Relational Properties of Domains*

Andrew M. Pitts†
Cambridge University Computer Laboratory
Pembroke Street
Cambridge CB2 3QG, England

23 January 1995

Abstract

New tools are presented for reasoning about properties of recursively defined domains. We work within a general, category-theoretic framework for various notions of 'relation' on domains and for actions of domain constructors on relations. Freyd's analysis of recursive types in terms of a property of mixed initiality/finality is transferred to a corresponding property of *invariant* relations. The existence of invariant relations is proved under completeness assumptions about the notion of relation. We show how this leads to simpler proofs of the computational adequacy of denotational semantics for functional programming languages with user-declared datatypes. We show how the initiality/finality property of invariant relations can be specialized to yield an induction principle for admissible subsets of recursively defined domains, generalizing the principle of structural induction for inductively defined sets. We also show how the initiality/finality property gives rise to the co-induction principle studied by the author (*Theoretical Computer Science* **124**, 195–219 (1994)), by which equalities between elements of recursively defined domains may be proved via an appropriate notion of 'bisimulation'.

*To appear in *Information & Computation*. Revised version of Cambridge Univ. Computer Laboratory Tech. Report No. 321.

†Research supported by UK SERC grant GR/G53279 and EC ESPRIT Basic Research project CLICS-II.

1 Introduction

A characteristic feature of higher-order functional languages such as Standard ML (Milner, Tofte and Harper, 1990) or Haskell (Hudak, Peyton Jones, and Wadler, 1991) is the facilities they provide for 'user-declared' recursive datatypes. However, this powerful feature comes at a price: because few syntactic restrictions are placed upon the form of a datatype declaration, in general it can be hard to understand what the values of such a datatype denote, and correspondingly hard to reason about the observable behaviour under evaluation of expressions involving such datatypes. One way round this problem is to consider only restricted forms of datatype declaration, for example just those giving rise to inductively defined sets of finitely constructible data values (lists, trees, etc.), for which there are well understood reasoning techniques such as the principle of structural induction. This is too restrictive since it cuts out 'lazy' datatypes (containing partial and potentially infinite data values), which are an important functional programming tool and unavoidable by design in non-strict languages such as Haskell. Whilst properties of lazy datatypes have been considered on a case-by-case basis (see (Thompson, 1989), for example), there are remarkably few reasoning principles in the literature that apply uniformly to all recursive datatypes including ones whose definitions contain negative occurrences of the declared type. This paper attempts to improve this situation by studying properties of (abstract) relations on the recursively defined domains that arise in the denotational semantics of such datatypes.

The key idea which is explored in this paper, and which goes back at least to (Milne, 1973; Plotkin, 1973; Reynolds 1974), is to take account of the fact that various domain constructors can be extended to corresponding constructors for relations on domains. Traditionally one considers only certain kinds of relation on domains—ones that are sufficiently complete, or 'admissible'. This is appropriate for various inductive properties of domains; but as (Pitts, 1992) shows, one should go beyond this to arbitrary set-theoretic relations in order to capture fully the co-inductive properties of recursively defined domains. In fact we need remarkably few properties of a general notion of 'relation' in order to establish our main results. These properties can be conveniently axiomatized using a framework adapted from the work of O'Hearn and Tennent (1993) on applying Reynolds' notion of relational parametricity to the semantics of local-variable declarations. Accordingly, results in this paper about relational properties of recursively defined domains are parameterized by the particular notion of 'relation' being used and by the particular way domain constructors act on these relations.

A preliminary version of the results in this paper appeared in the extended abstract (Pitts, 1993). I would like to thank one of the referees, whose comments helped to improve the exposition of this paper.

2 Overview

The purpose of this section is not just to provide the busy reader with some understanding of what is in this paper without having to read the rest of it. As well as summarising the main results and applications, it attempts to highlight

the key ideas and techniques used, which might otherwise get obscured by the detailed development of the subsequent sections. Throughout this section let

$$\alpha = \Phi(\alpha) \tag{1}$$

be a fixed domain equation. Here Φ is some domain constructor involving some of the standard constructions on domains that are used in the denotational semantics of deterministic programming languages (such as various kinds of product, sum and function space, lifting, etc.) and which are reviewed at the beginning of Sect. 3.

Techniques

Minimal invariant property of $rec\alpha.\Phi(\alpha)$. That a domain equation (1) has a solution at all, i.e. that there is a domain D isomorphic to $\Phi(D)$, was D. Scott's remarkable discovery of the late 1960's which initiated domain theory as a subject. However, the fact is that in general there may be many non-isomorphic solutions to (1) and usually one is interested in a particular solution that is *minimal* in a suitable sense and which we will denote by $rec\alpha.\Phi(\alpha)$. Although there are several ways to express this minimality property, we shall see in this paper that an extremely simple, but nonetheless convenient way is in terms of the so-called *minimal invariant* property of $rec\alpha.\Phi(\alpha)$. This states that the identity function on $D \stackrel{def}{=} rec\alpha.\Phi(\alpha)$ is the least fixed point of the continuous operator on the domain of strict continuous functions, $D \multimap D$, mapping e to

$$\delta_\Phi(e) \stackrel{def}{=} \left(D \cong \Phi(D) \stackrel{\Phi(e)}{\longrightarrow} \Phi(D) \cong D \right)$$

where $e \mapsto \Phi(e)$ is a (continuous) action of the domain constructor on strict continuous endofunctions that can be defined according to the structure of Φ. Of course the least fixed point of δ_Φ can be calculated as the least upper bound of $\perp, \delta_\Phi(\perp), \delta_\Phi^2(\perp), \ldots$. So the minimal invariant property amounts to the fact that there is a uniform expression for elements $d \in D$ as least upper bounds of chains of 'projected' elements

$$d = \bigsqcup_{n<\omega} \pi_n(d), \qquad \text{where } \begin{cases} \pi_0(d) \stackrel{def}{=} \perp \\ \pi_{n+1}(d) \stackrel{def}{=} \delta_\Phi(\pi_n)(d) \end{cases} .$$

See Sect. 3 for more details. Of course the minimal invariant property has to be proved by examining the detailed construction of $rec\alpha.\Phi(\alpha)$ (and was known since D. Scott's first work on solving domain equations): see Theorem 3.3. But once established, a remarkable number of properties of recursively defined domains follow from it in an elementary fashion. The relational properties presented in this paper are an illustration of this, since they derive directly from the minimal invariant property. In Example 3.6 we give another illustration—a pleasingly simple proof, due to Plotkin (private communication), that the Curry fixed point combinator coincides with the least fixed point operator in the canonical domain model of the lazy lambda calculus studied in (Abramsky, 1990) and (Abramsky and Ong, 1993).

Separation of positive and negative occurrences of variables. This is a key aspect of Freyd's recent work on recursive types via his notion of 'algebraic compactness' (Freyd, 1991, 1992). Let $\hat{\Phi}(\alpha^-, \alpha^+)$ denote the result of replacing all positive occurrences of α in Φ by a new variable α^+, and replacing all negative occurrences by a different variable α^-. An occurrence of α in Φ is positive (respectively, negative) if it is hereditarily to the left of an even (respectively, odd) number of function space constructors. For example, if $\Phi(\alpha) = (\alpha \to \alpha)_\perp$, then $\hat{\Phi}(\alpha^-, \alpha^+) = (\alpha^- \to \alpha^+)_\perp$. The original domain constructor can be recovered by diagonalization: $\Phi(\alpha) = \hat{\Phi}(\alpha, \alpha)$. However, $\hat{\Phi}(\alpha^-, \alpha^+)$ has better functoriality properties than $\Phi(\alpha)$. For example, the action of Φ on strict continuous endofunctions used in the definition of the minimal invariant property can now be seen as the diagonalization of a continuous, binary action mapping $(e^-, e^+) \in (E^- \multimap D^-) \times (D^+ \multimap E^+)$ to $\hat{\Phi}(e^-, e^+) \in (\hat{\Phi}(D^-, D^+) \multimap \hat{\Phi}(E^-, E^+))$, which preserves identity functions and composition (contravariantly in the left-hand argument and covariantly in the right-hand one). This elaboration of domain constructors as functors of mixed variance enables Freyd to establish a very powerful mixed initial-algebra/final-coalgebra property of $rec\,\alpha.\Phi(\alpha)$ (Theorem 3.4), which amongst other things shows that a minimal invariant for $\Phi(\alpha)$ is unique up to isomorphism. In this paper not only do we make use of Freyd's initial-algebra/final-coalgebra property of $rec\,\alpha.\Phi(\alpha)$, but also we apply the separation of variables trick at the level of actions of domain constructors on relations.

Action of domain constructions on relations. As mentioned in the Introduction, the key technique used in this paper is to take account of the fact that for various notions of relation, the various domain constructors can be extended to act on relations on domains. This observation is developed at length in Sect. 4 and applied in Sects 5 and 6. Here we wish to emphasise two general points.

First, by taking a fairly abstract view of what constitutes a 'relation' we can deduce a variety of results from a few theorems about invariant relations in general. Thus for the induction principle for $rec\,\alpha.\Phi(\alpha)$ (Theorem 6.5) we need to consider admissible (i.e. chain-complete and bottom-containing) subsets of a domain D as relations; for the co-induction principle (Theorem 6.12) we need to consider arbitrary subsets of $D \times D$ as relations on D; and for the computational adequacy result of Sect. 5 we need to consider a notion of relation on D that mixes syntax and semantics.

Secondly, even for a fixed notion of relation it may be that a given domain constructor can usefully be considered to have more than one kind of action on relations. For example, consider the continuous function space construction $\hat{\Phi}(\alpha^-, \alpha^+) = (\alpha^- \to \alpha^+)$. For fixed n, taking a relation on a domain D to mean a subset of the n-fold product D^n, it is natural to consider an action of $\hat{\Phi}$ on relations defined by:

$$(R^-, R^+) \mapsto \{(f_1, \ldots, f_n) \in (D^- \to D^+)^n \mid \\ \forall (d_1, \ldots, d_n) \in R^-.(f_1(d_1), \ldots, f_n(d_n)) \in R^+\} \ .$$

This action is characteristic of many uses of 'logical relations' and (in case $n = 2$) is the one used in our general co-induction principle. However, for the induction

principle we consider the $n = 1$ case of a simpler action that throws away R^-:

$$(R^-, R^+) \mapsto \{(f_1, \ldots, f_n) \in (D^- \to D^+)^n \mid \\ \forall (d_1, \ldots, d_n) \in (D^-)^n . (f_1(d_1), \ldots, f_n(d_n)) \in R^+\} \ .$$

Results

Invariant relations. The main technical results of the paper concern the existence (Theorem 4.16) and properties (Corollary 4.10) of certain recursively specified relations on recursively defined domains. For the purposes of this Overview, it will simplify matters slightly if we suppose that $D = rec\alpha.\Phi(\alpha)$ is a (minimal) solution of (1) up to equality rather than just up to isomorphism, i.e. that $D = \Phi(D)$. (This is always possible, using D. Scott's 'information systems'—see Winskel and Larsen (1984).) For various notions of relation, the construction $D \mapsto \Phi(D)$ on domains extends to a corresponding action on relations, sending a relation R on a domain D to a relation $\Phi(R)$ on the domain $\Phi(D)$. When $D \stackrel{def}{=} rec\alpha.\Phi(\alpha) = \Phi(rec\alpha.\Phi(\alpha))$, it makes sense to ask whether there is an *invariant* relation Δ on $rec\alpha.\Phi(\alpha)$, i.e. one satisfying $\Delta = \Phi(\Delta)$. For particular choices of the notion of relation, these invariant relations lie at the heart of some proofs of correspondence between denotational and operational semantics (as in (Plotkin, 1985), or (Meyer and Cosmodakis,1988, Appendix), for example), or between two denotational semantics (as in (Reynolds, 1974) for example). The existence of such an invariant relation Δ for (1) is not straightforward in the case that $\Phi(\alpha)$ contains negative occurrences of α. For then $R \mapsto \Phi(R)$ is not a monotone operator on relations; so even though the collection of relations on $rec\alpha.\Phi(\alpha)$ may form a complete lattice, we cannot simply appeal to the Tarski-Knaster fixed point theorem to construct Δ.

The various constructions of particular invariant relations that occur in the literature rely upon the quite heavy technical machinery used to establish the existence of solutions to (1) in the first place. By contrast, we get by here with quite elementary tools: the minimal invariant property of $rec\alpha.\Phi(\alpha)$ mentioned above, the Tarski-Knaster fixed point theorem, and D. Scott's principle of induction for least fixed points of continuous endofunctions of a domain. To be in a position to apply Scott induction, we first have to develop a suitable notion of *admissibility* within our general framework for relations, generalising the usual definition of admissible subset of a domain as a chain-closed subset containing the least element. Typically, a particular notion of relation will have the properties that the admissible relations on a domain D form a complete lattice $\mathcal{R}_{adm}(D)$ (for a suitable notion of inclusion between relations) and that the action $R \mapsto \Phi(R)$ of Φ preserves admissibility. More precisely, $\hat{\Phi}(R^-, R^+)$ should be admissible whenever R^+ is, where $\hat{\Phi}(\alpha^-, \alpha^+)$ is the constructor obtained from Φ by separating positive and negative occurrences of α. Although $R \mapsto \Phi(R)$ need not be monotone, by construction $(R^-, R^+) \mapsto \hat{\Phi}(R^-, R^+)$ is order-reversing in its left-hand argument and order-preserving in its left-hand one. Thus

$$(R^-, R^+) \mapsto (\hat{\Phi}(R^+, R^-), \hat{\Phi}(R^-, R^+)) \tag{2}$$

is a monotone operator on the complete lattice $\mathcal{R}_{adm}(D)^{\mathrm{op}} \times \mathcal{R}_{adm}(D)$ and hence has a least (pre)fixed point, (Δ^-, Δ^+) say. It then suffices to prove that

$\Delta^- = \Delta^+$, for then this relation will be the required invariant relation for the the diagonalization, Φ, of $\hat{\Phi}$. That Δ^+ is contained in Δ^- follows immediately from the symmetry properties of the operator (2). For the reverse containment, we exploit the minimal invariant property of D. One checks that if $e \in (D \multimap D)$ maps Δ^- into Δ^+, then so does $\delta_\Phi(e)$. The admissibility of Δ^+ means that the collection of such maps e is a chain-closed subset of $D \multimap D$ containing bottom, and hence by Scott induction this collection also contains the least fixed point of δ_Φ, which is id_D. Thus the identity function maps Δ^- into Δ^+, which is just to say that Δ^- is contained in Δ^+, as required. This, in outline, is the proof of the existence theorem (4.16) for invariant relations.

Computational adequacy. Sect. 5 illustrates the application of invariant relations to proofs of correspondence between operation and denotational semantics of a programming language. We consider a simple, but non-trivial fragment of Standard ML (Milner, Tofte and Harper, 1990) containing a datatype declaration

$$\texttt{datatype} \quad ty = \texttt{In}_1 \texttt{ of } \sigma_1 \mid \cdots \mid \texttt{In}_n \texttt{ of } \sigma_n \tag{3}$$

where the types $\sigma_1, \ldots, \sigma_n$ are built up from basic types (booleans, integers, characters, etc.) and the type variable ty, using any combination of the product ($*$) and function space ($\rightarrow$) constructors. Using standard techniques of denotational semantics, each type σ can be modelled by a domain $[\![\sigma]\!]$, with $[\![ty]\!] = rec\,\alpha.\Phi(\alpha)$ for a suitable choice of Φ; and then each (closed) expression $\texttt{e}$ of the language can be assigned a denotation as an element $[\![\texttt{e}]\!]$ of the domain $[\![\sigma]\!]$, where σ is the type of $\texttt{e}$. The denotational semantics induces a notion of equivalence between expression (of equal type), via equality $[\![\texttt{e}]\!] = [\![\texttt{e}']\!]$ of denotations.

Our example language, being equivalent to a fragment of Standard ML, comes equipped with an operational semantics via the definition in (Milner, Tofte and Harper, 1990). This determines a notion of *contextual equivalence* between language expressions (of equal type): two expressions are contextually equivalent if occurrences of one can be interchanged with occurrences of the other in any expression without affecting the observable results of expression evaluation. A denotational semantics of the language is *computationally adequate* if the equality of the denotations of any two expressions implies their contextual equivalence. Thus a computationally adequate denotational semantics provides a sound method for establishing instances of contextual equivalence. It is notoriously difficult to devise denotational semantics for which equality of denotations actually coincides with contextual equivalence—this is the so-called 'full abstraction' problem for the language. However, it is well-known that the standard, domain-theoretic semantics of the language is indeed computationally adequate. The proof of this reduces to showing that any closed expression $\texttt{e}$, of type σ say, evaluates to a value (rather than diverging) if its denotation $[\![\texttt{e}]\!]$ is not equal to the least element of the domain $[\![\sigma]\!]$. Following Plotkin (1985), this property can be deduced from the existence of a certain 'logical' relation between language expressions and domain elements. The presence in the language of recursive types ty imposes requirements on the logical relation that make proving its existence a non-trivial task. We show in Sect. 5 that for a suitable choice of notion of relation and of action of domain constructors on

relations, these requirements are just those of an invariant relation, so that the computational adequacy of the denotational semantics of the language can be derived from our general existence theorem (4.16) for invariant relations.

Induction and co-induction. Although an invariant relation Δ for Φ is defined simply as some fixed point of a certain operator induced by Φ, it is in fact the unique fixed point. This follows from the fact that Freyd's mixed initial-algebra/final-coalgebra property of $rec\alpha.\Phi(\alpha)$ (Theorem 3.4) induces a corresponding universal property of Δ (Proposition 4.9).

Now many notions of relation possess distinguished 'identity' relations at each domain that are admissible and that are preserved by the action of domain constructors. (Definition 6.1 gives our abstract requirements for such identity relations.) In such cases the identity relation is necessarily a fixed point of the operator defining Δ and hence by the uniqueness property, the invariant relation on a recursively defined domain is just this identity relation. In this situation, the universal property of Δ gives rise to a rule of inference of mixed inductive/co-inductive character. Sect. 6 demonstrates that by varying the notion of relation and the choice of action of the domain constructors on relations, this rule gives rise to induction and co-induction principles for a wide class of recursively defined domains.

The induction principle (Theorem 6.5) coincides with structural induction when the recursively defined domain is the lift of an inductively defined set. More generally, if the recursively defined domain is given by a type constructor in which the defined type only occurs positively, then the induction property coincides with an *initial algebra induction principle* in the sense of Lehmann and Smyth (1981). In particular it includes the induction principle for the *fixed point object* of the lifting monad, studied by Crole and Pitts (1992). Scott's principle of induction for admissible properties of least fixed points of continuous functions is a consequence of this case. We show that the induction property can yield a non-trivial proof principle even for problematic recursively defined domains, such as those that model untyped lambda calculus: see Example 6.8.

The co-induction principle (Theorem 6.12) is the one which the author established in (Pitts, 1992). It takes the form of an extensionality property: two elements d, d' of a recursively defined domain satisfy $d \sqsubseteq d'$ (respectively $d = d'$) if and only if there is a 'simulation' (respectively a 'bisimulation') relating them. In contrast to the induction principle, the co-induction principle deals with arbitrary (binary) relations rather than just admissible ones. The proof of it that we give here as a corollary of the general theory developed in Sect. 4 is simpler than that given in (Pitts, 1992). However, the method of *loc. cit.* can deal with recursively defined domains involving powerdomain constructors, whereas there are difficulties with defining the action on relations of such constructors needed for the theory developed here: see Remark 6.14.

Parameterized domain equations. For simplicity of exposition, most of the theory developed in this paper applies to domains recursively defined by a single equation (1). The extension of the theory to *simultaneous* domain equations is straightforward. However, the important case of domain equations with free parameters (needed to treat domain constructors that are themselves recursively defined) introduces some extra complications in connection with the

technique of separating positive and negative occurrences. This extension is considered briefly in the final section of the paper.

3 Minimal solutions of domain equations

This section reviews as much of the theory of recursively defined domains as we need. We draw upon the category-theoretic approach of Smyth and Plotkin (1982), revised in the light of the recent work of Freyd on 'algebraically compact' categories (Freyd, 1991, 1992).

