Structure Meets Power Workshop

(Contributed Talks)

25 June 2023

https://www.cst.cam.ac.uk/conference/structure-meets-power-2023

Preface

This volume contains the papers presented at SmP 2023 Workshop: Structure Meets Power 2023 held June 25, 2023 in Boston and online.

The volume includes the abstracts of 12 accepted contributed talks SmP 2023 is the third workshop in the series of Structure Meets Power workshops. Earlier instalments of the series have been held in 2021 (online, affiliated with LiCS 2021) and 2022 (in Paris and online, affiliated with ICALP 2022).

The program of the SmP 2023 workshop focuses on bridging the divide in the field of logic in Computer Science, between two distinct strands: one focusing on semantics and compositionality ("Structure"), the other on expressiveness and complexity ("Power"). It is remarkable because these two fundamental aspects of our field are studied using almost disjoint technical languages and methods, by almost disjoint research communities. We believe that bridging this divide is a major issue in Computer Science, and may hold the key to fundamental advances in the field. The aim of the Structure meets Power workshop is to cultivate interaction between researchers who are interested in combining ideas from these two strands.

Our main sponsor was the EPSRC 2020-2023 project *Resources and co-Resources: a junction between categorical semantics, model theory and descriptive complexity.*

On June 5, 2023
in London, Cambridge,
Nottingham, and Prague.

Samson Abramsky
Anuj Dawar
Tomáš Jakl
Dan Marsden
Yoàv Montacute
Nihil Shah

Table of Contents

CATEGORICAL STRUCTURE IN THEORY OF ARITHMETIC

LINGYUAN YE

1. Extended Abstract

This paper intends to provide a categorical analysis of the weak arithmetic theory $I\Sigma_1$. $I\Sigma_1$ is a weaker theory than Peano arithmetic PA, obtained by restricting the induction axioms to Σ_1-formulas only. Let T be any theory of arithmetic. One important parameter in characterising the strength of an arbitrary arithmetic system T is its class of provably total recursive functions. A function f is *provably total recursive* in T iff there exists a Σ_1-formula φ_f in T that defines f, and that T proves the functionality of φ_f as follows,[1]

$$T \vdash \forall \overline{x} \exists_! y \varphi(\overline{x}, y).$$

Below is a classical result in proof theory that connects arithmetic and complexity classes:

Theorem 1.1 (Provably Total Recursive Functions in $I\Sigma_1$)**.** *The provably total recursive functions in $I\Sigma_1$ are exactly the primitive recursive functions.*

We will show that the validity of the above theorem is due to the following structural/categorical reason: The theory of Σ_1 induction presents the *initial* object of certain classes of categories, and Σ_1 subsets and primitive recursive functions between them forms an *instance* of such a category.

From a categorical perspective, the characterisation of primitive recursive functions aligns quite well with the definition of *parametrised natural number objects* (PNO) in a general category. Roughly speaking, a PNO in an arbitrary category is an object that supports primitive recursive construction internally. Crucially, being a PNO is a *universal property*, and it makes the categorical machinary works well in this situation.

The class of categories important for us are *coherent categories equipped with a PNO*, or simply, *pr-coherent categories*. Coherent categories are the categorical counterpart of first-order theories. Just like any propositional

Date: May 3, 2023.

Key words and phrases. Arithmetic, Provably Total Resursive Function, Categorical Logic.

[1]Through out the paper, we will use $\overline{x}, \overline{y}$, etc., to abbreviate a list of variables.

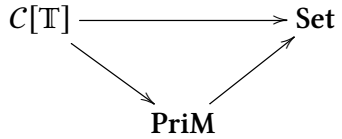
theory generates a Lindenbaum-Tarski algebra, every first-order theory generates a coherent category called its *syntactic category* (cf. [1]).

A very special object in the class of pr-coherent categories will be the *initial* pr-coherent category $\mathcal{C}$. Initiality means that for any other pr-coherent category $\mathcal{D}$, there exists an essentially unique functor from $\mathcal{C}$ to $\mathcal{D}$ that preserves the coherent structure and the PNO; such functors will be denoted as *pr-coherent functors*. For instance, **Set** is a pr-coherent category, and initiality of $\mathcal{C}$ induces a canonical interpretation $\mathcal{C} \to$ **Set** of objects in $\mathcal{C}$.

The main result of this paper is that the initial pr-coherent category can be presented as the *syntactic category* of some first-order theory $\mathbb{T}$, which will be a coherent theory of arithmetic. In coherent logic, the connectives are restricted to $\top, \bot, \wedge, \vee, \exists$, and in particular, $\neg, \to, \forall$ are *not allowed*. Hence, *a priori*, all formulas within $\mathbb{T}$ will automatically be Σ_1.

To arrive at an axiomatisation of $\mathbb{T}$, we need a detailed study of the semantic properties satisfied by a PNO in a general coherent category. The axiomatisation of $\mathbb{T}$ should be suitable, so as for it to be (a) interpretable in any coherent category equipped with a PNO, and (b) strong enough to show that its syntactic category $\mathcal{C}[\mathbb{T}]$ *has* a PNO, and proves its initiality.

Then, if we further observe that there is also a pr-coherent category **PriM** of recursively enumerable sets with primitive recursive functions between them (details see Section 5), and the canonical inclusion **PriM** $\hookrightarrow$ **Set** is also a pr-coherent functor, the initiality of $\mathcal{C}[\mathbb{T}]$ will imply we have the following factorisation of pr-coherent functors,



Such a factorisation means that the canonical interpretation of formulas in $\mathbb{T}$ in the standard model $\mathbb{N}$ *factors through* **PriM**, and hence all provable total functions in $\mathbb{T}$ will indeed be primitive recursive.

To go back to Theorem 1.1, we further study the relationship between $\mathbb{T}$ and the classical theory $I\Sigma_1$. We will prove that $I\Sigma_1$ is *conservative* over $\mathbb{T}$, in the sense that if a sequent in $\mathbb{T}$ is provable in $I\Sigma_1$, then it is already provable in $\mathbb{T}$ itself. This suffices to conclude that $\mathbb{T}$ is exactly the Π_2-fragment of $I\Sigma_1$, and thus they have the same class of provably total recursive functions.

Roughly speaking, our categorical analysis implies that the validity of Theorem 1.1 comes from the following *structural fact*: The category of Σ_1 formulas and provable Σ_1-functions between them in $I\Sigma_1$ turns out to be the initial pr-coherent category, and primitive recursive functions between recursively enumerable subsets is an instance of a pr-coherent category.

Acknowledgement

We want to thank Lev Beklemishev for reading the draft of this paper, and for presenting insightful questions and useful suggestions to us. We would also like to thank Simon Henry for pointing out an essential error in a previous version of this paper.

References

[1] Caramello, O. (2018). *Theories, Sites, Toposes: Relating and studying mathematical theories through topos-theoretic'bridges'*. Oxford University Press.

[2] Carboni, A. and Johnstone, P. (1995). Connected limits, familial representability and artin glueing. *Mathematical Structures in Computer Science*, 5(4):441–459.

[3] Coquand, T. and Dybjer, P. (1997). Intuitionistic model constructions and normalization proofs. *Mathematical Structures in Computer Science*, 7(1):75–94.

[4] Johnstone, P. T. (2002). *Sketches of an Elephant: A Topos Theory Compendium: Volume 1*, volume 1. Oxford University Press.

[5] Moerdijk, I. (1983). On the freyd cover of a topos. *Notre Dame Journal of Formal Logic*, 24(4):517–526.

[6] Rosser, B. (1936). Extensions of some theorems of gödel and church. *The journal of symbolic logic*, 1(3):87–91.

[7] Siders, A. (2015). Normalization proof for peano arithmetic. *Archive for Mathematical Logic*, 54(7):921–940.

[8] Sterling, J. (2021). *First Steps in Synthetic Tait Computability: The Objective Metatheory of Cubical Type Theory*. PhD thesis, Carnegie Mellon University.

ILLC, University of Amsterdam, Netherlands
Email address: ye.lingyuan.ac@gmail.com

Algebras for Automatic Relations*

Rémi Morvan
LaBRI, Univ. Bordeaux

14th May 2023

Abstract

We introduce the notion of synchronous algebras, which are algebras tailored to recognize automatic relations (*a.k.a.* regular, *i.e.* accepted by a synchronous automaton): any relation admits a syntactic synchronous algebra, and this relation is automatic if and only if this algebra is finite. We show that these algebras naturally arise by using Bojańczyk's theory of recognizability over monads on typed sets. We also highlight why this notion of algebras is inherently more suitable to studying properties of automatic relations compared to monoids or semigroups: using synchronous algebras, we prove the decidability of the membership and separation problems for the class of commutative relations. Lastly, we mention future work on projecting Eilenberg correspondances via monad functors.

1. Synchronous Automata

We study n-ary ($n \in \mathbf{N}_{>0}$) relations over finite words over some alphabet Σ, and start by describing the well-known model of synchronous automata. Define $\mathbf{S}_n\Sigma$ as the set of all "synchronous words", namely all words over the alphabet $(\Sigma \cup \{\bot\})^n \setminus \{(\bot, \dots, \bot)\}$, such that for each pair of positions $i < j$, for each $k \in [\![1, n]\!]$, if the k-th component of the i-th tuple is $\bot$, then so is the k-th component of the j-th tuple. For instance, $\binom{a}{b}\binom{a}{a}\binom{\bot}{a} \in \mathbf{S}_2\Sigma$ but $\binom{\bot}{a}\binom{b}{c} \notin \mathbf{S}_2\Sigma$ if $\Sigma = \{a, b, c\}$. By construction, $(\Sigma^*)^n$ and $\mathbf{S}_n\Sigma$ are naturally in bijection.

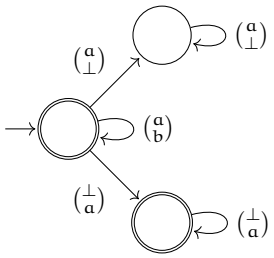


Figure 1.1: *Synchronous automaton accepting the binary relation* $\{(u, v) \in \Sigma^* \times \Sigma^* \mid |u| \leqslant |v|\}$. *Letters* a, b *in the transitions represent arbitrary elements of* Σ.

We define a *synchronous automaton* similarly to nondeterministic automata, except that transitions are labelled by n-tuple of letters or blank symbols, i.e. $(a_1, \dots, a_n)$ where each $a_i \in \Sigma \cup \{\bot\}$ with the constraint that not all a_i equal $\bot$—see Figure 1.1 for an example. Relations $R \subseteq \mathbf{S}_n\Sigma \cong (\Sigma^*)^n$ accepted by finite-state automata are called *automatic* or *regular*. Over unary relations (*i.e.* languages), recognizable relations (in the sense of being recognized by a monoid morphism $(\Sigma^*)^n \to M$ with finite domain) and automatic relations coincide, and define the class of regular languages. However, for n-ary relations with $n \geqslant 2$, the latter notion is strictly more expressive. For instance, the "has greater length" binary relation is automatic, as shown in Figure 1.1, but is not recognizable. See *e.g.* [CCG06] for pointers.

*Extended abstract submitted to Structure Meets Power 2023. Email: remi.morvan[at]u-bordeaux.fr. Work in progress.

2. Synchronous Algebras

In this section we only deal with binary relations but all definitions and results can be immediately lifted to n-ary relations. First, note that $\mathbf{S}_2\Sigma$ can be partitioned in six sets:

- the empty word;
- words in which the first and last letters belong to $\Sigma \times \Sigma$ (resp. $\Sigma \times \{\bot\}$, resp. $\{\bot\} \times \Sigma$);
- words in which the first letter belongs to $\Sigma \times \Sigma$ and last to $\Sigma \times \{\bot\}$ (resp. $\{\bot\} \times \Sigma$).

Moreover, for $u, v \in \mathbf{S}_2\Sigma$, to know whether the concatenation $u \cdot v$ belongs to $\mathbf{S}_2\Sigma$ it is necessary and sufficient to know to which of these six sets u and v belong. For the sake of simplicity, we now exclude the empty word and denote by $\mathbf{S}_2^+\Sigma$ the set $\mathbf{S}_2\Sigma \setminus \{(\varepsilon, \varepsilon)\}$.

Definition 2.1. *A* synchronous algebra $\mathbf{A}$ *consists of:*

- *five sets*[1] $(A_{\mathrm{LL}\to\mathrm{LL}}, A_{\mathrm{LL}\to\mathrm{LB}}, A_{\mathrm{LB}\to\mathrm{LB}}, A_{\mathrm{LL}\to\mathrm{BL}}, A_{\mathrm{BL}\to\mathrm{BL}})$*: we simply write* $x \in \mathbf{A}$ *if* $x \in A_{\alpha\to\beta}$ *for some* α, β*, and we say that* x *has* type $\alpha \to \beta$*;*
- *together with binary operators* $\cdot : A_{\alpha\to\beta} \times A_{\beta\to\gamma} \to A_{\alpha\to\gamma}$ *for all types* $\alpha \to \beta$ *and* $\beta \to \gamma$*, such that* $(x \cdot y) \cdot z = x \cdot (y \cdot z)$ *for all* $x, y, z \in \mathbf{A}$ *with compatible type.*

For instance, $\mathbf{S}_2^+\Sigma$ is a synchronous algebra. Note that synchronous algebra are an example of semigroupoids with three objects. *Morphisms between synchronous algebras* are defined naturally. This gives rise to a notion of "being recognized by a sychronous algebra (morphism)".

