endofunctors arising as in equation 6.1.3 provide important examples.

Morphisms of endofunctors. We introduce a 2-category Endo of endofunctors. This is somehow dual to the 2-category coEndo considered in Section 2.3.

- Objects of Endo are pairs $(\mathscr{C}, \Sigma)$ of a category $\mathscr{C}$ and an endofunctor Σ on $\mathscr{C}$.
- A morphism from $(\mathscr{C}, \Sigma)$ to $(\mathscr{C}', \Sigma')$ is given by a functor $F : \mathscr{C} \to \mathscr{C}'$ together with a natural transformation $\phi : \Sigma' F \to F\Sigma$; composition of two morphisms

  $$(\mathscr{C}, \Sigma) \xrightarrow{(F,\phi)} (\mathscr{C}', \Sigma') \xrightarrow{(G,\gamma)} (\mathscr{C}'', \Sigma'')$$

  is given by

  $$(GF,\ \Sigma'' GF \xrightarrow{\gamma F} G\Sigma' F \xrightarrow{G\phi} GF\Sigma) \quad .$$

- Let (F, ϕ) and (G, γ) be two morphisms from $(\mathscr{C}, \Sigma)$ to $(\mathscr{C}', \Sigma')$. A 2-cell from (F, ϕ) to (G, γ) is given by a natural transformation $\alpha : F \to G$ such that the following diagram commutes.

  $$\begin{array}{ccc} \Sigma' F & \xrightarrow{\phi} & F\Sigma \\ {\scriptstyle \Sigma'\alpha}\downarrow & & \downarrow{\scriptstyle \alpha\Sigma} \\ \Sigma' G & \xrightarrow[\gamma]{} & G\Sigma \end{array}$$

  Composition and identities of 2-cells are as in the category **CAT** of categories.

The construction of categories of algebras for an endofunctor extends to a 2-functor Endo $\to$ **CAT**, exactly as was the case for coalgebras in Section 2.3. This functor is isomorphic to the hom-functor $\mathsf{Endo}((1, \mathsf{id}_1), -)$, where we write $(1, \mathsf{id}_1)$ for the identity endofunctor on the terminal category.

For any endofunctor Σ on a category $\mathscr{C}$, we let Σ-Alg be the category of Σ-algebras.

Morphisms of monads. We will also need to work with the category Monad of monads; we reproduce here the definition of Street [1972], for the case of monads in **CAT**:

- Objects of Monad are pairs $(\mathscr{C}, \mathbf{T})$ of a category $\mathscr{C}$ and a monad $\mathbf{T}$ on $\mathscr{C}$. (As a convention, we will use a bold symbol for a monad, and an italic symbol for the underlying endofunctor.)
- A morphism from $(\mathscr{C}, \mathbf{T} = (T, \eta, \mu))$ to $(\mathscr{C}', \mathbf{T}' = (T', \eta', \mu'))$ is given by a functor $F : \mathscr{C} \to \mathscr{C}'$ together with a natural transformation $\phi : T'F \to FT$ for which the following two diagrams commute.

  $$\begin{array}{ccc} F & \xrightarrow{\eta' F} & T'F \\ & {\scriptstyle F\eta}\searrow & \downarrow{\scriptstyle \phi} \\ & & FT \end{array} \qquad \begin{array}{ccccc} T'T'F & \xrightarrow{T'\phi} & T'FT & \xrightarrow{\phi T} & FTT \\ {\scriptstyle \mu' F}\downarrow & & & & \downarrow{\scriptstyle F\mu} \\ T'F & & \xrightarrow{\qquad\phi\qquad} & & FT \end{array}$$

  Composition of two morphisms $(\mathscr{C}, \mathbf{T}) \xrightarrow{(F,\phi)} (\mathscr{C}', \mathbf{T}') \xrightarrow{(G,\gamma)} (\mathscr{C}'', \mathbf{T}'')$ is given by

  $$(GF,\ T''GF \xrightarrow{\gamma F} GT'F \xrightarrow{G\phi} GFT) \quad .$$

- Let (F, ϕ) and (G, γ) be two morphisms from $(\mathscr{C}, \mathbf{T})$ to $(\mathscr{C}', \mathbf{T}')$. A 2-cell from (F, ϕ) to (G, γ) is a natural transformation $\alpha : F \to G$ making the following diagram commute.

  $$\begin{array}{ccc} T'F & \xrightarrow{\phi} & FT \\ {\scriptstyle T'\alpha}\downarrow & & \downarrow{\scriptstyle \gamma T} \\ T'G & \xrightarrow{\gamma} & GT \end{array}$$

  Composition and identities of 2-cells are as in **CAT**.

Notice that we have a faithful forgetful 2-functor Monad → Endo, which is in fact locally full and faithful.

For any monad $\mathbf{T}$ on a category $\mathscr{C}$, we always understand a $\mathbf{T}$-*algebra* to be an algebra for the monad $\mathbf{T}$ in the sense of Eilenberg and Moore [see *e.g.* Mac Lane, 1998, Sec. VI.2], as opposed to the more liberal notion of an algebra for the endofunctor underlying $\mathbf{T}$.

The construction of algebras for a monad extends to a 2-functor Monad → **CAT**. Once again, this functor is isomorphic to the hom-functor $\mathsf{Monad}((1,\mathsf{id}_1),-)$, where $(1,\mathsf{id}_1)$ is the terminal monad, given by the identity endofunctor on the terminal category. In particular, to give a $\mathbf{T}$-algebra is to give a morphism $(1,\mathsf{id}_1)\to(\mathscr{C},\mathbf{T})$ in Monad.

For any monad $\mathbf{T}$ on a category $\mathscr{C}$, we let $\mathbf{T}$-Alg be the category of $\mathbf{T}$-algebras.

Liftings of endofunctors and monads. We say that a morphism $(F,\phi):(\mathscr{C},\Sigma)\to(\mathscr{C}',\Sigma')$ in Endo is a *lifting* if ϕ is an isomorphism. In the same way, we say that a morphism $(F,\phi):(\mathscr{C},\mathbf{T})\to(\mathscr{C}',\mathbf{T}')$ in Monad is a *lifting* if ϕ is an isomorphism.

Occasionally, we will refer to *strict liftings*, *i.e.* liftings for which the isomorphism is the identity.

We recall two results, which both follow immediately from a basic result of two-dimensional monad theory [see *e.g.* Kelly, 1972, Thm. 1.4], since both Endo and Monad are categories of algebras and lax morphisms for particular 2-monads on **CAT** [see *e.g.* Blackwell *et al.*, 1989, Sec. 6.1].

Proposition 6.1.4.

1. *A morphism of endofunctors $(F,\phi):(\mathscr{C}',\Sigma)\to(\mathscr{C},\Sigma')$ has a right adjoint in* Endo *if and only if the underlying functor F has a right adjoint in* **CAT** *and ϕ is an isomorphism.*

2. *A morphism of monads $(F,\phi):(\mathscr{C}',\mathbf{T})\to(\mathscr{C},\mathbf{T}')$ has a right adjoint in* Monad *if and only if the underlying functor F has a right adjoint in* **CAT** *and ϕ is an isomorphism.* □

Morphisms of model categories and morphisms of signature endofunctors. Fix a signature $\mathbb{S}$. We let $\mathscr{M}_{\mathbb{S}}$ be the 2-category of model categories: objects of $\mathscr{M}_{\mathbb{S}}$ are categories with finite products and $\mathsf{Op}_{\mathbb{S}}$-indexed coproducts; morphisms are finite product preserving functors; 2-cells are natural transformations. The construction of an endofunctor provided in (6.1.3) extends to a 2-functor $\mathscr{M}_{\mathbb{S}}\to\mathsf{Endo}$. In particular, any morphism $F:\mathscr{C}\to\mathscr{D}$ in $\mathscr{M}_{\mathbb{S}}$ induces a natural transformation $\Sigma_{\mathbb{S},\mathscr{D}}F\to F\Sigma_{\mathbb{S},\mathscr{C}}$ as a mediating morphism into the cone

$$\left\{(F(\mathsf{id}_{\mathscr{C}}))^{\mathsf{ar}(\mathsf{op})}\xrightarrow{\sim}F(\mathsf{id}_{\mathscr{C}}{}^{\mathsf{ar}(\mathsf{op})})\xrightarrow{F(\mathsf{inj}_{\mathsf{op}})}F\left(\coprod_{\mathsf{op}\in\mathsf{Op}_{\mathbb{S}}}\left(\mathsf{id}_{\mathscr{C}}{}^{\mathsf{ar}(\mathsf{op})}\right)\right)\xrightarrow{=}F\Sigma_{\mathbb{S},\mathscr{C}}\right\}_{\mathsf{op}\in\mathsf{Op}_{\mathbb{S}}}.$$

Hence $F:\mathscr{C}\to\mathscr{D}$ extends to a morphism $(\mathscr{C},\Sigma_{\mathbb{S},\mathscr{C}})\to(\mathscr{D},\Sigma_{\mathbb{S},\mathscr{D}})$ in Endo.

If a morphism $F:\mathscr{C}\to\mathscr{D}$ of model categories also preserves $\mathsf{Op}_{\mathbb{S}}$-indexed coproducts, then the induced natural transformation $\Sigma_{\mathbb{S},\mathscr{D}}F\to F\Sigma_{\mathbb{S},\mathscr{C}}$ is an isomorphism, and so we have a lifting of endofunctors.

6.1.3 Free monads on endofunctors

Theorem 6.1.5. *Let Σ be an endofunctor on a category $\mathscr{C}$, and let $\mathbf{T}_\Sigma$ be a monad on $\mathscr{C}$. The following are equivalent.*

1. *There is a natural family of isomorphisms of categories*

$$\left\{\mathsf{Endo}((\mathscr{D},T),(\mathscr{C},\Sigma))\xrightarrow{\sim}\mathsf{Monad}((\mathscr{D},\mathbf{T}),(\mathscr{C},\mathbf{T}_\Sigma))\right\}_{(\mathscr{D},\mathbf{T})\in\mathsf{Monad}}$$

for which the following diagram commutes, where the vertical arrows are the evident forgetful functors.

$$\begin{array}{ccc} \mathsf{Endo}((\mathscr{D},T),(\mathscr{C},\Sigma)) & \xrightarrow{\ \sim\ } & \mathsf{Monad}((\mathscr{D},\mathbf{T}),(\mathscr{C},\mathbf{T}_\Sigma)) \\ & \searrow \quad \swarrow & \\ & \mathbf{CAT}(\mathscr{D},\mathscr{C}) & \end{array}$$

2. *The forgetful functor* Σ-Alg $\to \mathscr{C}$ *has a left adjoint and the resulting monad on* $\mathscr{C}$ *is* $\mathbf{T}_\Sigma$.

Although this result is somewhat fundamental in understanding free monads, I could not find it stated in the literature and so a proof is provided in Appendix 6.A.

Whenever we have the situation in Theorem 6.1.5 then we say that the monad $\mathbf{T}_\Sigma$ is *the free monad on* Σ. We will write $\eta_\Sigma : 1 \to T_\Sigma$ for the unit of $\mathbf{T}_\Sigma$, and $\mu_\Sigma : T_\Sigma T_\Sigma \to T_\Sigma$ for the multiplication of $\mathbf{T}_\Sigma$. Using item (1), we can transpose the identity morphism on $(\mathscr{C},\mathbf{T}_\Sigma)$ in Monad to obtain a natural transformation $\sigma_\Sigma : \Sigma \to T_\Sigma$, exhibiting $\mathbf{T}_\Sigma$ as free on Σ in the slightly more general sense studied by (*e.g.*) Barr [1970], Kelly [1980, Sec. 22], and Barr and Wells [1984, Sec. 9.4].

It is well-known that for any endofunctor Σ on any category $\mathscr{C}$, if the forgetful functor Σ-Alg $\to \mathscr{C}$ has a left adjoint then it is monadic [see *e.g.* Barr and Wells, 1984, Prop. 9.4.1]. Indeed, this result can be derived from Theorem 6.1.5, by taking $(\mathscr{D},\mathbf{T})$ to be the terminal monad $(1,\mathsf{id}_1)$.

Suppose that we have the situation in item (2) of Theorem 6.1.5. Then the left adjoint to the forgetful functor Σ-Alg $\to \mathscr{C}$ sends an object X of $\mathscr{C}$ to an algebra with carrier $T_\Sigma X$; we will denote the structure map by $t_{\Sigma X} : \Sigma T_\Sigma X \to T_\Sigma X$. The adjunction can be understood as providing the following recursion principle: for any Σ-algebra (Y,y), and any morphism $f : X \to Y$, we have a unique morphism $(f,y)^\sharp : T_\Sigma X \to Y$ such that the following diagram commutes in $\mathscr{C}$.

$$\begin{array}{ccc} \Sigma T_\Sigma X & \xrightarrow{\Sigma(f,y)^\sharp} & \Sigma Y \\ {\scriptstyle t_{\Sigma X}}\downarrow & & \downarrow{\scriptstyle y} \\ T_\Sigma X & \xrightarrow{(f,y)^\sharp} & Y \\ {\scriptstyle \eta_\Sigma X}\uparrow & \nearrow_{f} & \\ X & & \end{array}$$

The family of structure maps for the free Σ-algebras $\{t_{\Sigma X} : \Sigma T_\Sigma X \to T_\Sigma X\}_{X\in\mathscr{C}}$ is natural; the natural transformation $t_\Sigma : \Sigma T_\Sigma \to T_\Sigma$ is such that $t_\Sigma = \mu_\Sigma \circ \sigma_\Sigma T_\Sigma$.

Existence of free monads. We record here a sufficient condition for the existence of free monads, that encompases all the cases that arise in this thesis. For a proof of this result, see, for instance, Barr and Wells [1984, Prop. 9.4.7].

Proposition 6.1.6. *If* $\mathscr{C}$ *is complete and has pushouts and colimits of* ω*-chains, and an endofunctor* Σ *on* $\mathscr{C}$ *preserves colimits of* ω*-chains, then the free monad on* $(\mathscr{C},\Sigma)$ *exists.* □

Functoriality of the free monad construction. The construction $\mathbf{T}_{(-)}$ of free monads is 2-functorial in the following sense. When the free monads on $(\mathscr{C},\Sigma)$ and $(\mathscr{C}',\Sigma')$ exist then we have a functor between hom-categories

$$\mathsf{Endo}((\mathscr{C},\Sigma),(\mathscr{C}',\Sigma')) \longrightarrow \mathsf{Monad}((\mathscr{C},\mathbf{T}_\Sigma),(\mathscr{C}',\mathbf{T}_{\Sigma'}))$$

sending a morphism $(F,\phi) : (\mathscr{C},\Sigma) \to (\mathscr{C}',\Sigma')$ in Endo to the monad morphism found by transposing (using Theorem 6.1.5(1)) the following composite morphism of endofunctors.

$$(\mathscr{C},T_\Sigma) \xrightarrow{(\mathsf{id}_\mathscr{C},\sigma_\Sigma)} (\mathscr{C},\Sigma) \xrightarrow{(F,\phi)} (\mathscr{C}',\Sigma')$$

Term monads for signatures. When $\Sigma = \Sigma_{\mathbb{S},\mathscr{C}}$ is the endofunctor on a category $\mathscr{C}$ according to equation 6.1.3 for a signature $\mathbb{S}$, then we let $\mathbf{T}_{\mathbb{S},\mathscr{C}} = (T_{\mathbb{S},\mathscr{C}}, t_{\mathbb{S},\mathscr{C}}, \eta_{\mathbb{S},\mathscr{C}}, \mu_{\mathbb{S},\mathscr{C}})$ be the corresponding free monad structure, whenever it exists.

For the signature $\mathbb{CCS}$ given in (6.1.2), then for any set X, we have a set $T_{\mathbb{S},\mathbf{Set}}(X)$ of CCS terms with free variables in X. Since the empty set is initial, and left adjoints preserve colimits, we know that the $\Sigma_{\mathbb{CCS},\mathbf{Set}}$-algebra $(T_{\mathbb{CCS},\mathbf{Set}}(0), t_{\mathbb{CCS},\mathbf{Set}_0})$ is initial in the category of $\Sigma_{\mathbb{CCS},\mathbf{Set}}$-algebras.

6.2 Mathematical operational semantics

We now present some aspects of the *mathematical operational semantics* introduced by Turi and Plotkin [1997].

We begin, in Section 6.2.1, by explaining how a lifting of a monad of syntax to a category of coalgebras corresponds to an operational semantics for which bisimulation is a congruence. In Section 6.2.2, we describe a priniciple of parameterised recursion. We use this principle in Section 6.2.3 to create a monad lifting from a mathematical structure that we call an abstract rule.

All the above work takes place in the context of structured coalgebras. In Section 6.2.4 we explain that if the structure map has a right adjoint then the same process can be carried out in the context of (non-structured) coalgebras.

6.2.1 Lifted monads and bisimulation congruences

Congruence. Let Σ be an endofunctor on $\mathscr{C}$ and let (X, x), (Y, y) be two Σ-algebras. A Σ-*congruence* between (X, x) and (Y, y) is a span $(X \xleftarrow{r_1} R \xrightarrow{r_2} Y)$ such that there exists a Σ-algebra structure $r : \Sigma R \to R$ lifting the span (R, r_1, r_2) to a span $\Big((X, x) \xleftarrow{r_1} (R, r) \xrightarrow{r_2} (Y, y)\Big)$ of Σ-algebra homomorphisms. In the terminology of Definition 2.1.3: a Σ-congruence is a *V-lifting span*, writing V for the forgetful functor $\Sigma\text{-Alg} \to \mathscr{C}$.

In the same way, we can define congruences between algebras for monads. Let $\mathbf{T}$ be a monad on $\mathscr{C}$; we say that a $\mathbf{T}$-*congruence* between $\mathbf{T}$-algebras (X, x) and (Y, y) is a span $(X \xleftarrow{r_1} R \xrightarrow{r_2} Y)$ for which there exists a $\mathbf{T}$-algebra structure $r : TR \to R$ lifting the span to a span of $\mathbf{T}$-algebra homomorphisms. So a $\mathbf{T}$-congruence is also a *V-lifting span*, this time writing V for the forgetful functor $\mathbf{T}\text{-Alg} \to \mathscr{C}$. Note that when $\mathbf{T}$ is the free monad $\mathbf{T}_\Sigma$ on an endofunctor Σ, we know that $\mathbf{T}_\Sigma\text{-Alg} \cong \Sigma\text{-Alg}$ and so that a $\mathbf{T}_\Sigma$-congruence between $\mathbf{T}_\Sigma$-algebras is the same thing as a Σ-congruence between the corresponding Σ-algebras.

Following the convention of Section 2.5.1, we will use the term *congruence relation* to describe a congruence whose span is jointly monic.

For a basic example we return to the example signature $\mathbb{CCS}$ introduced in (6.1.2). A $\Sigma_{\mathbb{CCS},\mathbf{Set}}$-congruence relation on the initial $\Sigma_{\mathbb{CCS},\mathbf{Set}}$-algebra is a binary relation R on the set $T_{\mathbb{CCS},\mathbf{Set}}(0)$ of closed CCS terms which is a congruence in the usual sense. For instance, suppose that we have terms $t_1, t_2, t_1', t_2' \in T_{\mathbb{CCS},\mathbf{Set}}(0)$ such that $(t_1, t_1') \in R$ and $(t_2, t_2') \in R$; then we must have $\big(\mathsf{par}(t_1, t_2), \mathsf{par}(t_1', t_2')\big) \in R$.

Monad liftings. Recall, from Section 2.1, that to give a labelled transition system is to give a coalgebra for the endofunctor $B_{\mathrm{lts}} = \mathscr{P}(\mathrm{Lab} \times -)$ on **Set**. For the case of pure CCS, the labels are actions, coactions or silent, and it is sensible to take $\mathrm{Lab} = \mathrm{Act} + \mathrm{Act} + 1$. A strict lifting of the term monad $\mathbf{T}_{\mathbb{CCS},\mathbf{Set}}$ on **Set** along the forgetful functor $B_{\mathrm{lts}}\text{-Coalg} \to \mathbf{Set}$ describes, in particular, a way of transforming a B_{lts}-coalgebra with carrier X into a B_{lts}-coalgebra with carrier $T_{\mathbb{CCS},\mathbf{Set}}(X)$. In other words: given a behaviour for variables, a monad lifting describes the behaviour of compound terms. Thus a monad lifting can be thought of as defining a compositional operational semantics.

In fact, as we now explain, such an operational semantics will necessarily be well-behaved.

The following central result has nothing in particular to do with coalgebras, and so we state and prove the general form using lifting spans as introduced in Definition 2.1.3. We explain the relevance of this result in Corollary 6.2.3.

Theorem 6.2.1. *Consider a functor $V : \mathscr{B} \to \mathscr{C}$ between categories. Consider a monad $\tilde{\mathbf{T}}$ on $\mathscr{B}$ which is a lifting along V of a monad $\mathbf{T}$ on $\mathscr{C}$.*

Let (X, x), (Y, y) be two $\tilde{\mathbf{T}}$-algebras. A final V-lifting span between X and Y is a $\mathbf{T}$-congruence.

Proof. As usual we write (T, η, μ) for the monad $\mathbf{T}$, and $(\tilde{T}, \tilde{\eta}, \tilde{\mu})$ for the monad $\tilde{\mathbf{T}}$. We let $\nu : TV \to V\tilde{T}$ be the lifting isomorphism.

Consider a final V-lifting span between X and Y.

$$VX \xleftarrow{r_1} R \xrightarrow{r_2} VY$$

This means that we have a span in $\mathscr{B}$

$$X \xleftarrow{\tilde{r}_1} \tilde{R} \xrightarrow{\tilde{r}_2} Y$$

such that $(R, r_1, r_2) = (V\tilde{R}, V\tilde{r}_1, V\tilde{r}_2)$.

Applying the monad $\tilde{\mathbf{T}}$ and composing with the algebra maps x, y gives the following span between X and Y in $\mathscr{B}$.

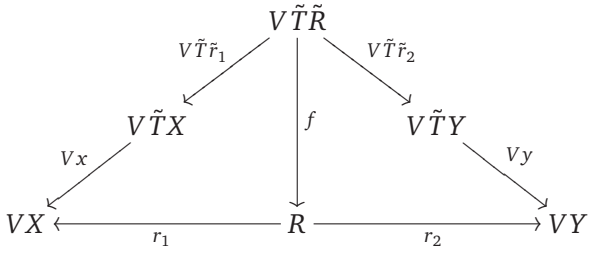
$$X \xleftarrow{x} \tilde{T}X \xleftarrow{\tilde{T}\tilde{r}_1} \tilde{T}\tilde{R} \xrightarrow{\tilde{T}\tilde{r}_2} \tilde{T}Y \xrightarrow{y} Y$$

Thus we have a V-lifting span in $\mathscr{C}$:

$$VX \xleftarrow{Vx} V\tilde{T}X \xleftarrow{V\tilde{T}\tilde{r}_1} V\tilde{T}\tilde{R} \xrightarrow{V\tilde{T}\tilde{r}_2} V\tilde{T}Y \xrightarrow{Vy} VY \quad .$$

Since (R, r_1, r_2) is a final span we have a unique morphism $f : V\tilde{T}\tilde{R} \to R$ such that the following diagram commutes in $\mathscr{C}$.

(6.2.2)



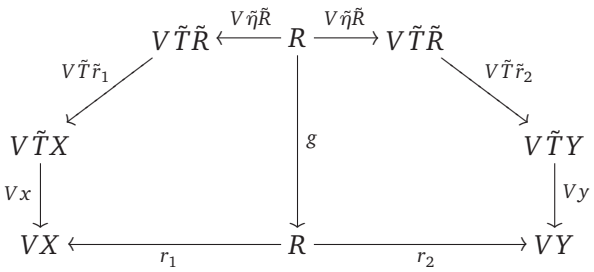
We will show that the composite $r = TR \xrightarrow{\nu\tilde{R}} V\tilde{T}\tilde{R} \xrightarrow{f} R$ equips TR with a $\mathbf{T}$-algebra structure. First, we prove the unit law. Consider the following span in $\mathscr{B}$.

$$X \xleftarrow{x} \tilde{T}X \xleftarrow{\tilde{T}\tilde{r}_1} \tilde{T}\tilde{R} \xleftarrow{\tilde{\eta}\tilde{R}} \tilde{R} \xrightarrow{\tilde{\eta}\tilde{R}} \tilde{T}\tilde{R} \xrightarrow{\tilde{T}\tilde{r}_2} \tilde{T}Y \xrightarrow{y} Y$$

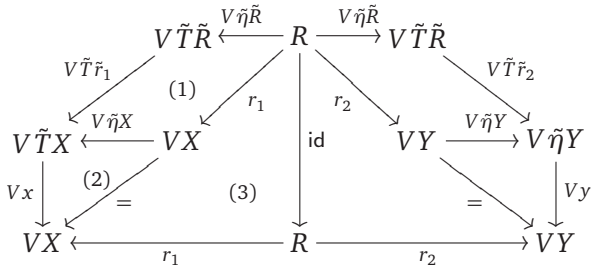
This gives a V-lifting span in $\mathscr{C}$:

$$VX \xleftarrow{Vx} V\tilde{T}X \xleftarrow{V\tilde{T}\tilde{r}_1} V\tilde{T}\tilde{R} \xleftarrow{V\tilde{\eta}\tilde{R}} R \xrightarrow{V\tilde{\eta}\tilde{R}} V\tilde{T}\tilde{R} \xrightarrow{V\tilde{T}r_2} V\tilde{T}Y \xrightarrow{Vy} Y \quad .$$

Because (R, r_1, r_2) is a final V-lifting span, we have a unique morphism $g : R \to R$ making the following diagram commute in $\mathscr{C}$.

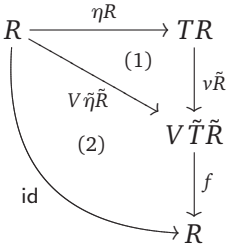


It is clear from diagram 6.2.2 that setting $g = f \circ V\tilde{\eta}R$ makes the diagram commute. But so does setting $g = \mathsf{id}_R$, as the following decomposition shows.



Using: (1) nat. of $\tilde{\eta}$; (2) V applied to the unit law for α; (3) equality. The right-hand side is similar.

Thus we have that $f \circ V\tilde{\eta}\tilde{R} = \mathsf{id}_R$. To conclude the unit law for $r : TR \to R$, we observe that the following diagram commutes in $\mathscr{C}$.

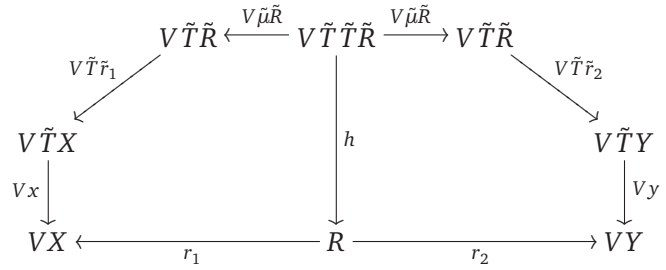


Using: 1: unit law for v; 2: just established.

As for the multiplication law, consider the following span in $\mathscr{B}$.

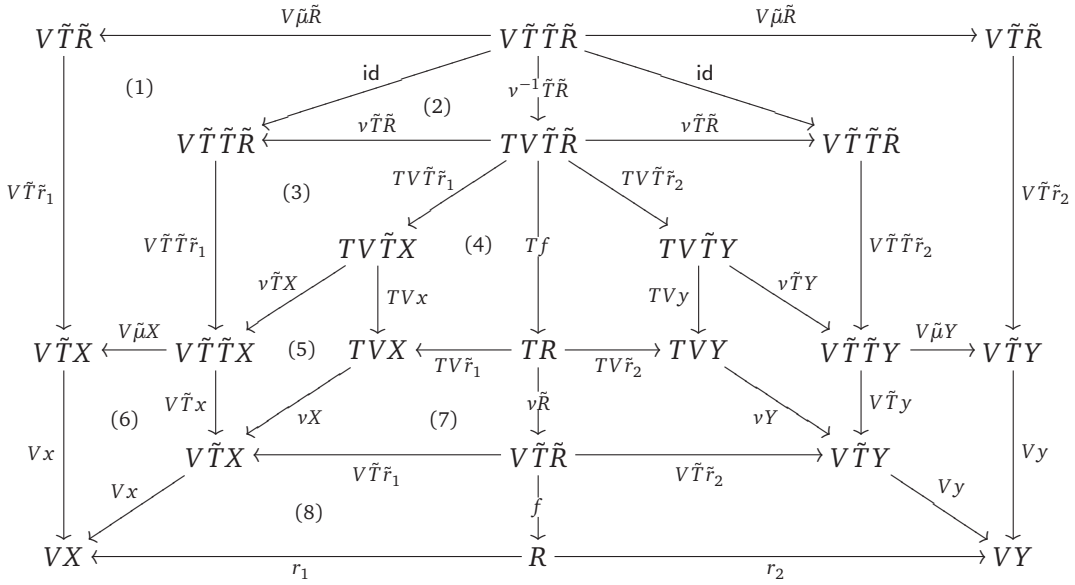
$$X \xleftarrow{x} \tilde{T}X \xleftarrow{\tilde{T}\tilde{r}_1} \tilde{T}\tilde{R} \xleftarrow{\tilde{\mu}\tilde{R}} \tilde{T}\tilde{T}\tilde{R} \xrightarrow{\tilde{\mu}\tilde{R}} \tilde{T}\tilde{R} \xrightarrow{\tilde{T}\tilde{r}_2} \tilde{T}Y \xrightarrow{y} Y \quad .$$

By finality of (R, r_1, r_2), there must be a unique morphism $h : V\tilde{T}\tilde{R} \to R$ in $\mathscr{C}$ making the following diagram commute (in $\mathscr{C}$).



It is clear from diagram 6.2.2 that $h = f \circ V\tilde{\mu}\tilde{R}$ is one such map. Another one is $h = f \circ v\tilde{R} \circ Tf \circ v^{-1}\tilde{T}\tilde{R}$,

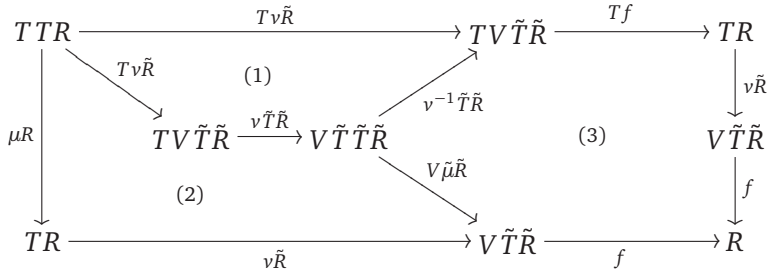
as illustrated in the following decomposition.



Using: (1) nat. of $\tilde{\mu}$; (2) $\nu \circ \nu^{-1} = \mathsf{id}$; (3) nat.of ν; (4) T applied to dgm. 6.2.2; (5) nat.of ν; (6) mult. law for x; (7) nat.of ν; (8) dgm. 6.2.2. The right-hand side is similar.

Hence $f \circ V\tilde{\mu}\tilde{R} = f \circ \nu\tilde{R} \circ Tf \circ \nu^{-1}\tilde{T}\tilde{R}$.

To conclude that $r = f \circ \nu\tilde{R} : TR \to R$ satisfies the multiplication law, consider the following diagram.



Using: (1) $\nu^{-1}\nu = \mathsf{id}$; (2) mult. law for ν; (3) as just established.

Thus we know that $r : TR \to R$ is a $\mathbf{T}$-algebra, and so the final V-lifting span (R, r_1, r_2) is a $\mathbf{T}$-congruence. □

By specialising Theorem 6.2.1 to the case where $\mathscr{B}$ is a category of structured coalgebras, and restricting attention to free $\tilde{\mathbf{T}}$-algebras, we arrive at the following corollary.

Corollary 6.2.3. *Let $U : \mathscr{D} \to \mathscr{C}$ be a functor between categories, and let B be an endofunctor on $\mathscr{C}$. Let $\tilde{\mathbf{T}}$ be a monad on (U,B)-*Coalg *which is a strict lifting of a monad $\mathbf{T}$ on $\mathscr{D}$. Let (X,h), (Y,k) be U-structured B-coalgebras.*

Every final U-structured B-bisimulation between $\tilde{T}(X,h)$ and $\tilde{T}(Y,k)$ is a $\mathbf{T}$-congruence between the free $\mathbf{T}$-algebra on X and the free $\mathbf{T}$-algebra on Y. □

Our approach to Corollary 6.2.3 differs from that of Turi and Plotkin [1997, Corollary 7.5], in that we do not make use of a final coalgebra, or their notion of bialgebra for a distributive law.

Characterisation of monad liftings along forgetful functors. We conclude this subsection with the following elementary observation. Turi and Plotkin [1997, Rem. 5.1] use this result for categories of coalgebras.

Remark 6.2.4. Let $V : \mathscr{B} \to \mathscr{C}$ be a faithful functor between categories. Let $\mathbf{T} = (T, \eta, \mu)$ be a monad on $\mathscr{C}$.

To give a strict lifting of $\mathbf{T}$ along the functor V is to assign to each object X in $\mathscr{B}$ an object $\tilde{T}X$ in $\mathscr{B}$ such that $V\tilde{T}X = TVX$, and such that

1. for each morphism $f : X \to Y$ in $\mathscr{B}$ there is a morphism $\tilde{T}f : \tilde{T}X \to \tilde{T}Y$ in $\mathscr{B}$ such that
$$V\tilde{T}f = TVf : TVX \to TVY \quad ;$$
2. for each object X of $\mathscr{B}$ there is a morphism $\tilde{\eta}_X : X \to \tilde{T}X$ in $\mathscr{B}$ such that
$$V\tilde{\eta}_X = \eta_{VX} : VX \to TVX \quad ; \quad \text{and}$$
3. for each object X of $\mathscr{B}$ there is a morphism $\tilde{\mu}_X : \tilde{T}\tilde{T}X \to X$ in $\mathscr{B}$ such that
$$V\tilde{\mu}_X = \mu_{VX} : TTVX \to TVX \quad .$$

Functoriality of $\tilde{T}$, and naturality of $\tilde{\eta}$ and $\tilde{\mu}$, are guaranteed because V is faithful. □

6.2.2 Parameterised recursion

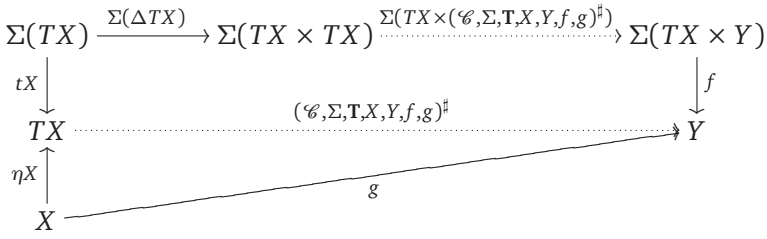
Let $\mathbf{T}$ be a free monad on an endofunctor Σ on a category $\mathscr{C}$ with binary products; we omit the usual subscript Σ for brevity. Let X, Y be objects of $\mathscr{C}$. For any pair of morphisms

$$f : \Sigma(TX \times Y) \to Y \qquad g : X \to Y \tag{6.2.5}$$

in $\mathscr{C}$, the principle of "parameterised recursion" gives a unique morphism

$$(\mathscr{C}, \Sigma, \mathbf{T}, X, Y, f, g)^\sharp : TX \to Y$$

making the following diagram commute.



(6.2.6)

We refer to such a tuple $(\mathscr{C}, \Sigma, \mathbf{T}, X, Y, f, g)$ as *(parameterised) recursion data.*

Taylor [1999, Example 6.1.7] discusses the various forms of recursion. As he demonstrates, parameterised recursion as defined above is exactly the notion that is often used in definitions of unary primitive recursive functions, for instance for the factorial function.

Morphisms of recursion data. By introducing morphisms between items of recursion data, we show, in Lemma 6.2.8, that the operation of parameterised recursion is functorial.

A morphism $(F, \phi, \psi, \alpha, \beta)$ between two items of recursion data, from $(\mathscr{C}, \Sigma, \mathbf{T}, X, Y, f, g)$ to $(\mathscr{C}', \Sigma', \mathbf{T}', X', Y', f', g')$, is a functor $F : \mathscr{C} \to \mathscr{C}'$ that preserves binary products, together with natural transformations $\phi : \Sigma' F \to F\Sigma$ and $\psi : T'F \to FT$, and morphisms $\alpha : X' \to FX$ and $\beta : Y' \to FY$ in $\mathscr{C}'$, all such that we have a morphism of endofunctors $(F, \phi) : (\mathscr{C}, \Sigma) \to (\mathscr{C}', \Sigma')$ inducing a

monad morphism $(F,\psi):(\mathscr{C},\mathbf{T})\to(\mathscr{C}',\mathbf{T}')$, and such that the following diagrams commute. (Diagram (a) is in the functor category $[\mathscr{C},\mathscr{C}']$, while diagrams (b) and (c) are in $\mathscr{C}'$.)

(a)
$$\begin{array}{ccc}
\Sigma'T'F & \xrightarrow{t'F} & T'F \\
{\scriptstyle\Sigma'\psi}\downarrow & & \downarrow{\scriptstyle\psi} \\
\Sigma'FT & & \\
{\scriptstyle\phi T'}\downarrow & & \\
F\Sigma T & \xrightarrow[Ft]{} & FT
\end{array}$$

(b)
$$\begin{array}{ccc}
X' & \xrightarrow{g'} & Y' \\
{\scriptstyle\alpha}\downarrow & & \downarrow{\scriptstyle\beta} \\
FX & \xrightarrow[Fg]{} & FY
\end{array}$$

(c)
$$\begin{array}{ccc}
\Sigma'(T'X'\times Y') & \xrightarrow{f'} & Y' \\
{\scriptstyle\Sigma'(T'\alpha\times\beta)}\downarrow & & \\
\Sigma'(T'FX\times FY) & & \\
{\scriptstyle\Sigma'(\psi X\times FY)}\downarrow & & \\
\Sigma'(FTX\times FY) & & \downarrow{\scriptstyle\beta} \\
{\scriptstyle\wr}\downarrow & & \\
\Sigma'F(TX\times Y) & & \\
{\scriptstyle\phi(TX\times Y)}\downarrow & & \\
F\Sigma(TX\times Y) & \xrightarrow[Ff]{} & FY
\end{array}$$

(6.2.7)

Note that diagram 6.2.7(a) says that $(F,\psi):(\mathscr{C},\mathbf{T})\to(\mathscr{C}',\mathbf{T}')$ is the monad morphism induced by the morphism of endofunctors, $(F,\phi):(\mathscr{C},\Sigma)\to(\mathscr{C}',\Sigma')$.

Lemma 6.2.8. Let $(F,\phi,\psi,\alpha,\beta)$ be a morphism between two items of recursion data, say from $d=(\mathscr{C},\Sigma,\mathbf{T},X,Y,f,g)$ to $d'=(\mathscr{C}',\Sigma',\mathbf{T}',X',Y',f',g')$. Then the following diagram commutes in $\mathscr{C}'$.

$$\begin{array}{ccc}
T'X' & \xrightarrow{d'^\sharp} & Y \\
{\scriptstyle T'\alpha}\downarrow & & \\
T'FX & & \downarrow{\scriptstyle\beta} \\
{\scriptstyle\psi X}\downarrow & & \\
FTX & \xrightarrow[Fd^\sharp]{} & FY
\end{array}$$

(6.2.9)

Proof. By considering the recursion data

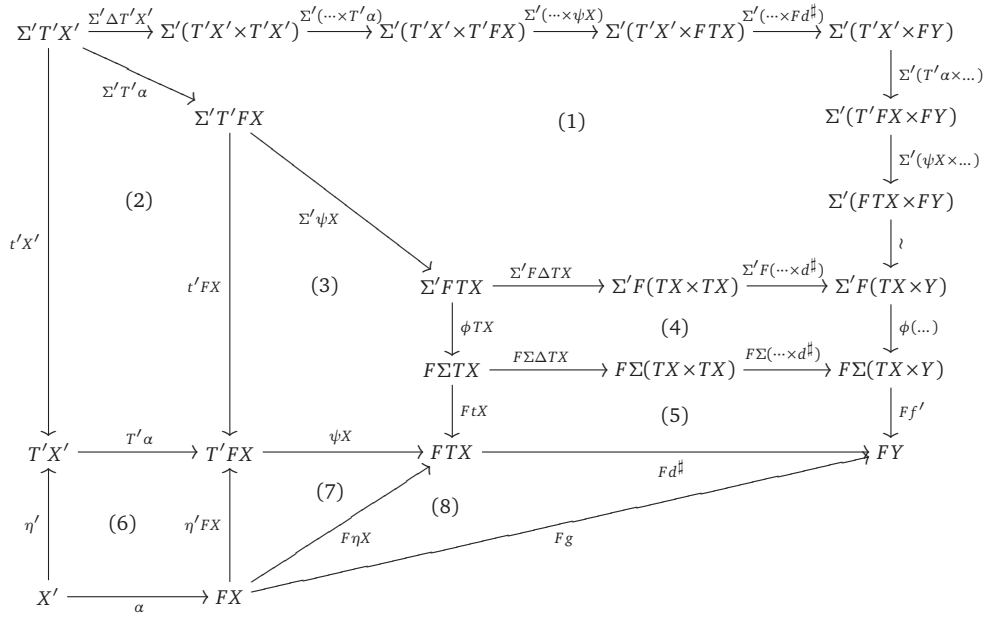
$$d''=\left(\begin{array}{l}
\mathscr{C}',\Sigma',\mathbf{T}',X',FY, \\
f''=\ \Sigma'(T'X'\times FY)\xrightarrow{\Sigma'(T'\alpha\times FY)}\Sigma'(T'FX\times FY)\xrightarrow{\Sigma'(\psi X\times FY)}\Sigma'(FTX\times FY) \\
\qquad\cong\Sigma'F(TX\times Y)\xrightarrow{\phi(\ldots)}F\Sigma(TX\times Y)\xrightarrow{Ff}FY\ , \\
g''=\ X'\xrightarrow{\alpha}FX\xrightarrow{Fg}FY
\end{array}\right)$$

we see that there is exactly one arrow $d''^\sharp:T'X'\to FY$ making the following diagram commute.

$$\begin{array}{ccccc}
\Sigma'(T'X') & \xrightarrow{\Sigma'(\Delta T'X')} & \Sigma'(T'X'\times T'X') & \xrightarrow{\Sigma'(T'X'\times d''^\sharp)} & \Sigma'(T'X'\times FY) \\
{\scriptstyle t'X'}\downarrow & & & & \downarrow{\scriptstyle f''} \\
T'X' & & \xrightarrow{d''^\sharp} & & FY \\
{\scriptstyle\eta'X'}\uparrow & & & \nearrow{\scriptstyle g''} & \\
X' & & & &
\end{array}$$

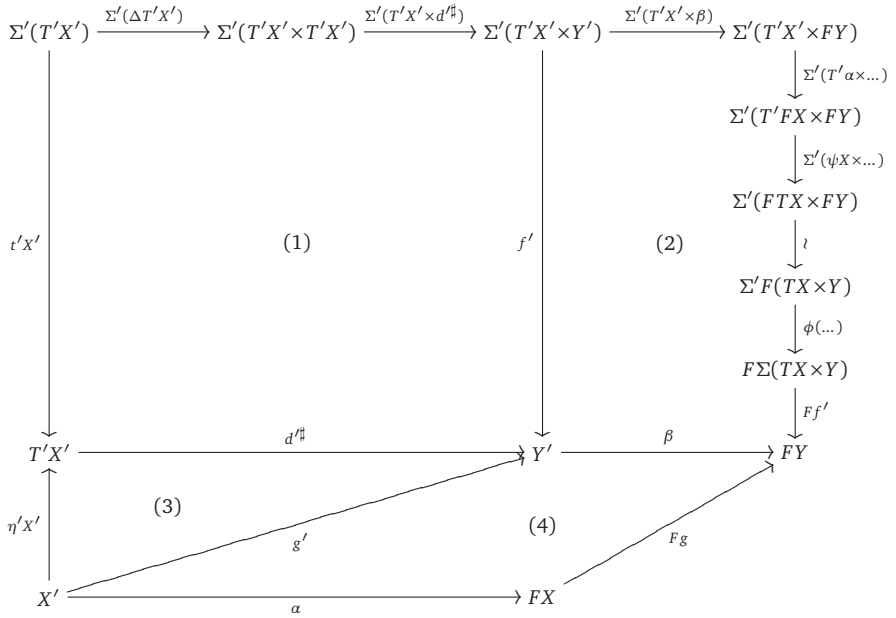
(6.2.10)

We will show that $d''^\sharp$ could be either of the two sides of diagram 6.2.9. For the case of $d''^\sharp=Fd^\sharp\circ\psi X\circ T'\alpha$, consider the following decomposition.



Using: (1) properties of products; (2) nat. of t'; (3) dgm. 6.2.7(a); (4) nat. of ϕ; (5) F applied to defn. of $d^\sharp$; (6) nat. of η'; (7) unit law for ψ; (8) F applied to defn. of $d^\sharp$.

For the case of $d''^\sharp = \beta \circ d'^\sharp$, consider the following decomposition.



Using: (1) defn. of $d'^\sharp$; (2) dgm. 6.2.7(c); (3) defn. of $d'^\sharp$; (4) dgm. 6.2.7(b).

Since the mediating morphism $d''^\sharp$ of diagram 6.2.10 must be unique, we can conclude that diagram 6.2.9 commutes. □

6.2.3 Abstract rules inducing monad liftings

Definition 6.2.11. Let $U : \mathscr{D} \to \mathscr{C}$ be a functor between categories, and let B and Σ be endofunctors on $\mathscr{C}$. Suppose that the free monad $\mathbf{T}$ on Σ exists, and that $\mathbf{T}$ lifts along U to a monad $\tilde{\mathbf{T}}$ on $\mathscr{D}$. We write $u : TU \to U\tilde{T}$ for the lifting isomorphism.

An *abstract rule for* $(\mathscr{C}, \mathscr{D}, U, B, \Sigma, \tilde{\mathbf{T}})$ is a natural transformation

$$\rho : \Sigma(U \times BU) \to BU\tilde{T} \quad .$$

Abstract rule induction. Together, an abstract rule of this form and a U-structured B-coalgebra, $(X, h : UX \to BUX)$, provide parameterised recursion data

$$(\mathscr{C}, \Sigma, \mathbf{T}, UX, BU\tilde{T}X, f_{\rho,X}, g_{X,h})$$

where the morphisms $f_{\rho,X}$, $g_{X,h}$ are defined as follows.

$$\begin{aligned} f_{\rho,X} &= \Sigma(TUX \times BU\tilde{T}X) \xrightarrow{\Sigma(uX\times\ldots)} \Sigma(U\tilde{T}X \times BU\tilde{T}X) \xrightarrow{\rho\tilde{T}X} BU\tilde{T}\tilde{T}X \xrightarrow{BU\tilde{\mu}X} BU\tilde{T}X \\ g_{X,h} &= UX \xrightarrow{h} BUX \xrightarrow{BU\tilde{\eta}X} BU\tilde{T}X. \end{aligned} \tag{6.2.12}$$

Hence an abstract rule ρ gives rise to an operator T_ρ which assigns to each U-structured B-coalgebra (X, h) a U-structured B-coalgebra with carrier $\tilde{T}X$, and with structure given by

$$T_\rho h = U\tilde{T}X \xrightarrow{u^{-1}X} TUX \xrightarrow{(\mathscr{C},\Sigma,\mathbf{T},UX,BU\tilde{T}X,f_{\rho,X},g_{X,h})^\sharp} BU\tilde{T}X \quad . \tag{6.2.13}$$

Theorem 6.2.14. *The operator T_ρ defined in equation 6.2.13 defines a strict lifting of the monad $\tilde{\mathbf{T}}$ along the forgetful functor from the category of U-structured B-coalgebras.*

Proof. The forgetful functor (U, B)-Coalg $\to \mathscr{D}$ is faithful, so we make use of the characterisation highlighted in Remark 6.2.4. We begin by proving requirement (1) of that remark: for any coalgebra homomorphism $\alpha : (X, h) \to (Y, k)$, the morphism $\tilde{T}\alpha$ in $\mathscr{D}$ is a map of coalgebras, *i.e.* the following diagram commutes in $\mathscr{C}$.

$$\begin{array}{ccc} U\tilde{T}X & \xrightarrow{U\tilde{T}\alpha} & U\tilde{T}Y \\ {\scriptstyle u^{-1}X}\downarrow & & \downarrow{\scriptstyle u^{-1}Y} \\ TUX & \xrightarrow{TU\alpha} & TUY \\ {\scriptstyle (\mathscr{C},\Sigma,\mathbf{T},UX,BU\tilde{T}X,f_{\rho,X},g_{X,h})^\sharp}\downarrow & & \downarrow{\scriptstyle (\mathscr{C},\Sigma,\mathbf{T},UY,BU\tilde{T}Y,f_{\rho,Y},g_{Y,k})^\sharp} \\ BU\tilde{T}X & \xrightarrow[BU\tilde{T}\alpha]{} & BU\tilde{T}Y \end{array} \tag{6.2.15}$$

The top square commutes by naturality of u^{-1}. For the lower square, we appeal to Lemma 6.2.8, using the following morphism between items of recursion data.

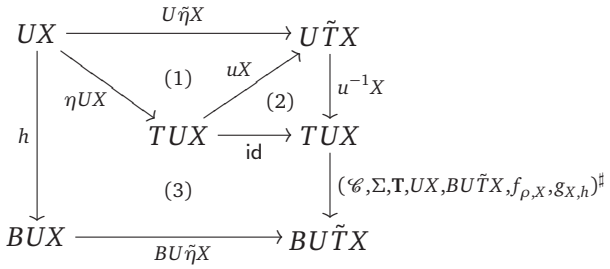
$$(\mathsf{id}_{\mathscr{C}}, \mathsf{id}_\Sigma, \mathsf{id}_{\mathbf{T}}, U\alpha, BU\tilde{T}\alpha) : (\mathscr{C}, \Sigma, \mathbf{T}, UX, BU\tilde{T}X, f_{\rho,X}, g_{X,h}) \to (\mathscr{C}, \Sigma, \mathbf{T}, UY, BU\tilde{T}Y, f_{\rho,Y}, g_{Y,k})$$

We check that this is a valid morphism, that is, that diagrams 6.2.7(a–c) commute. Diagram 6.2.7(a) is trivial in this case. Diagram 6.2.7(b) commutes since α is a U-structured B-coalgebra homomorphism and $\tilde{\eta}$ is natural. Diagram 6.2.7(c) commutes since $f_{\rho,X}$ is natural in X.

For requirements (2) and (3) of Remark 6.2.4, we must show that the unit and multiplication of the monad $\tilde{\mathbf{T}}$ lift to homomorphisms of coalgebras, *i.e.* that the following diagrams commute in $\mathscr{C}$.

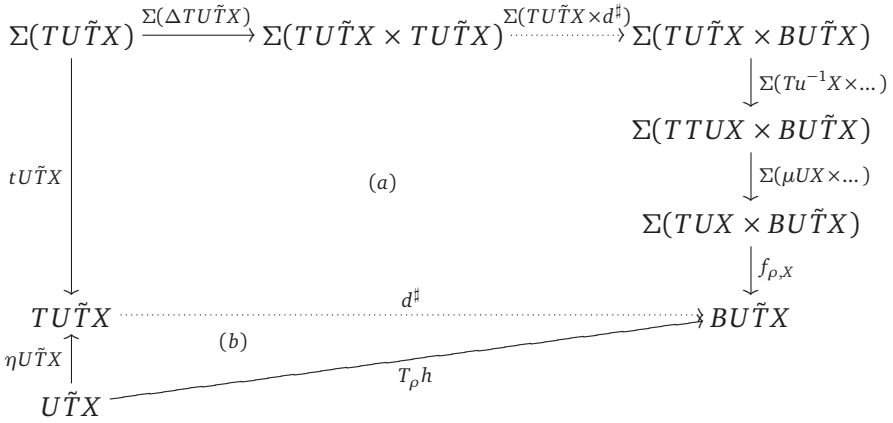
$$\text{(a)}\quad \begin{array}{ccc} UX & \xrightarrow{U\tilde{\eta}X} & U\tilde{T}X \\ {\scriptstyle h}\downarrow & & \downarrow{\scriptstyle T_\rho h} \\ BUX & \xrightarrow[BU\tilde{\eta}X]{} & BU\tilde{T}X \end{array} \qquad \text{(b)}\quad \begin{array}{ccc} U\tilde{T}\tilde{T}X & \xrightarrow{U\tilde{\mu}X} & U\tilde{T}X \\ {\scriptstyle T_\rho(T_\rho h)}\downarrow & & \downarrow{\scriptstyle T_\rho h} \\ BU\tilde{T}\tilde{T}X & \xrightarrow[BU\tilde{\mu}X]{} & BU\tilde{T}X \end{array} \tag{6.2.16}$$

To see that diagram 6.2.16(a) commutes, consider the following triangulation.



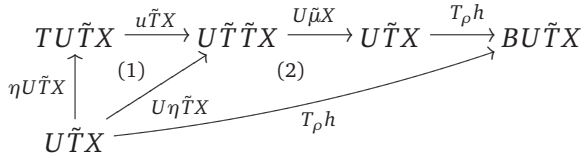
Using: (1) unit law for u; (2) $u^{-1} \circ u = \mathsf{id}$; (3) dgm. 6.2.6.

For diagram 6.2.16(b), multiplication, we proceed as follows. By parameterised initiality there is a unique map $d^\sharp : TU\tilde{T}X \to BTUX$ making the following diagram commute.



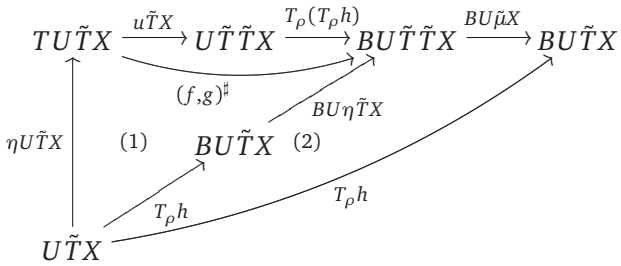
(6.2.17)

We will show that such $d^\sharp$ is found by precomposing either side of diagram 6.2.16(b) with $u\tilde{T}X$. For $d^\sharp = T_\rho h \circ U\tilde{\mu}X \circ u\tilde{T}X$, we see that part (a) of diagram 6.2.17 commutes by considering the decomposition in Figure 6.1. As for part (b) of diagram 6.2.17, consider the following decomposition.



Using: (1) unit law for u; (2) unit law for $\tilde{T}$.

For the other case, $d^\sharp = BU\tilde{\mu}X \circ T_\rho(T_\rho h) \circ u\tilde{T}X$, we see that part (a) of diagram 6.2.17 commutes by considering the decomposition in Figure 6.2. As for part (b) of diagram 6.2.17, consider the following decomposition. Here, the arrow labelled $(f,g)^\sharp$ represents the morphism $(\mathscr{C}, \Sigma, \mathbf{T}, U\tilde{T}X, BU\tilde{T}\tilde{T}X, f_{\rho,\tilde{T}X}, g_{\tilde{T}X,T_\rho h})^\sharp : TUX \to BU\tilde{T}X$.



Using: (1) defn. of $(f,g)^\sharp$; (2) unit law for $\tilde{\mathbf{T}}$.

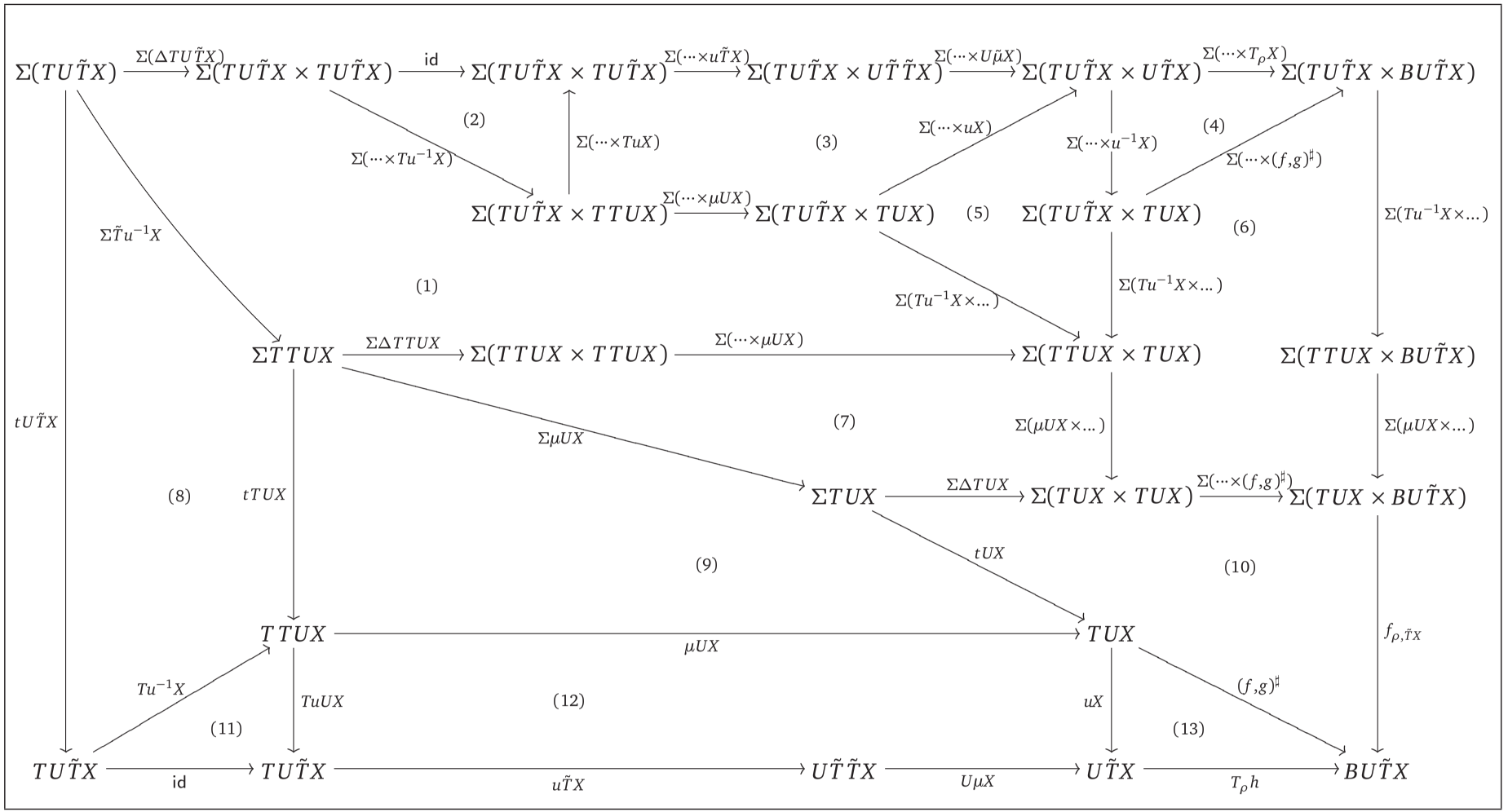


Figure 6.1: Proof that $d^\sharp = T_\rho h \circ U\tilde{\mu}X \circ u\tilde{T}X$ is a mediating morphism for diagram 6.2.17(a). Here, the arrows labelled $(f,g)^\sharp$ represent the morphism $(\mathscr{C}, \Sigma, \mathbf{T}, UX, BU\tilde{T}X, f_{\rho,X}, g_{X,h})^\sharp : TUX \to BU\tilde{T}X$.

Using: (1) nature of products; (2) $u \circ u^{-1} = \mathsf{id}$; (3) mult. law for u; (4) defn. of $T_\rho h$; (5) $u^{-1} \circ u = \mathsf{id}$; (6) nature of products; (7) nature of products; (8) nat. of t; (9) since μ is a homomorphism of Σ-algebras; (10) defn. of $(f,g)^\sharp$; (11) $u \circ u^{-1} = \mathsf{id}$; (12) mult. law for u; (13) defn. of $T_\rho h$.