We use the term *cpo* to mean a partially ordered set possessing least upper bounds of all countable chains, but not necessarily possessing a least element. A *pointed* cpo, or *cppo* for short, is a cpo D that does possess a least element, denoted $\bot_D$ (or just $\bot$). Let $\mathbf{Cpo}_\bot$ denote the category whose objects are cppos and whose morphisms are monotone functions that are both continuous (preserve least upper bounds of countable chains) and strict (preserve the least element). We will use the notation $f : D \circ\!\!\longrightarrow E$ to indicate that f is such a function between cppos. Our domain theoretic terminology generally follows that of the survey article of Gunter and Scott (1990). To fix notation we recall the following standard constructions on cpos and cppos.

Definition 3.1 (i) **Lifting and 'unlifting'.** Recall that each cpo X gives rise to a cppo

$$X_\bot \stackrel{def}{=} X \cup \{\bot\}$$

called its *lift*, by adjoining an element $\bot \notin X$ and extending the partial order by making $\bot$ least.

If D is a cppo, we denote by $D_\downarrow$ the cpo obtained by removing the least element of D. Thus

$$D_\downarrow \stackrel{def}{=} \{x \in D \mid x \neq \bot_D\} \ .$$

(ii) **Discrete cpos and flat cppos.** Each set X gives rise to a *discrete* cpo via the partial order: $x \sqsubseteq_X x'$ if and only if $x = x'$. The *flat* cppos are the lifts of discrete cpos.

(iii) **Cartesian and smash products.** The *cartesian product* of two cpos X and Y is the cpo with underlying set

$$X \times Y \stackrel{def}{=} \{(x, y) \mid x \in X \text{ and } y \in Y\}$$

partially ordered componentwise: $(x, y) \sqsubseteq_{X\times Y} (x', y')$ if and only if $x \sqsubseteq_X x'$ and $y \sqsubseteq_Y y'$. Clearly $X \times Y$ is pointed when X and Y are, with least element $(\bot_X, \bot_Y)$. The *smash product* of two cppos D and E is the cppo

$$D \otimes E \stackrel{def}{=} (D_\downarrow \times E_\downarrow)_\bot \ .$$

(iv) **Disjoint union and coalesced sum.** The *disjoint union* of finitely many cpos $X_1 \ldots, X_n$ is the cpo with underlying set

$$X_1 \uplus \cdots \uplus X_n \stackrel{def}{=} \{in_i(x) \mid i = 1, \ldots, n \wedge x \in X_i\}$$

where $x \mapsto in_i(x)$ $(i = 1, \ldots, n)$ are injective functions with disjoint images. The partial order on $X \uplus \cdots \uplus X_n$ is: $z \sqsubseteq_{X_1 \uplus \cdots \uplus X_n} z'$ if and only if $z = in_i(x)$ and $z' = in_i(x')$ for some i and $x, x' \in X_i$ with $x \sqsubseteq_{X_i} x'$.

The *coalesced sum* of cppos $D_1, \ldots, D_n$ is the cppo

$$D_1 \oplus \cdots \oplus D_n \stackrel{def}{=} ((D_1)_\downarrow \uplus \cdots \uplus (D_n)_\downarrow)_\perp \ .$$

The insertion functions $in_i : (D_i)_\downarrow \longrightarrow (D_1)_\downarrow \uplus \cdots \uplus (D_n)_\downarrow$ extend to injective, strict continuous functions $D_i \circ\!\!\longrightarrow D_1 \oplus \cdots \oplus D_n$ which we also denote by in_i.

(v) **Function spaces.** If X and Y are cpos, their *continuous function space* is the cpo $X \to Y$ with underlying set all continuous functions from X to Y, partially ordered pointwise from Y: $f \sqsubseteq_{X \to Y} f'$ if and only if for all $x \in X$, $f(x) \sqsubseteq_Y f'(x)$. Clearly if Y is pointed then so is $X \to Y$, with least element the constant function $\lambda x \in X.\perp_Y$.

The *strict continuous function space* of two cppos D and E is the cppo whose underlying set is

$$D \multimap E \stackrel{def}{=} \{f \in (D \to E) \mid f(\perp_D) = \perp_E\}$$

and whose partial order and least element are inherited from those of $D \to E$.

As is well known, these constructions on cppos are functorial for strict continuous functions. Thus lifting determines a functor $\mathbf{Cpo}_\perp \longrightarrow \mathbf{Cpo}_\perp$, the cartesian and smash products and the binary coalesced sum determine functors $\mathbf{Cpo}_\perp \times \mathbf{Cpo}_\perp \longrightarrow \mathbf{Cpo}_\perp$, and the strict and non-strict continuous function spaces determine functors $\mathbf{Cpo}_\perp^{\mathrm{op}} \times \mathbf{Cpo}_\perp \longrightarrow \mathbf{Cpo}_\perp$. Moreover, all these functors are *locally continuous*, in the sense that their action on morphisms preserves least upper bounds of countable chains.

Consider a domain equation of the form

$$\alpha = \Phi(\alpha) \tag{4}$$

where Φ is a formal expression built up from the variable α and constants ranging over cppos, using the constructors $(-)_\perp$, $\times$, $\otimes$, $\oplus$, $\to$, and $\multimap$. A solution to (4) is specified by a cppo D together with an isomorphism $D \cong \Phi(D)$. Here $\Phi(D)$ denotes the cppo that results from interpreting the variable α as the cppo D in Φ. The seminal work of Scott, Plotkin and several others shows that such solutions always exist: there is the category-theoretic construction in terms of the colimit of a chain of embedding-projection pairs which was Scott's original method and whose full ramifications are presented in (Smyth and Plotkin, 1982); there is the set-theoretic construction in terms of Scott's notion of 'information system' (Scott, 1982; Winskel and Larsen, 1984), which has the advantage of yielding solutions up to equality rather than just up to isomorphism; and there is the related approach using least fixed points of continuous operators on universal domains, described in (Gunter and Scott, 1990).

These various constructions serve not only to prove the existence of some solution to a domain equation, but in fact produce a solution that is *minimal*, in

a sense which we review in this section. The minimality property is important for at least two reasons. First, such minimal solutions to domain equations turn out to be unique up to isomorphism. Thus all the methods mentioned above construct essentially the same cppo, which we will write as $rec\alpha.\Phi(\alpha)$ and refer to as *the cppo recursively defined by* (4). Secondly, such minimal solutions are needed to ensure denotational semantics of programming language expressions are computationally adequate (in the technical sense described in the survey (Meyer and Cosmodakis, 1988)) for their operational behaviour. We give a concrete example of this in Sect. 5.

In order to express a minimality condition on solutions of (4), one needs some notion of comparison between cppos that is preserved by the action of the constructor Φ. The more liberal the notion of comparison, the stronger will be the minimality property. In view of the functoriality of the relevant constructors noted above, one might hope to compare two cppos via the morphisms between them in the category $\mathbf{Cpo}_\perp$. The problem is that because of the function space constructors, $\Phi(\alpha)$ may well contain both positive and negative occurrences of the variable α. (Recall that an occurrence of α in Φ is negative if one passes through an odd number of left-hand branches of (strict or non-strict) function space constructors between the occurrence and the root of the parse tree; the occurrence is positive otherwise.) If Φ only contains positive occurrences of α, then it determines a covariant locally continuous functor $\Phi(-) : \mathbf{Cpo}_\perp \longrightarrow \mathbf{Cpo}_\perp$. If it only contains negative occurrences, then it determines a contravariant locally continuous functor, $\Phi(-) : \mathbf{Cpo}_\perp^{\mathrm{op}} \longrightarrow \mathbf{Cpo}_\perp$. However, in the general case the assignment $D \mapsto \Phi(D)$ may not be the object part of either a co- or a contra-variant functor on $\mathbf{Cpo}_\perp$. The traditional way round this problem is to restrict the class of morphism that can act as valid 'comparisons' between cppos. For example by using *embeddings*, which come with associated *projections* in the reverse direction, one achieves functoriality by using the embedding part at positive occurrences of α and the projection part at negative ones.

Instead of complicating the notion of comparison, one can elaborate the constructor $\Phi(\alpha)$ by separating out the two types of occurrence of α. Let $\hat{\Phi}(\alpha^-, \alpha^+)$ denote the result of replacing all positive occurrences of α in Φ by a new variable α^+, and replacing all negative occurrences by a different variable α^-. Since the occurrences of α^+ (respectively α^-) in $\hat{\Phi}$ are all positive (respectively negative), it is now possible to build up, by induction on the structure of Φ, a locally continuous functor

$$\hat{\Phi}(-,+) : \mathbf{Cpo}_\perp^{\mathrm{op}} \times \mathbf{Cpo}_\perp \longrightarrow \mathbf{Cpo}_\perp \ .$$

For example when Φ is $\Phi_1 \multimap \Phi_2$, then the action of $\hat{\Phi}$ on objects sends a pair of cppos (D, E) to the cppo $\hat{\Phi}_1(E, D) \multimap \hat{\Phi}_2(D, E)$, and sends a pair of strict continuous functions $(f : D' \circ\!\!\longrightarrow D, g : E \circ\!\!\longrightarrow E')$ to the strict continuous function $\hat{\Phi}_1(g, f) \multimap \hat{\Phi}_2(f, g) : \hat{\Phi}(D, E) \circ\!\!\longrightarrow \hat{\Phi}(D', E')$ given by

$$\lambda h \in \hat{\Phi}(D, E).\hat{\Phi}_2(f, g) \circ h \circ \hat{\Phi}_1(g, f) \ .$$

Note that the original construction on cppos, $D \mapsto \Phi(D)$, can be recaptured by diagonalizing $\hat{\Phi}$, since clearly $\hat{\Phi}(D, D) = \Phi(D)$. So the study of solutions to domain equations (4) can be subsumed in the study of invariant objects for locally continuous functors $F : \mathbf{Cpo}_\perp^{\mathrm{op}} \times \mathbf{Cpo}_\perp \longrightarrow \mathbf{Cpo}_\perp$, where by an *invariant object* for F we mean a cppo D equipped with an isomorphism $i : F(D, D) \cong D$.

Definition 3.2 Let $F : \mathbf{Cpo}_{\perp}^{\mathrm{op}} \times \mathbf{Cpo}_{\perp} \longrightarrow \mathbf{Cpo}_{\perp}$ be a locally continuous functor. An invariant $i : F(D, D) \cong D$ is *minimal* if the least fixed point $fix(\delta)$ of the continuous function $\delta : (D \multimap D) \longrightarrow (D \multimap D)$ given by $\delta(e) = iF(e, e)i^{-1}$ is the identity function $id_D : D \circ\!\!\longrightarrow D$.

Theorem 3.3 (Existence of minimal invariants) *Any locally continuous functor $F : \mathbf{Cpo}_{\perp}^{op} \times \mathbf{Cpo}_{\perp} \longrightarrow \mathbf{Cpo}_{\perp}$ possesses a minimal invariant.*

Proof The existence of a minimal invariant for a locally continuous functor follows from Scott's original construction of recursively defined domains via colimits of chains of embedding-projection pairs, combined with the characterization of such colimits given by Smyth and Plotkin (1982, Theorem 2). For completeness, we review the details.

Recall that an *embedding* between cppos is a continuous function $f : D \longrightarrow E$ for which there exists a continuous function $f^{\circ} : E \longrightarrow D$ satisfying $id_D = f^{\circ} f$ and $f f^{\circ} \sqsubseteq id_E$. Note that f° is uniquely determined by these conditions; it is called the *projection* corresponding to the embedding f. Note that embeddings and projections are necessarily strict functions, so that we can apply F to them. Since F is locally continuous it is in particular locally monotone and hence $F(f^{\circ}, f) : F(D, D) \longrightarrow F(E, E)$ is an embedding, with corresponding projection $F(f, f^{\circ})$. The one-element cppo $\emptyset_{\perp} = \{\perp\}$ is initial for embeddings: for each cppo E, there is a unique embedding to it from $\emptyset_{\perp}$, given by the least element of $\emptyset_{\perp} \to E$, i.e. the constant function with value $\perp$. So starting from $\emptyset_{\perp}$ and iterating $F(-, +)$ we can build up an ω-chain of embeddings:

$$D_0 \xrightarrow{\ i_0\ } D_1 \xrightarrow{\ i_1\ } D_2 \xrightarrow{\ i_2\ } \cdots \tag{5}$$

$$\text{where} \quad \begin{cases} D_0 = \emptyset_{\perp} \\ D_{n+1} = F(D_n, D_n) \end{cases} \quad \text{and} \quad \begin{cases} i_0 = \perp_{\emptyset_{\perp} \to D_1} \\ i_{n+1} = F(i_n^{\circ}, i_n) \end{cases} .$$

Let D be the cppo with underlying set

$$D \stackrel{def}{=} \left\{ x \in \prod_{n<\omega} D_n \mid x_n = i_n^{\circ}(x_{n+1}),\ \text{all } n < \omega \right\}$$

partially ordered componentwise. This is the limit in $\mathbf{Cpo}_{\perp}$ of the ω^{op}-chain $(i_n^{\circ} \mid n < \omega)$. The projection functions $x \in D \mapsto x_n \in D_n$ are indeed the projection parts of embeddings $j_n : D_n \longrightarrow D$, where for all $y \in D_n$ and all $m < \omega$

$$(j_n(y))_m \stackrel{def}{=} \begin{cases} i_{m,n}^{\circ}(y) & \text{if } m < n \\ y & \text{if } m = n \\ i_{n,m}(y) & \text{if } m > n \ . \end{cases}$$

and where the embedding $i_{m,n} : D_m \longrightarrow D_n$ $(m < n)$ is the composition $i_{n-1} \cdots i_m$. The morphisms $(j_n \mid n < \omega)$ form a colimit for the ω-chain (5) in the category of cppos and embeddings. (Indeed, they form a colimit in the category of cpos and continuous functions.) The limit property of $(j_n^{\circ} \mid n < \omega)$

and the colimit property of $(j_n \mid n < \omega)$ in fact follow from some simple order-theoretic properties, namely

$$j_{n+1} i_n = j_n \ , \qquad id_{D_n} = j_n^\circ j_n \ , \qquad id_D = \bigsqcup_{n<\omega} j_n j_n^\circ \ .$$

Since these properties are preserved under the application of $F(-,+)$, the embeddings $F(j_n^\circ, j_n) : F(D_n, D_n) \longrightarrow F(D, D)$ form a colimit for the ω-chain

$$F(D_0, D_0) \xrightarrow{F(i_0^\circ, i_0)} F(D_1, D_1) \xrightarrow{F(i_0^\circ, i_0)} F(D_2, D_2) \xrightarrow{F(i_2^\circ, i_2)} \cdots \ .$$

Since this is just the chain $(D_n \mid n \geq 1)$ whose colimit is again D, there is an isomorphism $i : F(D, D) \cong D$ with $iF(j_n^\circ, j_n) = j_{n+1}$ for all $n < \omega$. Since i is an isomorphism, $i^\circ = i^{-1}$. Recall from above that for any embedding f, $F(f^\circ, f)^\circ = F(f, f^\circ)$. Combining these facts, we have:

$$\begin{aligned} j_{n+1} j_{n+1}^\circ &= (iF(j_n^\circ, j_n))(iF(j_n^\circ, j_n))^\circ \\ &= iF(j_n^\circ, j_n)F(j_n, j_n^\circ)i^{-1} \\ &= iF(j_n j_n^\circ, j_n j_n^\circ)i^{-1} \ . \end{aligned}$$

Since $j_0 j_0^\circ = \bot$, it follows by induction on n that $j_n j_n^\circ = \delta^n(\bot)$, where δ is the function defined in Definition 3.2. Thus $id_D = \bigsqcup_{n<\omega} j_n j_n^\circ = \bigsqcup_{n<\omega} \delta^n(\bot) = \mathit{fix}(\delta)$. Therefore $i : F(D, D) \cong D$ is indeed a minimal invariant for F. □

The concept of minimal invariant given by Freyd (1991) is apparently stronger than that given in Definition 3.2, in that he requires id_D to be the *unique* fixed point of δ. However, for locally continuous functors this strengthening is a corollary of the following result, which provides a very useful 'universal property' for recursively defined domains.

Theorem 3.4 (Freyd) *Let F, D and i be as in Definition 3.2. For each pair of cppos A, B and each pair of strict continuous functions*

$$f : A \circ\!\!\longrightarrow F(B, A) \qquad g : F(A, B) \circ\!\!\longrightarrow B$$

there are unique strict continuous functions $h : A \circ\!\!\longrightarrow D$ and $k : D \circ\!\!\longrightarrow B$ making the following squares of strict continuous functions commute:

$$\begin{array}{ccc} D & \xrightarrow{i^{-1}} & F(D,D) \\ {\scriptstyle h}\uparrow & & \uparrow{\scriptstyle F(k,h)} \\ A & \xrightarrow[f]{} & F(B,A) \end{array} \qquad \begin{array}{ccc} F(D,D) & \xrightarrow{i} & D \\ {\scriptstyle F(h,k)}\downarrow & & \downarrow{\scriptstyle k} \\ F(A,B) & \xrightarrow[g]{} & B \ . \end{array} \tag{6}$$

The mapping $(f, g) \mapsto (h, k)$ determines a continuous function

$$(A \multimap F(B, A)) \times (F(A, B) \multimap B) \longrightarrow (A \multimap D) \times (D \multimap B) \ .$$

Proof Since we are considering locally continuous functors rather than arbitrary ones as in (Freyd, 1991), we can give a more elementary proof than that in *loc. cit.*, simply relying upon the *uniformity property* enjoyed by least fixed points of continuous functions. This property is often called 'Plotkin's axiom' for least fixed points and states that

> *if* $f : D \circ\!\!\longrightarrow E$ *is a strict continuous function between cppos, and* $d : D \longrightarrow D$ *and* $e : E \longrightarrow E$ *are continuous functions satisfying* $e \circ f = f \circ d$*, then* $\mathit{fix}(e) = f \circ \mathit{fix}(d)$.

The property can be established via a simple fixed point induction: see (Gunter and Scott, 1990, Theorem 2.3).

Turning to the proof of the theorem, consider the function $\phi : (A \multimap D) \times (D \multimap B) \longrightarrow (A \multimap D) \times (D \multimap B)$ given by $\phi(h,k) = (iF(k,h)f, gF(h,k)i^{-1})$. This is a continuous function because F is locally continuous; taking its least fixed point yields a pair of strict continuous functions (h,k) with the required property. To see that this is the only such pair, suppose (h',k') is another and consider the function $\varepsilon : (D \multimap D) \longrightarrow (A \multimap D) \times (D \multimap B)$ defined by $\varepsilon(e) = (eh', k'e)$. Clearly ε is continuous, and it is strict because k' is strict; moreover the assumption that h' and k' satisfy (6) together with the functoriality of F imply that the following diagram commutes, where δ was defined in Definition 3.2:

$$\begin{array}{ccc} (D \multimap D) & \xrightarrow{\varepsilon} & (A \multimap D) \times (D \multimap B) \\ \Big\downarrow{\scriptstyle \delta} & & \Big\downarrow{\scriptstyle \phi} \\ (D \multimap D) & \xrightarrow[\varepsilon]{} & (A \multimap D) \times (D \multimap B) \end{array} .$$

Since the above square commutes and ε is strict, we can apply the uniformity property of least fixed points to conclude that $\mathit{fix}(\phi) = \varepsilon \circ \mathit{fix}(\delta)$. But by minimality of D, $\mathit{fix}(\delta) = id_{D \multimap D}$. Hence $(h,k) \stackrel{def}{=} \mathit{fix}(\phi) = \varepsilon \circ id = (h',k')$, as required.