Theorem 2.2.

1. *Each relation* $R \subseteq \mathbf{S}_2^+\Sigma$ *admits a* syntactic synchronous algebra (morphism)*, namely a pair* $\langle \mathbf{A}_R, f_R : \mathbf{S}_2^+\Sigma \to \mathbf{A}_R \rangle$ *such that for each synchronous algebra morphism* $f : \mathbf{S}_2^+\Sigma \to \mathbf{B}$ *which recognizes* R*, there exists a morphism* $g : \mathbf{A}_R \to \mathbf{B}$ *such that* $f = g \circ f_R$*;*
2. *A relation is automatic if and only if its syntactic synchronous algebra is finite;*
3. *Mapping a relation to its syntactic synchronous algebra yields a bijection between pseudovarieties of automatic relations and pseudovarieties of finite synchronous relations.*

Proof sketch. 1 and 3 essentially follow from Bojańczyk's theory of recognizability over monads [Boj15, Boj20] for the monad

$$\langle A, B, C, D, E \rangle \mapsto \langle A^+, A^*BC^*, C^+, A^*DE^*, E^+ \rangle$$

in the category of 5-typed sets, and 2 follows from the fact that we can associate to any synchronous automaton a transition synchronous algebra—defined analogously to transition monoids/semigroups. □

3. Commutative Relations

Note that each automatic relation $R \subseteq \mathbf{S}_2^+[\Sigma]$ can be seen as a regular language over the alphabet $(\Sigma \cup \{\bot\})^n$, and this regular language admits a syntactic semigroup. However, we claim that this semigroup has little interest to study R: its properties follows from R, but also from the constraint that each element in R is a synchronous word.

[1] L stands for "letter" and B for "blank".

For instance, say that a relation R is *commutative* if, for every words $u, v, w, x, y \in \mathbf{S}_2^+[\Sigma]$ such that both $uxvyz$ and $uyvxz$ belong to $\mathbf{S}_2^+[\Sigma]$, then $uxvyz \in R$ if and only if $uyvxz \in R$. Knowing the syntactic semigroup of R, seen as a regular language, is of little help to determine if R is commutative. However, using syntactic synchronous algebras, we can easily show:

Proposition 3.1. *Whether an automatic relation is commutative is decidable. Moreover, given two automatic relations R_1, R_2, we can decide if there exists a commutative relation (or equivalently, a commutative and automatic relation) S that separates R_1 from R_2.*

4. Future Work

Projecting Eilenberg correspondances via monad functors. We can show that there is a monad functor—in the sense of [Str72, §1]—from the monad used in Theorem 2.2 to Kleene's monad $X \mapsto X^+$ (in the category of sets). This monad functor naturally induces a functor from the category of synchronous algebras to the category of semigroups[2]. Dually, there is a monad functor from the latter monad to the former. This situation is far from being an exception: classical monads used in automata theory are very often linked by monad functors, for instance there are monad functors from Kleene's monad to Wilke's monad (which gives rise to Wilke's algebras), and reciprocally, but also from the former to the monad of finitary countable words. This begs the following question: given two monads $\mathbf{T}$ and $\mathbf{S}$ and monad functors between them, can we find sufficient conditions on these functors so that a correspondance $\mathcal{V} \leftrightarrow \mathbb{V}$ between pseudovarieties of finite $\mathbf{T}$-algebras and pseudovarieties of recognizable $\mathbf{T}$-languages yields a correspondance $\mathcal{V}' \leftrightarrow \mathbb{V}'$ between pseudovarieties of finite $\mathbf{S}$-algebras and pseudovarieties of recognizable $\mathbf{S}$-languages?

References

[Boj15] Mikołaj Bojańczyk. Recognisable languages over monads. In *International Conference on Developments in Language Theory*, pages 1–13. Springer, 2015. Consulted version: https://arxiv.org/abs/1502.04898v1. doi:10.1007/978-3-319-21500-6_1.

[Boj20] Mikołaj Bojańczyk. Languages recognised by finite semigroups, and their generalisations to objects such as trees and graphs, with an emphasis on definability in monadic second-order logic, 2020. arXiv:2008.11635, doi:10.48550/arXiv.2008.11635.

[CCG06] Olivier Carton, Christian Choffrut, and Serge Grigorieff. Decision problems among the main subfamilies of rational relations. *RAIRO-Theoretical Informatics and Applications*, 40(2):255–275, 2006. doi:doi.org/10.1051/ita:2006005.

[Str72] Ross Street. The formal theory of monads. *Journal of Pure and Applied Algebra*, 2(2):149–168, 1972. doi:10.1016/0022-4049(72)90019-9.

[2]This boils down to the classical construction which transforms a typed semigroup/semigroupoid into a semigroup by adding a zero.

Compositional Algorithms on Compositional Data: Deciding Sheaves on Presheaves

Benjamin Merlin Bumpus

Algorithmicists are well-aware that fast dynamic programming algorithms are very often the correct choice when computing on compositional (or even recursive) *graphs*. Here we initiate the study of how to generalize this folklore intuition to mathematical structures writ large. We achieve this horizontal generality by adopting a categorial perspective which allows us to show that: (1) *structured decompositions* (a recent, abstract generalization of many graph decompositions) define Grothendieck topologies on categories of data (adhesive categories) and that (2) any computational problem which can be represented as a *sheaf* with respect to these topologies can be decided in linear time on classes of inputs which admit decompositions of bounded width and whose decomposition shapes have bounded feedback vertex number. This immediately leads to algorithms on objects of any C-set category; these include – to name but a few examples – structures such as: symmetric graphs, directed graphs, directed multigraphs, hypergraphs, directed hypergraphs, databases, simplicial complexes, circular port graphs and half-edge graphs. Finally we pair our theoretical results with concrete implementations of our main algorithmic contribution in the *AlgebraicJulia* ecosystem. The paper-length version of this extended abstract – which is *joint work Ernst Althaus, James Fairbanks and Daniel Rosiak* – is available at https://arxiv.org/abs/2302.05575.

1 PHILOSOPHY

Among the many different incarnations of compositionality in mathematics, the following three will be the main characters of this story: (1) the *structural* compositionality arising in graph theory in the form of *graph decompositions* whereby one decomposes graphs into smaller and simpler constituent parts [2–7], (2) the *representational* compositionalityarising in the form of *sheaves* in algebraic topology (and elsewhere) and (3) the *algorithmic* compositionality embodied by the intricate *dynamic programming* routines found in parameterized complexity theory [2, 3, 5–7]. Our main contribution (Theorem 4.2) is an algorithmic meta-theorem obtained by amalgamating these three perspectives.

2 DECISION PROBLEMS

Computational problems are assignments of data – thought of as *solution spaces* – to some class of input objects. We think of them as functors $\mathcal{F}\colon \mathsf{C} \to \mathsf{Sol}$ taking objects of some category C to objects of some appropriately chosen category Sol of solution spaces. Rather than computing the entire solution space, we often have to settle for more approximate representations of the problem – in the form of decision/optimization/enumeration problems. When does a given computational problem admit a compositional structure that is well-behaved with respect to decompositions of the input data? One of our motivations was the observation that sheaves may be applicable to this situation. Here we will focus on *decision problems*: for a given computational problem $\mathcal{F}$ we define the **$\mathcal{F}$-decision problem** as the composite $\mathsf{C} \xrightarrow{\mathcal{F}} \mathsf{Sol} \xrightarrow{\mathrm{dec}} \mathbf{2}$ where dec is a functor into $\mathbf{2}$ mapping solutions spaces to either $\bot$ or $\top$ depending on whether they witness yes- or no-instances to $\mathcal{F}$. For example consider the $\textsc{GraphColoring}_n$ problem[1]; it can easily be encoded as the representable functor $\mathsf{SimpFinGr}(-, K_n)\colon \mathsf{SimpFinGr}^{op} \to \mathsf{FinSet}$ on the category of finite simple graphs. One turns this into decision problems by taking $\mathrm{dec}\colon \mathsf{FinSet} \to \mathbf{2}$ to be functor which takes any set to $\bot$ if and only if it is empty. By passing to other categories of graphs (for instance that of graphs and their *monomorphisms*) one can also easily encode other decision problems such as VertexCover and $\textsc{OddCycleTransversal}$.

3 COMPOSITIONAL DATA

Parameterized complexity [3] is a two-dimensional framework for complexity theory whose main insight is that one should not analyze running times only in terms of the total input size, but also in terms of other *parameters* of the inputs (such as measures of *compositional structure* [3]). Here we represent compositional structure via *diagrams*: we think of an object $c \in \mathsf{C}$ obtained as the colimit of a diagram $d\colon \mathsf{J} \to \mathsf{C}$ as being *decompoised* by d into smaller constituent pieces. In particular we work with a special class of diagrams (suited for algorithmic manipulation) called **structured decompositions** [1]. Roughly they consist of a collection of objects in a category and a collection of *spans* which relate these objects (just like the edges in a graph relate its vertices).

Definition 3.1. Fixing a base category K, we define a **K-valued structured decomposition of shape** G as a diagram[a] of the form $d\colon \int G \to \mathsf{K}$ whose arrows are all *monic* in K. Structured decompositions assemble

[1] It asks to determine whether a given input graph G admits a proper coloring with at most n colors.

into a subcategory $\mathfrak{D}_{\mathrm{m}}\,\mathsf{K}$ of the category of diagrams in K.

[a]We think of graphs as presheaves: the Grothendieck construction applied to a graph G yeilds a category whose arrows form spans from each edge of G to its endpoints.

4 DECISION PROBLEMS AS SHEAVES

One of our main contributions is to show that structured decompositions yield Grothendieck topologies on *adhesive* categories which are *adhesively cocomplete*[2].

Theorem 4.1. *Let* C *be an adhesively cocomplete category. There is a functor* $\mathsf{Dcmp}\colon \mathsf{C}^{op} \to \mathbf{Set}$ *defined on objects as* $\mathsf{Dcmp}\colon c \mapsto \{d \mid \mathrm{colim}\, d = c \text{ and } d \in \mathfrak{D}_{\mathrm{m}}\,\mathsf{C}\}$. *This functor makes the pair* $(\mathsf{C}, \mathsf{Dcmp})$ *a site.*

This result allows us to consider those decision problems which display compositional structure compatible with that highlighted by structured decompositions; namely any problem $\mathcal{F}\colon \mathsf{C}^{op} \to \mathrm{Set}$ which is a *sheaf* with respect to the decomposition topology of Theorem 4.1.

Roughly, our main algorithmic result shows that any such problem can be solved in time that grows linearly in the size of the decomposition and exponentially[3] in terms of the size of the objects in the decompositions (its *width*).

Theorem 4.2. *Let* D *be a small adhesively cocomplete category, let* $\mathcal{F}\colon \mathsf{C}^{op} \to \mathsf{FinSet}$ *be a sheaf on the site* $(\mathsf{C}, \mathsf{Dcmp})$. *If we are given an algorithm* $\mathcal{A}_{\mathcal{F}}$ *which computes* $\mathcal{F}$ *on any object* c *in time* $\alpha(c)$, *then there is an algorithm which, given any* C*-valued structured decomposition* $d\colon \int G \to \mathsf{C}$ *of an object* $c \in \mathsf{C}$ *and a feedback vertex set* S *of* G, *computes* $\mathsf{dec}\,\mathcal{F}\,c$ *in time* $O(\max_{x \in VG} \alpha(dx) + \kappa^{|S|}\kappa^2)|EG|$ *where* $\kappa = \max_{x \in VG} |\mathcal{F}\,dx|$.

4.1 Sketching Theorem 4.2

Notice that, if C has colimits, then, since sheaves preserve colimits (sending them to limits of sets) [8], the following diagram will always commute [1].

$$\begin{array}{ccccc} \mathsf{C} & \xrightarrow{\quad\mathcal{F}\quad} & \mathsf{FinSet}^{op} & \xrightarrow{\quad\mathsf{dec}^{op}\quad} & 2^{op} \\ {\scriptstyle\mathrm{colim}}\uparrow & \textit{comm.} & \uparrow{\scriptstyle\mathrm{colim}} & & \\ \mathfrak{D}_{\mathrm{m}}\,\mathsf{C} & \xrightarrow{\quad\mathfrak{D}_{\mathrm{m}}\mathcal{F}\quad} & \mathfrak{D}_{\mathrm{m}}\,\mathsf{FinSet}^{op} & & \end{array} \tag{1}$$

Unpacking the diagram, the blue path corresponds to forgetting the compositional structure and then solving the problem on the entire input. On the other hand the red path corresponds to a compositional algorithm for deciding sheaves: one first evaluates $\mathcal{F}$ on the constituent parts of the decomposition and then joins[4] these solutions together to find a solution on the whole.