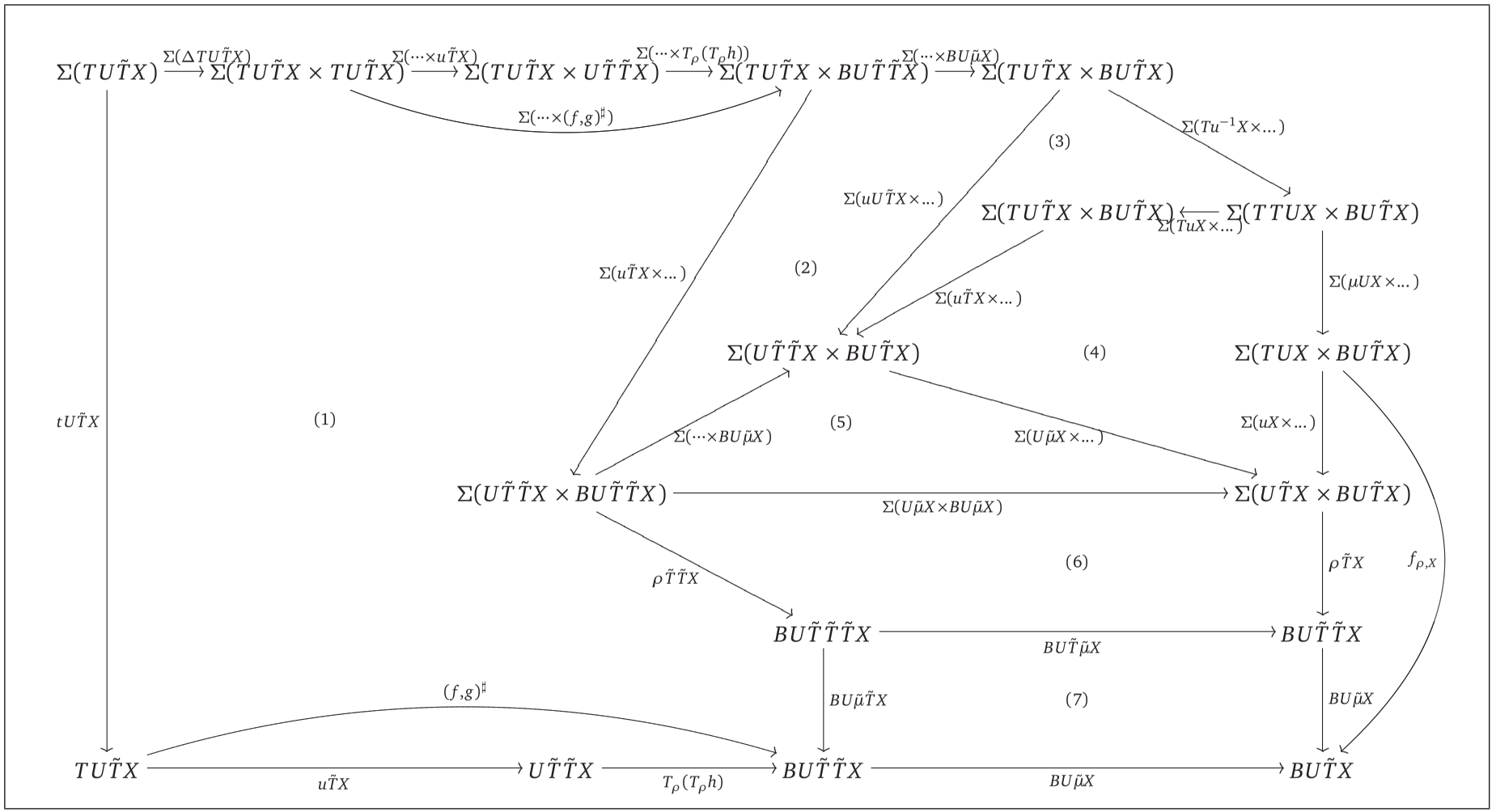


Figure 6.2: Proof that $d^\sharp = BU\tilde{\mu}X \circ T_\rho(T_\rho h) \circ uT\tilde{X}$ is a mediating morphism for diagram 6.2.17. Here, the arrows labelled $(f,g)^\sharp$ represent the morphism $(\mathscr{C}, \Sigma, \mathbf{T}, U\tilde{T}X, BU\tilde{T}\tilde{T}X, f_{\rho,\tilde{T}X}, g_{\tilde{T}X,T_\rho h})^\sharp : TUX \to BU\tilde{T}X$.

Using: (1) defn. of $(f,g)^\sharp$; (2) nat.of $\tilde{\mu}$; (3) $u \circ u^{-1} = \mathsf{id}$; (4) mult. law for u; (5) nature of product; (6) nat. of ρ; (7) mult. law.

Such a mediating map $d^\sharp$ must be unique, and so we know that

$$T_\rho h \circ U\tilde{\mu}X \circ u\tilde{T}X = BU\tilde{\mu}X \circ T_\rho(T_\rho h) \circ u\tilde{T}X \quad .$$

Because $u\tilde{T}X$ is an isomorphism, we can conclude that diagram 6.2.16(b) commutes.

Thus we have established all the requirements of Remark 6.2.4, and we can conclude that we have a lifting $\mathbf{T}_\rho$ of the monad $\mathbf{T}$. □

In Section 6.3.5, and again in Chapter 8, it will be seen that the inductive method of Theorem 6.2.14 for lifting a monad to a category of coalgebras can be understood as the definition of transition systems by rule induction.

6.2.4 Rule induction, when the structure functor has a right adjoint

We retain the notation of the previous subsection.

As we observed in Example 2.4.3(4), if the structure functor $U : \mathscr{D} \to \mathscr{C}$ has a right adjoint $R : \mathscr{C} \to \mathscr{D}$ then we have an isomorphism of categories (U,B)-Coalg $\cong RBU$-Coalg. We now explain how, under mild conditions, we can use an abstract rule of the type used in the previous subsection to lift the monad $\tilde{\mathbf{T}}$ on $\mathscr{D}$ directly to the category RBU-Coalg. Thus the problem is reduced to that considered by Turi and Plotkin [1997].

Preliminaries. Suppose now that the structure functor $U : \mathscr{D} \to \mathscr{C}$ has a right adjoint $R : \mathscr{C} \to \mathscr{D}$, and also that U preserves binary products. We write $\eta^\dashv : \mathrm{id}_\mathscr{D} \to RU$ for the unit of the adjunction $(U \dashv R)$, and $\varepsilon^\dashv : UR \to \mathrm{id}_\mathscr{C}$ for the counit; we use the superscript $\dashv$ here to distinguish from the units of the free monads that are under consideration.

We will also need to suppose that the endofunctor Σ on $\mathscr{C}$ lifts along U to an endofunctor $\tilde{\Sigma}$ on $\mathscr{D}$, and that $\tilde{\mathbf{T}}$ is the free monad on $\tilde{\Sigma}$. We write u both for the endofunctor lifting isomorphism $\Sigma U \to U\tilde{\Sigma}$ and for the monad lifting isomorphism $TU \to U\tilde{T}$; we assume that the latter is the extension of the former according to part (1) of Theorem 6.1.5.

By Prop. 6.1.4(1) we have a morphism of endofunctors $(R,r) : (\mathscr{C},\Sigma) \to (\mathscr{D},\tilde{\Sigma})$ which is right adjoint in Endo to the lifting $(U,u) : (\mathscr{D},\tilde{\Sigma}) \to (\mathscr{C},\Sigma)$. Similarly, by Prop. 6.1.4(2), we have a morphism of monads $(R,r) : (\mathscr{C},\mathbf{T}) \to (\mathscr{D},\tilde{\mathbf{T}})$ which is right adjoint in Monad to the lifting $(U,u) : (\mathscr{D},\tilde{\mathbf{T}}) \to (\mathscr{C},\mathbf{T})$. Note that, again, we use the same notation for both the morphism of endofunctors and the morphism of monads, since here the latter is the extension of the former.

The fact that the unit $\eta^\dashv$ of the adjunction is a morphism of endofunctors and of monads means that the following diagrams commute in the functor category $[\mathscr{D},\mathscr{D}]$.

$$\begin{array}{ccc} \tilde{\Sigma} & \xrightarrow{\tilde{\Sigma}\eta^\dashv} & \tilde{\Sigma}RU \\ & {\scriptstyle \eta^\dashv\tilde{\Sigma}} \searrow & \downarrow{\scriptstyle rU} \\ & & R\Sigma U \\ & & \downarrow{\scriptstyle Ru} \\ & & RU\tilde{\Sigma} \end{array} \qquad \begin{array}{ccc} \tilde{T} & \xrightarrow{\tilde{T}\eta^\dashv} & \tilde{T}RU \\ & {\scriptstyle \eta^\dashv\tilde{T}} \searrow & \downarrow{\scriptstyle rU} \\ & & RTU \\ & & \downarrow{\scriptstyle Ru} \\ & & RU\tilde{T} \end{array} \tag{6.2.18}$$

Derivation of an abstract rule for $(\mathscr{D},\mathscr{D},\mathrm{id}_\mathscr{D},RBU,\tilde{\Sigma},\tilde{\mathbf{T}})$. An abstract rule ρ for data $(\mathscr{C},\mathscr{D},U,B,\Sigma,\tilde{\mathbf{T}})$, in the form of Definition 6.2.11, gives rise to an abstract rule

$$\bar{\rho} : \tilde{\Sigma}(- \times RBU) \to RBU \tag{6.2.19}$$

for $(\mathscr{D},\mathscr{D},\mathrm{id}_\mathscr{D},RBU,\tilde{\Sigma},\tilde{\mathbf{T}})$, as follows: precompose ρ with the counit $\varepsilon^\dashv$,

$$\Sigma(U - \times URBU) \xrightarrow{\Sigma(\cdots\times\varepsilon^\dashv BU)} \Sigma(U - \times BU) \xrightarrow{\rho} BU\tilde{T} \quad ;$$

then use the lifting isomorphism and the fact that U preserves products to arrive at a natural transformation

$$U\tilde{\Sigma}(- \times RBU) \to BU \quad ;$$

and finally transpose across the adjunction $(U \dashv R)$ to obtain a natural transformation of the form of (6.2.19).

In summary, $\bar{\rho}$ is the following composite.

$$\begin{array}{c}
\tilde{\Sigma}(- \times RBU) \\
\downarrow{\scriptstyle \eta^{\dashv}\tilde{\Sigma}(\dots)} \\
RU\tilde{\Sigma}(- \times RBU) \\
\downarrow{\scriptstyle Ru^{-1}(\dots)} \\
R\Sigma U(- \times RBU) \\
\downarrow{\scriptstyle \wr} \\
R\Sigma(U \times URBU) \\
\downarrow{\scriptstyle R\tilde{\Sigma}(U \times \varepsilon^{\dashv} BU)} \\
R\Sigma(U \times BU) \\
\downarrow{\scriptstyle R\rho} \\
RBU\tilde{T}
\end{array}$$

Relating the derived rule with the original. Using this abstract rule $\bar{\rho}$, we can lift the monad $\tilde{\mathbf{T}}$ on $\mathscr{D}$ to an monad $\mathbf{T}_{\bar{\rho}}$ on the category RBU-Coalg of coalgebras. As we now show, this monad $\mathbf{T}_{\bar{\rho}}$ on the category RBU-Coalg corresponds to the monad $\mathbf{T}_{\rho}$ on the isomorphic category (U,B)-Coalg.

Theorem 6.2.20. *The isomorphism of categories* (U,B)-Coalg $\cong$ RBU-Coalg *lifts to an isomorphism* $((U,B)\text{-Coalg}, \mathbf{T}_{\rho}) \cong (RBU\text{-Coalg}, \mathbf{T}_{\bar{\rho}})$ *of monads.*

Proof. Any isomorphism is necessarily faithful, and so, following Remark 6.2.4, it is sufficient to show that

> for every U-structured B-coalgebra $(X, h : UX \to BUX)$, the left adjunct of $T_{\rho}(h)$ is $T_{\bar{\rho}}(h^{\dagger})$, writing $h^{\dagger}$ for the right adjunct of h.

In other words, we must show that, for every RBU-coalgebra $(X, h : X \to RBUX)$, the following diagram commutes.

$$\begin{array}{ccc}
\tilde{T}X & & \\
{\scriptstyle \eta^{\dashv}\tilde{T}X}\downarrow & \searrow^{T_{\bar{\rho}}(Rh \circ \eta^{\dashv} X)} & \\
RU\tilde{T}X & \xrightarrow[R\mathbf{T}_{\rho}h]{} & RBU\tilde{T}X
\end{array}$$

We prove this by considering the following decomposition.

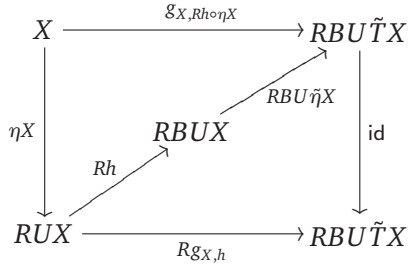
$$\begin{array}{ccccc}
 & & \tilde{T}X & \xrightarrow{T_{\bar{\rho}}(Rh \circ \eta^{\dashv} X)} & RUB\tilde{T}X \\
 & {\scriptstyle \eta^{\dashv}\tilde{T}X}\swarrow & {\scriptstyle \tilde{T}\eta^{\dashv}X}\downarrow & & \\
 & (1) & & & \\
RU\tilde{T}X & & \tilde{T}RUX & (3) & {\scriptstyle \mathsf{id}}\downarrow \\
{\scriptstyle \mathsf{id}}\downarrow & \overset{RuX}{\nwarrow} & {\scriptstyle rUX}\downarrow & & \\
 & (2) & & & \\
RU\tilde{T}X & \xrightarrow{Ru^{-1}X} & RTUX & \xrightarrow{(\mathscr{C},\Sigma,\mathbf{T},UX,BU\tilde{T}X,f_{\rho,X},g_{X,h})^{\sharp}} & RBU\tilde{T}X \\
 & & & \underset{RT_{\rho}h}{\searrow} & \uparrow
\end{array} \tag{6.2.21}$$

Part (1) is diagram 6.2.18, and part (2) follows since $u \circ u^{-1} = \mathsf{id}$. For part (3), note that, by definition, $T_{\bar{\rho}}(Rh \circ \eta^{\dashv} X) = (\mathscr{D}, \tilde{\Sigma}, \tilde{\mathbf{T}}, X, RBU\tilde{T}X, f_{\bar{\rho},X}, g_{X,(Rh\circ\eta^{\dashv}X)})^{\sharp}$. Now, we consider the morphism of recursion data

$$(R, r, r, \eta^{\dashv} X, \mathsf{id}) : (\mathscr{C}, \Sigma, \mathbf{T}, UX, BU\tilde{T}X, f_{\rho,X}, g_{X,h}) \to (\mathscr{D}, \tilde{\Sigma}, \tilde{\mathbf{T}}, X, RBU\tilde{T}X, f_{\bar{\rho},X}, g_{X,(Rh\circ\eta^{\dashv}X)})$$

and we will use Lemma 6.2.8 to show that part (3) commutes. Since R is a right adjoint, it certainly preserves binary products, and so it remains for us to show that the three diagrams in (6.2.7) commute. Diagram (a) is straightforward, since the morphism of monads $(R, r) : (\mathscr{C}, \mathbf{T}) \to (\mathscr{D}, \tilde{\mathbf{T}})$ is chosen to be the extension of the morphism of endofunctors $(R, r) : (\mathscr{C}, \Sigma) \to (\mathscr{D}, \tilde{\Sigma})$.

For diagram (b), we consider the following decomposition.



To see that diagram (c) commutes, consider the decomposition given in Figure 6.3. □

6.3 The Positive GSOS rule format

We now illustrate some aspects of the theory of the previous sections by considering concrete rules in a positive version of the GSOS format of Bloom, Istrail, and Meyer [1995].

Notation. As in Section 2.1, we fix a set of labels Lab. We also fix a signature $\mathbb{S}$, in the sense of Definition 6.1.1; we will make use of the endofunctor $\Sigma_{\mathbb{S},\mathbf{Set}}$ on the category **Set** of sets as defined in equation 6.1.3. (Hereafter we write $\Sigma_{\mathbb{S}}$ for $\Sigma_{\mathbb{S},\mathbf{Set}}$, since we will only consider the category of sets in this section.)

Because **Set** satisfies the conditions of Prop. 6.1.6, there is a free monad $\mathbf{T}_{\mathbb{S},\mathbf{Set}}$ on $\Sigma_{\mathbb{S},\mathbf{Set}}$. (Hereafter we write $\mathbf{T}_{\mathbb{S}} = (T_{\mathbb{S}}, \eta_{\mathbb{S}}, \mu_{\mathbb{S}})$ for this free monad.) For each set X, elements of the set $T_{\mathbb{S}}X$ are to be thought of as terms built out of the operators of $\mathbb{S}$, with free variables taken from X.

6.3.1 Rule structures

Definition 6.3.1. A *premise structure* over a set $\mathtt{X}$ of variables is a triple in $\mathtt{X} \times \mathrm{Lab} \times \mathtt{X}$ with the components respectively named the source, action and target. A *conclusion structure* over a set $\mathtt{X}$ of variables is a triple in $\Sigma_{\mathbb{S}}\mathtt{X} \times \mathrm{Lab} \times T_{\mathbb{S}}\mathtt{X}$ with the components respectively named the source, action and target.

A *rule structure* $\mathtt{R}$ over a set $\mathtt{X}$ of variables is a set Prems of premise structures over $\mathtt{X}$ and a conclusion structure $(\mathtt{src}, \underline{l}, \mathtt{tar})$ over $\mathtt{X}$.

We will often refer to classes $\mathscr{R}$ of rule structures $\mathtt{R}$; the intention is not that each rule structure $\mathtt{R}$ in $\mathscr{R}$ is over a common set of variables, but rather that each rule structure $\mathtt{R}$ has an implicit set of variables over which it is defined.

Notice that we use a typewriter font to denote the set $\mathtt{X}$ of variables. We will use the same font to notate elements of $\mathtt{X}$. So the symbol $\mathtt{x}$ is different from the symbol x.

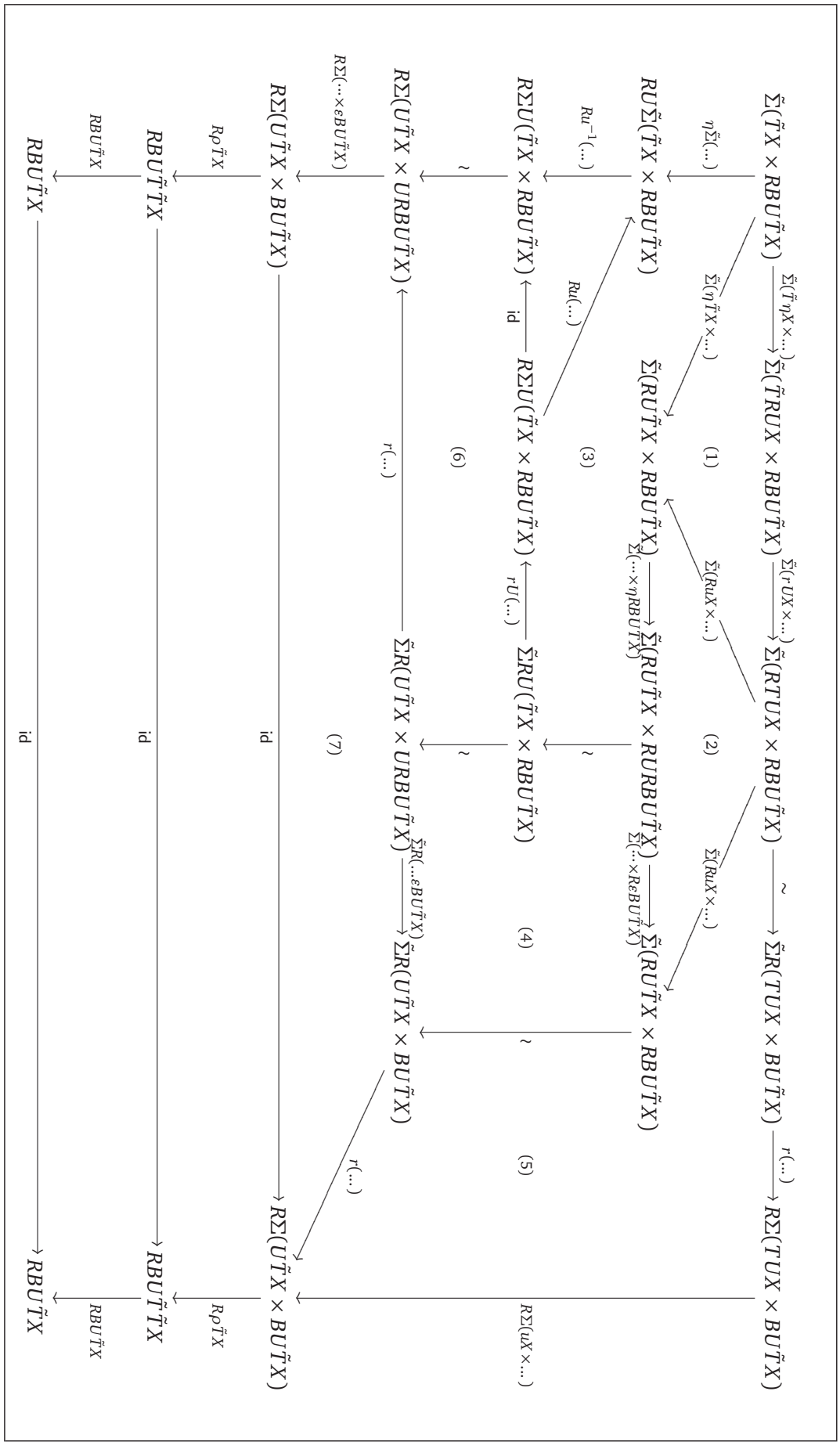


Figure 6.3: Proof that diagram 6.2.7(c) commutes, as required for using Lemma 6.2.8 to prove that part (3) of diagram 6.2.21 commutes.

Using: (1) dgm. 6.2.18; (2) triangle id.; (3) dgm. 6.2.18; (4) nat. of product preservation; (5) nat. of r; (6) nat. of r; (7) nat. of r.

Rule structures as first-order logic formulae. Suppose that we have a class $\mathscr{R}$ of rule structures. We will explain how this class can be considered as a theory of (potentially) infinite logic, for the language with operations $\mathbb{S}$ and with a binary predicate $\xrightarrow{l}$ for each label $l \in \mathrm{Lab}$. We note that, for each set $\mathtt{X}$ of variables, every element of $\Sigma_{\mathbb{S}}\mathtt{X}$ and every element of $T_{\mathbb{S}}\mathtt{X}$ is a term in the language, with free variables in the set $\mathtt{X}$.

For each rule structure $\mathtt{R} \in \mathscr{R}$, we define a formula $\Phi_{\mathtt{R}}$ as the following Horn clause.

$$\Phi_{\mathtt{R}} = \left(\bigwedge_{(\mathtt{x},l,\mathtt{y}) \in \mathrm{Prems}} \mathtt{x} \xrightarrow{l} \mathtt{y} \right) \implies \mathtt{src} \xrightarrow{l} \mathtt{tar} \tag{6.3.2}$$

Now, the theory associated to $\mathscr{R}$ has an axiom for each $\mathtt{R}$, as we now explain. If $\mathtt{X}$ is the set of variables over which $\mathtt{R}$ is defined, then the axiom associated to $\mathtt{R}$ is:

$$\forall \mathtt{X}.\ \Phi_{\mathtt{R}} \quad .$$

We use this notation to indicate simultaneous universal quantification in the formula $\Phi_{\mathtt{R}}$ of every variable in the set $\mathtt{X}$ — this is a convention that is common in infinitary logics. If we can enumerate the variables in $\mathtt{X}$, say $\mathtt{X} = \{\mathtt{x}_1, \ldots, \mathtt{x}_{|\mathtt{X}|}\}$, then we have

$$\forall \mathtt{X}.\ \Phi_{\mathtt{R}} \iff \forall \mathtt{x}_1, \ldots, \mathtt{x}_{|\mathtt{X}|}.\ \Phi_{\mathtt{R}} \quad .$$

Models of rule structures. A structure for the language of the above theory we call an *$\mathscr{R}$-structure*. That is, an $\mathscr{R}$-structure is a set X equipped with a $\Sigma_{\mathbb{S}}$-algebra structure $\alpha : \Sigma_{\mathbb{S}}X \to X$ and a labelled transition relation $\longrightarrow \subseteq X \times \mathrm{Lab} \times X$.

We say that such an $\mathscr{R}$-structure is an *$\mathscr{R}$-model* if the transition relation $\longrightarrow \subseteq X \times \mathrm{Lab} \times X$ satisfies every instance of each formula $\Phi_{\mathtt{R}}$, for $\mathtt{R} \in \mathscr{R}$. Simpson [1995, Sec. 2] restricts the notion of model by only allowing transitions out of elements in the image of α when they are derivable from some $\Phi_{\mathtt{R}}$. But of principle interest for operational semantics is the model that Simpson says is *intended*; in this case it is defined as follows:

- the carrier set is $T_{\mathbb{S}}\emptyset$, the set of free terms of the signature $\mathbb{S}$;
- the $\Sigma_{\mathbb{S}}$-algebra structure is the structure map $t_{\mathbb{S}\emptyset} : \Sigma_{\mathbb{S}} T_{\mathbb{S}}\emptyset \to T_{\mathbb{S}}\emptyset$ of the free $\Sigma_{\mathbb{S}}$ algebra;
- the labelled transition relation $\longrightarrow \subseteq X \times \mathrm{Lab} \times X$ is the smallest relation that satisfies every axiom of the theory.

6.3.2 GSOS conditions

We say that a rule structure $\mathtt{R}$ over variables $\mathtt{X}$ is in the *Positive GSOS rule format* if it satisfies GSOS$^+$-1–5 in Figure 6.4.

Discussion. The class of rules considered here differs from that introduced by Bloom, Istrail, and Meyer [1995, Defn. 4.3.2] in two ways: we have removed the image-finiteness condition, and we do not consider negative premises.

The image-finiteness condition can be reintroduced by considering a finite powerset functor; indeed, this is the approach taken by Turi and Plotkin [1997].

A positive version of the GSOS format has previously been studied by Groote and Vaandrager [1992, Sec. 9]. It is beyond the scope of this thesis to discuss the merits and demerits of negative premises in detail, but we can summarise some of the arguments.

- Against negative premises: the meaning (*i.e.*, the intended model) of rules with negative premises is a matter of debate [see *e.g.* Aceto *et al.*, 2001, Sec. 3].

Of a rule R over variables X, where

$$\mathtt{R} = \frac{\text{Prems}}{\underline{\mathtt{op}}(\underline{\mathtt{x}}_1, \ldots, \underline{\mathtt{x}}_{\mathrm{ar}(\mathtt{o})}) \xrightarrow{l} \mathtt{tar}}$$

we require:

GSOS$^+$-1. All variables appear either in the conclusion source or in the targets of the premises.

$$\forall \mathtt{x} \in \mathtt{X}. \begin{pmatrix} \exists j \in [1, \mathrm{ar}(\underline{\mathtt{op}})]. \ \underline{\mathtt{x}}_j = \mathtt{x} \\ \vee \ \exists l \in \mathrm{Lab}, \ \mathtt{y} \in \mathtt{X}. \ (\mathtt{x}, l, \mathtt{y}) \in \mathrm{Prems} \end{pmatrix}.$$

GSOS$^+$-2. The source of each premise appears in the conclusion source.

$$\forall (\mathtt{x}, l, \mathtt{y}) \in \mathrm{Prems}. \ \exists j \in [1, \mathrm{ar}(\underline{\mathtt{op}})]. \ \mathtt{x} = \underline{\mathtt{x}}_j.$$

GSOS$^+$-3. The target of any premise does not appear as the target of any other premise.

$$\forall (\mathtt{x}, l, \mathtt{y}), (\mathtt{x}', l', \mathtt{y}') \in \mathrm{Prems}. \ \mathtt{y} = \mathtt{y}' \implies \mathtt{x} = \mathtt{x}' \wedge l = l'.$$

GSOS$^+$-4. The target of any premise does not appear in the conclusion source.

$$\forall (\mathtt{x}, l, \mathtt{y}) \in \mathrm{Prems}, \ j \in [1, \mathrm{ar}(\underline{\mathtt{op}})]. \ \mathtt{y} \neq \underline{\mathtt{x}}_j.$$

GSOS$^+$-5. Variables in the conclusion source are distinct.

$$\forall j, j' \in [1, \mathrm{ar}(\underline{\mathtt{op}})]. \ \underline{\mathtt{x}}_j = \underline{\mathtt{x}}_{j'} \implies j = j'$$

Figure 6.4: The Positive GSOS format

- One might argue that if inaction is not observable then it cannot be used in system *definition*.
- In favour of negative premises, though, they do provide a convenient means of system *specification* for properties such as deadlock detection.

We will address some model-theoretic aspects of Positive GSOS in Section 9.3.4.

6.3.3 Natural transformations from single rules

In this subsection we fix a rule structure R over variables X, with premise set Prems, and conclusion $(\mathtt{src}, \underline{l}, \mathtt{tar})$. So we must have an operator $\underline{\mathtt{op}} \in \mathrm{Op}_{\mathbb{S}}$ and, for each $j \in [1, \mathrm{ar}(\underline{\mathtt{op}})]$ an element $\underline{\mathtt{x}}_j \in X$ such that

$$\mathtt{src} = \underline{\mathtt{op}}\left(\left(\underline{\mathtt{x}}_j\right)_{j \in [1, \mathrm{ar}(\underline{\mathtt{op}})]}\right) \quad .$$

(We underline variables that appear in the conclusion source to distinguish them from other variables.) We will show how R induces a family of maps

$$\left\{ [\![\mathtt{R}]\!]_X : \Sigma_{\mathbb{S}}(X \times BX) \to BT_{\mathbb{S}}X \right\}_{X \in \mathbf{Set}}$$

which will be natural if R is in the GSOS format.

Behaviour. Here, the endofunctor B on **Set** that we have in mind is the endofunctor

$$B = B_{\mathrm{lts}} = \mathscr{P}(\mathrm{Lab} \times -)$$

as discussed in Section 2.1. We will also make use of the endofunctor L on **Set** given by

$$L = \mathrm{Lab} \times (-) \quad .$$

This endofunctor describes labelled transition systems where each state can and must perform exactly one transition.

Valuations. A *valuation* of the variables of the rule $\mathtt{R}$ is a set X together with a function $\mathscr{V} : \mathtt{X} \to X$ between sets.

Instantiating premises. A valuation $\mathscr{V} : \mathtt{X} \to X$ determines, for each $j \in \mathrm{ar}(\underline{\mathtt{op}})$, a subset of LX that we denote $\mathscr{V}(\mathrm{Prems}_j)$. It describes the of "requirements on parameter j", and is given by

$$\mathscr{V}(\mathrm{Prems}_j) = \left\{ (l, y) \in LX \;\middle|\; \exists \mathtt{y} \in \mathtt{X}.\ \left(\underline{\mathtt{x}}_j, l, \mathtt{y}\right) \in \mathrm{Prems} \text{ and } \mathscr{V}(\mathtt{y}) = y \right\} \quad .$$

Instantiations of rules. We say that a valuation $\mathscr{V} : \mathtt{X} \to X$ is an *instantiation* of $\mathtt{R}$ into an element $s \in \Sigma_{\mathbb{S}}(X \times BX)$, if there is a family $(b_j \in BX)_{j \in [1, \mathrm{ar}(\underline{\mathtt{op}})]}$ such that

$$s = \underline{\mathtt{op}} \left(\left(\mathscr{V}(\underline{\mathtt{x}}_j), b_j \right)_{j \in [1, \mathrm{ar}(\underline{\mathtt{op}})]} \right)$$

and, for each $j \in [1, \mathrm{ar}(\underline{\mathtt{op}})]$, we have $\mathscr{V}(\mathrm{Prems}_j) \subseteq b_j$.

Archetypal results. For every valuation, we have a result given by

$$\left(\underline{l}, T_{\mathbb{S}} \mathscr{V}(\mathtt{tar}) \right) \in LT_{\mathbb{S}}X \quad .$$

Here, we are using the function $T_{\mathbb{S}}\mathscr{V} : T_{\mathbb{S}}\mathtt{X} \to T_{\mathbb{S}}X$.

Induced family of maps. A rule induces a family of maps

$$\left\{ [\![\mathtt{R}]\!]_X : \Sigma_{\mathbb{S}}(X \times BX) \to BT_{\mathbb{S}}X \right\}_{X \in \mathbf{Set}}$$

given by

$$[\![\mathtt{R}]\!]_X(s) = \left\{ (\underline{l}, T_{\mathbb{S}}\mathscr{V}(\mathtt{tar})) \;\middle|\; \mathscr{V} \text{ is an instantiation of } \mathtt{R} \text{ into } s \right\} \quad . \tag{6.3.3}$$

Thus $[\![\mathtt{R}]\!]_X$ can be thought of as taking an expression in which each element is equipped with a behaviour, and returning a set of all possible resumptions.

Naturality of the induced family of maps.

Theorem 6.3.4. *If* $\mathtt{R}$ *is in the GSOS format then the family* $\left\{ [\![\mathtt{R}]\!]_X \right\}_{X \in \mathbf{Set}}$ *is natural in* X.

Proof. Consider a function $f : X \to Y$; we must show that for any

$$s = \mathsf{op}\left((x_1, b_1), \ldots, (x_{\mathrm{ar}(\mathsf{op})}, b_{\mathrm{ar}(\mathsf{op})}) \right) \quad \in \Sigma(X \times BX)$$

we have

$$BT_{\mathbb{S}}f\Big(\llbracket \mathtt{R}\rrbracket_X\big(\ \mathsf{op}((x_1,b_1),\ldots,(x_{\mathrm{ar}(\mathsf{op})},b_{\mathrm{ar}(\mathsf{op})}))\ \big)\Big) = \llbracket \mathtt{R}\rrbracket_Y\big(\ \mathsf{op}((f(x_1),Bf(b_1)),\ldots,(f(x_{\mathrm{ar}(\mathsf{op})}),Bf(b_{\mathrm{ar}(\mathsf{op})})))\ \big)\ . \quad (6.3.5)$$

It is easy to show that LHS $\subseteq$ RHS in (6.3.5) — the GSOS conditions are not needed for this. Indeed, suppose that $(l,t)\in$ LHS. Then $l=\underline{l}$ and we must have an instantiation $\mathscr{V}$ of $\mathtt{R}$ into s such that $T_{\mathbb{S}}\mathscr{V}(\mathtt{tar})=t$. It is straightforward to see that the valuation $(f\circ\mathscr{V}):\mathtt{X}\to Y$ is an instantiation of $\mathtt{R}$ into $\Sigma_{\mathbb{S}}(f\times Bf)(s)$ for which $T_{\mathbb{S}}(f\circ\mathscr{V})(\mathtt{tar})=t$. Thus $(\underline{l},t)\in$ RHS; and so LHS $\subseteq$ RHS in (6.3.5).

To show that RHS $\subseteq$ LHS in (6.3.5) is more involved, and does use the GSOS conditions. Suppose that $(l,t)\in$ RHS. Then $\mathsf{op}=\underline{\mathsf{op}}$, and $l=\underline{l}$, and we must have an instantiation $\mathscr{V}':\mathtt{X}\to Y$ of $\mathtt{R}$ into $\Sigma_{\mathbb{S}}(f\times Bf)(s)$ for which $T_{\mathbb{S}}\mathscr{V}'(\mathtt{tar})=t$.

We will exhibit a valuation $\mathscr{V}:\mathtt{X}\to X$ satisfying

(i) $\forall j\in[1,\mathrm{ar}(\mathsf{op})].\ \mathscr{V}(\underline{\mathtt{x}}_j)=x_j$

(ii) $\forall l\in\mathrm{Lab},\ \mathtt{y}\in\mathtt{X},\ j\in[1,\mathrm{ar}(\mathsf{op})].$
$(\underline{\mathtt{x}}_j,l,\mathtt{y})\in\mathrm{Prems}\implies(l,\mathscr{V}(\mathtt{y}))\in b_j\ \wedge\ \mathscr{V}'(\mathtt{y})=f(\mathscr{V}(\mathtt{y}))\ .$

It follows immediately from these properties that $\mathscr{V}$ is an instantiation of $\mathtt{R}$ into s. We will later argue that it also follows that $(f\circ\mathscr{V})=\mathscr{V}'$, and hence that $(\underline{l},t)\in$ LHS.

To help us define such a valuation $\mathscr{V}$, we consider an equivalent formulation of the GSOS conditions. Indeed, the GSOS conditions ensure that the function

$$[1,\mathrm{ar}(\underline{\mathsf{op}})]+\coprod_{j\in[1,\mathrm{ar}(\underline{\mathsf{op}})]}\Big\{(\mathtt{x},l,\mathtt{y})\in\mathrm{Prems}\ \Big|\ \mathtt{x}=\underline{\mathtt{x}}_j\Big\}\longrightarrow\mathtt{X} \quad (6.3.6)$$

$$\mathsf{inl}(j)\longmapsto\underline{\mathtt{x}}_j$$

$$\mathsf{inr}(\mathsf{inj}_j(\mathtt{x},l,\mathtt{y}))\longmapsto\mathtt{y}$$

is a bijection. Surjectivity corresponds to Conditions GSOS$^+$-1,2. Injectivity for the upper map corresponds to Condition GSOS$^+$-5, and injectivity for the lower map corresponds to Condition GSOS$^+$-3. Injectivity for the sum of the two maps corresponds to Condition GSOS$^+$-4.

There may be many valuations that satisfy conditions (i) and (ii) above. We explain precisely why such a valuation can be found. (Note that we do use the axiom of choice for this.)

For each $l\in\mathrm{Lab}$ and $j\in[1,\mathrm{ar}(\underline{\mathsf{op}})]$, we define sets $(l^*\mathrm{Prems}_j)\subseteq\mathtt{X}$ and $(l^*b_j)\subseteq X$ by

$$l^*\mathrm{Prems}_j=\Big\{\mathtt{y}\in\mathtt{X}\ \Big|\ (\underline{\mathtt{x}}_j,l,\mathtt{y})\in\mathrm{Prems}\Big\}\qquad l^*b_j=\Big\{y\in X\ \Big|\ (l,y)\in b_j\Big\}\quad .$$

(The symbols $(l^*\mathrm{Prems}_j)$ and (l^*b_j) are only notation, but are chosen because (l^*b_j) is a pullback of the cospan $(b_j\rightarrowtail\mathrm{Lab}\times X\overset{(l,\mathsf{id}_X)}{\longleftarrow}X)$.)

We observe that because $\mathscr{V}'$ is an instantiation of $\mathtt{R}$ into $\Sigma_{\mathbb{S}}(f\times Bf)(s)$, we have

$$\mathscr{V}'(l^*\mathrm{Prems}_j)\subseteq f(l^*b_j)\quad .$$

Now, for the subset $X_{l,j}\subseteq X$ defined by by

$$X_{l,j}=\Big\{x\in l^*b_j\ \Big|\ \exists\mathtt{y}\in l^*\mathrm{Prems}_j.\ f(y)=\mathscr{V}'(\mathtt{x})\Big\}$$

we have that $f(X_{l,j}) = \mathcal{V}'(l^*\mathrm{Prems}_j)$; in other words we have a surjection

$$f|_{X_{l,j}} : X_{l,j} \twoheadrightarrow \mathcal{V}'(l^*\mathrm{Prems}_j) \quad .$$

We pick a section of this surjection,

$$m_{l,j} : \mathcal{V}'(l^*\mathrm{Prems}_j) \rightarrowtail X_{l,j} \quad .$$

We are now in a position to define our valuation $\mathcal{V}$. We do this using the bijection of (6.3.6):

- for $j \in [1, \mathrm{ar}(\underline{\mathtt{op}})]$, we let $\mathcal{V}(\underline{\mathtt{x}}_j) = x$;
- for $(\underline{\mathtt{x}}_j, l, \mathtt{y}) \in \mathrm{Prems}$, we let $\mathcal{V}(\mathtt{y}) = m_{l,j}(\mathcal{V}'(\mathtt{y}))$.

This valuation satisfies conditions (i) and (ii) above.

Finally, we explain that *any* valuation $\mathcal{V}$ satisfying conditions (i) and (ii) must have the property $f \circ \mathcal{V} = \mathcal{V}'$. To do this, we again make use of the bijection in (6.3.6). For variables $\mathtt{y} \in \mathtt{X}$ that appear as targets of premises, condition (ii) insists that $f(\mathcal{V}(\mathtt{y})) = \mathcal{V}'(\mathtt{y})$. For each $j \in [1, \mathrm{ar}(\underline{\mathtt{op}})]$, we know that $\mathcal{V}'(\underline{\mathtt{x}}_j) = f(x_j)$ because $\mathcal{V}'$ is an instantiation of $\mathtt{R}$ into $\Sigma_{\mathbb{S}}(f \times Bf)(s)$. Thus we have that $f(\mathcal{V}(\underline{\mathtt{x}}_j)) = \mathcal{V}'(\underline{\mathtt{x}}_j)$. □

6.3.4 Interpretation of multiple rules

The collection of natural transformations

$$\Sigma_{\mathbb{S}}((-) \times B(-)) \to BT_{\mathbb{S}}(-)$$

inherits a complete join semi-lattice structure from the powerset functor $\mathscr{P}$. We use this to give meaning to a class $\mathscr{R}$ of rule structures in the GSOS format: we define a natural transformation $[\![\mathscr{R}]\!] : \Sigma_{\mathbb{S}}((-) \times B(-)) \to BT_{\mathbb{S}}(-)$ by

$$[\![\mathscr{R}]\!] = \bigvee_{\mathtt{R} \in \mathscr{R}} [\![\mathtt{R}]\!] \quad .$$

So for any set X we have a function $[\![\mathscr{R}]\!]_X : \Sigma_{\mathbb{S}}(X \times BX) \to BT_{\mathbb{S}}X$ mapping an element $s \in \Sigma_{\mathbb{S}}(X \times BX)$ to the set

$$[\![\mathscr{R}]\!]_X(s) = \bigcup_{\mathtt{R} \in \mathscr{R}} ([\![\mathtt{R}]\!]_X(s)) \quad .$$

6.3.5 Relating our semantics of rules with the usual one

We have described how any class $\mathscr{R}$ of positive GSOS rule structures gives rise to a natural transformation $[\![\mathscr{R}]\!] : \Sigma_{\mathbb{S}}((-) \times B(-)) \to BT_{\mathbb{S}}(-)$. Using the techniques of Section 6.2.3, then, we arrive at a monad $\mathbf{T}_{\mathbb{S}[\![\mathscr{R}]\!]}$ on the category of B-coalgebras. The initial $\mathbf{T}_{\mathbb{S}[\![\mathscr{R}]\!]}$-algebra can be thought of as an $\mathscr{R}$-structure; its carrier is a B-coalgebra, indeed it is the set $T_{\mathbb{S}}\emptyset$ equipped with a labelled transition relation. We now explain why it is in fact the intended model.

Instantiations really are instantiations. For now, we consider single rules. We fix an $\mathtt{R}$-structure $(X, \alpha, \longrightarrow)$; that is, an $\mathscr{R}$-structure for the singleton class $\mathscr{R} = \{\mathtt{R}\}$. Our comments here only require Condition GSOS$^+$-2 to hold of the rule structure $\mathtt{R}$.

Note that a valuation in the sense used in Section 6.3.3 is the same thing as a valuation of the logic formula $\Phi_{\mathtt{R}}$ that we associated to the rule in (6.3.2).

Suppose that we have, for each $j \in [1, \mathrm{ar}(\underline{\mathtt{op}})]$, an element $x_j \in X$. We have the following result for every valuation $\mathcal{V}$ into X:

The valuation $\mathcal{V}$ is an instantiation of $\mathtt{R}$ into

$$\underline{\mathtt{op}}\left(\left(\left(x_j, \left\{(l,y) \;\middle|\; x_j \xrightarrow{l} y\right\}\right)\right)_{j\in[1,\mathrm{ar}(\underline{\mathtt{op}})]}\right)$$

if and only if the formula

$$\bigwedge_{(\mathtt{x},l,\mathtt{y})\in \mathrm{Prems}} \left(\mathtt{x} \xrightarrow{l} \mathtt{y}\right)$$

is true for the valuation $\mathcal{V}$ in the $\mathtt{R}$-structure $(X, \alpha, \longrightarrow)$.

The abstract rule specifies allowed transitions. We now consider a class $\mathscr{R}$ of rules structures, all of which we assume satisfy Condition GSOS$^+$-2.

The family of functions $[\![\mathscr{R}]\!]_X : \Sigma_{\mathbb{S}}(X \times BX) \to BT_{\mathbb{S}}X$ can be understood as follows. Consider a Lab-labelled transition system $(X, \longrightarrow)$, For any operator $\mathsf{op} \in \mathsf{Op}_{\mathbb{S}}$, and $x_j \in X$ for each $j \in [1, \mathrm{ar}(\mathsf{op})]$, and for any $(l,t) \in LT_{\mathbb{S}}X$, we have:

$$(l,t) \in [\![\mathscr{R}]\!]_X \left(\mathsf{op}\left(\left(\left(x_j, \left\{(l,y) \;\middle|\; x_j \xrightarrow{l} y\right\}\right)\right)_{j\in[1,\mathrm{ar}(\underline{\mathtt{op}})]}\right)\right)$$

if and only if for every $\Sigma_{\mathbb{S}}$-algebra structure extending $(X, \longrightarrow)$ to a $\mathscr{R}$-model allows the transition $\alpha\left(\mathsf{op}\left(\left(x_j\right)_{j\in[1,\mathrm{ar}(\underline{\mathtt{op}})]}\right)\right) \xrightarrow{l} \alpha^\sharp(t)$.

Here we write $\alpha^\sharp$ for the extension of the $\Sigma_{\mathbb{S}}$-algebra structure $\alpha : \Sigma_{\mathbb{S}}X \to X$ to a $\mathbf{T}_{\mathbb{S}}$-algebra structure $\alpha^\sharp : T_{\mathbb{S}}X \to X$.

Thus the intended $\mathscr{R}$-model can be found by using $[\![\mathscr{R}]\!]$ to build up a transition relation structure on the set $T_{\mathbb{S}}\emptyset$ of ground terms. This is precisely the role of the parameterised recursion in Section 6.2.3. So we have:

Theorem 6.3.7. *For a class $\mathscr{R}$ of rule structures in the positive GSOS format, the initial $\mathbf{T}_{\mathbb{S}[\![\mathscr{R}]\!]}$-algebra is the intended $\mathscr{R}$-model.* □

From Corollary 6.2.3, we conclude the following result.

Theorem 6.3.8. *In the intended model of a set $\mathscr{R}$ of rule structures in the positive GSOS format, bisimilarity is a congruence.* □

6.A Appendix to Chapter 6: Proof of Theorem 6.1.5

We prove Theorem 6.1.5. We begin by recalling the statement of the theorem from page 136.

Theorem 6.1.5. *Let Σ be an endofunctor on a category $\mathscr{C}$, and let $\mathbf{T}_\Sigma$ be a monad on $\mathscr{C}$. The following are equivalent.*

1. *There is a natural family of isomorphisms of categories*

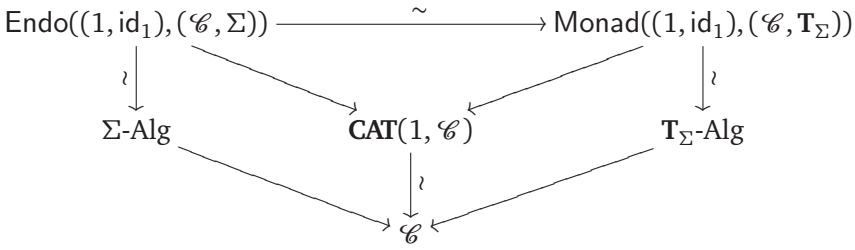
$$\Big\{\mathsf{Endo}(U(\mathscr{D},T),(\mathscr{C},\Sigma)) \xrightarrow{\sim} \mathsf{Monad}((\mathscr{D},\mathbf{T}),(\mathscr{C},\mathbf{T}_\Sigma))\Big\}_{(\mathscr{D},\mathbf{T})\in\mathsf{Monad}}$$

   (where we write U for the forgetful 2-functor Monad $\to$ Endo*) for which the following diagram commutes, where the vertical arrows are the evident forgetful functors.*

$$\begin{array}{ccc} \mathsf{Endo}((\mathscr{D},T),(\mathscr{C},\Sigma)) & \xrightarrow{\quad\sim\quad} & \mathsf{Monad}((\mathscr{D},\mathbf{T}),(\mathscr{C},\mathbf{T}_\Sigma)) \\ & \searrow \quad \swarrow & \\ & \mathbf{CAT}(\mathscr{D},\mathscr{C}) & \end{array}$$

2. *The forgetful functor Σ-Alg $\to \mathscr{C}$ has a left adjoint and the resulting monad on $\mathscr{C}$ is $\mathbf{T}_\Sigma$.*

Proof. From item (1) to item (2): observe the following diagram in **CAT**, where the unlabelled arrows are forgetful functors.



By definition, the forgetful functor $\mathbf{T}_\Sigma$-Alg $\to \mathscr{C}$ is monadic, and hence so is Σ-Alg $\to \mathscr{C}$.

To establish item (1) from item (2) is more involved. We suppose that the forgetful functor $U_\Sigma : \Sigma\text{-Alg} \to \mathscr{C}$ has a left adjoint $F_\Sigma : \mathscr{C} \to \Sigma\text{-Alg}$, and that the resulting monad on $\mathscr{C}$ is $\mathbf{T}_\Sigma$.

Let $\mathscr{D}$ be any category. We write $(\Sigma-)$ for the endofunctor on the functor category $[\mathscr{D},\mathscr{C}]$ given by postcomposition with Σ; and we write $(\mathbf{T}_\Sigma-)$ for the monad on the functor category $[\mathscr{D},\mathscr{C}]$, whose underlying endofunctor is postcomposition with T_Σ. We will now show that the forgetful functor $U_{(\Sigma-)} : (\Sigma-)\text{-Alg} \to [\mathscr{D},\mathscr{C}]$ has a left adjoint, and the resulting monad is $(\mathbf{T}_\Sigma-)$.

First, consider the hom 2-functor $[\mathscr{D},-]$ on **CAT**. By applying this to the adjunction $F_\Sigma \dashv U_\Sigma : \Sigma\text{-Alg} \to \mathscr{C}$ we arrive at the following adjunction in **CAT**

$$[\mathscr{D},\Sigma\text{-Alg}] \underset{[\mathscr{D},F_\Sigma]}{\overset{[\mathscr{D},U_\Sigma]}{\underset{\longleftarrow}{\overset{\longrightarrow}{\top}}}} [\mathscr{D},\mathscr{C}]$$

for which the resulting monad has underlying endofunctor $[\mathscr{D},U_\Sigma F_\Sigma] = [\mathscr{D},T_\Sigma]$. Now, to conclude that the forgetful functor $U_{(\Sigma-)} : (\Sigma-)\text{-Alg} \to [\mathscr{D},\mathscr{C}]$ has a left adjoint it remains for us to exhibit an isomorphism of categories $i : (\Sigma-)\text{-Alg} \xrightarrow{\sim} [\mathscr{D},\Sigma\text{-Alg}]$ making the following diagram commute.

$$\begin{array}{ccc} (\Sigma-)\text{-Alg} & \xrightarrow{\quad i \quad} & [\mathscr{D},\Sigma\text{-Alg}] \\ & {\scriptstyle U_{(\Sigma-)}}\searrow \quad \swarrow{\scriptstyle [\mathscr{D},U_\Sigma]} & \\ & [\mathscr{D},\mathscr{C}] & \end{array}$$

Such an isomorphism i is found as follows.

For each $(\Sigma-)$-algebra, which is a functor $G : \mathscr{D} \to \mathscr{C}$ together with a natural transformation $\gamma : \Sigma G \to G$, we have a functor $i(G,\gamma) : \mathscr{D} \to \Sigma\text{-Alg}$ which acts as follows:

- for any object D of $\mathscr{D}$, we let $i(G,\gamma)(D)$ be the Σ-algebra given by $(G(D), \gamma_D)$;
- for any morphism $f : D \to D'$ in $\mathscr{D}'$, we let $i(G,\gamma)(f)$ be the Σ-algebra homomorphism $i(G,\gamma)(D) \to i(G,\gamma)(D')$ given by $Gf : GD \to GD'$; this is a homomorphism because γ is natural.

Meanwhile, for every $(\Sigma-)$-algebra homomorphism from (G,γ) to (G',γ'), which is a natural transformation $\alpha : G \to G'$ making the following diagram commute,

$$\begin{array}{ccc} \Sigma G & \xrightarrow{\Sigma\alpha} & \Sigma G' \\ {\scriptstyle\gamma}\downarrow & & \downarrow{\scriptstyle\gamma'} \\ G & \xrightarrow[\alpha]{} & G' \end{array} \tag{6.A.1}$$

we have a natural transformation $i\alpha : i(G,\gamma) \to i(G',\gamma')$ between functors $\mathscr{D} \to \Sigma\text{-Alg}$ given as follows. For each $D \in \mathscr{D}$ we have a Σ-algebra homomorphism given by $(i\alpha)_D = \alpha_D$; this is a homomorphism because diagram 6.A.1 commutes.

It is straightforward to find an inverse for i. Thus we can conclude that the forgetful functor $U_{(\Sigma-)} : (\Sigma-)\text{-Alg} \to [\mathscr{D},\mathscr{C}]$ has a left adjoint, and that the resulting monad on $[\mathscr{D},\mathscr{C}]$ is $(\mathbf{T}_\Sigma-)$. So for any functor $F : \mathscr{D} \to \mathscr{C}$, the left adjoint provides a natural transformation $t_\Sigma F : \Sigma T_\Sigma F \to T_\Sigma F$ that is universal, in the sense that for any other $(\Sigma-)$-algebra, (G,γ), and any natural transformation $\alpha : F \to G$, there is a unique natural transformation $(\alpha,\gamma)^\sharp : TF \to G$ making the following diagram commute.

$$\begin{array}{ccc} \Sigma T_\Sigma F & \xrightarrow{\Sigma(\alpha,\gamma)^\sharp} & \Sigma G \\ {\scriptstyle t_\Sigma F}\downarrow & & \downarrow{\scriptstyle\gamma} \\ T_\Sigma F & \xrightarrow{(\alpha,\gamma)^\sharp} & G \\ {\scriptstyle\eta_\Sigma F}\uparrow & \nearrow_{\alpha} & \\ F & & \end{array}$$

The functoriality of the left adjoint to $U_{(\Sigma-)} : (\Sigma-)\text{-Alg} \to [\mathscr{D},\mathscr{C}]$ ensures that the family $\{\Sigma T_\Sigma F \to T_\Sigma F\}_{F:\mathscr{D}\to\mathscr{C}}$ is natural in F.

We are now in a position to define a isomorphism

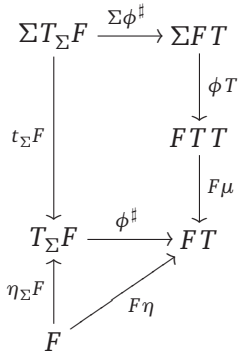
$$\mathsf{Endo}((\mathscr{D},T),(\mathscr{C},\Sigma)) \cong \mathsf{Monad}((\mathscr{D},\mathbf{T}),(\mathscr{C},\mathbf{T}_\Sigma))$$

as required by item (1). To this end, we define a functor

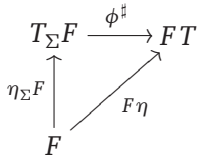
$$j : \mathsf{Endo}((\mathscr{D},T),(\mathscr{C},\Sigma)) \to \mathsf{Monad}((\mathscr{D},\mathbf{T}),(\mathscr{C},\mathbf{T}_\Sigma))$$

as follows. For any morphism $(F,\phi) : (\mathscr{D},T) \to (\mathscr{C},\Sigma)$ in Endo, we define a morphism $j(F,\phi) : (\mathscr{D},\mathbf{T}) \to (\mathscr{C},\mathbf{T}_\Sigma)$ in Monad by letting $j(F,\phi) = (F,\phi^\sharp)$, where we write $\phi^\sharp$ for the unique

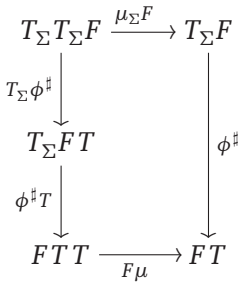
natural transformation $T_\Sigma F \to FT$ making the following diagram commute.



We have to check that $j(F,\phi) : (\mathscr{D}, \mathbf{T}) \to (\mathscr{C}, \mathbf{T}_\Sigma)$ is a monad morphism, that is, that it respects the units and multiplications. For the units, we must show that the following diagram of natural transformations commutes.

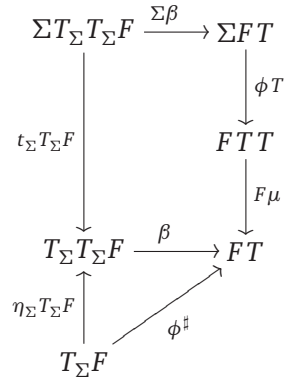


This follows immediately from the definition of $\phi^\sharp$. For the multiplications, we must show that the following diagram of natural transformations commutes.

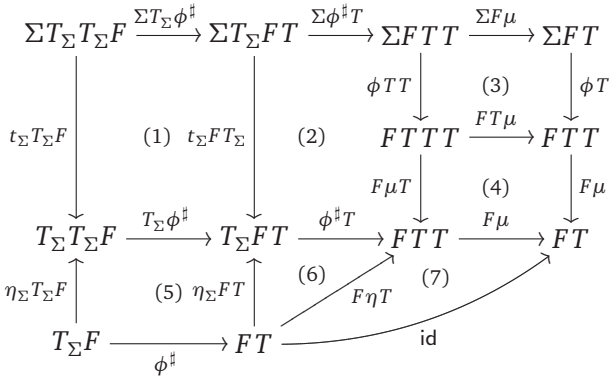


(6.A.2)

To show this, we first note that there is a unique natural transformation $\beta : T_\Sigma T_\Sigma F \to FT$ such that the following diagram commutes

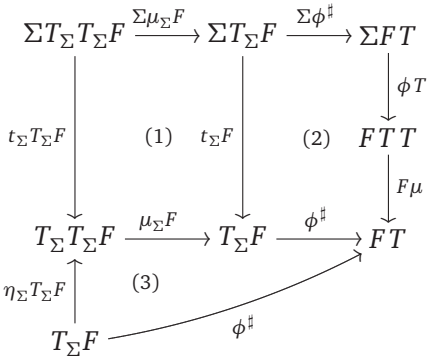


and then we explain that β could be either side of diagram 6.A.2. To show $\beta = F\mu_\Sigma \circ \phi^\sharp T \circ T_\Sigma \phi^\sharp$, we use the following decomposition.



Using: (1) nat. of t_Σ; (2) defn. of $\phi^\sharp$; (3) nat. of ϕ; (4) mult. law for $\mathbf{T}$; (5) nat. of η_Σ; (6) defn. of $\phi^\sharp$; (7) unit law for $\mathbf{T}$.

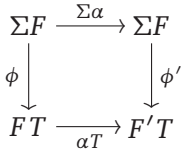
For the case $\beta = \phi^\sharp \circ \mu_\Sigma F$, we use the following decomposition.



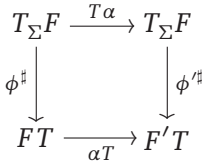
Using: (1) $\mu_\Sigma F$ is a Σ-algebra homomorphism, since it can be defined in terms of the counit; (2) defn. of $\phi^\sharp$; (3) unit law for $\mathbf{T}_\Sigma$.

Thus the action of $j : \mathsf{Endo}((\mathscr{D}, T), (\mathscr{C}, \Sigma)) \to \mathsf{Monad}((\mathscr{D}, \mathbf{T}), (\mathscr{C}, \mathbf{T}_\Sigma))$ on objects is described.

A morphism $(F, \phi) \to (F', \phi')$ in the category $\mathsf{Endo}((\mathscr{D}, T), (\mathscr{C}, \Sigma))$ is a natural transformation $\alpha : F \to F'$ making the following diagram commute.



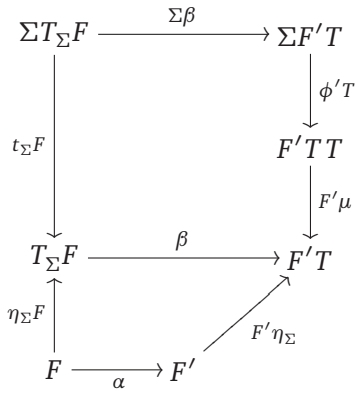
We let $j\alpha : j(F, \phi) \to j(F', \phi')$ be the same natural transformation, regarded as a 2-cell between monad morphisms. We must check that it is a valid 2-cell, that is, we must check that the following diagram commutes.



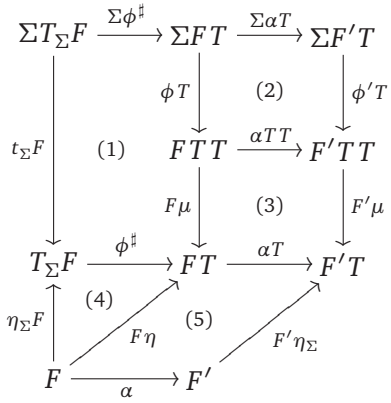
(6.A.3)

To show this, we first observe that there is exactly one natural transformation $\beta : T_\Sigma F \to F' T$ mak-

ing the following diagram commute.