The last sentence of the theorem follows from the continuity of the least fixed point operator, since the function referred to is

$$\lambda(f,g).\mathit{fix}(\lambda(h,k).(iF(k,h)f, gF(h,k)i^{-1})) \ .$$

□

Remark 3.5 One can paraphrase the above theorem in more categorical terms: if D is a minimal invariant, then the object (D,D) of $\mathbf{Cpo}_{\perp}^{\mathrm{op}} \times \mathbf{Cpo}_{\perp}$ is simultaneously an initial algebra and a final coalgebra for the endofunctor $(F(+,-), F(-,+)) : \mathbf{Cpo}_{\perp}^{\mathrm{op}} \times \mathbf{Cpo}_{\perp} \longrightarrow \mathbf{Cpo}_{\perp}^{\mathrm{op}} \times \mathbf{Cpo}_{\perp}$ (the structure isomorphism for the initial algebra being (i^{-1}, i) and for the final coalgebra being its inverse). This coincidence of initial algebra with final coalgebra is the concept of *free algebra* of an endofunctor that lies at the heart of Freyd's categorical analysis of recursive types. Whilst it is true that all the ingredients of the proofs of Theorems 3.3 and 3.4 have been present for some time—they can certainly

be found in (Smyth and Plotkin, 1982)—Freyd has abstracted from them what appears to be a very powerful property. Note also that since initial algebras (and final coalgebras) are unique up to isomorphism by virtue of the 'universal property' they enjoy, we have in particular that *any two minimal invariants for F are isomorphic.*

When F is the locally continuous functor $\hat{\Phi}(-,+) : \mathbf{Cpo}_{\perp}^{\mathrm{op}} \times \mathbf{Cpo}_{\perp} \longrightarrow \mathbf{Cpo}_{\perp}$ obtained by separating positive and negative occurrences of α in a domain constructor $\Phi(\alpha)$ as in (4), then the minimal invariant of F constructed in the proof of Theorem 3.3 is precisely the recursively defined cppo $rec\alpha.\Phi(\alpha)$ constructed via Scott's method as a colimit of embeddings. The minimal invariant property of Definition 3.2 provides a very powerful tool for reasoning about $rec\alpha.\Phi(\alpha)$ without having to delve into the details of its construction. The relational properties presented in this paper are an illustration of this, since they derive directly from the minimal invariant property. Here is another illustration—a pleasingly simple proof that the Curry fixed point combinator coincides with the least fixed point operator in a certain domain model of the untyped lambda calculus. The argument is due to Plotkin (private communication) and should be compared with the proofs of similar results in (Barendregt, 1984, Exercise 18.4.20) and (Winskel, 1993, Sect. 13.10.2).

Example 3.6 Let D be $rec\alpha.(\alpha \to \alpha)_{\perp}$, the canonical domain model of the lazy lambda calculus studied in (Abramsky, 1990) and (Abramsky and Ong, 1993). Thus D is the minimal invariant of the locally continuous functor

$$((-) \to (+))_{\perp} : \mathbf{Cpo}_{\perp}^{\mathrm{op}} \times \mathbf{Cpo}_{\perp} \longrightarrow \mathbf{Cpo}_{\perp}$$

and comes equipped with an isomorphism $i : (D \to D)_{\perp} \cong D$. In this case the minimal invariant property says that id_D is the least fixed point of the function mapping $e \in (D \multimap D)$ to $\delta(e) \in (D \multimap D)$, where for each $x \in D$

$$\delta(e)(x) \stackrel{def}{=} \begin{cases} i(e \circ f \circ e) & \text{if } i^{-1}(x) = f \in (D \to D) \\ \perp & \text{if } i^{-1}(x) = \perp. \end{cases}$$

Let $x, y \mapsto x \cdot y$ denote the application function on D: when $x = \perp$, then $x \cdot y = \perp$; and when $x \neq \perp$, then $x \cdot y = f(y)$ where $f \in (D \to D)$ is the unique element such that $x = i(f)$. For each $f \in (D \to D)$ the 'Curry' fixed point of f is given by $Y_f = \Delta_f \cdot \Delta_f \in D$, where

$$\Delta_f \stackrel{def}{=} i(\lambda x \in D.f(x \cdot x)) \ .$$

We wish to prove that $Y_f = \mathit{fix}(f)$, the least fixed point of f.

Certainly $\mathit{fix}(f) \sqsubseteq Y_f$, since Y_f is a fixed point of f. So it suffices to show that $Y_f \sqsubseteq \mathit{fix}(f)$. Consider the subset of $D \multimap D$ given by

$$E \stackrel{def}{=} \{e \mid e \sqsubseteq id_D \wedge e(\Delta_f) \cdot \Delta_f \sqsubseteq \mathit{fix}(f)\} \ .$$

We have to show that id_D, i.e. $\mathit{fix}(\delta)$, lies in E. Note that E is chain-complete and contains $\perp$. So by fixed point induction it suffices to show that δ maps E into itself. This is a straightforward calculation using the definitions of δ and Δ_f, together with the fact that elements of E satisfy $e \sqsubseteq id_D$.

4 Relational structures

One would like reasoning principles for recursively defined domains that apply to as wide a class of properties of domains as possible. For this reason we will employ a rather general axiomatic framework for relations on domains. The framework is surprisingly simple—remarkably few properties of a general notion of 'relation' are needed to establish quite powerful results. Accordingly there is a diversity of examples which fit into this framework. The structure for relations we use is based on that used by O'Hearn and Tennent (1993) in their work on applying relational parametricity to the study of local-variable declarations. To it we add a treatment of the crucial property of relations that in LCF (Gordon, Milner and Wadsworth, 1979) is called *admissibility* (meaning 'admitting induction').

Definition 4.1 A *relational structure*, $\mathcal{R}$, on a category $\mathbf{C}$ is specified by the following data:

- For each $\mathbf{C}$-object D, a set $\mathcal{R}(D)$ (of '$\mathcal{R}$-relations on D').
- For each $\mathbf{C}$-morphism $f : D \longrightarrow E$, a binary relation between elements $R \in \mathcal{R}(D)$ and elements $S \in \mathcal{R}(E)$, written $f : R \subset_{\mathcal{R}} S$ (or just $f : R \subset S$). These binary relations are required to satisfy the following properties:
  (a) $id_D : R \subset R$, for all $R \in \mathcal{R}(D)$;
  (b) if f and g are composable, $f : R \subset S$ and $g : S \subset T$, then $gf : R \subset T$.

This notion of relational structure is weaker than that employed by O'Hearn and Tennent (1993, Sect. 2) in that we do not require $\mathcal{R}$ to possess distinguished 'identity' relations $I_D \in \mathcal{R}(D)$ satisfying $f : I_D \subset I_E$, for any $f : D \longrightarrow E$. This level of generality is needed for the application to proofs of computational adequacy that we give in Sect. 5. However, identity relations will play a key role in formulating the induction and coinduction principles considered in Sect. 6. For all these applications we only need to consider 'unary' relational structures, whereas O'Hearn and Tennent need binary ones in order to use Reynolds' notion of relational parametricity (Reynolds, 1983). In general, an n-ary relational structure would be specified by sets $\mathcal{R}(D_1, \ldots, D_n)$ for each n-tuple of objects, and by relations $f_1, \ldots, f_n : R \subset S$ (where $f_i : D_i \longrightarrow E_i$, for $i = 1, \ldots, n$), satisfying analogues of conditions (a) and (b).

As will be seen from the examples of relational structures given below, the assertion '$f : R \subset S$' abstracts the notion of a function mapping related elements to related elements. Making this the fundamental notion contrasts with the usual axiomatic approach to predicates in categorical logic where, beginning with Lawvere's work on 'hyperdoctrines' (Lawvere, 1970), the notion of the inverse image of a predicate along a map is fundamental. There are close connections between the two notions (see Remark 4.15). The principle reason for starting with relational structures rather than hyperdoctrines is that we will be concerned with actions of functors on relations that do preserve the '$f : R \subset S$' relationship, but do not preserve the operation of taking inverse images of relations. Similarly, the notion of composition of (binary) relations is fundamental to much work on categories of relations (see (Freyd and Scedrov, 1990) for example), but plays little role here because it is not preserved by the functors we need to consider.

Here are some examples of relational structures on the category $\mathbf{Cpo}_\perp$ of cppos and strict continuous functions that will be relevant to this paper.

Example 4.2 (i) **'Strict' relations**. Fix a natural number $n \geq 1$. For each cppo D let $\mathcal{R}(D)$ consist of all subsets of the n-fold smash product $D \otimes \cdots \otimes D$ containing $\perp$. The relation $f : R \subset S$ is defined to hold if and only if for all $u \in R$, $(f \otimes \cdots \otimes f)(u) \in S$. We will use this example, for the case $n = 1$, to derive an induction principle for recursively defined domains in Sect. 6.

(ii) **'Non-strict' relations**. Fix a natural number $n \geq 1$, and for each cppo D let $\mathcal{R}(D)$ consist of all subsets of the n-fold product $D \times \cdots \times D$. The relation $f : R \subset S$ is defined to hold if and only if for all $u \in R$, $(f \times \cdots \times f)(u) \in S$. We will use this example, for the case $n = 2$, to derive a co-induction principle for recursively defined domains in Sect. 6.

(iii) Let I be a set. In Sect. 5 we will make use of a relational structure $\mathcal{R}$ that is the I-fold power of the $n = 1$ case of example (i). This can be described more explicitly as follows. For each cppo D let $\mathcal{R}(D)$ consist of all subsets of $D_\downarrow \times I$. (Recall that $D_\downarrow = D \setminus \{\perp\}$.) The relation $f : R \subset S$ is defined to hold if and only if for all $(d, i) \in R$, if $f(d) \neq \perp$ then $(f(d), i) \in S$.

Given a relational structure $\mathcal{R}$ on a category $\mathbf{C}$, the binary relation $\subset$ on each set $\mathcal{R}(D)$ defined by

$$R \subset R' \Leftrightarrow id_D : R \subset R'$$

is a preorder (by virtue of properties (a) and (b) in the definition). We say that $\mathcal{R}$ is *normal* if these preorders are all partial orders, i.e. if $R \subset R'$ and $R' \subset R$ imply $R = R'$. The examples of relational structures above are all normal. Moreover, an arbitrary relational structure can be turned into an equivalent normal one by quotienting by the equivalence relation associated with the preorders $\subset$. *From now on we will assume that all relational structures we deal with are normal.*

Definition 4.3 Let $\mathcal{R}$ be a relational structure on $\mathbf{Cpo}_\perp$. An $\mathcal{R}$-relation $R \in \mathcal{R}(D)$ is called *admissible* if for all other $\mathcal{R}$-relations $S \in \mathcal{R}(E)$ the subset

$$[S, R] \stackrel{def}{=} \{f \mid f : S \subset R\}$$

of the strict function space cppo $E \multimap D$ contains $\perp$ and is closed under taking least upper bounds of countable chains. We let $\mathcal{R}_{adm}(D)$ denote the subset of admissible $\mathcal{R}$-relations on D.

Remark 4.4 Although we have defined admissibility in terms of the concrete structure of cppos, the definition can in fact be phrased purely in terms of the *monoidal closed* structure $\langle 1_\perp, \otimes, \multimap \rangle$ on the category $\mathbf{Cpo}_\perp$. (See (MacLane, 1971, Chapter VII) for an introduction to this categorical notion.) Here $1_\perp$ is the lift of a one-element set: it is a unit for the smash product operation on cppos. Using the adjunction between $(-) \otimes E$ and $E \multimap (-)$, morphisms $1_\perp \circ\!\!\longrightarrow (E \multimap D)$ correspond naturally to morphisms $E \cong (1_\perp \otimes E) \circ\!\!\longrightarrow D$. Then $R \in \mathcal{R}(D)$ is admissible if and only if for each $S \in \mathcal{R}(E)$ there is a morphism $m : [S, R] \circ\!\!\longrightarrow (E \multimap D)$ with the property that composition with m induces a

bijection between morphisms $1_\perp \circ\!\!\longrightarrow [S,R]$ and morphisms $f : E \circ\!\!\longrightarrow D$ for which $f : S \subset R$ holds. (Such an m is necessarily a monomorphism, hence determines a sub-cppo of $(E \multimap D)$—namely the one given in Definition 4.3.)

Example 4.5 When $\mathcal{R}$ is the relational structure of Example 4.2(i), $R \in \mathcal{R}(D)$ is admissible if and only if the subset $R \subseteq D \otimes \cdots \otimes D$ is chain-complete. Since R contains $\perp$ by definition of $\mathcal{R}$, this condition is easily seen to be sufficient. To see that it is also necessary, suppose that $(x_m \mid m < \omega)$ is a countable chain in R and that x is its least upper bound in $D \otimes \cdots \otimes D$. For each $m < \omega$, let $f_m : n_\perp \multimap D$ be the strict continuous function mapping each $j \in n = \{0, 1, \ldots, n-1\}$ to the jth component of x_m. Taking S to be $\{(0, 1, \ldots, n-1), \perp\} \in \mathcal{R}(n_\perp)$, note that $f_m : S \subset R$. The functions f_m form a countable chain whose least upper bound f is the strict continuous function mapping each $j \in n$ to the jth component of x. Since R is admissible, we have $f : S \subset R$, and hence $x \in R$, as required.

A similar argument shows that for Example 4.2(ii) $R \in \mathcal{R}(D)$ is admissible if and only if it is a chain-complete subset of $D \times \cdots \times D$ containing $(\perp, \ldots, \perp)$. For Example 4.2(iii), $R \in \mathcal{R}(D)$ is admissible if and only if for each $i \in I$, $\{d \mid (d, i) \in R\}$ is a chain-complete subset of $D_\downarrow$.

Given a particular relational structure $\mathcal{R}$ on $\mathbf{Cpo}_\perp$, we are interested in lifting functorial constructions on cppos to constructions on $\mathcal{R}$-relations. The basic requirement is that the construction on $\mathcal{R}$-relations respects the relation $- : - \subset_{\mathcal{R}} -$. However, in some important examples such actions may be subject to certain admissibility conditions. The following definition sums up exactly what we require of a general, mixed variance functor on $\mathbf{Cpo}_\perp$, $F : (\mathbf{Cpo}_\perp^{\mathrm{op}})^m \times (\mathbf{Cpo}_\perp)^n \longrightarrow \mathbf{Cpo}_\perp$ $(m, n \geq 0)$. Note that by taking one or other of m and n to be 0, this includes the case of functors that are purely covariant or purely contravariant.

Definition 4.6 Let $\mathcal{R}$ be a relational structure on $\mathbf{Cpo}_\perp$ and let

$$F : (\mathbf{Cpo}_\perp^{\mathrm{op}})^m \times (\mathbf{Cpo}_\perp)^n \longrightarrow \mathbf{Cpo}_\perp$$

be a functor (where $m, n \geq 0$). An *admissible action of F on $\mathcal{R}$-relations* is an operation mapping tuples of $\mathcal{R}$-relations $R \in \prod_{i=1}^m \mathcal{R}(D_i)$ and $S \in \prod_{j=1}^n \mathcal{R}(E_j)$ to an $\mathcal{R}$-relation $F(R,S) \in \mathcal{R}(F(D_1, \ldots, D_m, E_1, \ldots, E_n))$, satisfying:

(i) if each S_j is admissible then so is $F(R,S)$, for any R;

(ii) if $f_i : R_i' \subset R_i$ and $g_j : S_j \subset S_j'$ *with each S_j' admissible*, then

$$F(f_1, \ldots, f_m, g_1, \ldots, g_n) : F(R,S) \subset F(R',S') \ .$$

Example 4.7 Examples of admissible actions will be given in Sects 5 and 6. In many cases condition (ii) of the definition holds without the admissibility restriction on S_j'. One example where it is needed occurs when we derive the co-induction principle introduced in (Pitts, 1994) by making use of actions of the familiar domain constructors on binary relations on cppos. The relational structure involved is the $n = 2$ case of Example 4.2(ii): $\mathcal{R}$-relations on a cppo D are subsets of $D \times D$, and $f : R \subset S$ holds if $f \times f$ maps R into S. The action of the function space constructors $\to, \multimap : \mathbf{Cpo}_\perp^{\mathrm{op}} \times \mathbf{Cpo}_\perp \longrightarrow \mathbf{Cpo}_\perp$ on $\mathcal{R}$-relations that will be needed is an 'intensional' one (relating two functions

if they give related results on any argument) rather than an 'extensional' one (relating two functions if they map related arguments to related results). In particular the action of $\multimap$ on $R \subseteq D \times D$ and $S \subseteq E \times E$ is the subset of $(D \multimap E) \times (D \multimap E)$ defined by

$$R \multimap S \stackrel{def}{=} \{(f_1, f_2) \mid \forall x \in D(x \neq \bot \Rightarrow (f_1(x), f_2(x)) \in S)\} \ .$$

Given $f : R' \subset R$ and $g : S \subset S'$, in general one does not have $f \multimap g : (R \multimap S) \subset (R' \multimap S')$, but this does hold if $(\bot, \bot) \in S'$. We noted above that for this relational structure, $S \in \mathcal{R}(D)$ is admissible if and only if it is chain-complete and contains $(\bot, \bot)$. So the mapping $R, S \mapsto R \multimap S$ does indeed satisfy conditions (i) and (ii) of the definition.

Note that this action is different from the one which arises in connection with 'logical relations', where one defines

$$R \multimap S \stackrel{def}{=} \{(f_1, f_2) \mid \forall (x_1, x_2) \in R.(f_1(x_1), f_2(x_2)) \in S\} \ .$$

This illustrates the fact that a domain constructor may have several different actions on the same relational structure.

The following definition contains the key notion of this paper.

Definition 4.8 (Invariant $\mathcal{R}$-relations) Let $\mathcal{R}$ be a relational structure on $\mathbf{Cpo}_\bot$. Let $F : \mathbf{Cpo}_\bot^{\mathrm{op}} \times \mathbf{Cpo}_\bot \longrightarrow \mathbf{Cpo}_\bot$ be a locally continuous functor equipped with an admissible action on $\mathcal{R}$-relations and let $i : F(D, D) \cong D$ be the minimal invariant for F whose existence is guaranteed by Theorem 3.3. An *invariant $\mathcal{R}$-relation* for F is an admissible $\mathcal{R}$-relation Δ on D satisfying $i^{-1} : \Delta \subset F(\Delta, \Delta)$ and $i : F(\Delta, \Delta) \subset \Delta$.

Proposition 4.9 *Let $\mathcal{R}$, F, (D, i) and Δ be as in the above definition. For any $f : A \circ\!\!\longrightarrow F(B, A)$, $g : F(A, B) \circ\!\!\longrightarrow B$, $R \in \mathcal{R}(A)$, and $S \in \mathcal{R}_{adm}(B)$, if*

$$f : R \subset F(S, R) \quad \textit{and} \quad g : F(R, S) \subset S$$

then

$$h : R \subset \Delta \quad \textit{and} \quad k : \Delta \subset S$$

where $h : A \circ\!\!\longrightarrow D$ and $k : D \circ\!\!\longrightarrow B$ are the strict continuous functions determined by f and g as in Theorem 3.4.

Proof For any $(u, v) \in (A \multimap D) \times (D \multimap B)$, if $u : R \subset \Delta$ and $v : \Delta \subset S$, then since Δ and S are admissible, we have

$$F(v, u) : F(S, R) \subset F(\Delta, \Delta) \quad \textit{and} \quad F(u, v) : F(\Delta, \Delta) \subset F(R, S)$$

and hence

$$iF(v, u)f : R \subset \Delta \quad \textit{and} \quad gF(u, v)i^{-1} : \Delta \subset S \ .$$

Thus the function $\phi = \lambda(u, v).(iF(v, u)f, gF(u, v)i^{-1})$ maps the set

$$[R, \Delta] \times [\Delta, S] = \{(h, k) \mid h : R \subset \Delta \wedge k : \Delta \subset S\}$$

into itself. Since Δ and S are admissible $\mathcal{R}$-relations, this is a chain-complete subset of $(A \multimap D) \times (D \multimap B)$ containing $\bot$. Hence by fixed point induction, $\mathit{fix}(\phi) \in [R, \Delta] \times [\Delta, S]$. But from the proof of Theorem 3.4 we have that $\mathit{fix}(\phi) = (h, k)$. Hence $h : R \subset \Delta$ and $k : \Delta \subset S$, as required. $\Box$

Corollary 4.10 *The invariant $\mathcal{R}$-relation Δ satisfies the following rule for all $R^-, R^+ \in \mathcal{R}(D)$:*

$$\frac{i^{-1} : R^- \subset F(R^+, R^-) \quad i : F(R^-, R^+) \subset R^+}{R^- \subset \Delta \subset R^+} \quad (R^+ \text{ admissible}). \qquad (7)$$

In particular, Δ is unique if it exists.