Unfortunately what we just described is still very inefficient since, for any input c and no matter which path we take in the diagram, we always end-up computing all of $\mathcal{F}(c)$: this is very large in general (think of coloring sheaf mentioned above). One might hope to overcome this difficulty by lifting[5] dec to a functor from FinSet^{op}-valued structured decompositions to 2^{op}-valued structured decompositions as is shown in the following diagram.

$$\begin{array}{ccccc} \mathsf{C} & \xrightarrow{\quad\mathcal{F}\quad} & \mathsf{FinSet}^{op} & \xrightarrow{\quad\mathsf{dec}^{op}\quad} & 2^{op} \\ {\scriptstyle\mathrm{colim}}\uparrow & \textit{comm.} & \uparrow{\scriptstyle\mathrm{colim}} & & \uparrow{\scriptstyle\mathrm{colim}=\wedge} \\ \mathfrak{D}_{\mathrm{m}}\,\mathsf{C} & \xrightarrow{\quad\mathfrak{D}_{\mathrm{m}}\mathcal{F}\quad} & \mathfrak{D}_{\mathrm{m}}\,\mathsf{FinSet}^{op} & \xrightarrow{\quad\mathfrak{D}_{\mathrm{m}}\,\mathsf{dec}^{op}\quad} & \mathfrak{D}_{\mathrm{m}}\,2^{op} \end{array}$$

However, this too is to no avail: the right-hand square of the above diagram does *not* commute in general. The main ingredient in proving our algorithmic result is to show that there is an *efficiently computable* endofunctor $\mathcal{A}\colon \mathfrak{D}_{\mathrm{m}}\,\mathsf{Set}^{op} \to \mathsf{Set}^{op}$ making the following diagram commute.

$$\begin{array}{ccccccc} \mathsf{C} & \xrightarrow{\quad\mathcal{F}\quad} & \mathsf{FinSet}^{op} & & \xrightarrow{\qquad\mathsf{dec}^{op}\qquad} & & 2^{op} \\ {\scriptstyle\mathrm{colim}}\uparrow & \textit{comm.} & \uparrow{\scriptstyle\mathrm{colim}} & & \textit{comm.} & & \uparrow{\scriptstyle\wedge} \\ \mathfrak{D}_{\mathrm{m}}\,\mathsf{C} & \xrightarrow{\quad\mathfrak{D}_{\mathrm{m}}\mathcal{F}\quad} & \mathfrak{D}_{\mathrm{m}}\,\mathsf{FinSet}^{op} & \xrightarrow{\quad\mathcal{A}\quad} & \mathfrak{D}_{\mathrm{m}}\,\mathsf{FinSet}^{op} & \xrightarrow{\mathfrak{D}_{\mathrm{m}}\,\mathsf{dec}^{op}} & \mathfrak{D}_{\mathrm{m}}\,2^{op} \end{array}$$

REFERENCES

[1] B. M. Bumpus, Z. A. Kocsis, and J. E. Master. Structured Decompositions: Structural and Algorithmic Compositionality. *arXiv preprint arXiv:2207.06091*, 2022. URL: https://doi.org/10.48550/arXiv.2207.06091.

[2] B. Courcelle and J. Engelfriet. *Graph structure and monadic second-order logic: a language-theoretic approach*, volume 138. Cambridge University Press, 2012. doi:https://doi.org/10.1017/CBO9780511977619.

[3] M. Cygan, F. V. Fomin, Ł. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. *Parameterized algorithms*. Springer, 2015. ISBN:978-3-319-21275-3. doi:https://doi.org/10.1007/978-3-319-21275-3.

[4] R. Diestel. *Graph theory*. Springer, 2010. ISBN:9783642142789.

[5] R. G. Downey and M. R. Fellows. *Fundamentals of parameterized complexity*, volume 4. Springer, 2013. URL: https://doi.org/10.1007/978-1-4471-5559-1.

[6] J. Flum and M. Grohe. Parameterized complexity theory. 2006. *Texts Theoret. Comput. Sci. EATCS Ser*, 2006. ISBN:978-3-540-29952-3. doi:https://doi.org/10.1007/3-540-29953-X.

[7] M. Grohe. Descriptive complexity, canonisation, and definable graph structure theory, cambridge university press, cambridge, 2017, x + 544 pp. *The Bulletin of Symbolic Logic*, 23(4):493–494, 2017. ISBN:1079-8986.

[8] Daniel Rosiak. *Sheaf Theory through Examples*. The MIT Press, 10 2022. URL: 10.7551/mitpress/12581.001.0001.

[2]i.e. they have colimits of diagrams of monomorphisms.

[3]which we expect since we are interested in NP-hard problems here

[4]Note that the colimit functor $\mathrm{colim}\colon \mathfrak{D}_{\mathrm{m}}\,\mathsf{FinSet}^{op} \to \mathsf{FinSet}^{op}$ – which is in red in Diagram 1 – is a *limit of sets*; we invite the reader to keep this in mind throughout.

[5]The construction of categories of structured decompositions is functorial [1], this is inherited from the analogous statement for categories of diagrams.

GADTs Are Not (Even Lax) Functors

Pierre Cagne
Appalachian State University

Abstract

We expose a fundamental incompatibility between GADTs' function-mapping operations and function composition. To do this, we consider sound and complete semantics of GADTs in a series of natural categorical settings. We first consider interpreting GADTs naively as functors on categories interpreting types. But GADTs' function-mapping operations are not total in general, so we consider semantics in categories that allow partial morphisms. Unfortunately, however, GADTs' function-mapping operations do not preserve composition of partial morphisms. This compels us to relax functors' composition-preserving property, and thus to interpret GADTs as lax functors. This exposes yet another behavior of GADTs that is not computationally reasonable: if G interprets a GADT, then $G(f)$ can be defined on more elements than $G(g)$ even though f is defined on fewer elements than g. This is joint work with Patricia Johann.

In this talk, I will present several obstructions to the generalization of the usual Initial Algebra Semantics (IAS) of Algebraic Data Types (ADTs) to their syntactic generalizations known as Generalized Algebraic Data Types (GADTs). ADTs — such are lists, trees, graphs, etc. — are ubiquitous in modern programming. They encode, for example, data structures that are critical in reducing the complexity of algorithms that process them. Moreover, they are safe and easy to reason with precisely because of the theoretical understanding that IAS affords them. Pattern-matching, folds, mapping, etc., are all justified by the fact that IAS is sound and complete for ADTs. For this reason, we would like to construct an IAS for GADTs in such a way that it specializes to the standard IAS for ADTs [MA86]. Although GADTs are extensively used in practice in, say, Haskell and OCaml, they are still missing a sound and complete semantics.

The obstructions to IAS for GADTs can be illustrated using the *equality* defined by

```
data Eq :: * → * → * where
  Refl :: ∀ α. Eq α α
```

According to the above Haskell syntax, `Eq` is a binary GADT with a unique constructor `Refl` inhabiting each instance of the form `Eq α α`. Although it seems simple at first glance, `Eq` support the definition of the following transport function, which will be important below:

```
  trp :: ∀ α β. Eq α β → (α → β)
  trp Refl x = x
```

Naive IAS for GADTs. Our first attempt at an IAS for GADTs is by direct extension. Recall that the standard IAS of ADTs associates to each n-ary ADT `A` a functor $A : \mathcal{C}^n \to \mathcal{C}$ where $\mathcal{C}$ is a category interpreting closed types. Moreover, this functor A is obtained as the (carrier of the) initial algebra of an endofunctor on the functor category $\mathrm{Func}\,(\mathcal{C}^n, \mathcal{C})$ derived from the body of the definition of the ADT `A`; taking $\mathcal{C}$ to be the category Set of sets and functions is usually enough to carry out most reasoning on ADTs. The completeness of IAS for ADTs implies in particular that, for every closed type τ with interpretation X, $A(X)$ is (in natural bijection with) the set of normal forms of closed terms of type `A` τ. The same property cannot be obtained for GADTs. Indeed, suppose that there is a functor $E : \mathrm{Set}^2 \to \mathrm{Set}$ interpreting `Eq`. If we want the interpretation to be sound, then for each type τ with interpretation X we need an element $r_X \in E(X, X)$ interpreting `Refl :: E` τ τ. Moreover, soundness and

completeness imply that the interpretation of the unit type must be a singleton set 1, so for any function $x : 1 \to X$, there must be an element $E(x, \mathrm{id}_1)(r_1) \in E(X, 1)$. We can prove that, for any types τ_1 and τ_2 with interpretations X_1 and X_2, respectively, the interpretation of the function `trp` sends elements of $E(X_1, X_2)$ to bijections $X_1 \simeq X_2$. Thus, whenever the type τ has at least one element, the element found in $E(X, 1)$ forces $X \simeq 1$. Clearly, this semantics fails to be complete. As shown in [CJ23], this argument is not specific to the category Set and can be replayed in any category $\mathcal{C}$ with the enough structure to express the usual IAS of ADTs.

Allowing for partial functions. The failure of the first attempt is instructive: the core issue there is that functions of the form $E(x, \mathrm{id}_1)$ must send the element r_1 to elements in $E(X, 1)$ even when completeness would require $E(X, 1)$ to be empty. In other words, the totality of functions of the form $E(x, \mathrm{id}_1)$ is problematic. We can try to overcome this problem by allowing such functions to be partial, but we need a notion of partiality that is coherent with computations. In our view, the core feature of such a notion is that computations cannot recover from failure. We therefore propose the following definition:

Definition 1. A *structure of computational partiality* on a category $\mathcal{C}$ is a wide subcategory of $\mathcal{C}$ whose complement is a cosieve.

A *wide subcategory* of $\mathcal{C}$ is a subcategory of $\mathcal{C}$ that contains all objects of $\mathcal{C}$, and a *cosieve* on $\mathcal{C}$ is a subclass S of the morphisms of $\mathcal{C}$ such that for all morphisms $f : A \to B$ and $g : B \to C$ in $\mathcal{C}$, if $f \in S$ then $gf \in S$. The category PSet of sets and partial functions between them is an example of such a category when we take the structure of computational partiality to be Set seen as a subcategory of PSet. For this reason, we often call the morphisms of a structure of computational partiality *total*, and the morphisms in the cosieve that is its complement *partial*.

Given a category $\mathcal{C}$ with a structure of computational partiality $\mathcal{D}$ on it, we aim to repair the naive attempt above by interpreting the ambient language in $\mathcal{D}$, except for the GADTs, which will be interpreted as functors from $\mathcal{C}^n$ to $\mathcal{C}$ rather than from $\mathcal{D}^n$ to $\mathcal{D}$. Informally, we interpret everything definable in syntax in $\mathcal{D}$, but we allow the function-mapping operations of GADTs to send total morphisms to partial morphisms. In particular, the unit type must be interpreted as a terminal object 1 in $\mathcal{D}$. Now the functor $E : \mathcal{C}^2 \to \mathcal{C}$ interpreting `Eq` is free, a priori, to give a partial morphism $E(x, \mathrm{id}_1)$ even when $x : 1 \to X$ is a total morphism. However, such a morphism $x : 1 \to X$ is always a section, with its retraction being the unique total morphism from X to 1 given by the fact that 1 is terminal in $\mathcal{D}$. Being a functor, $E(\_, 1)$ must send sections to sections, so that $E(x, \mathrm{id}_1)$ is a section as well. Now our hope to fix the issue is dashed because sections in $\mathcal{C}$ are always total: if $s : X \to Y$ were a partial section, then, for any retraction r of s, $\mathrm{id}_X = rs$ would need to be partial as well. But identities are total by definition. We can then replay the argument of the naive attempt: being total, $E(x, \mathrm{id}_1)$ must send the (now global) element r_1 of $E(1, 1)$ to a (now global) element of $E(X, 1)$, which again forces $X \simeq 1$. This shows, as before, that the interpretation of any type τ with at least one element is the object 1, making the interpretation highly non-complete.

Relaxing GADTs' functorial behavior. Allowing GADTs' functorial behavior to target partial morphisms failed because classes of morphisms defined by equations, such as the class of sections, are necessarily preserved by functors. We therefore can't expect GADTs' interpretations to respect composition on the nose. However, if we can map a function `f` over an element of a GADT, and if we can also map a function `g` over that result, then we should obtain the same result by mapping `g . f` in one go over the original element. That is, mapping a composition should be the same as mapping the components of the composition one after the other provided all these operations are well-defined. This suggests that $\mathcal{C}$ should include an order $\leq$ on each hom-set, where $f \leq g$ is read informally as "g is an extension of f to a bigger domain of

definition". We therefore require that $\mathcal{C}$ is enriched over the category Pos of posets and monotonic functions between them. We intend to interpret GADTs as normal lax functors, rather than simple functors, into $\mathcal{C}$. A*normal lax functor* $G : \mathcal{B} \to \mathcal{C}$ between Pos-enriched categories is defined in the same way as an enriched functor, except that we require $G(g)G(f) \leq G(gf)$ instead of $G(g)G(f) = G(gf)$ for all composable morphisms f and g of $\mathcal{B}$. Interpreting GADTs as normal lax functors resolves the issue raised by sections in the previous paragraph: if s is a section with retraction r, and G is a normal lax functor, then $G(r)G(s)$ only has to be less than or equal to $G(rs) = G(\mathrm{id}) = \mathrm{id}$, and so $G(s)$ must no longer be a section and can be non-total.