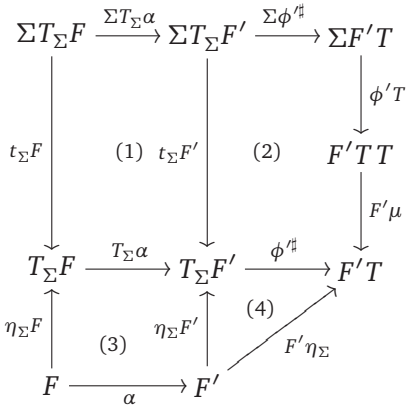


We now claim that this natural transformation, β, could be either side of diagram 6.A.3. For $\beta = \alpha T \circ \phi^\sharp$, consider the following decomposition.



Using: (1) defn. of $\phi^\sharp$; (2) since α is a 2-cell in Endo; (3) nat. of α; (4) defn. of $\phi^\sharp$; (5) nat. of α.

For $\beta = \phi'^\sharp \circ T_\Sigma \alpha$, consider the following decomposition.



Using: (1) nat. of t_Σ; (2) defn. of $\phi'^\sharp$; (3) nat. of η_Σ; (4) defn. of $\phi'^\sharp$.

In this way we define, for each monad $(\mathscr{D}, \mathbf{T})$, a functor

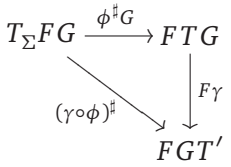
$$j_{(\mathscr{D},\mathbf{T})} : \mathsf{Endo}((\mathscr{D}, T), (\mathscr{C}, \Sigma)) \to \mathsf{Monad}((\mathscr{D}, \mathbf{T}), (\mathscr{C}, \mathbf{T}_\Sigma)) \quad .$$

We will show that this family of functors is natural in $(\mathscr{D}, T)$. We must show that for every monad morphism $(G, \gamma) : (\mathscr{D}', \mathbf{T}') \to (\mathscr{D}, \mathbf{T})$ and for every morphism of endofunctors $(F, \phi) : (\mathscr{D}, T) \to (\mathscr{C}, \Sigma)$

we have

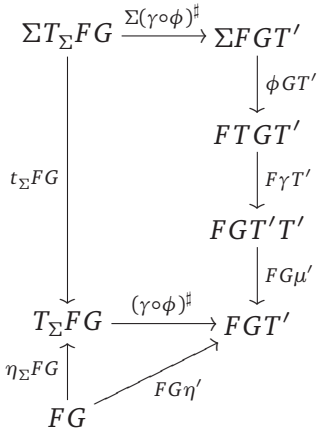
$$(j_{(\mathscr{D},\mathbf{T})}(F,\phi)) \circ (G,\gamma) \;=\; j_{(\mathscr{D}',\mathbf{T}')}((F,\phi)\circ(G,\gamma)) \quad .$$

By the definition of composition in Endo, it suffices for us to prove that the following diagram commutes in the category of functors $\mathscr{D}' \to \mathscr{C}$.

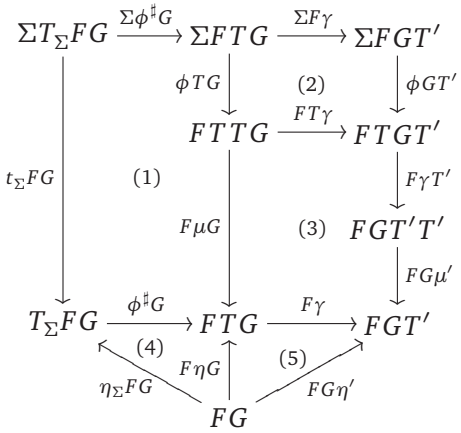


(6.A.4)

Here, $(\gamma \circ \phi)^\sharp$ is the unique morphism $T_\Sigma FG \to FGT'$ making the following diagram commute.



So, to show that diagram 6.A.4 commutes, we consider the following decomposition.



Using: (1) defn. of $\phi^\sharp$; (2) nat. of ϕ; (3) mult. law for γ; (4) defn. of $\phi^\sharp$; (5) unit law for γ.

It remains for us to show that j is an isomorphism. To define the inverse, we define, for each monad $(\mathscr{D},\mathbf{T})$, a functor

$$k_{(\mathscr{D},\mathbf{T})} : \mathsf{Monad}((\mathscr{D},\mathbf{T}),(\mathscr{C},\mathbf{T}_\Sigma)) \to \mathsf{Endo}((\mathscr{D},T),(\mathscr{C},\Sigma)) \quad .$$

This functor $k_{(\mathscr{D},\mathbf{T})}$ acts on the objects of $\mathsf{Monad}((\mathscr{D},\mathbf{T}),(\mathscr{C},\mathbf{T}_\Sigma))$ as follows. Given a monad morphism $(F,\phi) : (\mathscr{D},\mathbf{T}) \to (\mathscr{C},\mathbf{T}_\Sigma)$, we define a morphism of endofunctors $k_{(\mathscr{D},\mathbf{T})}(F,\phi) : (\mathscr{D},T) \to (\mathscr{C},\Sigma)$, by setting $k_{(\mathscr{D},\mathbf{T})}(F,\phi) = (F,\phi^\flat)$, where the natural transformation $\phi^\flat : \Sigma F \to TF$ is the composite

$$\Sigma F \xrightarrow{\Sigma\eta_\Sigma F} \Sigma T_\Sigma F \xrightarrow{t_\Sigma F} T_\Sigma F \xrightarrow{\phi} FT \quad .$$

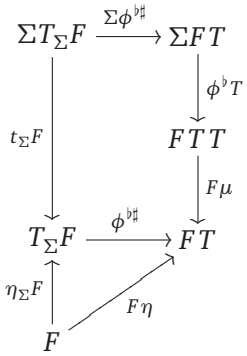
The functor $k_{(\mathscr{D},\mathbf{T})}$ acts as identity on 2-cells.

It is straightforward to show that the family of functors $k_{(\mathscr{D},\mathbf{T})}$ is natural in $(\mathscr{D},\mathbf{T})$.

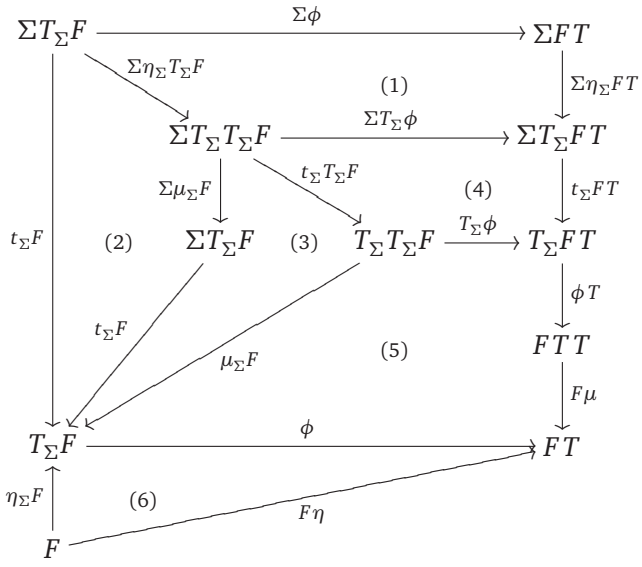
We now explain why $k_{(\mathscr{D},\mathbf{T})}$ is left and right inverse to $j_{(\mathscr{D},\mathbf{T})}$. For left inverse, it is sufficient to show that for any monad morphism $(F,\phi):(\mathscr{D},\mathbf{T})\to(\mathscr{C},\mathbf{T}_\Sigma)$ we have an equality of natural transformations

$$\phi^{\flat\sharp}=\phi \ :T_\Sigma F\to FT \quad . \tag{6.A.5}$$

Now, $\phi^{\flat\sharp}$ is defined to be the unique map $T_\Sigma F\to FT$ making the following diagram commute.



To conclude equation 6.A.5, we consider the following decomposition.



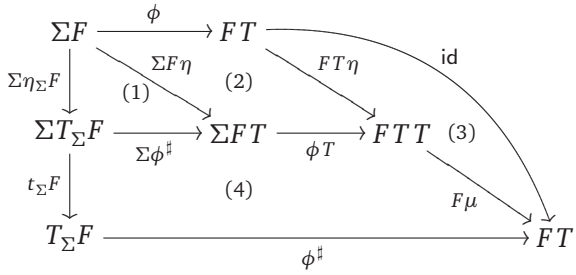
Using: (1) nat. of η_Σ; (2) unit law for $\mathbf{T}_\Sigma$; (3) since μ_Σ is a Σ-algebra homomorphism; (4) nat. of t_Σ; (5) mult. law for ϕ; (6) unit law for ϕ.

To show that $k_{(\mathscr{D},\mathbf{T})}$ is right inverse to $j_{(\mathscr{D},\mathbf{T})}$, it is sufficient to show that for any morphism of endofunctors $(F,\phi):(\mathscr{D},T)\to(\mathscr{C},\Sigma)$ we have an equality of natural transformations

$$\phi^{\sharp\flat}=\phi \ :\Sigma F\to FT \quad .$$

To see this, consider the following decomposition, in which the leftish outer leg is the definition

of $\phi^{\sharp\flat}$.



Using: (1) defn. of $\phi^\sharp$; (2) nat.of ϕ; (3) unit law for **T**; (4) defn. of $\phi^\sharp$.

Thus we have derived item (1) from item (2), and Theorem 6.1.5 is proved. □

Chapter 7

Nominal Sets for Syntax and Behaviour

The purposes of this chapter are threefold. Firstly, we address a problem with the approach to abstract syntax taken in Chapter 6: it is not immediately relevant for name-passing systems, which tend to involve name binding and α-equivalence. We address this problem by recalling the theory of nominal sets that has been championed by Gabbay and Pitts (and others) as an elegant framework for abstract syntax with binding.

Arbitrary, non-injective, substitution significantly enhances the model theory of name-passing systems, as was seen in Section 3.4. The second purpose of this chapter is to emphasise that arbitrary substitution is relevant for systems built of abstract syntax. To this end we introduce a theory of nominal substitutions to accommodate arbitrary substitution in the context of nominal sets.

It is convenient to work with syntax and behaviour within the same framework. The third purpose of this chapter is to exploit the equivalence between nominal sets and the Schanuel topos in order to study the models for name-passing of the first part of this thesis in the context of nominal sets. In doing this we arrive at a rather simple axiomatisation for a labelled transition system model of name-passing.

We begin in Section 7.1 by recalling some of the basic aspects of nominal sets. We include a sketch of a proof that the category **Nom** of nominal sets is equivalent to the Schanuel topos **Sh**(**I**) which was studied in Section 4.2. Using this equivalence, we translate an endofunctor for non-determinism from **Sh**(**I**) to **Nom**, so that it is possible to work with coalgebras for an endofunctor defined on **Nom**.

Having introduced nominal sets, we recall in Section 7.2 a variant of Pitts's nominal logic, which is essentially a first order fragment of the internal logic of the category of nominal sets. It is in this setting that we introduce, in Section 7.3, the theory of nominal substitutions, proving that the category of models of nominal substitutions is equivalent to the sheaf category **Sh**(**F**) that was studied in Section 4.3. This equivalence is important because it means that the model theory of Chapters 3 and 4 can be carried out in the setting of nominal sets, where syntax sits most naturally.

We investigate signatures for abstract syntax in Section 7.4. We introduce a restricted form of nominal logic signature, and illustrate this by showing that structures for such signatures in the category of nominal substitutions provide models of syntax up-to α-equivalence with arbitrary substitutions. On the other hand, structures for such signatures in the category of sets provide models of raw syntax, with no α-equivalence. The forgetful functor from nominal substitutions to the category of sets induces a way of instantiating raw syntax into abstract syntax.

The final section of this chapter, Section 7.5, is concerned with a nominal logic theory of labelled transition systems. We show that the models of our theory correspond to **I**-IL$_e$TSs over sheaves that satisfy Axioms **I**1–**I**6, as introduced in Section 3.3.

7.1 Nominal sets

In this section we recall some aspects of the nominal sets of Gabbay and Pitts. The reader will find further details and discussion elsewhere [*e.g.* Gabbay's thesis, 2001; Gabbay and Pitts, 2001, Sec. 3; Pitts, 2006, Sec. 3].

We begin, in Section 7.1.1, by recalling the basic definitions and properties of nominal sets. In Section 7.1.2 we briefly investigate connections between the category of nominal sets and a category of permutation actions, and the category of sets. We then turn, in Section 7.1.3, to recall various basic constructions of nominals sets — limits, colimits, the set of names, and the binding operator. In Section 7.1.4, we recall the equivalence between the category of nominal sets and the sheaf category $\mathbf{Sh}(\mathbf{I})$ considered in Section 4.2. We close in Section 7.1.5 by lifting the powerset endofunctor from $\mathbf{Sh}(\mathbf{I})$ to an endofunctor on **Nom**. by doing so we are able to consider the endofunctor for ground behaviour in the nominal setting.

7.1.1 Nominal sets

We recall the notion of nominal set, as an action of a permuation group subject to a finite-support condition. As a first example, we consider a nominal set of π-calculus terms. We recall some basic results about supports, in particular that every nominal set has a least support.

Actions of the symmetric group on $\mathcal{N}$. As in earlier chapters, we fix an infinite set $\mathcal{N}$ of names. We write $\mathsf{Sym}(\mathcal{N})$ for the group of permutations on $\mathcal{N}$. Recall that a $\mathsf{Sym}(\mathcal{N})$-*set* is a functor in the category $\mathbf{Set}^{\mathsf{Sym}(\mathcal{N})}$ (here, the group $\mathsf{Sym}(\mathcal{N})$ is regarded as a category with one object). More concretely, a $\mathsf{Sym}(\mathcal{N})$-set is a set X together with a function

$$\bullet_X : \mathsf{Sym}(\mathcal{N}) \times X \to X \quad .$$

This function is written infix and must be such that, for all $\sigma, \tau \in \mathsf{Sym}(\mathcal{N})$, $x \in X$,

$$\mathsf{id}_{\mathcal{N}} \bullet_X x = x \qquad \text{and} \qquad \tau \bullet_X (\sigma \bullet_X x) = (\tau \cdot \sigma) \bullet_X x \quad .$$

Morphisms between $\mathsf{Sym}(\mathcal{N})$-sets X and Y are *equivariant functions*; that is, functions $f : X \to Y$ that are such that for any $x \in X$ and any $\sigma \in \mathsf{Sym}(\mathcal{N})$, $f(\sigma \bullet_X x) = \sigma \bullet_Y (f(x))$. Composition and identities are found according to the underlying functions.

The permutations in $\mathsf{Sym}(\mathcal{N})$ that swap two names play an important role. We write $[a \leftrightarrow b]$ for the permutation of $\mathcal{N}$ that swaps a with b and does not affect any other names.

We will often abuse notation by writing X for a $\mathsf{Sym}(\mathcal{N})$-set $(X, \bullet_X)$.

Finite supports and nominal sets. A finite set $C \subseteq_{\mathrm{f}} \mathcal{N}$ of names is said to *support* an element $x \in X$ of a $\mathsf{Sym}(\mathcal{N})$-set if for any permutation $\sigma \in \mathsf{Sym}(\mathcal{N})$ we have

$$\sigma|_C = \mathsf{id}_C \implies \sigma \bullet x = x \quad .$$

A *nominal set* is a $\mathsf{Sym}(\mathcal{N})$-set X for which every element $x \in X$ has a (finite) support. We let **Nom** be the full subcategory of $\mathbf{Set}^{\mathsf{Sym}(\mathcal{N})}$ whose objects are nominal sets.

Example. As a first example, we give a nominal set of π-calculus terms. A grammar for a variant of the π-calculus was given in (3.1.1); recall that we work with terms up-to α-equivalence.

Thus we consider the set of π-calculus terms

$$X_\pi = \{t \mid t \text{ is a term of the } \pi\text{-calculus with names taken from } \mathcal{N}\} \tag{7.1.1a}$$

with an action given by

$$\sigma \bullet_{X_\pi} t \;=\; \text{the term } t \text{ renamed according to } \sigma \quad . \tag{7.1.1b}$$

For $t \in X_\pi$, a finite set $C \subseteq_{\mathrm{f}} \mathcal{N}$ supports t if and only if all the free names appearing in t are contained in C.

Nominal sets of π-calculus terms will be constructed in a more methodical way in Section 7.4.

(The same symbol, X_π, is used briefly here for the nominal set of π-calculus terms, and also in Section 3.4 for the presheaf over $\mathbf{F}$ of π-calculus terms. The reader should not confuse the two different entities.)

Basic results. We recall some basic results about nominal sets and equivariant functions.

Proposition 7.1.2.

1. *Let X be a $\mathsf{Sym}(\mathcal{N})$-set, and consider $x \in X$ together with $C, D \subseteq_{\mathrm{f}} \mathcal{N}$. If C supports x and $C \subseteq D$ then D supports x.*

2. *Let X be a $\mathsf{Sym}(\mathcal{N})$-set, and consider $x \in X$ together with $C, D \subseteq_{\mathrm{f}} \mathcal{N}$. If both C and D support x, then $(C \cap D)$ also supports x.*

3. *Let X be a $\mathsf{Sym}(\mathcal{N})$-set, and consider $x \in X$ together with $C \subseteq_{\mathrm{f}} \mathcal{N}$ and a permutation $\sigma \in \mathsf{Sym}(\mathcal{N})$. If C supports x then $\sigma(C)$ supports $\sigma \bullet_X x$.*

4. *Let $f : X \to Y$ be an equivariant function between nominal sets. If C supports $x \in X$, then C supports $f(x) \in Y$.*

Proof notes. Statements 1, 3 and 4 follow immediately from the definitions. For item 2, see *e.g.* the comments of Pitts [2006, discussion after Defn. 3.1]. □

Least support. It follows from Prop. 7.1.2(2) that every element x of a nominal set X has a *least support* which we will denote $\mathsf{supp}_X(x)$. (The subscript will often be omitted.)

For a variable $a \in \mathcal{N}$ and an element $x \in X$ of a $\mathcal{N}$-nominal set we write

$$a \# x \qquad \text{for} \qquad a \notin \mathsf{supp}_X(x) \quad .$$

The symbol $\#$ is pronounced "is fresh for". It follows from Prop. 7.1.2(4) that for any equivariant function $f : X \to Y$ between nominal sets,

$$\forall a \in \mathcal{N},\ x \in X.\ a \# x \implies a \# f(x) \quad .$$

For an element of our example nominal set X_π of π-calculus terms, the least support of a term is the set of its free names. It is often helpful to think of the supp function as an abstract form of the 'free variables' function that is used by many authors when dealing with syntax.

7.1.2 Relating Nom with $\mathbf{Set}^{\mathsf{Sym}(\mathcal{N})}$ and Set

We remark briefly how the category of nominal sets relates with the category of $\mathsf{Sym}(\mathcal{N})$-sets and with the category of sets. Our primary motivation for this is to explain the structure of limits and colimits of nominal sets.

An adjunction between Nom and $\mathbf{Set}^{\mathsf{Sym}(\mathcal{N})}$. The embedding

$$\mathbf{Nom} \hookrightarrow \mathbf{Set}^{\mathsf{Sym}(\mathcal{N})}$$

has a right adjoint $(-)_{\mathrm{fs}} : \mathbf{Set}^{\mathsf{Sym}(\mathcal{N})} \to \mathbf{Nom}$ which sends a $\mathsf{Sym}(\mathcal{N})$-set X to the nominal set with carrier

$$X_{\mathrm{fs}} = \left\{ x \in X \;\middle|\; x \text{ has finite support} \right\} \quad .$$

The embedding $\mathbf{Nom} \hookrightarrow \mathbf{Set}^{\mathsf{Sym}(\mathcal{N})}$ preserves finite limits, and so we have a geometric morphism $\mathbf{Set}^{\mathsf{Sym}(\mathcal{N})} \to \mathbf{Nom}$.

This geometric morphism can be seen to arise from a continuous map between topological groups [see *e.g.* Moerdijk, 1995, Sec. 2.3].

An adjunction between Nom and Set. The unique functor $1 \to \mathsf{Sym}(\mathcal{N})$ induces an essential geometric morphism $\mathbf{Set} \to \mathbf{Set}^{\mathsf{Sym}(\mathcal{N})}$ in the usual way; the inverse image is the forgetful functor $\mathbf{Set}^{\mathsf{Sym}(\mathcal{N})} \to \mathbf{Set}$ sending a $\mathsf{Sym}(\mathcal{N})$-set to its underlying set.

By considering the composite geometric morphism

$$\mathbf{Set} \longrightarrow \mathbf{Set}^{\mathsf{Sym}(\mathcal{N})} \longrightarrow \mathbf{Nom}$$

we see that the faithful forgetful functor $\mathbf{Nom} \to \mathbf{Set}$, that maps a nominal set to its underlying set, is the inverse image of a geometric morphism. Thus it preserves all colimits and finite limits.

7.1.3 Constructions on nominal sets

We recall some constructions in the category of nominal sets. Limits and colimits are determined according to the previous section, but we also recall the nominal set of names, and, crucially, the abstraction operator for nominal sets.

Limits and colimits. Above, we explained that the forgetful functor $\mathbf{Nom} \to \mathbf{Set}$ preserves finite limits and arbitrary colimits; this determines the underlying sets of limits (resp. colimits) of finite (resp. arbitrary) diagrams in **Nom**. Most relevant for the present work are finite products and arbitrary coproducts.

Because the embedding $\mathbf{Nom} \to \mathbf{Set}^{\mathsf{Sym}(\mathcal{N})}$ preserves finite limits and all colimits, the group actions are componentwise. Specifically, given $\mathsf{Sym}(\mathcal{N})$-sets X, Y, the product of the underlying sets is equipped with action given by

$$\sigma \bullet_{X \times Y} (x, y) = (\sigma \bullet_X x, \sigma \bullet_Y y)$$

for any $(x, y) \in X \times Y$. A finite set $C \subseteq_{\mathrm{f}} \mathcal{N}$ supports both x and y if and only if C supports $(x, y) \in X \times Y$.

For any set I, and any I-indexed family of $\mathsf{Sym}(\mathcal{N})$-sets, $\{X_i\}_{i \in I}$, the coproduct of the underlying sets is equipped with action given by

$$\sigma \bullet_{(\coprod_{i \in I} X_i)} (\mathsf{inj}_j(x)) = \mathsf{inj}_j(\sigma \bullet_{X_j} x)$$

for any $j \in I$ and any $x \in X_j$. A finite set $C \subseteq_{\mathrm{f}} \mathcal{N}$ supports $x \in X_j$ if and only if C supports $\mathsf{inj}_j(x) \in \coprod_{i \in I} X_i$.

Names. The set $\mathcal{N}$ of names has an action

$$\bullet_{\mathcal{N}} : \mathsf{Sym}(\mathcal{N}) \times \mathcal{N} \to \mathcal{N}$$

which is given by evaluating a permutation at a name. A set $C \subseteq_{\mathrm{f}} \mathcal{N}$ supports $a \in \mathcal{N}$ if and only if $a \in C$. Thus $\mathsf{supp}_{\mathcal{N}}(a) = \{a\}$.

Name abstraction. For each nominal set X we have a set $[\mathcal{N}]X$ of elements of X with names abstracted from them. Let

$$[\mathcal{N}]X = (\mathcal{N} \times X)/_{\sim}$$

where $(a, x) \sim (b, y)$ if there exists $c \in \mathcal{N}$ such that $c \# x$, $c \# y$, and

$$[c \leftrightarrow a] \bullet_X x = [c \leftrightarrow b] \bullet_X y \quad .$$

This equivalence relation can be seen as α-equivalence, an interpretation that will be particularly important in Section 7.4.

We will write $\langle a \rangle x$ for the $\sim$-equivalence class containing (a, x). Notice that if C supports $x \in X$ then $(C - \{a\})$ supports $\langle a \rangle x \in [\mathcal{N}]X$.

Proposition 7.1.3. *1. Let X be a nominal set.*

a) $\forall a \in \mathcal{N},\ x, y \in X.\ \langle a \rangle x = \langle a \rangle y \implies x = y$.

b) $\forall a \in \mathcal{N},\ x \in X.\ a \# \langle a \rangle x$.

2. There is a bijective correspondence between equivariant functions $Y \times [\mathcal{N}]X \to Z$ and equivariant functions $f : Y \times \mathcal{N} \times X \to Z$ for which

$$\forall y, a, x.\ a \# y \implies a \# f(y, a, x) \quad .$$

□

7.1.4 Nominal sets are the Schanuel topos

In this section we recall some constructions involved in the following well-known result.

Theorem 7.1.4. *The category* **Nom** *of nominal sets is equivalent to the category* $\mathbf{Sh}(\mathbf{I})$ *of sheaves.* □

Gabbay and Pitts [2001, Sec. 7] discuss this result and supply some references. Here, we provide an outline of a proof. We conclude this subsection by explaining how the operators on **Nom**, introduced in Section 7.1.3, correspond to the operators on $\mathbf{Sh}(\mathbf{I})$ considered in Theorem 4.2.6.

From nominal sets to sheaves. Given a nominal set $X \in \mathbf{Nom}$, we define a sheaf $P \in \mathbf{Set}^{\mathbf{I}}$ as follows. For any $C \in \mathbf{I}$, we let

$$P(C) = \left\{ x \in X \;\middle|\; C \text{ supports } x \right\}$$

while for any morphism $\iota : C \rightarrowtail D$ in $\mathbf{I}$, and any $x \in P(C)$, we let

$$P\iota(x) = \iota^\sharp \bullet_X x \quad .$$

Here, $\iota^\sharp \in \mathsf{Sym}(\mathcal{N})$ is a permutation such that $\iota^\sharp|_C = \iota$; since C supports x, it doesn't matter which one is chosen. There always is such a permutation, for the following reason. The restricted function $\iota|_C : C \rightarrowtail \iota(C)$ is a bijection, and so the set $(\iota(C) - C)$ is in bijection with $(C - \iota(C))$. We pick any bijection $\beta : (\iota(C) - C) \to (C - \iota(C))$, and now define a permutation $\iota^\sharp \in \mathsf{Sym}(\mathcal{N})$ according to the following diagram.

$$\begin{array}{ccc} \mathcal{N} & \text{—=—} & C \uplus (\iota(C) - C) \uplus (\mathcal{N} - C - \iota(C)) \\ {\scriptstyle \iota^\sharp}\downarrow & & \downarrow{\scriptstyle \iota|_C \uplus \beta \uplus \mathsf{id}} \\ \mathcal{N} & \text{—=—} & \iota(C) \uplus (C - \iota(C)) \uplus (\mathcal{N} - C - \iota(C)) \end{array}$$

(Here, and elsewhere in this thesis, we adopt the usual convention that set subtraction associates to the left: so $\mathcal{N} - C - \iota(C) = (\mathcal{N} - C) - \iota(C)$.)

To see that this presheaf $P \in \mathbf{Set}^{\mathbf{I}}$ satisfies the sheaf condition, observe that for any $C, D \in \mathbf{I}$ and $x \in P(C)$ such that $D \subseteq C$: the set D supports $x \in P(C)$ (in the sense of Section 4.2.1) if and only if D supports x in the nominal set X.

From equivariant functions to natural transformations. We now translate equivariant functions between nominal sets into natural transformations between the corresponding sheaves. Suppose we have two nominal sets, X and X', with associated sheaves P and P' in $\mathbf{Sh}(\mathbf{I})$. An equivariant function $f : X \to X'$ between the nominal sets induces a natural transformation $\alpha : P \to P'$ as follows. For any $C \in \mathbf{I}$, and any $x \in P(C)$ (*i.e.* $x \in X$ that is supported by C) we let

$$\alpha_C(x) = f(x) \quad .$$

This makes sense as a result of Prop. 7.1.2(4).

Equivalence. One can straightforwardly verify that the above construction yields a functor $\mathbf{Nom} \to \mathbf{Sh}(\mathbf{I})$ that is full and faithful. To conclude that it is an equivalence, we will outline a construction that demonstrates that the functor is essentially surjective. To this end, we consider a sheaf P, and from this construct a nominal set that corresponds to this sheaf.

To construct this nominal set, we first consider the presheaf $\langle P \rangle \in \mathbf{Set}^{\mathbf{B}}$ of seeds of P, introduced in (4.4.1). From this we form the set $\int \langle P \rangle$ of elements of $\langle P \rangle$, as elsewhere in this thesis, following (3.2.2). The set $\int \langle P \rangle$ is to be thought of as a set of *minimal elements* of the presheaf P. This set has $\mathsf{Sym}(\mathcal{N})$-action given by

$$\sigma \bullet (C \vdash p) = \big(\sigma(C) \vdash P\sigma|_C(p)\big) \quad .$$

To see that the sheaf corresponding to this nominal set is indeed isomorphic to P, observe that for any $C, D \in \mathbf{I}$ and $x \in P(C)$ such that $D \subseteq C$, the set D supports $x \in P(C)$ if and only if D supports $(D \vdash x)$ in the nominal set $\int \langle P \rangle$.

Relating constructions in Nom with constructions in Sh(I). Because **Nom** and $\mathbf{Sh}(\mathbf{I})$ are equivalent, limits and colimits correspond across the equivalence. It is straightforward to show that the nominal set of names $\mathcal{N} \in \mathbf{Nom}$ corresponds to the sheaf $N \in \mathbf{Sh}(\mathbf{I})$. Moreover, the binding operator $[\mathcal{N}](-)$ on **Nom** corresponds to the name generation operator δ on $\mathbf{Sh}(\mathbf{I})$.

It is easy to see that the sheaf associated to the nominal set X_π of π-calculus terms, introduced in equations 7.1.1, is precisely the sheaf $P_\pi \in \mathbf{Sh}(\mathbf{I})$ introduced in equations 3.2.1.

7.1.5 Non-determinism in nominal sets

The pointwise powerset endofunctor on $\mathbf{Set}^{\mathbf{I}}$ (of equations 3.2.6) was used in Section 3.2.2 to introduce non-determinism into the coalgebraic model of ground transitions. We now explictly describe an endofunctor on **Nom** that corresponds to the pointwise powerset functor, via the equivalence $\mathbf{Nom} \simeq \mathbf{Sh}(\mathbf{I})$ considered in the previous subsection.

We begin by introducing the notion of *support-bounded subset* of a nominal set. We use this notion to define an endofunctor on **Nom**. We prove that this endofunctor corresponds to the pointwise powerset functor on $\mathbf{Sh}(\mathbf{I})$, and thus arrive at an explicit description of an endofunctor on **Nom** for ground behaviour.

Support-bounded subsets. Let X be a nominal set. We say that a subset S of X is *support-bounded* if there is a finite set of names $C \subseteq_{\mathrm{f}} \mathcal{N}$ that supports every element of S. We write $S \subseteq_{\mathrm{sb}} X$ to indicate that S is a support-bounded subset of X.

A powerset endofunctor on Nom. We introduce an endofunctor $\mathscr{P}_{\mathrm{sb}}$ on **Nom** as follows. For any nominal set X, we define the set $\mathscr{P}_{\mathrm{sb}}X$ to be the set of support-bounded subsets of X. The group action on $\mathscr{P}_{\mathrm{sb}}X$ is given by, for each $\sigma \in \mathsf{Sym}(\mathcal{N})$, and each $S \subseteq_{\mathrm{sb}} X$,

$$\sigma \bullet_{\mathscr{P}_{\mathrm{sb}}X} S = \{\sigma \bullet_X x \mid x \in S\} \quad .$$

To show that $\mathscr{P}_{\mathrm{sb}}X$ is a nominal set we provide an explicit description of the support of its elements in the following proposition.

Proposition 7.1.5. *A set of names $C \subseteq_{\mathrm{f}} \mathcal{N}$ supports S in $\mathscr{P}_{\mathrm{sb}}X$ if and only if C supports every element of S.*

Proof. From right-to-left is straightforward, so we concentrate on proving the left-to-right direction. To this end, consider $S \in \mathscr{P}_{\mathrm{sb}}X$ that is supported by C. We know that there is a finite set that supports every element of S. Here, we let D be the smallest set that supports every element of S. (Such a smallest set can always be found, since if both D' and D'' support every element of S, then so does $(D' \cap D'')$.) We will show that $D \subseteq C$. To do this, we suppose not, and establish a contradiction.

If $D \not\subseteq C$ then there must be a name $c \in (D - C)$. We pick another name $z \in (\mathcal{N} - C - D)$ — this is possible since $\mathcal{N}$ is infinite, while C and D are finite. Since the set D is the smallest set that supports every element of S, there must be an element $x \in S$ for which $c \in \mathsf{supp}_X(x)$; otherwise the smaller set $(D - \{c\})$ would support every element of S. By Prop. 7.1.2(3), the least support of the permuted element $[c \leftrightarrow z] \bullet_X x$ is $\big((\mathsf{supp}_X(x) - \{c\}) \cup \{z\}\big)$. Since $z \notin D$, we see that D does *not* support $[c \leftrightarrow z] \bullet_X x$, and so this permuted element cannot be in S.

On the other hand, our assumption is that C supports S, and certainly the swapping permutation $[c \leftrightarrow z]$ fixes C. So the permuted element $[c \leftrightarrow z] \bullet_X x$ *is* in S — a contradiction.

Thus $D \subseteq C$, and, by Prop. 7.1.2(1), we know that C supports every element of S. □

The functorial action of $\mathscr{P}_{\mathrm{sb}}$ is as follows: for any equivariant function $f : X \to Y$, and any $S \subseteq_{\mathrm{sb}} X$, we let

$$(\mathscr{P}_{\mathrm{sb}}f)(S) = \{ f(x) \mid x \in S \} \quad .$$

The subset $(\mathscr{P}_{\mathrm{sb}}f)(S) \subseteq Y$ is necessarily support-bounded; this follows from Prop. 7.1.2(4).

Support bounded powersets and sheaves.

Proposition 7.1.6. *The support-bounded powerset endofunctor $\mathscr{P}_{\mathrm{sb}}$ on* **Nom** *lifts the pointwise powerset endofunctor on* $\mathbf{Sh}(\mathbf{I})$ *across the equivalence* $\mathbf{Nom} \simeq \mathbf{Sh}(\mathbf{I})$ *of Section 7.1.4.*

Proof. Consider a nominal set X, and let P be the corresponding sheaf in $\mathbf{Sh}(\mathbf{I})$, constructed according to the techniques of Section 7.1.4. We must show that the pointwise powerset sheaf $\mathscr{P}P$ is the sheaf corresponding to the nominal set $(\mathscr{P}_{\mathrm{sb}}X)$. In other words, we must show, for each $C \in \mathbf{I}$, that

$$\mathscr{P}(PC) = \left\{ S \subseteq_{\mathrm{sb}} X \;\middle|\; C \text{ supports } S \right\} \quad .$$

This result follows immediately from Prop. 7.1.5. It is equally straightforward to show a correspondence for the actions of the two endofunctors. □

Coalgebras for ground behaviour over Nom. The endofunctor L_{g} on $\mathbf{Sh}(\mathbf{I})$ (of equation 3.2.15), for determinstic ground behaviour, is lifted to **Nom** as follows.

$$\begin{array}{rcll} L_{\mathrm{g}}(-) & = & \mathsf{binp} : \mathcal{N} \times [\mathcal{N}](-) & \textit{Bound input} \\ & + & \mathsf{out} : \mathcal{N} \times \mathcal{N} \times (-) & \textit{Free output} \\ & + & \mathsf{bout} : \mathcal{N} \times [\mathcal{N}](-) & \textit{Bound output} \\ & + & \mathsf{tau} : (-) & \textit{Silent}. \end{array} \tag{7.1.7a}$$

Using Prop. 7.1.6, we see that the endofunctor B_{g} on $\mathbf{Sh}(\mathbf{I})$ (of equation 3.2.16), for nondeterminstic ground behaviour, is lifted as follows.

$$B_{\mathrm{g}}(-) = \mathscr{P}_{\mathrm{sb}}(L_{\mathrm{g}}(-)) \quad . \tag{7.1.7b}$$

7.2 Nominal logic

Nominal logic was introduced by Pitts [2003] as a first-order theory of names and binding. We introduce the syntax of the variant that we will use in Section 7.2.1. In Section 7.2.2 we define structures for signatures, and briefly note how models of theories are defined.

Our presentation here differs from that of Pitts in two ways. Firstly, we only allow one sort of names, while Pitts allows a countable number of name sorts. Secondly, we consider structures for signatures within a class of different categories, whereas Pitts only considers structures in **Nom**. This latter generalisation is important in Section 7.4.

For models of nominal logic theories, though, we only consider structures in **Nom** (and, indeed, the reader is referred to Pitts's article for full details) primarily because these kinds of models are sufficient for this thesis.

7.2.1 Syntax of nominal logic

Nominal logic signatures. Here, a *nominal logic signature* is understood as a collection of basic sorts together with a collection of relation symbols and a collection of function symbols. The sorts of a signature are defined inductively as follows: every basic sort is a sort; there is a sort N of names; for every sort S there is a sort $[\mathsf{N}]\mathsf{S}$ of abstractions. Each relation symbol is equipped with an arity consisting of a list of sorts (we write $R \subseteq \mathsf{S}_1, \ldots, \mathsf{S}_n$); each function symbol is equipped with an arity consisting of a list of sorts, together with a result sort (we write $f : \mathsf{S}_1, \ldots, \mathsf{S}_n \to \mathsf{S}$).

Constructing terms and formulas. Terms and formulas over the signature are built up as in many-sorted first-order logic with equality, with three additional constructions: if a, b are terms of sort N and t is a term of sort S then: $\langle a \rangle t$ is a term of sort $[\mathsf{N}]\mathsf{S}$; $[a \leftrightarrow b]t$ is a term of sort S; and $a \# t$ is a formula.

Nominal logic theories. A *nominal logic theory* is a nominal logic signature together with a collection of axioms in the language of nominal logic.

7.2.2 Semantics of nominal logic

Structures for nominal logic signatures. Let $\mathbb{S}$ be a nominal logic signature. Let $\mathscr{C}$ be a category with finite products, a distinguished object $\mathscr{N}$ ('of names'), and a distinguished 'binding' endofunctor $[\mathscr{N}]- : \mathscr{C} \to \mathscr{C}$. An $\mathbb{S}$*-structure* M in $\mathscr{C}$ is given as follows.

- To each basic sort X of the signature is assigned an object $[\![\mathsf{X}]\!]_M$. Compound sorts are then interpreted as follows: $[\![\mathsf{N}]\!]_M = \mathscr{N}$, and $[\![[\mathsf{N}]\mathsf{S}]\!]_M = [\mathscr{N}][\![\mathsf{S}]\!]_M$.
- To each relation symbol $\mathsf{R} \subseteq \mathsf{S}_1, \ldots, \mathsf{S}_n$ is assigned an a subobject
$$[\![\mathsf{R}]\!]_M \subseteq [\![\mathsf{S}_1]\!]_M \times \cdots \times [\![\mathsf{S}_n]\!]_M \quad .$$
- To each function symbol $\mathsf{f} : \mathsf{S}_1, \ldots, \mathsf{S}_n \to \mathsf{S}$ is assigned a morphism
$$[\![\mathsf{f}]\!]_M : [\![\mathsf{S}_1]\!]_M \times \cdots \times [\![\mathsf{S}_n]\!]_M \to [\![\mathsf{S}]\!]_M \quad .$$

Morphisms between $\mathbb{S}$-structures. Let M, M' be two $\mathbb{S}$-structures in $\mathscr{C}$. Suppose we are given, for each basic sort X, a morphism
$$f_{\mathsf{X}} : [\![\mathsf{X}]\!]_M \to [\![\mathsf{X}]\!]_{M'} \quad \text{in } \mathscr{C}.$$

Then for a compound sort S, we define f_{S} inductively on the structure of sorts, as follows: we let $f_{\mathsf{N}} = \mathsf{id}_{\mathscr{N}}$, and we let $f_{[\mathsf{N}]\mathsf{S}} = [\mathscr{N}]f_{\mathsf{S}}$. A *homomorphism* of $\mathbb{S}$-structures $M \to M'$ is such a sort indexed family of morphisms

$$\{f_{\mathsf{S}} : [\![\mathsf{S}]\!]_M \to [\![\mathsf{S}]\!]_{M'}\}_{\mathsf{S}}$$

such that the following conditions hold.

- For each function symbol $\mathsf{f} : \mathsf{S}_1, \ldots, \mathsf{S}_n \to \mathsf{S}$ the following diagram commutes.

$$\begin{array}{ccc} [\![\mathsf{S}_1]\!]_M \times \cdots \times [\![\mathsf{S}_n]\!]_M & \xrightarrow{[\![\mathsf{f}]\!]_M} & [\![\mathsf{S}]\!]_M \\ \downarrow{\scriptstyle f_{\mathsf{S}_1} \times \cdots \times f_{\mathsf{S}_n}} & & \downarrow{\scriptstyle f_{\mathsf{S}}} \\ [\![\mathsf{S}_1]\!]_{M'} \times \cdots \times [\![\mathsf{S}_n]\!]_{M'} & \xrightarrow[[\![\mathsf{f}]\!]_{M'}]{} & [\![\mathsf{S}]\!]_{M'} \end{array}$$

- For each relation symbol $\mathsf{R} \subseteq \mathsf{S}_1, \ldots, \mathsf{S}_n$ there exists a morphism (necessarily unique)

$$f_{\mathsf{R}} : [\![\mathsf{R}]\!]_M \to [\![\mathsf{R}]\!]_{M'}$$

  such that the following diagram commutes.

$$\begin{array}{ccc} [\![\mathsf{R}]\!]_M & \rightarrowtail & [\![\mathsf{S}_1]\!]_M \times \cdots \times [\![\mathsf{S}_n]\!]_M \\ \downarrow{\scriptstyle f_{\mathsf{R}}} & & \downarrow{\scriptstyle f_{\mathsf{S}_1} \times \cdots \times f_{\mathsf{S}_n}} \\ [\![\mathsf{R}]\!]_{M'} & \rightarrowtail & [\![\mathsf{S}_1]\!]_{M'} \times \cdots \times [\![\mathsf{S}_n]\!]_{M'} \end{array}$$

$\mathbb{S}$-structure homomorphisms are composed by composing the underlying morphisms of base sorts; the identity $\mathbb{S}$-structure homomorphism is found by taking the identity morphisms for the interpretations of each of the base sorts. Thus we have a category of $\mathbb{S}$-structures.

Structures in Nom. The case $\mathscr{C} = \mathbf{Nom}$ is particularly important — indeed it is the only case originally considered by Pitts [2003]. In this case, the object of names and the binding endofunctor are taken to be the corresponding structures in **Nom** as introduced in Section 7.1.3.

Models. Nominal logic formulas over a signature $\mathbb{S}$ are interpreted in $\mathbb{S}$-structures in **Nom**, in a straightforward way. Pitts [2003, Defn. 2] provides the details and a thorough analysis. Thus we arrive at a category of models for a nominal logic theory over a signature $\mathbb{S}$; it is the full subcategory of the category of $\mathbb{S}$-structures, containing those $\mathbb{S}$-structures that satisfy interpretations of the axioms.

7.3 Nominal substitutions

The framework of nominal sets has provided a notion of α-equivalence, for which permutation actions were required. When defining name-passing systems, though, *non*-injective substitutions are indispensable — for instance, in the rule for communication in Figure 3.3, and also in the models of uniform input of Section 3.4. For this reason we now introduce a theory of nominal substitutions which, roughly speaking, is a theory of nominal sets that additionally allow arbitrary non-injective substitution of names.

We begin this section with the nominal logic theory of nominal substitutions. Two presentations are possible. The first has a more elaborate signature but a simpler axiomatisation. The second has a simpler signature but requires an extra axiom. It will be useful in Section 8.4 to have the extra

axiom pushed into the signature (in a sense we will make precise), but for proving basic results about the theory it is convenient to have the extra axiom explicit.

In Section 7.3.1 we prove that the category **NomSub** of nominal substitutions is equivalent to the sheaf category **Sh**(**F**) considered in Section 4.3. By doing so we can more readily understand the exactness properties of **NomSub**, and in Section 7.3.2 we give explicit descriptions of various operations in **NomSub**.

We conclude this introductory paragraph by noting that Fiore, Plotkin, and Turi [1999] took the category $\mathbf{Set}^{\mathbf{F}}$ as a basis for their study of abstract syntax with variable binding. Thus, by working with nominal substitutions, the work of Fiore *et al.* can be recast in the context of nominal sets, and two approaches to abstract syntax are related.

The theory of nominal substitutions. The signature for nominal substitutions has one sort X and one function symbol $\mathsf{sub} : \mathsf{N}, [\mathsf{N}]\mathsf{X} \to \mathsf{X}$. We use '$[b/a]x$' as shorthand for the expression '$\mathsf{sub}(b, \langle a \rangle\, x)$'.

The theory of nominal substitutions consists of the signature for nominal substitutions together with the following four axioms.

NomSub-1. $\forall a : \mathsf{N}.\ \forall x : \mathsf{X}.\ [a/a]x = x.$

NomSub-2. $\forall a, b : \mathsf{N}.\ \forall x : \mathsf{X}.\ a \# x \implies [b/a]x = x.$

NomSub-3. $\forall a, b, c : \mathsf{N}.\ \forall x : \mathsf{X}.\ [c/b][b/a]x = [c/b][c/a]x.$

NomSub-4. $\forall a, b, c, d : \mathsf{N}.\ \forall x : \mathsf{X}.\ c \neq b \neq a \neq d \implies [d/b][c/a]x = [c/a][d/b]x.$

We write **NomSub** for the category of models of this theory in **Nom**.

We will abuse notation by writing X for a nominal substitution (X, sub_X).

For a first example, notice that the nominal set X_π of π-calculus processes, introduced in equation 7.1.1, can be given a nominal substitution in a straightforward manner. For any π-calculus process $t \in X_\pi$, and any names a, b, we let $[b/a]t$ be the process t in which all free occurrences of the name a are replaced with b.

An alternative presentation. An alternative theory of nominal substitutions is inspired by Prop. 7.1.3(2) above, as follows. As with the original theory, the signature has one sort, X, and an operator sub with result sort X, but this time with arity $(\mathsf{N}, \mathsf{N}, \mathsf{X})$. Writing '$[b/a]x$' for '$\mathsf{sub}(b, a, x)$', Axioms NomSub-1–4 above can be interpreted for this signature. To these axioms we add the following additional one.

NomSub-0. $\forall a, b : N, x : \mathsf{X}.\ a \neq b \implies a \# [b/a]x.$

Proposition 7.3.1. *The category of models for this alternative theory is isomorphic to the category* **NomSub** *introduced above.*

A proof of this statement is provided in Appendix 7.A.

7.3.1 Nominal substitutions are sheaves

We now explain that the category **NomSub** of nominal substitutions is equivalent to the sheaf category **Sh**(**F**) introduced in Section 4.3. To do this, we begin by explaining that the forgetful functor **Sh**(**F**) → **Sh**(**I**) is monadic. We then interpret the theory of nominal substitutions directly in **Sh**(**I**). So, to conclude the main theorem (Theorem 7.3.2), it remains to show that the category of nominal substitutions in **Sh**(**I**) is equivalent to the category of algebras for the monad arising from the monadic forgetful functor **Sh**(**F**) → **Sh**(**I**).

$\mathbf{Set}^{\mathbf{F}}$ is monadic over $\mathbf{Set}^{\mathbf{I}}$. Before looking at the relationships between the sheaf subcategories, we remark that $\mathbf{Set}^{\mathbf{F}}$ is monadic over $\mathbf{Set}^{\mathbf{I}}$. The forgetful functor $U^{\mathbf{I}}_{\mathbf{F}} : \mathbf{Set}^{\mathbf{F}} \to \mathbf{Set}^{\mathbf{I}}$ is the inverse image of the essential geometric morphism induced by the inclusion functor $j^{\mathbf{I}}_{\mathbf{F}} : \mathbf{I} \to \mathbf{F}$, and so it has a left adjoint $(j^{\mathbf{I}}_{\mathbf{F}})_! : \mathbf{Set}^{\mathbf{I}} \to \mathbf{Set}^{\mathbf{F}}$. To conclude that the adjunction $(j^{\mathbf{I}}_{\mathbf{F}})_! \dashv U^{\mathbf{I}}_{\mathbf{F}} : \mathbf{Set}^{\mathbf{F}} \to \mathbf{Set}^{\mathbf{I}}$ is monadic, we appeal to the weak monadicity theorem: the category $\mathbf{Set}^{\mathbf{F}}$ has all colimits, and certainly coequalisers; the functor $U^{\mathbf{I}}_{\mathbf{F}} : \mathbf{Set}^{\mathbf{F}} \to \mathbf{Set}^{\mathbf{I}}$ has a right adjoint and so preserves colimits, and coequalisers in particular; and the functor $U^{\mathbf{I}}_{\mathbf{F}} : \mathbf{Set}^{\mathbf{F}} \to \mathbf{Set}^{\mathbf{I}}$ reflects isomorphisms (because $j^{\mathbf{I}}_{\mathbf{F}} : \mathbf{I} \to \mathbf{F}$ is surjective on objects).

$\mathbf{Sh}(\mathbf{F})$ is monadic over $\mathbf{Sh}(\mathbf{I})$: outline. In Prop. 4.3.1 we showed that the forgetful functor $U^{\mathbf{I}}_{\mathbf{F}} : \mathbf{Set}^{\mathbf{F}} \to \mathbf{Set}^{\mathbf{I}}$ restricts to a functor $U^{\mathbf{I}}_{\mathbf{F}} : \mathbf{Sh}(\mathbf{F}) \to \mathbf{Sh}(\mathbf{I})$. To show that this functor is monadic, we can use the weak monadicity theorem in essentially the same way as above. The right adjoint for $U^{\mathbf{I}}_{\mathbf{F}} : \mathbf{Sh}(\mathbf{F}) \to \mathbf{Sh}(\mathbf{I})$ was discussed at the end of Section 4.3. It remains for us to show that the left adjoint $(j^{\mathbf{I}}_{\mathbf{F}})_! : \mathbf{Set}^{\mathbf{I}} \to \mathbf{Set}^{\mathbf{F}}$ restricts to a functor $(j^{\mathbf{I}}_{\mathbf{F}})_! : \mathbf{Sh}(\mathbf{I}) \to \mathbf{Sh}(\mathbf{F})$.

A left adjoint for the forgetful functor $\mathbf{Sh}(\mathbf{F}) \to \mathbf{Sh}(\mathbf{I})$. In order to reason about the left adjoint $(j^{\mathbf{I}}_{\mathbf{F}})_! : \mathbf{Set}^{\mathbf{I}} \to \mathbf{Set}^{\mathbf{F}}$, we write down an explicit description. For $P \in \mathbf{Set}^{\mathbf{I}}$, and $C \in \mathbf{F}$, we have

$$(j^{\mathbf{I}}_{\mathbf{F}})_! P(C) = \left(\coprod_{D \in \mathbf{F}} (\mathbf{F}(D, C) \times P(D)) \right) \Big/ \sim_C$$

where $\sim_C$ is the equivalence relation on the set $\left(\coprod_{D \in \mathbf{F}} (\mathbf{F}(D, C) \times P(D)) \right)$ generated by

$$\mathsf{inj}_D \left(f\iota, p \right) \sim_C \mathsf{inj}_{D'} \left(f, P\iota(p) \right) \qquad \begin{array}{l} \text{for any } D, D' \in \mathbf{I},\ \iota : D \rightarrowtail D' \text{ in } \mathbf{I}, \\ \quad \text{any } f : D' \to C \text{ in } \mathbf{F},\ p \in P(D). \end{array}$$

An equivalence class $[\mathsf{inj}_D(f, p)]$ in $(j^{\mathbf{I}}_{\mathbf{F}})_! P(C)$ is to be thought of as describing the element p subject to the substitution f.

The action of $(j^{\mathbf{I}}_{\mathbf{F}})_! P$ is as follows. For any function $g : C \to C'$ in $\mathbf{F}$, and any $\sim_C$-equivalence class $[\mathsf{inj}_D(f, p)]$ in $(j^{\mathbf{I}}_{\mathbf{F}})_! P(C)$, we let

$$(j^{\mathbf{I}}_{\mathbf{F}})_! P g [\mathsf{inj}_D(f, p)] = [\mathsf{inj}_D(g \circ f, p)] \qquad \text{in } (j^{\mathbf{I}}_{\mathbf{F}})_! P(C').$$

A natural transformation $\alpha : P \to Q$ between presheaves in $\mathbf{Set}^{\mathbf{I}}$ induces a natural transformation $(j^{\mathbf{I}}_{\mathbf{F}})_! \alpha : (j^{\mathbf{I}}_{\mathbf{F}})_! P \to (j^{\mathbf{I}}_{\mathbf{F}})_! Q$ given as follows. For each $C \in \mathbf{F}$, and each $[\mathsf{inj}_D(f, p)]$ in $(j^{\mathbf{I}}_{\mathbf{F}})_! P(C)$, we let

$$((j^{\mathbf{I}}_{\mathbf{F}})_! \alpha)_C [\mathsf{inj}_D(f, p)] = [\mathsf{inj}_D(f, \alpha_D(p))] \quad .$$

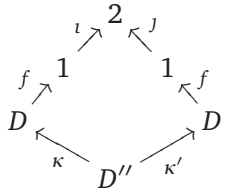
If P is a sheaf in $\mathbf{Sh}(\mathbf{I})$, then $(j^{\mathbf{I}}_{\mathbf{F}})_! P$ is a sheaf in $\mathbf{Sh}(\mathbf{F})$, as we now explain.

We begin by noting that when P is a sheaf then the equivalence relation used to define $(j^{\mathbf{I}}_{\mathbf{F}})_! P$ has the following characterisation. For any set $C \in \mathbf{F}$ we define a relation $\equiv_C$ on the set $\coprod_{D \in \mathbf{F}} (\mathbf{F}(D, C) \times P(D))$ as follows. For any two components $\mathsf{inj}_D(f, p)$, $\mathsf{inj}_{D'}(f', p')$, we let $\mathsf{inj}_D(f, p) \equiv_C \mathsf{inj}_{D'}(f', p')$ if and only if there is a set $D'' \in \mathbf{F}$, an element $p'' \in P(D'')$, and injections $\kappa : D'' \rightarrowtail D$, $\kappa' : D'' \rightarrowtail D'$ in $\mathbf{I}$ such that $f \circ \kappa = f' \circ \kappa'$ and such that $P\kappa(p'') = p$ and $P\kappa'(p'') = p'$. It is easy to show that $\equiv_C$ is an equivalence relation; the sheaf condition on P (in the form of Prop. 4.2.3) is required for transitivity. In this circumstance, it is straightforward to show that the relation $\equiv_C$ is equal to the relation $\sim_C$ used in the definition of $(j^{\mathbf{I}}_{\mathbf{F}})_! P$.

Following Prop. 4.3.3, to check separatedness for the presheaf $(j^{\mathbf{I}}_{\mathbf{F}})_! P$ in $\mathbf{Set}^{\mathbf{F}}$, it is sufficient to check that the action $(j^{\mathbf{I}}_{\mathbf{F}})_! P(! : 0 \to 1)$ is injective. (Here, $! : 0 \to 1$ denotes the initial and terminal morphism into an arbitrary singleton name set.) This follows straightforwardly from the above characterisation of $(j^{\mathbf{I}}_{\mathbf{F}})_! P$.

To check that $(j_{\mathbf{F}}^{\mathbf{I}})_! P$ satisfies the sheaf condition, we again use Prop. 4.3.3: it is sufficient to check the sheaf condition for the elements of $(j_{\mathbf{F}}^{\mathbf{I}})_! P(1)$ with empty support.

To this end, consider an element $[\mathsf{inj}_D(f,p)] \in (j_{\mathbf{F}}^{\mathbf{I}})_! P(1)$ with empty support. Note that 1 is terminal, and so $f : D \to 1$ can only be the unique terminal map. Consider a two element set, $2 \in \mathbf{I}$, and distinct injections $\iota, \jmath : 1 \rightarrowtail 2$. Since the element $[\mathsf{inj}_D(f,p)]$ has empty support, we know that $[\iota][\mathsf{inj}_D(f,p)] = [\jmath][\mathsf{inj}_D(f,p)]$ in $(j_{\mathbf{F}}^{\mathbf{I}})_! P(2)$; so $[\mathsf{inj}_D(\iota f,p)] = [\mathsf{inj}_D(\jmath f,p)]$. The above characterisation $\equiv_C$ of $\sim_C$ means that there is a set D'', and an element $p'' \in P(D'')$, and injections $\kappa : D'' \rightarrowtail D$, $\kappa' : D'' \rightarrowtail D'$ in $\mathbf{I}$, such that $[\kappa]p'' = p = [\kappa']p''$ and making the following diagram commute.



Thus the set D'' must be empty, and we have an amalgamation $p'' \in P(\emptyset)$ for $p \in P(1)$, and hence an amalgamation $[\mathsf{inj}_\emptyset(\mathsf{id}_\emptyset, p'')] \in (j_{\mathbf{F}}^{\mathbf{I}})_! P(\emptyset)$, as required.

It follows that the functor $(j_{\mathbf{F}}^{\mathbf{I}})_! : \mathbf{Set}^{\mathbf{I}} \to \mathbf{Set}^{\mathbf{F}}$ restricts to give a left adjoint for $U_{\mathbf{F}}^{\mathbf{I}} : \mathbf{Sh}(\mathbf{F}) \to \mathbf{Sh}(\mathbf{I})$, as required. Thus the forgetful functor $U_{\mathbf{F}}^{\mathbf{I}} : \mathbf{Sh}(\mathbf{F}) \to \mathbf{Sh}(\mathbf{I})$ is monadic.

The monad on Sh(I). We record here explicit descriptions of the unit and multiplication of the monad $(U_{\mathbf{F}}^{\mathbf{I}} \circ (j_{\mathbf{F}}^{\mathbf{I}})_!)$ on $\mathbf{Sh}(\mathbf{I})$.

The unit is a natural transformation $\eta : \mathsf{id} \to (U_{\mathbf{F}}^{\mathbf{I}} \circ (j_{\mathbf{F}}^{\mathbf{I}})_!)$ between endofunctors on $\mathbf{Set}^{\mathbf{I}}$ given as follows. For each $P \in \mathbf{Set}^{\mathbf{I}}$, $C \in \mathbf{I}$, and $p \in P(C)$, we let

$$\eta_{P,C}(p) = [\mathsf{inj}_C(\mathsf{id}_C, p)] \quad .$$

The multiplication is a natural transformation

$$\mu : (U_{\mathbf{F}}^{\mathbf{I}} \circ (j_{\mathbf{F}}^{\mathbf{I}})_! \circ U_{\mathbf{F}}^{\mathbf{I}} \circ (j_{\mathbf{F}}^{\mathbf{I}})_!) \to (U_{\mathbf{F}}^{\mathbf{I}} \circ (j_{\mathbf{F}}^{\mathbf{I}})_!)$$

between endofunctors on $\mathbf{Set}^{\mathbf{I}}$ given as follows. For each $P \in \mathbf{Set}^{\mathbf{I}}$, $C, D, E \in \mathbf{I}$, and $f : E \to D$, $g : D \to C$ in $\mathbf{F}$, and $p \in P(E)$ we let

$$\mu_{P,C}\left[\mathsf{inj}_D(g, [\mathsf{inj}_E(f,p)])\right] = [\mathsf{inj}_E(g \circ f, p)] \quad .$$

Nominal substitutions in Sh(I). We can interpret the theory of nominal substitutions directly in $\mathbf{Sh}(\mathbf{I})$ by using the equivalence between $\mathbf{Sh}(\mathbf{I})$ and **Nom** recalled in Theorem 7.1.4. Here, we focus on the second, alternative presentation of the theory because it is convenient not to have the binder in the arity.

A nominal substitution in $\mathbf{Sh}(\mathbf{I})$ is given by a sheaf P together with a natural transformation $N \times N \times P \to P$, satisfying the following properties, for all $C \in \mathbf{I}$, $a, b, c, d \in C$, and all $p \in P(C)$:

NomSub-0. $a \neq b \implies a \notin \mathsf{supp}(\mathsf{sub}_C(b, a, p))$.

NomSub-1. $\mathsf{sub}_C(a, a, p) = p$.

NomSub-2. $a \notin \mathsf{supp}(p) \implies \mathsf{sub}_C(b, a, p) = p$.

NomSub-3. $\mathsf{sub}_C\left(c, b, \mathsf{sub}(b, a, p)\right) = \mathsf{sub}_C\left(c, b, \mathsf{sub}(c, a, p)\right)$.

NomSub-4. If $c \neq b \neq a \neq d$
then $\mathsf{sub}_C\left(d, b, \mathsf{sub}(c, a, p)\right) = \mathsf{sub}_C\left(c, a, \mathsf{sub}(d, b, p)\right)$.

(Here, we are using the least support construction supp on elements of sheaves in **Sh**(**I**), defined in Section 4.4.1.)

A morphism between nominal substitutions in **Sh**(**I**), say (P, sub_P) and (Q, sub_Q), is a natural transformation $\alpha : P \to Q$ such that

$$\mathsf{sub}_{Q,C}(b, a, \alpha_C(p)) = \alpha_C(\mathsf{sub}_{P,C}(b, a, p)) \qquad \text{for all } C \in \mathbf{I},\ a, b \in C, \text{ and all } p \in P(C).$$

Theorem 7.3.2. *The category* **NomSub** *is equivalent to* **Sh**(**F**).