Proof Take $A = B = D$, $f = i^{-1}$, $g = i$, $R = R^-$, and $S = R^+$ in the proposition. In this case $h = k = id_D$, since id_D satisfies the commutative squares (6) that determine (h, k) uniquely. So the conclusion of the proposition is $id_D : R^- \subset \Delta$ and $id_D : \Delta \subset R^+$, as required. If Δ' is another invariant $\mathcal{R}$-relation, we can apply (7) with $R^- = R^+ = \Delta'$ to conclude that $\Delta' \subset \Delta$ and $\Delta \subset \Delta'$ and hence that $\Delta' = \Delta$ (since we are assuming that $\mathcal{R}$ is normal). $\square$

What is the significance of this notion of invariant $\mathcal{R}$-relation? We will see in the next section that the existence of such a relation lies at the heart of proofs of 'computational adequacy' for denotational semantics involving recursively defined domains. Then in Sect. 6 we will see that the property of Δ given in the above corollary gives rise to various reasoning principles for recursively defined domains, including a general form of induction for admissible properties and the co-induction principle introduced in (Pitts, 1994).

Remark 4.11 Proposition 4.9 can be viewed as a result about Freyd's notion of 'free algebra' (cf. Remark 3.5). First note that each relational structure $\mathcal{R}$ on $\mathbf{Cpo}_\perp$ gives rise to a new (cpo enriched) category $\mathbf{Cpo}_\perp\{\mathcal{R}\}$ of *cppos equipped with admissible $\mathcal{R}$-relations*. The objects are pairs, written $\{D|R\}$, where D is a cppo and $R \in \mathcal{R}_{adm}(D)$; morphisms from $\{D|R\}$ to $\{E|S\}$ are morphisms $f : D \circ\!\!\rightarrow E$ in $\mathbf{Cpo}_\perp$ for which $f : R \subset S$ holds; composition and identities are those of $\mathbf{Cpo}_\perp$. Clearly there is a forgetful functor $U : \mathbf{Cpo}_\perp\{\mathcal{R}\} \longrightarrow \mathbf{Cpo}_\perp$ which is faithful. If $F : \mathbf{Cpo}_\perp^{\mathrm{op}} \times \mathbf{Cpo}_\perp \longrightarrow \mathbf{Cpo}_\perp$ has an admissible action on $\mathcal{R}$-relations, it lifts to a functor $\hat{F}$ whose action on objects is given by

$$\hat{F}(\{D|R\}, \{E|S\}) \stackrel{def}{=} \{F(D, E)|F(R, S)\}$$

and which makes the following diagram commute

$$\begin{array}{ccc} \mathbf{Cpo}_\perp\{\mathcal{R}\}^{\mathrm{op}} \times \mathbf{Cpo}_\perp\{\mathcal{R}\} & \xrightarrow{\hat{F}} & \mathbf{Cpo}_\perp\{\mathcal{R}\} \\ \downarrow{\scriptstyle U \times U} & & \downarrow{\scriptstyle U} \\ \mathbf{Cpo}_\perp^{\mathrm{op}} \times \mathbf{Cpo}_\perp & \xrightarrow[\hat{F}]{} & \mathbf{Cpo}_\perp \end{array} .$$

Then Proposition 4.9 implies that $\{D|\Delta\}$ is simultaneously an initial algebra and a final coalgebra for the functor $(\hat{F}(+,-), \hat{F}(-,+)) : \mathbf{Cpo}_\perp\{\mathcal{R}\}^{\mathrm{op}} \times \mathbf{Cpo}_\perp\{\mathcal{R}\} \longrightarrow \mathbf{Cpo}_\perp\{\mathcal{R}\}^{\mathrm{op}} \times \mathbf{Cpo}_\perp\{\mathcal{R}\}$.

We will show that invariant $\mathcal{R}$-relations exist when the relational structure $\mathcal{R}$ has certain completeness properties, summed up by the following definition.

Definition 4.12 Let $\mathcal{R}$ be a relational structure on a category $\mathbf{C}$.

(i) $\mathcal{R}$ has *inverse images* if for all $f : D \longrightarrow E$ and $S \in \mathcal{R}(E)$, there is an $\mathcal{R}$-relation f^*S on D satisfying

$$g : R \subset f^*S \Leftrightarrow fg : R \subset S$$

for all g and R.

(ii) $\mathcal{R}$ has *intersections* if for all $\mathcal{S} \subseteq \mathcal{R}(E)$, there is an $\mathcal{R}$-relation $\bigcap \mathcal{S}$ on E satisfying

$$g : R \subset \bigcap \mathcal{S} \Leftrightarrow \forall S \in \mathcal{S}(g : R \subset S)$$

for all g and R.

Note that $f^{-1}S$ and $\bigcap \mathcal{S}$ are determined by the above properties uniquely (up to equivalence in general, and up to equality in normal relational structures).

Example 4.13 All the relational structures of Example 4.2 have inverse images and intersections. In example (i), $f^*S = \{u \in D \otimes \cdots \otimes D \mid (f \otimes \cdots \otimes f)(u) \in S\}$; in example (ii) $f^*S = \{u \in D \times \cdots \times D \mid (f \times \cdots \times f)(u) \in S\}$; and in example (iii) $f^*S = \{(d, i) \mid (f(d), i) \in S\}$. In each case $\bigcap \mathcal{S}$ is given by set-theoretic intersection.

The following consequences of the definition are easily established.

Lemma 4.14 *Let $\mathcal{R}$ be a (normal) relational structure on $\mathbf{Cpo}_\perp$ possessing inverse images and intersections. Each inverse image operator f^* is a monotone function, and taking inverse images is contravariantly functorial in the sense that $id_D^* = id_{\mathcal{R}(D)}$ and $(fg)^* = g^*f^*$. Each $\mathcal{R}(D)$ is a complete lattice under the $\subset$ ordering, with meets given by intersections. Admissible $\mathcal{R}$-relations are closed under taking intersections and inverse images.*

Remark 4.15 From the lemma, we have that any normal relational structure on $\mathbf{C}$ with inverse images determines a '$\mathbf{C}$-indexed poset', i.e. a functor $\mathcal{R}(-) : \mathbf{C}^{\mathrm{op}} \longrightarrow \mathcal{P}oset$, where $\mathcal{P}oset$ is the category of posets and monotone functions. (The action of $\mathcal{R}(-)$ on morphisms is defined by $R(f) = f^*$.) In fact, any $\mathbf{C}$-indexed poset arises in this way: given $\mathcal{R}(-) : \mathbf{C}^{\mathrm{op}} \longrightarrow \mathcal{P}oset$, the corresponding relational structure has relations given by the elements of the posets $\mathcal{R}(D)$ and $f : R \subset S$ is defined to hold if and only if $R \subset \mathcal{R}(f)(S)$, where $\subset$ denotes the partial order on $\mathcal{R}(D)$.

Theorem 4.16 (Existence of invariant relations) *Let $\mathcal{R}$ be a relational structure on $\mathbf{Cpo}_\perp$ possessing inverse images and intersections . Let $F : \mathbf{Cpo}_\perp^{op} \times \mathbf{Cpo}_\perp \longrightarrow \mathbf{Cpo}_\perp$ be a locally continuous functor equipped with an admissible action on $\mathcal{R}$-relations. Let $i : F(D, D) \cong D$ be the minimal invariant for F (whose existence is guaranteed by Theorem 3.3). Then the invariant $\mathcal{R}$-relation for F exists.*

Proof Using inverse images, the existence of an invariant $\mathcal{R}$-relation reduces to the solution of a fixed point equation on the poset $\mathcal{R}(D)$. For using the facts about inverse images stated in Lemma 4.14, it is not hard to see that Δ

is the invariant $\mathcal{R}$-relation for F if and only if it is a fixed point of the function $\phi : \mathcal{R}_{adm}(D) \longrightarrow \mathcal{R}_{adm}(D)$ defined by

$$\phi(R) \stackrel{def}{=} (i^{-1})^* F(R, R) \ .$$

Since $\mathcal{R}$ has intersections (and admissible $\mathcal{R}$-relations are closed under them), $\mathcal{R}_{adm}(D)$ is a complete lattice. However, we cannot immediately appeal to the Tarski-Knaster fixed point theorem to find an element satisfying $\Delta = \phi(\Delta)$, because ϕ is not necessarily a monotone function—since the action of $F(-,+)$ on admissible relations is order reversing in its left-hand argument. Instead we apply the technique of separating positive and negative arguments. For $R^-, R^+ \in \mathcal{R}_{adm}(D)$, define

$$\psi(R^-, R^+) \stackrel{def}{=} (i^{-1})^* F(R^-, R^+) \ .$$

If $id_D : S^- \subset R^-$ and $id_D : R^+ \subset S^+$ with S^+ admissible, then the admissible action of F yields

$$id_{F(D,D)} = F(id_D, id_D) : F(R^-, R^+) \subset F(S^-, S^+)$$

and hence $\psi(R^-, R^+) \subset \psi(S^-, S^+)$. Thus ψ determines a monotone function $\mathcal{R}_{adm}(D)^{\mathrm{op}} \times \mathcal{R}_{adm}(D) \longrightarrow \mathcal{R}_{adm}(D)$. Note that ϕ can be obtained from ψ by diagonalizing. Thus to construct the invariant $\mathcal{R}$-relation it suffices to find Δ satisfying $\Delta = \psi(\Delta, \Delta)$.

'Symmetrizing' ψ, we obtain a monotone operator

$$\psi^{\S} : \mathcal{R}_{adm}(D)^{op} \times \mathcal{R}_{adm}(D) \longrightarrow \mathcal{R}_{adm}(D)^{op} \times \mathcal{R}_{adm}(D)$$

on the complete lattice $\mathcal{R}_{adm}(D)^{op} \times \mathcal{R}_{adm}(D)$ defined by:

$$\psi^{\S}(R^-, R^+) \stackrel{def}{=} (\psi(R^+, R^-), \psi(R^-, R^+)) \ .$$

Now we can apply the Tarski-Knaster fixed point theorem to obtain the least (pre)fixed point of $\psi^{\S}$, which we will denote by (Δ^-, Δ^+). Thus Δ^- and Δ^+ are admissible $\mathcal{R}$-relations on D satisfying:

$$\Delta^- = \psi(\Delta^+, \Delta^-) \quad \wedge \quad \psi(\Delta^-, \Delta^+) = \Delta^+ \tag{8}$$

$$\begin{aligned} &\forall R^-, R^+ \in \mathcal{R}_{adm}(D). \\ &\quad (R^- \subset \psi(R^+, R^-) \wedge \psi(R^-, R^+) \subset R^+) \Rightarrow (R^- \subset \Delta^- \wedge \Delta^+ \subset R^+) \ . \end{aligned} \tag{9}$$

In view of (8), to complete the proof it suffices to show that $\Delta^- = \Delta^+$, i.e. that $\Delta^+ \subset \Delta^-$ and $\Delta^- \subset \Delta^+$. The first of these follows immediately, since by (8) we can take $R^- = \Delta^+$ and $R^+ = \Delta^-$ in (9) to conclude that $\Delta^+ \subset \Delta^-$. So it just remains to prove that $\Delta^- \subset \Delta^+$, i.e. that id_D lies in the set

$$[\Delta^-, \Delta^+] = \{e \mid e : \Delta^- \subset \Delta^+\} \ .$$

Since Δ^+ is admissible, this is a is chain-complete subset of $D \multimap D$ containing $\perp$. By the minimal invariant property of D and fixed point induction, it suffices to check that $[\Delta^-, \Delta^+]$ is closed under the function $\delta : (D \multimap D) \longrightarrow (D \multimap D)$

whose least fixed point is id_D. But if $e : \Delta^- \subset \Delta^+$, then the admissible action of F yields $F(e,e) : F(\Delta^+,\Delta^-) \subset F(\Delta^-,\Delta^+)$. On the other hand property (8) implies $i^{-1} : \Delta^- \subset F(\Delta^+,\Delta^-)$ and $i : F(\Delta^-,\Delta^+) \subset \Delta^+$. Hence if $e : \Delta^- \subset \Delta^+$, then

$$\delta(e) \stackrel{def}{=} iF(e,e)i^{-1} : \Delta^- \subset \Delta^+$$

as required. □

5 Computational adequacy

A denotational semantics of a programming language is said to be *computationally adequate* for an operationally defined notion of program equivalence provided any two programs are equivalent whenever the programs have equal denotations. The study of this property has its origin in the work of Wadsworth (1976), Milne (1973) and Plotkin (1977). Plotkin (1985) reduces the proof of adequacy for a recursively typed functional language to showing the existence of a relation of 'formal approximation' between domain elements and programs, satisfying certain properties. One can see these properties as specifying the relation of formal approximation as the fixed point of a certain operator on relations. Unfortunately this operator is not monotonic, so that establishing that it has a fixed point is non-trivial. In general, this difficulty with non-monotonicity occurs when the denotational semantics of the programming language requires the use of recursively defined domains $rec\alpha.\Phi(\alpha)$ for which α occurs negatively in $\Phi(\alpha)$. Traditionally one has to delve into the detailed construction of the recursively defined domain to establish the existence of the formal approximation relation. In fact, we will see that for a suitable choice of relational structure $\mathcal{R}$, the requirements on the formal approximation relation are just those of the invariant $\mathcal{R}$-relation of Definition 4.8, and one can simply apply Theorem 4.16 to deduce its existence.

We will illustrate this method of proving computational adequacy using a programming language equivalent to a small, but non-trivial fragment of Standard ML (Milner, Tofte and Harper, 1990) containing a single recursive datatype declaration:

$$\texttt{datatype} \quad ty = \texttt{In}_1\ \texttt{of}\ \sigma_1 \mid \cdots \mid \texttt{In}_n\ \texttt{of}\ \sigma_n \ . \tag{10}$$

The types $\sigma_1, \ldots, \sigma_n$ are built up from the ground types *unit* (one-element type), *bool* (booleans), *int* (integers), and the single type identifier *ty*, using any combination of the product ($*$) and function space ($\rightarrow$) constructors. The expressions of the language are given by:

e	::=	x	identifiers
	|	()	unit value
	|	true | false	booleans
	|	if e then e else e	conditional
	|	$\underline{n}$	numerals
	|	op(e, e)	operations on *int* and *bool*
	|	$\mathtt{In}_i$(e)	datatype value
	|	case e of $\{\mathtt{In}_1(\mathtt{x})\Rightarrow\mathtt{e} \mid \cdots \mid \mathtt{In}_n(\mathtt{x})\Rightarrow\mathtt{e}\}$	case expression
	|	(e,e)	pairing
	|	fst(e) | snd(e)	product projections
	|	λx:σ.e	function abstraction
	|	ee	function application
	|	letrec fx = e:σ in fe	recursive functions.

We only consider well-typed expressions: if $\mathtt{x}_1, \ldots, \mathtt{x}_m$ are distinct identifiers and $\tau_1, \ldots, \tau_m$ are types, then the typing assertion

$$\mathtt{x}_1{:}\tau_1, \ldots, \mathtt{x}_m{:}\tau_m \vdash \mathtt{e}{:}\sigma \tag{11}$$

> "if for $i = 1, \ldots, m$ the identifier $\mathtt{x}_i$ has type τ_i, then the expression e has type σ (and has its free identifiers contained in the set $\{\mathtt{x}_1, \ldots, \mathtt{x}_m\}$)"

can be defined inductively in the usual way. For example the rules for datatype values and case expressions are:

$$\frac{\Gamma \vdash \mathtt{e}{:}\sigma_i}{\Gamma \vdash \mathtt{In}_i(\mathtt{e}){:}ty} \qquad \frac{\Gamma \vdash \mathtt{e}{:}ty \quad \Gamma, \mathtt{x}_1{:}\sigma_1 \vdash \mathtt{e}_1{:}\sigma \quad \cdots \quad \Gamma, \mathtt{x}_n{:}\sigma_n \vdash \mathtt{e}_n{:}\sigma}{\Gamma \vdash \mathtt{case\ e\ of}\ \{\mathtt{In}_1(\mathtt{x}_1)\Rightarrow\mathtt{e}_1 \mid \cdots \mid \mathtt{In}_n(\mathtt{x}_n)\Rightarrow\mathtt{e}_n\} : \sigma}$$

where the types $\sigma_1, \ldots, \sigma_n$ are those occurring in the fixed declaration (10). The rule for recursive function declarations is:

$$\frac{\Gamma, \mathtt{f}{:}\sigma'\to\sigma, \mathtt{x}'{:}\sigma' \vdash e{:}\sigma \quad \Gamma \vdash \mathtt{e}'{:}\sigma'}{\Gamma \vdash \mathtt{letrec\ fx}' = \mathtt{e}{:}\sigma\ \mathtt{in\ fe}' : \sigma} \ .$$

We omit the other type assignment rules. (See (Gunter, 1992, Tables 2.1, 3.1 and 4.1) for example.)

Let $Expr_\sigma$ denote the collection of *closed expressions of type* σ, i.e. the expressions e for which the typing assertion $\emptyset \vdash \mathtt{e}{:}\sigma$ holds. Let Can_σ denote the subset of $Expr_\sigma$ consisting of the expressions in *canonical form*. These are given by:

$$\mathtt{c} ::= \mathtt{()} \mid \mathtt{true} \mid \mathtt{false} \mid \underline{n} \mid \mathtt{In}_i(\mathtt{c}) \mid \mathtt{(c,c)} \mid \lambda\mathtt{x}{:}\sigma.\mathtt{e} \ .$$

The operational semantics of the programming language can be given via a type-indexed family of *evaluation relations*

$$\mathtt{e} \Downarrow_\sigma \mathtt{c} \qquad (\mathtt{e} \in Expr_\sigma, \mathtt{c} \in Can_\sigma) \ .$$

These relations are inductively defined by the rules in Table 1, which follow the dynamic semantics of the corresponding Standard ML expressions, as specified in (Milner, Tofte and Harper, 1990).