Now consider the Pos-enriched category PSet, where $f \leq g$ holds for $f, g : X \to Y$ if and only if, for all $x \in X$, f defined on x implies $g(x) = f(x)$. A sound and complete interpretation of the type `Bool` of booleans in PSet must be a 2-element set, say $B = \{\bot, \top\}$, where $\bot$ interprets `false` and $\top$ interprets `true`. It must also take product types must take sets X and Y to the cartesian product $X \times Y$. Now, consider the partial function $f : B \times B \to B \times B$ defined only on the pairs of the form $(\bot, y)$, and given by $f(\bot, y) = (\bot, y)$. Then $f \leq g$, where g is total and defined by $g(x, y) = (x, x \vee y)$. The monotonicity of the normal lax functor E interpreting `Eq` implies $E(f, f) \leq E(g, f)$. In particular, $E(g, f)$ is defined on any element of $E(B \times B, B \times B)$ on which $E(f, f)$ is defined, and agrees with $E(f, f)$ on it. Thus, $E(g, f)(r_{B\times B}) = E(f, f)(r_{B\times B}) = r_{B\times B}$. According to the algorithm given in [JC22], this is a contradiction: only the pairs of function of the form (h, h) for a given $h : B \times B \to B \times B$ can be mapped over $r_{B\times B}$.

It might be difficult to spot the source of the problem with `Eq` because it is so degenerate, but the above example is important because `Eq` is quintessential in the theory of GADTs. Nevertheless, the issue deriving from $E(f, f) \leq E(g, f)$ is perhaps better illustrated by a properly recursive GADT that uses `Eq`, such as

```
data Seq :: * → * where
  inj :: ∀ α. α → Seq α
  pair :: ∀ α β γ. Seq α → Seq β → Eq γ (α,β) → Seq γ
```

Write $S : \mathrm{PSet} \to \mathrm{PSet}$ for the normal lax functor interpreting `Seq`, $i_B : B \to S(B)$ for the interpretation of `inj` instantiated at `Bool`, and $p_B : S(B) \times S(B) \times E(B \times B, B \times B) \to S(B \times B)$ for the interpretation of `pair` instantiated at `Bool`, `Bool`, and `(Bool,Bool)`. It should be intuitively clear that the partial function f can be mapped over (the interpretation of) `pair (inj false) (inj true) Refl :: S (Bool,Bool)`. The formal justification, given in [JC22], is that f can be written as $(x, y) \mapsto (f_1(x), f_2(y))$ for the partial functions $f_1, f_2 : B \to B$, where f_1 is defined only on $\bot$ with value $\bot$ and f_2 is id_B. To perform this mapping operation, we simply strip the constructors, apply f_1 and f_2 to the relevant data, and reapply the constructors. The result is the element we started with, namely, $S(f)(p_B(i_B(\bot), i_B(\top), r_{B\times B})) = p_B(i_B(\bot), i_B(\top), r_{B\times B})$. Since $f \leq g$, since $S(g)$ is defined wherever $S(f)$ is defined, and since $S(g)$ agrees with $S(f)$ there, $S(g)$ is also defined on $p_B(i_B(\bot), i_B(\top), r_{B\times B})$ with value $p_B(i_B(\bot), i_B(\top), r_{B\times B})$. This implies that g is also of the form $(x, y) \mapsto (g_1(x), g_2(y))$ for some (necessarily total) functions $g_1, g_2 : B \to B$. But we easily check that it is not: $g(\bot, \bot) = (\bot, \bot)$ implies that $g_2(\bot) = \bot$ and $g(\top, \bot) = (\top, \top)$ implies that $g_2(\bot) = \top$.

References

[CJ23] Pierre Cagne and Patricia Johann. Are GADTS really data structures? Submitted, 2023.

[JC22] Patricia Johann and Pierre Cagne. Characterizing Functions Mappable over GADTs. In Ilya Sergey, editor, *Programming Languages and Systems*. Springer Nature Switzerland, 2022.

[MA86] Ernest G. Manes and Michael A. Arbib. *Algebraic Approaches to Program Semantics*. Monographs in Computer Science. Springer New York, 1986.

Capturing Quantum Isomorphism in the Kleisli Category of the Quantum Monad

Amin Karamlou

February 2023

Graph[1] Homomorphism and isomorphism are two of the most central problems in computer science. Naturally, relaxations of these problems have been introduced and studied. Many of these relaxations can be phrased in terms of spoiler-duplicator games. The recently introduced game comonads (see e.g. [ADW17, AS20, Abr22, MS21]) provide a structural, and compositional account of this class of relaxations, whereby winning strategies for duplicator in various spoiler-duplicator games can be captured in the Kleisli category of the relevant comonads. Examples include k-consistency [ABD07], and the Weisfeiler-Lehman method [Kie20], which can both be characterised by suitable pebble games. In some sense the comonads formalise the intuition that these relaxations are coalgebraic in nature, pointing towards potential connections with ideas of universal coalgebra [Rut00] and the tradition of applying coalgebraic methods in computer science.

On the other hand, there are also relaxations of graph homomorphism and isomorphism which appear to be algebraic in nature. Examples of this second class of relaxations include fractional homomorphism [BD21] , and fractional isomorphism [RSU94], which both admit linear algebraic characterisations. In the language of category theory, one might expect that these relaxations can be captured by monads, the dual construction to comonads, which are known to have intimate links with the study of universal algebra. Indeed, an example of such a construction appeared in [ABdSZ17] where it was shown that quantum homomorphisms [MR16] are in bijective correspondence with homomorphisms in the Kleisli category of a certain graded quantum monad. [ABdSZ17] left open the question of providing an account of the related quantum isomorphism [AMR$^+$19] relaxation in this categorical framework.

In this talk, we will discuss ongoing work on answering the above question. We will show how quantum isomorphism can be captured by the existence of suitable back-and-forth morphisms in the Kleisli category of the quantum monad. We will also show how non-signalling graph homomorphism and isomorphism can be captured using similar constructions in the Kleisli category of a version of the distribution monad, defined on the category of graphs.

1 Non-local Games

A non-local game is a cooperative game played between a verifier and two players, usually referred to as Alice and Bob. Informally, the game consists of one round of the following protocol: (1) The verifier sends a question to each player. (2) Without communicating, each player responds with an answer. (3) The players win the game if their answers satisfy a pre-defined winning condition.

Given two graphs G, H, the (G, H)-homomorphism game is played as follows: (1) The verifier sends each player a vertex from G. (2) Each player responds with a vertex from H. (3) The players win the game if they respond with

[1] We consider simple loopless undirected graphs for simplicity, however, the results discussed here hold more generally for arbitrary relational structures.

the same vertex whenever they receive the same vertex, and with adjacent vertices whenever they receive adjacent vertices. The (G, H)-isomorphism game is a natural isomorphism variant of the above game. We refer the reader to [AMR$^+$19] for a definition.

With access to classical resources, the players win the (G, H)-homomorphism (isomorphism) game precisely when there is a homomorphism (isomorphism) between G and H. What is interesting is that when given access to quantum resources, specifically the ability to share an entangled quantum state between them, the players can win the game even in cases where there is no homomorphism (isomorphism) between the underlying graphs. We write $G \overset{q}{\to} H$ ($G \overset{q}{\cong} H$) whenever such a quantum strategy exists. One can even consider a larger class of strategies, called non-signalling strategies, where the only restriction is that Alice's answer cannot depend on Bob's question and vice-versa. We write $G \overset{ns}{\to} H$ ($G \overset{ns}{\cong} H$) in this case. It is worth mentioning that an arbitrary non-signalling strategy may not be producible by any physical device.

2 Monads

In [ABdSZ17] the quantum monad $\mathcal{Q}_d$ is introduced and used to provide a categorical account of quantum homomorphisms.

Proposition 1. *([ABdSZ17, proposition 8]) $G \overset{q}{\to} H$ iff there exists a Kleisli morphism $G \to \mathcal{Q}_d H$.*

The quantum monad can be seen as a variant of the well-known distribution monad $\mathcal{D}$, where probabilities are replaced with projectors. In fact, we can show that a version of the $\mathcal{D}$ defined over the category of graphs can be used to capture non-signalling homomorphisms.

Proposition 2. *$G \overset{ns}{\to} H$ iff there exists a Kleisli morphism $G \to \mathcal{D}H$.*

Given the above propositions, one might expect that quantum and non-signalling isomorphisms are captured by Kleisli isomorphisms of $\mathcal{Q}_d$ and $\mathcal{D}$. This turns out to not be the case. In fact, these Kleisli isomorphisms correspond to classical graph isomorphism.

Proposition 3. *$G \cong_{kl(\mathcal{Q}_d)} H$ iff $G \cong_{kl(\mathcal{D})} H$ iff $G \cong H$*

Our next result shows that quantum and non-signalling isomorphism can still be captured in terms of the Kleisli categories of $\mathcal{Q}_d$ and $\mathcal{D}$. This is achieved by introducing an intermediate notion of equivalence, based on the existence of suitable back-and-forth Kleisli morphisms. Before defining this back-and-forth equivalence we recall that Kleisli morphisms of $\mathcal{D}$ correspond to stochastic matrices. The Kleisli morphisms of $\mathcal{Q}_d$ can also be seen as matrices, which we refer to as *projective stochastic matrices*. These are matrices where each entry is a projector, and where the projectors in each column form a PVM.

Definition 1. *Let T be the quantum monad $\mathcal{Q}_d$ or the distribution monad $\mathcal{D}$. We define the relation $G \overset{Kl(T)}{\rightleftarrows} H$ whenever there are Kleisli morphisms $M : G \to TH$ and $N : H \to TG$ such that $N = M^T$.*

We can now prove the following.

Proposition 4. *$G \overset{q}{\cong} H$ iff $G \overset{Kl(\mathcal{Q}_d)}{\rightleftarrows} H$*

A similar result also holds in the case of non-signalling strategies[2].

[2]It is worth noting that [AMR$^+$19] showed that non-signalling isomorphism also corresponds to fractional isomorphism.

Proposition 5. $G \overset{ns}{\cong} H$ *iff* $G \overset{Kl(\mathcal{D})}{\rightleftarrows} H$

This account of quantum and non-signalling isomorphism in terms of back-and-forth Kleisli arrows is reminiscent of the formulation of winning strategies for duplicator in the k-pebble game in terms of back-and-forth coKleisli morphisms in [ADW17]. This later equivalence was eventually phrased in terms of the existence of a span of open pathwise embeddings in the coEilenberg-Moore (coEM) category of the pebbling comonad [AS20]. Thus, it would be interesting to see if a similar span (or cospan) construction in the EM categories of $\mathcal{Q}_d$ and $\mathcal{D}$ could capture the $G \overset{Kl(T)}{\rightleftarrows} H$ relation.

References

[ABD07] Albert Atserias, Andrei Bulatov, and Victor Dalmau. On the power of k-consistency. In *Automata, Languages and Programming: 34th International Colloquium, ICALP 2007, Wrocław, Poland, July 9-13, 2007. Proceedings 34*, pages 279–290. Springer, 2007.

[ABdSZ17] Samson Abramsky, Rui Soares Barbosa, Nadish de Silva, and Octavio Zapata. The quantum monad on relational structures, 2017.

[Abr22] Samson Abramsky. Structure and power: an emerging landscape. *Fundamenta Informaticae*, 186, 2022.

[ADW17] Samson Abramsky, Anuj Dawar, and Pengming Wang. The pebbling comonad in finite model theory. In *Logic in Computer Science (LICS), 2017 32nd Annual ACM/IEEE Symposium on*, pages 1–12. IEEE, 2017.

[AMR+19] Albert Atserias, Laura Mančinska, David E Roberson, Robert Šámal, Simone Severini, and Antonios Varvitsiotis. Quantum and non-signalling graph isomorphisms. *Journal of Combinatorial Theory, Series B*, 136:289–328, 2019.

[AS20] Samson Abramsky and Nihil Shah. Relating structure and power: Extended version. *arXiv preprint arXiv:2010.06496*, 2020.

[BD21] Silvia Butti and Victor Dalmau. Fractional homomorphism, weisfeiler-leman invariance, and the sherali-adams hierarchy for the constraint satisfaction problem, 2021.

[Kie20] Sandra Kiefer. The weisfeiler-leman algorithm: An exploration of its power. *ACM SIGLOG News*, 7(3):5–27, nov 2020.

[MR16] Laura Mančinska and David E. Roberson. Quantum homomorphisms. *Journal of Combinatorial Theory, Series B*, 118:228–267, 2016.

[MS21] Yoàv Montacute and Nihil Shah. The pebble-relation comonad in finite model theory. *arXiv preprint arXiv:2110.08196*, 2021.

[RSU94] Motakuri V. Ramana, Edward R. Scheinerman, and Daniel Ullman. Fractional isomorphism of graphs. *Discrete Mathematics*, 132(1):247–265, 1994.

[Rut00] J.J.M.M. Rutten. Universal coalgebra: a theory of systems. *Theoretical Computer Science*, 249(1):3–80, 2000. Modern Algebra.

Concurrent Games over Relational Structures

Glynn Winskel
Edinburgh Research Centre, Central Software Institute, Huawei
University of Strathclyde, UK

Abstract

Spoiler-Duplicator games are used in finite model theory (FMT) to study the expressive power of logical languages. These games make use of resources (e.g. number of rounds) which correspond to restrictions on the fragments of first-order logic considered (e.g. quantifier rank). One may venture beyond Spoiler-Duplicator games to a framework that supports both resources and structure. In the semantic world, concurrent games and strategies based on event structures [12, 13, 14] have been demonstrated to achieve that goal: concurrent games and strategies can be composed and also support quantitative extensions, e.g. to probabilistic and quantum computation, as well as realise the usage resources [15, 6, 5, 4, 16]. As a result, concurrent games and strategies provide a rich arena in which structure meets power – a line of research that became prominent with the paper of Abramsky, Dawar and Wang [1] on the pebbling comonad in finite model theory.