A proof of this theorem is provided in Appendix 7.B. There, we show that the category of models in **Sh**(**I**) for the theory of nominal substitutions is isomorphic to the category of algebras for the monad $(U^{\mathbf{I}}_{\mathbf{F}} \circ (j^{\mathbf{I}}_{\mathbf{F}})_!)$ on **Sh**(**I**).

7.3.2 Constructions of nominal substitutions

The forgetful functor **NomSub** → **Nom** lifts much of the structure of **Nom** into **NomSub**, as we now explain. We provide explicit descriptions of products and coproducts; we define a nominal substitution structure for the object of names; and we lift the binding operator from **Nom** to **NomSub**.

Products and coproducts. Since **NomSub** is a Grothendieck topos, it has limits and colimits, and the forgetful functor **NomSub** → **Nom** has left and right adjoints, and so preserves them. Thus, in particular, products and coproducts of nominal substitutions are computed according to their underlying sets.

Specifically, a singleton nominal set $\{*\}$ has the nominal substitution given by the terminal map $\mathcal{N} \times [\mathcal{N}]\{*\} \to \{*\}$. Given two nominal substitutions, X and Y, the product set $X \times Y$ has substitution action

$$[b/a](x, y) = ([b/a]x, [b/a]x) \quad .$$

The initial nominal set $\emptyset$ has nominal substitution structure given by the identity map, since $\mathcal{N} \times [\mathcal{N}]\emptyset = \emptyset$. Given two nominal substitutions, X and Y, the coproduct set $X + Y$ has substitution action

$$[b/a](\mathsf{inl}\, x) = \mathsf{inl}\, ([b/a]x) \qquad [b/a](\mathsf{inr}\, y) = \mathsf{inr}\, \left([b/a]y\right) \quad .$$

Names. The nominal set $\mathcal{N}$ of names has exactly one nominal substitution structure that is a model of the theory, given by

$$[b/a]c \;=\; \begin{cases} b & \text{if } a = c \\ c & \text{if } a \neq c. \end{cases}$$

The first case is forced by equivariance, and the second by Axiom NomSub-2.

The nominal substitution of names corresponds to the sheaf N in **Sh**(**F**) that is the inclusion of the category **F** into **Set**.

Abstraction. The abstraction operator $[\mathcal{N}]-$ on **Nom** lifts along the forgetful functor **NomSub** → **Nom** as follows. If a nominal set X is equipped with a nominal substitution structure, then a nominal substitution structure on $[\mathcal{N}]X$ can be given as follows:

$$[b/a]\left(\langle c \rangle\, x\right) = \langle c \rangle\, ([b/a]x) \qquad \text{provided } a \neq c \neq b \quad .$$

(Some such c can always be chosen.) Axioms NomSub-1–4 hold of this induced structure because they hold of the structure on X.

It might be interesting to investigate whether this lifting is the only one that satisfies the axioms and is functorial.

Internalising the structure. Using the above operations, we see that, for every nominal substitution $X \in \mathbf{NomSub}$ we can build a nominal substitution $\mathcal{N} \times [\mathcal{N}]X$. Using Axioms NOMSUB-3 and 4, we can conclude that the function $\mathsf{sub}_X : \mathcal{N} \times [\mathcal{N}]X \to X$ is a homomorphism of nominal substitutions. It is easy to show that the resulting family $\{\mathsf{sub}_X : \mathcal{N} \times [\mathcal{N}]X \to X\}_{X \in \mathbf{NomSub}}$ of homomorphisms is natural.

The reader might notice that in this subsection we have mentioned Axiom NOMSUB-2 as being necessary to ensure the uniqueness of a structure $\mathcal{N}$ for names, and Axioms NOMSUB-3 and 4 as being necessary to ensure that the structure can be internalised.

7.4 Nominal algebraic signatures

This section redevelops the theory of syntax of Section 6.1.1 in the context of name binding. We begin with a notion of *nominal algebraic signature*, which, as we explain, isolates a restricted class of the nominal logic signatures considered in Section 7.2.1. We illustrate this kind of signature with a signature for the π-calculus.

Next, we provide explicit descriptions of structures for such signatures. We provide examples, specifically: the π-calculus up-to α-equivalence, with substitution action, by interpreting in **NomSub**; and the π-calculus in raw syntax, by interpreting in **Set**. Certain special morphisms between categories induce morphisms of endofunctors, and hence functors between categories of algebras. We illustrate this by relating syntax in **NomSub** with syntax in **Nom**, and also by showing how raw syntax (in **Set**) can be instantiated into abstract syntax (in **Nom**).

Definition 7.4.1. A *nominal algebraic signature* $\mathbb{S}$ is given by a collection $\mathsf{Op}_{\mathbb{S}}$ of operators, where each operator $\mathsf{op} \in \mathsf{Op}_{\mathbb{S}}$ is associated with

- an arity of names, $\mathrm{arn}(\mathsf{op}) \in \mathbb{N}$
- an arity of terms, $\mathrm{art}(\mathsf{op}) \in \mathbb{N}$
- a binding depth $\mathrm{dep}_j(\mathsf{op}) \in \mathbb{N}$ for each term parameter $j \in [1, \mathrm{art}(\mathsf{op})]$.

The data of Definition 7.4.1 defines a nominal logic theory as follows.

- The theory has one ground sort: the term sort X.
- There are no relation symbols.
- The function symbols are the operators in $\mathsf{Op}_{\mathbb{S}}$.
- The arity of a function symbol $\mathsf{op} \in \mathsf{Op}_{\mathbb{S}}$ is given by the string $\mathsf{N}^{\mathrm{arn}(\mathsf{op})}$ concatenated with the string $\left([\mathsf{N}]^{\mathrm{dep}_j(\mathsf{op})}\mathsf{X}\right)_{j \in [1,\mathrm{art}(\mathsf{op})]}$.
- The result sort of every function symbol is the term sort X.

Example: Signature for the π-calculus. We consider a nominal algebraic signature $\mathbb{PI}$ for the π-calculus grammar that was introduced in equation 3.1.1. We have a collection of operators

$$\mathsf{Op}_{\mathbb{PI}} = \{\mathsf{nil}, \mathsf{par}, \mathsf{sum}, \mathsf{inp}, \mathsf{out}, \mathsf{tau}, \mathsf{match}, \mathsf{mismatch}, \mathsf{restrct}\} \quad .$$

The operators are assigned arities according to the following table.

$\mathsf{op} \in \mathsf{Op}_{\mathbb{PI}}$	$\mathrm{arn}(\mathsf{op})$	$\mathrm{art}(\mathsf{op})$	$(\mathrm{dep}_i(\mathsf{op}))_{i\in[1,\mathrm{art}(\mathsf{op})]}$	
nil	0	0	()	*deadlock*
par	0	2	(0,0)	*parallel composition*
sum	0	2	(0,0)	*non-deterministic choice*
inp	1	1	(1)	*input prefix*
out	2	1	(0)	*output prefix*
tau	0	1	(0)	*silent prefix*
match	2	1	(0)	*match prefix*
mismatch	2	1	(0)	*mismatch prefix*
restrct	0	1	(1)	*restriction*

(7.4.2)

Structures for signatures. Let $\mathbb{S}$ be a nominal algebraic signature. We will consider structures for this signature, as in Section 7.2.2. Thus, for any category $\mathscr{C}$ with finite products, a distinguished object $\mathcal{N}$, and an endofunctor $[\mathcal{N}]$, a $\mathbb{S}$-structure in $\mathscr{C}$ is given by an object and X in $\mathscr{C}$ together with, for each operation $\mathsf{op} \in \mathsf{Op}_{\mathbb{S}}$, a morphism

$$[\![\mathsf{op}]\!]_X : \left(\mathcal{N}^{\mathrm{arn}(\mathsf{op})} \times \prod_{i\in[1,\mathrm{art}(\mathsf{op})]} [\mathcal{N}]^{\mathrm{dep}_i(\mathsf{op})} X \right) \longrightarrow X \quad .$$

(Here, we write $[\mathcal{N}]^n X$ for the n-fold application of the endofunctor $[\mathcal{N}](-)$ to the object X.)

We will illustrate this later by considering structures for the signature of the π-calculus.

Categories of structures as categories of algebras. If the category $\mathscr{C}$ has $\mathsf{Op}_{\mathbb{S}}$-indexed coproducts as well as the structure of the previous paragraph (finite products, an object, and an endofunctor) then a nominal algebraic signature $\mathbb{S}$ induces the following endofunctor on $\mathscr{C}$.

$$\Sigma_{\mathbb{S},\mathscr{C}} X = \coprod_{\mathsf{op}\in\mathsf{Op}_{\mathbb{S}}} \left(\mathcal{N}^{\mathrm{arn}(\mathsf{op})} \times \prod_{j\in[1,\mathrm{art}(\mathsf{op})]} [\mathcal{N}]^{\mathrm{dep}_j(\mathsf{op})} X \right) \tag{7.4.3}$$

The category of $\Sigma_{\mathbb{S},\mathscr{C}}$-algebras is isomorphic to the category of $\mathbb{S}$-structures in $\mathscr{C}$.

Following the developments of Section 6.1.1, if a free monad on $\Sigma_{\mathbb{S},\mathscr{C}}$ exists then it is denoted $\mathbf{T}_{\mathbb{S},\mathscr{C}} = (T_{\mathbb{S},\mathscr{C}}, \eta_{\mathbb{S},\mathscr{C}}, \eta_{\mathbb{S},\mathscr{C}})$.

Example: Syntax of the π-calculus up-to-α-equivalence. A structure for the π-calculus signature $\mathbb{PI}$ in **NomSub** is a nominal substitution X together with interpretations of the operators, including, for instance, an input operator as a nominal substitution homomorphism

$$[\![\mathsf{inp}]\!]_X : \mathcal{N} \times [\mathcal{N}]X \to X \quad ,$$

an output operator as a nominal substitution homomorphism

$$[\![\mathsf{out}]\!]_X : \mathcal{N} \times \mathcal{N} \times X \to X \quad ,$$

and a restriction operator as a nominal substitution homomorphism

$$[\![\mathsf{restrct}]\!]_X : [\mathcal{N}]X \to X \quad .$$

In this way the signature induces an endofunctor $\Sigma_{\mathbb{PI},\mathbf{NomSub}}$ on **NomSub**, and, indeed, the free monad $\mathbf{T}_{\mathbb{PI},\mathbf{NomSub}}$ on $\Sigma_{\mathbb{PI},\mathbf{NomSub}}$ exists. In a similar way one can consider $\mathbb{PI}$-structures in **Nom**, and we have an endofunctor $\Sigma_{\mathbb{PI},\mathbf{Nom}}$ and a monad $\mathbf{T}_{\mathbb{PI},\mathbf{Nom}}$ on **Nom**. For example, the nominal set $T_{\mathbb{PI},\mathbf{Nom}}\emptyset$ is the nominal set X_π of π-calculus terms up-to α-equivalence, introduced in equation 7.1.1, and the nominal substitution $T_{\mathbb{PI},\mathbf{NomSub}}\emptyset$ is the nominal substitution of π-calculus terms, as suggested at the beginning of Section 7.3.

Congruence in Nom and NomSub. We remark that the notions of congruence introduced in Section 6.2.1 specialise to the usual, intuitive notions of congruence for syntax with variable binding: closure under all contexts, including those that may capture variables. (For the π-calculus, this notion is considered by Sangiorgi and Walker [2001, Defn. 1.2.2], although their care over non-degeneracy is not relevant here.)

Example: Raw syntax of the π-calculus. We can consider $\mathbb{PI}$-structures in the category of sets as follows. We will interpret the sort of names as a set $\mathtt{N}$ of name *metavariables*; from this set we can derive an endofunctor $[\mathtt{N}] : \mathbf{Set} \to \mathbf{Set}$ given by $[\mathtt{N}]X = \mathtt{N} \times X$. We will write $\langle c \rangle\, x$ for an element $(c, x) \in [\mathtt{N}]X$. This is merely suggestive notation; there are no quotients involved.

We write $\mathbf{Set}_{\mathtt{N}}$ for the category of sets, with the understanding that $\mathtt{N}$ is the object of names, and $[\mathtt{N}]$ is the binding endofunctor.

There is a free monad $\mathbf{T}_{\mathbb{PI},\mathbf{Set}_{\mathtt{N}}}$ for the endofunctor $\Sigma_{\mathbb{PI},\mathbf{Set}_{\mathtt{N}}}$. For each set X, we have the set $T_{\mathbb{PI},\mathbf{Set}_{\mathtt{N}}}X$ of π-calculus terms with name variables from $\mathtt{N}$, and free term variables in X; the terms in $T_{\mathbb{PI},\mathbf{Set}_{\mathtt{N}}}X$ are *not* quotiented by α-equivalence.

Model categories and morphisms of endofunctors. A *nominal $\mathbb{S}$-model category* is a category $\mathscr{C}$ with finite products, $\mathsf{Op}_{\mathbb{S}}$-indexed coproducts, a distinguished object $\mathcal{N}$, and a distinguished endofunctor $[\mathcal{N}]$. (Above, we have been abusing notation by writing $\mathscr{C}$ for the structure $(\mathscr{C}, \mathcal{N}, [\mathcal{N}])$.)

A morphism between nominal $\mathbb{S}$-model categories

$$(\mathscr{C}, \mathcal{N}_{\mathscr{C}}, [\mathcal{N}_{\mathscr{C}}]) \to (\mathscr{D}, \mathcal{N}_{\mathscr{D}}, [\mathcal{N}_{\mathscr{D}}])$$

is given by a triple (F, f, ϕ), where $F : \mathscr{C} \to \mathscr{D}$ is a functor that preserves finite products, $f : \mathcal{N}_{\mathscr{D}} \to F(\mathcal{N}_{\mathscr{C}})$ is a morphism in $\mathscr{D}$, and $\phi : [\mathcal{N}_{\mathscr{D}}]F \to F[\mathcal{N}_{\mathscr{C}}]$ is a natural transformation between functors in $[\mathscr{C}, \mathscr{D}]$. The situation here is analogous to the situation at the end of Section 6.1.2: these data induce a natural transformation $(F, f, \phi)^* : \Sigma_{\mathbb{S},\mathscr{D}}F \to F\Sigma_{\mathbb{S},\mathscr{C}}$ in an obvious way, and so we have a morphism of endofunctors $(\mathscr{C}, \Sigma_{\mathbb{S},\mathscr{C}}) \to (\mathscr{D}, \Sigma_{\mathbb{S},\mathscr{D}})$. According to Section 6.1.3, if free monads on $\Sigma_{\mathbb{S},\mathscr{C}}$ and $\Sigma_{\mathbb{S},\mathscr{D}}$ exist then a monad morphism $(\mathscr{C}, \mathbf{T}_{\mathbb{S},\mathscr{C}}) \to (\mathscr{D}, \mathbf{T}_{\mathbb{S},\mathscr{D}})$ is induced.

Note that if $F : \mathscr{C} \to \mathscr{D}$ preserves coproducts, and $f : \mathcal{N}_{\mathscr{D}} \to F(\mathcal{N}_{\mathscr{C}})$ is an isomorphism, and $F_{[\mathcal{N}]} : [\mathcal{N}_{\mathscr{D}}]F \to F[\mathcal{N}_{\mathscr{C}}]$ is a natural isomorphism, then the induced natural transformation $(F, f, \phi)^* : \Sigma_{\mathbb{S},\mathscr{D}}F \to F\Sigma_{\mathbb{S},\mathscr{C}}$ is an isomorphism. In the terminology of Section 6.1.2, (F, f, ϕ) defines a *lifting* of the endofunctor $\Sigma_{\mathbb{S},\mathscr{D}}$ along the functor $F : \mathscr{C} \to \mathscr{D}$.

The 2-categorical situation is as follows. We write $\mathscr{M}_{\mathbb{S}}$ for the 2-category of nominal $\mathbb{S}$-model categories, with objects and morphisms as introduced above; a 2-cell between morphisms

$$(F, f, \phi), (G, g, \gamma) : (\mathscr{C}, \mathcal{N}_{\mathscr{C}}, [\mathcal{N}_{\mathscr{C}}]) \to (\mathscr{D}, \mathcal{N}_{\mathscr{D}}, [\mathcal{N}_{\mathscr{D}}])$$

is a natural transformation $\alpha : F \to G$ making the following diagrams commute.

$$\begin{array}{ccc} \mathcal{N}_{\mathscr{D}} & \xrightarrow{f} & F(\mathcal{N}_{\mathscr{C}}) \\ & {\scriptstyle g}\searrow & \downarrow{\scriptstyle \alpha\mathcal{N}_{\mathscr{C}}} \\ & & G\mathcal{N}_{\mathscr{C}} \end{array} \qquad \begin{array}{ccc} [\mathcal{N}_{\mathscr{D}}]F & \xrightarrow{\phi} & F[\mathcal{N}_{\mathscr{C}}] \\ {\scriptstyle [\mathcal{N}_{\mathscr{D}}]\alpha}\downarrow & & \downarrow{\scriptstyle \alpha[\mathcal{N}_{\mathscr{C}}]} \\ [\mathcal{N}_{\mathscr{D}}]G & \xrightarrow[\gamma]{} & G[\mathcal{N}_{\mathscr{C}}] \end{array}$$

Then the construction of equation 7.4.3 extends to a 2-functor $\mathscr{M}_{\mathbb{S}} \to \mathsf{Endo}$.

Example: Lifting π-calculus syntax from Nom to NomSub. As discussed in Section 7.3.1, the forgetful functor **NomSub** $\to$ **Nom** preserves limits and colimits. It is clear that it also preserves the object of names and the binding endofunctor. Thus we know that the endofunctor $\Sigma_{\mathbb{PI},\mathbf{NomSub}}$ on **NomSub** is a lifting, along the forgetful functor **NomSub** $\to$ **Nom**, of the endofunctor $\Sigma_{\mathbb{PI},\mathbf{Nom}}$ on **Nom**.

Since the forgetful functor **NomSub** → **Nom** has a right adjoint, we know from Prop. 6.1.4(1) that the induced morphism $(\mathbf{NomSub}, \Sigma_{\mathbb{PI},\mathbf{NomSub}}) \to (\mathbf{Nom}, \Sigma_{\mathbb{PI},\mathbf{Nom}})$ of endofunctors has a right adjoint in Endo. The 2-functoriality of the free algebra construction ensures that the induced morphism of monads $(\mathbf{NomSub}, \mathbf{T}_{\mathbb{PI},\mathbf{NomSub}}) \to (\mathbf{Nom}, \mathbf{T}_{\mathbb{PI},\mathbf{Nom}})$ has a right adjoint in Monad; *i.e.* that the monad $\mathbf{T}_{\mathbb{PI},\mathbf{NomSub}}$ on **NomSub** is a lifting, along the forgetful functor **NomSub** → **Nom**, of the monad $\mathbf{T}_{\mathbb{PI},\mathbf{Nom}}$ on **Nom**.

Of course, this phenomenon is not specific to the π-calculus, and similar results hold for every signature $\mathbb{S}$.

Example: Instantiating name metavariables. Let $\mathtt{N}$ be a set of name metavariables. Consider a function $\mathscr{V} : \mathtt{N} \to \mathcal{N}$ between sets, to be thought of as a valuation of the name metavariables as names. As remarked in Section 7.1, the forgetful functor $U : \mathbf{Nom} \to \mathbf{Set}$ preserves finite limits. The valuation function supplies a function $\mathtt{N} \to U(\mathcal{N})$ and so induces a natural transformation

$$[\mathtt{N}](U(-)) = \mathtt{N} \times U(-) \xrightarrow{\mathscr{V}\times \mathsf{id}_U} U(\mathcal{N}) \times U(-) \cong U(\mathcal{N} \times (-)) \to U([\mathcal{N}_{\mathbf{Nom}}](-))$$

between functors **Nom** → **Set**. Here, the rightmost morphism is the quotient arising from the binding construction in nominal sets.

In this way the valuation function $\mathscr{V} : \mathtt{N} \to \mathcal{N}$ induces a morphism of endofunctors $(\mathbf{Nom}, \Sigma_{\mathbb{PI},\mathbf{Nom}}) \to (\mathbf{Set}, \Sigma_{\mathbb{PI},\mathbf{Set}_{\mathtt{N}}})$, and hence also a monad morphism $(\mathbf{Nom}, \mathbf{T}_{\mathbb{PI},\mathbf{Nom}}) \to (\mathbf{Set}, \mathbf{T}_{\mathbb{PI},\mathbf{Set}_{\mathtt{N}}})$. This can be regarded as an operation transforming π-calculus expressions in raw syntax with name metavariables into π-calculus expressions (involving names) up-to α-equivalence.

7.5 Nominal transition systems

We now investigate how nominal logic can be used to axiomatise labelled transition systems for name-passing. Thus we arrive at an alternative to the indexed labelled transition systems studied in Sections 3.3, 3.4.3 and 4.4.2. We introduce a notion of nominal labelled transition system, and prove a correspondence with **I**-IL$_e$TSs over sheaves in **Sh**(**I**). We then briefly explain how the uniform input behaviour studied in Section 3.4.3 can be axiomatised in nominal logic using the theory of nominal substitutions.

An axiomatisation of ground labelled transition systems is provided next, and we have a correspondence with the B_g-coalgebras of Section 3.2.2. We conclude this section by considering notions of ground and wide open bisimulation in this setting.

Nominal early labelled transition systems.

Definition 7.5.1. The nominal logic theory of *nominal early labelled transition systems* ($И_e$-LTSs) has one ground sort X and three relation symbols:

- an *input transition* relation symbol $\left(\xrightarrow{-?-}\right)$ with arity $\mathsf{X}, \mathsf{N}, \mathsf{N}, \mathsf{X}$;
- an *output transition* relation symbol $\left(\xrightarrow{-!-}\right)$ with arity $\mathsf{X}, \mathsf{N}, \mathsf{N}, \mathsf{X}$;
- a *silent transiton* relation symbol $\left(\xrightarrow{\tau}\right)$ with arity X, X

subject to Axioms $И_e1$–$И_e3$ in Figure 7.1a.

A model of this theory (in **Nom**) is given by a nominal 'carrier' set X together with three equivariant relations

$$\left(\xrightarrow{-?-}\right) \subseteq X \times \mathcal{N} \times \mathcal{N} \times X \qquad \left(\xrightarrow{-!-}\right) \subseteq X \times \mathcal{N} \times \mathcal{N} \times X \qquad \left(\xrightarrow{\tau}\right) \subseteq X \times X$$

$И_e1$. The channel is known.

$$\forall x, y : \mathsf{X},\ c, d : \mathsf{N}. \qquad \left(x \xrightarrow{c?d} y \implies \neg(c \# x)\right) \wedge \left(x \xrightarrow{c!d} y \implies \neg(c \# x)\right)$$

$И_e2$. Names in the derivative depend only on names in the source and communication data.

$$\forall x, y : \mathsf{X},\ a, c, d : \mathsf{N}. \qquad \left(x \xrightarrow{c?d} y \wedge a \# (x, d) \implies a \# y\right) \wedge \left(x \xrightarrow{c!d} y \wedge a \# (x, d) \implies a \# y\right) \wedge \left(x \xrightarrow{\tau} y \wedge a \# x \implies a \# y\right)$$

$И_e3$. If one name can be input then so can any other.

$$\forall x, y : \mathsf{X},\ c, d, d' : \mathsf{N}. \qquad x \xrightarrow{c?d} y \implies \exists y' : \mathsf{X}.\ x \xrightarrow{c?d'} y'$$

Figure 7.1a: Axioms for the nominal logic theory of nominal early labelled transition systems.

$И_e1$. The channel is known.

$$\forall x, y \in X,\ c, d \in \mathcal{N}. \qquad \left(x \xrightarrow{c?d} y \implies c \in \mathsf{supp}_X(x)\right) \wedge \left(x \xrightarrow{c!d} y \implies c \in \mathsf{supp}_X(x)\right)$$

$И_e2$. Names in the derivative depend only on names in the source and communication data.

$$\forall x, y \in X,\ c, d \in \mathcal{N}. \qquad \left(x \xrightarrow{c?d} y \implies \mathsf{supp}_X(y) \subseteq \mathsf{supp}_X(x) \cup \{d\}\right) \wedge \left(x \xrightarrow{c!d} y \implies \mathsf{supp}_X(y) \subseteq \mathsf{supp}_X(x) \cup \{d\}\right) \wedge \left(x \xrightarrow{\tau} y \implies \mathsf{supp}_X(y) \subseteq \mathsf{supp}_X(x)\right)$$

$И_e3$. If one name can be input then so can any other.

$$\forall x, y \in X,\ c, d, d' \in \mathcal{N}. \qquad x \xrightarrow{c?d} y \implies \exists y' \in X.\ x \xrightarrow{c?d'} y'$$

Figure 7.1b: Requirements on a nominal early labelled transition system with nominal carrier set X.

that together satisfy the interpretations of Axioms $И_e1$–$И_e3$. For reference, these interpretations are given explicitly in Figure 7.1b.

Relating $И_e$-LTSs with I-IL$_e$TSs. In Section 7.1.4 we gave a construction that exhibits the Schanuel topos as equivalent to the category of nominal sets. By using this construction, we now establish a correspondence with some of the **I**-IL$_e$TSs of Section 3.3. Consider a $И_e$-LTS $\longrightarrow$ over X. An **I**-IL$_e$TS $\longrightarrow_{\mathbf{I}}$ over the sheaf corresponding to X (as in Section 7.1.4) is induced as follows.

$$\begin{aligned}
&\text{If } x \xrightarrow{c?d} y \text{ and } C \text{ supports } x \text{ and } C \cup \{d\} \text{ supports } y\\
&\qquad \text{then } C \vdash x \xrightarrow{c?d}_{\mathbf{I}} C \cup \{d\} \vdash y.\\
&\text{If } x \xrightarrow{c!d} y \text{ and } d \in \mathsf{supp}_X(x) \text{ and } C \text{ supports } x \text{ and } C \text{ supports } y\\
&\qquad \text{then } C \vdash x \xrightarrow{c!d}_{\mathbf{I}} C \vdash y.\\
&\text{If } x \xrightarrow{c!d} y \text{ and } d \notin C \text{ and } C \text{ supports } x \text{ and } C \cup \{d\} \text{ supports } y\\
&\qquad \text{then } C \vdash x \xrightarrow{c!d}_{\mathbf{I}} C \cup \{d\} \vdash y.\\
&\text{If } x \xrightarrow{\tau} y \text{ and } C \text{ supports } x \text{ and } C \text{ supports } y\\
&\qquad \text{then } C \vdash x \xrightarrow{\tau}_{\mathbf{I}} C \vdash y.
\end{aligned} \tag{7.5.2}$$

Theorem 7.5.3. *The mapping described in* (7.5.2) *takes a* $И_e$-LTS *to an* **I**-IL$_e$TS *satisfying Axioms* **I**1–**I**6. *For each nominal set* X*, the mapping defines a bijective correspondence between* $И_e$-LTS*s over* X *and* **I**-IL$_e$TS*s over the sheaf in* $\mathbf{Sh}(\mathbf{I})$ *corresponding to* X*, that satisfy Axioms* **I**1–**I**6.

A proof of this theorem is provided in Appendix 7.C.

Nominal transition systems for the π-calculus. The early transition system for the π-calculus, recalled in Section 3.1.1, provides a model of the theory of $И_e$-LTSs, with the nominal set X_π of π-calculus terms (7.1.1) as the carrier. Equivariance of these relations corresponds to Prop. 3.1.12(1). Axioms $И_e1$ and $И_e2$ are Prop. 3.1.2, and Axiom $И_e3$ is Corollary 3.1.4(1).

Uniform input. We form the theory of *nominal labelled transition systems with uniform input* by combining the theory of nominal labelled transition systems with the theory of nominal substitutions (Section 7.3), and adding the following axiom.

$И_e4$. Uniform input.

$$\forall x, y : \mathsf{X},\ c, d, d' : \mathsf{N}.\ \left(d \# x \wedge x \xrightarrow{c?d} y\right) \implies x \xrightarrow{c?d'} [d'/d]y$$

For the $И_e$-LTS for the π-calculus, Axiom $И_e4$ corresponds to Prop. 3.1.3. Using the construction of Theorem 7.5.3, we have the following result.

Proposition 7.5.4. *There is a bijective correspondence between nominal labelled transition systems with uniform input over a nominal substitution* X*, and* **F**-IL$_e$TS*s over the sheaf in* $\mathbf{Sh}(\mathbf{F})$ *corresponding to* X*, that satisfy Axioms* **I**1–**I**6 *and* **F**2'. □

Nominal ground labelled transition systems. We now axiomatise a class of ground labelled transition systems.

Definition 7.5.5. The nominal theory of *nominal ground labelled transition systems (*$И_g$-LTS*s)* has one ground sort X and four relation symbols:

- a *bound input transition* relation symbol $\left(\overset{-?(-)}{\longrightarrow}\right)$ with arity $\mathsf{X}, \mathsf{N}, \mathsf{N}, \mathsf{X}$;
- an *output transition* relation symbol $\left(\overset{-!-}{\longrightarrow}\right)$ with arity $\mathsf{X}, \mathsf{N}, \mathsf{N}, \mathsf{X}$;
- a *bound output transition* relation symbol $\left(\overset{-!(-)}{\longrightarrow}\right)$ with arity $\mathsf{X}, \mathsf{N}, \mathsf{N}, \mathsf{X}$;
- a *silent transiton* relation symbol $\left(\overset{\tau}{\longrightarrow}\right)$ with arity X, X

subject to Axioms $И_g1$ and $И_g2$ in Figure 7.2.

$И_g1$. The channel and free data are known, while binding data is fresh.

$$\begin{aligned}\forall x, y : \mathsf{X},\ c, d : \mathsf{N}. \qquad & \left(x \overset{c?(d)}{\longrightarrow} y \implies \neg(c \# x) \wedge (d \# x)\right) \\ \wedge\ & \left(x \overset{c!d}{\longrightarrow} y \implies \neg(c \# x) \wedge \neg(d \# x)\right) \\ \wedge\ & \left(x \overset{c!(d)}{\longrightarrow} y \implies \neg(c \# x) \wedge (d \# x)\right)\end{aligned}$$

$И_g2$. Names in the derivative depend only on names in the source and communication data.

$$\begin{aligned}\forall x, y : \mathsf{X},\ a, c, d : \mathsf{N}. \qquad & \left(x \overset{c?(d)}{\longrightarrow} y \wedge a \#(x, d) \implies a \# y\right) \\ \wedge\ & \left(x \overset{c!d}{\longrightarrow} y \wedge a \# x \implies a \# y\right) \\ \wedge\ & \left(x \overset{c!(d)}{\longrightarrow} y \wedge a \#(x, d) \implies a \# y\right) \\ \wedge\ & \left(x \overset{\tau}{\longrightarrow} y \wedge a \# x \implies a \# y\right)\end{aligned}$$

Figure 7.2: Axioms for the nominal logic theory of nominal ground labelled transition systems.

For a first example, we note that the ground transition system for the π-calculus, recalled in Section 3.1.2, provides a model of the theory of $И_g$-LTSs, with the nominal set X_π of π-calculus terms (7.1.1) as the carrier.

Every $И_e$-LTS induces a $И_g$-LTS, according to a straightforward procedure akin to (3.3.21). Conversely, a model of the combined theory of nominal substitutions and $И_g$-LTSs, induces an $И_e$-LTS with uniform input, via a process akin to (3.4.10). In this way, one establishes that models of the combined theory of nominal substitutions and $И_g$-LTSs are in bijective correspondence with $И_e$-LTSs with uniform input. In other words:

Proposition 7.5.6. *To give a* $И_g$-LTS *together with a nominal substitution structure on its carrier, is to give a coalgebra for the endofunctor* B_g *on* **Nom** *that is structured by the forgetful functor* **NomSub** $\to$ **Nom**. □

Bisimulation for nominal transition systems. We conclude this section by introducing notions of bisimulation for $И_g$-LTSs. This amounts to translating the definitions of Definition 3.3.4 to the nominal setting.

Definition 7.5.7. Consider structures $(X, \longrightarrow_X)$ and $(Y, \longrightarrow_Y)$ in **Nom** for the theory of $И_g$-LTSs, and let R be an equivariant relation between X and Y.

1. R is a *ground simulation* if, for all $x, x' \in X$, $y \in Y$, and $a, c, d \in \mathcal{N}$:
   - whenever $x\,R\,y$ and $x \xrightarrow{c?(a)}_X x'$, with $a \# (x, y)$, then we have $y' \in Y$ with $y \xrightarrow{c?(a)}_Y y'$ and $x'\,R\,y'$; and
   - whenever $x\,R\,y$ and $x \xrightarrow{c!d}_X x'$ then we have $y' \in Y$ with $y \xrightarrow{c!d}_Y y'$ and $x'\,R\,y'$; and
   - whenever $x\,R\,y$ and $x \xrightarrow{c!(a)}_X x'$, with $a \# (x, y)$, then we have $y' \in Y$ with $y \xrightarrow{c!(a)}_Y y'$ and $x'\,R\,y'$; and
   - whenever $x\,R\,y$ and $x \xrightarrow{\tau}_X x'$ then we have $y' \in Y$ with $y \xrightarrow{\tau}_Y y'$ and $x'\,R\,y'$.
2. R is a *ground bisimulation* if both R and R^{op} are ground simulations.
3. Suppose that X and Y are equipped with nominal substitution structures. Then an equivariant relation R between nominal sets X and Y is a *wide open bisimulation* if it is a ground bisimulation and it is closed under substitution, *i.e.*
$$\forall x \in X,\ y \in Y.\ \forall b, a \in \mathcal{N}.\ x\,R\,y \implies [b/a]x\ R\ [b/a]y \quad .$$
4. An element x of X is *ground bisimilar* to an element y of Y if there exists an (equivariant) ground bisimulation that relates them.
5. If X and Y are equipped with nominal substitution structures, then an element x of X is *wide open bisimilar* to an element y of Y if there exists an (equivariant) wide open bisimulation that relates them.

7.A Appendix to Chapter 7: Proof of Prop. 7.3.1

We now prove Proposition 7.3.1, from page 176:

Proposition 7.3.1. *The category of models for the alternative theory is isomorphic to the category* **NomSub**.

Proof notes. Starting with a conventional nominal substitution $\mathsf{sub} : \mathcal{N} \times [\mathcal{N}]X \to X$, one obtains a nominal substitution in the alternative presentation by precomposing with the quotient map $\mathcal{N} \times X \to [\mathcal{N}]X$. It is immediate that axioms NomSub-1–4 remain true, and easy to see that NomSub-0 is also true.

Starting with a model of the alternative presentation, that is, an equivariant function $\mathsf{sub} : \mathcal{N} \times \mathcal{N} \times X \to X$, we can find a conventional nominal substitution, $\mathsf{sub}' : \mathcal{N} \times [\mathcal{N}]X \to X$, such that

$$\forall a, b \in \mathcal{N}, x \in X.\ a \neq b \implies \mathsf{sub}'(b, \langle a \rangle x) = \mathsf{sub}(b, a, x) \tag{7.A.1}$$

because sub satisfies NomSub-0. This is the essence of Prop. 7.1.3(2). One must then verify that axioms NomSub-1–4 hold of sub'. Axiom NomSub-1 requires the most work. Pick $b, c \# (x, a)$, with $b \neq c$; then

$$\begin{aligned}
\mathsf{sub}'(a, \langle a \rangle x) &= \mathsf{sub}'(a, \langle b \rangle [a \leftrightarrow b] \bullet_X x) && \text{(defn. of } [\mathcal{N}]X\text{)} \\
&= \mathsf{sub}(a, b, [a \leftrightarrow b] \bullet_X x) && \text{(defn. of } \mathsf{sub}', \text{ (7.A.1))} \\
&= [a/b][a \leftrightarrow b] \bullet_X x && \text{(change notation)} \\
&= [a/c][a/b][a \leftrightarrow b] \bullet_X x && \text{(NomSub-2)} \\
&= [a/c][c/b][a \leftrightarrow b] \bullet_X x && \text{(NomSub-3)} \\
&= [a/c][c/a]x && \text{(NomSub-0, and equivariance)} \\
&= [a/c][a/a]x && \text{(NomSub-2)} \\
&= [a/c]x && \text{(NomSub-1)} \\
&= x && \text{(NomSub-2)}
\end{aligned}$$

To show that the rest of the axioms, NomSub-2–4, hold of sub', one has to break the axioms down into cases, depending on which names are disjoint. For example, to show axiom NomSub-2,

$$\forall a, b : \mathsf{N}.\ \forall x : \mathsf{X}.\ a \# x \implies [b/a]x = x \quad ,$$

we consider separately the cases where $a = b$, and where $a \neq b$. When $a = b$, axiom NomSub-2 follows immediately from NomSub-1; when $a \neq b$, we use the definition of sub' (7.A.1) together with the fact that NomSub-2 holds of sub. □

7.B Appendix to Chapter 7: Proof of Theorem 7.3.2

We now prove Theorem 7.3.2, from page 179:

Theorem 7.3.2. *The category* **NomSub** *is equivalent to* $\mathbf{Sh}(\mathbf{F})$.

Proof. We will show that the category of models in $\mathbf{Sh}(\mathbf{I})$ for the theory of nominal substitutions is isomorphic to the category of algebras for the monad $(U^{\mathbf{I}}_{\mathbf{F}} \circ (j^{\mathbf{I}}_{\mathbf{F}})_!)$ on $\mathbf{Sh}(\mathbf{I})$.

Let P be a sheaf in $\mathbf{Sh}(\mathbf{I})$. Given a $(U^{\mathbf{I}}_{\mathbf{F}} \circ (j^{\mathbf{I}}_{\mathbf{F}})_!)$-algebra structure $\alpha : U^{\mathbf{I}}_{\mathbf{F}}((j^{\mathbf{I}}_{\mathbf{F}})_!(P)) \to P$ for P, we define a nominal substitution structure $\mathsf{sub} : N \times N \times P \to P$ as follows. For each $C \in \mathbf{I}$, and each $a, b \in C$, $p \in P(C)$, we let

$$\mathsf{sub}_C(b, a, p) = \begin{cases} p & \text{if } a = b \\ \alpha_C[\mathsf{inj}_C\left([C - \{a\} \hookrightarrow C][b/a], p\right)] & \text{if } a \neq b. \end{cases}$$

(Here, as usual, $[b/a]$ denotes the function $C \to (C - \{a\})$ acting as the identity except that a is mapped to b.) To see that the family $\{\mathsf{sub}_C : C \times C \times P(C) \to P(C)\}_C$ is natural, we consider an injection $\iota : C \rightarrowtail D$ in $\mathbf{I}$, and $(b, a, p) \in (C \times C \times P(C))$. If we have $a = b$ then we must also have $\iota(a) = \iota(b)$, and so in this case it follows immediately that

$$[\iota]\left(\mathsf{sub}_C(b, a, p)\right) = \left(\mathsf{sub}_D(\iota(b), \iota(a), [\iota]p)\right) \quad .$$

For the case where $a \neq b$, then we must also have $\iota(a) \neq \iota(b)$, and in this case,

$$\begin{aligned}
[\iota]\left(\mathsf{sub}_C(b, a, p)\right) &= [\iota]\left(\alpha_C\left[\mathsf{inj}_C\left([C - \{a\} \hookrightarrow C][b/a], p\right)\right]\right) && (1)\\
&= \alpha_D[\iota]\left[\mathsf{inj}_C\left([C - \{a\} \hookrightarrow C][b/a], p\right)\right] && (2)\\
&= \alpha_D\left[\mathsf{inj}_C\left(\iota \circ [C - \{a\} \hookrightarrow C] \circ [b/a], p\right)\right] && (3)\\
&= \alpha_D\left[\mathsf{inj}_C\left([D - \{\iota(a)\} \hookrightarrow D] \circ [\iota(b)/\iota(a)] \circ \iota, p\right)\right] && (4)\\
&= \alpha_D\left[\mathsf{inj}_D\left([D - \{\iota(a)\} \hookrightarrow D] \circ [\iota(b)/\iota(a)], [\iota]p\right)\right] && (5)\\
&= \mathsf{sub}_D(\iota(b), \iota(a), [\iota]p) \quad . && (6)
\end{aligned}$$

Using: (1) defn. of sub_C; (2) nat. of α; (3) defn. of action of ι; (4) factorisation; (5) defn. of $\sim_D$; (6) defn. of sub_D.

We now explain why Axioms NomSub-0–4 hold of this structure. For Axiom NomSub-0, we consider $C \in \mathbf{I}$, and $(b, a, p) \in (C \times C \times P(C))$. Suppose that $b \neq a$; then

$$\begin{aligned}
\mathsf{sub}_C(b, a, p) &= \alpha_C\left[\mathsf{inj}_C\left([C - \{a\} \hookrightarrow C][b/a], p\right)\right]\\
&= \alpha_C\left([C - \{a\} \hookrightarrow C]\left[\mathsf{inj}_{C-\{a\}}\left([b/a], p\right)\right]\right)\\
&= [C - \{a\} \hookrightarrow C]\left(\alpha_{C-\{a\}}\left[\mathsf{inj}_{C-\{a\}}\left([b/a], p\right)\right]\right)
\end{aligned}$$

and hence $(C - \{a\})$ supports $\mathsf{sub}_C(b, a, p)$.

Axiom NomSub-1 follows immediately from the definition of sub. For Axiom NomSub-2, consider $C \in \mathbf{I}$, and $(b, a, p) \in (C \times C \times P(C))$. Suppose that $a \notin \mathsf{supp}(p)$. For the case where $a = b$, the result follows from axiom NomSub-1; for the case where $a \neq b$, we have

$$\begin{aligned}
\mathsf{sub}_C(b, a, p) &= \alpha_C\left[\mathsf{inj}_C\left([C - \{a\} \hookrightarrow C][b/a], p\right)\right] && (1)\\
&= \alpha_C\left[\mathsf{inj}_C\left([C - \{a\} \hookrightarrow C][b/a], [C - \{a\} \hookrightarrow C]\mathsf{seed}\left(p@C - \{a\}\right)\right)\right] && (2)\\
&= \alpha_C\left[\mathsf{inj}_C\left([C - \{a\} \hookrightarrow C][b/a][C - \{a\} \hookrightarrow C], \mathsf{seed}\left(p@C - \{a\}\right)\right)\right] && (3)\\
&= \alpha_C\left[\mathsf{inj}_C\left([C - \{a\} \hookrightarrow C], \mathsf{seed}\left(p@(C - \{a\})\right)\right)\right] && (4)\\
&= \alpha_C\left[\mathsf{inj}_C\left(\mathsf{id}_C, p\right)\right] && (5)\\
&= p \quad . && (6)
\end{aligned}$$

Using: (1) defn. of sub_C; (2) since $a \notin \mathsf{supp}(p)$; (3) defn. of $\sim_C$; (4) equality of functions; (5) defn. of $\sim_C$; (6) unit law for α.

To show Axiom NomSub-3, we consider $C \in \mathbf{I}$ and $a, b, c \in C$, together with $p \in P(C)$. We must show that $\mathsf{sub}_C(c, b, \mathsf{sub}_C(b, a, p)) = \mathsf{sub}_C(c, b, \mathsf{sub}_C(c, a, p))$. For the case where $b = c$, this follows immediately from the definition of sub. When $b \neq c$ but $a = b$, the result follows from Axioms

NomSub-0–2, already established. When $a \neq b \neq c$ yet $a = c$, we have:

$$\begin{aligned}
&\mathsf{sub}_C(c, b, \mathsf{sub}_C(b, a, p)) \\
&\quad = \mathsf{sub}_C(a, b, \mathsf{sub}_C(b, a, p)) && (1)\\
&\quad = \alpha_C\left[\mathsf{inj}_C\left([C - \{b\} \hookrightarrow C][a/b], \alpha_C\left[\mathsf{inj}_C\left([C - \{a\} \hookrightarrow C][b/a], p\right)\right]\right)\right] && (2)\\
&\quad = \left(\alpha \circ (U_{\mathbf{F}}^{\mathbf{I}}((j_{\mathbf{F}}^{\mathbf{I}})_!(\alpha)))\right)_C\left[\mathsf{inj}_C\begin{pmatrix}[C - \{b\} \hookrightarrow C][a/b], \\ \left[\mathsf{inj}_C\left([C - \{a\} \hookrightarrow C][b/a], p\right)\right]\end{pmatrix}\right] && (3)\\
&\quad = \alpha_C\left[\mathsf{inj}_C\left([C - \{b\} \hookrightarrow C][a/b][C - \{a\} \hookrightarrow C][b/a], p\right)\right] && (4)\\
&\quad = \alpha_C\left[\mathsf{inj}_C\left([C - \{b\} \hookrightarrow C][a/b], p\right)\right] && (5)\\
&\quad = \mathsf{sub}_C(a, b, p) && (6)\\
&\quad = \mathsf{sub}_C(c, b, \mathsf{sub}_C(c, a, p)) \quad . && (7)
\end{aligned}$$

Using: (1) since $a = c$; (2) defn. of sub_C; (3) defn. of action of $(U_{\mathbf{F}}^{\mathbf{I}} \circ (j_{\mathbf{F}}^{\mathbf{I}})_!)$; (4) mult. law for α; (5) equality of functions; (6) defn. of sub_C; (7) since $a = c$, and using NomSub-1.

Finally, in the case when a, b and c are all distinct, we have:

$$\begin{aligned}
&\mathsf{sub}_C(c, b, \mathsf{sub}_C(b, a, p)) \\
&\quad = \alpha_C\left[\mathsf{inj}_C\left([C - \{b\} \hookrightarrow C][c/b], \alpha_C\left[\mathsf{inj}_C\left([C - \{a\} \hookrightarrow C][b/a], p\right)\right]\right)\right] && (1)\\
&\quad = \left(\alpha \circ (U_{\mathbf{F}}^{\mathbf{I}}((j_{\mathbf{F}}^{\mathbf{I}})_!(\alpha)))\right)_C\left[\mathsf{inj}_C\begin{pmatrix}[C - \{b\} \hookrightarrow C][c/b], \\ \left[\mathsf{inj}_C\left([C - \{a\} \hookrightarrow C][b/a], p\right)\right]\end{pmatrix}\right] && (2)\\
&\quad = \alpha_C\left[\mathsf{inj}_C\left([C - \{b\} \hookrightarrow C][c/b][C - \{a\} \hookrightarrow C][b/a], p\right)\right] && (3)\\
&\quad = \alpha_C\left[\mathsf{inj}_C\left([C - \{b\} \hookrightarrow C][c/b][C - \{a\} \hookrightarrow C][c/a], p\right)\right] && (4)\\
&\quad = \left(\alpha \circ (U_{\mathbf{F}}^{\mathbf{I}}((j_{\mathbf{F}}^{\mathbf{I}})_!(\alpha)))\right)_C\left[\mathsf{inj}_C\begin{pmatrix}[C - \{b\} \hookrightarrow C][c/b], \\ \left[\mathsf{inj}_C\left([C - \{a\} \hookrightarrow C][c/a], p\right)\right]\end{pmatrix}\right] && (5)\\
&\quad = \alpha_C\left[\mathsf{inj}_C\left([C - \{b\} \hookrightarrow C][c/b], \alpha_C\left[\mathsf{inj}_C\left([C - \{a\} \hookrightarrow C][c/a], p\right)\right]\right)\right] && (6)\\
&\quad = \mathsf{sub}_C(c, b, \mathsf{sub}_C(c, a, p)) \quad . && (7)
\end{aligned}$$

Using: (1) defn. of sub_C; (2) defn. of action of $(U_{\mathbf{F}}^{\mathbf{I}} \circ (j_{\mathbf{F}}^{\mathbf{I}})_!)$; (3) mult. law for α; (4) equality of functions; (5) mult. law for α; (6) defn. of action of $(U_{\mathbf{F}}^{\mathbf{I}} \circ (j_{\mathbf{F}}^{\mathbf{I}})_!)$; (7) defn. of sub_C.

Axiom NomSub-4 is verified in a similar manner. Thus a $(U_{\mathbf{F}}^{\mathbf{I}} \circ (j_{\mathbf{F}}^{\mathbf{I}})_!)$-algebra induces a nominal substitution.

Given a nominal substitution $\mathsf{sub} : N \times N \times P \to P$ in $\mathbf{Sh}(\mathbf{I})$, we define a $(U_{\mathbf{F}}^{\mathbf{I}} \circ (j_{\mathbf{F}}^{\mathbf{I}})_!)$-algebra structure $\alpha : U_{\mathbf{F}}^{\mathbf{I}}((j_{\mathbf{F}}^{\mathbf{I}})_!(P)) \to P$ as follows. Consider some $C \in \mathbf{I}$, and consider $[\mathsf{inj}_D(f, p)] \in U_{\mathbf{F}}^{\mathbf{I}}((j_{\mathbf{F}}^{\mathbf{I}})_!(P))(C)$. By definition of $\sim_C$, we can assume that D and C are disjoint. We now pick an ordering of D, say $D = \left\{d_1, \ldots, d_{|D|}\right\}$, and we consider the following element of $P(C \cup D)$:

$$\mathsf{sub}_{C \cup D}\left(f\, d_{|D|}, d_{|D|}, \left(\ldots \mathsf{sub}_{C \cup D}\left(f\, d_1, d_1, [D \hookrightarrow C \cup D]p\right) \ldots\right)\right) \quad .$$

By repeated use of NomSub-0, one concludes that C supports this element, and so we let

$$\alpha_C\left[\mathsf{inj}_D(f, p)\right] \;=\; \mathsf{seed}\left(\mathsf{sub}_{C \cup D}\left(f\, d_{|D|}, d_{|D|}, \left(\ldots \mathsf{sub}_{C \cup D}\left(f\, d_1, d_1, [D \hookrightarrow C \cup D]p\right) \ldots\right)\right)@C\right) \; .$$

It follows from NomSub-4 that this definition is independent of the ordering of D.

We now show that the function $\alpha_C : U_{\mathbf{F}}^{\mathbf{I}}((j_{\mathbf{F}}^{\mathbf{I}})_!(P))(C) \to P(C)$ respects equivalence classes. Every injection decomposes into a bijection and an inclusion, and we will treat these two special types of injection separately.

Consider $D, D' \in \mathbf{I}$, both disjoint from C, and consider a bijection $\beta : D \xrightarrow{\sim} D'$, a function $f : D' \to C$ and an element $p \in P(D)$. Then, picking an ordering of D, say $D = \left\{d_1, \ldots d_{|D|}\right\}$, we

have

$$\begin{aligned}
\alpha_C\left[\mathsf{inj}_D(f\circ\beta,p)\right] &= \mathsf{seed}\left(\begin{array}{l}\mathsf{sub}_{C\cup D}(f\,\beta d_{|D|},d_{|D|},\dots\\ \quad \mathsf{sub}_{C\cup D}(f\,\beta d_1,d_1,[D\hookrightarrow C\cup D]p)\dots)\ @C\end{array}\right) && (1)\\
&= \mathsf{seed}\left([\mathsf{id}_C\uplus\beta]\left(\begin{array}{l}\mathsf{sub}_{C\cup D}(f\,\beta d_{|D|},d_{|D|},\dots\\ \quad \mathsf{sub}_{C\cup D}(f\,\beta d_1,d_1,[D\hookrightarrow C\cup D]p)\dots)\end{array}\right)\ @C\right) && (2)\\
&= \mathsf{seed}\left(\begin{array}{l}\mathsf{sub}_{C\cup D'}(f\,\beta d_{|D|},\beta d_{|D|},\dots\\ \quad \mathsf{sub}_{C\cup D'}(f\,\beta d_1,\beta d_1,[D'\hookrightarrow C\cup D][\beta]p)\dots)\ @C\end{array}\right) && (3)\\
&= \alpha_C\left[\mathsf{inj}_{D'}(f,[\beta]p)\right]\quad . && (4)
\end{aligned}$$

Using: (1) defn. of α_C; (2) since C supports the expression, and $[\mathsf{id}_C\uplus\beta]:C\cup D\xrightarrow{\sim}C\cup D'$ acts as identity on C; (3) nat. of sub; (4) defn. of α_C, bearing in mind that, since β is a bijection, we have $D'=\left\{\beta d_1,\dots\beta d_{|D|}\right\}$.

To deal with the case of inclusions, we consider three disjoint sets of names, $C,D,E\in\mathbf{I}$, a function $f:D\cup E\to C$, and an element $p\in P(D)$. We will show that, according to the definition above,

$$\alpha_C\left[\mathsf{inj}_D(f\circ[D\hookrightarrow D\cup E],p)\right]=\alpha_C\left[\mathsf{inj}_{D\cup E}(f,[D\hookrightarrow D\cup E]p)\right]\quad .$$

Indeed, pick orderings of D and E, say $D=\left\{d_1,\dots,d_{|D|}\right\}$, $E=\left\{e_1,\dots,e_{|E|}\right\}$, and then

$$\begin{aligned}
&\alpha_C\left[\mathsf{inj}_D(f\circ[D\hookrightarrow D\cup E],p)\right]\\
&\quad= \mathsf{seed}\left(\begin{array}{l}\mathsf{sub}_{C\cup D\cup E}(f\,d_{|D|},d_{|D|},\dots\\ \quad \mathsf{sub}_{C\cup D\cup E}(f\,d_1,d_1,[D\hookrightarrow C\cup D\cup E]p)\dots)\ @C\end{array}\right) && (1)\\
&\quad= \mathsf{seed}\left(\begin{array}{l}\mathsf{sub}_{C\cup D\cup E}(f\,e_{|E|},e_{|E|},\dots\\ \quad \mathsf{sub}_{C\cup D\cup E}(f\,e_1,e_1,\\ \quad\quad \mathsf{sub}_{C\cup D\cup E}(f\,d_{|D|},d_{|D|},\dots\\ \quad\quad\quad \mathsf{sub}_{C\cup D\cup E}(f\,d_1,d_1,[D\hookrightarrow C\cup D\cup E]p)\dots))\dots)\ @C\end{array}\right) && (2)\\
&\quad= \alpha_C\left[\mathsf{inj}_D(f,[D\hookrightarrow D\cup E]p)\right]\quad . && (3)
\end{aligned}$$

Using: (1) defn. of α_C; (2) repeated use of NomSub-2; (3) defn. of α_C.

The family $\left\{\alpha_C:U^{\mathbf{I}}_{\mathbf{F}}((j^{\mathbf{I}}_{\mathbf{F}})_!(P))(C)\to P(C)\right\}_{C\in\mathbf{I}}$ is natural in C, since sub is natural.

We now show that $\alpha:U^{\mathbf{I}}_{\mathbf{F}}((j^{\mathbf{I}}_{\mathbf{F}})_!(P))\to P$ is an algebra for the monad $(U^{\mathbf{I}}_{\mathbf{F}}\circ(j^{\mathbf{I}}_{\mathbf{F}})_!)$, by showing that it satisfies the unit and multiplication laws.

For the unit law, consider $C\in\mathbf{I}$, and $p\in P(C)$. We must show that

$$\alpha_C\left[\mathsf{inj}_C(\mathsf{id}_C,p)\right]=p\quad .$$

We do this by considering the following sequence of equations. Pick some set $D\in\mathbf{I}$ disjoint from C, for which there is a bijection $\beta:D\xrightarrow{\sim}C$, and suppose that $D=\left\{d_1,\dots,d_{|D|}\right\}$. Then

$$
\begin{aligned}
\alpha_C\left[\mathsf{inj}_C(\mathsf{id}_C, p)\right] &= \alpha_C\left[\mathsf{inj}_D(\beta, [\beta^{-1}]p)\right] && (1)\\
&= \mathsf{seed}\left(\begin{array}{l}\mathsf{sub}_{C\cup D}(\beta d_{|D|}, d_{|D|}, \ldots\\ \quad \mathsf{sub}_{C\cup D}\left(\beta d_1, d_1, [D \hookrightarrow C\cup D][\beta^{-1}]p\right)\ldots)\ @C\end{array}\right) && (2)\\
&= \mathsf{seed}\left(\left([\beta d_{|D|} \leftrightarrow d_{|D|}]\ldots[\beta d_1 \leftrightarrow d_1][D \hookrightarrow C\cup D][\beta^{-1}]p\right)@C\right) && (3)\\
&= \mathsf{seed}\left(\left([\beta^{-1} \uplus \beta][D \hookrightarrow C\cup D][\beta^{-1}]p\right)@C\right) && (4)\\
&= \mathsf{seed}\left(\left([C \hookrightarrow C\cup D][\beta][\beta^{-1}]p\right)@C\right) && (5)\\
&= \mathsf{seed}\left(\left([C \hookrightarrow C\cup D]p\right)@C\right) && (6)\\
&= p \quad . && (7)
\end{aligned}
$$

Using: (1) defn. of $\sim_C$; (2) defn. of α_C; (3) repeated use of NomSub-1; (4–6) manipulating functions; (7) defn. of seed.

For the multiplication law, consider distinct $C, D, E \in \mathbf{I}$, and $f : E \to D$, $g : D \to C$ in $\mathbf{F}$, and $p \in P(E)$. We must show that

$$\alpha_C\left[\mathsf{inj}_E(g \circ f, p)\right] = \alpha_C\left[\mathsf{inj}_D\left(g, \alpha_D\left[\mathsf{inj}_E(f, p)\right]\right)\right] \quad .$$

We do this by considering the following sequence of equations. Suppose that $D = \left\{d_1, \ldots, d_{|D|}\right\}$ and $E = \left\{e_1, \ldots, e_{|E|}\right\}$. Then

$$
\begin{aligned}
\alpha_C\left[\mathsf{inj}_E(g \circ f, p)\right] &= \mathsf{seed}\left(\begin{array}{l}\mathsf{sub}_{C\cup E}(gf e_{|E|}, e_{|E|}, \ldots\\ \quad \mathsf{sub}_{C\cup E}(gf e_1, e_1, [E \hookrightarrow C\cup E]p)\ldots)\ @C\end{array}\right) && (1)\\
&= \mathsf{seed}\left(\begin{array}{l}\mathsf{sub}_{C\cup D\cup E}(gf e_{|E|}, e_{|E|}, \ldots\\ \quad \mathsf{sub}_{C\cup D\cup E}(gf e_1, e_1, [E \hookrightarrow C\cup D\cup E]p)\ldots)\ @C\end{array}\right) && (2)\\
&= \mathsf{seed}\left(\begin{array}{l}\mathsf{sub}_{C\cup D\cup E}(gf e_{|E|}, fe_{|E|},\\ \quad \mathsf{sub}_{C\cup D\cup E}(fe_{|E|}, e_{|E|}, \ldots\\ \quad\quad \mathsf{sub}_{C\cup D\cup E}(gf e_1, fe_1,\\ \quad\quad\quad \mathsf{sub}_{C\cup D\cup E}(fe_1, e_1, [E \hookrightarrow C\cup D\cup E]p))\ldots))\ @C\end{array}\right) && (3)\\
&= \mathsf{seed}\left(\begin{array}{l}\mathsf{sub}_{C\cup D\cup E}(gd_{|D|}, d_{|D|}, \ldots\\ \quad \mathsf{sub}_{C\cup D\cup E}(gd_1, d_1,\\ \quad\quad \mathsf{sub}_{C\cup D\cup E}(fe_{|E|}, e_{|E|}, \ldots\\ \quad\quad\quad \mathsf{sub}_{C\cup D\cup E}(fe_1, e_1, [E \hookrightarrow C\cup D\cup E]p))\ldots))\ @C\end{array}\right) && (4)\\
&= \mathsf{seed}\left(\begin{array}{l}\mathsf{sub}_{C\cup D}(gd_{|D|}, d_{|D|}, \ldots\\ \quad \mathsf{sub}_{C\cup D}(gd_1, d_1,\\ \quad\quad [D \hookrightarrow C\cup D]\mathsf{seed}(\\ \quad\quad\quad \mathsf{sub}_{D\cup E}(fe_{|E|}, e_{|E|}, \ldots\\ \quad\quad\quad\quad \mathsf{sub}_{D\cup E}(fe_1, e_1, [E \hookrightarrow D\cup E]p)\ldots)\ @D))\ldots)\ @C\end{array}\right) && (5)\\
&= \alpha_C\left[\mathsf{inj}_D\left(g, \alpha_D\left[\mathsf{inj}_E(f, p)\right]\right)\right] \quad . && (6)
\end{aligned}
$$

Using: (1) defn. of α_C; (2) nat. of sub; (3) repeated use of NomSub-3 and NomSub-2; (4) repeated use of NomSub-2 and NomSub-4; (5) properties of supports and seeds; (7) defn. of α_C and α_D.

Thus a nominal substitution induces a $(U^{\mathbf{I}}_{\mathbf{F}} \circ (j^{\mathbf{I}}_{\mathbf{F}})_!)$-algebra.

The operation converting a $(U^{\mathbf{I}}_{\mathbf{F}} \circ (j^{\mathbf{I}}_{\mathbf{F}})_!)$-algebra to a nominal substitution is left and right inverse to the operation converting a nominal substitution to a $(U^{\mathbf{I}}_{\mathbf{F}} \circ (j^{\mathbf{I}}_{\mathbf{F}})_!)$-algebra. To see this, it is a matter

of stepping through the conversion processes, using Axioms NOMSUB-0–4 on the one hand, and the laws for $(U^{\mathbf{I}}_{\mathbf{F}} \circ (j^{\mathbf{I}}_{\mathbf{F}})_!)$-algebras on the other.