Table 1: Rules for expression evaluation.

$$\frac{}{\mathtt{c} \Downarrow_\sigma \mathtt{c}}$$

$$\frac{\mathtt{b} \Downarrow_{bool} \mathtt{true} \qquad \mathtt{e} \Downarrow_\sigma \mathtt{c}}{(\mathtt{if\ b\ then\ e\ else\ e'}) \Downarrow_\sigma \mathtt{c}} \qquad \frac{\mathtt{b} \Downarrow_{bool} \mathtt{false} \qquad \mathtt{e'} \Downarrow_\sigma \mathtt{c'}}{(\mathtt{if\ b\ then\ e\ else\ e'}) \Downarrow_\sigma \mathtt{c'}}$$

$$\frac{\mathtt{e} \Downarrow_{int} \underline{n} \qquad \mathtt{e'} \Downarrow_{int} \underline{n'}}{\mathtt{op(e,e')} \Downarrow_\sigma \mathtt{c}} \quad \text{(where c names the value of } op(n,n') \text{ and } \sigma \text{ is its type)}$$

$$\frac{\mathtt{e} \Downarrow_{\sigma_i} \mathtt{c}}{\mathtt{In}_i\mathtt{(e)} \Downarrow_{ty} \mathtt{In}_i\mathtt{(c)}} \qquad \frac{\mathtt{e} \Downarrow_{ty} \mathtt{In}_i\mathtt{(c)} \qquad \mathtt{e}_i\mathtt{[c/x}_i\mathtt{]} \Downarrow_\sigma \mathtt{d}}{(\mathtt{case\ e\ of\ \{}\cdots\mathtt{|\,In}_i\mathtt{(x}_i\mathtt{)=>e}_i\mathtt{\,|}\cdots\mathtt{\}}) \Downarrow_\sigma \mathtt{d}}$$

$$\frac{\mathtt{e} \Downarrow_\sigma \mathtt{c} \qquad \mathtt{e'} \Downarrow_{\sigma'} \mathtt{c'}}{\mathtt{(e,e')} \Downarrow_{\sigma * \sigma'} \mathtt{(c,c')}} \qquad \frac{\mathtt{e} \Downarrow_{\sigma * \sigma'} \mathtt{(c,c')}}{\mathtt{fst(e)} \Downarrow_\sigma \mathtt{c}} \qquad \frac{\mathtt{e} \Downarrow_{\sigma * \sigma'} \mathtt{(c,c')}}{\mathtt{snd(e)} \Downarrow_\sigma \mathtt{c'}}$$

$$\frac{\mathtt{e} \Downarrow_{\sigma' \to \sigma} \lambda\mathtt{x'}{:}\sigma'.\mathtt{e''} \qquad \mathtt{e'} \Downarrow_{\sigma'} \mathtt{c'} \qquad \mathtt{e''[c'/x']} \Downarrow_\sigma \mathtt{c}}{\mathtt{ee'} \Downarrow_\sigma \mathtt{c}}$$

$$\frac{\mathtt{e'} \Downarrow_{\sigma'} \mathtt{c'} \qquad \mathtt{e}[(\lambda\mathtt{x}{:}\sigma'.\mathtt{letrec\ fx'} = \mathtt{e}{:}\sigma\ \mathtt{\ in\ fx})/\mathtt{f}, \mathtt{c'/x'}] \Downarrow_\sigma \mathtt{c}}{\mathtt{letrec\ fx'} = \mathtt{e}{:}\sigma\ \mathtt{\ in\ fe'} \Downarrow_\sigma \mathtt{c}}$$

Turning now to the denotational semantics of this language, for each type σ, let $F_\sigma(-,+) : \mathbf{Cpo}_\perp^{\mathrm{op}} \times \mathbf{Cpo}_\perp \longrightarrow \mathbf{Cpo}_\perp$ be the locally continuous functor defined by:

$$\begin{aligned}
F_{unit}(-,+) &\stackrel{def}{=} 1_\perp \\
F_{bool}(-,+) &\stackrel{def}{=} 2_\perp \\
F_{int}(-,+) &\stackrel{def}{=} \mathbb{Z}_\perp \\
F_{ty}(-,+) &\stackrel{def}{=} (+) \\
F_{\sigma * \sigma'}(-,+) &\stackrel{def}{=} F_\sigma(-,+) \otimes F_{\sigma'}(-,+) \\
F_{\sigma \to \sigma'}(-,+) &\stackrel{def}{=} (F_\sigma(+,-) \multimap F_{\sigma'}(-,+))_\perp \ .
\end{aligned}$$

Here $1_\perp$, $2_\perp$, and $\mathbb{Z}_\perp$ are the cppos obtained by lifting the discrete cpos $1 = \{0\}$, $2 = \{0,1\}$, and $\mathbb{Z} = \{\ldots, -1, 0, 1, \ldots\}$. Let $F(-,+) : \mathbf{Cpo}_\perp^{\mathrm{op}} \times \mathbf{Cpo}_\perp \longrightarrow \mathbf{Cpo}_\perp$ be the locally continuous functor

$$F(-,+) \stackrel{def}{=} F_{\sigma_1}(-,+) \oplus \cdots \oplus F_{\sigma_n}(-,+)$$

and let D be its minimal invariant, with associated isomorphism $i : F(D,D) \cong D$. Then we can define the *denotation* $[\![\sigma]\!]$ *of the type* σ to be

$$[\![\sigma]\!] \stackrel{def}{=} F_\sigma(D,D) \ .$$

In particular, i is an isomorphism from $[\![\sigma_1]\!] \oplus \cdots \oplus [\![\sigma_n]\!]$ to $[\![ty]\!]$.

Each closed expression $\mathtt{e} \in \mathit{Expr}_\sigma$ can be given a denotation $[\![\mathtt{e}]\!]$ which is an element of the cppo $[\![\sigma]\!]$. More generally, an open expression $\mathtt{e}$ for which the typing assertion (11) holds determines a strict continuous function

$$[\![\mathtt{e}]\!] : [\![\tau_1]\!] \otimes \cdots \otimes [\![\tau_m]\!] \circ\!\!\longrightarrow [\![\sigma]\!]$$

or, equivalently, a continuous function $[\![\tau_1]\!]_\downarrow \times \cdots \times [\![\tau_m]\!]_\downarrow \longrightarrow [\![\sigma]\!]$. These functions are defined by induction on the structure of $\mathtt{e}$ using the corresponding domain-theoretic operations. For example, for each $\vec{d} = (d_1, \ldots, d_m) \in [\![\tau_1]\!]_\downarrow \times \cdots \times [\![\tau_m]\!]_\downarrow$ one defines

$$[\![\mathtt{In}_i(\mathtt{e})]\!](\vec{d}) \stackrel{def}{=} (i \circ in_i \circ [\![\mathtt{e}]\!])(\vec{d})$$

$$[\![\mathtt{case\ e\ of\ \{} \cdots \mid \mathtt{In}_i(\mathtt{x}_i)\mathtt{=>e}_i \mid \cdots \mathtt{\}}]\!](\vec{d}) \stackrel{def}{=} [\![\mathtt{e}_i]\!](\vec{d}, d) \quad \text{where}$$
$$i^{-1}([\![\mathtt{e}]\!](\vec{d})) = in_i(d), \text{ for some}$$
$$i = 1, \ldots, n \text{ and } d \in [\![\sigma_i]\!]_\downarrow$$

where $in_i : [\![\sigma_i]\!] \circ\!\!\longrightarrow [\![\sigma_1]\!] \oplus \cdots \oplus [\![\sigma_n]\!]$ is the insertion function associated with the coalesced sum, as in Definition 3.1(iv). The clauses of the definition of $[\![\mathtt{e}]\!]$ are all quite standard and we omit them. (See (Winskel, 1993, Sect. 13.3) for example.)

Similarly, for each context $C[-_\tau]$ of type σ say, containing a 'hole' of type τ, one can define a continuous function $[\![C[-_\tau]]\!] : [\![\tau]\!] \longrightarrow [\![\sigma]\!]$ by induction on

the structure of $C[-_\tau]$. Here we take a *context* C to be an extended expression containing instances of the place-holder $-_\tau$, and which becomes a closed expression $C[\mathsf{e}] \in \mathit{Expr}_\sigma$ whenever a closed expression $\mathsf{e} \in \mathit{Expr}_\tau$ is substituted for the place-holder. (We are simplifying matters by only considering the notion of context, and contextual equivalence, for *closed* expressions.)

We wish to show that the denotational semantics is adequate for establishing that two expressions have the same observable behaviour under evaluation in all contexts. We will use a notion of observation based upon the form of a canonical expression of arbitrary type. The relation

$$\mathsf{c} \equiv_\sigma \mathsf{c}' \qquad (\mathsf{c}, \mathsf{c}' \in \mathit{Can}_\sigma)$$

of 'having the same form' is inductively defined by:

$$() \equiv_{unit} () \qquad \mathtt{true} \equiv_{bool} \mathtt{true} \qquad \mathtt{false} \equiv_{bool} \mathtt{false} \qquad \underline{n} \equiv_{int} \underline{n}$$

$$\frac{\mathsf{c} \equiv_{\sigma_i} \mathsf{c}'}{\mathtt{In}_i(\mathsf{c}) \equiv_{ty} \mathtt{In}_i(\mathsf{c}')} \qquad \frac{\mathsf{c}_1 \equiv_\sigma \mathsf{c}_2 \qquad \mathsf{c}_1' \equiv_{\sigma'} \mathsf{c}_2'}{(\mathsf{c}_1, \mathsf{c}_1') \equiv_{\sigma * \sigma'} (\mathsf{c}_2, \mathsf{c}_2')}$$

$$\lambda \mathsf{x}{:}\sigma.\mathsf{e} \equiv_{\sigma \to \sigma'} \lambda \mathsf{x}'{:}\sigma.\mathsf{e}' \ .$$

Given $\mathsf{e}_1, \mathsf{e}_2 \in \mathit{Expr}_\tau$, let us write $\mathsf{e}_1 \lesssim_\tau \mathsf{e}_2$ to mean that for all contexts $C[-_\tau]$ of type σ (any σ), and all $\mathsf{c}_1 \in \mathit{Can}_\sigma$:

$$C[\mathsf{e}_1] \Downarrow_\sigma \mathsf{c}_1 \Rightarrow \exists \mathsf{c}_2 \in \mathit{Can}_\sigma (C[\mathsf{e}_2] \Downarrow_\sigma \mathsf{c}_2 \quad \wedge \quad \mathsf{c}_1 \equiv_\sigma \mathsf{c}_2) \ .$$

Then we say that e_1 and e_2 are *contextually equivalent*, and write $\mathsf{e}_1 \approx_\tau \mathsf{e}_2$, if both $\mathsf{e}_1 \lesssim_\tau \mathsf{e}_2$ and $\mathsf{e}_2 \lesssim_\tau \mathsf{e}_1$ hold. (For this particular language, with its strict function application, the notion of contextual equivalence remains unchanged if in the definition we restrict contexts to be of type *bool*, or if we merely observe convergence.)

Proposition 5.1 (Computational adequacy) *For all types τ and all $\mathsf{e}_1, \mathsf{e}_2 \in \mathit{Expr}_\tau$, if $[\![\mathsf{e}_1]\!] = [\![\mathsf{e}_2]\!]$ then $\mathsf{e}_1 \approx_\tau \mathsf{e}_2$.*

This proposition can be deduced easily from the following properties of the denotational semantics:

$$[\![C[\mathsf{e}]]\!] = [\![C[-]]\!]([\![\mathsf{e}]\!]) \tag{12}$$

$$\forall \mathsf{c} \in \mathit{Can}_\sigma ([\![\mathsf{c}]\!] \neq \bot) \tag{13}$$

$$\forall \mathsf{c}_1, \mathsf{c}_2 \in \mathit{Can}_\sigma ([\![\mathsf{c}_1]\!] = [\![\mathsf{c}_2]\!] \Rightarrow \mathsf{c}_1 \equiv_\sigma \mathsf{c}_2) \tag{14}$$

$$\forall \mathsf{e} \in \mathit{Expr}_\sigma, \mathsf{c} \in \mathit{Can}_\sigma (\mathsf{e} \Downarrow_\sigma \mathsf{c} \Rightarrow [\![\mathsf{e}]\!] = [\![\mathsf{c}]\!]) \tag{15}$$

$$\forall \mathsf{e} \in \mathit{Expr}_\sigma([\![\mathsf{e}]\!] \neq \bot \Rightarrow \exists \mathsf{c} \in \mathit{Can}_\sigma.\mathsf{e} \Downarrow_\sigma \mathsf{c}) \ . \tag{16}$$

Suppose $[\![\mathsf{e}_1]\!] = [\![\mathsf{e}_2]\!]$. If $C[\mathsf{e}_1] \Downarrow_\sigma \mathsf{c}_1$, then using (12), (15), and (13) one has

$$[\![C[\mathsf{e}_2]]\!] = [\![C[-_\tau]]\!]([\![\mathsf{e}_2]\!]) = [\![C[-_\tau]]\!]([\![\mathsf{e}_1]\!]) = [\![C[\mathsf{e}_1]]\!] = [\![\mathsf{c}_1]\!] \neq \bot$$

so that by (16) there is some c_2 such that $C[\mathsf{e}_2] \Downarrow_\sigma \mathsf{c}_2$, and hence also with $[\![\mathsf{c}_1]\!] = [\![\mathsf{c}_2]\!]$ (by (15) again); then $\mathsf{c}_1 \equiv_\sigma \mathsf{c}_2$ by (14). Thus $\mathsf{e}_1 \lesssim_\tau \mathsf{e}_2$; and $\mathsf{e}_2 \lesssim_\tau \mathsf{e}_1$ holds by a symmetrical argument. Therefore (12)–(16) do indeed imply Proposition 5.1.

Now (12) is the basic property of *compositionality* of the denotational semantics and can be established from its definition by induction on the structure of $C[-]$. Properties (13) and (14) are simple to prove by induction on the structure of the canonical form c. Property (15) is often called the *soundness* of the denotational semantics with respect to the operational semantics and can be proved by checking that the relation $[\![\mathsf{e}]\!] = [\![\mathsf{c}]\!]$ is closed under the rules in Table 1 defining the evaluation relation.

So it only remains to prove (16). Adapting the method in (Plotkin, 1985), this can be done by constructing a type-indexed family of 'formal approximation' relations

$$\lhd_\sigma \subseteq [\![\sigma]\!]_\downarrow \times \mathit{Can}_\sigma$$

with the following properties:

$$\forall \mathsf{c} \in \mathit{Can}_\sigma(\{d \mid d \lhd_\sigma \mathsf{c}\} \text{ is chain-complete}) \tag{17}$$

$$d \lhd_{unit} () \Leftrightarrow d = 0 \tag{18}$$

$$\begin{aligned} d \lhd_{bool} \mathtt{false} &\Leftrightarrow d = 0 \\ d \lhd_{bool} \mathtt{true} &\Leftrightarrow d = 1 \end{aligned} \tag{19}$$

$$d \lhd_{int} \underline{n} \Leftrightarrow d = n \tag{20}$$

$$d \lhd_{ty} \mathtt{In}_i(\mathsf{c}) \Leftrightarrow \exists d_i \in [\![\sigma_i]\!]_\downarrow (d = (i \circ in_i)(d_i) \wedge d_i \lhd_{\sigma_i} \mathsf{c}) \tag{21}$$

$$(d, d') \lhd_{\sigma * \sigma'} (\mathsf{c}, \mathsf{c}') \Leftrightarrow (d \lhd_\sigma \mathsf{c} \wedge d' \lhd_{\sigma'} \mathsf{c}') \tag{22}$$

$$f \lhd_{\sigma \to \sigma'} \lambda \mathtt{x}{:}\sigma.\mathsf{e} \Leftrightarrow \forall d, \mathsf{c}(d \lhd_\sigma \mathsf{c} \Rightarrow f(d)(\lhd_{\sigma'})_\bot \mathsf{e}[\mathsf{c}/\mathtt{x}]) \tag{23}$$

where the auxiliary relation $(\lhd_\sigma)_\perp \subseteq [\![\sigma]\!] \times \mathit{Expr}_\sigma$ is defined from $\lhd_\sigma$ by

$$d(\lhd_\sigma)_\perp \mathsf{e} \Leftrightarrow (d \neq \perp \Rightarrow \exists \mathsf{c}(\mathsf{e} \Downarrow_\sigma \mathsf{c} \wedge d \lhd_\sigma \mathsf{c})) \ . \tag{24}$$

Suppose for the moment that such a family of relations $\lhd_\sigma$ exists. For any valid typing assertion (11), properties (17)–(23) imply that

$$d_1 \lhd_{\tau_1} \mathsf{c}_1 \wedge \cdots \wedge d_m \lhd_{\tau_m} \mathsf{c}_m \Rightarrow [\![\mathsf{e}]\!](\vec{d})(\lhd_\sigma)_\perp \mathsf{e}[\vec{\mathsf{c}}/\vec{\mathsf{x}}] \ . \tag{25}$$

The proof of this proceeds by induction on the derivation of the typing assertion. (The induction step in the case when e is $\mathtt{letrec\ fx} = \mathsf{e}'{:}\sigma\ \mathtt{in\ fe}''$ requires fixed point induction, relying upon the fact that each $\{d \mid d(\lhd_\sigma)_\perp \mathsf{e}\}$ is chain-complete and contains $\perp$, by (17) and (24).)

The $m = 0$ case of (25) implies that for any closed expression $\mathsf{e} \in \mathit{Expr}_\sigma$ we have $[\![\mathsf{e}]\!](\lhd_\sigma)_\perp \mathsf{e}$. Hence if $[\![\mathsf{e}]\!] \neq \perp$, (24) implies that $\mathsf{e} \Downarrow_\sigma \mathsf{c}$ for some c. Thus (16) and hence also Proposition 5.1 do indeed follow from the existence of relations $\lhd_\sigma$ satisfying (17)–(23).

The existence of such relations is not straightforward. Because of clause (21), one cannot simply define $\lhd_\sigma$ by induction on the structure of σ. And one cannot use the right-to-left implications in (17)–(23) as the clauses of a simultaneous inductive definition of the relations $\lhd_\sigma$, since the clause corresponding to (23) contains a negative occurrence of the relation being defined.

In the literature, two methods can be found for constructing relations on recursively defined domains with such non-monotonic fixed point properties. One method, due to Milne, Plotkin and Reynolds, makes use of Scott's construction of a recursively defined cppo as the colimit of a chain of embeddings $D_0 \longrightarrow D_1 \longrightarrow \cdots$, where the cppo D_n is obtained by iterating the domain constructor n times, starting with the trivial cppo $\{\perp\}$ (cf. the proof of Theorem 3.3). Then the desired relation can be constructed as an intersection of countably many relations defined by induction on n at the same time as the D_n are defined; see (Reynolds, 1974). A second method, essentially due to Martin-Löf, applies only to Scott domains (precluding the use of constructors like the Plotkin powerdomain) since it makes use of their presentation in terms of 'information systems' (Scott, 1982). This method hinges upon the fact that each $\{d \mid d(\lhd_\sigma)_\perp \mathsf{e}\}$ is in fact a Scott-closed subset of $[\![\sigma]\!]$. Hence it suffices to construct the relations $\lhd_\sigma$ only for compact elements of $[\![\sigma]\!]$, since $d(\lhd_\sigma)_\perp \mathsf{e}$ holds if and only if $a(\lhd_\sigma)_\perp \mathsf{e}$ holds for all compact a with $a \sqsubseteq d$. Information systems provide a formal language for compact elements of (recursively defined) Scott domains, and $a(\lhd_\sigma)_\perp \mathsf{e}$ (a compact) can be defined by a well-founded induction on the size of (a formal representation of) a; see (Winskel, 1993, Sect. 13.4).

Here we show that the existence of relations $\lhd_\sigma$ satisfying (17)–(23) follows directly from Theorem 4.16 by choosing a suitable relational structure $\mathcal{R}$ on $\mathbf{Cpo}_\perp$ and defining admissible actions of the appropriate domain constructors. This provides a new method of construction that is more abstract than the above two, in that it relies upon the 'minimal invariant' property of the recursively defined cppo $[\![ty]\!]$ rather than upon the techniques for giving concrete constructions of recursively defined domains.

Let $\mathcal{R}$ be the relational structure of Example 4.2(iii) with the set I equal to

$$\mathit{Can} \stackrel{def}{=} \{\mathsf{c}{:}\sigma \mid \mathsf{c} \in \mathit{Can}_\sigma\}$$

the set of all canonical forms, tagged with their types. Thus for each cppo D, $\mathcal{R}(D)$ consists of all subsets of $D_\downarrow \times \mathit{Can}$; and the relation $f : R \subset S$ holds if and only if

$$\forall (d, \mathtt{c}{:}\sigma) \in R (f(d) \neq \bot \Rightarrow (f(d), \mathtt{c}{:}\sigma) \in S) \ .$$

From Example 4.5 we have that $R \in \mathcal{R}(D)$ is admissible if and only if for each σ and each $\mathtt{c} \in \mathit{Can}_\sigma$, $\{d \mid (d, \mathtt{c}{:}\sigma) \in R\}$ is a chain-complete subset of $D_\downarrow$.