We introduce concurrent games and strategies over relational structures with many-sorted signature Σ. The use of many sorts allows individual games and strategies to involve multiple relational structures. Through their foundation in event structures, concurrent games and strategies represent closely the operational nature of games, their interactivity, dependence, independence, and conflict of moves. One advantage of concurrent games and strategies is that they support composition of strategies to form a bicategory, and a category in the case of deterministic strategies.

As a first exploration of the expressivity of this framework, we exhibit a deconstruction of the traditional examples of games and strategies of FMT as specifying strategies *from* one game *to* another. It applies to pebbling games, such as the k-pebble game and the all-in-one k pebble game; and their variants on transition systems determining simulation and trace inclusion. We thus reconcile the composition of strategies as coKleisli maps, prevalent in recent algebraisation of finite model theory (e.g. [1, 3, 8, 11]), with the more usual composition of strategies following the paradigm of Conway and Joyal [7, 10].

We provide a first systematic method for constructing comonads for all one-sided Spoiler-Duplicator games describable in our framework. Game comonads are now realised by adjunctions to a category of δ-games; a comonad δ in a bicategory of signature games gives a generic specification of Spoiler-Duplicator interaction.

We characterise strategies of δ-games in the one-sided case via functions taking the form of δ-homomorphisms which extend the role of coKleisli maps w.r.t. game comonads. [The one-sided case is treated in a recent submission to MFCS by Montacute and W.]

A recent characterisation of strategies of δ-games in the two-sided case is more subtle; it involves a pair of functions which interact in the manner of Geometry of Interaction [9, 2].

2012 ACM Subject Classification Theory of computation

Keywords and phrases Event Structures, concurrent games, Spoiler-Duplicator games

Digital Object Identifier 10.4230/LIPIcs...

References

1 Samson Abramsky, Anuj Dawar, and Pengming Wang. The pebbling comonad in finite model theory. In *32nd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2017, Reykjavik, Iceland, June 20-23, 2017*, pages 1–12. IEEE Computer Society, 2017.

2 Samson Abramsky and Radha Jagadeesan. New foundations for the geometry of interaction. *Inf. Comput.*, 111(1):53–119, 1994. doi:10.1006/inco.1994.1041.

3 Samson Abramsky and Nihil Shah. Relating structure and power: Comonadic semantics for computational resources. *J. Log. Comput.*, 31(6):1390–1428, 2021. doi:10.1093/logcom/exab048.

4 Aurore Alcolei. *Enriched concurrent games : witnesses for proofs and resource analysis. (Jeux concurrents enrichis : témoins pour les preuves et les ressources)*. PhD thesis, University of Lyon, France, 2019. URL: https://tel.archives-ouvertes.fr/tel-02448974.

5 Pierre Clairambault and Marc de Visme. Full abstraction for the quantum lambda-calculus. *Proc. ACM Program. Lang.*, 4(POPL):63:1–63:28, 2020. doi:10.1145/3371131.

6 Pierre Clairambault, Marc de Visme, and Glynn Winskel. Game semantics for quantum programming. *Proc. ACM Program. Lang.*, 3(POPL):32:1–32:29, 2019. doi:10.1145/3290345.

7 John Conway. *On Numbers and Games*. Wellesley, MA: A K Peters, 2000.

8 Anuj Dawar, Tomás Jakl, and Luca Reggio. Lovász-type theorems and game comonads. In *36th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2021, Rome, Italy, June 29 - July 2, 2021*, pages 1–13. IEEE, 2021. doi:10.1109/LICS52264.2021.9470609.

9 Jean-Yves Girard. Geometry of interaction i: Interpretation of system f. In C. Bonotto, S R. Ferro, Valentini, and A. Zanardo, editors, *Logic Colloquium '88*, page 221–260. North-Holland, 1989.

10 Andre Joyal. Remarques sur la théorie des jeux à deux personnes. *Gazette des sciences mathématiques du Québec, 1(4)*, 1997.

11 Yoàv Montacute and Nihil Shah. The pebble-relation comonad in finite model theory. In *Proceedings of the 37th Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '22, New York, NY, USA, 2022. Association for Computing Machinery. doi:10.1145/3531130.3533335.

12 Mogens Nielsen, Gordon Plotkin, and Glynn Winskel. Petri nets, event structures and domains. *TCS*, 13:85–108, 1981.

13 Glynn Winskel. *Events in computation*. 1980. PhD thesis, University of Edinburgh.

14 Glynn Winskel. Event structures. In *Advances in Petri Nets*, volume 255 of *LNCS*, pages 325–392. Springer, 1986.

15 Glynn Winskel. Probabilistic and quantum event structures. In *Festschrift for Prakash Panangaden*, volume 8464 of *Lecture Notes in Computer Science*, pages 476–497. Springer, 2014.

16 Glynn Winskel. *ECSYM Notes: Event Structures, Stable Families and Concurrent Games*. http://www.cl.cam.ac.uk/∼gw104/ecsym-notes.pdf, 2016.

The category of MSO transductions

Mikołaj Bojańczyk

May 29, 2023

This talk is about the connection between logic and automata, which is one of the central subjects of logic in computer science and formal language theory. The original result, due to Büchi, Elgot and Trakhtenbrot (see [14, Theorem 3.1] in the survey of Thomas), says that for finite words, the languages recognized by finite automata (or equivalently, semigroups) are the same as those definable in monadic second-order logic (MSO). This result has seen countless extensions, to objects such as trees or graphs, and also for infinite objects; see [14, 2] for surveys. In some cases, the recognisability side is best modeled by automata (e.g. nondeterministic automata for infinite trees, as surveyed in [14, Section 6]). In other cases it is best described by algebras (e.g. Courcelle's algebras for graphs [9], or generalizations of semigroups for countable infinite words [5, Section 3]). In some cases, both automata and algebras are useful, e.g. finite words and trees.

Since the relationship between recognisability and definability (in MSO) can be studied in so many settings, and since these studies share many techniques, it is natural to search for common generalizations. An early example of this kind is a paper of Eilenberg and Wright [10], which studies automata in general algebras, i.e. algebras that do not necessarily describe finite words. The underlying definition is Lawvere theories, which are the same as finitary monads. Although the early results of Eilenberg and Wright were about free theories, which are the same as automata on ranked trees, the abstract framework has been further developed to handle other kinds of objects, as witnessed by variety theorems for general algebras that were proposed in [13, 2, 6]. An attempt to connect logic and recognisability in the abstract setup of monads was made in [4]. In that work, the point of departure is recognizability, and the connection with MSO is obtained by simulating logical operations (such as Boolean combinations or set quantification) using operations on algebras (such as product or powerset).

In this talk, we take the opposite approach. Our point of departure is MSO logic, and we connect it to recognizability by framing recognizability in terms of logic. For many kinds of structures, the logical aspect is simpler than the algebra aspect, hence the usefulness of reducing algebra to logic in a uniform way. An example of this phenomenon is graphs. It is clear how MSO logic can be applied to graphs (although there are certain design decisions to be made, e.g. whether or not the set quantifiers can range over sets of edges). In contrast, the algebraic approach to graphs, namely Courcelle's HR and VR algebras [9, Section 2.3 and 2.5], is arguably less natural and requires a non-trivial amount of book-keeping, such as infinitely many sorts and extra anno-

tation with distinguished vertices or colours. A similar situation arises for matroids, where MSO definability is a clear notion, while a definition of recognizability requires a careful analysis of how a matroid can be parsed by an automaton, see [12, Section 3].

Our approach of reducing recognizability to logic is based on MSO transductions [1, 7, 11]. These are binary relations between structures. The structures could be strings, trees, graphs, etc. The inputs and outputs need not have the same type, e.g. a string-to-graph transduction inputs a string and outputs a graph, with the output graph being not necessarily unique. Two key properties of MSO transductions are: (a) they are closed under composition; and (b) in the case of languages, i.e. MSO transductions with Boolean outputs, they coincide with MSO definable languages. Thanks to (a), the MSO transductions form a category (objects are classes of structures, and morphisms are MSO transductions), and thanks to (b), this category describes MSO definable properties. (This work is not intedned to study this category from the point of view of category theory.)

We argue that many notions from language theory can be described in terms of this category. Our point of departure is based on a result of Courcelle and Engelfriet [8, Theorem 2.6], which characterizes treewidth and cliquewidth purely in terms of MSO transductions. This characterization says that a class of graphs has bounded treewidth if and only if it is contained in the set of outputs of some tree-to-graph MSO transduction. For cliquewidth, the characterization is the same, only with a different representation of graphs as logical structures. Motivated by this result, we propose the following notion of width.

Bounded width. A class of structures has *bounded MSO width* if it is contained in the image of an MSO transduction that inputs trees.

The characterizations of Courcelle and Engelfriet show that this notion of width captures treewidth and cliquewidth for graphs; we show that it also captures branchwidth for matroids representable over finite fields, as well as other notions of recognisability, including strings, trees and Mazurkiewicz traces. The essential idea behind the definition is that we do not fix any pre-defined semantics for a tree decomposition, but allow instead any semantics that can be formalized by an MSO transduction (as is the case for known kinds of tree decompositions). This level of generality greatly simplifies notation, while retaining all applications of tree decompositions that are related to MSO.

The logical definition of bounded width described above is the obvious generalization of the characterizations of Courcelle and Engelfriet [8], and not a new idea on its own. However, the transduction approach can also capture other concepts, such as recognizability or having definable tree decompositions. Finally, we propose a new order on classes of structures, called *MSO encodings*, which is meant to play the same role as reductions in complexity theory. The paper is available at arxiv [3].

References

[1] Stefan Arnborg, Jens Lagergren, and Detlef Seese. Problems easy for tree-decomposable graphs extended abstract. In Timo Lepistö and Arto Salomaa, editors, *Automata, Languages and Programming*, pages 38–51, Berlin, Heidelberg, 1988. Springer Berlin Heidelberg.

[2] Mikołaj Bojańczyk. Languages recognised by finite semigroups, and their generalisations to objects such as trees and graphs, with an emphasis on definability in monadic second-order logic. `https://www.mimuw.edu.pl/bojan/papers/algebra-26-aug-2020.pdf`, 2020.

[3] Mikołaj Bojańczyk. The category of MSO transductions. arXiV, 2023.

[4] Mikołaj Bojańczyk, Bartek Klin, and Julian Salamanca. Monadic monadic second order logic. In *Samson Abramsky on Logic and Structure in Computer Science and Beyond*. Springer, 2023.

[5] Olivier Carton, Thomas Colcombet, and Gabriele Puppis. Regular Languages of Words over Countable Linear Orderings. In Luca Aceto, Monika Henzinger, and Jiří Sgall, editors, *Automata, Languages and Programming*, Lecture Notes in Computer Science, pages 125–136. Springer Berlin Heidelberg, 2011.

[6] Liang-Ting Chen, Jiří Adámek, Stefan Milius, and Henning Urbat. Profinite monads, profinite equations, and reiterman's theorem. In *International Conference on Foundations of Software Science and Computation Structures*, pages 531–547. Springer, 2016.

[7] Bruno Courcelle. The monadic second-order logic of graphs v: on closing the gap between definability and recognizability. *Theoretical Computer Science*, 80(2):153 – 202, 1991.

[8] Bruno Courcelle and Joost Engelfriet. A logical characterization of the sets of hypergraphs defined by hyperedge replacement grammars. *Mathematical Systems Theory*, 28(6):515–552, 1995.

[9] Bruno Courcelle and Joost Engelfriet. *Graph Structure and Monadic Second-Order Logic - A Language-Theoretic Approach*, volume 138 of *Encyclopedia of Mathematics and Its Applications*. Cambridge University Press, 2012.

[10] Samuel Eilenberg and Jesse Wright. Automata in General Algebras. *Information and Control*, 11:452–470, 1967.

[11] Joost Engelfriet. A characterization of context-free nce graph languages by monadic second-order logic on trees. In Hartmut Ehrig, Hans-Jörg Kreowski, and Grzegorz Rozenberg, editors, *Graph Grammars and Their Application to Computer Science*, pages 311–327, Berlin, Heidelberg, 1991. Springer Berlin Heidelberg.

[12] Petr Hlinený. Branch-width, parse trees, and monadic second-order logic for matroids. *J. Comb. Theory, Ser. B*, 96(3):325–351, 2006.

[13] Magnus Steinby. General varieties of tree languages. *Theoretical Computer Science*, 205(1):1–43, September 1998.

[14] Wolfgang Thomas. Languages, automata, and logic. In *Handbook of formal languages, Vol. 3*, pages 389–455. Springer, Berlin, 1997.

Enumerating Error Bounded Polytime Algorithms Through Arithmetical Theories*

Ugo Dal Lago

The study of complexity classes through mathematical logic dates back to the seventies [15, 9], but grew in scope especially from in the eighties and nineties [4, 17, 3, 21, 23]. The logical characterization of several crucial classes has made it possible to consider them from a new viewpoint, less dependent on concrete machine models and explicit resource bounds. Characterizing complexity classes by way of simple enough proof-or-recursion theoretical systems also means being able to *enumerate* the problems belonging to them, and thus to devise sound and complete languages for the class, from which type systems and static analysis methodologies can be derived [19].