Moreover, for any two $(U^{\mathbf{I}}_{\mathbf{F}} \circ (j^{\mathbf{I}}_{\mathbf{F}})_!)$-algebras, (P, α) and (Q, β), a natural transformation $P \to Q$ is a homomorphism of $(U^{\mathbf{I}}_{\mathbf{F}} \circ (j^{\mathbf{I}}_{\mathbf{F}})_!)$-algebras if and only if is a homomorphism between the corresponding nominal substitutions. □

7.C Appendix to Chapter 7: Proof of Theorem 7.5.3

We now prove Theorem 7.5.3. We begin by recalling the statement of the theorem from page 185.

Theorem 7.5.3. *The mapping described in* (7.5.2) *takes a* И$_e$-LTS *to an* **I**-IL$_e$TS *satisfying Axioms* **I**1–**I**6. *For each nominal set X, the mapping defines a bijective correspondence between* И$_e$-LTS*s over X and* **I**-IL$_e$TS*s over the sheaf in* $\mathbf{Sh}(\mathbf{I})$ *corresponding to X, that satisfy Axioms* **I**1–**I**6.

Proof. Let $P \in \mathbf{Sh}(\mathbf{I})$ be the sheaf corresponding to X, according to Section 7.1.4. We first verify that Axioms **I**1–**I**6 of Figure 3.4 hold of the **I**-IL$_e$TS $\longrightarrow_{\mathbf{I}}$ over P induced from a И$_e$-LTS $\longrightarrow$ over X. As regards Axiom **I**1, suppose that

$$C \vdash x \xrightarrow{\ell}_{\mathbf{I}} D \vdash y$$

is induced. We will focus on the case $\ell = c?d$, cases for the other modes of communication are treated similarly. We know that

$$x \xrightarrow{c?d} y$$

and that C supports x, while $D = C \cup \{d\}$ — *the data is learnt*. By Axiom И$_e$1, $c \in \mathsf{supp}_X(x)$. But $\mathsf{supp}_X(x) \subseteq C$, so $c \in C$ — *the channel is known*.

Turning to Axiom **I**2: suppose that

$$C \vdash x \xrightarrow{c?d}_{\mathbf{I}} C \cup \{d\} \vdash y$$

is induced, and consider a name $d' \in \mathcal{N}$. We know that

$$x \xrightarrow{c?d} y$$

and that C supports x. By Axiom И$_e$3, we have $y' \in X$ such that

$$x \xrightarrow{c?d'} y' \quad .$$

Axiom И$_e$2 enforces that $(C \cup \{d'\})$ must support y'; thus

$$C \vdash x \xrightarrow{c?d'}_{\mathbf{I}} C \cup \{d'\} \vdash y'$$

as required.

We turn now to Axiom **I**3. Suppose that

$$C \vdash x \xrightarrow{\ell}_{\mathbf{I}} C \cup \{d\} \vdash y$$

is induced, with $\mathsf{ch}(\ell) \in C$. We consider a bijection $\beta : C \cup \{d\} \xrightarrow{\sim} D$. We will focus on the case of free output, where $\ell = c!d$ and $d \in \mathsf{supp}_X(x)$, but other modes of communication are treated similarly. We must have that C supports x and that C supports y, and that

$$x \xrightarrow{c!d} y \quad .$$

Since $\xrightarrow{-!-}$ is equivariant,

$$\left(\beta^\sharp \bullet_X x\right) \xrightarrow{(\beta c)!(\beta d)} \left(\beta^\sharp \bullet_X y\right)$$

for any permutation $\beta^\sharp$ of $\mathcal{N}$ that acts as β on $(C \cup \{d\})$; we outlined how such a permutation can be constructed in the proof of Section 7.1.4. By Prop. 7.1.2(3), $\beta(C) = D$ supports $\sigma \bullet_X x$ and D supports $\sigma \bullet_X y$. By definition of P,

$$\beta^\sharp \bullet_X x = P(\beta|_C)(x) \qquad \beta^\sharp \bullet_X y = P(\beta)(y) \quad .$$

Thus Axiom **I**3 holds.

Axiom **I**4a follows directly from the definition of $\longrightarrow$. As regards Axiom **I**4b: suppose that

$$C \cup \{d\} \vdash [C \hookrightarrow C \cup \{d\}]x \xrightarrow{c!d}_{\mathbf{I}} C \cup \{d\} \vdash y \quad .$$

There are two ways that an output transition can be induced; here though, we know that the name context of the transition source is the same as the name context of the transition target. Thus the transition must be induced by

$$x \xrightarrow{c!d} y$$

with $d \in \mathsf{supp}_X(x)$. Since x is supported by C, we have that $d \in C$, as required.

Axioms **I**5 and **I**6 arise because if C supports $x \in X$ then for any $D \supseteq C$ we have that D supports x.

This concludes our proof that Axioms **I**1–**I**6 hold of the induced transition system $\longrightarrow_{\mathbf{I}}$ over P.

We now describe how to induce a $И_e$-LTS from an **I**-IL$_e$TS. We continue to consider a nominal set $X \in \mathbf{Nom}$, with corresponding sheaf $P \in \mathbf{Sh}(\mathbf{I})$ according to Section 7.1.4. Suppose that we have an **I**-IL$_e$TS $\longrightarrow$ over P, that satisfies Axioms **I**1–**I**6 of Figure 3.4. We induce a $И_e$-LTS $(\xrightarrow{-?-}_И, \xrightarrow{-!-}_И, \xrightarrow{\tau}_И)$ over X as follows.

$$\begin{aligned} &\text{If } C \vdash x \xrightarrow{c?d} D \vdash y \text{ then } x \xrightarrow{c?d}_И y. \\ &\text{If } C \vdash x \xrightarrow{c!d} D \vdash y \text{ then } x \xrightarrow{c!d}_И y. \\ &\text{If } C \vdash x \xrightarrow{\tau} D \vdash y \text{ then } x \xrightarrow{\tau}_И y \end{aligned} \tag{7.C.1}$$

We must show that these relations are equivariant and satisfy Axioms $И_e$1–$И_e$3.

As for equivariance, suppose that

$$x \xrightarrow{c?d}_И y$$

is induced, and consider $\sigma \in \mathsf{Sym}(\mathcal{N})$. We must have some C, D such that C supports x and D supports y and

$$C \vdash x \xrightarrow{c?d} D \vdash y \quad .$$

By Axiom **I**1, $c \in C$ and $D = C \cup \{d\}$. By Axiom **I**3,

$$\sigma(C) \vdash P\left((\sigma|_D)|_C\right)(x) \xrightarrow{c?d} \sigma(D) \vdash P(\sigma|_D)(y) \quad .$$

But, by definition, $P\left((\sigma|_D)|_C\right)(x) = \sigma \bullet_X x$ and $P(\sigma|_D)(y) = \sigma \bullet_X y$. So we have

$$\sigma(C) \vdash (\sigma \bullet_X x) \xrightarrow{c?d} \sigma(D) \vdash (\sigma \bullet_X y) \quad .$$

Thus

$$(\sigma \bullet_X x) \xrightarrow{c?d}_И (\sigma \bullet_X y)$$

is induced, and so the relation $\xrightarrow{-?-}_И$ is equivariant. The relations for other modes of communication are seen to be equivariant in a similar way.

We turn now to show that Axiom $И_e1$ is satisfied by the induced relations. Suppose, for instance, that

$$x \xrightarrow{c!d}_{И} y$$

is induced. Then we must have sets of names C and D such that C supports x and

$$C \vdash x \xrightarrow{c!d} D \vdash y.$$

By Axiom **I**1, $D = C \cup \{d\}$. Now, note that $d \notin \big(C \setminus (\mathsf{supp}_X(x) \cup (C \cap \{d\}))\big)$. Thus, by Axiom **I**6,

$$\big(\mathsf{supp}_X(x) \cup (C \cap \{d\})\big) \vdash x \xrightarrow{c!d} \big(\mathsf{supp}_X(x) \cup \{d\}\big) \vdash y'$$

(We simplify using the definition of the action of P.) By Axiom **I**4b, if $d \in C$ then we have $d \in \mathsf{supp}_X(x)$. So

$$\big(\mathsf{supp}_X(x) \cup (C \cap \{d\})\big) = \mathsf{supp}_X(x) \quad .$$

Axiom **I**1 ensures that $c \in \mathsf{supp}_X(x)$; thus Axiom $И_e1$ is satisfied for the induced output relation. Axiom $И_e1$ is proved for the input relation in a similar way.

As regards Axiom $И_e2$, we proceed as follows. Suppose, for instance, that

$$x \xrightarrow{\tau}_{И} y$$

is induced. Then we have name sets C, D such that C supports x and

$$C \vdash x \xrightarrow{\tau} D \vdash y \quad .$$

Axiom **I**1 enforces that $D = C$. By Axiom **I**6, we have $y' \in P(\mathsf{supp}_X(x))$ such that

$$P(\mathsf{supp}_X(x) \hookrightarrow C)(y') = y \quad \text{and} \quad \mathsf{supp}_X(x) \vdash x \xrightarrow{\tau} \mathsf{supp}_X(x) \vdash y' \quad .$$

By definition of P, the first condition amounts to requiring that $y = y'$, and thus we have $\mathsf{supp}_X(y) \subseteq \mathsf{supp}_X(x)$. Axiom $И_e2$ is verified for other modes of communication in a similar manner.

Axiom $И_e3$ is quickly seen to be a consequence of Axioms **I**2 in the presence of Axiom **I**1.

It remains for us to show that for any $И_e$-LTS $\longrightarrow$ over X and any **I**-IL$_e$TS $\longrightarrow$ over P,

$$\text{(i)} \ \longrightarrow = \big(\longrightarrow_{\mathbf{I}}\big)_{И} \qquad \text{and} \qquad \text{(ii)} \ \longrightarrow = \big(\longrightarrow_{И}\big)_{\mathbf{I}} \quad . \tag{7.C.2}$$

We begin by showing that LHS $\subseteq$ RHS in (7.C.2)(i): that if $x \xrightarrow{\ell} y$ then $x \xrightarrow{\ell}_{\mathbf{I}И} y$. We concentrate on the case where $\ell = c!d$; reasoning is similar for the other modes of communication. If $x \xrightarrow{c!d} y$ then, by Axiom $И_e2$, $(\mathsf{supp}_X(x) \cup \{d\})$ supports y. Whether or not $d \in \mathsf{supp}_X(x)$ we have that the transition

$$\mathsf{supp}_X(x) \vdash x \xrightarrow{c!d}_{\mathbf{I}} \mathsf{supp}_X(x) \cup \{d\} \vdash y$$

is induced. Thus the transition

$$x \xrightarrow{c!d}_{\mathbf{I}И} y$$

is induced, as required.

We now show the converse: if $x \xrightarrow{\ell}_{\mathbf{I}И} y$ then $x \xrightarrow{\ell} y$. Indeed, suppose that $x \xrightarrow{c!d}_{\mathbf{I}И} y$ is induced, this time with $\ell = c?d$. Then we must have $C, C' \subseteq_{\mathrm{f}} \mathcal{N}$ such that C supports x and C' supports y and such that

$$C \vdash x \xrightarrow{c?d}_{\mathbf{I}} C' \vdash y \quad .$$

This, in turn, must be have been induced by a transition

$$x \xrightarrow{c?d} y \quad .$$

Thus property (i) of (7.C.2) is proved.

Turning to property (ii) of (7.C.2), we show that if $C \vdash x \xrightarrow{\ell} C' \vdash y$ then we have $C \vdash x \xrightarrow{\ell}_{\mathit{И}\mathbf{I}} C' \vdash y$. Indeed, suppose that

$$C \vdash x \xrightarrow{\ell} C' \vdash y$$

is induced, with $\ell = c!d$. Then we have that C supports x and the transition

$$x \xrightarrow{c!d}_{\mathit{И}} y$$

has been induced. If $d \in C$ then, by **I**4b, $d \in \mathsf{supp}_X(x)$. So, whether or not $d \in C$, the transition

$$C \vdash x \xrightarrow{c!d}_{\mathit{И}\mathbf{I}} C \cup \{d\} \vdash y$$

is induced. Also, by Axiom **I**1, $C' = C \cup \{d\}$. Other modes of communication are treated similarly; in this way one direction of property (ii) is proved.

For the other direction, that if $C \vdash x \xrightarrow{\ell}_{\mathit{И}\mathbf{I}} C' \vdash y$ then $C \vdash x \xrightarrow{\ell} C' \vdash y$, we proceed as follows. Suppose that

$$C \vdash x \xrightarrow{\ell}_{\mathit{И}\mathbf{I}} C' \vdash y$$

is induced. We will consider the case $\ell = c?d$. We must have that $C' = C \cup \{d\}$, and a transition

$$x \xrightarrow{c?d}_{\mathit{И}} y$$

must have been induced. Thus we must have $D, D' \subseteq_{\mathrm{f}} \mathcal{N}$ such that D supports x, D' supports y, and we have a transition

$$D \vdash x \xrightarrow{c?d} D' \vdash y \quad .$$

By Axiom **I**4a we can assume that $d \in D$. Axiom **I**1 ensures that $D' = D \cup \{d\}$. Since $\mathsf{supp}_X(x)$ supports x, so does $(\mathsf{supp}_X(x) \cup \{d\})$, and so, by Axiom **I**6, we have the transition

$$(\mathsf{supp}_X(x) \cup \{d\}) \vdash x \xrightarrow{c?d} (\mathsf{supp}_X(x) \cup \{d\}) \vdash y \quad .$$

We know that $(\mathsf{supp}_X(x) \cup \{d\}) \subseteq C \cup \{d\}$; thus Axiom **I**5 gives the transition

$$C \cup \{d\} \vdash x \xrightarrow{c?d} C \cup \{d\} \vdash y$$

and Axiom **I**4a provides

$$C \vdash x \xrightarrow{c?d} C \cup \{d\} \vdash y \quad .$$

The cases for other modes of communication are treated similarly; for output, Axiom **I**4b is required.

Thus properties (i) and (ii) of (7.C.2) are established, and Theorem 7.5.3 is proved. □

Chapter 8

Operational Semantics for Name-Passing

For various reasons, the GSOS format (as recalled in Section 6.3) is not relevant for name-passing systems. The purpose of this chapter is to provide a rule format for name-passing systems, together with an analysis based on the mathematical structural operational semantics of Section 6.2. We do all this by using the models of syntax and behaviour that were developed in the previous chapter.

We begin, in Section 8.1, by providing a notion of rule structure for name-passing systems. As we explain, these rule structures can be understood as nominal logic formulae. Thus the problem of finding a rule format for name-passing systems becomes the problem of finding conditions on rule structures that ensure that the intended models of the corresponding nominal logic theories are well-behaved. We introduce these conditions in Section 8.2; we call the resulting format *the* И-GSOS$^+$ *format*. This format serves, in particular, to explain the good behaviour of the π-calculus: its models are nominal ground labelled transition systems, in the sense of Section 7.5, and wide open bisimilarity is a congruence.

In Section 8.3 we provide examples of rule structures that violate the conditions, and we observe how the semantics induced by such rule structures may break the requirements of well-behavedness.

Finally, in Section 8.4, we explain how rule structures give rise to abstract rules and hence to monad liftings. In this way, we conclude that every system that is induced from a set of rules in the И-GSOS$^+$ format is well-behaved.

8.1 Rules for name-passing

In this section we redevelop the notion of rule structure from Section 6.3.1 in the context of name-passing systems. The development of that section is not adequate for this context. To see this, recall (*e.g.* from Figure 3.3) one of the rules of communication and the rule for scope opening, for ground transitions in the π-calculus:

$$\frac{p \xrightarrow[\pi]{\bar{c}d} q \qquad p' \xrightarrow[\pi]{c(z)} q'}{p \,|\, p' \xrightarrow[\pi]{\tau} q \,|\, [d/z]q'} \qquad\qquad \frac{p \xrightarrow[\pi]{\bar{c}z} q}{\boldsymbol{\nu} z.p \xrightarrow[\pi]{\bar{c}(z)} q} (z \neq c)\,.$$

Note that, in the communication rule (on the left) there is an explicit substitution operator; meanwhile, in the rule for scope opening (on the right), we see that the syntax and also the labels include binding operators, while the rule has side conditions about the distinctness of names. None of these aspects can be accomodated within the rule structures of Section 6.3.1, nor indeed within the GSOS framework of Bloom *et al.* [1995]. Thus we move from the standard algebraic treatment of syntax used in Section 6.3.1, to the treatment based on nominal sets and nominal substitutions developed in the previous chapter. Indeed, instead of interpreting the rule structures as theories of first order logic, as in Section 6.3.1, rule structures for name-passing are to be understood as theories of nominal logic.

We introduce the notion of rule structure for name-passing in Section 8.1.1. Section 8.1.2 is dedicated to an exposition of the intended meaning of rule structures.

Throughout this section we fix a nominal algebraic signature $\mathbb{S}$ (in the sense of Definition 7.4.1).

8.1.1 Rule structures for name passing

In this subsection we introduce and recall various notions, culminating, in Definition 8.1.1, with the notion of rule structure for name-passing. We begin by recalling our treatment of raw syntax, and introduce a simple mechanism for handling the explicit substitutions that appear in rules. We then discuss why it is that rule structures are built from raw syntax, rather than abstract syntax with α-equivalence. Before introducing rule structures, we recall a set of labels for ground behaviour, and some basic properties of such labels.

Raw syntax. In Section 7.4 we showed how, for a given set $\mathtt{N}$ of name metavariables, the nominal signature $\mathbb{S}$ induces an endofunctor $\Sigma_{\mathbb{S},\mathbf{Set}_{\mathtt{N}}}$ on **Set** such that for any set $\mathtt{X}$, the set $\Sigma_{\mathbb{S},\mathbf{Set}_{\mathtt{N}}}\mathtt{X}$ contains all basic expressions of raw syntax, of the form

$$\mathsf{op}\left((\mathtt{c}_i)_{i\in[1,\mathrm{arn}(\mathsf{op})]},\left(\left\langle \mathtt{a}_k^j\right\rangle_{k\in[1,\mathrm{dep}_j(\mathsf{op})]}\mathtt{x}_j\right)_{j\in[1,\mathrm{art}(\mathsf{op})]}\right)$$

with name metavariables $\mathtt{c}_i$, $\mathtt{a}_k^j$ taken from the set $\mathtt{N}$ and term variables $\mathtt{x}_j$ taken from the set $\mathtt{X}$. Recall that the suggestive notation

$$\left\langle \mathtt{a}_k^j\right\rangle_{k\in[1,\mathrm{dep}_j(\mathsf{op})]}\mathtt{x}_j$$

denotes simply a tuple $(\mathtt{a}_1^j,\ldots,\mathtt{a}_{\mathrm{dep}_j(\mathsf{op})}^j,\mathtt{x}_j)$; there is no quotient or α-equivalence involved in the raw syntax.

In the introduction to this section we noted that explicit substitutions may appear in the conclusion of the rule. To accommodate this, we will sometimes add to the nominal algebraic signature $\mathbb{S}$ the nominal substitution operator sub, from Section 7.3, with

$$\mathrm{arn}(\mathsf{sub})=1 \qquad \mathrm{art}(\mathsf{sub})=1 \qquad \mathrm{dep}_1(\mathsf{sub})=1 \quad .$$

We write $(\mathbb{S}+\mathsf{sub})$ for the resulting signature. This corresponds to the combination of the nominal logic signature for $\mathbb{S}$ with the signature for the theory of nominal substitutions. It makes no sense, though, to consider Axioms NOMSUB-1–NOMSUB-3 in this raw syntax setting.

The free monad $\mathbf{T}_{\mathbb{S}+\mathsf{sub},\mathbf{Set}_{\mathtt{N}}}$ on the endofunctor $\Sigma_{\mathbb{S}+\mathsf{sub},\mathbf{Set}_{\mathtt{N}}}$ exists. The set $T_{\mathbb{S}+\mathsf{sub},\mathbf{Set}_{\mathtt{N}}}(\mathtt{X})$ contains all compound terms of raw syntax with explicit substitutions and with name variables taken from the set $\mathtt{N}$ and term variables from the set $\mathtt{X}$.

Discussion: Raw syntax in the rules. We use raw syntax in the definition of rule structures, as opposed to α-equivalence classes of syntax. This is because it is not immediately clear how to define α-equivalence for expressions that have free term variables. Indeed consider one of the π-calculus rules for transition under restriction, written here according to the conventions of syntax in $\mathbf{Set}_{\mathtt{N}}$.

$$\frac{\mathtt{x}\xrightarrow{\mathtt{c!d}}\mathtt{y}}{\mathsf{restrct}(\langle\mathtt{a}\rangle\,\mathtt{x})\xrightarrow{\mathtt{c!d}}\mathsf{restrct}(\langle\mathtt{a}\rangle\,\mathtt{y})}(\mathtt{c}\neq\mathtt{a}\neq\mathtt{d})$$

Suppose we were to consider $\langle\mathtt{a}\rangle\,\mathtt{x}$ as an α-equivalence class using the machinery of nominal sets introduced in Chapter 7. A first question is: What is the support of $\mathtt{x}$? If the support is empty, then

$$\mathsf{restrct}(\langle\mathtt{a}\rangle\,\mathtt{x})=\mathsf{restrct}(\langle\mathtt{c}\rangle\,\mathtt{x})$$

and so we equally have the rule

$$\frac{\mathtt{x} \xrightarrow{\mathtt{c!d}} \mathtt{y}}{\mathsf{restrct}(\langle \mathtt{c} \rangle \mathtt{x}) \xrightarrow{\mathtt{c!d}} \mathsf{restrct}(\langle \mathtt{a} \rangle \mathtt{y})} (\mathtt{c} \neq \mathtt{a} \neq \mathtt{d})$$

which was certainly *not* intended. On the other hand, if the term variable $\mathtt{x}$ has non-empty support then there must be a permutation σ of name variables such that $\sigma \bullet \mathtt{x}$ is also a term variable and such that $\sigma \bullet \mathtt{x} \neq \mathtt{x}$. Such relationships between term variables would surely make for a very elaborate notion of rule [but see *e.g.* Urban *et al.*, 2004].

Thus, while α-equivalence is fundamental to our treatment of syntax of process terms, the rules that we consider will not involve any α-equivalence.

In the following subsection we will explain how rules can be understood from the standpoint of nominal logic (as recalled in Section 7.2). Nominal logic is simply a first-order theory, so the logical equality of the theory, which involves an axiomatisation of α-equivalence, is stronger than syntactic equality, which treats syntax as raw.

Labels. In (3.3.18) we introduced the set of ground labels as

$$Lab_g = \mathcal{N} \times \mathcal{N} \;+\; \mathcal{N} \times \mathcal{N} \;+\; \mathcal{N} \times \mathcal{N} \;+\; 1 \quad .$$

Our rules will involve name metavariables, and so it is helpful to parameterise the set of labels, by the set of names under consideration. To this end, we define, for each set $\mathtt{N}$, the set of labels

$$Lab_g(\mathtt{N}) = \mathtt{N} \times \mathtt{N} \;+\; \mathtt{N} \times \mathtt{N} \;+\; \mathtt{N} \times \mathtt{N} \;+\; 1 \quad .$$

As with the set of (3.3.18), we consider the components as describing input of a fresh name (written $\mathtt{c?(a)}$), output ($\mathtt{c!a}$), bound output ($\mathtt{c!(a)}$), and silent action (τ). Note that here it is essential to distinguish between free and bound outputs, because it does not make sense to refer to the free names of a term metavariable.

For each label $\mathtt{l} \in Lab_g(\mathtt{N})$, we define the sets of free names $\mathsf{fn}(\mathtt{l})$ of $\mathtt{l}$ and bound names $\mathsf{bn}(\mathtt{l})$ of $\mathtt{l}$ in the following table.

l	$\mathsf{fn}(l)$	$\mathsf{bn}(l)$
$\mathtt{c?(a)}$	$\{\mathtt{c}\}$	$\{\mathtt{a}\}$
$\mathtt{c!d}$	$\{\mathtt{c},\mathtt{d}\}$	$\emptyset$
$\mathtt{c!(a)}$	$\{\mathtt{c}\}$	$\{\mathtt{a}\}$
τ	$\emptyset$	$\emptyset$

The function bn is essentially the assignment for π-calculus ground labels provided in Section 3.1.2.

Rule structures.

Definition 8.1.1. Let $(\mathtt{X},\mathtt{N})$ be a pair of sets, with $\mathtt{N}$ finite. A *premise structure* over $(\mathtt{X},\mathtt{N})$ is an element of the set $\mathtt{X} \times Lab_g(\mathtt{N}) \times \mathtt{X}$. A *conclusion structure* over $(\mathtt{X},\mathtt{N})$ is a tuple in the set

$$\Sigma_{\mathbb{S},\mathbf{Set}_{\mathtt{N}}}(\mathtt{X}) \times Lab_g(\mathtt{N}) \times T_{\mathbb{S}+\mathsf{sub},\mathbf{Set}_{\mathtt{N}}}(\mathtt{X}).$$

(Notice that in the last component of the product, the signature $(\mathbb{S}+\mathsf{sub})$ with explicit substitutions is used.)

A *rule structure for name-passing* over $(\mathtt{X},\mathtt{N})$ is a pair of a finite set of premise structures over $(\mathtt{X},\mathtt{N})$ and a conclusion structure over $(\mathtt{X},\mathtt{N})$.

8.1.2 Understanding rule structures

The reader may already have an inkling of the intended meaning of the rule structures of Definition 8.1.1. The purpose of this subsection is to develop this further. First, we explain our intention that rule structures be interpreted as having implicit side conditions about the freshness of names. We then provide some examples of rule structures, including, in Figure 8.1, rule structures describing the π-calculus. Apart from the implicit side conditions, the rule structures essentially follow the presentation of the π-calculus of Figure 3.3. We conclude this subsection by making the interpretation of rule structures precise, by exposing rule structures as a restricted class of nominal logic formulae.

Implicit side conditions. In the introduction to this section we observed that rules for name-passing systems often include side conditions. For instance, consider the π-calculus rules for parallel transitions and scope opening. (The rules here are essentially as they appear in Figure 3.3, only we use a more formal syntax and the nominal logic $\#$ operator.)

$$\frac{\mathtt{x} \xrightarrow{\mathtt{c!a}} \mathtt{y}}{\mathsf{restrct}(\langle \mathtt{a} \rangle \mathtt{x}) \xrightarrow{\mathtt{c!(a)}} \mathtt{y}} (\mathtt{a} \neq \mathtt{c}) \qquad \frac{\mathtt{x}' \xrightarrow{\mathtt{c!(a)}} \mathtt{y}'}{\mathsf{par}(\mathtt{x}, \mathtt{x}') \xrightarrow{\mathtt{c!(a)}} \mathsf{par}(\mathtt{x}, \mathtt{y}')} (\mathtt{a} \# \mathtt{x})$$

These side conditions, insisting on freshness and distinctness of names, are the only kinds of side conditions that need to be considered. To make matters simpler, we will make these side conditions implicit by *always* imposing the following side conditions:

1. All names must be as distinct as they are in the rule.
2. Bound data in the conclusion label must be fresh for the conclusion source.

This will all be made precise below, when we introduce an interpretation of a rule structure as a nominal logic formula.

Because the side conditions are always implicit, in order to describe rules with*out* side conditions it may be necessary to consider several versions of the rule together; examples are provided in the following paragraph. Including several versions is rather inconvenient, and one might concoct a notion of 'rule structure with explicit side conditions' from which a family of rule structures with *implicit* side conditions could be derived via a standard procedure.

Examples. As a first example we show how part of the π-calculus rule for communication arises as a rule structure for the nominal algebra signature $\mathbb{PI}$ introduced in (7.4.2). The term variables for this rule structure are $\mathtt{X} = \{\mathtt{x}, \mathtt{x}', \mathtt{y}, \mathtt{y}'\}$ and the name metavariables are $\mathtt{N} = \{\mathtt{a}, \mathtt{c}, \mathtt{d}\}$. There are two premises

$$(\mathtt{x}, \mathtt{c!d}, \mathtt{y}) \quad \text{and} \quad (\mathtt{x}', \mathtt{c?(a)}, \mathtt{y}')$$

and the conclusion is the triple

$$\Big(\mathsf{par}(\mathtt{x}, \mathtt{x}'), \tau, \mathsf{par}(\mathtt{y}, [\mathtt{d}/\mathtt{a}]\mathtt{y}')\Big) \quad .$$

We will often write rule structures in a more suggestive way, with the premises above and the conclusion below, as follows.

$$\frac{\mathtt{x} \xrightarrow{\mathtt{c!d}} \mathtt{y} \qquad \mathtt{x}' \xrightarrow{\mathtt{c?(a)}} \mathtt{y}'}{\mathsf{par}(\mathtt{x}, \mathtt{x}') \xrightarrow{\tau} \mathsf{par}(\mathtt{y}, [\mathtt{d}/\mathtt{a}]\mathtt{y}')}$$

For this rule structure, the implicit side conditions amount to requiring that $\mathtt{a} \neq \mathtt{c}$, $\mathtt{a} \neq \mathtt{d}$ and $\mathtt{c} \neq \mathtt{d}$. One would need a separate rule structure to handle the case where $\mathtt{c} = \mathtt{d}$.

Further examples of rules from the π-calculus are given in Figure 8.1. Because the side conditions are implicit it is not necessary to include side conditions such as "$\mathtt{c} \neq \mathtt{d}$" in the rule for mismatch, or "$\mathtt{a}\#\mathtt{x}$" in the rule for transitions in parallel. On the other hand it is often necessary to include several copies of rules. For example, in Figure 8.1, we have included one rule for output of data that is distinct from the channel, and one rule for output of data that is the same as the channel. For free output under a mismatch operator, a full presentation would require seven rules to describe all the possible coincidences of variables.

As mentioned in Section 3.1, we have not explicitly introduced any facilities for infinitary behaviour in this thesis. There are various ways of adding replication to the calculus, and we note that the semantics suggested by Sangiorgi and Walker [2001, Table 1.5] does fit into our rule format.

Interpretation in nominal logic. Suppose that we have a set $\mathscr{R}$ of rule structures. This set induces a nominal logic theory, in the sense of Section 7.2. The theory has one basic sort, X, and the signature has all the function symbols of the signature $\mathbb{S}$ together with the nominal substitution operator sub, and also four relation symbols:

- a *bound input transition* relation symbol $\left(\overset{-?(-)}{\longrightarrow}\right)$ with arity $(\mathsf{X},\mathsf{N},\mathsf{N},\mathsf{X})$;
- an *output transition* relation symbol $\left(\overset{-!-}{\longrightarrow}\right)$ with arity $(\mathsf{X},\mathsf{N},\mathsf{N},\mathsf{X})$;
- a *bound output transition* relation symbol $\left(\overset{-!(-)}{\longrightarrow}\right)$ with arity $(\mathsf{X},\mathsf{N},\mathsf{N},\mathsf{X})$;
- a *silent transition* relation symbol $\left(\overset{\tau}{\longrightarrow}\right)$ with arity (X,X).

(The reader will recognise these relation symbols as the symbols in the theory of nominal ground transition systems, introduced in Definition 7.5.5.)

The axioms of the theory are, firstly, the axioms NOMSUB-1–3 of nominal substitutions, and, secondly, axioms corresponding to the rule structures in $\mathscr{R}$, which are determined as follows.

Consider sets $\mathtt{X}$ and $\mathtt{N}$, with $\mathtt{N}$ finite. For each rule structure $\mathtt{R}$ over $(\mathtt{X},\mathtt{N})$ in $\mathscr{R}$, we define a nominal logic formula $\Phi_{\mathtt{R}}$, with free term variables in $\mathtt{X}$ and free name variables in $\mathtt{N}$, as follows. Let Prem be the set of premises of $\mathtt{R}$, and let $(\mathtt{src},\underline{\mathtt{l}},\mathtt{tar})$ be the conclusion of $\mathtt{R}$. As such, $\mathtt{src}$ and $\mathtt{tar}$ can be considered as expressions of nominal logic, of sort X, with free term variables in $\mathtt{X}$ and free name variables in $\mathtt{N}$. Then

$$\Phi_{\mathtt{R}} = \left(\bigwedge_{\substack{\mathtt{c},\mathtt{d}\in\mathtt{N}\\ \mathtt{c}\neq\mathtt{d}}} \mathtt{c}\neq\mathtt{d} \;\wedge\; \bigwedge_{\mathtt{c}\in\mathsf{bn}(\underline{\mathtt{l}})} \mathtt{c}\#\mathtt{src} \;\wedge\; \bigwedge_{(\mathtt{x},\mathtt{l},\mathtt{y})\in\mathrm{Prem}} \mathtt{x} \xrightarrow{\mathtt{l}} \mathtt{y} \right) \Longrightarrow \mathtt{src} \xrightarrow{\underline{\mathtt{l}}} \mathtt{tar}\,.$$

For each rule structure $\mathtt{R}$ over $(\mathtt{X},\mathtt{N})$ in $\mathscr{R}$, we enumerate the sets of variables, say

$$\mathtt{X} = \left\{\mathtt{x}_1,\ldots,\mathtt{x}_{|\mathtt{X}|}\right\} \qquad \mathtt{N} = \left\{\mathtt{a}_1,\ldots,\mathtt{a}_{|\mathtt{N}|}\right\}$$

and include, in the theory associated to $\mathscr{R}$, an axiom

$$\forall \mathtt{x}_1,\ldots,\mathtt{x}_{|\mathtt{X}|} : \mathsf{X}.\ \forall \mathtt{a}_1,\ldots,\mathtt{a}_{|\mathtt{N}|} : \mathsf{N}.\ \ \Phi_{\mathtt{R}} \quad .$$

Models of rule structures. As in Section 6.3.1, there is an 'intended' model of the nominal logic theory associated with a set $\mathscr{R}$ of rules, that is particularly important. This model has a carrier set $T_{\mathbb{S},\mathbf{Nom}}\emptyset$, the nominal set of terms of the signature $\mathbb{S}$. An $\mathbb{S}$-structure is given by the structure map $\Sigma_{\mathbb{S},\mathbf{Nom}} T_{\mathbb{S},\mathbf{Nom}}\emptyset \to T_{\mathbb{S},\mathbf{Nom}}\emptyset$. Recall, from Section 7.4, that the monad $\mathbf{T}_{\mathbb{S},\mathbf{Nom}}$ on **Nom** lifts along

(a) **Silent**
$\mathtt{X} = \{\mathtt{x}\}$, $\mathtt{N} = \emptyset$

$$\frac{}{\mathsf{tau}(\mathtt{x}) \xrightarrow{\tau} \mathtt{x}}$$

(b) **Input**
$\mathtt{X} = \{\mathtt{x}\}$, $\mathtt{N} = \{\mathtt{a}, \mathtt{c}\}$

$$\frac{}{\mathsf{inp}(\mathtt{c}, \langle \mathtt{a} \rangle \mathtt{x}) \xrightarrow{\mathtt{c?(a)}} \mathtt{x}}$$

(c) **Output** of the channel name
$\mathtt{X} = \{\mathtt{x}\}$, $\mathtt{N} = \{\mathtt{c}\}$

$$\frac{}{\mathsf{out}(\mathtt{c}, \mathtt{c}, \mathtt{x}) \xrightarrow{\mathtt{c!c}} \mathtt{x}}$$

(d) **Output** of a name distinct from the channel name
$\mathtt{X} = \{\mathtt{x}\}$, $\mathtt{N} = \{\mathtt{c}, \mathtt{d}\}$

$$\frac{}{\mathsf{out}(\mathtt{c}, \mathtt{d}, \mathtt{x}) \xrightarrow{\mathtt{c!d}} \mathtt{x}}$$

(e) **Match** for input transitions where channel is match name
$\mathtt{X} = \{\mathtt{x}, \mathtt{y}\}$, $\mathtt{N} = \{\mathtt{a}, \mathtt{c}\}$

$$\frac{\mathtt{x} \xrightarrow{\mathtt{c?(a)}} \mathtt{y}}{\mathsf{match}(\mathtt{c}, \mathtt{c}, \mathtt{x}) \xrightarrow{\mathtt{c?(a)}} \mathtt{y}}$$

(f) **Mismatch** for input transitions with channel as second mismatch name
$\mathtt{X} = \{\mathtt{x}, \mathtt{y}\}$, $\mathtt{N} = \{\mathtt{c}, \mathtt{d}, \mathtt{e}\}$

$$\frac{\mathtt{x} \xrightarrow{\mathtt{d?(e)}} \mathtt{y}}{\mathsf{mismatch}(\mathtt{c}, \mathtt{d}, \mathtt{x}) \xrightarrow{\mathtt{d?(e)}} \mathtt{y}}$$

(g) **Sum** for bound output on the left
$\mathtt{X} = \{\mathtt{x}, \mathtt{x}', \mathtt{y}\}$, $\mathtt{N} = \{\mathtt{a}, \mathtt{c}\}$

$$\frac{\mathtt{x} \xrightarrow{\mathtt{c!(a)}} \mathtt{y}}{\mathsf{sum}(\mathtt{x}, \mathtt{x}') \xrightarrow{\mathtt{c!(a)}} \mathtt{y}}$$

(h) **Parallel** for bound output on the right
$\mathtt{X} = \{\mathtt{x}, \mathtt{x}', \mathtt{y}'\}$, $\mathtt{N} = \{\mathtt{a}, \mathtt{c}\}$

$$\frac{\mathtt{x}' \xrightarrow{\mathtt{c!(a)}} \mathtt{y}'}{\mathsf{par}(\mathtt{x}, \mathtt{x}') \xrightarrow{\mathtt{c!(a)}} \mathsf{par}(\mathtt{x}, \mathtt{y}')}$$

(i) **Communication** with output on left, where data differs from channel
$\mathtt{X} = \{\mathtt{x}, \mathtt{x}', \mathtt{y}, \mathtt{y}'\}$, $\mathtt{N} = \{\mathtt{a}, \mathtt{c}, \mathtt{d}\}$

$$\frac{\mathtt{x} \xrightarrow{\mathtt{c!d}} \mathtt{y} \qquad \mathtt{x}' \xrightarrow{\mathtt{c?(a)}} \mathtt{y}'}{\mathsf{par}(\mathtt{x}, \mathtt{x}') \xrightarrow{\tau} \mathsf{par}(\mathtt{y}, [\mathtt{d}/\mathtt{a}]\mathtt{y}')}$$

(j) **Restriction** for bound output
$\mathtt{X} = \{\mathtt{x}, \mathtt{y}\}$, $\mathtt{N} = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$

$$\frac{\mathtt{x} \xrightarrow{\mathtt{c!(b)}} \mathtt{y}}{\mathsf{restrct}(\langle \mathtt{a} \rangle \mathtt{x}) \xrightarrow{\mathtt{c!(b)}} \mathsf{restrct}(\langle \mathtt{a} \rangle \mathtt{y})}$$

(k) **Scope closure** for output on left
$\mathtt{X} = \{\mathtt{x}, \mathtt{x}', \mathtt{y}, \mathtt{y}'\}$, $\mathtt{N} = \{\mathtt{a}, \mathtt{c}\}$

$$\frac{\mathtt{x} \xrightarrow{\mathtt{c!(a)}} \mathtt{y} \qquad \mathtt{x}' \xrightarrow{\mathtt{c?(a)}} \mathtt{y}'}{\mathsf{par}(\mathtt{x}, \mathtt{x}') \xrightarrow{\tau} \mathsf{restrct}\left(\langle \mathtt{a} \rangle \mathsf{par}(\mathtt{y}, \mathtt{y}')\right)}$$

(l) **Scope opening**
$\mathtt{X} = \{\mathtt{x}, \mathtt{y}\}$, $\mathtt{N} = \{\mathtt{a}, \mathtt{c}\}$

$$\frac{\mathtt{x} \xrightarrow{\mathtt{c!a}} \mathtt{y}}{\mathsf{restrct}(\langle \mathtt{a} \rangle \mathtt{x}) \xrightarrow{\mathtt{c!(a)}} \mathtt{y}}$$

Figure 8.1: Examples of rule structures for the π-calculus. Note that the side conditions are implicit.

the forgetful functor **NomSub** → **Nom**, and so we have a nominal substitution structure on $T_{\mathbb{S},\mathbf{Nom}}\emptyset$. The transition relations of the model are the smallest (equivariant) relations that satisfy axioms arising from the rule structures in $\mathscr{R}$. Thus the intended model of a set $\mathscr{R}$ of rule structures is in fact the initial model in the category of models for the nominal logic theory associated with $\mathscr{R}$.

8.2 Rules that induce well-behaved semantics

In this section we introduce conditions on rule structures that, as will be shown in Section 8.4, ensure that the intended model will be well-behaved. We begin, in Section 8.2.1, with a discussion of appropriate notions of well-behavedness for name-passing systems. Following this, in Section 8.2.2, we translate the conditions of the positive GSOS format to the setting of name-passing systems. These are not the only conditions needed, however, and in Sections 8.2.3 and 8.2.4 we develop some properties of rule structures that are used in Section 8.2.5 to define further conditions on rule structures, resulting in the *И-GSOS$^+$ format*.

Throughout this section we continue to fix an arbitrary nominal algebraic signature $\mathbb{S}$, and we consider finite sets $\mathtt{X}$ and $\mathtt{N}$, and an arbitrary rule structure $\mathtt{R}$ over $(\mathtt{X},\mathtt{N})$. Let Prem be the set of premises of $\mathtt{R}$, and let $(\mathtt{src},\underline{\mathtt{l}},\mathtt{tar})$ be the conclusion of $\mathtt{R}$; so we have an operator $\underline{\mathtt{op}} \in \mathrm{Op}_{\mathbb{S}}$ and

$$\begin{array}{ll} \underline{\mathtt{c}}_i \in \mathtt{N} & (\text{for } i \in [1,\mathrm{arn}(\underline{\mathtt{op}})]) \\ \underline{\mathtt{x}}_j \in \mathtt{X} & (\text{for } j \in [1,\mathrm{art}(\underline{\mathtt{op}})]) \\ \underline{\mathtt{a}}^j_k \in \mathtt{N} & (\text{for } j \in [1,\mathrm{art}(\underline{\mathtt{op}})],\ k \in [1,\mathrm{dep}_j(\underline{\mathtt{op}})]) \end{array}$$

such that

$$\mathtt{src} = \underline{\mathtt{op}}\left((\underline{\mathtt{c}}_i)_{i\in[1,\mathrm{arn}(\mathtt{op})]}, \left(\left\langle \underline{\mathtt{a}}^j_k \right\rangle_{k\in[1,\mathrm{dep}_j(\mathtt{op})]} \underline{\mathtt{x}}_j\right)_{j\in[1,\mathrm{art}(\mathtt{op})]}\right) \quad .$$

Here, as in Section 6.3.3, we underline the variables and the operator symbol that appear in the conclusion source. We refer to the elements $\underline{\mathtt{c}}_i$ as *free name variables* and the elements $\underline{\mathtt{a}}^j_k$ as *binding name variables*.

8.2.1 Notions of well-behavedness

In Section 6.3, we developed a rule format that ensured that the induced semantics would be well-behaved; there, that meant that bisimilarity would be a congruence. This is certainly not the only notion of well-behavedness that one can take for labelled transition systems. For instance, in their presentation of the GSOS format, Bloom *et al.* [1995, Sec. 5] take as a basic property of transition systems that they are finitely branching.

For name-passing systems, there are yet more notions of well-behavedness available. In Section 3.1.3 we recalled four notions of bisimilarity for the π-calculus; if we are to insist that one is a congruence, which one should it be? Neither early, late nor ground bisimilarity is a congruence for the full π-calculus, so we will not use them here. Thus we are left with the simpler notion of *wide open* bisimilarity.

Another consideration is that, in Chapters 3 and 4, and Section 7.5, we have axiomatised various properties of transition systems for name-passing. Moreover, the intended model of the nominal logic theory associated with a class of rule structures will certainly provide a *structure* for the theory of nominal ground labelled transition systems (Definition 7.5.5), and so it makes sense to insist that this intended model is also a *model* of the theory of nominal ground labelled transition systems.

In summary, we say that the intended model of a class of rule structures is *well-behaved* if (a) wide open bisimilarity is a congruence, and (b) Axioms $И_g1$ and $И_g2$ of Figure 7.2 are satisfied.

И-GSOS$^+$-1. Every element of $\mathtt{X}$ appears in the conclusion source or as a premise target.

$$\forall \mathtt{x} \in \mathtt{X}.\ \begin{pmatrix} \exists j \in [1, \mathrm{ar}(\underline{\mathtt{op}})].\ \underline{\mathtt{x}}_j = \mathtt{x} \\ \vee\ \exists \mathtt{y} \in \mathtt{X},\ \mathtt{l} \in Lab_g(\mathtt{N}).\ (\mathtt{y}, \mathtt{l}, \mathtt{x}) \in \mathrm{Prem} \end{pmatrix}$$

И-GSOS$^+$-2. The source of each premise appears in the conclusion source.

$$\forall (\mathtt{x}, \mathtt{l}, \mathtt{y}) \in \mathrm{Prem}.\ \exists j \in [1, \mathrm{ar}(\underline{\mathtt{op}})].\ \underline{\mathtt{x}}_j = \mathtt{x}$$

И-GSOS$^+$-3. The target of any premise does not appear in any other premise.

$$\forall (\mathtt{x}, \mathtt{l}, \mathtt{y}), (\mathtt{x}', \mathtt{l}', \mathtt{y}') \in \mathrm{Prem}.\ \mathtt{y} = \mathtt{y}' \implies \mathtt{x} = \mathtt{x}' \wedge \mathtt{l} = \mathtt{l}'$$

И-GSOS$^+$-4. The target of any premise does not appear in the conclusion source.

$$\forall (\mathtt{x}, \mathtt{l}, \mathtt{y}) \in \mathrm{Prem},\ j \in [1, \mathrm{ar}(\underline{\mathtt{op}})].\ \mathtt{y} \neq \underline{\mathtt{x}}_j$$

И-GSOS$^+$-5. Each variable in the conclusion source is distinct.

$$\forall j, j' \in [1, \mathrm{ar}(\underline{\mathtt{op}})].\ \underline{\mathtt{x}}_j = \underline{\mathtt{x}}_{j'} \implies j = j'$$

Figure 8.2: Conditions on a rule structure for name-passing. Notation is defined at the beginning of Section 8.2.

8.2.2 First conditions on rule structures

We introduce Conditions И-GSOS$^+$-1–5 (Figure 8.2) on rule structures. The reader will recognise Conditions И-GSOS$^+$-1–5 as direct translations of Conditions GSOS$^+$-1–5 of Figure 6.4.

Alone, Conditions И-GSOS$^+$-1–5 are insufficient to guarantee well-behavedness. Instead, they allow us to define further conditions from which we will be able to derive such a result.

For the rest of this section we assume that Conditions И-GSOS$^+$-1–5 hold of the ambient rule structure $\mathtt{R}$. The following result — which is essentially (6.3.6) — is useful when defining functions with domain $\mathtt{X}$, the set of term variables.

Proposition 8.2.1. *The function*

$$[1, \mathrm{art}(\underline{\mathtt{op}})] \ +\ \coprod_{j \in [1, \mathrm{art}(\underline{\mathtt{op}})]} \left\{ (\mathtt{x}, \mathtt{l}, \mathtt{y}) \in \mathrm{Prem} \;\middle|\; \mathtt{x} = \underline{\mathtt{x}}_j \right\} \longrightarrow \mathtt{X}$$

$$\mathsf{inl}(j) \longmapsto \underline{\mathtt{x}}_j$$

$$\mathsf{inr}(\mathsf{inj}_j(\mathtt{x}, \mathtt{l}, \mathtt{y})) \longmapsto \mathtt{y}$$

is a bijection. □

8.2.3 Associating names with components of the rule

We now introduce a variety of functions that associate name variables to components of the rule structure. By doing this we are able to approximate the variables that may appear once the term variables have been instantiated.

We define a function $\mathtt{BN} : \mathtt{X} \to \mathscr{P}(\mathtt{N})$ which assigns to each term variable $\mathtt{x} \in \mathtt{X}$ a set of names which are 'bound' in $\mathtt{x}$ in the rule structure. We use the bijection introduced in Prop. 8.2.1 to make this definition.

- For $j \in [1, \mathrm{art}(\underline{\mathtt{op}})]$, we let $\mathtt{BN}(\underline{\mathtt{x}}_j) = \left\{ \underline{\mathtt{a}}_k^j \;\middle|\; k \in [1, \mathrm{dep}_j(\underline{\mathtt{op}})] \right\}$.
- For $(\mathtt{x}, \mathtt{l}, \mathtt{y}) \in \mathrm{Prem}$ and each $j \in \mathrm{art}(\underline{\mathtt{op}})$ such that $\mathtt{x} = \underline{\mathtt{x}}_j$:
  we let $\mathtt{BN}(\mathtt{y}) = \left\{ \underline{\mathtt{a}}_k^j \;\middle|\; k \in [1, \mathrm{dep}_j(\underline{\mathtt{op}})] \right\} \cup \mathsf{bn}(\mathtt{l})$.

By way of example consider rule structure (j) of Figure 8.1, for bound output under restriction. There, we have $\mathtt{BN}(\mathtt{x}) = \{\mathtt{a}\}$ while $\mathtt{BN}(\mathtt{y}) = \{\mathtt{a}, \mathtt{b}\}$.

We will also find it useful to refer to the set $\mathtt{BN} \subseteq \mathtt{N}$ of all names that appear in binding position anywhere in the premise labels or in the conclusion source, given by

$$\mathtt{BN} = \bigcup \{ \mathtt{BN}(\mathtt{x}) \mid \mathtt{x} \in \mathtt{X} \} \quad .$$

For example: in the case that $\mathtt{R}$ is rule structure (j) of Figure 8.1, we have $\mathtt{BN} = \{\mathtt{a}, \mathtt{b}\}$.

We define a function $\mathtt{FN} : \mathtt{X} \to \mathscr{P}(\mathtt{N})$ which assigns to each term variable a set of names to be thought of as 'those names that can be assumed free in the expression to which the variable is instantiated'.

First, define a set $\mathtt{FN} \subseteq \mathtt{N}$ of names 'free in the conclusion source or the premises' by

$$\mathtt{FN} = \quad \left\{ \underline{\mathtt{c}}_1, \ldots, \underline{\mathtt{c}}_{\mathrm{arn}(\underline{\mathtt{op}})} \right\} \cup \bigcup_{\substack{j \in [1, \mathrm{art}(\underline{\mathtt{op}})] \\ \mathtt{l} \in Lab_g(\mathtt{N})}} \left\{ \mathsf{fn}(\mathtt{l}) - \mathtt{BN}(\underline{\mathtt{x}}_j) \;\middle|\; \exists \mathtt{y} \in \mathtt{X}.\ (\underline{\mathtt{x}}_j, \mathtt{l}, \mathtt{y}) \in \mathrm{Prem} \right\} \quad . \tag{8.2.2}$$

Next we define the function $\mathtt{FN} : \mathtt{X} \to \mathscr{P}(\mathtt{N})$ by

$$\mathtt{FN}(\mathtt{x}) = \mathtt{FN} \cup \mathtt{BN}(\mathtt{x}) \quad . \tag{8.2.3}$$

We extend this function $\mathtt{FN}$ to raw terms with explicit substitutions: let

$$\mathtt{FN} : T_{\mathbb{S}+\mathsf{sub}, \mathbf{Set}_{\mathtt{N}}}(\mathtt{X}) \to \mathscr{P}(\mathtt{N})$$

be the unique function satisfying (8.2.3) and

$$\mathtt{FN}\left(\mathtt{op}\left((\mathtt{c}_i)_{i \in [1, \mathrm{arn}(\mathsf{op})]}, \left(\left\langle \mathtt{a}_k^j \right\rangle_{k \in [1, \mathrm{dep}_j(\mathsf{op})]} \mathtt{t}_j \right)_{j \in [1, \mathrm{art}(\mathsf{op})]} \right) \right)$$
$$= \{ \mathtt{c}_i \mid i \in [1, \mathrm{arn}(\mathsf{op})] \} \cup \bigcup_{j \in [1, \mathrm{art}(\mathsf{op})]} \left(\mathtt{FN}(\mathtt{t}_j) - \left\{ \mathtt{a}_k^j \;\middle|\; k \in [1, \mathrm{dep}_j(\mathsf{op})] \right\} \right)$$

for every $\mathsf{op} \in \mathrm{Op}_{\mathbb{S}+\mathsf{sub}}$, all $\mathtt{c}_i$ (for $i \in [1, \mathrm{arn}(\mathsf{op})]$), all $\mathtt{a}_k^j$ (for $j \in [1, \mathrm{arn}(\mathsf{op})]$, $k \in [1, \mathrm{dep}_j(\mathsf{op})]$), and all $\mathtt{x}_j \in \mathtt{X}$ (for $j \in [1, \mathrm{art}(\mathsf{op})]$).

Consider some examples:

- When $\mathtt{R}$ is the input rule structure, item (b) of Figure 8.1, then $\mathtt{FN} = \{\mathtt{c}\}$ while $\mathtt{BN}(\mathtt{x}) = \{\mathtt{a}\}$, and so $\mathtt{FN}(\mathtt{x}) = \{\mathtt{a}, \mathtt{c}\}$.
- When $\mathtt{R}$ is rule structure (h) of Figure 8.1, for bound output under parallel composition, we have $\mathtt{FN} = \{\mathtt{c}\}$, and $\mathtt{FN}(\mathtt{x}) = \mathtt{FN}(\mathtt{x}') = \{\mathtt{c}\}$, while $\mathtt{FN}(\mathtt{y}') = \{\mathtt{a}, \mathtt{c}\}$. It follows that $\mathtt{FN}(\mathsf{par}(\mathtt{x}, \mathtt{y}')) = \{\mathtt{a}, \mathtt{c}\}$.
- When $\mathtt{R}$ is rule structure (i) of Figure 8.1, for communication, we have $\mathtt{FN} = \{\mathtt{c}, \mathtt{d}\}$, and
  $$\mathtt{FN}(\mathtt{x}) = \mathtt{FN}(\mathtt{x}') = \mathtt{FN}(\mathtt{y}) = \{\mathtt{c}, \mathtt{d}\} \quad \text{while } \mathtt{FN}(\mathtt{y}') = \{\mathtt{a}, \mathtt{c}, \mathtt{d}\} \quad .$$
  As for the target of the conclusion, we have $\mathtt{FN}([\mathtt{d}/\mathtt{a}]\mathtt{y}') = \{\mathtt{c}, \mathtt{d}\}$ and so
  $$\mathtt{FN}(\mathsf{par}(\mathtt{y}, [\mathtt{d}/\mathtt{a}]\mathtt{y}')) = \{\mathtt{c}, \mathtt{d}\} \quad .$$

8.2.4 Well-formedness of terms

We now define a predicate on raw terms that will be used to ensure that the binding names in the conclusion source and in the premises are not used to bind 'different' variables in the conclusion target. We define a predicate WF on $T_{\mathbb{S}+\mathsf{sub},\mathbf{Set}_{\mathbb{N}}}(\mathtt{X})$ inductively, as follows.

- For $\mathtt{x} \in \mathtt{X}$, we always let $\mathrm{WF}(\mathtt{x})$.
- For $\mathtt{t} = \mathtt{op}\left((\mathtt{c}_i)_{i\in[1,\mathrm{arn}(\mathtt{op})]}, \left(\left\langle \mathtt{a}_k^j \right\rangle_{k\in[1,\mathrm{dep}_j(\mathtt{op})]} \mathtt{t}_j\right)_{j\in[1,\mathrm{art}(\mathtt{op})]}\right)$, we let $\mathrm{WF}(\mathtt{t})$ if for all $j \in [1,\mathrm{art}(\mathtt{op})]$ we have $\mathrm{WF}(\mathtt{t}_j)$, and furthermore for all $k \in [1,\mathrm{dep}_j(\mathtt{op})]$, if $\mathtt{a}_k^j \in \mathtt{BN}$ then for all $\mathtt{x}$ appearing in $\mathtt{t}_j$ we have $\mathtt{a}_k^j \in \mathtt{FN}(\mathtt{x})$.

For instance, suppose that $\mathtt{R}$ is rule structure (i) of Figure 8.1, for communication, and consider the raw term $[\mathtt{d}/\mathtt{a}]\mathtt{y}'$ that appears in the conclusion target. By definition we have $\mathrm{WF}(\mathtt{y}')$. The name metavariable $\mathtt{a}$ appears in binding position, and it is in $\mathtt{BN}$, and so to conclude that $\mathrm{WF}([\mathtt{d}/\mathtt{a}]\mathtt{y}')$ we observe that $\mathtt{a} \in \mathtt{FN}(\mathtt{y}')$.

On the other hand, the raw term $[\mathtt{d}/\mathtt{a}]\mathtt{y}$ does not satisfy WF; although $\mathrm{WF}(\mathtt{y})$ by definition, the name metavariable $\mathtt{a}$ is in $\mathtt{BN}$ while $\mathtt{a} \notin \mathtt{FN}(\mathtt{y})$. Thus the following rule structure is bizarre: the name variable $\mathtt{a}$ binds in $\mathtt{y}'$ through the premise transition, but in the conclusion target, it is used to bind in $\mathtt{y}$.

$$\frac{\mathtt{x} \xrightarrow{\mathtt{c!d}} \mathtt{y} \qquad \mathtt{x}' \xrightarrow{\mathtt{c?(a)}} \mathtt{y}'}{\mathsf{par}(\mathtt{x},\mathtt{x}') \xrightarrow{\tau} \mathsf{par}([\mathtt{d}/\mathtt{a}]\mathtt{y},\mathtt{y}')}$$

8.2.5 The И-GSOS$^+$ format

We are now in a position to introduce the final set of conditions on the rule structures that we consider. Conditions И-GSOS$^+$-6–12 on rule structures are presented, using the function $\mathtt{FN}$ and the predicate WF, in Figure 8.3.

Definition 8.2.4. A rule structure is in И-GSOS$^+$ *format* if it satisfies Conditions И-GSOS$^+$-1–12 of Figures 8.2 and 8.3.

8.3 The conditions are necessary: Examples and counter-examples

It is straightforward to check that all the rule structures for the π-calculus given in Figure 8.1 are in the И-GSOS$^+$ format. In this section we will consider some rule structures that are *not* in the И-GSOS$^+$ format, and explain why they should indeed be disallowed.

Conditions И-GSOS$^+$-1–5 are very similar to Conditions GSOS$^+$-1–5, and as such they are necessary for well-behavedness, specifically for congruence of bisimilarity, for the reasons discussed by Bloom *et al.* [1995, App. A]. So we concentrate here on Conditions И-GSOS$^+$-6–12.

As we now explain, all these conditions, bar И-GSOS$^+$-10, are necessary if wide open bisimilarity is to be a congruence. Some of the conditions also help to ensure that Axioms $И_g1$ and $И_g2$ hold of the induced system.

In what follows, we will make use of the following two π-calculus terms

$$t_1 = \mathsf{nil} \qquad t_2 = \mathsf{match}(a, b, \mathsf{nil})$$

which are wide open bisimilar. Neither term has any behaviour, under any renaming; the only difference is that t_1 has no free name variables while t_2 has a and b free.

И-GSOS$^+$-6. The binding variables in the conclusion source are not also free.

$$\forall j \in [1, \mathrm{art}(\underline{\mathtt{op}})],\ k \in [1, \mathrm{dep}_j(\underline{\mathtt{op}})].\ \underline{\mathtt{a}}^j_k \notin \mathtt{FN}$$

И-GSOS$^+$-7. For each term parameter in the conclusion source, the binding variables are all distinct.

$$\forall j \in [1, \mathrm{art}(\underline{\mathtt{op}})],\ k, k' \in [1, \mathrm{dep}_j(\underline{\mathtt{op}})].\ \underline{\mathtt{a}}^j_k = \underline{\mathtt{a}}^j_{k'} \implies k = k'$$

И-GSOS$^+$-8. Bound names in premise labels are fresh for the premise sources.

$$\forall (\mathtt{x}, \mathtt{l}, \mathtt{y}) \in \mathrm{Prem}.\ \mathsf{bn}(\mathtt{l}) \cap \mathtt{FN}(\mathtt{x}) = \emptyset$$

И-GSOS$^+$-9. Free names of the conclusion label appear in the conclusion source or in the premises.

$$\mathsf{fn}(\underline{\mathtt{l}}) \subseteq \mathtt{FN}$$

И-GSOS$^+$-10. Bound names of the conclusion label are fresh for the conclusion source.

$$\mathsf{bn}(\underline{\mathtt{l}}) \cap \mathtt{FN} = \emptyset$$

И-GSOS$^+$-11. Renamings in the conclusion target only affect relevant names.

$$\mathrm{WF}(\mathtt{tar})$$

И-GSOS$^+$-12. No names become unbound in the induced transition.

$$\mathtt{FN}(\mathtt{tar}) \subseteq \mathtt{FN} \cup \mathsf{bn}(\underline{\mathtt{l}})$$

Figure 8.3: Further conditions on a rule structure for name-passing.