The locally continuous functors F_σ and F used above to give the denotation of types can be endowed with admissible actions on $\mathcal{R}$-relations. Given any D, E, $R \in \mathcal{R}(D)$ and $S \in \mathcal{R}(E)$, we define $F_\sigma(R, S) \in \mathcal{R}(F_\sigma(D, E))$ by induction on the structure of σ as follows:

$$F_{\mathit{unit}}(R, S) = \{(0, ()\!:\!\mathit{unit})\} \tag{26}$$

$$F_{\mathit{bool}}(R, S) = \{(0, \mathtt{false}\!:\!\mathit{bool}), (1, \mathtt{true}\!:\!\mathit{bool})\} \tag{27}$$

$$F_{\mathit{int}}(R, S) = \{(n, \underline{n}\!:\!\mathit{int}) \mid n \in \mathbb{Z}\} \tag{28}$$

$$F_{\mathit{ty}}(R, S) = S \tag{29}$$

$$\begin{aligned} F_{\sigma * \sigma'}(R, S) = \{&((d, d'), (\mathtt{c}, \mathtt{c}')\!:\!\sigma * \sigma') \mid \\ &(d, \mathtt{c}{:}\sigma) \in F_\sigma(R, S) \wedge (d', \mathtt{c}'{:}\sigma') \in F_{\sigma'}(R, S)\} \end{aligned} \tag{30}$$

$$\begin{aligned} F_{\sigma \to \sigma'}(R, S) = \{&(f, \lambda \mathtt{x}{:}\sigma.\mathtt{e}{:}\sigma \to \sigma') \mid \\ &\forall (d, \mathtt{c}{:}\sigma) \in F_\sigma(S, R) (f(d) \neq \bot \Rightarrow \exists \mathtt{c}' \in \mathit{Can}_{\sigma'} \\ &\mathtt{e}[\mathtt{c}/\mathtt{x}] \Downarrow_{\sigma'} \mathtt{c}' \wedge (f(d), \mathtt{c}'{:}\sigma') \in F_{\sigma'}(R, S))\} \ . \end{aligned} \tag{31}$$

Then the action of the functor $F = F_{\sigma_1} \oplus \cdots \oplus F_{\sigma_n}$ is defined by:

$$F(R, S) = \{(\mathit{in}_i(d), \mathtt{In}_i(\mathtt{c})\!:\!\mathit{ty}) \mid i = 1, \ldots, n \wedge (d, \mathtt{c}{:}\sigma_i) \in F_{\sigma_i}(R, S)\} \ .$$

It is not hard to verify (by induction on the structure of σ) that these definitions satisfy the requirements of Definition 4.6 for admissible actions of functors $\mathbf{Cpo}_\bot^{\mathrm{op}} \times \mathbf{Cpo}_\bot \longrightarrow \mathbf{Cpo}_\bot$. (The determinacy of the evaluation relation $- \Downarrow_\sigma -$ is needed at the induction step for a function type when proving that $F_\sigma(R, S)$ is admissible when S is.)

Recall that by definition, $[\![\mathit{ty}]\!]$ is the minimal invariant cppo for the functor F. We noted in Example 4.13 that $\mathcal{R}$ has inverse images and intersections. So we can apply Theorem 4.16 to deduce that F possesses an invariant $\mathcal{R}$-relation $\Delta \in \mathcal{R}([\![\mathit{ty}]\!])$. For each type σ define

$$\lhd_\sigma \stackrel{def}{=} \{(d, \mathtt{c}) \mid (d, \mathtt{c}{:}\sigma) \in F_\sigma(\Delta, \Delta)\} \ .$$

We claim that this is the family of 'formal approximation' relations required in the proof of computational adequacy. Since $[\![\sigma]\!] = F_\sigma(D, D)$, it is certainly the case that $\lhd_\sigma$ is a subset of $[\![\sigma]\!]_\downarrow \times \mathit{Can}_\sigma$. Since Δ is admissible by definition and because the action of each F_σ on $\mathcal{R}$-relations satisfies clause (i) of Definition 4.6, $\lhd_\sigma$ satisfies property (17). Properties (18)–(20), (22), and (23) follow immediately from the corresponding clauses (26)–(28), (30), and (31) of the definition of the action of the functors F_σ on $\mathcal{R}$-relations. Finally, property (21) also follows from the corresponding clause (29) together with the defining property of the invariant $\mathcal{R}$-relation Δ, namely that $i : F(\Delta, \Delta) \subset \Delta$ and $i^{-1} : \Delta \subset F(\Delta, \Delta)$.

Remark 5.2 It should be clear how to extend the proof of computational adequacy we have given here to the case of *simultaneous* recursive datatype declarations. To treat the case of *local* datatype declarations, one has to extend the approach developed in Sects 3 and 4 to the case of domain equations with extra parameters. This is considered briefly in Sect. 7. Rather than deducing adequacy results from a general theory of relational properties, one can also apply the key techniques in our proof on an *ad hoc* basis. (Pitts, 1993a) illustrates this direct approach with respect to a simple language based upon the untyped lambda calculus with arithmetic operations.

6 Induction and co-induction principles

Many kinds of relational structure on $\mathbf{Cpo}_\perp$ possess distinguished 'identity' relations. The following definition axiomatizes the properties we require of such identity relations (cf. (O'Hearn and Tennent, 1993, Sect. 2)).

Definition 6.1 A *unitary relational structure*, $\mathcal{R}$, on a category $\mathbf{C}$ is a relational structure in the sense of Definition 4.1 together with a family of $\mathcal{R}$-relations $I_D \in \mathcal{R}(D)$ (for each $\mathbf{C}$-object D) satisfying

$$f : I_D \subset I_E, \qquad \text{for all } f : D \longrightarrow E. \tag{32}$$

We call I_D the 'identity $\mathcal{R}$-relation on D'.

Example 6.2 We can make the relational structure 4.2(i) unitary by defining I_D to be $\{(x, \ldots, x) \mid x \neq \perp\} \cup \{\perp\}$. Example (ii) in 4.2 could be made unitary by defining I_D to be $\{(x_1, \ldots, x_n) \mid x_1 \sqsubseteq x_2 \sqsubseteq \cdots \sqsubseteq x_n\}$. Another natural candidate for the identity relation on D in this case would be $I_D = \{(x_1, \ldots, x_n) \mid x_1 = x_2 = \cdots = x_n\}$.

Condition (32) is not much of a constraint upon the family $(I_D \mid D \in \mathbf{C})$ and as the last example illustrates, there may be several different ways to enrich a relational structure with identity relations. However, we will be interested in actions of functors on relations which preserve identity relations and this further constrains matters.

Definition 6.3 Let $\mathcal{R}$ be a unitary relational structure on $\mathbf{Cpo}_\perp$ and let

$$F : (\mathbf{Cpo}_\perp^{\mathrm{op}})^m \times (\mathbf{Cpo}_\perp)^n \longrightarrow \mathbf{Cpo}_\perp$$

be a functor (where $m, n \geq 0$) equipped with an admissible action on $\mathcal{R}$-relations, as in Definition 4.6. We say that this action is *unitary* provided for all cppos $D_1, \ldots, D_m, E_1, \ldots, E_n$

$$F(I_{D_1}, \ldots, I_{D_m}, I_{E_1}, \ldots, I_{E_n}) = I_{F(D_1, \ldots, D_m, E_1, \ldots, E_n)} \ . \tag{33}$$

Let $F : \mathbf{Cpo}_\perp^{\mathrm{op}} \times \mathbf{Cpo}_\perp \longrightarrow \mathbf{Cpo}_\perp$ be as in the above definition. Suppose also that F is locally continuous and hence possesses a minimal invariant cppo D, with associated isomorphism $i : F(D, D) \cong D$. Then from (32) and (33) we have

$$\begin{aligned} i^{-1} &: I_D \subset I_{F(D,D)} = F(I_D, I_D) \\ i &: F(I_D, I_D) = I_{F(D,D)} \subset I_D \ . \end{aligned}$$

So *provided I_D is admissible*, it has the defining properties of the invariant $\mathcal{R}$-relation (cf. Definition 4.8) and so Corollary 4.10 holds with $\Delta = I_D$. We have proved the following result.

Proposition 6.4 *Suppose $\mathcal{R}$ is a unitary relational structure on $\mathbf{Cpo}_\perp$ whose identity relations are admissible. Let $F : \mathbf{Cpo}_\perp^{op} \times \mathbf{Cpo}_\perp \longrightarrow \mathbf{Cpo}_\perp$ be a locally continuous functor equipped with a unitary admissible action on $\mathcal{R}$-relations. Then the minimal invariant cppo for F, $i : F(D,D) \cong D$, satisfies the following rule for all $R^-, R^+ \in \mathcal{R}(D)$:*

$$\frac{i^{-1} : R^- \subset F(R^+, R^-) \quad i : F(R^-, R^+) \subset R^+}{R^- \subset I_D \subset R^+} \quad (R^+ \text{ admissible}). \tag{34}$$

We will show how this rule gives rise to families of induction and co-induction principles for recursively defined cppos.

Induction

Let $\mathcal{P}$ be the relational structure on $\mathbf{Cpo}_\perp$ of Example 4.2(i) in case $n = 1$, with identity relations given as in Example 6.2. Thus for each cppo D, $\mathcal{P}(D)$ consists of all subsets of D that contain $\perp$; the relation $f : R \subset S$ holds just in case f maps R into S; and the identity $\mathcal{P}$-relation I_D is the whole of D. From Example 4.5 we have that $R \in \mathcal{P}(D)$ is admissible if and only if it is a chain-complete subset of D (containing $\perp$, as must any $\mathcal{P}$-relation). Note in particular that each identity relation is admissible.

Theorem 6.5 (Induction property of minimal invariant cppos) *Suppose $F : \mathbf{Cpo}_\perp^{op} \times \mathbf{Cpo}_\perp \longrightarrow \mathbf{Cpo}_\perp$ is a locally continuous functor and let D be its minimal invariant cppo, with associated isomorphism $i : F(D,D) \cong D$. With $\mathcal{P}$ as above, suppose that F possesses a unitary admissible action on $\mathcal{P}$-relations satisfying for all $R \in \mathcal{P}(D)$ that $F(R, I_D) = I_{F(D,D)}$. Then D satisfies the following rule of induction for any chain-complete subset $R \subseteq D$ containing $\perp$.*

$$\frac{\forall u \in F(D,D)(u \in F(I_D, R) \Rightarrow i(u) \in R)}{\forall d \in D.d \in R} \ .$$

Proof Since identity $\mathcal{P}$-relations are admissible, we can apply Proposition 6.4 to conclude that D satisfies rule (34). Note that since I_D is the whole of D, $R^- \subset I_D$ holds for any R^-. So the first half of the conclusion of (34) tells us nothing in this case. This suggests that we take $R^- = I_D$. Then second half of the conclusion tells us that the subset R^+ is the whole of D. Moreover since by assumption $F(R^-, I_D) = I_{F(D,D)}$, the first half of the hypothesis on R^- and R^+, namely that $i^{-1} : R^- \subset F(R^+, R^-)$, is automatically satisfied when $R^- = I_D$. So all in all, we have that in case $R^- = I_D, R^+ = R$, (34) reduces to the desired induction rule. □

We will show that this theorem applies to a wide range of recursively defined cppos. For $R \in \mathcal{P}(D)$ and $S \in \mathcal{P}(E)$ define $R_\perp \in \mathcal{P}(D_\perp)$, $R \times S \in \mathcal{P}(D \times E)$, $R \otimes S \in \mathcal{P}(D \otimes E)$, $R \oplus S \in \mathcal{P}(D \oplus E)$, $R \to S \in \mathcal{P}(D \to E)$, and $R \multimap S \in$

$\mathcal{P}(D \multimap E)$, as follows:

$$\begin{aligned}
R_\perp &\stackrel{def}{=} R \cup \{\perp\} \\
R \times S &\stackrel{def}{=} \{(d,e) \mid d \in R \wedge e \in S\} \\
R \otimes S &\stackrel{def}{=} \{(d,e) \mid \perp \neq d \in R \wedge \perp \neq e \in S\} \cup \{\perp\} \\
R \oplus S &\stackrel{def}{=} \{in_1(d) \mid \perp \neq d \in R\} \cup \{in_2(e) \mid \perp \neq y \in S\} \cup \{\perp\} \\
R \to S &\stackrel{def}{=} \{f \mid \forall d \in R. f(d) \in S\} \\
R \multimap S &\stackrel{def}{=} (R \to S) \cap (D \multimap E) \ .
\end{aligned}$$

Lemma 6.6 *(i) If $f : R \subset R'$ and $g : S \subset S'$, then $f_\perp : R_\perp \subset S_\perp$, $f \times g : (R \times S) \subset (R' \times S')$, $f \otimes g : (R \otimes S) \subset (R' \otimes S')$, $f \oplus g : (R \oplus S) \subset (R' \oplus S')$, $f \to g : (R' \to S) \subset (R \to S')$, and $f \multimap g : (R' \multimap S) \subset (R \multimap S')$.*

(ii) $(I_D)_\perp = I_{D_\perp}$, $I_D \times I_E = I_{D \times E}$, $I_D \otimes I_E = I_{D \otimes E}$, $I_D \oplus I_E = I_{D \oplus E}$, $I_D \to I_E = I_{D \to E}$, and $I_D \multimap I_E = I_{D \multimap E}$.

(iii) Let $\Phi(\alpha)$ be a cppo-constructor built up from the variable α and constants ranging over cppos, using $(-)_\perp$, $\times$, $\otimes$, $\oplus$, $\to$, and $\multimap$. Then the locally continuous functor $\hat{\Phi} : \mathbf{Cpo}_\perp^{op} \times \mathbf{Cpo}_\perp \longrightarrow \mathbf{Cpo}_\perp$ associated with Φ as in Sect. 3, possesses an admissible action on $\mathcal{P}$-relations. (For this relational structure, property (ii) of Definition 4.6 holds without the restriction concerning admissibility.)

(iv) With Φ as in (iii), the action of $\hat{\Phi}$ on $\mathcal{P}$-relations satisfies $\hat{\Phi}(R, I_E) = I_{\hat{\Phi}(D,E)}$, for all $R \in \mathcal{P}(D)$. In particular, the action of $\hat{\Phi}$ on $\mathcal{P}$-relations is unitary.

Proof Properties (i) and (ii) are easily verified, and then (iii) follows from them by induction on the structure of Φ; in the inductive definition of $R, S \mapsto \hat{\Phi}(R,S)$, in case Φ is α we take $\hat{\Phi}(R,S) = S$, and in case Φ is a constant K we take $\hat{\Phi}(R,S) = I_K$. For (iv), first note that $R \to I_E = I_{D \to E}$ and $R \multimap I_E = I_{D \multimap E}$; these facts together with (ii) gives the desired result by induction on the structure of Φ. □

Corollary 6.7 *Let $\Phi(\alpha)$ be a cppo-constructor built up from the variable α and constants K ranging over cppos, using $(-)_\perp$, $\times$, $\otimes$, $\oplus$, $\to$, and $\multimap$. Let $D = rec\alpha.\Phi(\alpha)$ be the cppo recursively defined by $\alpha = \Phi(\alpha)$, with associated isomorphism $i : \Phi(D) \cong D$. Then D satisfies the following rule of induction for any chain-complete subset $R \subseteq D$ containing $\perp$*

$$\frac{\forall u \in \Phi(D)(u \in \Phi(R) \Rightarrow i(u) \in R)}{\forall d \in D. d \in R}$$

where $\Phi(R) \subseteq \Phi(D)$ is defined by induction on the structure of Φ as follows, using the operations on subsets defined before Lemma 6.6.

$$\begin{aligned}
\alpha(R) &\stackrel{def}{=} R \\
K(R) &\stackrel{def}{=} I_K \\
\Phi_\perp(R) &\stackrel{def}{=} \Phi(R) \cup \{\perp\} \\
(\Phi_1 \times \Phi_2)(R) &\stackrel{def}{=} \Phi_1(R) \times \Phi_2(R) \\
(\Phi_1 \otimes \Phi_2)(R) &\stackrel{def}{=} \Phi_1(R) \otimes \Phi_2(R) \\
(\Phi_1 \oplus \Phi_2)(R) &\stackrel{def}{=} \Phi_1(R) \oplus \Phi_2(R) \\
(\Phi_1 \to \Phi_2)(R) &\stackrel{def}{=} I_{\Phi_1(D)} \to \Phi_2(R) \\
(\Phi_1 \multimap \Phi_2)(R) &\stackrel{def}{=} I_{\Phi_1(D)} \multimap \Phi_2(R) \ .
\end{aligned}$$

Proof Lemma 6.6 implies that Theorem 6.5 holds for the functor $F = \hat{\Phi}$, whose minimal invariant is $D = rec\alpha.\Phi(\alpha)$. It is easy to verify that $\hat{\Phi}(I_D, R)$ is precisely the subset $\Phi(R)$ defined above. So we have the stated induction rule for $rec\alpha.\Phi(\alpha)$. □

Example 6.8 (Lazy Lambda Calculus) Let $\Phi(\alpha) = (\alpha \to \alpha)_\perp$. In this case $D = rec\alpha.\Phi(\alpha)$ is the canonical domain model of the lazy lambda calculus studied in (Abramsky, 1990) and (Abramsky and Ong, 1993). In this case, for each $R \in \mathcal{P}(D)$, $\Phi(R)$ is the subset of $(D \to D)_\perp$ containing $\perp$ and all elements of the form f where $f \in (D \to D)$ satisfies that $f(d) \in R$ for all $d \in D$. Thus for each $d \in D$, $i^{-1}(d) \in \Phi(R)$ if and only if $\forall x \in D(d \cdot x \in R)$ (where $d \cdot x$ is the application defined in Example 3.6). Note that since i is a bijection, $i : \Phi(R) \subset R$ holds if and only if for all $d \in D$, $i^{-1}(d) \in \Phi(R)$ implies $d \in R$. So Corollary 6.7 yields the following induction principle for chain-complete, $\perp$-containing subsets $R \subseteq D$ of the canonical domain model of the lazy lambda calculus:

$$\frac{\forall d(\forall x(d \cdot x \in R) \Rightarrow d \in R)}{\forall d(d \in R)}$$

This principle looks like a kind of (impredicative!) 'structural induction' for D, in which a 'function' $d \in D$ is decomposed into its vector of values $(d \cdot x \mid x \in D)$.

Example 6.9 (Fixpoint induction) When $\Phi(\alpha) = \alpha_\perp$, $D = rec\alpha.\Phi(\alpha)$ is the cpo

$$\{0 \sqsubseteq 1 \sqsubseteq 2 \sqsubseteq \cdots \sqsubseteq \top\} \ .$$

This is a *fixpoint object* for the strong monad $(-)_\perp$ on the category of cpos, in the sense of Crole and Pitts (1992). In this case Corollary 6.7 is the induction principle which motivated the one for fixpoint objects of strong monads in general, studied in *loc. cit.* Scott's principle of induction for admissible properties of least fixed points of continuous functions is a formal consequence of this instance of the theorem.

Example 6.10 (Structural induction) Suppose $\Phi(\alpha)$ is built up from α using only *flat* cppos, $\otimes$, and $\oplus$. Then $rec\alpha.\Phi(\alpha)$ is also a flat cppo, i.e. is

of the form $X_\perp$ for some discrete cpo X. Indeed, X is the initial algebra for an appropriate endofunctor on the category of sets and functions—the functor being built up using the set operations of cartesian product (corresponding to $\otimes$) and disjoint union (corresponding to $\oplus$) according to the structure of Φ. In this case the induction principle of Corollary 6.7 is an instance of the principle of *initial algebra induction* studied by Lehmann and Smyth (19, Sect. 5.2). As is well known, for various choices of such Φ, X yields inductively defined sets of numbers, lists, trees, etc., and initial algebra induction coincides with the corresponding principle of structural induction.

Remark 6.11 Jensen (1981) also derives a family of induction principles for recursively defined domains starting from their minimal invariant property. Corollary 6.7 differs from Jensen's induction property in a couple of respects. First, although the proof of 6.7 ultimately depends on properties of continuous endofunctions on the recursively defined domain, the induction hypothesis is stated purely in terms of subsets of the domain, whereas Jensen's principle in general contains hypotheses still involving endofunctions. Secondly, for the problematic case of a constructor $\Phi(\alpha)$ involving negative occurrences of α, such as Example 6.8, Jensen's principle can be vacuous (in the sense that the conclusion occurs as part of the induction hypothesis) when 6.7 is not. Nevertheless, 6.7 does not yield useful information about $rec\alpha.\Phi(\alpha)$ in all cases. For example, when $\Phi(\alpha)$ is $\alpha \to K$ (with K some fixed cppo) then $\Phi(R) = I_{D \to K}$; so the hypothesis $i : \Phi(R) \subset R$ is just $i : I_{D \to K} \subset R$, which is the same as $R = D$, since i is an isomorphism. So the induction principle is vacuous in this case.

Co-induction

Let $\mathcal{S}$ be the relational structure on $\mathbf{Cpo}_\perp$ of Example 4.2(ii) in case $n = 2$. Thus for each cppo D, $\mathcal{S}(D)$ consists of all subsets of $D \times D$; and the relation $f : R \subset S$ holds if and only if for all $(d_1, d_2) \in R$, $(f(d_1), f(d_2)) \in S$. Endow $\mathcal{S}$ with identity relations as in Example 6.2: thus I_D is the partial order relation $\sqsubseteq_D$. From Example 4.5 we have that $R \in \mathcal{S}(D)$ is admissible if and only if it is a chain-complete subset of $D \times D$ containing $(\perp, \perp)$. Note in particular that each I_D is admissible.