Among the various classes of problems with which computational complexity has been concerned, those defined on the basis of *randomized* algorithms [26] have appeared difficult to capture with the tools of logic. These include important and well-studied classes like **BPP** or **ZPP**. The former, in particular, is often considered as *the* class of feasible problems, and most complexity theorists conjecture that it actually coincides with **P**. However, by simply looking at its definition, **BPP** looks pretty different from **P**. Notably, the former, but not the latter, is an example of what is usually called a *semantic* class: the definition of **BPP** relies on algorithms which are both efficient and not *too erratic*; in other words, once an input is fixed, one of the two possible outputs must clearly prevail over the other, i.e. it must occur with some probability, *strictly* greater than one half, independent from the input. By their very nature, semantic classes, like **BPP**, are more challenging to be logically captured with respect to other (syntactic) classes, like **P** or **PP**: indeed, *enumerating* them through the underlying machines is harder. Currently, it is simply not known whether an effective enumeration of the aforementioned randomized classes is possible. Indeed, the sparse contributions along these lines are either themselves *semantic*, i.e. do not capture the limitations on the probability of error within the logical system [14, 13] (an interesting exception being [22]), or deal with classes, as **PP**, which are not classifiable as semantic [11, 12].

In this work, we make a step towards a logical characterization of randomized classes by considering a language in which the probability of error can be kept under control *from within* the logic. We introduce a language, called $\mathcal{RL}$, inspired by the one presented in Ferreira's [16], but in which formulas can access a source of randomness through a distinguished unary predicate `Flip`, this way naturally capturing randomized algorithms.

We define a bounded theory of arithmetic, called $R\Sigma_1^b-$NIA, as the randomized analogue of Buss' S_2^1 [4] and Ferreira's Σ_1^b-NIA [16], and show that the functions which can be proved total in $R\Sigma_1^b-$NIA are precisely the polytime *random* functions [27], i.e. those functions from strings to *distributions* of strings which can be computed by polytime probabilistic Turing machines (PTM, for short). Using this result, we provide two characterizations of the problems in **BPP**: one relies on *measure quantifiers* [25, 24, 1], i.e. well-studied second-order quantifiers capable of measuring *the extent* to which a formula is true; the other consists in showing that these quantifiers, when applied to bounded formulas, can be encoded via *standard* first-order quantification.

Both these approaches provide precise characterizations of **BPP** but are still semantic in nature: the entropy check is translated into conditions which are not based on provability in some formal system, but rather on the truth of some formula in the standard model of first-order arithmetic. Our arithmetization of **BPP**, however, naturally leads to the introduction of a family of new *syntactic* subclasses of **BPP**, namely $\mathbf{BPP}_\mathsf{T}$, made of languages for which the

*The talk is based on a joint work with Melissa Antonelli, Davide Davoli, Isabel Oitavem, and Paolo Pistone

24

error-bounding condition is *provable* in a (non necessarily bounded) theory T. We show that full first-order Peano Arithmetic (PA, for short) provides an interesting candidate theory, as $\mathbf{BPP}_{\mathsf{PA}}$ includes *polynomial identity testing* (PIT), which is one of the few problems in **BPP** currently not known to be in **P**. This fact seems very promising, suggestingan avenue towards a new form of *reverse* computational complexity in the framework of first-order arithmetic.

The main technical contributions of our work can then be summarized as follows:

- We introduce the arithmetical theory $R\Sigma_1^b-$NIA and prove that the functions which are Σ_1^b-representable in it are precisely the random functions which can be computed in polynomial time. The proof of the correspondence goes through the definition of a class of *oracle recursive* functions, called $\mathcal{POR}$, which is shown equivalent to that of functions which are Σ_1^b-representable in $R\Sigma_1^b-$NIA and, then, to the class of probabilistic polytime random functions.
- Then, we use the aforementioned result to obtain a new syntactic characterization of **PP** and, more interestingly, two semantic characterizations of **BPP**, the first based on measure quantifiers and the second relying on standard, first-order quantification.
- Finally, we introduce a family of syntactic subclasses $\mathbf{BPP}_{\mathsf{T}} \subseteq \mathbf{BPP}$, parametrically on a theory T. The core idea is to consider a (sound) theory T in which error-bound checks can be syntactically considered, this way potentially restricting the class of problems to be captured. We then prove that PIT is in $\mathbf{BPP}_{\mathsf{PA}}$, thus identifying a nontrivial effectively enumerable subclass of **BPP**. We believe this to be the most interesting and potentially impactful of the presented results. We conclude by showing how our approach relates to the literature on capturing **BPP** languages in bounded arithmetic [22].

References

[1] M. Antonelli, U. Del Lago, and P. Pistone. On Measure Quantifiers in First-Order Arithmetic. In L. De Mol, A. Weiermannn, F. Manea, and D. Fernández-Duque, editors, *Connecting with Computabiliy*, pages 12–24. Springer, 2021.

[2] S. Arora and B. Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009.

[3] S. Bellantoni and S. Cook. A New Recursion-Theoretic Characterization of the Polytime Functions. *Computational Complexity*, 2:97–110, 1992.

[4] S.R. Buss. *Bounded Arithmetic*. PhD thesis, Princeton University, 1986.

[5] A. Church. An Unsolvable Problem of Elementary Number Theory. *American J. of Mathematics*, 58(2):345–363, 1992.

[6] A. Cobham. The Intrinsic Computational Difficulty of Functions. In Y. Bar-Hillel, editor, *Logic, Methodology and Philosophy of Science: Proc. of the 1964 International Congress (Studies in Logic and the Foundations of Mathematics)*, pages 24–30. North-Holland Publishing, 1965.

[7] E.F. Codd. Relational Completeness of Data Base Sublanguages. In *Data Base Systems. Proc. of 6th Courant Computer Science Symposium, May 24-25, 1971, New York, N.Y.*, pages 65–98, 1972.

[8] S. Cook. The Complexity of Theorem Proving Procedures. In *Proc. Third Annual ACM Symposium on Theory of Computing*, pages 151–158, 1971.

[9] S.A. Cook and R.A. Reckhow. Efficiency of Propositional Proof Systems. *Journal of Symbolic Logic*, 44(1):36–50, 1979. 25

[10] H. B. Curry. Functionality in Combinatory Logic*. *Proc. of the National Academy of Sciences*, 20(11):584–590, 1934.

[11] U. Dal Lago, R. Kahle, and I. Oitavem. A Recursion-Theoretic Characterization of the probabilistic Class PP. In F. Bonchi and S.J. Puglisi, editors, *46th International Symposium on Mathematical Foundations of Computer Science (MFCS 2021)*, volume 202 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 1–12. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021.

[12] U. Dal Lago, R. Kahle, and I. Oitavem. Implicit Recursion-Theoretic Characterizations of Counting Classes. *Archive for Mathematical Logic*, May 2022.

[13] U. Dal Lago and P. Parisen Toldin. A Higher-Order Characterization of Probabilistic Polynomial Time. *Information and Computation*, 241:114–141, 2015.

[14] K. Eickmeyer and M. Grohe. Randomisation and Derandomisation in Descriptive Complexity Theory. In A. Dawar and H. Veith, editors, *Computer Science Logic*. Springer Berlin Heidelberg, 2010.

[15] R. Fagin. Generalized First-Order Spectra and Polynomial-Time Recognizable Sets. In *Complexity of Computing: SIAM-AMS Proc.*, pages 43–73, 1974.

[16] F. Ferreira. Polynomial-Time Computable Arithmetic. In W. Sieg, editor, *Logic and Computation*, volume 106 of *Contemporary Mathematics*, pages 137–156. AMS, 1990.

[17] J.-Y. Girard, A. Scedrov, and P. Scott. Bounded Linear Logic: A Modular Approach to Polynomial-Time Computability. *Theoretical Computer Science*, 97(1):1–66, 1992.

[18] J. Hartmanis and R.E. Stearns. On the Computational Complexity of Algorithms. *Transactions of the AMS*, 117:285–306, 1965.

[19] M. Hofmann. Programming Languages Capturing Complexity Classes. *SIGACT News*, 31(1):31–42, mar 2000.

[20] H. A. Howard. The Formulae-as-Types Notion of Construction. In *To H. B. Curry: Essays on Combinatory Logic, Lambda Calculus, and Formalism*. Academic Press, 1980.

[21] N. Immerman. *Descriptive Complexity*. Springer, 1999.

[22] E. Jeràbek. Approximate Counting in Bounded Arithmetic. *Journal of Symbolic Logic*, 72(3):959–993, 2007.

[23] J. Krajicek and P. Pudlak. Propositional Proof Systems, the Consistency of First-Order Theories and the Complexity of Computations. *Journal of Symboic Logic*, 54(3):1063–1079, 1989.

[24] H. Michalewski and M. Mio. Measure Quantifiers in Monadic Second Order Logic. In *Proc. of Logical Foundations of Computer Science*, pages 267–282, Cham, 2016. Springer.

[25] C. Morgenstern. The Measure Quantifier. *Journal of Symbolic Logic*, 44(1):103–108, 1979.

[26] R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge University Press, Cambridge; NY, 1995.

[27] E.S. Santos. Probabilistic Turing Machines and Computability. *AMS*, 22(3):704–710, 1969.

[28] M.H. Sorensen and P. Urzyczyn. *Lectures on the Curry-Howard Isomorphism*. Elsevier, 2006.

[29] A. Turing. On Computable Numbers, with an Application to the Entscheidungsproblem. *Proc. London Mathematical Society*, pages 2–42, 230–265, 1936-37.

Higher-dimensional subdiagram matching

Diana Kessler
Tallinn University of Technology
diana-maria.kessler@taltech.ee

Joint work with Amar Hadzihasanovic.

Abstract of arXiv:2304.09216. Accepted for LICS 2023.

Higher-dimensional rewriting [2, 3] is founded on a duality of rewrite systems and cell complexes, connecting computational mathematics to higher categories and homotopy theory: the two sides of a rewrite rule are two halves of the boundary of an $(n+1)$-cell, which are (pasting) diagrams of n-cells. Proofs by diagrammatic rewriting can be used in the formalisation of homotopical algebra and higher category theory, provided that one has a *topologically sound* formal model, where diagrams admit a functorial interpretation as homotopies in cell complexes.

Beyond the formalisation of mathematics, it is interesting to look at higher-dimensional diagram rewriting as a model of computation in itself. String diagram rewriting, a form of 3-dimensional rewriting, is arguably the characteristic computational mechanism of applied category theory. It has been suggested [1] that even "classical" forms of rewriting are more faithfully represented as diagram rewriting: for example, term rewriting implemented as rewriting in monoidal categories with cartesian structure explicitates the "hidden costs" of copying and deleting terms.

We continue the work started in [5] studying higher-dimensional rewriting as a *mechanism of computation*, and try to answer the question: given a machine that operates by higher-dimensional rewriting, is the obvious cost model that attributes constant cost to each rewrite step a "reasonable" cost model?

The basic computational step of any such machine may be described as follows. The machine has a list $(r_i)_{i=1}^m$ of rewrite rules, which are $(n+1)$-cells, and whose input boundaries $(\partial^- r_i)_{i=1}^m$ and output boundaries $(\partial^+ r_i)_{i=1}^m$ are n-dimensional diagrams. Given an n-dimensional diagram t as input, the machine tries to match one of the input boundaries with a rewritable *subdiagram* of t. If it finds a match with $\partial^- r_i$ for some $i \in \{1,\dots,m\}$, it substitutes $\partial^+ r_i$ for the match in t; otherwise it stops.

Evidently, our question is answered in the affirmative for such a machine if and only if the *subdiagram matching problem* admits a (preferably low-degree) polynomial-time algorithm with respect to a reasonable size measure for diagrams. Since a cognate problem such as subgraph matching is notoriously NP-complete, it is not at all obvious that this should be true.

We study the higher-dimensional subdiagram matching problem in the "topologically sound" *diagrammatic sets* model [4]. Our main contribution is an algorithm for subdiagram matching in arbitrary dimension. On top of this, we are also giving an improved running-time for the traversal algorithm presented in [5].

In the first part of the talk, I will introduce the framework in which we are working and the first part of the algorithm. In the theory of diagrammatic sets, we represent the shape of a diagram with its *face poset* which records whether an $(n-1)$-dimensional cell is in the input or output half of the boundary of an n-dimensional cell. We call this structure an *oriented graded poset* and represent it as a Hasse diagram with oriented edges. The well-formed shapes of diagrams form an inductive subclass of oriented graded posets called *regular molecules*. A diagram, in our setting, is then a labelling $t\colon U \to \mathbb{V}$ of a regular molecule U into a set of variables.

Submitted to:
Structure meets Power 2023

Given diagrams $s\colon V \to \mathbb{V}$ and $t\colon U \to \mathbb{V}$, the subdiagram matching problem can be split into three subproblems:

1. find, if any, the inclusions of the shape of s into the shape of t;
2. decide if an inclusion is a *subdiagram* inclusion;
3. check that the labelling is preserved.

The subdiagram inclusions are those corresponding to portions of a diagram that can be rewritten producing another well-formed diagram; these are a proper subclass of all inclusions. The third problem is easy, so we focus on the first two.