Condition И-GSOS$^+$-6. This condition helps to ensure that bisimilarity is a congruence. Suppose that we add an operator if-fresh to the π-calculus, with arities given by

$$\mathrm{arn}(\text{if-fresh}) = 1 \quad \mathrm{art}(\text{if-fresh}) = 1 \quad \mathrm{dep}_1(\text{if-fresh}) = 1 \quad .$$

Let the semantics of if-fresh be given by the following rule structure, with term variables $\mathtt{X} = \{\mathtt{x}, \mathtt{y}\}$ and name variable $\mathtt{N} = \{\mathtt{a}\}$.

$$\frac{\mathtt{x} \xrightarrow{\tau} \mathtt{y}}{\text{if-fresh}(\mathtt{a}, \langle \mathtt{a} \rangle \mathtt{x}) \xrightarrow{\tau} \mathtt{y}} \tag{8.3.1}$$

Consider this extension of the π-calculus, under the nominal logic semantics proposed in Section 8.1.2. If $a \# \langle b \rangle x$ then the term $\text{if-fresh}(a, \langle b \rangle x)$ behaves exactly as x with regard to silent actions; otherwise $\text{if-fresh}(a, \langle b \rangle x)$ cannot reduce.

The context $\text{if-fresh}(a, \langle b \rangle \mathsf{tau}(-))$ distinguishes t_1 from t_2. For

$$\text{if-fresh}(a, \langle b \rangle \mathsf{tau}(t_1)) \ = \ \text{if-fresh}(a, \langle b \rangle \mathsf{tau}(\mathsf{nil})) \ = \ \text{if-fresh}(a, \langle a \rangle \mathsf{tau}(\mathsf{nil})) \quad .$$

(The first equality is the defn. of t_1; the second equality is α-equivalence.) So $\text{if-fresh}(a, \langle b \rangle \mathsf{tau}(t_1))$ can perform a silent action. On the other hand, $\text{if-fresh}(a, \langle b \rangle \mathsf{tau}(t_2))$ *cannot* perform a silent action because there is no term t with $\langle b \rangle \mathsf{tau}(t_2) = \langle a \rangle \mathsf{tau}(t)$.

Condition И-GSOS$^+$-6 disallows the rule structure (8.3.1), because $\mathtt{FN} = \{\mathtt{a}\}$ and a also appears as a binder in the conclusion source.

The alternative rule structure

$$\frac{\mathtt{x} \xrightarrow{\tau} \mathtt{y}}{\mathsf{if\text{-}fresh}(\mathtt{b}, \langle \mathtt{a} \rangle \mathtt{x}) \xrightarrow{\tau} \mathtt{y}}$$

is in the И-GSOS$^+$ format but gives rise to a different semantics. If we introduce this rule structure instead of (8.3.1) then the term $\mathsf{if\text{-}fresh}(b, \langle a \rangle x)$ can perform exactly the silent actions of x, whether or not $a = b$. The implicit side condition, which enforces that $a \neq b$, is somewhat redundant, since for any $b \in \mathcal{N}$, and any π-calculus term t, there is a name $a \in \mathcal{N}$ and a term t' such that

$$\mathsf{if\text{-}fresh}(b, \langle b \rangle t) = \mathsf{if\text{-}fresh}(b, \langle a \rangle t') \quad .$$

Condition И-GSOS$^+$-7. This condition helps to ensure that bisimilarity is a congruence, for a similar reason to that given for Condition И-GSOS$^+$-6. Suppose that we add an operator if-fresh2 to the π-calculus, with arities given by

$$\mathrm{arn}(\mathsf{if\text{-}fresh2}) = 0 \quad \mathrm{art}(\mathsf{if\text{-}fresh2}) = 1 \quad \mathrm{dep}_1(\mathsf{if\text{-}fresh2}) = 2 \quad .$$

Let the semantics of if-fresh2 be given by the following rule structure, which clearly violates Condition И-GSOS$^+$-7. The rule structure has term variables $\mathtt{X} = \{\mathtt{x}, \mathtt{y}\}$ and a name variable $\mathtt{N} = \{\mathtt{a}\}$.

$$\frac{\mathtt{x} \xrightarrow{\tau} \mathtt{y}}{\mathsf{if\text{-}fresh2}(\langle \mathtt{a}\ \mathtt{a} \rangle \mathtt{x}) \xrightarrow{\tau} \mathtt{y}}$$

Informally, if $b \# \langle a \rangle x$ then the term $\mathsf{if\text{-}fresh2}(\langle b\ a \rangle x)$ behaves exactly as x with regard to silent actions; otherwise $\mathsf{if\text{-}fresh}(\langle b\ a \rangle x)$ cannot reduce.

The context $\mathsf{if\text{-}fresh2}(\langle b\ a \rangle \mathsf{tau}(-))$ distinguishes t_1 from t_2. For

$$\mathsf{if\text{-}fresh2}(\langle b\ a \rangle \mathsf{tau}(t_1)) \ = \ \mathsf{if\text{-}fresh2}(\langle b\ a \rangle \mathsf{tau}(\mathsf{nil})) \ = \ \mathsf{if\text{-}fresh2}(\langle a\ a \rangle \mathsf{tau}(\mathsf{nil}))$$

and so the term $\mathsf{if\text{-}fresh2}(\langle b\ a \rangle \mathsf{tau}(t_1))$ can perform a silent action, while the term $\mathsf{if\text{-}fresh2}(\langle b\ a \rangle \mathsf{tau}(t_2))$ cannot.

Condition И-GSOS$^+$-8. Condition И-GSOS$^+$-8 is necessary to ensure that bisimilarity is a congruence, as the following example illustrates. Suppose we add an operator tau-if-bout to the π-calculus, with arities given by

$$\mathrm{arn}(\mathsf{tau\text{-}if\text{-}bout}) = 1 \quad \mathrm{art}(\mathsf{tau\text{-}if\text{-}bout}) = 1 \quad \mathrm{dep}_1(\mathsf{tau\text{-}if\text{-}bout}) = 0 \quad .$$

Let the semantics of tau-if-bout be given by the following rule structure, which has term variables $\mathtt{X} = \{\mathtt{x}, \mathtt{y}\}$ and name variables $\mathtt{N} = \{\mathtt{a}, \mathtt{c}\}$.

$$\frac{\mathtt{x} \xrightarrow{\mathtt{c!(a)}} \mathtt{y}}{\mathsf{tau\text{-}if\text{-}bout}(\mathtt{a}, \mathtt{x}) \xrightarrow{\tau} \mathtt{x}} \tag{8.3.2}$$

So if a process x can perform a bound output of the specific value a then $\mathsf{tau\text{-}if\text{-}bout}(a, x)$ can perform a silent action.

Now, the context $\mathsf{tau\text{-}if\text{-}bout}(a, \mathsf{restrct}(\langle b \rangle \mathsf{out}(c, b, (-))))$ can distinguish t_1 from t_2. For $\mathsf{restrct}(\langle b \rangle \mathsf{out}(c, b, t_1))$ can perform a bound output (of a) on channel c, while $\mathsf{restrct}(\langle b \rangle \mathsf{out}(c, b, t_2))$ cannot.

Condition И-GSOS$^+$-8 disallows the rule structure of (8.3.2) because $\mathtt{FN}(\mathtt{x}) = \{\mathtt{a}, \mathtt{c}\}$ while a also appears as a binder in the premise label.

Condition И-GSOS$^+$-9. In a moment we will explain why Condition И-GSOS$^+$-9 is necessary to ensure that wide open bisimilarity is a congruence; first we illustrate how Condition И-GSOS$^+$-9 helps to enforce Axiom $И_g1$. Without this condition we could violate Axiom $И_g1$ by adding an operator noise to the π-calculus, with arities given by

$$\mathsf{arn}(\mathsf{noise}) = 0 \quad \mathsf{art}(\mathsf{noise}) = 0$$

and with semantics given by the following rule structure, that violates Condition И-GSOS$^+$-9. The rule structure has no term variables and has name variables $\mathtt{N} = \{\mathtt{c}, \mathtt{d}\}$.

$$\frac{}{\mathsf{noise} \xrightarrow{\mathtt{c!d}} \mathsf{noise}}$$

The process noise will repeatedly output data on any distinct channel, violating Axiom $И_g1$.

Indeed, Conditions И-GSOS$^+$-1–И-GSOS$^+$-12 enforce that any nullary operator (that takes no parameters) can only perform silent actions in the intended model.

We now introduce a more sophisticated example to illustrate that Condition И-GSOS$^+$-9 is necessary if we are to guarantee that wide open bisimilarity is a congruence. Consider an operator bout-to-out with arities

$$\mathsf{arn}(\text{bout-to-out}) = 0 \quad \mathsf{art}(\text{bout-to-out}) = 1 \quad \mathsf{dep}_1(\text{bout-to-out}) = 0$$

and with semantics given by the following rule structure, which has term variables $\mathtt{X} = \{\mathtt{x}, \mathtt{y}\}$, and has name variables $\mathtt{N} = \{\mathtt{a}, \mathtt{c}\}$.

$$\frac{\mathtt{x} \xrightarrow{\mathtt{c!(a)}} \mathtt{y}}{\text{bout-to-out}(\mathtt{x}) \xrightarrow{\mathtt{c!a}} \mathtt{y}} \tag{8.3.3}$$

So: whenever x can perform a bound output then bout-to-out(x) can perform a free output with the same channel and data.

The induced behaviour will once again violate Axiom $И_g1$, but here, moreover, wide open bisimilarity will not be a congruence: the context bout-to-out(restrct($\langle a \rangle$ out$(c, a, (-))$)) will distinguish between t_1 and t_2. For the term

$$\begin{aligned}\text{bout-to-out}(\text{restrct}(\langle a \rangle \,\text{out}(c, a, t_1))) &= \text{bout-to-out}(\text{restrct}(\langle a \rangle \,\text{out}(c, a, \text{nil}))) \\ &= \text{bout-to-out}(\text{restrct}(\langle b \rangle \,\text{out}(c, b, \text{nil})))\end{aligned}$$

can output b on channel c, while bout-to-out(restrct($\langle a \rangle$ out(c, a, t_2))) cannot output b.

Condition И-GSOS$^+$-9 disallows the rule structure of (8.3.3) because $\mathtt{FN} = \{\mathtt{c}\}$ while $\mathsf{fn}(\underline{1}) = \{\mathtt{c}, \mathtt{a}\}$.

Condition И-GSOS$^+$-10. In the presence of the other conditions, Condition И-GSOS$^+$-10 is redundant — indeed, we will not use this condition in the proofs of Section 8.4. Moreover, a rule that violates this condition can never be applied, according to the nominal logic semantics introduced in Section 8.1.2. By way of example, consider an operator out-to-bout with arities

$$\mathsf{arn}(\text{out-to-bout}) = 0 \quad \mathsf{art}(\text{out-to-bout}) = 1 \quad \mathsf{dep}_1(\text{out-to-bout}) = 0 \quad .$$

Suppose the semantics of out-to-bout is given by the following rule, which violates Condition И-GSOS$^+$-10. Term variables are $\mathtt{X} = \{\mathtt{x}, \mathtt{y}\}$ and name variables are $\mathtt{N} = \{\mathtt{c}, \mathtt{d}\}$.

$$\frac{\mathtt{x} \xrightarrow{\mathtt{c!d}} \mathtt{y}}{\text{out-to-bout}(\mathtt{x}) \xrightarrow{\mathtt{c!(d)}} \mathtt{y}}$$

The implicit side conditions on this rule are that (i) $\mathtt{c} \neq \mathtt{d}$, and (ii) $\mathtt{d} \# \mathtt{x}$. So: for any π-calculus term t, out-to-bout(t) can only progress if t can perform a free output of fresh data. But Axiom $И_g1$ says that free output data must not be fresh, and so the term out-to-bout(t) can do nothing — the same behaviour would be achieved if the rule structure was omitted.

Condition И-GSOS$^+$-11. The WF construction that is used in Condition И-GSOS$^+$-11 ensures that bisimilarity will be a congruence. To see this, consider an operator strange with

$$\mathsf{arn}(\mathsf{strange}) = 0 \quad \mathsf{art}(\mathsf{strange}) = 2 \quad \mathsf{dep}_1(\mathsf{strange}) = 1 \quad \mathsf{dep}_1(\mathsf{strange}) = 0 \quad .$$

Consider a semantics described by the following rule structure, which violates Condition И-GSOS$^+$-11. The rule structure has term variables $\mathtt{X} = \{\mathtt{x}, \mathtt{x}'\}$ and name variable $\mathtt{N} = \{\mathtt{a}\}$.

$$\frac{\text{—}}{\mathsf{strange}(\langle \mathsf{a} \rangle\, \mathtt{x}, \mathtt{x}') \xrightarrow{\tau} \mathsf{restrct}(\langle \mathsf{a} \rangle\, \mathsf{par}(\mathtt{x}, \mathtt{x}'))}$$

So the scope of the binder $\mathtt{a}$ can extrude during a τ transition, to include $\mathtt{x}'$. The context $\mathsf{strange}(\langle b \rangle\, (-), \mathsf{out}(c, a, \mathsf{nil}))$ distiguishes between t_1 and t_2. For

$$\begin{aligned} \mathsf{strange}(\langle b \rangle\, t_1, \mathsf{out}(c, a, \mathsf{nil})) &= \mathsf{strange}(\langle b \rangle\, \mathsf{nil}, \mathsf{out}(c, a, \mathsf{nil})) \\ &= \mathsf{strange}(\langle a \rangle\, \mathsf{nil}, \mathsf{out}(c, a, \mathsf{nil})) \end{aligned}$$

and so $\mathsf{strange}(\langle b \rangle\, t_1, \mathsf{out}(c, a, \mathsf{nil}))$ can perform a silent action and follow this with a bound output on channel c; the term $\mathsf{strange}(\langle b \rangle\, t_2, \mathsf{out}(c, a, \mathsf{nil}))$ cannot perform this sequence of actions.

Condition И-GSOS$^+$-12. Condition И-GSOS$^+$-12 ensures both that Axiom И$_g$2 holds of the induced transition system, and also that wide open bisimilarity will be a congruence. Consider the semantics described by the following rule structure, which has a term variable $\mathtt{X} = \{\mathtt{x}\}$ and a name variable $\mathtt{N} = \{\mathtt{a}\}$.

$$\frac{\text{—}}{\mathsf{restrct}(\langle \mathsf{a} \rangle\, \mathtt{x}) \xrightarrow{\tau} \mathtt{x}} \tag{8.3.4}$$

Informally: restrictions can be silently forgotten. This behaviour violates Axiom И$_g$2 because the name a is fresh for the term $\mathsf{restrct}(\langle a \rangle\, \mathsf{out}(c, a, \mathsf{nil}))$, and this term can perform a silent action to become the term $\mathsf{out}(c, a, \mathsf{nil})$, for which a is no longer fresh.

Moreover, wide open bisimilarity will not be a congruence for the induced semantics: the context $\mathsf{restrct}(\langle a \rangle\, \mathsf{out}(c, a, (-)))$ can distinguish t_1 from t_2. For we have

$$\mathsf{restrct}(\langle a \rangle\, \mathsf{out}(c, a, t_1)) \;=\; \mathsf{restrct}(\langle a \rangle\, \mathsf{out}(c, a, \mathsf{nil})) \;=\; \mathsf{restrct}(\langle b \rangle\, \mathsf{out}(c, b, \mathsf{nil}))$$

so that $\mathsf{restrct}(\langle a \rangle\, \mathsf{out}(c, a, t_1))$ can perform a silent action and follow this with output of b on c. But $\mathsf{restrct}(\langle a \rangle\, \mathsf{out}(c, a, t_2))$ can perform no such sequence of transitions.

The rule structure in (8.3.4) violates Condition И-GSOS$^+$-12 because $\mathtt{FN} = \mathsf{bn}(\tau) = \emptyset$, while $\mathtt{FN}(\mathtt{tar}) = \mathtt{FN}(\mathtt{x}) = \{\mathtt{a}\}$.

8.4 Inducing abstract rules from rule structures

In the final section of this chapter, we explain how a set $\mathscr{R}$ of rule structures induces an abstract rule

$$[\![\mathscr{R}]\!] : \Sigma_{\mathbb{S},\mathbf{Nom}}(|-| \times B_g|-|) \to B_g T_{\mathbb{S},\mathbf{Nom}}|-| \tag{8.4.1}$$

in the sense of Definition 6.2.11, for which the induced monad lifting corresponds to the intended model of the nominal logic theory induced by $\mathscr{R}$ introduced in Section 8.1.2. Here, we write $|-|$ for the forgetful functor **NomSub** $\to$ **Nom**. The endofunctor B_g is the endofunctor on **Nom** for ground bisimulation considered in equations 7.1.7; recall that we have $B_g = \mathscr{P}_{\mathrm{sb}} L_g$.

Most of the work involved is in deriving a natural transformation of the form (8.4.1) for a single rule structure. In Section 8.4.1 we explain how a single rule structure gives rise to a family of functions in the shape of (8.4.1); we delay proving that these functions are equivariant and that the family is natural until Section 8.4.3. Central to the derivation of this family is the notion of

valuation, which amounts to the usual notion of valuation for the nominal logic formula $\Phi_{\mathtt{R}}$ for a rule structure $\mathtt{R}$. Section 8.4.2 is dedicated to exploring a special class of valuations, which are useful in the proofs of Section 8.4.3.

We conclude this section in Section 8.4.4 by explaining how this derivation of abstract rules extends to sets of rule structures, and by explaining why one can deduce that the intended model is well-behaved.

In the first three subsections of this section we fix a rule structure $\mathtt{R}$ in the ∇-GSOS$^+$ format. As in Section 8.2, we let Prem be the set of premises of $\mathtt{R}$, and let $(\mathtt{src}, \underline{\mathtt{l}}, \mathtt{tar})$ be the conclusion of $\mathtt{R}$. Then we have an operator $\underline{\mathtt{op}} \in \mathsf{Op}_{\mathbb{S}}$ and

$$\begin{aligned} &\underline{\mathtt{c}}_i && i \in [1, \mathrm{arn}(\underline{\mathtt{op}})] \\ &\underline{\mathtt{x}}_j && j \in [1, \mathrm{art}(\underline{\mathtt{op}})] \\ &\underline{\mathtt{a}}^j_k && j \in [1, \mathrm{art}(\underline{\mathtt{op}})],\ k \in [1, \mathrm{dep}_j(\underline{\mathtt{op}})] \end{aligned}$$

such that

$$\mathtt{src} = \underline{\mathtt{op}} \left((\underline{\mathtt{c}}_i)_{i \in [1, \mathrm{arn}(\mathtt{op})]}, \left(\left\langle \underline{\mathtt{a}}^j_k \right\rangle_{k \in [1, \mathrm{dep}_j(\mathtt{op})]} \underline{\mathtt{x}}_j \right)_{j \in [1, \mathrm{art}(\mathtt{op})]} \right) \quad .$$

8.4.1 Instantiating parts of the rule

We begin this subsection by introducing a notion of valuation; this is a valuation for the nominal logic formula $\Phi_{\mathtt{R}}$ that was associated to the rule $\mathtt{R}$ in Section 8.1.2, with the extra requirement that the nominal set in which the term variables are valued must be equipped with a nominal substitution structure.

We then proceed to explain how such a valuation enables one to create instances of terms of raw syntax, and explicit substitutions. One can also use the valuation to instantiate behaviours, and the premises of $\mathtt{R}$ in particular. Thus we are able to recognise when a valuation matches with an element of the domain, $\Sigma_{\mathbb{S},\mathbf{Nom}}(|X| \times B_{\mathrm{g}}|X|)$, of the natural transformation of (8.4.1). We are further able to use a valuation to instantiate the conclusion label and conclusion target to obtain an element of the codomain, $B_{\mathrm{g}} T_{\mathbb{S},\mathbf{Nom}}|X|$, of the natural transformation of (8.4.1).

Valuations. A *valuation* $\mathscr{V}$ of the pair $(\mathtt{N}, \mathtt{X})$ into a nominal substitution $X \in \mathbf{NomSub}$ is given by two functions of sets: a valuation of name metavariables, $\mathscr{V}_{\mathrm{n}} : \mathtt{N} \to \mathcal{N}$, and a valuation of term variables, $\mathscr{V}_{\mathrm{t}} : \mathtt{X} \to X$. (That is, $\mathscr{V}_{\mathrm{t}}$ is a function from the set $\mathtt{X}$ to the set underlying X.)

Instantiating terms. A valuation $\mathscr{V}$ into some X can be used to instantiate terms of raw syntax, with explicit substitutions, into abstract syntax. This is done via the composite function

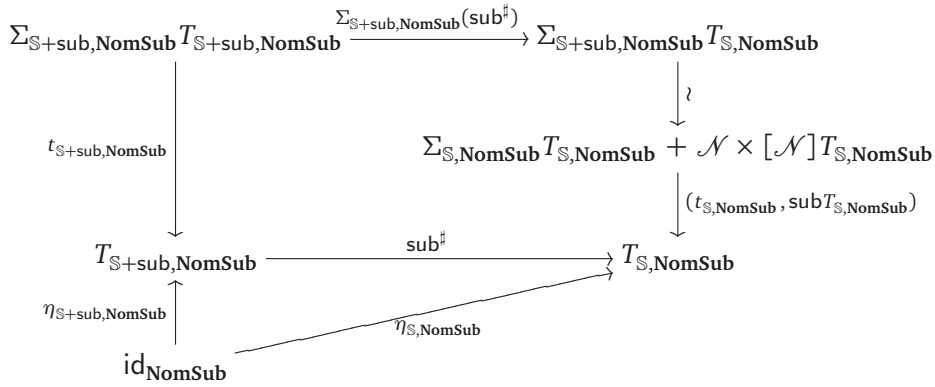
$$\begin{array}{c} T_{\mathbb{S}+\mathsf{sub},\mathbf{Set}_{\mathtt{N}}}(\mathtt{X}) \\ \downarrow \\ T_{\mathbb{S}+\mathsf{sub},\mathbf{Set}_{\mathtt{N}}}(U^{\mathbf{NS}}_{\mathbf{S}} X) \\ \downarrow \\ U^{\mathbf{NS}}_{\mathbf{S}} T_{\mathbb{S}+\mathsf{sub},\mathbf{NomSub}}(X) \\ \downarrow \\ U^{\mathbf{NS}}_{\mathbf{S}} T_{\mathbb{S},\mathbf{NomSub}}(X) \\ \downarrow \\ U^{\mathbf{N}}_{\mathbf{S}} T_{\mathbb{S},\mathbf{Nom}}(U^{\mathbf{NS}}_{\mathbf{N}} X) \end{array} \tag{8.4.2}$$

mapping a raw term with explicit substitutions, and with name metavariables from $\mathtt{N}$ and term variables from $\mathtt{X}$, into an abstract term with the explicit substitutions carried out, and with variables

from X. (Here, we are using the forgetful functors $U_{\mathbf{S}}^{\mathbf{NS}} : \mathbf{NomSub} \to \mathbf{Set}$, $U_{\mathbf{S}}^{\mathbf{N}} : \mathbf{Nom} \to \mathbf{Set}$, and $U_{\mathbf{N}}^{\mathbf{NS}} : \mathbf{NomSub} \to \mathbf{Nom}$; the last functor is elsewhere denoted $|-|$.)

The first function in the composite (8.4.2) instantiates the term variables according to $\mathscr{V}_{\mathrm{t}}$, using the functorial action of $T_{\mathbb{S}+\mathsf{sub},\mathbf{Set}_{\mathbb{N}}}$. The second function instantiates the name metavariables, and quotients by α-equivalence: in Section 7.4 we explained that a function $\mathscr{V}_{\mathrm{n}} : \mathbb{N} \to \mathscr{N}$ induces a monad morphism $(\mathbf{Set}_{\mathbb{N}}, \mathbf{T}_{\mathbb{S},\mathbf{Set}_{\mathbb{N}}}) \to (\mathbf{NomSub}, \mathbf{T}_{\mathbb{S},\mathbf{NomSub}})$, and it is the natural transformation $T_{\mathbb{S},\mathbf{Set}_{\mathbb{N}}} U_{\mathbf{S}}^{\mathbf{NS}} \to U_{\mathbf{S}}^{\mathbf{NS}} T_{\mathbb{S},\mathbf{Nom}}$ which is key to this second function.

The third function in the composite (8.4.2) evaluates the explicit substitutions using the mechanism built into the nominal substitution X. Formally, this step can be performed by considering the unique natural transformation $\mathsf{sub}^\sharp : T_{\mathbb{S}+\mathsf{sub},\mathbf{NomSub}} \to T_{\mathbb{S},\mathbf{NomSub}}$ making the following diagram commute.



(On the right of the diagram, we are using the natural transformation for substitution, $\mathsf{sub} : \mathscr{N} \times [\mathscr{N}](-) \to (-)$, between endofunctors on **NomSub**.)

The fourth and final function in the composite (8.4.2) arises since, as explained in Section 7.4, the monad $\mathbf{T}_{\mathbb{S},\mathbf{NomSub}}$ is a lifting of the monad $\mathbf{T}_{\mathbb{S},\mathbf{Nom}}$ along the forgetful functor $U_{\mathbf{N}}^{\mathbf{NS}} : \mathbf{NomSub} \to \mathbf{Nom}$.

We denote the composite function of (8.4.2) by the symbol $\mathscr{V}$.

Instantiating behaviour. Consider a nominal set X and a name valuation function $\mathscr{V}_{\mathrm{n}} : \mathbb{N} \to \mathscr{N}$. Every label-element pair $(l \in Lab_{\mathrm{g}}(\mathbb{N}), x \in X)$ is associated with a behaviour in $L_g|X|$ that we denote $\mathscr{V}(l)(x)$, and which is given as follows.

If $l = \mathsf{c?(d)}$,
$\mathscr{V}(l)(x) = \mathsf{inj}_{\mathsf{binp}}(\mathscr{V}_{\mathrm{n}}(\mathsf{c}), \langle \mathscr{V}_{\mathrm{n}}(\mathsf{d}) \rangle (x))$.

If $l = \mathsf{c!d}$,
$\mathscr{V}(l)(x) = \mathsf{inj}_{\mathsf{out}}(\mathscr{V}_{\mathrm{n}}(\mathsf{c}), \mathscr{V}_{\mathrm{n}}(\mathsf{d}), x)$.

If $l = \mathsf{c!(d)}$,
$\mathscr{V}(l)(x) = \mathsf{inj}_{\mathsf{bout}}(\mathscr{V}_{\mathrm{n}}(\mathsf{c}), \langle \mathscr{V}_{\mathrm{n}}(\mathsf{d}) \rangle (x))$.

If $l = \tau$,
$\mathscr{V}(l)(x) = \mathsf{inj}_{\mathsf{tau}}(x)$.

Instantiating the premises. For each $j \in [1, \mathrm{art}(\underline{\mathrm{op}})]$ we use a valuation $\mathscr{V}$ into X to instantiate those premises with source $\underline{\mathrm{x}}_j$, by defining an element of $B_g|X|$ that we notate $\mathscr{V}(\mathrm{Prem}_j)$. We make this definition by collecting instantiations of all the relevant premises, using the behaviour instantiation technique of the last paragraph:

$$\mathscr{V}(\mathrm{Prem}_j) = \left\{ \mathscr{V}(l)(\mathscr{V}_{\mathrm{t}}(\mathrm{y})) \;\middle|\; (\underline{\mathrm{x}}_j, l, \mathrm{y}) \in \mathrm{Prem} \right\} \quad .$$

This set is finite, and so certainly support bounded. The intention is that $\mathscr{V}(\mathrm{Prem}_j)$ is the smallest behaviour that satisfies the premises corresponding to element $\underline{\mathrm{x}}_j$, under the valuation $\mathscr{V}$.

Instantiating the conclusion source and the premises. Consider a nominal substitution X, and let s be an element of $\Sigma_{\mathbb{S},\mathbf{Nom}}(|X| \times B_g|X|)$. We say that a valuation $\mathscr{V}$ into X is *an instantiation into* s if $\mathscr{V}_{\mathrm{n}}$ is injective, and

$$\mathscr{V}_{\mathrm{n}}(\mathsf{bn}(\underline{\mathtt{l}})) \cap \mathsf{supp}(s) = \emptyset$$

and if there are $\beta_j \in B_g|X|$ for each $j \in [1, \mathrm{art}(\underline{\mathtt{op}})]$ such that

$$s = \underline{\mathtt{op}}\left((\mathscr{V}_{\mathrm{n}}(\underline{\mathtt{c}}_i))_{i\in[1,\mathrm{arn}(\underline{\mathtt{op}})]}, \left(\left\langle \mathscr{V}_{\mathrm{n}}(\underline{\mathtt{a}}_k^j) \right\rangle_{k\in[1,\mathrm{dep}_j(\underline{\mathtt{op}})]} (\mathscr{V}_{\mathrm{t}}(\underline{\mathtt{x}}_j), \beta_j) \right)_{j\in[1,\mathrm{art}(\underline{\mathtt{op}})]} \right)$$

and such that $\mathscr{V}(\mathrm{Prem}_j) \subseteq \beta_j$, for all $j \in [1, \mathrm{art}(\underline{\mathtt{op}})]$.

Instantiating the conclusion label and target. Each valuation $\mathscr{V}$ induces an archetypal behaviour, denoted $\mathscr{V}(\underline{\mathtt{l}}, \mathtt{tar})$, in $L_g T_{\mathbb{S},\mathbf{Nom}}|X|$, by

$$\mathscr{V}(\underline{\mathtt{l}}, \mathtt{tar}) = \mathscr{V}(\underline{\mathtt{l}})(\mathscr{V}(\mathtt{tar})) \quad .$$

Proposition 8.4.3. *Consider a nominal substitution X, and let s be an element of $\Sigma_{\mathbb{S},\mathbf{Nom}}(|X| \times B_g|X|)$. If $\mathscr{V}$ is an instantiation into s, and C supports s in $\Sigma_{\mathbb{S},\mathbf{Nom}}(|X| \times B_g|X|)$, then C also supports $\mathscr{V}(\underline{\mathtt{l}}, \mathtt{tar})$ in $L_g T_{\mathbb{S},\mathbf{Nom}}|X|$.*

This basic property, which relies on Condition ν-GSOS$^+$-9 and 12, is proved in Appendix 8.A.

Sets of induced behaviours. For each set X and each $s \in \Sigma_{\mathbb{S},\mathbf{Nom}}(|X| \times B_g|X|)$ we consider the set $[\![\mathtt{R}]\!]_X(s) \subseteq L_g T_{\mathbb{S},\mathbf{Nom}}|X|$ of those archetypal results that arise from instantiating the conclusion label and target using instantiations into s. Precisely, we let

$$[\![\mathtt{R}]\!]_X(s) = \left\{ b \in L_g T_{\mathbb{S},\mathbf{Nom}}|X| \;\middle|\; \begin{array}{l} \text{There is an instantiation } \mathscr{V} \\ \quad \text{of rule } \mathtt{R} \text{ into } s \\ \quad \text{such that } b = \mathscr{V}(\underline{\mathtt{l}}, \mathtt{tar}) \end{array} \right\} . \tag{8.4.4}$$

It follows immediately from Prop. 8.4.3 that the set $[\![\mathtt{R}]\!]_X(s) \subseteq L_g T_{\mathbb{S},\mathbf{Nom}}|X|$ is support bounded (by $\mathsf{supp}(s)$) — so we know that $[\![\mathtt{R}]\!]_X(s)$ is in $B_g T_{\mathbb{S},\mathbf{Nom}}|X|$.

8.4.2 Valuations that provide fresh binders

In the theory of abstract syntax, there is usually an expectation that binders can be chosen so as to be sufficiently fresh, and indeed making such choices for binders tends to make reasoning easier. In this short subsection we explain the extent to which every valuation can be replaced by one that does provide sufficiently fresh binders.

Throughout this discussion we fix a nominal substitution X.

Definition 8.4.5. Let $\mathscr{V}$ be a valuation into X. Let C be finite set of names. When $\mathscr{V}_{\mathrm{n}} : \mathtt{N} \to \mathscr{N}$ is injective and $\mathscr{V}_{\mathrm{n}}(\mathtt{BN} \cup \mathsf{bn}(\underline{\mathtt{l}})) \cap C = \emptyset$ then we say that $\mathscr{V}$ *provides fresh binders* for C.

For any element $s \in \Sigma_{\mathbb{S},\mathbf{Nom}}(|X| \times B_g|X|)$, if $\mathscr{V}$ provides fresh binders for $\mathsf{supp}(s)$ then we say that $\mathscr{V}$ *provides fresh binders* for s.

For any valuation $\mathscr{V}$ that provides fresh binders for an element $s \in \Sigma_{\mathbb{S},\mathbf{Nom}}(|X| \times B_g|X|)$, we have the following useful property.

Proposition 8.4.6. *Let $\mathcal{V}$ be a valuation that provides fresh binders for an element $s \in \Sigma_{\mathbb{S},\mathbf{Nom}}(|X| \times B_g|X|)$ of component type $\underline{\mathrm{op}}$. Then there exist*

$$\begin{aligned} c_i &\in \mathcal{N} \qquad \text{for } i \in [1, \mathrm{arn}(\underline{\mathrm{op}})] \\ x_j \in X,\ \beta_j &\in B_g|X| \qquad \text{for } j \in [1, \mathrm{art}(\underline{\mathrm{op}})] \end{aligned}$$

such that

$$s = \underline{\mathrm{op}}\left((c_i)_{i\in[1,\mathrm{arn}(\underline{\mathrm{op}})]}, \left(\left\langle \mathcal{V}_{\mathrm{n}}(\underline{\mathtt{a}}^j_k) \right\rangle_{k\in[1,\mathrm{dep}_j(\underline{\mathrm{op}})]} (x_j, \beta_j) \right)_{j\in[1,\mathrm{art}(\underline{\mathrm{op}})]} \right) \quad .$$

Proof notes. This follows from the definition of name abstraction, and Condition И-GSOS$^+$-7. □

Lemma 8.4.7. Consider some $s \in \Sigma_{\mathbb{S},\mathbf{Nom}}(|X| \times B_g|X|)$, and $C \subseteq_{\mathrm{f}} \mathcal{N}$. For any instantiation $\mathcal{V}$ into s, there is another instantiation $\mathcal{V}'$ into s that provides fresh binders for C and which is such that $\mathcal{V}(\underline{\mathtt{l}}, \mathtt{tar}) = \mathcal{V}'(\underline{\mathtt{l}}, \mathtt{tar})$.

Appendix 8.B is dedicated to a proof of this lemma. Conditions И-GSOS$^+$-6–8 and И-GSOS$^+$-11–12 are involved here.

Worked example. We illustrate the developments of this subsection. We will suppose that the rule structure $\mathtt{R}$ is the rule (i) of Figure 8.1, for communication, and we will consider the element $s \in \Sigma_{\mathbb{S},\mathbf{Nom}}(|T_{\mathbb{S},\mathbf{NomSub}}\emptyset| \times B_g|T_{\mathbb{S},\mathbf{NomSub}}\emptyset|)$ given by

$$s \;=\; \mathsf{par}\left(\begin{array}{l} \big(\mathsf{out}(c,d,\mathsf{out}(c,a,\mathsf{nil})), \{\mathsf{inj}_{\mathsf{out}}(c,d,\mathsf{out}(c,a,\mathsf{nil}))\}\big)\,, \\ \Big(\mathsf{inp}(c,\langle a\rangle(\mathsf{out}(c,a,\mathsf{nil}))), \big\{\mathsf{inj}_{\mathsf{binp}}(c,\langle a\rangle(\mathsf{out}(c,a,\mathsf{nil})))\big\}\Big) \end{array} \right) \quad .$$

(Here a, c and d are fixed name constants from $\mathcal{N}$.) This element would arise during the process of parameterised recursion for defining the π-calculus semantics.

There is an instantiation $\mathcal{V}$ into s with

$$\begin{aligned} \mathcal{V}_{\mathrm{n}}(\mathtt{a}) &= a & \mathcal{V}_{\mathrm{t}}(\mathtt{x}) &= \mathsf{out}(c,d,\mathsf{out}(c,a,\mathsf{nil})) \\ \mathcal{V}_{\mathrm{n}}(\mathtt{c}) &= c & \mathcal{V}_{\mathrm{t}}(\mathtt{x}') &= \mathsf{inp}(c,\langle a\rangle(\mathsf{out}(c,a,\mathsf{nil}))) \\ \mathcal{V}_{\mathrm{n}}(\mathtt{d}) &= d & \mathcal{V}_{\mathrm{t}}(\mathtt{y}) &= \mathsf{out}(c,a,\mathsf{nil}) \\ & & \mathcal{V}_{\mathrm{t}}(\mathtt{y}') &= \mathsf{out}(c,a,\mathsf{nil}) \quad . \end{aligned}$$

For this instantiation, the archetypal behaviour is

$$\mathcal{V}(\underline{\mathtt{l}}, \mathtt{tar}) \;=\; \mathsf{inj}_{\mathsf{tau}}\big(\mathsf{par}(\mathsf{out}(c,a,\mathsf{nil}), \mathsf{out}(c,d,\mathsf{nil}))\big) \quad .$$

When working with this instantiation, it may be important to assume that the binders are not given values inside some finite set C of names. For example, this instantiation is a little strange because $\mathtt{a}$ is a binder in the rule while a is free in s. Lemma 8.4.7 states that it is appropriate to make assumptions about the freshness of binders. For the present example, one can pick a name $b \in \mathcal{N}$ which is fresh for s and then consider the valuation, $\mathcal{V}'$, given as follows.

$$\begin{aligned} \mathcal{V}'_{\mathrm{n}}(\mathtt{a}) &= b & \mathcal{V}'_{\mathrm{t}}(\mathtt{x}) &= \mathsf{out}(c,d,\mathsf{out}(c,a,\mathsf{nil})) \\ \mathcal{V}'_{\mathrm{n}}(\mathtt{c}) &= c & \mathcal{V}'_{\mathrm{t}}(\mathtt{x}') &= \mathsf{inp}(c,\langle a\rangle(\mathsf{out}(c,a,\mathsf{nil}))) \\ \mathcal{V}'_{\mathrm{n}}(\mathtt{d}) &= d & \mathcal{V}'_{\mathrm{t}}(\mathtt{y}) &= \mathsf{out}(c,a,\mathsf{nil}) \\ & & \mathcal{V}'_{\mathrm{t}}(\mathtt{y}') &= \mathsf{out}(c,b,\mathsf{nil}) \end{aligned}$$

Notice that $\mathcal{V}'$ is also an instantiation into s. A simple calculation reveals that

$$\mathcal{V}'(\underline{\mathtt{l}}, \mathtt{tar}) \;=\; \mathsf{inj}_{\mathsf{tau}}\big(\mathsf{par}(\mathsf{out}(c,a,\mathsf{nil}), \mathsf{out}(c,d,\mathsf{nil}))\big)$$

so that $\mathcal{V}$ and $\mathcal{V}'$ give rise to the same archetypal result.

8.4.3 Inducing an abstract rule

We introduced in equation 8.4.4 a family of functions

$$\left\{ [\![\mathtt{R}]\!]_X : \Sigma_{\mathbb{S},\mathbf{Nom}}(|X| \times B_g|X|) \to B_g T_{\mathbb{S},\mathbf{Nom}}|X| \right\}_{X \in \mathbf{NomSub}} \quad .$$

We now show that each function $[\![\mathtt{R}]\!]_X$ is equivariant, and subsequently that the family $\{ [\![\mathtt{R}]\!]_X \}_X$ is natural.

Proposition 8.4.8. *Each function $[\![\mathtt{R}]\!]_X$ is equivariant.*

Proof. Consider some $\sigma \in \mathsf{Sym}(\mathcal{N})$ and $s \in \Sigma_{\mathbb{S},\mathbf{Nom}}(|X| \times B_g|X|)$. We must show that

$$\begin{aligned} &\left\{ b \;\middle|\; \exists \mathcal{V}.\ \mathcal{V} \text{ is an instantiation into } \sigma \bullet s \text{ and } b = \mathcal{V}(\underline{1}, \mathtt{tar}) \right\} \\ &\qquad = \left\{ \sigma \bullet b \;\middle|\; \exists \mathcal{V}.\ \mathcal{V} \text{ is an instantiation into } s \text{ and } b = \mathcal{V}(\underline{1}, \mathtt{tar}) \right\} \quad . \end{aligned} \tag{8.4.9}$$

To see that RHS $\subseteq$ LHS in equation 8.4.9, we write $(\sigma\mathcal{V})$ for the valuation into X given by $(\sigma \circ \mathcal{V}_{\mathrm{n}}, (\sigma \bullet_X -) \circ \mathcal{V}_{\mathrm{t}})$, and observe that

(i) $\mathcal{V}$ is an instantiation into s if and only if $(\sigma\mathcal{V})$ is an instantiation into $\sigma \bullet s$, and

(ii) $\sigma \bullet \mathcal{V}(\underline{1}, \mathtt{tar}) = (\sigma\mathcal{V})(\underline{1}, \mathtt{tar})$.

To derive that LHS $\subseteq$ RHS in equation 8.4.9, replace $\mathcal{V}$ with $\sigma^{-1}\mathcal{V}$ in (i) and (ii) above. □

Theorem 8.4.10. *The family $\{ [\![\mathtt{R}]\!]_X \}_{X \in \mathbf{NomSub}}$ is natural.*

Proof. Proof of this theorem bears a strong resemblance to the proof of Theorem 6.3.4 for the Positive GSOS format. We must show that for any homomorphism $f : X \to Y$ in **NomSub** we have that

$$B_g T_{\mathbb{S},\mathbf{Nom}}|f|\left([\![\mathtt{R}]\!]_X(s) \right) = [\![\mathtt{R}]\!]_Y(\Sigma_{\mathbb{S},\mathbf{Nom}}(|f| \times B_g|f|)(s)) \quad . \tag{8.4.11a}$$

That is, we must show that

$$\begin{aligned} &\left\{ L_g T_{\mathbb{S},\mathbf{Nom}}|f|(b) \;\middle|\; \begin{array}{l} \exists \mathcal{V}.\ \mathcal{V} \text{ is an instantiation into } s \\ \qquad \text{and } b = \mathcal{V}(\underline{1}, \mathtt{tar}) \end{array} \right\} \\ &\qquad = \left\{ b \;\middle|\; \begin{array}{l} \exists \mathcal{V}.\ \mathcal{V} \text{ is an instantiation into } \Sigma_{\mathbb{S},\mathbf{Nom}}(|f| \times B_g|f|)(s) \\ \qquad \text{and } b = \mathcal{V}(\underline{1}, \mathtt{tar}) \end{array} \right\} . \end{aligned} \tag{8.4.11b}$$

To see that LHS $\subseteq$ RHS we consider an instantiation $\mathcal{V}$ into s and find an instantiation $\mathcal{V}'$ into $\Sigma_{\mathbb{S},\mathbf{Nom}}(|f| \times B_g|f|)(s)$ such that $L_g T_{\mathbb{S},\mathbf{Nom}}|f|(\mathcal{V}(\underline{1}, \mathtt{tar})) = \mathcal{V}'(\underline{1}, \mathtt{tar})$.

We write $(f\mathcal{V})$ for the valuation $(\mathcal{V}_{\mathrm{n}}, f \circ \mathcal{V}_{\mathrm{t}})$ into Y, and we now explain that $\mathcal{V}' = f\mathcal{V}$ is an appropriate valuation. It follows immediately from the definition of $\mathcal{V}(\underline{1}, \mathtt{tar})$ that

$$L_g T_{\mathbb{S},\mathbf{Nom}}|f|(\mathcal{V}(\underline{1}, \mathtt{tar})) = (f\mathcal{V})(\underline{1}, \mathtt{tar}) \quad .$$

It remains for us to show that this choice of $\mathcal{V}'$ is an instantiation into s.

Since $\mathcal{V}$ is an instantiation into s, we have, for each $j \in [1, \mathrm{art}(\underline{\mathtt{op}})]$, a $\beta_j \in B_g|X|$ such that

$$s = \underline{\mathtt{op}}\left((\mathcal{V}_{\mathrm{n}}(\underline{\mathtt{c}}_i))_{i \in [1, \mathrm{arn}(\underline{\mathtt{op}})]}, \left(\left\langle \mathcal{V}_{\mathrm{n}}(\underline{\mathtt{a}}_k^j) \right\rangle_{k \in [1, \mathrm{dep}_j(\underline{\mathtt{op}})]} (\mathcal{V}_{\mathrm{t}}(\underline{\mathtt{x}}_j), \beta_j) \right)_{j \in [1, \mathrm{art}(\underline{\mathtt{op}})]} \right) \tag{8.4.12a}$$

and such that for all $j \in [1, \mathrm{art}(\underline{\mathtt{op}})]$,

$$\mathcal{V}(\mathrm{Prem}_j) \subseteq \beta_j \quad . \tag{8.4.12b}$$

It follows from equation 8.4.12a, and from the action of the functor $\Sigma_{\mathbb{S},\mathbf{Nom}}(|-|\times B_g|-|)$, that

$$\Sigma_{\mathbb{S},\mathbf{Nom}}(|f|\times B_g|f|)(s) = \underline{\mathtt{op}}\left(\begin{array}{l}(\mathcal{V}_{\mathrm{n}}(\underline{\mathtt{c}}_i))_{i\in[1,\mathrm{arn}(\underline{\mathtt{op}})]},\\ \left(\left\langle \mathcal{V}_{\mathrm{n}}(\underline{\mathtt{a}}_k^j)\right\rangle_{k\in[1,\mathrm{dep}_j(\underline{\mathtt{op}})]}(f(\mathcal{V}_{\mathrm{t}}(\underline{\mathtt{x}}_j)),B_g|f|(\beta_j))\right)_{j\in[1,\mathrm{art}(\underline{\mathtt{op}})]}\end{array}\right)$$
$$= \underline{\mathtt{op}}\left(\begin{array}{l}((f\mathcal{V}_{\mathrm{n}})(\underline{\mathtt{c}}_i))_{i\in[1,\mathrm{arn}(\underline{\mathtt{op}})]},\\ \left(\left\langle (f\mathcal{V}_{\mathrm{n}})(\underline{\mathtt{a}}_k^j)\right\rangle_{k\in[1,\mathrm{dep}_j(\underline{\mathtt{op}})]}((f\mathcal{V}_{\mathrm{t}})(\underline{\mathtt{x}}_j),B_g|f|(\beta_j))\right)_{j\in[1,\mathrm{art}(\underline{\mathtt{op}})]}\end{array}\right).$$

From (8.4.12b), together with the action of the functor $B_g|-|$, we have

$$(f\mathcal{V})(\mathrm{Prem}_j)\subseteq B_g|f|(\beta_j)$$

for all $j\in[1,\mathrm{art}(\underline{\mathtt{op}})]$. We also know that $\mathcal{V}_{\mathrm{n}}(\mathsf{bn}(\underline{\mathtt{1}}))\#s$, since $\mathcal{V}$ is an instantiation into s. It follows from Prop. 7.1.2(4) that $\mathcal{V}_{\mathrm{n}}(\mathsf{bn}(\underline{\mathtt{1}}))\#\Sigma_{\mathbb{S},\mathbf{Nom}}(|f|\times B_g|f|)(s)$; that is,

$$(f\mathcal{V}_{\mathrm{n}})(\mathsf{bn}(\underline{\mathtt{1}}))\#\Sigma_{\mathbb{S},\mathbf{Nom}}(|f|\times B_g|f|)(s)\quad.$$

Thus $(f\mathcal{V}')$ is an instantiation into s.

We now turn to prove that RHS $\subseteq$ LHS in equation 8.4.11b. We consider an instantiation $\mathcal{V}$ into $\Sigma_{\mathbb{S},\mathbf{Nom}}(|f|\times B_g|f|)(s)$ and exhibit an instantiation $\mathcal{V}'$ into s which is such that $\mathcal{V}(\underline{\mathtt{1}},\mathtt{tar}) = L_gT_{\mathbb{S},\mathbf{Nom}}|f|(\mathcal{V}'(\underline{\mathtt{1}},\mathtt{tar}))$. Lemma 8.4.7 allows us to assume that $\mathcal{V}$ provides fresh binders for s (and here we mean s, not $\Sigma_{\mathbb{S},\mathbf{Nom}}(|f|\times B_g|f|)(s)$).

We will find an instantiation $\mathcal{V}'$ into s such that $\mathcal{V} = f\mathcal{V}'$. (Here, as above, we write $(f\mathcal{V}')$ for the valuation $(\mathcal{V}'_{\mathrm{n}}, f\circ\mathcal{V}'_{\mathrm{t}})$ into Y.) As observed previously, given such a $\mathcal{V}'$, it follows from the definitions that $(f\mathcal{V}')(\underline{\mathtt{1}},\mathtt{tar}) = L_gT_{\mathbb{S},\mathbf{Nom}}|f|(\mathcal{V}'(\underline{\mathtt{1}},\mathtt{tar}))$, and so equation 8.4.11b will also follow.

In general, there is no canonical instantiation $\mathcal{V}'$ into s with $\mathcal{V} = f\mathcal{V}'$; we make the choice as follows.

Since $\mathcal{V}$ is an instantiation into $\Sigma_{\mathbb{S},\mathbf{Nom}}(|f|\times B_g|f|)(s)$, we have $\beta'_j\in B_g|Y|$, for each $j\in[1,\mathrm{art}(\underline{\mathtt{op}})]$, such that

$$\Sigma_{\mathbb{S},\mathbf{Nom}}(|f|\times B_g|f|)(s) = \underline{\mathtt{op}}\left((\mathcal{V}_{\mathrm{n}}(\underline{\mathtt{c}}_i))_{i\in[1,\mathrm{arn}(\underline{\mathtt{op}})]},\left(\left\langle \mathcal{V}_{\mathrm{n}}(\underline{\mathtt{a}}_k^j)\right\rangle_{k\in[1,\mathrm{dep}_j(\underline{\mathtt{op}})]}(\mathcal{V}_{\mathrm{t}}(\underline{\mathtt{x}}_j),\beta'_j)\right)_{j\in[1,\mathrm{art}(\underline{\mathtt{op}})]}\right)$$

and such that $\mathcal{V}(\mathrm{Prem}_j)\subseteq\beta'_j$ for all $j\in[1,\mathrm{art}(\underline{\mathtt{op}})]$.

Moreover, since $\mathcal{V}$ provides fresh binders for s, we have, by Prop. 8.4.6, that there are, for each $i\in[1,\mathrm{arn}(\underline{\mathtt{op}})]$, names $c_i\in C$, and for each $j\in[1,\mathrm{art}(\underline{\mathtt{op}})]$, elements $x_j\in X$ and $\beta_j\in B_g|X|$ such that

$$s = \underline{\mathtt{op}}\left((c_i)_{i\in[1,\mathrm{arn}(\underline{\mathtt{op}})]},\left(\left\langle \mathcal{V}_{\mathrm{n}}(\underline{\mathtt{a}}_k^j)\right\rangle_{k\in[1,\mathrm{dep}_j(\underline{\mathtt{op}})]}(x_j,\beta_j)\right)_{j\in[1,\mathrm{art}(\underline{\mathtt{op}})]}\right)\quad.$$

For each $i\in[1,\mathrm{arn}(\underline{\mathtt{op}})]$ it is immediate that $c_i = \mathcal{V}_{\mathrm{n}}(\underline{\mathtt{c}}_i)$. Also, for each $j\in[1,\mathrm{art}(\underline{\mathtt{op}})]$ we have, by Prop. 7.1.3(1),

$$f(x_j) = \mathcal{V}_{\mathrm{t}}(\underline{\mathtt{x}}_j)\quad\text{and}\quad B_g|f|(\beta_j) = \beta'_j\quad. \tag{8.4.13}$$

For each $j\in[1,\mathrm{art}(\underline{\mathtt{op}})]$ and $\mathtt{l}\in Lab_g(\mathtt{N})$, we define the sets $\mathtt{X}_{j,\mathtt{l}}\subseteq\mathtt{X}$, $X_{j,\mathtt{l}}\subseteq X$ as follows.

$$\mathtt{X}_{j,\mathtt{l}} = \left\{\mathtt{x}\in\mathtt{X}\;\middle|\;(\underline{\mathtt{x}}_j,\mathtt{l},\mathtt{x})\in\mathrm{Prem}\right\}$$
$$X_{j,\mathtt{l}} = \left\{x\in X\;\middle|\;\mathcal{V}(\mathtt{l})(x)\in\beta_j\right\}$$

Note that for each $j \in [1, \mathrm{art}(\underline{\mathtt{op}})]$ and $\mathtt{l} \in Lab_g(\mathtt{N})$ we have

$$\mathscr{V}_t(\mathtt{X}_{j,\mathtt{l}}) \subseteq f(X_{j,\mathtt{l}}) \quad (\subseteq Y) \quad . \tag{8.4.14}$$

This follows from (8.4.13): suppose, for instance, that $(\underline{\mathtt{x}}_j, \mathtt{l}, \mathtt{x}) \in \mathrm{Prem}$, in which case we must have $\mathscr{V}(\mathtt{l})(\mathscr{V}_t(\mathtt{x})) \in \beta'_j$. There must be $b \in \beta_j$ such that $\mathscr{V}(\mathtt{l})(\mathscr{V}_t(\mathtt{x})) = L_g|f|(b)$. We focus on the case where $\mathtt{l} = \mathtt{c?a}$; then we have $x \in X$, and $a \in \mathscr{N}$ such that

$$b = \mathsf{inj}_{\mathsf{binp}}\left(\mathscr{V}_n(\mathtt{c}), \langle a \rangle\, x\right) \quad .$$

Since $\mathscr{V}$ provides fresh binders for s, we know that $\mathscr{V}_n(\mathtt{a}) \# b$; hence we can assume that $a = \mathscr{V}_n(\mathtt{a})$, and so $\mathscr{V}_t(\mathtt{x}) \in f(X_{j,\mathtt{l}})$.

The next step is to define sets $Y'_{j,\mathtt{l}} \subseteq Y$ and $X'_{j,\mathtt{l}} \subseteq X_{j,\mathtt{l}}$ as follows.

$$\begin{aligned} Y'_{j,\mathtt{l}} &= \mathscr{V}_t(\mathtt{X}_{j,\mathtt{l}}) \\ X'_{j,\mathtt{l}} &= (f|_{X_{j,\mathtt{l}}})^{-1}(Y'_{j,\mathtt{l}}) \end{aligned}$$

Thus $Y'_{j,\mathtt{l}}$ is the image of the function $f|_{X_{j,\mathtt{l}}} : X_{j,\mathtt{l}} \to Y$ on the set $X'_{j,\mathtt{l}} \subseteq X_{j,\mathtt{l}}$. For each appropriate j and $\mathtt{l}$ we choose a section $m_{j,\mathtt{l}} : Y'_{j,\mathtt{l}} \rightarrowtail X'_{j,\mathtt{l}}$ of the quotient $f|_{X'_{j,\mathtt{l}}} : X'_{j,\mathtt{l}} \twoheadrightarrow Y'_{j,\mathtt{l}}$. So $f|_{X'_{j,\mathtt{l}}} \circ m_{j,\mathtt{l}} = \mathrm{id}_{Y'_{j,\mathtt{l}}}$.

We are now in a position to define a valuation $\mathscr{V}'$ into X. We let $\mathscr{V}'_n = \mathscr{V}_n : \mathtt{N} \rightarrowtail \mathscr{N}$. To define $\mathscr{V}'_t$ we use the bijection introduced in Prop. 8.2.1.

- For $j \in [1, \mathrm{art}(\underline{\mathtt{op}})]$, we let $\mathscr{V}'_t(\underline{\mathtt{x}}_j) = x_j$.
- For each premise $(\mathtt{x}, \mathtt{l}, \mathtt{y}) \in \mathrm{Prem}$ and each $j \in [1, \mathrm{art}(\underline{\mathtt{op}})]$ such that $\mathtt{x} = \underline{\mathtt{x}}_j$, we proceed as follows. Note that $\mathscr{V}_t(\mathtt{y}) \in \mathscr{V}_t(\mathtt{X}_{j,\mathtt{l}})$, and so we know that $\mathscr{V}_t(\mathtt{y})$ lies in the domain of $m_{j,\mathtt{l}}$. We let
$$\mathscr{V}'_t(\mathtt{y}) = m_{j,\mathtt{l}}(\mathscr{V}_t(\mathtt{y})) \quad .$$

By construction, we have

$$s = \underline{\mathtt{op}}\left((\mathscr{V}'_n(\underline{\mathtt{c}}_i))_{i\in[1,\mathrm{arn}(\underline{\mathtt{op}})]}, \left(\left\langle \mathscr{V}'_n(\underline{\mathtt{a}}^j_k)\right\rangle_{k\in[1,\mathrm{dep}_j(\underline{\mathtt{op}})]} (\mathscr{V}'_t(\underline{\mathtt{x}}_j), \beta'_j)\right)_{j\in[1,\mathrm{art}(\underline{\mathtt{op}})]}\right)$$

and $\mathscr{V}'(\mathrm{Prem}_j) \subseteq \beta'_j$ for all $j \in [1, \mathrm{art}(\underline{\mathtt{op}})]$. So, since $\mathscr{V}'$ provides fresh binders for s, we know that $\mathscr{V}'$ is an instantiation into s.

We have $\mathscr{V} = f\mathscr{V}'$ for the following reason. For $\mathtt{x}$ appearing in the source of the conclusion, $\mathscr{V}_t(\mathtt{x}) = f(\mathscr{V}'_t(\mathtt{x}))$ follows from (8.4.13). For $\mathtt{y}$ appearing as the target of a premise $(\underline{\mathtt{x}}_j, \mathtt{l}, \mathtt{y})$, we have $f(\mathscr{V}'_t(\mathtt{y})) = f(m_{j,\mathtt{l}}(\mathscr{V}_t(\mathtt{y}))) = \mathscr{V}_t(\mathtt{y})$.

Thus $[\![\mathtt{R}]\!]$ is natural. □

8.4.4 Revisiting the intended model

We conclude this section by relating this development back to the intended model of the nominal logic theory associated to a collection of rules, as considered in 8.1.2. The development of this section so far has explained the derivation of an abstract rule $[\![\mathtt{R}]\!]$ from a solitary rule structure, $\mathtt{R}$, and so our first task is to explain how a set of rule structures gives rise to an abstract rule.

We conclude by briefly relating the induced monad lifting with the intended model, concluding that the intended model of a set of rules in the ν-GSOS$^+$ format will be well-behaved, in the precise sense proposed in Section 8.2.1.

Instantiating a collection of rules. A class $\mathscr{R}$ of rule structures in the $\mathcal{U}$-GSOS$^+$ format gives rise to a natural transformation

$$[\![\mathscr{R}]\!] : \Sigma_{\mathbb{S},\mathbf{Nom}}(|-| \times B_g|-|) \to B_g T_{\mathbb{S},\mathbf{Nom}}|-|$$

given by, for each $X \in \mathbf{NomSub}$ and every $s \in \Sigma_{\mathbb{S},\mathbf{Nom}}(|X| \times B_g|X|)$,

$$[\![\mathscr{R}]\!]_X(s) = \bigcup_{\mathtt{R}\in\mathscr{R}} [\![\mathtt{R}]\!]_X(s) \quad .$$

The only cause for concern in this definition is that the set

$$\bigcup_{\mathtt{R}\in\mathscr{R}} [\![\mathtt{R}]\!]_X(s) \subseteq L_g T_{\mathbb{S},\mathbf{Nom}}|X|$$

ought to be support bounded. Indeed, it follows from Prop. 8.4.3 that each set $[\![\mathtt{R}]\!]_X(s)$ is supported by $\mathsf{supp}(s)$.

The intended model is well-behaved. Let $\mathscr{R}$ be a class of rule structures in the $\mathcal{U}$-GSOS$^+$ format. Using the techniques of Section 6.2.3, the induced natural transformation

$$[\![\mathscr{R}]\!] : \Sigma_{\mathbb{S},\mathbf{Nom}}(|-| \times B_g|-|) \to B_g T_{\mathbb{S},\mathbf{Nom}}|-|$$

lifts the monad $T_{\mathbb{S},\mathbf{Nom}}$ on **Nom** to a monad $T_{(\mathbb{S},\mathbf{Nom})[\![\mathscr{R}]\!]}$ on the category of $|-|$-structured B_g-coalgebras. By adapting the discussion of Section 6.3.5 to this context, we arrive at the following result.

Theorem 8.4.15. *For a class $\mathscr{R}$ of rules in the $\mathcal{U}$-GSOS$^+$ format, the initial $\mathbf{T}_{(\mathbb{S},\mathbf{Nom})[\![\mathscr{R}]\!]}$-algebra is the intended model of the nominal logic theory arising from the class $\mathscr{R}$.* □

The carrier of this initial $\mathbf{T}_{(\mathbb{S},\mathbf{Nom})[\![\mathscr{R}]\!]}$-algebra is the nominal substitution $T_{\mathbb{S},\mathbf{NomSub}}\emptyset$ of $\mathbb{S}$-terms, and we have a B_g-coalgebra structure

$$|T_{\mathbb{S},\mathbf{NomSub}}\emptyset| \to B_g|T_{\mathbb{S},\mathbf{NomSub}}\emptyset|$$

which corresponds to the transition relation of the intended model.