Theorem 6.12 (Co-induction property of minimal invariant cppos) *Suppose $F : \mathbf{Cpo}_\perp^{op} \times \mathbf{Cpo}_\perp \longrightarrow \mathbf{Cpo}_\perp$ is a locally continuous functor and let D be its minimal invariant cppo, with associated isomorphism $i : F(D, D) \cong D$. With $\mathcal{S}$ as above, suppose that F possesses a unitary admissible action on $\mathcal{S}$-relations satisfying for all $R \in \mathcal{S}(D)$ that $F(R, I_D) = I_{F(D,D)}$. Call $R \subseteq D \times D$ an* F-simulation *if it satisfies*

$$\forall (d, d') \in R.(i^{-1}(d), i^{-1}(d')) \in F(I_D, R) \tag{35}$$

Then for any $d, d' \in D$, $d \sqsubseteq_D d'$ if and only if $(d, d') \in R$ for some F-simulation R.

Proof Since identity $\mathcal{S}$-relations are admissible, we can apply Proposition 6.4 to conclude that D satisfies rule (34). Consider the special case when the admissible $\mathcal{S}$-relation R^+ is just I_D. Then second part of the conclusion of the

rule (*viz.* $I_D \subset R^+$) is automatically satisfied. The same is true for the second half of the hypothesis (*viz.* $i : F(R^-, R^+) \subset R^+$), since by assumption $F(R^-, I_D) = I_{F(D,D)}$. Thus taking $R^- = R$ and $R^+ = I_D$ in (34), we have

$$i^{-1} : R \subset F(I_D, R) \Rightarrow R \subset I_D \ . \tag{36}$$

But $i^{-1} : R \subset F(I_D, R)$ holds just in case R is an F-simulation; and $I_D = \sqsubseteq_D$. So (36) says that any F-simulation is contained in the partial order relation. This establishes the 'if' direction of the theorem. For the 'only if' direction, just note that since $F(I_D, I_D) = I_{F(D,D)}$ and i^{-1} is monotone, $\sqsubseteq_D$ is itself an F-simulation. □

To apply this theorem we need to produce suitable actions of cppo constructors on $\mathcal{S}$-relations. For $R \in \mathcal{S}(D)$ and $S \in \mathcal{S}(E)$ define $R_\perp \in \mathcal{S}(D_\perp)$, $R \times S \in \mathcal{S}(D \times E)$, $R \otimes S \in \mathcal{S}(D \otimes E)$, $R \oplus S \in \mathcal{S}(D \oplus E)$, $R \to S \in \mathcal{S}(D \to E)$, and $R \multimap S \in \mathcal{S}(D \multimap E)$, as follows:

$$\begin{aligned}
(d, d') \in R_\perp &\Leftrightarrow d \neq \perp \Rightarrow (d' \neq \perp \wedge (d, d') \in R) \\
((d, e), (d', e')) \in R \times S &\Leftrightarrow (d, d') \in R \wedge (e, e') \in S \\
(u, u') \in R \otimes S &\Leftrightarrow \forall d \in D_\downarrow, e \in E_\downarrow (u = (d, e) \Rightarrow \\
&\qquad \exists d' \in D_\downarrow, e' \in E_\downarrow ((d, d') \in R \wedge (e, e') \in S)) \\
(u, u') \in R \oplus S &\Leftrightarrow \forall d \in D_\downarrow (u = in_1(d) \Rightarrow \\
&\qquad \exists d' \in D_\downarrow (u' = in_1(d') \wedge (d, d') \in R)) \\
&\quad \wedge \\
&\quad \forall e \in E_\downarrow (u = in_2(e) \Rightarrow \\
&\qquad \exists e' \in E_\downarrow (u' = in_2(e') \wedge (e, e') \in S)) \\
(f, f') \in R \to S &\Leftrightarrow \forall d \in D. (f(d), f'(d)) \in S \\
(f, f') \in R \multimap S &\Leftrightarrow \forall d \in D_\downarrow. (f(d), f'(d)) \in S \ .
\end{aligned}$$

It is easy to verify that these constructions on $\mathcal{S}$-relations have all the properties stated in Lemma 6.6, except that for

$$f : R \subset R' \wedge g : S \subset S' \Rightarrow f \multimap g : (R' \multimap S) \subset (R \multimap S')$$

we need to assume $(\perp, \perp) \in S$. Since this is certainly the case when S is admissible, one does indeed get admissible actions on $\mathcal{S}$ for the functors $\hat{\Phi}$, as in part (iii) of the lemma. Note that the definitions of $R \to S$ and $R \multimap S$, 'throw away' the relation R (cf. the discussion in Example 4.7). This ensures that these constructions satisfy $R \to I_E = I_{D \to E}$ and $R \multimap I_E = I_{D \multimap E}$ and hence that part (iv) of the lemma holds. So if $\Phi(\alpha)$ is a cppo constructor built up from the variable α and constants K ranging over cppos, using $(-)_\perp$, $\times$, $\otimes$, $\oplus$, $\to$, and $\multimap$, then the associated locally continuous functor $F = \hat{\Phi} : \mathbf{Cpo}_\perp^{\mathrm{op}} \times \mathbf{Cpo}_\perp \longrightarrow \mathbf{Cpo}_\perp$ has a unitary admissible action on $\mathcal{S}$-relations. Therefore its minimal invariant $D = rec\,\alpha.\Phi(\alpha)$ satisfies the Theorem 6.12.

Writing $\Phi(R)$ for $\hat{\Phi}(I_D, R)$, we can use the definitions above to give a direct description of this subset of $\Phi(D) \times \Phi(D)$ by induction on the structure of Φ

(and hence explain when a relation $R \subseteq D \times D$ is a $\hat{\Phi}$-simulation).

$$
\begin{aligned}
(d,d') \in \alpha(R) &\Leftrightarrow (d,d') \in R \\
(d,d') \in K(R) &\Leftrightarrow d \sqsubseteq_K d' \\
(d,d') \in \Phi_\perp(R) &\Leftrightarrow d \neq \perp \Rightarrow (d' \neq \perp \wedge (d,d') \in \Phi(R)) \\
((d,e),(d',e')) \in (\Phi_1 \times \Phi_2)(R) &\Leftrightarrow (d,d') \in \Phi_1(R) \wedge (e,e') \in \Phi_2(R) \\
(u,u') \in (\Phi_1 \otimes \Phi_2)(R) &\Leftrightarrow \forall d \in (\Phi_1(D))_\downarrow, e \in (\Phi_2(E))_\downarrow (u = (d,e) \Rightarrow \\
&\qquad \exists d' \in (\Phi_1(D))_\downarrow, e' \in (\Phi_2(E))_\downarrow \\
&\qquad\quad (d,d') \in \Phi_1(R) \wedge (e,e') \in \Phi_2(R)) \\
(u,u') \in (\Phi_1 \oplus \Phi_2)(R) &\Leftrightarrow \forall d \in (\Phi_1(D))_\downarrow (u = in_1(d) \Rightarrow \exists d' \in (\Phi_1(D))_\downarrow \\
&\qquad u' = in_1(d') \wedge (d,d') \in \Phi_1(R)) \\
&\qquad \wedge \\
&\qquad \forall e \in (\Phi_2(E))_\downarrow (u = in_2(e) \Rightarrow \exists e' \in (\Phi_2(E))_\downarrow \\
&\qquad\quad u' = in_2(e') \wedge (e,e') \in \Phi_2(R)) \\
(f,f') \in (\Phi_1 \to \Phi_2)(R) &\Leftrightarrow \forall d \in \Phi_1(D).(f(d), f'(d)) \in \Phi_2(R) \\
(f,f') \in (\Phi_1 \multimap \Phi_2)(R) &\Leftrightarrow \forall d \in \Phi_1(D)_\downarrow.(f(d), f'(d)) \in \Phi_2(R) \ .
\end{aligned}
$$

Thus Theorem 6.12 yields the following result about $rec\alpha.\Phi(\alpha)$.

Corollary 6.13 *Let $\Phi(\alpha)$ be a cppo-constructor built up from the variable α and constants K ranging over cppos, using $(-)_\perp$, $\times$, $\otimes$, $\oplus$, $\to$, and $\multimap$. Let $D = rec\alpha.\Phi(\alpha)$ be the cppo recursively defined by $\alpha = \Phi(\alpha)$, with associated isomorphism $i : \Phi(D) \cong D$. Then for any $d, d' \in D$, to prove $d \sqsubseteq d'$ it suffices to show $(d,d') \in R$ for some subset $R \subseteq D \times D$ satisfying:*

$$\forall (x,x') \in R.(i^{-1}(x), i^{-1}(x')) \in \Phi(R) \ . \tag{37}$$

The co-induction principle for recursively defined *cpos* established in (Pitts, 1994) can be deduced from Corollary 6.13 by restricting attention to cppo constructors $\Phi(\alpha)$ just involving the constructions $(-)_\perp$, $\otimes$, $\oplus$, and $((-) \multimap (+))_\perp$. For such Φ we have $\Phi((\xi_\perp)) = (\Psi(\xi))_\perp$, where $\Psi(\xi)$ is a corresponding cpo constructor built up using $(-)_\perp$, $\times$, $\uplus$, and the partial continuous function space constructor $(-) \to (+)_\perp$. Then $rec\xi.\Psi(\xi) = (rec\alpha.\Phi(\alpha))_\downarrow$ and the co-induction principle of Theorem 2.5 of *loc. cit.* is precisely that of Corollary 6.13 for this case. As shown in *loc. cit.*, binary relations satisfying (37) give a uniform notion of 'simulation' (for the cppo-constructors considered), which includes for example (one-sided) *applicative bisimulation*, used by Abramsky (1990) in connection with the lazy lambda calculus. We refer the reader to (Pitts, 1994) for further discussion and applications of this co-inductive characterization of the partial order on recursively defined cppos. (See also (Fiore, 1993) and (Rutten, 1993); and (Paulson, 1993) for related applications.)

The existence of a simulation (i.e. a relation satisfying (37)) containing two elements of a recursively defined cppo establishes that the order relation holds between them. Thus equality of elements d and d' can be established by exhibiting two simulations, one containing (d,d') and the other (d',d). There is a more symmetric notion of 'bisimulation' that establishes directly that two elements are equal. This can be obtained by changing the notion of identity relation for the relational structure $\mathcal{S}$ used in this section from $I_D = \sqsubseteq_D$ to

equality relations $I_D = \{(x, x) \mid x \in D\}$. In order to retain the necessary property $\hat{\Phi}(R, I_D) = I_{\hat{\Phi}(D,D)}$, one has to 'symmetrize' the definition of the action of cppo-constructors on $\mathcal{S}$-relations. For example $R_\perp$ would now be

$$\begin{aligned}(d, d') \in R_\perp \Leftrightarrow \quad & d \neq \perp \Rightarrow (d' \neq \perp \wedge (d, d') \in R) \\ \wedge \\ & d' \neq \perp \Rightarrow (d \neq \perp \wedge (d, d') \in R) \ .\end{aligned}$$

With similar changes for the other cppo constructors, Corollary 6.13 remains valid as stated except that $d \sqsubseteq d'$ is replaced by $d = d'$.

Remark 6.14 We have seen that it is fruitful to treat domain constructions via functors (of mixed variance) on a suitable category of domains. However, the approach taken in this paper to properties of recursively defined domains exploits structure of domain constructions over and above their functoriality properties. The unitary admissible action on relations that a particular construction possesses may well not be a consequence of its functoriality properties. Indeed the very notion of relation may take us outside the category of domains (as is the case for the notion of relation used in Sect. 5, for example). Rutten (1993) and Fiore (1993) develop results analogous to Theorem 6.12, but where relations are subobjects in the category and the notion of (bi)simulation is phrased purely in terms of (order-enriched) categorical properties of the functor. Such an approach has the advantage of relaxing the conditions on a functor needed to establish a co-inductive property of its minimal invariant. However, it has the disadvantage of tightening the conditions required of a simulation. In concrete instances it leads to results like Corollary 6.13, but weakened by adding the requirement that the simulation R is also a chain-complete subset of $D \times D$. Unlike the case for induction properties, such a chain-completeness condition is unnecessary. It can also be inconvenient, since in practice one seeks simulations with as few elements as possible when establishing a specific instance of the order relation on $rec\,\alpha.\Phi(\alpha)$.

Nevertheless, it can be useful to consider formulations of co-induction principles like Theorem 6.12 which relax the conditions imposed on the functor F. For example, (Pitts, 1994, Sect. 5) extends Corollary 6.13 to domain constructors $\Phi(\alpha)$ involving the Plotkin powerdomain, $P^\natural$ (Plotkin, 1976). In this case the proof is not quite a corollary of Theorem 6.12, because the operation $R \in \mathcal{S}(D) \mapsto P^\natural(R)$ given there does not obviously satisfy the requirements of Definition 4.6 to be an admissible action. It does satisfy a weaker version of condition (ii) of that definition, with $S' = \sqsubseteq_E$ and g the projection half of an embedding-projection pair; and as *loc. cit.* shows, this is sufficient to establish the co-induction result, which includes Abramsky's 'internal full abstraction' theorem for his domain model of SCCS (Abramsky, 1991, Proposition 3.11) as an instance. However, Rutten (1993, Sect. 5) obtains Abramsky's result (modulo the use of chain-complete simulations) just using the local continuity of the functor $P^\natural : \mathbf{Cpo}_\perp \longrightarrow \mathbf{Cpo}_\perp$ via his order-enriched categorical notion of bisimulation.

7 Parameterized recursive domains

The results in this paper exploit the fact that various simple domain constructors have well-behaved actions on relations. The solution of recursive domain equations with parameters provides an important source of more complicated domain constructors. For example, if $\Psi(\alpha,\beta)$ is a binary domain constructor built up from variables α and β using $(-)_\perp$, $\times$, $\otimes$, $\oplus$, $\to$, and $\multimap$, we get a unary domain constructor

$$\Phi(\alpha) \stackrel{def}{=} rec\,\beta.\Psi(\alpha,\beta) \tag{38}$$

whose value $\Phi(D)$ at a cppo D is the minimal solution of the domain equation $\beta = \Psi(D,\beta)$. To extend the results of Sect. 4 to include constructors like (38), we first have to see how the treatment in Sect. 3 of domain constructors in terms of locally continuous functors of mixed variance can be extended to cope with parameterization.

Parameterized minimal invariant cppos

We sketch how to associate a locally continuous functor $F : \mathbf{Cpo}_\perp^{\mathrm{op}} \times \mathbf{Cpo}_\perp \longrightarrow \mathbf{Cpo}_\perp$ to the domain constructor (38). Let $\hat{\Psi}(\alpha^-,\alpha^+,\beta^-,\beta^+)$ denote the result of separating positive and negative occurrences of α and β in $\Psi(\alpha,\beta)$. Then as in Sect. 3, $\hat{\Psi}$ determines a locally continuous functor

$$G : \mathbf{Cpo}_\perp^{\mathrm{op}} \times \mathbf{Cpo}_\perp \times \mathbf{Cpo}_\perp^{\mathrm{op}} \times \mathbf{Cpo}_\perp \longrightarrow \mathbf{Cpo}_\perp \tag{39}$$

with the property that $G(D,D,E,E) = \Psi(D,E)$. It is possible to construct a family of cppos $(F(D^-,D^+) \mid D^-,D^+ \in \mathbf{Cpo}_\perp)$ together with isomorphisms

$$i_{D^-,D^+} : G(D^-,D^+,F(D^+,D^-),F(D^-,D^+)) \cong F(D^-,D^+) \tag{40}$$

satisfying the following, *simultaneous minimal invariant property*:

> for each pair of cppos D^-, D^+, $(id_{F(D^+,D^-)}, id_{F(D^-,D^+)})$ is the least fixed point of the continuous endofunction δ_{D^-,D^+} on
>
> $$(F(D^+,D^-) \multimap F(D^+,D^-)) \times (F(D^-,D^+) \multimap F(D^-,D^+))$$
>
> given by $\delta_{D^-,D^+}(e^-,e^+) = (\delta^-_{D^-,D^+}(e^-,e^+), \delta^+_{D^-,D^+}(e^-,e^+))$, where
>
> $$\delta^-_{D^-,D^+}(e^-,e^+) \stackrel{def}{=} i_{D^+,D^-} \circ G(id_{D^+}, id_{D^-}, e^+, e^-) \circ (i_{D^+,D^-})^{-1}$$
> $$\delta^+_{D^-,D^+}(e^-,e^+) \stackrel{def}{=} i_{D^-,D^+} \circ G(id_{D^-}, id_{D^+}, e^-, e^+) \circ (i_{D^-,D^+})^{-1} \ .$$

The construction of these cppos can be carried out as in the proof of Theorem 3.3, except that one works with doubly-indexed families of cppos and embeddings to build up the desired doubly-indexed family of cppos as colimits of chains of embeddings. Just as in Theorem 3.4, the simultaneous minimal invariant property leads to a universal property for (40), namely:

> for all $f : A \circ\!\!\longrightarrow G(D^-,D^+,B,A)$ and $g : G(D^+,D^-,A,B) \circ\!\!\longrightarrow B$, there are unique $h : A \circ\!\!\longrightarrow F(D^-,D^+)$ and $k : F(D^+,D^-) \circ\!\!\longrightarrow B$ making the following squares commute

$$
\begin{array}{ccc}
F(D^-, D^+) & \xrightarrow{(i_{D^-,D^+})^{-1}} & G(D^-, D^+, F(D^+, D^-), F(D^-, D^+)) \\
\Big\uparrow{\scriptstyle h} & & \Big\uparrow{\scriptstyle G(id,id,k,h)} \\
A & \xrightarrow[f]{} & G(D^-, D^+, B, A) \\
G(D^+, D^-, F(D^-, D^+), F(D^+, D^-)) & \xrightarrow{i_{D^+,D^-}} & F(D^+, D^-) \\
\Big\downarrow{\scriptstyle G(id,id,h,k)} & & \Big\downarrow{\scriptstyle k} \\
G(D^+, D^-, A, B) & \xrightarrow[g]{} & B \ .
\end{array}
\tag{41}
$$

Moreover, the assignment $(f, g) \mapsto (h, k)$ determines a continuous function.

This universal property allows us to extend $(D^-, D^+) \mapsto F(D^-, D^+)$ to a locally continuous functor $F : \mathbf{Cpo}_\perp^{\mathrm{op}} \times \mathbf{Cpo}_\perp \longrightarrow \mathbf{Cpo}_\perp$. Indeed, given $u^- : D_2^- \circ\!\!\longrightarrow D_1^-$ and $u^+ : D_1^+ \circ\!\!\longrightarrow D_2^+$, we can define $F(u^-, u^+) : F(D_1^-, D_1^+) \circ\!\!\longrightarrow F(D_2^-, D_2^+)$ as the first component of the pair (h, k) corresponding to

$$
\begin{aligned}
f &= G(u^-, u^+, id, id) \circ (i_{D_1^-,D_1^+})^{-1} \\
g &= i_{D_1^-,D_1^+} \circ G(u^+, u^-, id, id) \ .
\end{aligned}
$$

The uniqueness part of the universal property ensures that this action on morphisms preserves identities and composition; and the the local continuity of the functor F follows from the fact that the assignment $(f, g) \mapsto (h, k)$ is continuous.