For a fixed regular molecule, U, we let U_n be the subset of n-dimensional elements in U, $|U|$ be the number of elements in U, $|U_n|$ the number of n-dimensional elements in U, $|\mathscr{E}_n U|$ the number of edges between n and $(n-1)$-dimensional elements in the Hasse diagram of U, and $|\mathscr{E}_\vee U|$ the maximum of the $|\mathscr{E}_n U|$ if non-zero, 1 otherwise.

For the first problem, we obtain an algorithm with the following polynomial time bound.

Theorem 1 — *The problem of finding all inclusions in dimension n can be solved in time*

$$O(|U_n|\,|V_n|\,|V|\,|\mathscr{E}_\vee V|\log|\mathscr{E}_\vee V|).$$

The second problem turns out to be highly non-trivial. Our best general solution has the following worst-case time complexity upper bound.

Theorem 2 — *The problem of deciding if an inclusion is a subdiagram inclusion in dimension n can be solved in time*

$$O\left(\prod_{k\le n} |U_k|!\,|U_k|\right).$$

The algorithm relies on new combinatorial results on *layerings* of diagrams, which are decompositions into layers containing each a single cell of the highest dimension. One can associate to each n-dimensional diagram a directed acyclic graph whose vertex set is U_n, such that each layering induces a distinct topological sort, which we call an *ordering*. The factorial factors come from the fact that our algorithm searches for a valid layering by trying out different orderings, which are as many as topological sorts of the graph, which are factorial in the number of vertices in the worst case of a discrete graph.

On the other hand, if we restrict our attention to a special class of diagrams that we call *stably-frame acyclic*, then there is a perfect correspondence between layerings and orderings, and we can avoid this iteration.

Theorem 3 — *If a diagram is guaranteed to be stably frame-acyclic, the problem of deciding if an inclusion is a subdiagram inclusion can be solved in linear time in the size of its Hasse diagram.*

Fortunately, the following holds.

Theorem 4 — *Every diagram of dimension ≤ 3 is stably frame-acyclic.*

We conclude that subdiagram matching is feasible up to dimension 3.

In the 4-dimensional case, not every shape of diagram is stably frame-acyclic and the main problem to overcome is the expensive iteration on orderings. The problem seems to arise from the possibility of rewrites causing non-local obstructions which may prevent other "disjoint" rewrites to be applied. This behaviour may hint to a potential gap in complexity between the 3-dimensional and 4-dimensional cases. We leave the existence of a polynomial time algorithm for subdiagram matching in dimension 4 or higher as an open problem and we hope that a deeper understanding of such cases will either lead to an improved algorithm or a proof of NP-completeness.

References

[1] G. Bonfante & Y. Guiraud (2009): *Polygraphic programs and polynomial-time functions*. *Logical Methods in Computer Science* 5(2), doi:10.2168/lmcs-5(2:14)2009. Available at `https://doi.org/10.2168%2Flmcs-5%282%3A14%292009`.

[2] A. Burroni (1993): *Higher-dimensional word problems with applications to equational logic*. *Theoretical Computer Science* 115(1), pp. 43–62, doi:10.1016/0304-3975(93)90054-w. Available at `https://doi.org/10.1016%2F0304-3975%2893%2990054-w`.

[3] Y. Guiraud (2019): Rewriting methods in higher algebra. Thèse d'habilitation à diriger des recherches, Université Paris 7.

[4] A. Hadzihasanovic (2020): *Diagrammatic sets and rewriting in weak higher categories*. *arXiv preprint arXiv:2007.14505*.

[5] A. Hadzihasanovic & D. Kessler (2022): *Data structures for topologically sound higher-dimensional diagram rewriting*. In: *Proceedings Fifth International Conference on Applied Category Theory (ACT2022)*.

FINITELY ACCESSIBLE ARBOREAL ADJUNCTIONS AND HINTIKKA FORMULAE

COLIN RIBA (J.W.W. LUCA REGGIO)

Abstract. Arboreal categories provide an axiomatic framework in which abstract notions of bisimilarity and back-and-forth games can be defined. They act on extensional categories, typically consisting of relational structures, via arboreal adjunctions. In the examples, equivalence of structures in various fragments of infinitary first-order logic can be captured by transferring bisimilarity along the adjunction.

In most applications, the categories involved are locally finitely presentable and the adjunctions finitely accessible. Drawing on this insight, we identify the expressive power of this class of adjunctions. We show that the ranks of back-and-forth games in the arboreal category are definable by formulae *à la* Hintikka. Thus, the relation between extensional objects induced by bisimilarity is always coarser than equivalence in infinitary first-order logic. Our approach relies on Gabriel-Ulmer duality for locally finitely presentable categories, and Hodges' word-constructions.

This extended abstract is based on the preprint [27].

Introduction. Model comparison games are one of the few tools of model theory which is still available in finite model theory, see e.g. [19, 14, 26]. Important examples include Ehrenfeucht-Fraïssé games for first-order equivalence (possibly with bounded quantifier rank), pebble games for the finite-variable fragments, and modal bisimulation games for the modal fragment.

Game comonads. We are interested in abstract notions of model comparison games provided by *arboreal categories* [6, 5], an axiomatic framework for *games comonads*. Game comonads were introduced in [2] as a categorical syntax-free approach to (finite) model theory, in which coalgebras encode important combinatorial parameters of structures (see [1] for a recent survey). A crucial step for their development was taken in [7, 8], where various back-and-forth games (such as Ehrenfeucht-Fraïssé, pebble, and modal bisimulation games) were captured by means of a refinement of Joyal, Nielsen and Winskel's notion of bisimulation via open maps [22, 23, 24].

Arboreal adjunctions. The framework of arboreal categories takes as input a class $\mathcal{M}$ of monomorphisms, called *embeddings*, and provides concrete notions of bisimilarity and back-and-forth games. An arboreal category $\mathcal{A}$ acts on an extensional category $\mathcal{E}$ (typically consisting of relational structures) via an *arboreal adjunction*:

$$\mathcal{A} \underset{R}{\overset{L}{\rightleftarrows}} \mathcal{E} \qquad (1)$$

Transferring the bisimilarity relation along the adjunction, one recovers relevant logical equivalences of structures in $\mathcal{E}$. In many applications, the categories involved are locally finitely presentable, the arboreal adjunctions are finitely accessible and the logics are fragments of infinitary first-order logic $\mathcal{L}_{\infty,\omega}$. In fact, the full logic $\mathcal{L}_{\infty,\omega}$ is captured by an arboreal adjunction.

Hintikka formulae for finitely accessible arboreal adjunctions. Our main result yields a converse to this for finitely accessible arboreal adjunctions. Under mild assumptions, we show that for each $M \in \mathcal{E}$, the ranks of the back-and-forth games played in RM are definable in $\mathcal{L}_{\infty,\omega}$. Hence, arboreal bisimilarity cannot distinguish between $\mathcal{L}_{\infty,\omega}$-equivalent models. This identifies the expressive power of finitely accessible arboreal adjunctions. Note that while the isomorphism type of a finite structure is definable in finitary first-order logic, Corollary 8 in [2] (for pebble games) forces $\mathcal{E}$ to contain infinite objects in general, on which $\mathcal{L}_{\infty,\omega}$ is strictly more expressive.

Our approach relies on Gabriel-Ulmer duality for locally finitely presentable categories [16] (cf. also [10, 9]): each locally finitely presentable category is equivalent to the category of models of a theory written using only finite conjunctions and provably unique existential quantifiers. The back-and-forth games in $\mathcal{A}$ are closed in the usual sense, and are equipped with a customary notion of ordinal rank of positions (cf. e.g. [19, §3.4] or [25, §20B]). Working with Coste's syntactic categories [13] (cf. also [21, §§D1–2]), we devise "Hintikka formulae" for RM for each ordinal rank, along the usual pattern (cf. e.g. [19, §3.5] or [14, §3]). These formulae are then translated to formulae for M in $\mathcal{E}$ along an interpretation induced by the right adjoint R.

In most applications, the class $\mathcal{M}$ of embeddings in $\mathcal{A}$ is determined by the class of substructure embeddings in $\mathcal{E}$. The Hintikka formulae for RM in $\mathcal{A}$ then depend on formulae over LRM in $\mathcal{E}$. To translate the infinitary theory of LRM to the infinitary theory of RM, we use a slight generalization of Hodges' *word-constructions* [17, 18] (cf. also [19, §9.3]). This yields the following result. A *path* is an object P of $\mathcal{A}$ whose $\mathcal{M}$-subobjects form a finite chain.

Theorem ([27])**.** *Assume* (1) *is a finitely accessible arboreal adjunction in which $\mathcal{E}$ is the category of models of a theory of signature $\mathbf{\Sigma}$, and such that the paths of $\mathcal{A}$ are finitely presentable. Assume further that a morphism $f\colon P \to a$ of $\mathcal{A}$ is an embedding if and only if Lf is a substructure embedding in $\mathcal{E}$. Given models $M, N \in \mathcal{E}$, if M and N are $\mathcal{L}_{\infty,\omega}(\mathbf{\Sigma})$-equivalent, then RM and RN are bisimilar in $\mathcal{A}$.*

Our assumptions are satisfied by the comonadic approach to Ehrenfeucht-Fraïssé games. This applies also to pebble and modal bisimulation games [8, 5], and we expect that the same methodology will extend to guarded fragments [3] and hybrid logic [4]. On the other hand, our results show that game comonads for MSO [20] cannot lead to finitely accessible arboreal adjunctions with the obvious choice of embeddings.

Future work. Our results suggest that in order to handle stronger logics, one would have to consider κ-accessible arboreal adjunctions for uncountable regular cardinals κ. We believe this is a natural line for future investigation. In particular, games for generalized Lindström quantifiers [11, 12] may be instrumental in developing a systematic study. Other interesting examples to look at include [15].

In a different direction, having provided in [27] novel examples of arboreal categories based on presheaves, we intend to carry out a thorough comparison with bisimulation via open maps [24].

Acknowledgment. Research supported by the EPSRC grant EP/V040944/1 and the ANR project ReCiProg (ANR-21-CE48-0019).

References

1. S Abramsky, *Structure and Power: an Emerging Landscape*, Fundam. Informaticae **186** (2022), no. 1-4, 1–26.

2. S. Abramsky, A. Dawar, and P. Wang, *The Pebbling Comonad in Finite Model Theory*, Proceedings of LICS'17, 2017.
3. S. Abramsky and D. Marsden, *Comonadic semantics for guarded fragments*, Proceedings of LICS'21, 2021, pp. 1–13.
4. ———, *Comonadic semantics for hybrid logic*, 47th International Symposium on Mathematical Foundations of Computer Science, MFCS 2022, August 22-26, 2022, Vienna, Austria, 2022, pp. 7:1–7:14.
5. S. Abramsky and L. Reggio, *Arboreal Categories: An Axiomatic Theory of Resources*, abs/2102.08109, 2021.
6. ———, *Arboreal Categories and Resources*, 48th International Colloquium on Automata, Languages, and Programming, ICALP 2021, July 12-16, 2021, Glasgow, Scotland (Virtual Conference), 2021, pp. 115:1–115:20.
7. S. Abramsky and N. Shah, *Relating structure and power: Comonadic semantics for computational resources*, Proceedings of CSL'18, 2018, pp. 2:1–2:17.
8. ———, *Relating structure and power: Comonadic semantics for computational resources*, J. Log. Comput. **31** (2021), no. 6, 1390–1428.
9. J. Adámek and H.-E. Porst, *Algebraic Theories of Quasivarieties*, Journal of Algebra **208** (1998), no. 2, 379 – 398.
10. J. Adámek and J. Rosický, *Locally Presentable and Accessible Categories*, London Mathematical Society Lecture Notes Series, vol. 189, Cambridge University Press, 1994.
11. X. Caicedo, *Back-and-Forth Systems for Arbitrary Quantifiers*, Mathematical Logic in Latin America (A.I. Arruda, R. Chuaqui, and N.C.A. Da Costa, eds.), Studies in Logic and the Foundations of Mathematics, vol. 99, Elsevier, 1980, pp. 83–102.
12. A. Ó Conghaile and A. Dawar, *Game Comonads & Generalised Quantifiers*, 29th EACSL Annual Conference on Computer Science Logic, CSL, vol. 183, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021, pp. 16:1–16:17.
13. M. Coste, *Une approche logique des théories définissables par limites projectives finies*, Séminaire Bénabou, Université Paris-Nord, 1976.
14. H.-D. Ebbinghaus and J. Flum, *Finite model theory*, second ed., Springer Monographs in Mathematics, Springer, 1999.
15. C. Ford, S. Milius, L. Schröder, H. Beohar, and B König, *Graded Monads and Behavioural Equivalence Games*, Proceedings of LICS'22 (Christel Baier and Dana Fisman, eds.), ACM, 2022, pp. 61:1–61:13.
16. P. Gabriel and F. Ulmer, *Lokal Präsentierbare Kategorien*, Lecture Notes in Mathematics, vol. 221, Springer, 1971.
17. W. Hodges, *A normal form for algebraic constructions*, Bulletin of the London Mathematical Society **6** (1974), no. 1, 57–60.
18. ———, *A normal form for algebraic constructions II*, Logique et Analyse **18** (1975), no. 71/72, 429–487.
19. ———, *Model theory*, Encyclopedia of Mathematics and its Applications, vol. 42, Cambridge University Press, 1993.
20. T. Jakl, D. Marsden, and N. Shah, *A game comonadic account of Courcelle and Feferman-Vaught-Mostowski theorems*, https://arxiv.org/abs/2205.05387, 2022.
21. P.T. Johnstone, *Sketches of an Elephant: A Topos Theory Compendium*, Oxford Logic Guides, Clarendon Press, 2002.
22. A. Joyal and I. Moerdijk, *A completeness theorem for open maps*, Ann. Pure Appl. Log. **70** (1994), no. 1, 51–86.
23. A. Joyal, M. Nielsen, and G. Winskel, *Bisimulation and open maps*, Proceedings of LICS'93, 1993, pp. 418–427.
24. ———, *Bisimulation from Open Maps*, Inf. Comput. **127** (1996), no. 2, 164–185.
25. A. S. Kechris, *Classical Descriptive Set Theory*, Graduate Texts in Mathematics, vol. 156, Springer, 1995.
26. L. Libkin, *Elements of Finite Model Theory*, Texts in Theoretical Computer Science. An EATCS Series, Springer Berlin, Heidelberg, 2004.
27. L. Reggio and C. Riba, *Finitely accessible arboreal adjunctions and Hintikka formulae*, https://arxiv.org/abs/2304.12709, 2023.