Moreover, the nature of the model theory that we have used, together with Corollary 6.2.3, gives rise to the following result.

Theorem 8.4.16. *The intended model of the nominal logic theory arising from a class $\mathscr{R}$ of rule structures in the $\mathcal{U}$-GSOS$^+$ format has the following properties:*

1. *Axioms $\mathcal{U}_g1$ and $\mathcal{U}_g2$ (of Figure 7.2) are satisfied.*
2. *Wide open bisimilarity is a congruence.* □

8.A Appendix to Chapter 8: Proof of Prop. 8.4.3

We now proceed to prove Proposition 8.4.3.

Proposition 8.4.3. *Consider a nominal substitution X, and let s be an element of $\Sigma_{\mathbb{S},\mathbf{Nom}}(|X| \times B_g|X|)$. If $\mathscr{V}$ is an instantiation into s, and C supports s in $\Sigma_{\mathbb{S},\mathbf{Nom}}(|X| \times B_g|X|)$, then C also supports $\mathscr{V}(\underline{\mathtt{l}}, \mathtt{tar})$ in $L_g T_{\mathbb{S},\mathbf{Nom}}|X|$.*

Proof. We will show that for every raw term $\mathtt{t} \in T_{\mathbb{S}+\mathsf{sub},\mathbf{Set}_{\mathbb{N}}}\mathtt{X}$ we have

$$\mathsf{supp}(\mathscr{V}(\mathtt{t})) \subseteq \mathsf{supp}(s) \cup \mathscr{V}_{\mathrm{n}}(\mathtt{FN}(\mathtt{t})) \quad . \tag{8.A.1}$$

We establish this property by induction on the structure of $\mathtt{t}$. We begin with the base case, when $\mathtt{t} = \mathtt{x}$, for some $\mathtt{x} \in \mathtt{X}$. In this case, (8.A.1) holds since $\mathscr{V}$ is an instantiation into s, and by the properties of supports for sums, products, support-bounded powersets, and abstractions. For instance, suppose that $\mathtt{x}$ appears as the target of a premise; indeed suppose that this premise is $(\underline{\mathtt{x}}_j, \mathtt{c?(a)}, \mathtt{x})$. Since $\mathscr{V}$ is an instantiation into s, we know that

$$s = \underline{\mathtt{op}} \left((\mathscr{V}_{\mathrm{n}}(\underline{\mathtt{c}}_i))_{i \in [1,\mathrm{arn}(\underline{\mathtt{op}})]}, \left(\left\langle \mathscr{V}_{\mathrm{n}}(\underline{\mathtt{a}}_k^j) \right\rangle_{k \in [1,\mathrm{dep}_j(\underline{\mathtt{op}})]} (\mathscr{V}_{\mathrm{t}}(\underline{\mathtt{x}}_j), \beta_j) \right)_{j \in [1,\mathrm{art}(\underline{\mathtt{op}})]} \right)$$

and also that

$$\mathsf{inj}_{\mathsf{binp}}(\mathscr{V}_{\mathrm{n}}(\mathtt{c}), \langle \mathscr{V}_{\mathrm{n}}(\mathtt{a}) \rangle \, \mathscr{V}_{\mathrm{t}}(\mathtt{x})) \in \beta_j \qquad \text{for all } j \in [1, \mathrm{art}(\underline{\mathtt{op}})].$$

By using properties of supports for products, we know that the set $\mathsf{supp}(s)$ supports

$$\left\langle \mathscr{V}_{\mathrm{n}}(\underline{\mathtt{a}}_k^j) \right\rangle_{k \in \mathrm{dep}_j(\underline{\mathtt{op}})} (\mathscr{V}_{\mathrm{t}}(\underline{\mathtt{x}}_j), \beta_j) \quad .$$

Properties of supports for products and abstractions allow us to deduce that the set

$$\mathsf{supp}(s) - \left\{ \mathscr{V}_{\mathrm{n}}(\underline{\mathtt{a}}_k^j) \;\middle|\; k \in [1, \mathrm{dep}_j(\underline{\mathtt{op}})] \right\} \tag{8.A.2}$$

supports β_j in $B_g|X|$. Since β_j is a support bounded subset of $L_g|X|$, this means that the set of (8.A.2) also supports every element of β_j, including, in particular, the element $\mathsf{inj}_{\mathsf{binp}}(\mathscr{V}_{\mathrm{n}}(\mathtt{c}), \langle \mathscr{V}_{\mathrm{n}}(\mathtt{a}) \rangle \, \mathscr{V}_{\mathrm{t}}(\mathtt{x}))$. Using the properties of supports for sums, and for products and abstractions again, we conclude that the set

$$\mathsf{supp}(s) - \left\{ \mathscr{V}_{\mathrm{n}}(\underline{\mathtt{a}}_k^j) \;\middle|\; k \in [1, \mathrm{dep}_j(\underline{\mathtt{op}})] \right\} - \left\{ \mathscr{V}_{\mathrm{n}}(\mathtt{a}) \right\}$$

supports $\mathscr{V}_{\mathrm{t}}(\mathtt{x})$. By definition, the set $\mathtt{FN(x)}$ contains $\underline{\mathtt{a}}_k^j$ for every $k \in [1, \mathrm{dep}_j(\underline{\mathtt{op}})]$, and it also contains $\mathtt{a}$. Thus the base case for (8.A.1) is established.

For the inductive step, we consider an operator $\mathtt{op} \in \mathsf{Op}_{\mathbb{S}+\mathsf{sub}}$, together with: name variables $\mathtt{c}_i \in \mathtt{N}$, for each $i \in [1, \mathrm{arn}(\mathtt{op})]$; name variables $\mathtt{a}_k^j \in \mathtt{N}$, for each $j \in [1, \mathrm{art}(\mathtt{op})]$ and $k \in [1, \mathrm{dep}_i(\mathtt{op})]$; and terms $\mathtt{t}_j \in T_{\mathbb{S}+\mathsf{sub},\mathbf{Set}_{\mathbb{N}}}(\mathtt{X})$ for each $j \in [1, \mathrm{art}(\mathtt{op})]$. We suppose that

$$\mathtt{t} = \mathtt{op} \left((\mathtt{c}_i)_{i \in [1,\mathrm{arn}(\mathtt{op})]}, \left(\left\langle \mathtt{a}_k^j \right\rangle_{k \in [1,\mathrm{dep}_j(\mathtt{op})]} \mathtt{t}_j \right)_{j \in [1,\mathrm{art}(\mathtt{op})]} \right) \quad .$$

Our induction hypotheses are that property (8.A.1) holds of $\mathtt{t}_j$, for each $j \in [1, \mathrm{art}(\mathtt{op})]$; we must now show that (8.A.1) holds of $\mathtt{t}$.

The induction hypotheses ensure that for each $j \in [1, \mathrm{art}(\mathtt{op})]$ we have

$$\mathsf{supp}(\mathscr{V}(\mathtt{t}_j)) \subseteq \mathsf{supp}(s) \cup \mathscr{V}_{\mathrm{n}}(\mathtt{FN}(\mathtt{t}_j)) \quad .$$

So, from the nature of supports of abstractions, we have

$$\mathsf{supp}(\left\langle \mathscr{V}_{\mathrm{n}}(\mathtt{a}_k^j)\right\rangle_{k\in[1,\mathrm{dep}_j(\mathsf{op})]} \mathscr{V}\left(\mathtt{t}_j\right)) \subseteq \left(\mathsf{supp}(s)\cup\mathscr{V}_{\mathrm{n}}(\mathtt{FN}(\mathtt{t}_j))\right) - \left\{\mathscr{V}_{\mathrm{n}}(\mathtt{a}_k^j) \;\middle|\; k\in[1,\mathrm{dep}_j(\mathsf{op})]\right\} \quad .$$

So certainly

$$\mathsf{supp}(\left\langle \mathscr{V}_{\mathrm{n}}(\mathtt{a}_k^j)\right\rangle_{k\in[1,\mathrm{dep}_j(\mathsf{op})]} \mathscr{V}\left(\mathtt{t}_j\right)) \subseteq \mathsf{supp}(s)\cup\mathscr{V}_{\mathrm{n}}\left(\mathtt{FN}(\mathtt{t}_j) - \left\{\mathtt{a}_k^j \;\middle|\; k\in[1,\mathrm{dep}_j(\mathsf{op})]\right\}\right) \quad .$$

By properties of supports of products and abstractions,

$$\mathsf{supp}(\mathscr{V}(\mathtt{t})) \subseteq \left\{\mathscr{V}_{\mathrm{n}}(\mathtt{c}_i) \;\middle|\; i\in[1,\mathrm{arn}(\mathsf{op})]\right\} \cup \bigcup\left\{\mathsf{supp}(\left\langle \mathscr{V}_{\mathrm{n}}(\mathtt{a}_k^j)\right\rangle_{k\in[1,\mathrm{dep}_j(\mathsf{op})]} \mathscr{V}\left(\mathtt{t}_j\right)) \;\middle|\; j\in[1,\mathrm{art}(\mathsf{op})]\right\} \quad .$$

Moreover, from the definition of FN we have

$$\mathtt{FN}(\mathtt{t}_j) - \left\{\mathtt{a}_k^j \;\middle|\; k\in[1,\mathrm{dep}_j(\mathsf{op})]\right\} \subseteq \mathtt{FN}(\mathtt{t})$$

and by definition of $\mathtt{FN}(\mathtt{t})$ we have $\left\{\mathscr{V}_{\mathrm{n}}(\mathtt{c}_i) \;\middle|\; i\in[1,\mathrm{arn}(\mathsf{op})]\right\} \subseteq \mathscr{V}_{\mathrm{n}}(\mathtt{FN}(\mathtt{t}))$. Putting this all together, we conclude that $\mathsf{supp}(\mathscr{V}(\mathtt{t})) \subseteq \mathsf{supp}(s)\cup\mathscr{V}_{\mathrm{n}}(\mathtt{FN}(\mathtt{t}))$, as required. Thus (8.A.1) is established.

The statement of the proposition follows from (8.A.1), via Condition И-GSOS$^+$-9 and Condition И-GSOS$^+$-12; for the case when $\underline{\mathtt{l}} = \mathtt{c?(a)}$ we proceed as follows. We know that

$$\mathscr{V}(\underline{\mathtt{l}},\mathtt{tar}) = \mathsf{inj}_{\mathsf{binp}}(\mathscr{V}_{\mathrm{n}}(\mathtt{c}), \langle\mathscr{V}_{\mathrm{n}}(\mathtt{a})\rangle\,\mathscr{V}(\mathtt{tar}))$$

so, by basic properties of supports of products and abstractions, we have

$$\mathsf{supp}(\mathscr{V}(\underline{\mathtt{l}},\mathtt{tar})) = \{\mathscr{V}_{\mathrm{n}}(\mathtt{c})\} \cup \left(\mathsf{supp}(\mathscr{V}(\mathtt{tar})) - \{\mathscr{V}_{\mathrm{n}}(\mathtt{a})\}\right) \quad .$$

Using property (8.A.1) we can deduce that

$$\mathsf{supp}(\mathscr{V}(\underline{\mathtt{l}},\mathtt{tar})) \subseteq \{\mathscr{V}_{\mathrm{n}}(\mathtt{c})\} \cup \left(\left(\mathsf{supp}(s)\cup\mathscr{V}_{\mathrm{n}}(\mathtt{FN}(\mathtt{tar}))\right) - \{\mathscr{V}_{\mathrm{n}}(\mathtt{a})\}\right)$$

and so certainly

$$\mathsf{supp}(\mathscr{V}(\underline{\mathtt{l}},\mathtt{tar})) \subseteq \{\mathscr{V}_{\mathrm{n}}(\mathtt{c})\} \cup \mathsf{supp}(s) \cup \left(\mathscr{V}_{\mathrm{n}}(\mathtt{FN}(\mathtt{tar})) - \{\mathscr{V}_{\mathrm{n}}(\mathtt{a})\}\right) \quad .$$

Using Condition И-GSOS$^+$-9, we have

$$\mathsf{supp}(\mathscr{V}(\underline{\mathtt{l}},\mathtt{tar})) \subseteq \mathsf{supp}(s) \cup \left(\mathscr{V}_{\mathrm{n}}(\mathtt{FN}(\mathtt{tar})) - \{\mathscr{V}_{\mathrm{n}}(\mathtt{a})\}\right)$$

since $\mathrm{fn}(\underline{\mathtt{l}}) = \{\mathtt{c}\}$, and using properties of supports of products, abstractions and support-bounded powersets, coupled with the fact that $\mathscr{V}$ is an instantiation into s.

Finally, we appeal to Condition И-GSOS$^+$-12 to conclude that

$$\mathsf{supp}(\mathscr{V}(\underline{\mathtt{l}},\mathtt{tar})) \subseteq \mathsf{supp}(s)$$

since $\mathtt{BN}(\underline{\mathtt{l}}) = \{\mathtt{a}\}$ and, again, since $\mathscr{V}$ is an instantiation into s. Other kinds of conclusion label are treated similarly; thus Proposition 8.4.3 is proved. □

8.B Appendix to Chapter 8: Proof of Lemma 8.4.7

We now establish Lemma 8.4.7.

Lemma 8.4.7. *Consider some $s \in \Sigma_{\mathbb{S},\mathbf{Nom}}(|X| \times B_g|X|)$, and $C \subseteq_f \mathcal{N}$. For any instantiation $\mathcal{V}$ into s, there is another instantiation $\mathcal{V}'$ into s that provides fresh binders for C and which is such that $\mathcal{V}(\underline{\mathtt{l}}, \mathtt{tar}) = \mathcal{V}'(\underline{\mathtt{l}}, \mathtt{tar})$.*

Proof. Let $\mathcal{V}$ be an instantiation into s. So we have, for each $j \in [1, \mathrm{art}(\underline{\mathtt{op}})]$, some $\beta_j \in B_g|X|$ such that

$$s = \underline{\mathtt{op}}\left((\mathcal{V}_{\mathrm{n}}(\underline{\mathtt{c}}_i))_{i\in[1,\mathrm{arn}(\underline{\mathtt{op}})]}, \left(\left\langle \mathcal{V}_{\mathrm{n}}(\underline{\mathtt{a}}_k^j)\right\rangle_{k\in[1,\mathrm{dep}_j(\underline{\mathtt{op}})]}(\mathcal{V}_{\mathrm{t}}(\underline{\mathtt{x}}_j), \beta_j)\right)_{j\in[1,\mathrm{art}(\underline{\mathtt{op}})]}\right)$$

and such that $\mathcal{V}(\mathrm{Prem}_j) \subseteq \beta_j$ for all $j \in [1, \mathrm{art}(\underline{\mathtt{op}})]$.

For convenience, we define $\mathtt{A} = \left\{\mathtt{a} \in (\mathtt{N} - \mathtt{BN}) \;\middle|\; \mathcal{V}_{\mathrm{n}}(\mathtt{a}) \# s\right\}$. We pick an injection

$$\xi : \mathtt{N} \rightarrowtail \left(\mathcal{N} - \left(\mathsf{supp}(s) \cup \mathsf{im}(\mathcal{V}_{\mathrm{n}}) \cup C\right)\right)$$

This is possible as the sets $\mathsf{supp}(s)$, $\mathsf{im}(\mathcal{V}_{\mathrm{n}})$, C are all finite. We define $\mathcal{V}'_{\mathrm{n}} : \mathtt{N} \to \mathcal{N}$ by

$$\mathcal{V}'_{\mathrm{n}}(\mathtt{c}) = \begin{cases} \xi(\mathtt{c}) & \text{if } \mathtt{c} \in \mathtt{BN} \cup \mathtt{A} \\ \mathcal{V}_{\mathrm{n}}(\mathtt{c}) & \text{if } \mathtt{c} \in (\mathtt{N} - (\mathtt{BN} \cup \mathtt{A})). \end{cases} \tag{8.B.1a}$$

We define $\mathcal{V}'_{\mathrm{t}} : \mathtt{X} \to X$ by

$$\mathcal{V}'_{\mathrm{t}}(\mathtt{x}) = [\mathcal{V}_{\mathrm{n}}(\mathtt{a}) \leftrightarrow \mathcal{V}'_{\mathrm{n}}(\mathtt{a})]_{\mathtt{a}\in\mathtt{BN}(\mathtt{x})} \bullet \mathcal{V}_{\mathrm{t}}(\mathtt{x}) \quad . \tag{8.B.1b}$$

(It follows from the definition of $\mathcal{V}'_{\mathrm{n}}$ that the order of the swaps, $[\mathcal{V}_{\mathrm{n}}(\mathtt{a}) \leftrightarrow \mathcal{V}'_{\mathrm{n}}(\mathtt{a})]_{\mathtt{a}\in\mathtt{BN}(\mathtt{x})}$, doesn't matter.) We now explain that this instantiation, $\mathcal{V}'$, is an instantiation into s. It is clear that $\mathcal{V}'_{\mathrm{n}} : \mathtt{N} \to \mathcal{N}$ is injective, and indeed satisfies (8.B.1a).

We explain why $\mathcal{V}'_{\mathrm{n}}(\mathsf{bn}(\underline{\mathtt{l}})) \cap \mathsf{supp}(s) = \emptyset$. Since $\mathcal{V}$ is an instantiation into s, we have that $\mathcal{V}_{\mathrm{n}}(\mathsf{bn}(\underline{\mathtt{l}})) \cap \mathsf{supp}(s) = \emptyset$. So, by (8.B.1a), we know that, for any $\mathtt{a} \in \mathsf{bn}(\underline{\mathtt{l}})$, we have $\mathcal{V}'_{\mathrm{n}}(\mathtt{a}) = \xi(\mathtt{a})$. It follows, by considering the codomain of ξ, that $\mathcal{V}'_{\mathrm{n}}(\mathsf{bn}(\underline{\mathtt{l}})) \cap \mathsf{supp}(s) = \emptyset$.

To conclude that $\mathcal{V}'$ is an instantiation into s, we must find $(\beta'_j \in B_g|X|)_{j\in[1,\mathrm{art}(\underline{\mathtt{op}})]}$ such that

$$s = \underline{\mathtt{op}}\left((\mathcal{V}'_{\mathrm{n}}(\underline{\mathtt{c}}_i))_{i\in[1,\mathrm{arn}(\underline{\mathtt{op}})]}, \left(\left\langle \mathcal{V}'_{\mathrm{n}}(\underline{\mathtt{a}}_k^j)\right\rangle_{k\in\mathrm{dep}_j(\underline{\mathtt{op}})}(\mathcal{V}'_{\mathrm{t}}(\underline{\mathtt{x}}_j), \beta'_j)\right)_{j\in\mathrm{art}(\underline{\mathtt{op}})}\right) \tag{8.B.2a}$$

and such that for all $j \in [1, \mathrm{art}(\underline{\mathtt{op}})]$,

$$\mathcal{V}'(\mathrm{Prem}_j) \subseteq \beta'_j \quad . \tag{8.B.2b}$$

To this end, we let $\beta'_j = [\mathcal{V}_{\mathrm{n}}(\underline{\mathtt{a}}_k^j) \leftrightarrow \mathcal{V}'_{\mathrm{n}}(\underline{\mathtt{a}}_k^j)]_{k\in[1,\mathrm{dep}_j(\underline{\mathtt{op}})]} \bullet \beta_j$. By Conditions И-GSOS$^+$-6 and И-GSOS$^+$-8, we know that, for each $i \in [1, \mathrm{arn}(\underline{\mathtt{op}})]$, $\underline{\mathtt{c}}_i \notin \mathtt{BN}$; moreover, it is clear that for each $i \in [1, \mathrm{arn}(\underline{\mathtt{op}})]$ it is not the case that $\mathcal{V}_{\mathrm{n}}(\underline{\mathtt{c}}_i) \# s$. Thus, by considering the definition of $\mathcal{V}'_{\mathrm{n}}$, we can conclude that, for each $i \in [1, \mathrm{arn}(\underline{\mathtt{op}})]$, $\mathcal{V}_{\mathrm{n}}(\underline{\mathtt{c}}_i) = \mathcal{V}'_{\mathrm{n}}(\underline{\mathtt{c}}_i)$.

To conclude that equation 8.B.2a holds, it remains for us to show that for each $j \in [1, \mathrm{art}(\underline{\mathtt{op}})]$,

$$\left\langle \mathcal{V}_{\mathrm{n}}(\underline{\mathtt{a}}_k^j)\right\rangle_{k\in[1,\mathrm{dep}_j(\underline{\mathtt{op}})]}(\mathcal{V}_{\mathrm{t}}(\underline{\mathtt{x}}_j), \beta_j) = \left\langle \mathcal{V}'_{\mathrm{n}}(\underline{\mathtt{a}}_k^j)\right\rangle_{k\in[1,\mathrm{dep}_j(\underline{\mathtt{op}})]}(\mathcal{V}'_{\mathrm{t}}(\underline{\mathtt{x}}_j), \beta'_j) \quad . \tag{8.B.3}$$

We will show that for each $k' \in [0, \mathrm{dep}_j(\underline{\mathtt{op}})]$ we have

$$\left\langle \mathcal{V}_{\mathrm{n}}(\underline{\mathtt{a}}_k^j) \right\rangle_{k\in[1,\mathrm{dep}_j(\underline{\mathtt{op}})]} (\mathcal{V}_{\mathrm{t}}(\underline{\mathtt{x}}_j), \beta_j) = \left\langle \mathcal{V}_{\mathrm{n}}(\underline{\mathtt{a}}_k^j) \right\rangle_{k\in[k'+1,\mathrm{dep}_j(\underline{\mathtt{op}})]} \left\langle \mathcal{V}'_{\mathrm{n}}(\underline{\mathtt{a}}_k^j) \right\rangle_{k\in[1,k']} [\mathcal{V}_{\mathrm{n}}(\underline{\mathtt{a}}_k^j) \leftrightarrow \mathcal{V}'_{\mathrm{n}}(\underline{\mathtt{a}}_k^j)]_{k\in[1,k']} \bullet (\mathcal{V}_{\mathrm{t}}(\underline{\mathtt{x}}_j), \beta_j) \quad .$$

For the case $k' = 0$, this result is trivial; for the inductive step $k' = k'' + 1$, then we arrive at this equation by the following sequence.

$$\begin{aligned}
&\left\langle \mathcal{V}_{\mathrm{n}}(\underline{\mathtt{a}}_k^j) \right\rangle_{k\in[1,\mathrm{dep}_j(\underline{\mathtt{op}})]} (\mathcal{V}_{\mathrm{t}}(\underline{\mathtt{x}}_j), \beta_j) \\
&\qquad = \left\langle \mathcal{V}_{\mathrm{n}}(\underline{\mathtt{a}}_k^j) \right\rangle_{k\in[k''+1,\mathrm{dep}_j(\underline{\mathtt{op}})]} \left\langle \mathcal{V}'_{\mathrm{n}}(\underline{\mathtt{a}}_k^j) \right\rangle_{k\in[1,k'']} && \text{(8.B.4a)} \\
&\qquad\qquad [\mathcal{V}_{\mathrm{n}}(\underline{\mathtt{a}}_k^j) \leftrightarrow \mathcal{V}'_{\mathrm{n}}(\underline{\mathtt{a}}_k^j)]_{k\in[1,k'']} \bullet (\mathcal{V}_{\mathrm{t}}(\underline{\mathtt{x}}_j), \beta_j) \\
&\qquad = \left\langle \mathcal{V}_{\mathrm{n}}(\underline{\mathtt{a}}_k^j) \right\rangle_{k\in[k'+1,\mathrm{dep}_j(\underline{\mathtt{op}})]} \left\langle \mathcal{V}_{\mathrm{n}}(\underline{\mathtt{a}}_{k'}^j) \right\rangle \left\langle \mathcal{V}'_{\mathrm{n}}(\underline{\mathtt{a}}_k^j) \right\rangle_{k\in[1,k'']} && \text{(8.B.4b)} \\
&\qquad\qquad [\mathcal{V}_{\mathrm{n}}(\underline{\mathtt{a}}_k^j) \leftrightarrow \mathcal{V}'_{\mathrm{n}}(\underline{\mathtt{a}}_k^j)]_{k\in[1,k'']} \bullet (\mathcal{V}_{\mathrm{t}}(\underline{\mathtt{x}}_j), \beta_j) \\
&\qquad = \left\langle \mathcal{V}_{\mathrm{n}}(\underline{\mathtt{a}}_k^j) \right\rangle_{k\in[k'+1,\mathrm{dep}_j(\underline{\mathtt{op}})]} \left\langle \mathcal{V}'_{\mathrm{n}}(\underline{\mathtt{a}}_{k'}^j) \right\rangle && \text{(8.B.4c)} \\
&\qquad\qquad [\mathcal{V}_{\mathrm{n}}(\underline{\mathtt{a}}_{k'}^j) \leftrightarrow \mathcal{V}'_{\mathrm{n}}(\underline{\mathtt{a}}_{k'}^j)] \bullet \left\langle \mathcal{V}'_{\mathrm{n}}(\underline{\mathtt{a}}_k^j) \right\rangle_{k\in[1,k'']} \\
&\qquad\qquad\qquad [\mathcal{V}_{\mathrm{n}}(\underline{\mathtt{a}}_k^j) \leftrightarrow \mathcal{V}'_{\mathrm{n}}(\underline{\mathtt{a}}_k^j)]_{k\in[1,k'']} \bullet (\mathcal{V}_{\mathrm{t}}(\underline{\mathtt{x}}_j), \beta_j) \\
&\qquad = \left\langle \mathcal{V}_{\mathrm{n}}(\underline{\mathtt{a}}_k^j) \right\rangle_{k\in[k'+1,\mathrm{dep}_j(\underline{\mathtt{op}})]} \left\langle \mathcal{V}'_{\mathrm{n}}(\underline{\mathtt{a}}_k^j) \right\rangle_{k\in[1,k']} && \text{(8.B.4d)} \\
&\qquad\qquad [\mathcal{V}_{\mathrm{n}}(\underline{\mathtt{a}}_k^j) \leftrightarrow \mathcal{V}'_{\mathrm{n}}(\underline{\mathtt{a}}_k^j)]_{k\in[1,k']} \bullet (\mathcal{V}_{\mathrm{t}}(\underline{\mathtt{x}}_j), \beta_j)
\end{aligned}$$

Using: (8.B.4a): ind. hyp.; (8.B.4b): since $k' = k'' + 1$; (8.B.4c): since $\mathcal{V}'_{\mathrm{n}}(\underline{\mathtt{a}}_{k'}^j)$ is fresh for the pair $[\mathcal{V}_{\mathrm{n}}(\underline{\mathtt{a}}_k^j) \leftrightarrow \mathcal{V}'_{\mathrm{n}}(\underline{\mathtt{a}}_k^j)]_{k\in[1,k'']} \bullet (\mathcal{V}_{\mathrm{t}}(\underline{\mathtt{x}}_j), \beta_j)$; (8.B.4d): defn. of permutation action on abstractions, using Condition И-GSOS$^+$-7.

Thus (8.B.3) is established, and we can conclude equation 8.B.2a.

To show (8.B.2b), we must show that for each $j \in [1, \mathrm{art}(\underline{\mathtt{op}})]$, and each $(\underline{\mathtt{x}}_j, \mathtt{l}, \mathtt{y}) \in \mathrm{Prem}$ we have $\mathcal{V}'(\mathtt{l})(\mathtt{y}) \in \beta'_j$.

Consider some $(\mathtt{x}, \mathtt{l}, \mathtt{y}) \in \mathrm{Prem}$. Since $\mathcal{V}$ is an instantiation into s, we know that $\mathcal{V}(\mathtt{l})(\mathtt{y}) \in \beta_j$. We will show that $\mathcal{V}'(\mathtt{l})(\mathtt{y}) \in \beta'_j$.

We will focus on the case of input premises; let $\mathtt{l} = \mathtt{c?(a)}$. In this situation we know that $\mathsf{inj}_{\mathsf{binp}}\left(\mathcal{V}_{\mathrm{n}}(\mathtt{c}), \langle \mathcal{V}_{\mathrm{n}}(\mathtt{a}) \rangle\, \mathcal{V}_{\mathrm{t}}(\mathtt{y})\right) \in \beta_j$. Thus, by definition,

$$[\mathcal{V}_{\mathrm{n}}(\underline{\mathtt{a}}_k^j) \leftrightarrow \mathcal{V}'_{\mathrm{n}}(\underline{\mathtt{a}}_k^j)]_{k\in[1,\mathrm{dep}_j(\underline{\mathtt{op}})]} \bullet \mathsf{inj}_{\mathsf{binp}}\left(\mathcal{V}_{\mathrm{n}}(\mathtt{c}), \langle \mathcal{V}_{\mathrm{n}}(\mathtt{a}) \rangle\, \mathcal{V}_{\mathrm{t}}(\mathtt{y})\right) \in \beta'_j \quad .$$

That is, using Condition И-GSOS$^+$-8,

$$\mathsf{inj}_{\mathsf{binp}}\left(\mathcal{V}'_{\mathrm{n}}(\mathtt{c}), \langle \mathcal{V}_{\mathrm{n}}(\mathtt{a}) \rangle \left([\mathcal{V}_{\mathrm{n}}(\underline{\mathtt{a}}_k^j) \leftrightarrow \mathcal{V}'_{\mathrm{n}}(\underline{\mathtt{a}}_k^j)]_{k\in[1,\mathrm{dep}_j(\underline{\mathtt{op}})]} \bullet \mathcal{V}_{\mathrm{t}}(\mathtt{y})\right)\right) \in \beta'_j \quad .$$

By definition of $\mathcal{V}'_{\mathrm{n}}$, we know that $\mathcal{V}'_{\mathrm{n}}(\mathtt{a})$ is fresh for s; so we know that $\mathcal{V}'_{\mathrm{n}}(\mathtt{a})$ is fresh for $\left\langle \mathcal{V}'_{\mathrm{n}}(\underline{\mathtt{a}}_k^j) \right\rangle_{k\in\mathrm{dep}_j(\underline{\mathtt{op}})} (\mathcal{V}'_{\mathrm{t}}(\underline{\mathtt{x}}_j), \beta'_j)$. Since $\mathcal{V}'_{\mathrm{n}}$ is injective, we know from Condition И-GSOS$^+$-8 that $\mathcal{V}'_{\mathrm{n}}(\mathtt{a})$ is distinct from each binder $\mathcal{V}'_{\mathrm{n}}(\underline{\mathtt{a}}_k^j)$. Thus we know that $\mathcal{V}'_{\mathrm{n}}(\mathtt{a})$ is fresh for β'_j, and in particular $\mathcal{V}'_{\mathrm{n}}(\mathtt{a})$ is fresh for

$$\langle \mathcal{V}_{\mathrm{n}}(\mathtt{a}) \rangle \left([\mathcal{V}_{\mathrm{n}}(\underline{\mathtt{a}}_k^j) \leftrightarrow \mathcal{V}'_{\mathrm{n}}(\underline{\mathtt{a}}_k^j)]_{k\in[1,\mathrm{dep}_j(\underline{\mathtt{op}})]} \bullet \mathcal{V}_{\mathrm{t}}(\mathtt{y})\right) \quad .$$

Hence, by definition of abstraction, we have

$$\mathsf{inj}_{\mathsf{binp}}\left(\mathscr{V}'_{\mathrm{n}}(\mathtt{c}),\left\langle\mathscr{V}'_{\mathrm{n}}(\mathtt{a})\right\rangle\left(\begin{array}{l}[\mathscr{V}_{\mathrm{n}}(\mathtt{a})\leftrightarrow\mathscr{V}'_{\mathrm{n}}(\mathtt{a})]\\ \quad\bullet\,[\mathscr{V}_{\mathrm{n}}(\underline{\mathtt{a}}^j_k)\leftrightarrow\mathscr{V}'_{\mathrm{n}}(\underline{\mathtt{a}}^j_k)]_{k\in[1,\mathrm{dep}_j(\underline{\mathtt{op}})]}\\ \quad\bullet\,\mathscr{V}_{\mathrm{t}}(\mathtt{y})\end{array}\right)\right)\in\beta'_j\quad.$$

That is,

$$\mathsf{inj}_{\mathsf{binp}}\left(\mathscr{V}'_{\mathrm{n}}(\mathtt{c}),\left\langle\mathscr{V}'_{\mathrm{n}}(\mathtt{a})\right\rangle\mathscr{V}'_{\mathrm{t}}(\mathtt{y})\right)\in\beta'_j$$

as required. Thus (8.B.2b) is established, and we can conclude that $\mathscr{V}'$ is an instantiation into s.

To prove that $\mathscr{V}(\underline{\mathtt{l}},\mathtt{tar})=\mathscr{V}'(\underline{\mathtt{l}},\mathtt{tar})$, we proceed as follows. We will show that for each $\mathtt{t}\in T_{\mathbb{S}+\mathsf{sub},\mathbf{Set}_{\mathbb{N}}}(\mathtt{X})$ we have properties 8.B.5a and b:

$$\mathscr{V}'_{\mathrm{n}}(\mathtt{BN}\cup\mathtt{A})\cap\mathsf{supp}(\mathscr{V}(\mathtt{t}))=\emptyset \tag{8.B.5a}$$

and

$$\begin{array}{l}\text{For any set }\mathtt{C}\subseteq_{\mathrm{f}}\mathtt{N}:\\ \quad\text{if }\mathrm{WF}(\mathtt{t})\text{ and }(\mathtt{BN}\cup\mathtt{A})\cap\mathrm{FN}(\mathtt{t})\subseteq\mathtt{C}\\ \qquad\text{and for all }\mathtt{x}\text{ appearing in }\mathtt{t}\text{ and all }\mathtt{a}\in\mathtt{C}\\ \qquad\qquad\text{we have either }\mathtt{a}\in\mathrm{FN}(\mathtt{x})\text{ or }\mathscr{V}_{\mathrm{n}}(\mathtt{a})\#\mathscr{V}'_{\mathrm{t}}(\mathtt{x})\\ \quad\text{then }[\mathscr{V}_{\mathrm{n}}(\mathtt{a})\leftrightarrow\mathscr{V}'_{\mathrm{n}}(\mathtt{a})]_{\mathtt{a}\in\mathtt{C}}\bullet\mathscr{V}(\mathtt{t})=\mathscr{V}'(\mathtt{t})\quad.\end{array}\tag{8.B.5b}$$

The order of the swaps in property 8.B.5b is irrelevant since $\mathscr{V}_{\mathrm{n}}$ and $\mathscr{V}'_{\mathrm{n}}$ are both injective and since whenever $\mathscr{V}_{\mathrm{n}}(\mathtt{a})\neq\mathscr{V}'_{\mathrm{n}}(\mathtt{a})$ we have $\mathtt{a}\in(\mathtt{BN}\cup\mathtt{A})$, and hence $\mathscr{V}_{\mathrm{n}}(\mathtt{a})\notin\mathsf{im}(\mathscr{V}'_{\mathrm{n}})$ and $\mathscr{V}'_{\mathrm{n}}(\mathtt{a})\notin\mathsf{im}(\mathscr{V}_{\mathrm{n}})$.

We prove properties 8.B.5a and b by induction on the structure of terms $\mathtt{t}\in T_{\mathbb{S}+\mathsf{sub},\mathbf{Set}_{\mathbb{N}}}(\mathtt{X})$. For convenience, we prove the two statements together.

For the base case $\mathtt{t}=\mathtt{x}$, property 8.B.5a follows by definition of $\mathscr{V}'_{\mathrm{n}}$: we know that $\mathsf{im}(\beta)\cap\big(\mathsf{im}(\mathscr{V}_{\mathrm{n}})\cup\mathsf{supp}(s)\big)=\emptyset$, and also that $\mathsf{supp}(\mathscr{V}_{\mathrm{t}}(\mathtt{x}))\subseteq\big(\mathsf{im}(\mathscr{V}_{\mathrm{n}})\cup\mathsf{supp}(s)\big)$.

We tackle the base case of property 8.B.5b as follows. Suppose that we have $\mathtt{C}\subseteq_{\mathrm{f}}\mathtt{N}$ such that $(\mathtt{BN}\cup\mathtt{A})\cap\mathrm{FN}(\mathtt{x})\subseteq\mathtt{C}$ and that for each $\mathtt{a}\in\mathtt{C}$ we have either that $\mathtt{a}\in\mathrm{FN}(\mathtt{x})$ or $\mathscr{V}_{\mathrm{n}}(\mathtt{a})\#\mathscr{V}'_{\mathrm{t}}(\mathtt{x})$. Let $\mathtt{C}'=(\mathtt{C}-\mathrm{FN}(\mathtt{x}))$; so for each $\mathtt{a}\in\mathtt{C}'$ we have $\mathscr{V}_{\mathrm{n}}(\mathtt{a})\#\mathscr{V}'_{\mathrm{t}}(\mathtt{x})$. It follows from Conditions И-GSOS$^+$-6 and И-GSOS$^+$-8 that $\mathtt{BN}\cap\mathrm{FN}(\mathtt{x})=\mathtt{BN}(\mathtt{x})$; note further that, by definition of $\mathtt{A}$, we have $\mathtt{A}\cap\mathrm{FN}(\mathtt{x})=\emptyset$; so $\mathtt{BN}(\mathtt{x})\subseteq\mathtt{C}$. Hence $\mathtt{C}=\mathtt{C}'\cup\mathtt{BN}(\mathtt{x})\cup(\mathtt{C}\cap\mathrm{FN})$. Thus we must prove that

$$[\mathscr{V}_{\mathrm{n}}(\mathtt{a})\leftrightarrow\mathscr{V}'_{\mathrm{n}}(\mathtt{a})]_{\mathtt{C}'\cup\mathtt{BN}(\mathtt{x})\cup(\mathtt{C}\cap\mathrm{FN})}\bullet\mathscr{V}_{\mathrm{t}}(\mathtt{x})=\mathscr{V}'_{\mathrm{t}}(\mathtt{x})\quad.$$

Property 8.B.1b ensures that $[\mathscr{V}_{\mathrm{n}}(\mathtt{a})\leftrightarrow\mathscr{V}'_{\mathrm{n}}(\mathtt{a})]_{\mathtt{a}\in\mathtt{BN}(\mathtt{x})}\bullet\mathscr{V}_{\mathrm{t}}(\mathtt{t})=\mathscr{V}'_{\mathrm{t}}(\mathtt{t})$. As we have remarked, the order of the swaps doesn't matter, so it remains for us to explain why the remaining swaps, $[\mathscr{V}_{\mathrm{n}}(\mathtt{a})\leftrightarrow\mathscr{V}'_{\mathrm{n}}(\mathtt{a})]_{\mathtt{a}\in\mathtt{C}'\cup(\mathtt{C}\cap\mathrm{FN})}$, fix $\mathscr{V}'_{\mathrm{t}}(\mathtt{t})$. That is, we must show that

$$[\mathscr{V}_{\mathrm{n}}(\mathtt{a})\leftrightarrow\mathscr{V}'_{\mathrm{n}}(\mathtt{a})]_{\mathtt{a}\in\mathtt{C}'\cup(\mathtt{C}\cap\mathrm{FN})}\bullet\mathscr{V}'_{\mathrm{t}}(\mathtt{x})=\mathscr{V}'_{\mathrm{t}}(\mathtt{x})\quad.$$

For each $\mathtt{a}\in(\mathtt{C}\cap\mathrm{FN})$, Conditions И-GSOS$^+$-6 and И-GSOS$^+$-8 ensure that $\mathtt{a}\notin\mathtt{BN}$; furthermore, since $\mathscr{V}$ is an instantiation into s, we cannot have that $\mathscr{V}_{\mathrm{n}}(\mathtt{a})\#s$. Hence, by definition of $\mathscr{V}'_{\mathrm{n}}$, we must have $\mathscr{V}_{\mathrm{n}}(\mathtt{a})=\mathscr{V}'_{\mathrm{n}}(\mathtt{a})$, and so $[\mathscr{V}_{\mathrm{n}}(\mathtt{a})\leftrightarrow\mathscr{V}'_{\mathrm{n}}(\mathtt{a})]\bullet\mathscr{V}'_{\mathrm{t}}(\mathtt{x})=\mathscr{V}'_{\mathrm{t}}(\mathtt{x})$:

For each $\mathtt{a}\in\mathtt{C}'$, either $\mathtt{a}$ is in $(\mathtt{BN}\cup\mathtt{A})$ or not. We will show that in both cases we have $[\mathscr{V}_{\mathrm{n}}(\mathtt{a})\leftrightarrow\mathscr{V}'_{\mathrm{n}}(\mathtt{a})]\bullet\mathscr{V}'_{\mathrm{t}}(\mathtt{x})=\mathscr{V}'_{\mathrm{t}}(\mathtt{x})$:

- If $\mathtt{a}\in(\mathtt{C}'\cap(\mathtt{BN}\cup\mathtt{A}))$, then since $\mathtt{a}\in\mathtt{C}'$ we know that $\mathscr{V}_{\mathrm{n}}(\mathtt{a})\#\mathscr{V}'_{\mathrm{t}}(\mathtt{x})$. Moreover, since $\mathtt{a}\in(\mathtt{BN}\cup\mathtt{A})$, then by definition of $\mathscr{V}'_{\mathrm{n}}$ we have that $\mathscr{V}'_{\mathrm{n}}(\mathtt{a})=\xi(\mathtt{a})$, and so $\mathscr{V}'_{\mathrm{n}}(\mathtt{a})\notin\big(\mathsf{im}(\mathscr{V}_{\mathrm{n}})\cup\mathsf{supp}(s)$, and hence $\mathscr{V}'_{\mathrm{n}}(\mathtt{a})\#\mathscr{V}'_{\mathrm{t}}(\mathtt{x})$. Thus for $\mathtt{a}\in(\mathtt{C}'\cap(\mathtt{BN}\cup\mathtt{A}))$ we have $[\mathscr{V}_{\mathrm{n}}(\mathtt{a})\leftrightarrow\mathscr{V}'_{\mathrm{n}}(\mathtt{a})]\bullet\mathscr{V}'_{\mathrm{t}}(\mathtt{x})=\mathscr{V}'_{\mathrm{t}}(\mathtt{x})$.

- If $\mathtt{a} \in (\mathtt{C}' - (\mathtt{BN} \cup \mathtt{A}))$, then we know that $\mathscr{V}_{\mathrm{n}}(\mathtt{a}) = \mathscr{V}'_{\mathrm{n}}(\mathtt{a})$, from which it follows that $[\mathscr{V}_{\mathrm{n}}(\mathtt{a}) \leftrightarrow \mathscr{V}'_{\mathrm{n}}(\mathtt{a})] \bullet \mathscr{V}'_{\mathrm{t}}(\mathtt{x}) = \mathscr{V}'_{\mathrm{t}}(\mathtt{x})$.

So we have $[\mathscr{V}_{\mathrm{n}}(\mathtt{a}) \leftrightarrow \mathscr{V}'_{\mathrm{n}}(\mathtt{a})]_{\mathtt{a} \in \mathtt{C}' \cup (\mathtt{C} \cap \mathtt{FN})} \bullet \mathscr{V}'_{\mathrm{t}}(\mathtt{x}) = \mathscr{V}'_{\mathrm{t}}(\mathtt{x})$. We conclude that

$$[\mathscr{V}_{\mathrm{n}}(\mathtt{a}) \leftrightarrow \mathscr{V}'_{\mathrm{n}}(\mathtt{a})]_{\mathtt{a} \in \mathtt{C}} \bullet \mathscr{V}_{\mathrm{t}}(\mathtt{x}) = \mathscr{V}'_{\mathrm{t}}(\mathtt{x}) \quad .$$

We turn now to the inductive steps for properties 8.B.5a and b. We consider an operator $\mathsf{op} \in \mathsf{Op}_{\mathbb{S}+\mathsf{sub}}$, together with name variables $\mathtt{c}_i \in \mathtt{N}$, for $i \in [1, \mathrm{arn}(\mathsf{op})]$; name variables $\mathtt{a}^j_k \in \mathtt{N}$, for $j \in [1, \mathrm{art}(\mathsf{op})]$ and $k \in [1, \mathrm{dep}_i(\mathsf{op})]$; and terms $\mathtt{t}_j \in T_{\mathbb{S}+\mathsf{sub},\mathbf{Set}_{\mathbb{N}}}(\mathtt{X})$ for $j \in [1, \mathrm{art}(\mathsf{op})]$. We let

$$\mathtt{t} = \mathsf{op}\left((\mathtt{c}_i)_{i \in [1,\mathrm{arn}(\mathsf{op})]}, \left(\left\langle \mathtt{a}^j_k \right\rangle_{k \in [1,\mathrm{dep}_j(\mathsf{op})]} \mathtt{t}_j\right)_{j \in [1,\mathrm{art}(\mathsf{op})]}\right)$$

and we will show that properties 8.B.5a and b hold of $\mathtt{t}$.

First we explain the inductive step for property 8.B.5a. We know that

$$\mathsf{supp}(\mathscr{V}(\mathtt{t})) \subseteq \mathsf{im}(\mathscr{V}_{\mathrm{n}}) \cup \bigcup \left\{ \mathsf{supp}(\mathscr{V}\left(\mathtt{t}_j\right)) \;\middle|\; j \in [1, \mathrm{art}(\mathsf{op})] \right\} \quad .$$

So

$$\mathscr{V}'_{\mathrm{n}}(\mathtt{BN} \cup \mathtt{A}) \cap \mathsf{supp}(\mathscr{V}(\mathtt{t})) \subseteq \left(\mathscr{V}'_{\mathrm{n}}(\mathtt{BN} \cup \mathtt{A}) \cap \mathsf{im}(\mathscr{V}_{\mathrm{n}})\right) \cup \bigcup_{j \in [1,\mathrm{art}(\mathsf{op})]} \left(\mathscr{V}'_{\mathrm{n}}(\mathtt{BN} \cup \mathtt{A}) \cap \mathsf{supp}(\mathscr{V}\left(\mathtt{t}_j\right))\right) \quad .$$

By definition of $\mathscr{V}'_{\mathrm{n}}$ we have $\mathscr{V}'_{\mathrm{n}}(\mathtt{BN} \cup \mathtt{A}) \cap \mathsf{im}(\mathscr{V}_{\mathrm{n}}) = \emptyset$. By the induction hypothesis, for each $j \in [1, \mathrm{art}(\mathsf{op})]$ we have $\mathscr{V}'_{\mathrm{n}}(\mathtt{BN} \cup \mathtt{A}) \cap \mathsf{supp}(\mathscr{V}\left(\mathtt{t}_j\right)) = \emptyset$. Hence

$$\mathscr{V}'_{\mathrm{n}}(\mathtt{BN} \cup \mathtt{A}) \cap \mathsf{supp}(\mathscr{V}(\mathtt{t})) = \emptyset$$

as required.

To conclude, we explain the inductive step for property 8.B.5b. We assume $\mathrm{WF}(\mathtt{t})$ and that for any $\mathtt{C} \subseteq_{\mathrm{f}} \mathtt{N}$ such that $(\mathtt{BN} \cup \mathtt{A}) \cap \mathtt{FN}(\mathtt{t}) \subseteq \mathtt{C}$ we have, for all $\mathtt{x}$ appearing in $\mathtt{t}$ and each $\mathtt{a} \in \mathtt{C}$, either $\mathtt{a} \in \mathtt{FN}(\mathtt{x})$ or $\mathscr{V}_{\mathrm{n}}(\mathtt{a}) \# \mathscr{V}'_{\mathrm{t}}(\mathtt{x})$. We must show that

$$[\mathscr{V}_{\mathrm{n}}(\mathtt{a}) \leftrightarrow \mathscr{V}'_{\mathrm{n}}(\mathtt{a})]_{\mathtt{a} \in \mathtt{C}} \bullet \mathscr{V}(\mathtt{t}) = \mathscr{V}'(\mathtt{t})$$

We will justify the following sequence of equations.

$$\begin{aligned}
&[\mathscr{V}_{\mathrm{n}}(\mathtt{a}) \leftrightarrow \mathscr{V}'_{\mathrm{n}}(\mathtt{a})]_{\mathtt{a} \in \mathtt{C}} \bullet \mathscr{V}(\mathtt{t}) \\
&\quad = [\mathscr{V}_{\mathrm{n}}(\mathtt{a}) \leftrightarrow \mathscr{V}'_{\mathrm{n}}(\mathtt{a})]_{\mathtt{a} \in \mathtt{C}} \bullet \mathsf{op}\left(\begin{array}{l} (\mathscr{V}_{\mathrm{n}}(\mathtt{c}_i))_{i \in [1,\mathrm{arn}(\mathsf{op})]}, \\ \left(\left\langle \mathscr{V}_{\mathrm{n}}(\mathtt{a}^j_k) \right\rangle_{k \in [1,\mathrm{dep}_j(\mathsf{op})]} \mathscr{V}\left(\mathtt{t}_j\right)\right)_{j \in [1,\mathrm{art}(\mathsf{op})]} \end{array}\right) && (8.\mathrm{B}.6a) \\
&\quad = \mathsf{op}\left(\begin{array}{l} \left(\mathscr{V}'_{\mathrm{n}}(\mathtt{c}_i)\right)_{i \in [1,\mathrm{arn}(\mathsf{op})]}, \\ \left(\left[\mathscr{V}_{\mathrm{n}}(\mathtt{a}) \leftrightarrow \mathscr{V}'_{\mathrm{n}}(\mathtt{a})\right]_{\mathtt{a} \in \mathtt{C}} \bullet \left\langle \mathscr{V}_{\mathrm{n}}(\mathtt{a}^j_k) \right\rangle_{k \in [1,\mathrm{dep}_j(\mathsf{op})]} \mathscr{V}\left(\mathtt{t}_j\right)\right)_{j \in [1,\mathrm{art}(\mathsf{op})]} \end{array}\right) && (8.\mathrm{B}.6b) \\
&\quad = \mathsf{op}\left(\begin{array}{l} \left(\mathscr{V}'_{\mathrm{n}}(\mathtt{c}_i)\right)_{i \in [1,\mathrm{arn}(\mathsf{op})]}, \\ \left(\begin{array}{l} \left\langle \mathscr{V}'_{\mathrm{n}}(\mathtt{a}^j_k) \right\rangle_{k \in [1,\mathrm{dep}_j(\mathsf{op})]} \\ \left[\mathscr{V}_{\mathrm{n}}(\mathtt{a}) \leftrightarrow \mathscr{V}'_{\mathrm{n}}(\mathtt{a})\right]_{\mathtt{a} \in \mathtt{C} \cup \left((\mathtt{BN} \cup \mathtt{A}) \cap \left\{ \mathtt{a}^j_k \;\middle|\; k \in [1,\mathrm{dep}_j(\mathsf{op})] \right\}\right)} \bullet \mathscr{V}\left(\mathtt{t}_j\right) \end{array}\right)_{j \in [1,\mathrm{art}(\mathsf{op})]} \end{array}\right) && (8.\mathrm{B}.6c)
\end{aligned}$$

$$= \mathsf{op}\left(\begin{array}{l}\left(\mathscr{V}'_{\mathrm{n}}(\mathtt{c}_i)\right)_{i\in[1,\mathrm{arn}(\mathsf{op})]}, \\ \left(\left\langle \mathscr{V}'_{\mathrm{n}}(\mathtt{a}^j_k)\right\rangle_{k\in[1,\mathrm{dep}_j(\mathsf{op})]} \mathscr{V}'\left(\mathtt{t}_j\right)\right)_{j\in[1,\mathrm{art}(\mathsf{op})]}\end{array}\right) \tag{8.B.6d}$$

Equation 8.B.6a expands the definition of $\mathscr{V}(\mathtt{t})$. To arrive at Equation 8.B.6b, we firstly expand the action of a product of group actions. Then we observe that, for any $i \in [1, \mathrm{arn}(\mathsf{op})]$, we have $[\mathscr{V}_{\mathrm{n}}(\mathtt{a}) \leftrightarrow \mathscr{V}'_{\mathrm{n}}(\mathtt{a})]_{\mathtt{a}\in\mathtt{C}}\left(\mathscr{V}_{\mathrm{n}}(\mathtt{c}_i)\right) = \mathscr{V}'_{\mathrm{n}}(\mathtt{c}_i)$. This is clearly true if $\mathtt{c}_i \in \mathtt{C}$. If $\mathtt{c}_i \notin \mathtt{C}$, then because $(\mathtt{BN} \cup \mathtt{A}) \cap \mathtt{FN}(\mathtt{t}) \subseteq \mathtt{C}$, and $\mathtt{c}_i \in \mathtt{FN}(\mathtt{t})$, we know that $\mathtt{c}_i \notin (\mathtt{BN} \cup \mathtt{A})$. Therefore, by definition of $\mathscr{V}'_{\mathrm{n}}$, we have $\mathscr{V}_{\mathrm{n}}(\mathtt{c}_i) = \mathscr{V}'_{\mathrm{n}}(\mathtt{c}_i)$.

To show equation 8.B.6c, we show that for each $j \in [1, \mathrm{art}(\mathsf{op})]$, $k' \in [0, \mathrm{dep}_j(\mathsf{op}) - 1]$,

$$\begin{aligned}&[\mathscr{V}_{\mathrm{n}}(\mathtt{a}) \leftrightarrow \mathscr{V}'_{\mathrm{n}}(\mathtt{a})]_{\mathtt{a}\in\mathtt{C}} \bullet \left\langle \mathscr{V}_{\mathrm{n}}(\mathtt{a}^j_k)\right\rangle_{k\in[1,k']} \mathscr{V}\left(\mathtt{t}_j\right) \\ &\qquad = \left\langle \mathscr{V}'_{\mathrm{n}}(\mathtt{a}^j_k)\right\rangle_{k\in[1,k']} [\mathscr{V}_{\mathrm{n}}(\mathtt{a}) \leftrightarrow \mathscr{V}'_{\mathrm{n}}(\mathtt{a})]_{\mathtt{a}\in\mathtt{C}\cup((\mathtt{BN}\cup\mathtt{A})\cap\left\{\mathtt{a}^j_k \mid k\in[1,k']\right\})} \bullet \mathscr{V}\left(\mathtt{t}_j\right) \quad . \end{aligned} \tag{8.B.7}$$

We show this property (8.B.7) by induction on $k' \in [0, \mathrm{dep}_j(\mathsf{op})]$. The base step, $k' = 0$, is trivial. As for the inductive step: if $\mathtt{a}^j_{k'+1} \notin (\mathtt{BN} \cup \mathtt{A})$ then $\mathscr{V}_{\mathrm{n}}(\mathtt{a}^j_{k'+1}) = \mathscr{V}'_{\mathrm{n}}(\mathtt{a}^j_{k'+1})$ and the step follows from the induction hypothesis. Otherwise, if $\mathtt{a}^j_{k'+1} \in (\mathtt{BN} \cup \mathtt{A})$, then property 8.B.5a ensures that we have $\mathscr{V}'_{\mathrm{n}}(\mathtt{BN} \cup \mathtt{A}) \cap \mathsf{supp}(\mathscr{V}\left(\mathtt{t}_j\right)) = \emptyset$. So certainly $\mathscr{V}'_{\mathrm{n}}(\mathtt{a}^j_{k'+1}) \# \mathscr{V}\left(\mathtt{t}_j\right)$, and hence

$$\mathscr{V}'_{\mathrm{n}}(\mathtt{a}^j_{k'+1}) \# \left\langle \mathscr{V}_{\mathrm{n}}(\mathtt{a}^j_{k+1})\right\rangle_{k\in[1,k']} \mathscr{V}\left(\mathtt{t}_j\right) \quad .$$

Finally, equation 8.B.6d follows from the induction hypothesis. To see this, for each $j \in [1, \mathrm{art}(\mathsf{op})]$, we let

$$\mathtt{C}_j = \mathtt{C} \cup \left((\mathtt{BN} \cup \mathtt{A}) \cap \left\{\mathtt{a}^j_k \;\middle|\; k \in \mathrm{dep}_j(\mathsf{op})\right\}\right) \quad .$$

It is clear that $(\mathtt{BN} \cup \mathtt{A}) \cap \mathtt{FN}(\mathtt{t}_j) \subseteq \mathtt{C}_j$. Moreover, for every $\mathtt{a} \in \mathtt{C}_j$ we have that

$$\text{for each } \mathtt{x} \text{ appearing in } \mathtt{t}_j \text{ we have either } \mathtt{a} \in \mathtt{FN}(\mathtt{x}) \text{ or } \mathscr{V}_{\mathrm{n}}(\mathtt{a}) \# \mathscr{V}'_{\mathrm{t}}(\mathtt{x}).$$

We will prove this statement by considering the subsets of $\mathtt{C}_j$ in which $\mathtt{a}$ may lie. If $\mathtt{a}$ is in $\mathtt{C}$ then this follows by assumption. If there is $k \in [1, \mathrm{dep}_j(\mathsf{op})]$ with $\mathtt{a} = \mathtt{a}^j_k$, and $\mathtt{a} \in \mathtt{BN}$, then, since $\mathrm{WF}(\mathtt{t})$, we know that $\mathtt{a} \in \mathtt{FN}(\mathtt{x})$. On the other hand, if $k \in [1, \mathrm{dep}_j(\mathsf{op})]$ with $\mathtt{a} = \mathtt{a}^j_k$, and $\mathtt{a} \in \mathtt{A}$, then we know $\mathscr{V}_{\mathrm{n}}(\mathtt{a}) \# s$. By definition of $\mathscr{V}'_{\mathrm{n}}$, we know that $\mathscr{V}'_{\mathrm{n}}(\mathtt{BN}(\mathtt{x})) \cap \mathsf{im}(\mathscr{V}_{\mathrm{n}}) = \emptyset$, and thus we can conclude that $\mathscr{V}_{\mathrm{n}}(\mathtt{a}) \# \mathscr{V}'_{\mathrm{t}}(\mathtt{x})$.

Thus properties 8.B.5a and b are established.

We conclude by using properties 8.B.5a and b to show that $\mathscr{V}(\underline{\mathtt{l}}, \mathtt{tar}) = \mathscr{V}'(\underline{\mathtt{l}}, \mathtt{tar})$. Condition И-GSOS$^+$-12 asserts that $\mathtt{FN}(\mathtt{tar}) \subseteq \mathtt{FN} \cup \mathsf{bn}(\underline{\mathtt{l}})$. It follows from Conditions И-GSOS$^+$-6 and И-GSOS$^+$-8 that $\mathtt{BN} \cap \mathtt{FN} = \emptyset$; hence

$$\mathtt{BN} \cap \mathtt{FN}(\mathtt{tar}) \subseteq \mathsf{bn}(\underline{\mathtt{l}}) \quad .$$

Moreover, for any $\mathtt{x} \in \mathtt{X}$ and any $\mathtt{a} \in \mathsf{bn}(\underline{\mathtt{l}})$ we can show that $\mathscr{V}_{\mathrm{n}}(\mathtt{a}) \# \mathscr{V}'_{\mathrm{t}}(\mathtt{x})$. Consider some $\mathtt{x} \in \mathtt{X}$ and $\mathtt{a} \in \mathsf{bn}(\underline{\mathtt{l}})$. By assumption, $\mathscr{V}_{\mathrm{n}}(\mathtt{a}) \# s$. For any $\mathtt{x} \in \mathtt{X}$ we have that $\mathsf{supp}(\mathscr{V}'_{\mathrm{t}}(\mathtt{x})) \subseteq \mathsf{supp}(s) \cup \mathscr{V}'_{\mathrm{n}}(\mathtt{BN}(\mathtt{x}))$ — this is a basic case of Prop. 8.4.3. The definition of $\mathscr{V}'_{\mathrm{n}}$ is such that $\mathscr{V}'_{\mathrm{n}}(\mathtt{BN})$ is disjoint from $\mathsf{im}(\mathscr{V}_{\mathrm{n}})$, and hence we must have $\mathscr{V}_{\mathrm{n}}(\mathtt{a}) \# \mathscr{V}'_{\mathrm{t}}(\mathtt{x})$.