We have constructed $F : \mathbf{Cpo}_\perp^{\mathrm{op}} \times \mathbf{Cpo}_\perp \longrightarrow \mathbf{Cpo}_\perp$ out of $G : \mathbf{Cpo}_\perp^{\mathrm{op}} \times \mathbf{Cpo}_\perp \times \mathbf{Cpo}_\perp^{\mathrm{op}} \times \mathbf{Cpo}_\perp \longrightarrow \mathbf{Cpo}_\perp$. We write

$$
F(-, +) = rec\, X.G(-, +, X^-, X^+)
$$

to indicate this dependence. On the diagonal, i.e. in case $D^- = D = D^+$, the simultaneous minimal invariant property yields that $(F(D, D), i_{D,D})$ is the minimal invariant (in the sense of Definition 3.2) for the locally continuous functor $G(D, D, -, +) : \mathbf{Cpo}_\perp^{\mathrm{op}} \times \mathbf{Cpo}_\perp \longrightarrow \mathbf{Cpo}_\perp$. So when $G = \hat{\Psi}$ is the functor associated with the binary constructor $\Psi(\alpha, \beta)$, then for each D we have that $(F(D, D), i_{D,D})$ is the minimal invariant for the locally continuous functor $\hat{\Psi}(D, D, -, +)$. But the latter is precisely the functor obtained by separating positive and negative occurrences of β in the domain constructor $\Psi(D, \beta)$, whose minimal invariant is by definition $rec\,\beta.\Psi(D, \beta)$. Thus $rec\,\beta.\Psi(D, \beta) \cong F(D, D)$ is indeed the diagonalization of a locally continuous functor $F : \mathbf{Cpo}_\perp^{\mathrm{op}} \times \mathbf{Cpo}_\perp \longrightarrow \mathbf{Cpo}_\perp$.

Remark 7.1 Note that off the diagonal, i.e. in case $D^- \neq D^+$, the simultaneous minimal invariant property of $F(D^+, D^-)$ and $F(D^-, D^+)$ is more subtle than the property of a minimal invariant $D(D^-, D^+)$ for the functor $G(D^-, D^+, -, +)$. Indeed we cannot expect such a $D(D^-, D^+)$ to be functorial in D^-, D^+, since $D(D^-, D^+) \cong G(D^-, D^+, D(D^-, D^+), D(D^-, D^+))$ and the latter expression does not respect the variances of D^- and D^+.

Parameterized invariant relations

Now suppose $\mathcal{R}$ is a relational structure on $\mathbf{Cpo}_\perp$ for which the functor (39) has an admissible action on $\mathcal{R}$-relations (as in Definition 4.6 in case $m = n = 2$). In order to extend the results of Sects 5 and 6 to encompass the use of domain constructors like (38), we have to investigate whether the action of G on $\mathcal{R}$-relations gives rise to a similar action of $F(-,+) = rec\,X.G(-,+,X^-,X^+)$. Such an action should also be invariant under the isomorphisms (40) in the sense that

$$\begin{aligned} i_{D^-,D^+} &: G(R^-,R^+,F(R^+,R^-),F(R^-,R^+)) \subset F(R^-,R^+) \\ (i_{D^-,D^+})^{-1} &: F(R^-,R^+) \subset G(R^-,R^+,F(R^+,R^-),F(R^-,R^+)) \ . \end{aligned} \tag{42}$$

Proposition 7.2 *For each pair of cppos D^-, D^+ any family of* admissible $\mathcal{R}$*-relations*

$$F(R^-,R^+) \in \mathcal{R}_{adm}(F(D^-,D^+)) \qquad (R^- \in \mathcal{R}(D^-), R^+ \in \mathcal{R}(D^+))$$

satisfying (42) has the following universal property:

> *for all $f : A \circ\!\!\longrightarrow G(D^-,D^+,B,A)$, $g : G(D^+,D^-,A,B) \circ\!\!\longrightarrow B$, $R^- \in \mathcal{R}(D^-)$, $R^+ \in \mathcal{R}(D^+)$, $R \in \mathcal{R}(A)$, and $S \in \mathcal{R}_{adm}(B)$, if*
>
> $$f : R \subset G(R^-,R^+,S,R) \quad \text{and} \quad g : G(R^+,R^-,R,S) \subset S$$
>
> *then the unique $h : A \circ\!\!\longrightarrow F(D^-,D^+)$ and $k : F(D^+,D^-) \circ\!\!\longrightarrow B$ determined by f and g satisfy*
>
> $$h : R \subset F(R^-,R^+) \quad \text{and} \quad k : F(R^+,R^-) \subset S \ .$$

Proof This can be proved in the same way as Proposition 4.9, using the least fixed point characterization of (h,k), □

Note that the universal property given in the above proposition specializes to the following 'mixed' fixed point rule for the family of $\mathcal{R}$-relations $F(R^-,R^+)$:

$$\frac{R \subset G(R^-,R^+,S,R) \wedge G(R^+,R^-,R,S) \subset S}{R \subset F(R^-,R^+) \wedge F(R^+,R^-) \subset S} \quad (S \text{ admissible}). \tag{43}$$

In particular a family of admissible $\mathcal{R}$-relations satisfying (42) is unique if it exists. The universal property in the proposition also implies that $R^-, R^+ \mapsto F(R^-,R^+)$ satisfies the conditions of Definition 4.6 required for an admissible action on $\mathcal{R}$-relations. For $F(R^-,R^+)$ is admissible by assumption whether or not R^+ is; and if $u^- : R_2^- \subset R_1^-$ and $u^+ : R_1^+ \subset R_2^+$, then $F(u^-,u^+) : F(R_1^-,R_1^+) \subset F(R_2^-,R_2^+)$ follows by taking $f = G(u^-,u^+,id,id) \circ (i_{D_1^-,D_1^+})^{-1}$ and $g = i_{D_1^-,D_1^+} \circ G(u^+,u^-,id,id)$ (for which the corresponding h,k are $F(u^-,u^+)$ and $F(u^+,u^-)$) and taking $R = F(R_1^-,R_1^+)$ and $S = F(R_1^+,R_1^-)$.

If $\mathcal{R}$ has sufficient completeness properties, namely inverse images and intersections (cf. Definition 4.12), then the existence of relations satisfying the hypotheses of Proposition 7.2 can be established in much the same way that

the existence of invariant relations was established in Theorem 4.16. First note that when the relational structure $\mathcal{R}$ has inverse images, the conditions in (42) can be reformulated as a family of fixed point equations:

$$F(R^-, R^+) = (i^{-1}_{D^-, D^+})^* G(R^-, R^+, F(R^+, R^-), F(R^-, R^+)) \ . \qquad (44)$$

Fixing D^- and D^+, let $\mathcal{P}$ be the set of all pairs (ρ^-, ρ^+) of functions

$$\rho^- : \mathcal{R}(D^-) \times \mathcal{R}(D^+) \longrightarrow \mathcal{R}_{adm}(F(D^+, D^-))$$
$$\rho^+ : \mathcal{R}(D^-) \times \mathcal{R}(D^+) \longrightarrow \mathcal{R}_{adm}(F(D^-, D^+)) \ .$$

If $\mathcal{R}$ has intersections each $\mathcal{R}_{adm}(-)$ is a complete lattice and hence $\mathcal{P}$ is a complete lattice under the partial order given by

$$(\rho_1^-, \rho_1^+) \leq (\rho_2^-, \rho_2^+) \Leftrightarrow$$
$$\forall R^-, R^+ (\rho_2^-(R^-, R^+) \subset \rho_1^-(R^-, R^+) \wedge \rho_1^+(R^-, R^+) \subset \rho_2^+(R^-, R^+)) \ .$$

Let $\psi : (\rho^-, \rho^+) \mapsto (\psi^-(\rho^-, \rho^+), \psi^+(\rho^-, \rho^+))$ be the endofunction of $\mathcal{P}$ given by:

$$\psi^-(\rho^-, \rho^+)(R^-, R^+) \stackrel{def}{=} (i^{-1}_{D^+, D^-})^* G(R^+, R^-, \rho^+(R^-, R^+), \rho^-(R^-, R^+))$$
$$\psi^+(\rho^-, \rho^+)(R^-, R^+) \stackrel{def}{=} (i^{-1}_{D^-, D^+})^* G(R^-, R^+, \rho^-(R^-, R^+), \rho^+(R^-, R^+)) \ .$$

Then ψ is a monotone endofunction on the complete lattice $\mathcal{P}$. So we can apply the Tarski-Knaster theorem to obtain its least fixed point, (ϕ^-, ϕ^+). It suffices to prove $\phi^- = \phi^+$, since then $F(R^-, R^+) \stackrel{def}{=} \phi^-(R^-, R^+) = \phi^+(R^-, R^+)$ is an admissible $\mathcal{R}$-relation solving (44). The least prefixed point property of (ϕ^-, ϕ^+) together with the symmetry of ρ immediately gives $\phi^+ \leq \phi^-$. The reverse inequality is proved by appealing to the simultaneous minimal invariant property of $F(D^-, D^+)$. For the set

$$\{(e^-, e^+) \mid e^- : \phi^-(R^-, R^+) \subset \phi^+(R^+, R^-) \wedge e^+ : \phi^-(R^+, R^-) \subset \phi^+(R^-, R^+)\}$$

is a chain-complete subset of

$$(F(D^+, D^-) \multimap F(D^+, D^-)) \times (F(D^-, D^+) \multimap F(D^-, D^+))$$

containing $(\bot, \bot)$. One can verify that this subset is closed under the action of the continuous endofunction δ_{D^-, D^+} that, by the simultaneous minimal invariant property, has least fixed point (id, id). Hence by fixed point induction, (id, id) belongs to this subset. So $id : \phi^-(R^-, R^+) \subset \phi^+(R^+, R^-)$ for all R^-, R^+; and thus $\phi^- \leq \phi^+$.

To summarize, we have established the following result.

Theorem 7.3 *Let $\mathcal{R}$ be a relational structure on $\mathbf{Cpo}_\bot$ possessing inverse images and intersections . Let $G : \mathbf{Cpo}_\bot^{op} \times \mathbf{Cpo}_\bot \times \mathbf{Cpo}_\bot^{op} \times \mathbf{Cpo}_\bot \longrightarrow \mathbf{Cpo}_\bot$ be a locally continuous functor equipped with an admissible action on $\mathcal{R}$-relations. Then the locally continuous functor $F(-,+) = rec\, X.G(-,+,X^-,X^+)$ also possesses an admissible action on $\mathcal{R}$-relations, for which each $F(R^-, R^+)$ is admissible and satisfies* (42).

This theorem generalizes easily to the case of functors G with more than one extra parameter, i.e. of the form $(\mathbf{Cpo}_\perp^{\mathrm{op}} \times \mathbf{Cpo}_\perp)^n \times \mathbf{Cpo}_\perp^{\mathrm{op}} \times \mathbf{Cpo}_\perp \longrightarrow \mathbf{Cpo}_\perp$. Using it, the method developed in Sect. 5 for proving computational adequacy can be applied to functional languages involving recursive types with parameters, such as the metalanguage in (Plotkin, 1985).

Note that if $\mathcal{R}$ has identity relations and the action of G on $\mathcal{R}$-relations is unitary (in the sense of Definition 6.3), then so is the action of $rec\,X.G(-,+,X^-,X^+)$: just apply the rule (43) with R^-, R^+, R, S all identity $\mathcal{R}$-relations to see that $F(I_{D^-}, I_{D^+}) = I_{F(D^-,D^+)}$. Thus the mixed inductive/co-inductive rule established in Proposition 6.4 can be applied to recursively defined functors such as $rec\,X.G(-,+,X^-,X^+)$. To go on to deduce from this proposition the results in Sect. 6 about induction and co-induction extended to recursively defined constructors, one needs to establish (for particular relational structures) the property $F(R,I) = I$ which was used there. If G has such a property, it does not seem automatic that $rec\,X.G(-,+,X^-,X^+)$ will have it. Nevertheless the co-induction result of Corollary 6.13 does extend to the case of recursively defined constructors, as is indicated briefly in (Pitts, 1994, Remark 4.2). We leave consideration of a similarly extended induction principle to another occasion.

Concluding remarks

As far as this paper is concerned, expressions like '$rec\,X.G(-,+,X^-,X^+)$' are an informal notation for certain locally continuous functors. However, it would be interesting to extend the work in (Plotkin, 1985) to a metalanguage for recursive types whose syntax enforces the separation of positive and negative occurrences, whilst somehow still allowing diagonalization back to the 'usual' language of recursive types; together with a logic formalizing the various relational properties of (parameterized recursive) domains we have established here.

The utility of considering actions of type constructors on relations is perhaps best known from the body of work beginning with (Reynolds, 1983) on relational properties of parametric polymorphism: see (Abadi, Cardelli and Curien, 1993) and (Plotkin and Abadi, 1993) and the references therein. It is well known that inductive and co-inductive types can be encoded in the Girard-Reynolds polymorphic lambda calculus. Hence they inherit relational properties from those of $\forall$-types. Domain theory brings into the picture considerations of partiality and more particularly, the distinctive *combination* of inductive and co-inductive properties enjoyed by a recursively defined domain. The 'nearly ultimate' metalanguage for domain theory will presumably incorporate a syntactic version of Reynolds' relational parametricity for $\forall$-types (along the lines of *loc. cit.*), and derive from it the kind of proof principles for recursively defined domains that we have established here.

References

Abadi, M., Cardelli, L., and Curien, P.-L. (1993), Formal parametric polymorphism, *Theoretical Computer Science* **121**, 9–58.

Abramsky, S. (1990), The lazy lambda calculus, *in* "Research Topics in Functional Programming" (Turner, D., Ed.), pp. 65–116, Addison-Wesley, Inc.

Abramsky, S. (1991), A domain equation for bisimulation, *Information and Computation* **92**, 161–218.

Abramsky, S., and Ong, C.-H. L. (1993), Full abstraction in the lazy lambda calculus, *Inform. and Computation* **105**, 159–267.

Barendregt, H. (1984), "The Lambda Calculus. Its Syntax and Semantics" (revised edition), North-Holland, Amsterdam.

Crole, R. L., and Pitts, A. M. (1992), New foundations for fixpoint computations: FIX-hyperdoctrines and the FIX-logic, *Inform. and Computation* **98**, 171–210.

Fiore, M. P. (1993), A coinduction principle for recursive datatypes based on bisimulation, *in* "Proc. 8th Annual Symposium on Logic in Computer Science, Montréal", pp. 110–119, IEEE Computer Society Press, Washington.

Freyd, P. J. (1991), Algebraically complete categories, *in* "Proc. 1990 Como Category Theory Conference", Lec. Notes in Math., Vol. 1488 (A. Carboni, *et al*, Eds), pp. 95–104, Springer-Verlag, Berlin.

Freyd, P. J. (1992), Remarks on algebraically compact categories, *in* "Applications of Categories in Computer Science" (Fourman, M. P., Johnstone, P. T., and Pitts, A. M., Eds), pp. 95–106, L.M.S. Lecture Note Series 177, Cambridge University Press.

Freyd, P. J., and Scedrov, A. (1990), "Categories, Allegories", North-Holland, Amsterdam.

Gordon, M. G. C., Milner, R., and Wadsworth, C. P. (1979), "Edinburgh LCF", Lecture Notes in Computer Science, Vol. 78, Springer-Verlag, Berlin.

Gunter, C. A. (1992), "Semantics of Programming Languages. Structures and Techniques", MIT Press.

Gunter, C. A., and Scott, D. S. (1990), Semantic domains, *in* "Handbook of Theoretical Computer Science" (Leeuwen, J. van, Ed.), pp. 634–674, Elsevier Science Publishers B.V, Amsterdam.

Hudak, P., Peyton Jones, S., and Wadler, P. (1991), Report on the programming language Haskell: version 1.1, Technical Report, Yale Univ. and Glasgow Univ.

Jensen, F. V. (1981), Inductive inference in reflexive domains, Edinburgh Univ. Dept. Computer Science Report No. CSR 86-81.

Lawvere, F. W. (1970), Equality in hyperdoctrines and the comprehension schema as an adjoint functor, *in* "Applications of Categorical Algebra" (Heller, A., Ed.), pp. 1–14, Amer. Math. Soc., Providence RI.

LEHMANN, D. J., AND SMYTH, M. B. (1981), Algebraic specification of datatypes: a synthetic approach, *Math. Systems Theory* **14**, 97–139.

MAC LANE, S. (1971), "Categories for the Working Mathematician", Springer-Verlag, New York.

MEYER, A. R., AND COSMODAKIS, S. S. (1988), Semantical paradigms: notes for an invited lecture, *in* "Proc. 3rd Annual Symposium on Logic in Computer Science, Edinburgh", pp. 236–253, IEEE Computer Society Press, Washington.

MILNE, R. E. (1973), "The formal semantics of computer languages and their implementations", Ph.D. Thesis, Cambridge University.

MILNER, R., TOFTE, M., AND HARPER, R. (1990), "The Definition of Standard ML", MIT Press.

O'HEARN, P. W., AND TENNENT, R. D. (1993), Relational Parametricity and Local Variables, *in* "Conf. Record 20th Symp. on Principles of Programming Languages, Charleston", pp. 171–184, ACM, New York.

PAULSON, L. C. (1993) Co-induction and co-recursion in higher-order logic, Cambridge Univ. Computer Laboratory Tech. Rept. No. 304.

PITTS, A. M. (1993), Relational properties of recursively defined domains, *in* "Proc. 8th Annual Symp. on Logic in Computer Science, Montréal", pp. 86–97, IEEE Computer Soc. Press, Washington.

PITTS, A. M. (1993a), Computational adequacy via 'mixed' inductive definitions, *in* "Mathematical Foundations of Programming Language Semantics, Proc. 9th Int. Conf., New Orleans, LA, USA, April 1993", Lecture Notes in Computer Science, Vol. 802, pp. 72–82, Springer-Verlag, Berlin.

PITTS, A. M. (1994), A co-induction principle for recursively defined domains, *Theoretical Computer Science* **124**, 195–219.

PLOTKIN, G. D. (1973), Lambda definability and logical relations, Memorandum SAI-RM-4, Univ. Edinburgh School of Artificial Intelligence.

PLOTKIN, G. D. (1976), A powerdomain construction, *SIAM J. Comput.* **5**, 452–487.

PLOTKIN, G. D. (1977), LCF considered as a programming language, *Theoretical Computer Science* **5**, 223–255.

PLOTKIN, G. D. (1985), "Lectures on Predomains and Partial Functions". Notes for a course at CSLI, Stanford University.

PLOTKIN, G. D., AND ABADI, M. (1993), A logic for parametric polymorphism, *in* "Proceedings of the Conference on Typed Lambda Calculus and its Applications, Utrecht", Lecture Notes in Computer Science Vol. 664, pp. 361-375, Springer-Verlag, Berlin.

REYNOLDS, J. C. (1974), On the relation between direct and continuation semantics, *in* "2nd Int. Colloq. on Automata, Languages and Programming" (Loeckx, J., Ed.), Lecture Notes in Computer Science, Vol. 14, pp. 141–156, Springer-Verlag, Berlin.

REYNOLDS, J. C. (1983), Types, abstraction and parametric polymorphism, *in* "Information Processing 83" (Mason, R. E. A., Ed.), pp. 513–523, Elsevier Science Publishers B.V., Amsterdam.

RUTTEN, J. J. M. M. (1993), A structural co-induction theorem, *in* "Proc. 9th Int. Conf. on the Math. Foundations of Programming Language Semantics, New Orleans" (Brookes, S. *et al*, Eds), Lecture Notes in Computer Science, Vol. 802, pp. 83–102, Springer-Verlag, Berlin.

SCOTT, D. S. (1982), Domains for denotational semantics, *in* "Proc. 9th Internat. Coll. on Automata, Languages and Programming" (Nielsen, M., and Schmidt, E. M., Eds), Lecture Notes in Computer Science, Vol. 140, pp 577–613, Springer-Verlag, Berlin.

SMYTH, M. B., AND PLOTKIN, G. D. (1982), The category-theoretic solution of recursive domain equations, *SIAM J. Computing* **11**, 761–783.

THOMPSON, S. (1989), A logic for Miranda, *Formal Aspects of Computing* **1**, 339–365.

WADSWORTH, C. P. (1976), The relation between computational and denotational properties for Scott's D_∞ models of the lambda-calculus, *SIAM J. Comput.* **5**, 488-521.

WINSKEL, G. (1993), "The Formal Semantics of Programming Languages. An Introduction", MIT Press.

WINSKEL, G., AND LARSEN, K. G. (1984), Using information systems to solve recursive domain equations effectively, *in* "Semantics of Data Types" (Kahn, G., *et al*, Eds), Lecture Notes in Computer Science, Vol. 173, pp. 109–130, Springer-Verlag, Berlin.