We are now in a position to use property 8.B.5b. Condition И-GSOS$^+$-11 ensures that $\mathrm{WF}(\mathtt{tar})$. So we have that

$$[\mathscr{V}_{\mathrm{n}}(\mathtt{a}) \leftrightarrow \mathscr{V}'_{\mathrm{n}}(\mathtt{a})]_{\mathtt{a}\in\mathsf{bn}(\underline{\mathtt{l}})} \bullet \mathscr{V}(\mathtt{tar}) = \mathscr{V}'(\mathtt{tar}) \quad . \tag{8.B.8}$$

Now, for $\mathtt{a} \in (\mathsf{bn}(\underline{\mathtt{l}}) - \mathtt{BN})$ we have $\mathscr{V}_{\mathrm{n}}(\mathtt{a}) = \mathscr{V}'_{\mathrm{n}}(\mathtt{a})$. On the other hand, we know from property 8.B.5a that for $\mathtt{a} \in \mathtt{BN} \cap \mathsf{bn}(\underline{\mathtt{l}})$ we have $\mathscr{V}'_{\mathrm{n}}(\mathtt{a}) \# \mathscr{V}(\mathtt{tar})$. Hence

$$\left\langle \mathscr{V}_{\mathrm{n}}(\mathtt{a})\right\rangle_{\mathtt{a}\in\mathsf{bn}(\underline{\mathtt{l}})} \mathscr{V}(\mathtt{tar}) \;=\; \left\langle \mathscr{V}'_{\mathrm{n}}(\mathtt{a})\right\rangle_{\mathtt{a}\in\mathsf{bn}(\underline{\mathtt{l}})} [\mathscr{V}_{\mathrm{n}}(\mathtt{a}) \leftrightarrow \mathscr{V}'_{\mathrm{n}}(\mathtt{a})]_{\mathtt{a}\in\mathsf{bn}(\underline{\mathtt{l}})} \bullet \mathscr{V}(\mathtt{tar}) \quad .$$

So, combining with (8.B.8) we have

$$\langle \mathscr{V}_{\mathrm{n}}(\mathtt{a})\rangle_{\mathtt{a}\in\mathsf{bn}(\underline{\mathtt{l}})}\,\mathscr{V}(\mathtt{tar}) \;=\; \big\langle \mathscr{V}'_{\mathrm{n}}(\mathtt{a})\big\rangle_{\mathtt{a}\in\mathsf{bn}(\underline{\mathtt{l}})}\,\mathscr{V}'(\mathtt{tar}) \quad .$$

It follows that $\mathscr{V}(\underline{\mathtt{l}},\mathtt{tar}) = \mathscr{V}'(\underline{\mathtt{l}},\mathtt{tar})$. For instance, in the case $\underline{\mathtt{l}} = \mathtt{c}(\mathtt{b})$ we proceed as follows. We have, by Condition ν-GSOS$^+$-9, that $\mathtt{c}\in\mathtt{FN}$ and so, by Conditions ν-GSOS$^+$-6 and ν-GSOS$^+$-8, we have $\mathscr{V}_{\mathrm{n}}(\mathtt{c}) = \mathscr{V}'_{\mathrm{n}}(\mathtt{c})$. Hence

$$\begin{aligned}\mathscr{V}(\underline{\mathtt{l}},\mathtt{tar}) \;&=\; \mathsf{inj}_{\mathsf{binp}}\left(\mathscr{V}_{\mathrm{n}}(\mathtt{c}),\langle \mathscr{V}_{\mathrm{n}}(\mathtt{b})\rangle\,\mathscr{V}(\mathtt{tar})\right)\\ &=\; \mathsf{inj}_{\mathsf{binp}}\Big(\mathscr{V}'_{\mathrm{n}}(\mathtt{c}),\big\langle \mathscr{V}'_{\mathrm{n}}(\mathtt{b})\big\rangle\,\mathscr{V}'(\mathtt{tar})\\ &=\; \mathscr{V}'(\underline{\mathtt{l}},\mathtt{tar}) \quad .\end{aligned}$$

Other modes of communication are treated in a very similar manner.

To conclude, we explain why $\mathscr{V}'_{\mathrm{n}}$ provides fresh binders for C. Notice that for all $\mathtt{a}\in(\mathtt{BN}\cup\mathsf{bn}(l))$ we have that $\mathscr{V}'_{\mathrm{n}}(\mathtt{a}) = \xi(\mathtt{a})$: when $\mathtt{a}\in\mathtt{BN}$, this follows from the definition of $\mathscr{V}'_{\mathrm{n}}$; on the other hand, when $\mathtt{a}\in(\mathsf{bn}(l)-\mathtt{BN})$, we know that $\mathscr{V}_{\mathrm{n}}(\mathtt{a})\#s$, since $\mathscr{V}$ is an instantiation into s, and so $\mathtt{a}\in\mathtt{A}$. By looking at the codomain of the chosen ξ, we have $\mathscr{V}'_{\mathrm{n}}(\mathtt{a})\notin C$. Thus Lemma 8.4.7 is proved. □

Chapter 9

Concluding Discussion

We conclude by recalling the main contributions of this thesis, and then highlighting some new questions and research directions that are suggested by this work.

9.1 Contributions

We recall the main contributions of this thesis, following the outline of Section 1.2.

Indexed labelled transition systems and coalgebras. In Chapter 3, we recalled and developed models of name-passing that are based on coalgebras over presheaf categories. We provided an explicit characterisation of these models in terms of indexed labelled transition systems. Thus the coalgebraic approach of Fiore and Turi [2001] is related with transition system approach of Cattani and Sewell [2004].

Sheaf conditions on states spaces. We have investigated the relevance of a sheaf condition on the carriers of coalgebras. In Section 4.2 we introduced and developed a category of sheaves over $\mathbf{I}$, commonly called the Schanuel topos. We showed that the behaviour endofunctors on $\mathbf{Set}^{\mathbf{I}}$ of Section 3.2 restrict to the Schanuel topos. In Section 4.4 we introduced indexed labelled transition systems over presheaves on $\mathbf{B}$.

We showed, in Section 5.1, that the Schanuel topos is equivalent to a category of named-sets with symmetries, as proposed by Ferrari, Montanari, and Pistore [2002]. Thus the coalgebraic and labelled transition system models of name-passing are related with the work on History Dependent Automata. The named-sets can be seen as efficient presentations of sheaves.

In Chapter 7 we exploited the equivalence between the Schanuel topos and the category of nominal sets of Gabbay and Pitts [2001] to develop a model of name-passing behaviour in that setting. We introduced a nominal logic theory of transition systems for name-passing, translating the axioms on labelled transition systems from Chapter 3 to the language of nominal logic.

Name-for-name substitution. In Section 4.3 we introduced a coverage on the category $\mathbf{F}$ in order to restrict the adjunction between $\mathbf{Set}^{\mathbf{I}}$ and $\mathbf{Set}^{\mathbf{F}}$ to an adjunction between related sheaf subcategories. It was seen that the sheaf condition for this coverage on $\mathbf{F}$ has a notably simple description.

To work with name-for-name subsitutions in the context of nominal sets, we introduced in Section 7.3 a theory of nominal substitutions, proving an equivalence between the category of models of this theory and the category of sheaves on $\mathbf{F}$.

Structured coalgebras and wide open bisimulation. In Section 2.4 we developed a theory of structured coalgebras. This was used in Section 3.4, and in similar developments in later chapters, to give a model theoretic account of wide open bisimulation and to develop a more refined model theory.

A rule format for name-passing calculi. In Chapter 8 we presented a concrete rule format for name-passing calculi. The format was extracted from the abstract developments of Section 6.2, using the structured-coalgebra model of behaviour over nominal sets and the model of syntax over nominal substitutions.

Final coalgebraic bisimulations. The coalgebraic aspects of this thesis have been developed without reference to final coalgebras. In Section 2.5 we provided basic results for relating notions of bisimulation. In Section 5.2 we developed a theory of final bisimulations, providing an algorithm for computing them. In Section 6.2.1 we established results about congruence of bisimilarity in complete generality, without recourse to final coalgebras or bialgebras, thus redeveloping the presentation of Turi and Plotkin [1997].

9.2 Other rule formats for name-passing calculi

The concrete rule format developed in Chapter 8 is not the only format proposed for name-passing calculi. A distinguishing feature of the rule format presented here, though, is the model-theoretic perspective from which it is developed.

We now very briefly discuss approaches to rule formats for name-passing proposed by Bernstein [1998], by Weber and Bloom [1996], and by Ziegler, Miller, and Palamidessi [2006].

Bernstein [1998]: Promoted tyft/tyxt. Bernstein has presented a generalisation of the tyft/tyxt format of Groote and Vaandrager [1992] that allows higher-order systems to be described by allowing arbitrary terms to appear in the transition labels. Her format is sufficiently general as to allow a semantics of the π-calculus. There, syntax is not considered up-to α-equivalence, and rules are used not only for describing the behaviour, but also for deriving the free and bound names of terms. The format shows that a variant of wide open bisimilarity is a congruence for the π-calculus. Because the assignment of free and bound names is part of the transition relation, the notion of bisimilarity that arises will distinguish near-identical processes when they have different free names. For instance, her notion of bisimilarity distiguishes the process $[a = b]\,0$ from the nil process 0 [see Bernstein, 1998, Prop. 5.2].

Weber and Bloom [1996]: Meta-π. Weber and Bloom have presented a framework for adding operators to the π-calculus. The semantics of these operators are specified in terms of rules in a GSOS-like format. There is a built-in restriction operator, which distributes over certain operators. Syntax is considered up-to α-equivalence with respect to the binding of the restriction operator. Other operators involving binding, though, are considered in the style of higher-order abstract syntax: for instance, the input operator constructs a process out of a channel, together with a function from channels to processes. In this respect, the framework is similar in style to an earlier congruence format for syntax with binding proposed by Bloom and Vaandrager [1994].

A notion of bisimulation is introduced, and it is proved that the corresponding bisimilarity is a congruence. It seems that, for the π-calculus, this notion of bisimulation is the open bisimulation of Sangiorgi [1996], because of the way that name variables and constants are treated differently. In their article, however, they make no connections of these kinds.

Full details of this framework, and proofs, can be found in Weber's PhD thesis [1995, Secs. 8–13].

Ziegler, Miller, and Palamidessi [2006]: tyft/tyxt in $FO\lambda^{\Delta\nabla}$. Recently, Ziegler *et al.* have considered the tyft/tyxt format of Groote and Vaandrager [1992] within the logical framework $FO\lambda^{\Delta\nabla}$ of Miller and Tiu [2005]. The work is carried out almost entirely in this proof-theoretic domain. By working with higher-order abstract syntax, and with the quantifier ∇ for quantifying 'new' names, they arrive at a rule format that is suitable for the π-calculus. They avoid having to impose conditions such as those in Figures 8.2 and 8.3 by treating term parameters as metavariables, and working with a variable-capture convention.

A notion of bisimilarity is introduced, which is shown to correspond to the open bisimilarity of Sangiorgi [1996]. It is shown that, for systems defined using rules in their format, this bisimilarity is a congruence.

9.3 Research directions

We now discuss some miscellaneous issues that arose in the development of this thesis. We mention refinements and revisions of the model for the name-passing case: different approaches to the behavioural model theory (Section 9.3.1); different powersets for different notions of nondeterminism (Section 9.3.2); different logical frameworks (Section 9.3.3); and models that are complete for Positive GSOS (Section 9.3.4). It is hoped that the developments of this thesis can serve as a foundation for the mathematical study of yet more elaborate process calculi and programming languages: here we outline possible approaches to open bisimulation (Section 9.3.5) and guarded recursion (Section 9.3.6).

9.3.1 Different operational models for name-passing

There is a quirk that is common to both the labelled transition system and the coalgebraic models developed in Chapter 3. For the variation in name contexts, one form of Kripke semantics is used, namely functor categories; for the behavioural aspects of the model, another form of Kripke semantics is used, either coalgebras or transition relations. It is convenient to distinguish these two aspects of the models in this way, but, as other authors have shown, it is not necessary.

The models of name-passing proposed by Honsell, Lenisa, Montanari, and Pistore [1998] and by Baldamus [2000] solely involve coalgebras over **Set**. The coalgebra structure assigns to each state not only a description of its possible evolution, but also the set of names 'free' in that state. These authors report a problem of 'junk' in the models, which indicates that the models are far from canonical. Thus these model theories are arguably less successful than those studied here at providing an abstract account of name-passing behaviour.

The semantics of the π-calculus given by Bernstein [1998, Sec. 5], mentioned above, is given soley in terms of inductively defined transition relations. We have already mentioned that the natural notion of bisimilarity for this semantics is obscure.

A model of Cattani, Stark, and Winskel [1997] is based entirely on functor categories. A category of presheaf categories is a category of domains, and the model of name-passing is based on functors from **I** to this category. There remains an inequality here, in that the domains of behaviour come first, and the variation of name contexts comes second.

We turn now to consider models in the style of Chapter 7. In Section 7.5 we studied models of name-passing in terms of transition systems in the category of nominal sets. The models of Buscemi and Montanari [2002] and of Montanari and Pistore [2005] also work with categories of group actions, but *without* the finite support restriction, and so, for instance, the treatment of bound output is more involved.

9.3.2 Different powerset functors

Throughout this thesis we have focussed on a notion of non-determinism based on the pointwise powerset functor $\mathscr{P}$ on $\mathbf{Set}^{\mathbf{I}}$: for $P \in \mathbf{Set}^{\mathbf{I}}$, and $C \in \mathbf{I}$, we have that $(\mathscr{P}(P))(C) = \mathscr{P}(P(C))$. In Theorem 4.2.6(6) we established that this operator restricts to the category $\mathbf{Sh}(\mathbf{I})$ of sheaves, and in Section 7.1.5 we lifted this operator to the category **Nom** of nominal sets to obtain the endofunctor $\mathscr{P}_{\mathrm{sb}}$ that we called the support-bounded powerset.

A topos-theoretic approach. The categories $\mathbf{Set}^{\mathbf{I}}$, $\mathbf{Sh}(\mathbf{I})$ and **Nom** are all toposes. From a topos-theoretic perspective, the natural endofunctor for non-determinism is the covariant powerobject functor [see *e.g.* Johnstone, 2002, Sec. A2.3]. This construction has been described concretely on **Nom** as the finite-support powerset $\mathscr{P}_{\mathrm{fs}}$ by Gabbay and Pitts [2001, Example 3.5(iv)]. On $\mathbf{Set}^{\mathbf{I}}$, the covariant powerobject functor is an especially elaborate construction. In all cases the powerobject operator differs from the construction inspired by the pointwise considerations.

Theories of lattice. In any topos, the powerobject construction provides free models of the second-order theory $\mathbb{CJSL}$ of partially ordered objects in which every internal subobject has a join. Models of the theory $\mathbb{CJSL}$ in **Set** are complete join semilattices. (By a simple argument, a lattice with all joins also has all meets [Crawley and Dilworth, 1973, p. 9]; here, though, we are not concerned with meets.)

The theory $\mathbb{CJSL}$ is not the only way to capture the notion of complete join semilattice, though. There is an alternative presentation that has a more algebraic flavour. We let $\mathbb{ORD}$-$\mathbb{JSL}$ be the infinitary algebraic theory whose signature contains, for each ordinal α, an α-ary join operator $\bigvee_\alpha$. There is a class of equations in $\mathbb{ORD}$-$\mathbb{JSL}$ ensuring that the join operators are idempotent and expressing appropriate laws of commutativity, associativity, and distributivity. This theory can be interpreted in any category with small products.

In the category of sets, the two theories, $\mathbb{CJSL}$ and $\mathbb{ORD}$-$\mathbb{JSL}$, have exactly the same models. This is not true in **Nom** or in $\mathbf{Set}^{\mathbf{I}}$, however; from a logical standpoint, this is because the proof of equivalence of the theories involves the axiom of choice [see *e.g.* Crawley and Dilworth, 1973, 2.1].

The pointwise powerset functor on $\mathbf{Set}^{\mathbf{I}}$ provides free models of the theory $\mathbb{ORD}$-$\mathbb{JSL}$. For every object $C \in \mathbf{I}$, the corresponding point $1 \to \mathbf{I}$ induces a functor $(-) \bullet C : \mathbf{Set}^{\mathbf{I}} \to \mathbf{Set}$, sending a presheaf $P \in \mathbf{Set}^{\mathbf{I}}$ to the set $P(C)$. This functor has a left adjoint, and so preserves all limits, and in particular preserves models of $\mathbb{ORD}$-$\mathbb{JSL}$; thus we see that the free models of $\mathbb{ORD}$-$\mathbb{JSL}$ in $\mathbf{Set}^{\mathbf{I}}$ are the pointwise powersets. A similar argument can be made for the sheaf category $\mathbf{Sh}(\mathbf{I})$. Indeed, in the category **Nom** of nominal sets, the carrier of the free model of $\mathbb{ORD}$-$\mathbb{JSL}$ on a nominal set X is the support-bounded powerset $\mathscr{P}_{\mathrm{sb}}X$, as introduced in Section 7.1.5.

A different behaviour endofunctor. From a pragmatic point of view, we have seen in Chapters 3, 4 and 7 that the pointwise powerset endofunctor is sufficient for modelling non-determinism in systems such as the π-calculus. The topos-theoretic ideas above suggest alternative behaviour endofunctors, such as the endofunctor $(\mathscr{P}_{\mathrm{fs}}L_{\mathrm{g}})$ on **Nom**, using the finite-support powerset. Indeed, to give a $(\mathscr{P}_{\mathrm{fs}}L_{\mathrm{g}})$-coalgebra is to give a nominal set X together with an equivariant subset of $X \times L_{\mathrm{g}}X$. This is rather similar to the kinds of transition system used by Gabbay [2003, Sec. 3.2] in his treatment of the π-calculus.

Relating the two notions of non-determinism. We briefly consider the relationship between the endofunctor $B_{\mathrm{g}} = \mathscr{P}_{\mathrm{sb}}L_g$ on **Nom** which has been used so far, and the alternative endofunctor $(\mathscr{P}_{\mathrm{fs}}L_{\mathrm{g}})$ on **Nom** that is suggested by topos-theoretic ideas. It is clear that $\mathscr{P}_{\mathrm{sb}}$ is a subfunctor of $\mathscr{P}_{\mathrm{fs}}$, and so the category of $(\mathscr{P}_{\mathrm{sb}}L_{\mathrm{g}})$-coalgebras embeds into the category of $(\mathscr{P}_{\mathrm{fs}}L_{\mathrm{g}})$-coalgebras.

In fact, the embedding $(\mathscr{P}_{\mathrm{sb}}L_{\mathrm{g}})$-Coalg $\hookrightarrow$ $(\mathscr{P}_{\mathrm{fs}}L_{\mathrm{g}})$-Coalg has a retraction, mapping a $(\mathscr{P}_{\mathrm{fs}}L_{\mathrm{g}})$-coalgebra $h : X \to \mathscr{P}_{\mathrm{fs}}(L_{\mathrm{g}}(X))$ to the restricted B_{g}-coalgebra with carrier

$$\bigcup \left\{ S \subseteq X \;\middle|\; \forall x \in S.\, h(x) \subseteq_{\mathrm{sb}} L_{\mathrm{g}}S \right\} \quad .$$

It follows from Theorem 2.5.7 that, for $(\mathscr{P}_{\mathrm{sb}}L_{\mathrm{g}})$-coalgebras, $(\mathscr{P}_{\mathrm{sb}}L_{\mathrm{g}})$-bisimulation is the same thing as $(\mathscr{P}_{\mathrm{fs}}L_{\mathrm{g}})$-bisimulation.

Behaviour in categories of substitutions. Once one begins to consider different notions of powerset, it becomes possible to consider behaviour endofunctors for name-passing over categories such as $\mathbf{Set}^{\mathbf{F}}$ and **NomSub**, as we now explain.

The definition of the endofunctor L_{g} on **Nom**, given in equation 7.1.7a, can be considered as the definition of an endofunctor on **NomSub**, by following the developments of Section 7.3.2. We thus have an endofunctor L_{g} on **NomSub** for deterministic ground systems.

The problem that remains is that of finding an appropriate powerset endofunctor on **NomSub**. A first idea that might spring to mind is to lift the endofunctor $\mathscr{P}_{\mathrm{sb}}$ on **Nom** along the forgetful functor **NomSub** $\to$ **Nom**. A natural way to do this is to consider the pointwise powerset endofunctor on the presheaf category $\mathbf{Set}^{\mathbf{F}}$, and translate this construction into the category **NomSub**.

The model arising from the resulting endofunctor $\mathscr{P}_{\mathrm{sb}}$ on **NomSub** is not right, though. For any $X \in$ **NomSub**, and any coalgebra $h : X \to \mathscr{P}_{\mathrm{sb}}L_{\mathrm{g}}X$, we have the formulas

$$l \in h(x) \implies [b/a]l \in h([b/a]x) \tag{9.3.1a}$$

$$l \in h([b/a]x) \implies \exists l' \in h(x).\, [b/a]l' = l \quad . \tag{9.3.1b}$$

Formula (9.3.1a) is a common property of name-passing calculi. (It is not ubiquitous, though, for the mismatch operator of the π-calculus will violate it.) Formula (9.3.1b), though, is obscure and unwanted.

Similar problems arise if one lifts the endofunctor $\mathscr{P}_{\mathrm{fs}}$ on **Nom** along **NomSub** $\to$ **Nom**, and also if one considers the free finite-join-semilattice monad on **NomSub**.

The developments of this subsection suggest a different powerset endofunctor on **NomSub**: this category is a topos, and so we can work with its powerobject endofunctor $\mathscr{P}$. This endofunctor does not arise from lifting either of the endofunctors, $\mathscr{P}_{\mathrm{sb}}$ or $\mathscr{P}_{\mathrm{fs}}$, considered on **Nom**. It is left for the reader to calculate an explicit description of $\mathscr{P}$, but we note here that a coalgebra for the resulting endofunctor $(\mathscr{P}L_{\mathrm{g}})$ on **NomSub** can be given by a nominal substitution X together with a subobject $\longrightarrow$ of $X \times L_{\mathrm{g}}X$. Such a subobject necessarily satisfies the property

$$\text{If } x \longrightarrow l \text{ then } [b/a]x \longrightarrow [b/a]l$$

(but, in general, not its converse). Thus we arrive at a model for wide open bisimulation in the π-calculus (without mismatch).

The more general results of this thesis (and Theorem 6.2.1 in particular) can be proved at the level of final lifting spans. From this perspective, it may be profitable to move away from modelling behaviour in categories of coalgebras, and to work instead with suitable categories of transition systems, and suitable functional bisimulations between them.

9.3.3 Different logical frameworks for syntax with variable binding

The formats of Weber and Bloom [1996] and of Ziegler *et al.* [2006] work with higher-order abstract syntax, and this approach seems to lead to simpler conditions. With this in mind, it may be profitable to use frameworks other than nominal logic as a logical basis for the notions of rule. To adopt the $FO\lambda^{\Delta\nabla}$ framework of Miller and Tiu [2005] within the general approach presented here, it would be necessary to develop a model theory, perhaps following the proposals of Miculan and Yemane [2005] and of Schöpp [2006].

9.3.4 Partial orders and completeness for Positive GSOS

In the developments of Section 6.3 and Chapter 8 we were concerned with finding explicit, logical characterisations of the abstract rules that arose from the model theory. An interesting question is: how complete is this characterisation — are all the abstract rules definable in terms of concrete rules?

The answer is: no. There are abstract rules that do not arise from rule structures. The primary reason for this is that we have not allowed negative premises. Negative premises increase the expressivity of the GSOS rule format. As Turi and Plotkin [1997, Thm. 1.1] explain, a complete characterisation of the abstract rules in the form studied in Chapter 6 must allow for rules with negative premises.

There are two possible resolutions for this mismatch between the concrete and abstract domains. The first resolution is to achieve completeness by allowing a more generous class of concrete rules. This amounts to considering negative premises for rules for name-passing. We leave the precise definitions and calculations to future work (if indeed the problems that arise in this direction are sufficiently interesting to warrant much study).

The second possible resolution is to keep the same concrete rule format, and constrain the model so as to achieve a completeness result. One can ask: what is a category-theoretic model for Positive GSOS? We now outline some issues related to this question, for the simple case without binding or name-passing.

A model of Positive GSOS. The model that we propose here involves working with the category **Poset** of partial orders and monotone functions between them, instead of with the category **Set** of sets. The category **Poset** is complete, and so we can model syntax (without binding) there, following Section 6.1. For behaviour, an appropriate endofunctor on **Poset** is the endofunctor $\mathscr{D}(Lab \times (-))$; here $\mathscr{D}$ is the endofunctor mapping a poset to the poset of its down-closed subsets.

The appropriate abstract notion of rule is that of natural transformations of the form of Definition 6.2.11, *i.e.* natural transformations

$$\Sigma((-) \times \mathscr{D}(Lab \times (-))) \longrightarrow \mathscr{D}(Lab \times T(-))$$

between endofunctors on **Poset**. The positivity of rules amounts to the requirement that these natural transformations be componentwise monotone.

As in Section 6.3.4, the collection of abstract rules forms a complete lattice: working with a collection of concrete rules amounts to taking joins of the corresponding abstract rules. The single concrete rules correspond to the completely prime elements of the lattice.

Precongruence of similarity. A distinguishing property of a system defined by GSOS rules with no negative premises is that *the similarity preorder is a precongruence*: that is, the greatest simulation is respected by the operators of the syntax. For example, for any unary operator op, if there is a simulation relating processes x and y, then there will also be a simulation relating processes $\mathrm{op}(x)$ and $\mathrm{op}(y)$. This property can be established at the abstract level, as follows.

In this order-enriched setting, simulations can be defined in terms of spans of coalgebras by considering lax morphisms of coalgebras, following Fiore [1996, Sec. 7]. The precongruence of similarity can be understood by adapting Theorem 6.2.1 and Corollary 6.2.3 to the order-enriched setting.

A model of Positive GSOS for name-passing. There is still work to be done if this tighter model of Positive GSOS is to be adapted to the name-passing setting. For instance, it is not clear how to extend the support-bounded powerset functor $\mathscr{P}_{\mathrm{sb}}$ on **Nom** to give an endofunctor on a suitable category of 'nominal posets'.

9.3.5 Semantics of proper open bisimulation

It could be argued that wide open bisimilarity (Definition 3.1.13) is *too* fine, in that it distinguishes processes that could perhaps could never be told apart. For instance, the process $P_1 = \nu d.\bar{c}d.[c=d]\bar{c}d.0$ is not wide open bisimilar with $P_2 = \nu d.\bar{c}d.0$. This is essentially because the subprocesses $[c=d]\bar{c}d.0$ and 0 are not wide open bisimilar. Some would assert that this is unreasonable, because names 'generated' by the ν operator, such as the name d in process P_1, should never be identified with other names.

Open bisimulation. Sangiorgi [1996] has introduced a refined notion of bisimulation that resolves this issue. To define this notion, we recall the concept of *distinction* as a finite symmetric irreflexive binary relation on the set $\mathcal{N}$ of names. We say that a substitution function $f : \mathcal{N} \to \mathcal{N}$ *respects* a distinction $\#$ if whenever $c\#c'$ then we have $f(c) \neq f(d)$; in this circumstance we write $(f\#)$ for the image of the relation $\#$ under f. We write $\mathsf{fn}(\#)$ for the set of names appearing in the distinction $\#$. For any binary relation R, we write $R^{=}$ for its symmetric closure.

Definition 9.3.2 (*c.f.* Sangiorgi [1996, Defn. 6.2 and Sec. 7])**.** Let $\{R_\#\}_\#$ be a distinction-indexed binary relation on π-calculus terms. (So for each distinction $\#$, a binary relation $R_\#$ is given.)

This family $\{R_\#\}_\#$ is an *open simulation* if for all distinctions $\#$ and all substitutions f that respect $\#$, then whenever $p\,R_\#\,q$ and $[f]p \xrightarrow[\pi]{\ell} p'$ with $\mathsf{bn}(\ell) \cap \mathsf{fn}(p,q,\#) = \emptyset$, we have

1. if $\ell = \bar{c}(z)$ then there exists q' such that $[f]q \xrightarrow[\pi]{\ell} q'$ and $p'\,R_{\#'}\,q'$, where
$$\#' = (f\#) \cup (\{z\} \times \mathsf{fn}([f]p, [f]q))^{=} \quad , \text{ and}$$
2. if ℓ is not a bound output label then there exists q' such that $[f]q \xrightarrow[\pi]{\ell} q'$ and $p'\,R_{f\#}\,q'$.

A distinction indexed family of binary relations $\{R_\#\}_\#$ on π-calculus terms is an *open bisimulation* if both $\{R_\#\}_\#$ and $\{R_\#^{\mathrm{op}}\}_\#$ are open simulations.

A model of Ghani, Yemane and Victor. Ghani *et al.* [2004] suggest a modification of the coalgebraic analysis of Fiore and Turi [2001] to capture this notion of open bisimulation. They suggest that, in place of the category $\mathbf{F}$ of finite sets of names and functions between them, a category $\mathbf{F}_\#$ is used, defined as follows: objects are finite sets of names and distinction relations over them; morphisms are functions that preserve these distinctions. (They denote this category by $\mathcal{D}$.)

However, the model presented there uses the pointwise finite powerset functor on $\mathbf{Set}^{\mathbf{F}_\#}$, and thus the problems highlighted in (9.3.1), at the end of Section 9.3.2, will arise. In their model, the transitions must be preserved and reflected by substitutions. The obscure substitution-reflection property does not hold for the π-calculus, and so, as it stands, this model cannot be used to model name-passing.

A possible remedy. The development at the end of Section 9.3.2 suggests a simple remedy for the problems with the model of Ghani *et al.* [2004]. Instead of using a pointwise powerset functor on $\mathbf{Set}^{\mathbf{F}_\#}$, the topos-theoretic powerobject functor could be used. The details have to be checked, however. The resulting model will not be able to model mismatching, but many argue that open bisimilarity is not relevant for the calculus with mismatch anyway [*e.g.* Sangiorgi, 1996, Sec. 8].

Structured coalgebras for open bisimulation. We now sketch a model for open bisimulation that is suggested from the developments in this thesis. The definition of open bisimulation for the π-calculus can be straightforwardly reformulated at the model-theoretic level for $\mathbf{F}$-IL_gTSs, as introduced in Section 3.4.

There is a forgetful functor $\mathbf{F}_\# \to \mathbf{F}$, inducing an inclusion of $\mathbf{Set}^{\mathbf{F}}$ into $\mathbf{Set}^{\mathbf{F}_\#}$. One can thus define a *distinction indexed relation* between presheaves over $\mathbf{F}$ as a relation in $\mathbf{Set}^{\mathbf{F}_\#}$ between the induced presheaves over $\mathbf{F}_\#$. It is then straightforward to translate the remainder of Definition 9.3.2 to this abstract setting. Thus open bisimulation can be defined for $\mathbf{F}$-IL$_g$TSs.

For a coalgebraic model, the important step is to introduce a new category $\mathbf{I}_\#$: objects are finite sets of names and distinction relations on them, and morphisms $(C, \#_C) \to (D, \#_D)$ are injections $\iota : C \to D$ that preserve and reflect the distinction structure, in the sense that

$$\forall c, c' \in C.\ c \#_C c' \iff \iota(c) \#_C \iota(c') \quad .$$

On the category $\mathbf{Set}^{\mathbf{I}_\#}$ there are two choices for the name generation operator, δ. Both are defined in a manner analgous to (3.2.4), although care must be taken over the distinction relation. The first operator, which we also denote δ, describes the generation of new names that are not marked as distinct from the other names. The second operator, which we denote $\delta^\#$, describes the generation of new names that are marked (in the distinction relation) as distinct from all the other names in the current context.

We can now define a behaviour endofunctor $B_{g\#}$ on $\mathbf{Set}^{\mathbf{I}_\#}$ using a definition analogous to equations 3.2.15 and 3.2.16. The operator δ is used for the bound input component, and the operator $\delta^\#$ is used for the bound output component.

The inclusion functor $\mathbf{I}_\# \to \mathbf{F}_\#$ induces a forgetful functor $U^{\mathbf{I}_\#}_{\mathbf{F}_\#} : \mathbf{Set}^{\mathbf{F}_\#} \to \mathbf{Set}^{\mathbf{I}_\#}$. Now, $U^{\mathbf{I}_\#}_{\mathbf{F}_\#}$-structured $B_{g\#}$-coalgebras are models for open bisimulation, just as $U^{\mathbf{I}}_{\mathbf{F}}$-structured B_g-coalgebras are models for wide open bisimulation.

The forgetful functors $\mathbf{F}_\# \to \mathbf{F}$, $\mathbf{I}_\# \to \mathbf{I}$ suggest how to convert a $U^{\mathbf{I}}_{\mathbf{F}}$-structured B_g-coalgebra into a $U^{\mathbf{I}_\#}_{\mathbf{F}_\#}$-structured $B_{g\#}$-coalgebra, and it is then straightforward to correspond $U^{\mathbf{I}_\#}_{\mathbf{F}_\#}$-structured $B_{g\#}$-bisimulations with open bisimulations on the corresponding $\mathbf{F}$-IL$_g$TSs.

Open bisimulation and nominal sets. In Chapter 7 of this thesis, we have seen that the theory of nominal sets provides a elegant framework within which a model of name-passing behaviour can be presented. It remains to be seen how well-suited the framework of nominal sets is to more sophisticated notions of behaviour.

As regards the models of open bisimulation introduced briefly in this subsection, a 'nominal' presentation might be found as follows. The set of all distinction relations can be considered as a nominal set with an equivariant lattice structure. One can then study notions of 'nominal presheaf' over this lattice, by considering the notion of presheaf as internal to the category of nominal sets [as described by, *e.g.* Johnstone, 2002, Sec. B2.3], in order to arrive at a category related to $\mathbf{Set}^{\mathbf{I}_\#}$.

We leave the full investigation of these ideas for possible future research.

9.3.6 Semantics of guarded recursion

An operational semantics for recursive definitions is often given by a rule of the following form [see *e.g.* Milner, 1989, Sec. 2.9].

$$\frac{P\{\mathbf{fix}(X = P)/X\} \xrightarrow{l} P'}{\mathbf{fix}(X = P) \xrightarrow{l} P'} \tag{9.3.3a}$$

This rule is certainly not in the GSOS format: there are variables in the syntax, and the premise contains a substitution. Reasoning about recursive definitions in this way can difficult. For these reasons, the following simpler rule has been suggested.

$$\frac{P \xrightarrow{l} P'}{\mathbf{fix}(X = P) \xrightarrow{l} P'\{\mathbf{fix}(X = P)/X\}} \tag{9.3.3b}$$

Badouel and Darondeau [1991] provide conditions under which Rule (9.3.3a) is equivalent to Rule (9.3.3b). These conditions include the requirement that the recursion is *guarded*: informally, occurrences of X in P are all preceded by action prefixes.

Rule (9.3.3b) is roughly in the shape of a GSOS rule. It is not in the GSOS format because variables are involved, and transitions are over open terms, and there is a substitution in the conclusion target. All these issues have been addressed in the development of this thesis, except that term-for-name substitution has not been considered.

We now outline how the developments of this thesis can be adapted to explain rules of the form (9.3.3b).

It is not difficult to invent a 'nominal algebraic theory' for term-for-name substitutions. The substitution algebras of Fiore, Plotkin, and Turi [1999, Defn. 3.1] provide inspiration. Writing **SubstAlg** for the resulting category of models, we have a forgetful functor $U : \mathbf{SubstAlg} \to \mathbf{Nom}$. Fixing the set Lab of labels, we let B be the endofunctor $\mathscr{P}(Lab \times (-))$ on **Nom**. The theory of U-structured B-coalgebras provides an appropriate model of behaviour for understanding Rule (9.3.3b). To model syntax, one must show that the nominal algebraic signatures of Definition 7.4.1 can be modelled in **SubstAlg**, giving rise to a monad T on **SubstAlg**. (This development is carried out by Fiore *et al.* [1999].) The Rule (9.3.3b) can then be seen as recursion data for lifting the monad T on **SubstAlg** to the category of U-structured B-coalgebras.

Rule formats for recursion have been proposed before, by various authors: for instance the format of Middelburg [2001] allows recursion. From the category-theoretic perspective, there have been developments by Turi [1997, Sec. 8], Plotkin [2001] and Klin [2004], although these three reports side-step the issue of variable binding.

Future research. By working with increasingly sophisticated notions of behaviour and of substitution, we are able to study more elaborate process calculi and programming languages. For instance, the kinds of model considered above for the semantics of recursion provide a first step towards models for the higher-order systems, involving the communication of processes, as permitted in calculi such as CHOCS [Thomsen, 1995] and HOπ [Sangiorgi, 1992]. Indeed, part of the presentation of CHOCS considered by Amadio and Dam [1995] seems to already fit into the framework alluded to above. One might also investigate the kinds of model theory required for calculi involving the communication of encrypted values and keys, as occurs in the spi-calculus of Abadi and Gordon [1999].

Bibliography

Note: The abbreviated reference [SW01] for [Sangiorgi and Walker, 2001] has been used in Chapter 3.

M. Abadi and A. D. Gordon. A calculus for cryptographic protocols: The spi calculus. *Inform. and Comput.*, 148(1):1–70, 1999. (Cited on page 235.)

S. Abramsky. A domain equation for bisimulation. *Inform. and Comput.*, 92:161–218, 1991. (Cited on page 14.)

L. Aceto, W. Fokkink, and F. Vaandrager. Structural operational semantics. In Bergstra, Ponse, and Smolka, editors, *Handbook of Process Algebra*. Elsevier, 2001. Available in BRICS Report Series, RS-99-30, University of Aarhus. (Cited on pages 15 and 153.)

P. Aczel. *Non-Well-Founded Sets*, volume 14 of *CSLI Lecture Notes*. CSLI, 1988. (Cited on pages 14 and 22.)

J. Adámek, S. Milius, and J. Velebil. A general final coalgebra theorem. *Math. Structures Comput. Sci.*, 15:409–432, 2005. (Cited on page 22.)

J. Adámek and J. Rosický. *Locally Presentable and Accessible Categories*. Number 189 in London Math. Soc. Lecture Note Ser.. Cambridge University Press, 1994. (Cited on page 111.)

R. M. Amadio and M. Dam. Reasoning about higher-order processes. In *Proceedings of Theory and Practice of Software Development, Sixth International Joint Conference CAAP/FASE (TAPSOFT'95)*, volume 915 of *Lecture Notes in Comput. Sci.*, pages 202–216. Springer, 1995. (Cited on page 235.)

E. Badouel and P. Darondeau. On guarded recursion. *Theoret. Comput. Sci.*, 82(2):403–408, 1991. (Cited on page 235.)

M. Baldamus. Compositional constructor interpretation over coalgebraic models for the π-calculus. In *Proceedings of the Third International Workshop on Coalgebraic Methods in Computer Science (CMCS'00)*, volume 33 of *Electron. Notes Theor. Comput. Sci.*, pages 13–41, 2000. (Cited on page 229.)

H. P. Barendregt. *The Lambda Calculus: Its Syntax and Semantics*. Number 103 in Studies in Logic and the Foundations of Mathematics. Elsevier, 1981. (Cited on page 17.)

M. Barr. Coequalisers and free triples. *Math. Z.*, 116:307–322, 1970. (Cited on page 137.)

M. Barr and C. Wells. *Toposes, Triples and Theories*. Springer-Verlag, 1984. Republished in: Reprints in Theory Appl. of Categ., No. 12 (2005) pp. 1-287. (Cited on page 137.)

K. L. Bernstein. A congruence theorem for structured operational semantics of higher-order languages. In *Proceedings of the Thirteenth Annual IEEE Symposium on Logic in Computer Science (LICS'98)*, pages 153–164. IEEE Computer Society Press, 1998. (Cited on pages 228 and 229.)

R. Blackwell, G. M. Kelly, and A. J. Power. Two-dimensional monad theory. *J. Pure Appl. Algebra*, 59:1–41, 1989. (Cited on page 136.)

B. Bloom, S. Istrail, and A. R. Meyer. Bisimulation can't be traced. *J. ACM*, 42(1):232–268, 1995. (Cited on pages 15, 21, 133, 151, 153, 197, 203, and 206.)

B. Bloom and F. Vaandrager. SOS rule formats for parameterized and state-bearing processes. Draft, 1994. (Cited on page 228.)

M. Boreale and R. D. Nicola. A symbolic semantics for the π-calculus. *Inform. and Comput.*, 126: 34–52, 1996. (Cited on page 13.)

R. Bruni, F. Honsell, M. Lenisa, and M. Miculan. Modeling fresh names in the π-calculus using abstractions. In *Proceedings of the Seventh International Workshop on Coalgebraic Methods in Computer Science (CMCS'04)*, volume 106 of *Electron. Notes Theor. Comput. Sci.*, pages 25–41, 2004. (Cited on page 14.)

A. Burroni. *T*-catégories: catégories dans un triple. *Cah. Topol. Géom. Différ. Catég.*, XII(3):215–321, 1971. (Cited on page 121.)

M. G. Buscemi and U. Montanari. A first order coalgebraic model of π-calculusearly observational equivalence. In *Proceedings of the Thirteenth International Conference on Concurrency Theory (CONCUR'02)*, volume 2421 of *Lecture Notes in Comput. Sci.*, pages 449–465. Springer-Verlag, 2002. (Cited on page 229.)

A. Carboni and E. M. Vitale. Regular and exact completions. *J. Pure Appl. Algebra*, 125:79–116, 1998. (Cited on page 113.)

G. L. Cattani and P. Sewell. Models for name-passing processes: interleaving and causal. *Inform. and Comput.*, 190:136–178, 2004. (Cited on pages 13, 14, 18, 45, 46, 60, 61, 62, 72, 73, 74, 76, 79, 80, 81, and 227.)

G. L. Cattani, I. Stark, and G. Winskel. Presheaf models for the pi-calculus. In *Proceedings of the Sixth Conference on Category Theory and Computer Science (CTCS'97)*, volume 1290 of *Lecture Notes in Comput. Sci.*, pages 106–126, 1997. (Cited on page 229.)

E. M. Clarke, O. Grumberg, and D. A. Peled. *Model Checking*. MIT Press, 2000. (Cited on page 111.)

P. M. Cohn. *Basic Algebra: Groups, Rings and Fields*. Springer, 2003. (Cited on page 111.)

P. Crawley and R. P. Dilworth. *Algebraic Theory of Lattices*. Prentice-Hall, 1973. (Cited on page 230.)

N. G. de Bruijn. Lambda-calculus notation with nameless dummies, a tool for automatic formula manipulation, with application to the Church-Rosser theorem. *Indagationes Math.*, 34:381–392, 1972. (Cited on page 17.)

R. de Simone. Higher-level synchronising devices in Meije-SCCS. *Theoret. Comput. Sci.*, 37:245–267, 1985. (Cited on page 15.)

U. H. Engberg and M. Nielsen. A calculus of communicating systems with label passing – ten years after. In G. D. Plotkin, C. Stirling, and M. Tofte, editors, *Proof, Language, and Interaction: Essays in Honour of Robin Milner*, pages 599–622. MIT Press, 2000. (Cited on page 11.)

G. L. Ferrari, S. Gnesi, U. Montanari, and M. Pistore. A model-checking verification environment for mobile processes. *ACM Trans. Softw. Eng. Methodol.*, 12(4):440–473, 2003. (Cited on page 111.)

G. L. Ferrari, U. Montanari, and M. Pistore. Minimizing transition systems for name passing calculi: A co-algebraic formulation. In *Proceedings of the Fifth International Conference on Foundations of Software Science and Computation Structures (FOSSACS'02)*, volume 2303 of *Lecture Notes in Comput. Sci.*, pages 129–158. Springer-Verlag, 2002. (Cited on pages 19, 111, 113, 114, 118, 119, and 227.)

M. P. Fiore. A coinduction principle for recursive data types based on bisimulation. *Inform. and Comput.*, 127:186–198, 1996. (Cited on page 232.)

M. P. Fiore. Notes on combinatorial functors. Available from the author's home page, January 2001. (Cited on pages 19, 23, 101, 103, and 118.)

M. P. Fiore and M. Menni. Reflective Kleisli subcategories of the category of Eilenberg-Moore algebras for factorization monads. *Theory Appl. of Categ.*, 15(2):40–65, 2005. (Cited on page 103.)

M. P. Fiore, E. Moggi, and D. Sangiorgi. A fully abstract model for the π-calculus. *Inform. and Comput.*, 179(1):76–117, 2002. (Cited on pages 13 and 14.)

M. P. Fiore, G. D. Plotkin, and D. Turi. Abstract syntax and variable binding (extended abstract). In *Proceedings of the Fourteenth Annual IEEE Symposium on Logic in Computer Science (LICS'99)*, pages 193–202. IEEE Computer Society Press, 1999. (Cited on pages 17, 176, and 235.)

M. P. Fiore and S. Staton. Comparing operational models of name-passing process calculi. *Inform. and Comput.*, 204(4):435–678, 2006a. Extended abstract appeared in *Proceedings of the Seventh International Workshop on Coalgebraic Methods in Computer Science (CMCS'04)*. (Cited on page 24.)

M. P. Fiore and S. Staton. A congruence rule format for name-passing process calculi from mathematical structural operational semantics. In *Proceedings of the Twenty-First Annual IEEE Symposium on Logic in Computer Science (LICS'06)*, pages 49–58. IEEE Computer Society Press, 2006b. (Cited on page 24.)

M. P. Fiore and D. Turi. Semantics of name and value passing (extended abstract). In *Proceedings of the Sixteenth Annual IEEE Symposium on Logic in Computer Science (LICS'01)*, pages 93–104. IEEE Computer Society Press, 2001. (Cited on pages 5, 13, 14, 18, 33, 35, 45, 53, 55, 57, 58, 66, 73, 75, 76, 227, and 233.)

W. J. Fokkink and C. Verhoef. A conservative look at operational semantics with variable binding. *Inform. and Comput.*, 146(1):24–54, 1998. (Cited on page 17.)

P. Freyd. Algebraically complete categories. In A. Carboni, M. C. Pedicchio, and G. Rosolini, editors, *Proceedings of Category Theory, Como '90*, volume 1488 of *Lecture Notes in Comput. Sci.*, pages 95–104. Springer-Verlag, 1991. (Cited on page 14.)

P. Freyd. Remarks on algebraically compact categories. In M. P. Fourman, P. T. Johnstone, and A. M. Pitts, editors, *Applications of Categories in Computer Science: Proceedings of the LMS Symposium, Durham 1991*, volume 177 of *London Math. Soc. Lecture Note Ser.*, pages 95–106. Cambridge University Press, 1992. (Cited on page 14.)

M. J. Gabbay. The π-calculus in FM. In F. D. Kamareddine, editor, *Thirty Five Years of Automating Mathematics*, volume 28 of *Applied Logic Series*. Kluwer, 2003. (Cited on page 230.)

M. J. Gabbay. *A Theory of Inductive Definitions with Alpha-Equivalence*. PhD thesis, University of Cambridge, 2001. (Cited on page 168.)

M. J. Gabbay and A. Mathijssen. Capture-avoiding substitution as a nominal algebra. In *Proceedings of Theoretical Aspects of Computing – ICTAC 2006, Third International Colloquium*, volume 4281 of *Lecture Notes in Comput. Sci.*, pages 198–212. Springer, 2006. (Cited on page 17.)

M. J. Gabbay and A. M. Pitts. A new approach to abstract syntax with variable binding. *Formal Aspects of Computing*, 13:341–363, 2001. (Cited on pages 17, 19, 24, 167, 168, 171, 227, and 230.)

F. Gadducci, M. Miculan, and U. Montanari. About permutation algebras, (pre)sheaves and named sets. *Higher-Order Symb. Comput.*, 19(2–3):283–304, 2006. (Cited on page 121.)

N. Ghani, K. Yemane, and B. Victor. Relationally staged computations in calculi ofmobile processes. In *Proceedings of the Seventh International Workshop on Coalgebraic Methods in Computer Science (CMCS'04)*, volume 106 of *Electron. Notes Theor. Comput. Sci.*, pages 105–120. Elsevier, 2004. (Cited on page 233.)

J. A. Goguen, J. W. Thatcher, and E. G. Wagner. An initial algebra approach to the specification, correctness and implementation of abstract data types. In R. T. Yeh, editor, *Current Trends in Programming Methodology*, volume IV, chapter 5, pages 80–149. Prentice-Hall, 1978. (Cited on pages 15 and 17.)

A. D. Gordon. Bisimilarity as a theory of functional programming. *Theoret. Comput. Sci.*, 228:5–47, 1999. (Cited on page 12.)

J. F. Groote and F. Vaandrager. Structured operational semantics and bisimulation as a congruence. *Inform. and Comput.*, 100(2):202–260, 1992. (Cited on pages 153, 228, and 229.)

I. Hasuo, B. Jacobs, and A. Sokolova. Generic trace theory. In *Proceedings of the Eighth International Workshop on Coalgebraic Methods in Computer Science (CMCS'06)*, volume 164 of *Electron. Notes Theor. Comput. Sci.*, pages 47–65, 2006. (Cited on page 60.)

M. Hennessy and H. Lin. Symbolic bisimulations. *Theoret. Comput. Sci.*, 138(2):353–389, 1995. (Cited on page 13.)

F. Honsell, M. Lenisa, U. Montanari, and M. Pistore. Final semantics for the pi-calculus. In *Proceedings of the International Conference on Programming Concepts and Methods (PROCOMET'98)*, volume 125 of *IFIP Conference Proceedings*, pages 225–243. Chapman & Hall, 1998. (Cited on page 229.)

A. Jeffrey and J. Rathke. Contextual equivalence for higher-order pi-calculus revisited. *Logical Methods in Computer Science*, 1(1), 2005. (Cited on page 12.)

P. T. Johnstone. A topos theorist looks at dilators. *J. Pure Appl. Algebra*, 58:235–249, 1989. (Cited on page 94.)

P. T. Johnstone. *Sketches of an Elephant: A Topos Theory Compendium*. Number 43, 44 in Oxford Logic Guides. Oxford University Press, 2002. (Cited on pages 89, 90, 91, 93, 94, 98, 124, 230, and 234.)

G. M. Kelly. Doctrinal adjunction. In *Proceedings, Sydney Category Theory Seminar*, volume 420 of *Lecture Notes in Math.*, pages 257–280. Springer, 1972. (Cited on pages 34 and 136.)

G. M. Kelly. A unified treatment of transfinite constructions for free algebras, free monoids, colimits, associated sheaves, and so on. *Bull. Austral. Math. Soc.*, 22:1–83, 1980. (Cited on page 137.)

B. Klin. Adding recursive constructs to bialgebraic semantics. *J. Log. Algebr. Program.*, 60–61: 259–286, 2004. (Cited on page 235.)

A. Kock. Monads for which structures are adjoint to units. *J. Pure Appl. Algebra*, 104:41–59, 1995. (Cited on page 112.)

J. Leifer and R. Milner. Deriving bisimulation congruences for reactive systems. In *Proceedings of the Eleventh International Conference on Concurrency Theory (CONCUR'00)*, volume 1877 of *Lecture Notes in Comput. Sci.*. Springer-Verlag, 2000. (Cited on page 12.)

M. Lenisa, J. Power, and H. Watanabe. Distributivity for endofunctors, pointed and co-pointed endofunctors, monads and comonads. In H. Reichel, editor, *Proceedings of the Third International Workshop on Coalgebraic Methods in Computer Science (CMCS'00)*, volume 33 of *Electron. Notes Theor. Comput. Sci.*, pages 230–260. Elsevier, 2000. (Cited on page 34.)

H. Lin. Complete inference systems for weak bisimulation equivalences in the π-calculus. *Inform. and Comput.*, 180(1):1–29, 2003. (Cited on page 13.)

S. Mac Lane. *Categories for the working mathematician*. Graduate Texts in Mathematics. Springer, second edition, 1998. (Cited on pages 31 and 136.)

S. Mac Lane and I. Moerdijk. *Sheaves in Geometry and Logic: a First Introduction to Topos Theory*. Universitext. Springer-Verlag, 1992. (Cited on pages 89, 94, and 95.)

M. Miculan and K. Yemane. A unifying model of variables and names. In *Proceedings of the Eighth International Conference on Foundations of Software Science and Computation Structures (FOSSACS'05)*, volume 3441 of *Lecture Notes in Comput. Sci.*, pages 170–186, 2005. (Cited on page 231.)

C. A. Middelburg. Variable binding operators in transition system specifications. *J. Log. Algebr. Program.*, 47:15–45, 2001. (Cited on pages 17 and 235.)

D. Miller. A proof theoretic approach to operational semantics. In L. Aceto and A. D. Gordon, editors, *Proceedings of Algebraic Process Calculi: The First Twenty Five Years and Beyond*, volume 162 of *Electron. Notes Theor. Comput. Sci.*, pages 243–247, 2006. (Cited on page 13.)

D. Miller and A. Tiu. A proof theory for generic judgments. *ACM Trans. Comput. Log.*, 6(4):749–783, 2005. (Cited on pages 229 and 231.)

R. Milner. *Communication and Concurrency*. Prentice Hall, 1989. (Cited on pages 11, 124, 125, 134, and 234.)

R. Milner. *A calculus of communicating systems*, volume 92 of *Lecture Notes in Comput. Sci.*. Springer-Verlag, 1980. (Cited on pages 11, 14, 15, 34, and 36.)

R. Milner. *Communicating and Mobile Systems : The π-Calculus*. Cambridge University Press, 1999. (Cited on page 11.)

R. Milner, J. Parrow, and D. Walker. A calculus of mobile processes, I and II. *Inform. and Comput.*, 100(1):1–77, 1992. (Cited on pages 11, 13, 21, 46, 47, 49, 50, and 53.)

R. Milner, J. Parrow, and D. Walker. Modal logics for mobile processes. *Theoret. Comput. Sci.*, 114 (1):149–171, 1993. (Cited on page 47.)

I. Moerdijk. *Classifying Spaces and Classifying Topoi*. Number 1616 in Lecture Notes in Math.. Springer-Verlag, 1995. (Cited on page 170.)

U. Montanari and M. Pistore. An introduction to history dependent automata. In *Proceedings of HOOTS II: Second Workshop on Higher-Order Operational Techniques in Semantics*, volume 10 of *Electron. Notes Theor. Comput. Sci.*, pages 170–188, 1997. (Cited on pages 14, 111, 112, and 113.)

U. Montanari and M. Pistore. Structured coalgebras and minimal HD-automata for the π-calculus. *Theoret. Comput. Sci.*, 340(4):539–576, 2005. (Cited on page 229.)

U. Montanari and V. Sassone. Dynamic congruence vs. progressing bisimulation for CCS. *Fund. Inform.*, 16(1):171–199, 1992. (Cited on page 13.)

R. M. Needham. Names. In S. Mullender, editor, *Distributed Systems*, chapter 5, pages 89–101. ACM Press, 1989. (Cited on page 11.)

R. Paige and R. E. Tarjan. Three partition refinement algorithms. *SIAM J. on Comput.*, 16(6): 973–989, 1987. (Cited on pages 12 and 23.)

J. Parrow. An introduction to the pi-calculus. In Bergstra, Ponse, and Smolka, editors, *Handbook of Process Algebra*, pages 479–543. Elsevier, 2001. (Cited on pages 11 and 53.)

J. Parrow and B. Victor. The update calculus (extended abstract). In *Proceedings of the Sixth International Conference on Algebraic Methodology and Software Technology (AMAST'97)*, volume 1349 of *Lecture Notes in Comput. Sci.*, pages 409–423. Springer, 1997. (Cited on page 13.)

M. Pistore. *History Dependent Automata*. PhD thesis, University of Pisa, Dipartimento di Informatica, 1999. (Cited on pages 113 and 114.)

M. Pistore and D. Sangiorgi. A partition refinement algorithm for the π-calculus. *Inform. and Comput.*, 164:264–321, 2001. (Cited on page 111.)

A. M. Pitts. A co-induction principle for recursively defined domains. *Theoret. Comput. Sci.*, 124: 195–219, 1994. (Cited on page 14.)

A. M. Pitts. Nominal logic, a first order theory of names and binding. *Inform. and Comput.*, 186: 165–193, 2003. (Cited on pages 21, 174, and 175.)

A. M. Pitts. Alpha-structural recursion and induction. *J. ACM*, 53(3):459–506, 2006. (Cited on pages 168 and 169.)

A. M. Pitts and I. Stark. On the observable properties of higher order functions that dynamically create local names (preliminary report). In *Proceedings of the ACM SIGPLAN Workshop on State in Programming Languages (SIPL'93)*, pages 31–45, 1993. Technical Report YALE/DCS/TR968, Yale University Department of Computer Science. (Cited on page 18.)

G. D. Plotkin. Bialgebraic semantics and recursion (extended abstract). In *Proceedings of the Fourth International Workshop on Coalgebraic Methods in Computer Science (CMCS'01)*, volume 44(1) of *Electron. Notes Theor. Comput. Sci.*, pages 1–4, 2001. (Cited on page 235.)

G. D. Plotkin. A structural approach to operational semantics. Technical Report DAIMI-FN-19, Computer Science Dept., Aarhus University, Denmark, 1981. Republished in J. Log. Algebr. Program., 60–61:17–140. (Cited on pages 13 and 15.)

G. D. Plotkin. The origins of structural operational semantics. *J. Log. Algebr. Program.*, 60–61:3–15, 2004. (Cited on page 16.)

J. Power and D. Turi. A coalgebraic foundation for linear time semantics. In *Proceedings of the Eighth Conference on Category Theory and Computer Science (CTCS'99)*, volume 29 of *Electron. Notes Theor. Comput. Sci.*, pages 259–274. Elsevier, 1999. (Cited on page 60.)

J. J. M. M. Rutten. Universal coalgebra: a theory of systems. *Theoret. Comput. Sci.*, 249(1):3–80, 2000. (Cited on pages 14, 22, and 31.)

D. Sangiorgi. On the bisimulation proof method. *Math. Structures Comput. Sci.*, 8(5):447–479, 1998. (Cited on page 12.)

D. Sangiorgi. A theory of bisimulation for the pi-calculus. *Acta Inform.*, 33(1):69–97, 1996. (Cited on pages 13, 37, 52, 53, 228, 229, and 233.)

D. Sangiorgi. *Expressing Mobility in Process Algebras: First-Order and Higher-Order Paradigms*. PhD thesis, University of Edinburgh, 1992. (Cited on page 235.)

D. Sangiorgi and D. Walker. *The π-calculus: a theory of mobile processes*. Cambridge University Press, 2001. (Cited on pages 11, 12, 13, 47, 48, 49, 50, 51, 52, 53, 182, 201, and 237.)

U. Schöpp. Modelling generic judgements. In *Proceedings of the International Workshop on Logical Frameworks and Meta-Languages: Theory and Practice (LFMTP'06)*, Seattle, 2006. (Cited on page 231.)

P. Sewell. π-calculi. In H. Bowman and J. Derrick, editors, *Formal Methods for Distributed Processing: A Survey of Object-Oriented Approaches*, chapter 9, pages 177–197. Cambridge University Press, 2001. (Cited on page 14.)

P. Sewell. From rewrite rules to bisimulation congruences. *Theoret. Comput. Sci.*, 274:183–230, 2002. (Cited on page 12.)

A. K. Simpson. Compositionality via cut-elimination: Hennessy-Milner logic for an arbitrary GSOS. In *Proceedings of the Tenth Annual IEEE Symposium on Logic in Computer Science (LICS'95)*, pages 420–430. IEEE Computer Society Press, 1995. (Cited on page 153.)

I. Stark. A fully abstract domain model for the π-calculus. In *Proceedings of the Eleventh Annual IEEE Symposium on Logic in Computer Science (LICS'96)*, pages 36–42. IEEE Computer Society Press, 1996. (Cited on pages 13 and 14.)

R. Street. The formal theory of monads. *J. Pure Appl. Algebra*, 2:149–168, 1972. (Cited on pages 34 and 135.)

P. Taylor. *Practical Foundations of Mathematics*. Number 59 in Cambridge Studies in Advanced Mathematics. Cambridge University Press, 1999. (Cited on page 142.)

B. Thomsen. A theory of higher order communicating systems. *Inform. and Comput.*, 116:38–57, 1995. (Cited on page 235.)

D. Turi. Categorical modelling of structural operational rules: Case studies. In *Proceedings of the Sixth Conference on Category Theory and Computer Science (CTCS'97)*, volume 1290 of *Lecture Notes in Comput. Sci.*, pages 127–146. Springer, 1997. (Cited on page 235.)

D. Turi. *Functorial Operational Semantics and its Denotational Dual*. PhD thesis, Free University, Amsterdam, June 1996. (Cited on page 31.)

D. Turi and G. D. Plotkin. Towards a mathematical operational semantics. In *Proceedings of the Twelfth Annual IEEE Symposium on Logic in Computer Science (LICS'97)*, pages 280–291. IEEE Computer Society Press, 1997. (Cited on pages 15, 20, 24, 35, 36, 133, 138, 141, 149, 153, 228, and 232.)

C. Urban, A. M. Pitts, and M. J. Gabbay. Nominal unification. *Theoret. Comput. Sci.*, 323:473–497, 2004. (Cited on page 199.)

S. M. Weber. *Process Algebras and Meta-Algebras: Theory and Practice*. PhD thesis, Cornell University, August 1995. (Cited on page 228.)

S. M. Weber and B. Bloom. Metatheory of the π-calculus. Technical Report TR96-1564, Cornell University, Computer Science, 1996. (Cited on pages 228 and 231.)

J. Worrell. On the final sequence of a finitary set functor. *Theoret. Comput. Sci.*, 338:184–199, 2005. (Cited on pages 23 and 127.)

A. Ziegler, D. Miller, and C. Palamidessi. A congruence format for name-passing calculi. In *Proceedings of the Second Workshop on Structural Operational Semantics (SOS'05)*, volume 156 of *Electron. Notes Theor. Comput. Sci.*, pages 169–189, 2006. (Cited on pages 228, 229, and 231.)