Profinite λ-terms and parametricity

Vincent Moreau

Université Paris Cité, IRIF, Inria Paris

The aim of this work is to combine profinite methods and models of the λ-calculus to obtain a notion of *profinite λ-term*. First steps in that direction were already presented at *Structure meets Power 2022*, although they were mainly focused on the relation between profinite words and Reynolds parametricity. This abstract presents new work relating profinite λ-terms to Salvati's notion of regular language of λ-terms, through Stone duality.

This is joint work with Sam van Gool and Paul-André Melliès [2].

Languages of finite words and profiniteness. Automata theory has a central role in theoretical computer science. In its most basic form, it deals with regular languages of finite words. If M is a finite monoid and $\varphi : \Sigma^* \to M$ is a monoid homomorphism, then each subset S of M induces the regular language

$$L_S \quad := \quad \{w \in \Sigma^* \mid \varphi(w) \in S\} \ ,$$

that is the set of words which, when interpreted in the monoid M with the morphism φ, yield an element of S. We recover all the regular languages in this way:

$$\mathrm{Reg}\langle\Sigma\rangle \quad = \quad \{L_S \mid M \text{ a finite monoid, } S \subseteq M\} \ .$$

Two finite words can be given a distance measuring the minimal cardinality of a finite monoid in which their behaviors are different. The monoid of finite words Σ^* can then be completed into a topological monoid $\widehat{\Sigma^*}$ called the free profinite monoid. Its points, known as profinite words, provide a way to speak about limiting behavior of finite words with respect to finite monoids [4].

Regular languages are closed under union, intersection and complement, which means that the set $\mathrm{Reg}\langle\Sigma\rangle$, ordered under the inclusion, is a Boolean algebra. By Stone duality, it has an associated space of ultrafilters which is in fact homeomorphic to $\widehat{\Sigma^*}$. The monoid structure on $\widehat{\Sigma^*}$ can be seen as the dual of residual operations on $\mathrm{Reg}\langle\Sigma\rangle$, see [1]. In summary,

$$\widehat{\Sigma^*} \quad \text{is the Stone dual of} \quad \mathrm{Reg}\langle\Sigma\rangle \ . \tag{1}$$

From words to λ-terms: the Church encoding. We consider the simply-typed λ-calculus with one base type $\mathfrak{o}$. For any simple type A, we denote by $\Lambda\langle A\rangle$ the set of closed λ-terms of type A, taken modulo $\beta\eta$-conversion. To any finite alphabet Σ, we associate the simple type

$$\mathrm{Church}_\Sigma \quad := \quad \underbrace{(\mathfrak{o} \to \mathfrak{o}) \to \ldots \to (\mathfrak{o} \to \mathfrak{o})}_{|\Sigma| \text{ times}} \to \mathfrak{o} \to \mathfrak{o}$$

and we encode finite words over $\Sigma = \{a_1, \ldots, a_n\}$ as terms of this type in the following way:

$$w = a_{w_1} \ldots a_{w_k} \quad \text{is encoded as} \quad \lambda(a_1 : \mathfrak{o} \to \mathfrak{o}) \ldots \lambda(a_n : \mathfrak{o} \to \mathfrak{o}) \lambda(c : \mathfrak{o}).\, a_{w_k} \left(\ldots (a_{w_1} c)\right) \ .$$

We use the finite standard model of the simply-typed λ-calculus, that is we interpret it in the cartesian closed category **FinSet**. This means that, for any simple type A and finite set Q, we obtain a finite set $[\![A]\!]_Q$ and a function

$$[\![-]\!]_Q \quad : \quad \Lambda\langle A\rangle \longrightarrow [\![A]\!]_Q \ .$$

In particular, a word $w \in \Sigma^*$ encoded as a simply-typed λ-term of type Church_Σ will be interpreted as a function

$$[\![w]\!]_Q \quad \in \quad (Q \Rightarrow Q) \Rightarrow \ldots \Rightarrow (Q \Rightarrow Q) \Rightarrow Q \Rightarrow Q$$

taking as inputs a deterministic transition function for each letter of the alphabet Σ and an initial state and giving as output the state the automaton arrives at after reading the word w. This shows that the semantics in **FinSet** generalize at any type the usual notion of run into an automaton; see also [3].

Recognizable languages of λ-terms. In analogy with the monoid case, we define regular languages of λ-terms as sets of λ-terms interpreted as certain semantic elements. Following [6], for any simple type A, finite set Q and subset F of $[\![A]\!]_Q$, we define the language L_F as

$$L_F \quad := \quad \{M \in \Lambda\langle A\rangle \mid [\![M]\!]_Q \in F\} \ .$$

We can therefore define the set $\mathrm{Reg}\langle A\rangle$ of all regular languages of λ-terms of type A as

$$\mathrm{Reg}\langle A\rangle \quad := \quad \{L_F \mid Q \text{ a finite set}, F \subseteq [\![A]\!]_Q\} \ .$$

Using logical relations, we can prove again that $\mathrm{Reg}\langle A\rangle$ is a Boolean algebra and then, in analogy with (1), we *define* the space $\widehat{\Lambda}\langle A\rangle$ of profinite λ-terms of type A such that

$$\widehat{\Lambda}\langle A\rangle \quad \text{is the Stone dual of} \quad \mathrm{Reg}\langle A\rangle \ . \tag{2}$$

There is a natural map $\Lambda\langle A\rangle \to \widehat{\Lambda}\langle A\rangle$, which we prove is injective using [7].

Profinite λ-terms and parametricity. As **FinSet** is a cartesian closed category, we can use logical relations. For any relation $R \subseteq P \times Q$, we have a relation $[\![A]\!]_R \subseteq [\![A]\!]_P \times [\![A]\!]_Q$. Following [5], we say that a family θ of elements $\theta_Q \in [\![A]\!]_Q$, where Q ranges over finite sets, is parametric if for any relation $R \subseteq P \times Q$, the elements θ_P and θ_Q are related by $[\![A]\!]_R$. We denote by $\mathrm{Para}\langle A\rangle$ the set of all parametric families associated to a simple type A.

Parametricity can be thought of as the notion of naturality, adapted to the higher-order setting. The fundamental lemma of λ-calculus states that the interpretation of a λ-term is a parametric family. Each profinite λ-term can be thought as a family θ verifying certain properties, see Definition 9 in [2]. We first show the more general result that profinite λ-terms are in particular parametric.

Theorem 1. *For any simple type A, we have $\widehat{\Lambda}\langle A\rangle \subseteq \mathrm{Para}\langle A\rangle$ i.e. for any profinite λ-term θ of type A and relation $R \subseteq P \times Q$, the points θ_P and θ_Q are related by $[\![A]\!]_R$.*

Our main contribution is a parametricity theorem for Church types, which amounts to saying that any parametric family of type Church_Σ is the interpretation of a profinite λ-term.

Theorem 2. *For any finite set Σ, we have $\widehat{\Lambda}\langle \mathrm{Church}_\Sigma\rangle = \mathrm{Para}\langle \mathrm{Church}_\Sigma\rangle$.*

As future work, we would like to investigate the status of this equality for other higher-order types. We are also interested in studying the possibly different notions of profinite λ-term that we obtain when using other cartesian closed categories as models.

References

[1] Mai Gehrke, Serge Grigorieff, and Jean-Éric Pin. Duality and equational theory of regular languages. In L. Aceto and al., editors, *ICALP 2008, Part II*, volume 5126 of *Lecture Notes in Computer Science*, pages 246–257, Berlin, 2008. Springer.

[2] Sam van Gool, Paul-André Melliès, and Vincent Moreau. Profinite lambda-terms and parametricity. To appear in the Proceedings of the 39th International Conference on Mathematical Foundations of Programming Semantics, 2023.

[3] Paul-André Melliès. Higher-order parity automata. In *Proceedings of the 32nd Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '17. IEEE Press, 2017.

[4] Jean-Éric Pin. Profinite Methods in Automata Theory. In Susanne Albers and Jean-Yves Marion, editors, *26th International Symposium on Theoretical Aspects of Computer Science STACS 2009*, Proceedings of the 26th Annual Symposium on the Theoretical Aspects of Computer Science, pages 31–50, Freiburg, Germany, February 2009. IBFI Schloss Dagstuhl.

[5] John C. Reynolds. Types, abstraction and parametric polymorphism. In R. E. A. Mason, editor, *Information Processing 83, Proceedings of the IFIP 9th World Computer Congress, Paris, France, September 19-23, 1983*, pages 513–523. North-Holland/IFIP, 1983.

[6] Sylvain Salvati. Recognizability in the simply typed lambda-calculus. In *Logic, Language, Information and Computation*, pages 48–60. Springer Berlin Heidelberg.

[7] Richard Statman and Gilles Dowek. On Statman's finite completeness theorem. Technical report, Carnegie Mellon University, 1992. CMU-CS-92-152.

On Conway's proof of undecidability in elementary arithmetic (extended abstract)

Peter M. Hines – University of York

Historical Background

Although undecidability in mathematics has long been a theoretical possibility, in the early 1970s, John Conway demonstrated [5] that even simple iterated arithmetic problems could exhibit undecidable behaviour[1]. A common route to proving undecidability is via a reduction to the Halting Problem for a computationally universal system. Conway's result was no exception : he demonstrated an encoding of Universal Register Machines as iterative problems on simple piecewise-linear arithmetic functions (his well-known 'congruential functions').

This was subsequently given an interpretation as a simple imperative programming language, similar to a low-level assembler, that he called `FRACTRAN` (the 'fraction translator'; an obvious pun on the somewhat better-known `FORTRAN`, the 'formula translator'), which could compute $k.2^{2^n} \mapsto 2^{2^{f(n)}}$, for constant k and arbitrary partial recursive f. Other computationally universal systems based on the same family of arithmetic primitives include S. Burckel's encoding of Minsky Machines [3], and Sergei Maslov's representation of Post Production Systems [18].

The number theorists' loss is then our gain: undecidability is simply a natural side-effect of the ability to express universal computation – and universal computability in elementary arithmetic is ultimately much more interesting than questions of whether natural numbers have finite or infinite orbits under seemingly arbitrary functions.

Conway's motivating example

There is an unfortunate disconnect between the universal computational systems used to demonstrate undecidability (e.g. `FRACTRAN` or Universal Register Machines), and the arithmetic problems that motivated Conway.

Although Collatz's notorious $3x + 1$ problem is often claimed to have inspired Conway, he explicitly mentioned a distinct, albeit rather similar conjecture[2] as his motivation. He further claimed it to be (probabilistically) *'obviously true but also probably undecidable'*. Somewhat provocatively, he coined the term *'probvious'* for this property, and pointed out that the probability of this conjecture being false is significantly less than the probability of an error being found in any given mathematical proof.

It is natural to wonder how this motivating conjecture interprets computationally as, for example, a `FRACTRAN` program. Although this would not help in the slightest to resolve the actual conjecture (even assuming Conway's unprovable claim of undecidability is incorrect!), it would nevertheless provide a computational perspective on a significant arithmetic problem that raises important foundational questions.

Unfortunately, Conway's motivation cannot be expressed as such a program; the subset of congruential functions that may be interpreted as `FRACTRAN` programs is in fact remarkably restrictive. This talk fills this gap, and gives a structural interpretation, based on categorical logic and coherence, of a more general range of congruential functions – including his motivating example as a very important special case.

The amusing musical permutation, and the original Collatz conjecture

Conway's claimed motivation was the **Original Collatz Conjecture** (O.C.C.). This is based on a bijection that he named the **amusical permutation**. The somewhat idiosyncratic terminology arises from a very close connection between this conjecture and long-established musical theory : precisely, the Circle of Fifths, and the Pythagorean Comma. We briefly describe this, with suitable accommodations for a potentially non-musically minded audience, and give justification for his claim that, *'there really is a connection with music'* [4].

However, our primary interest is in understanding it as Theoretical Computer Science. To this end, we point out where it, and closely related congruential functions, occur in category theory & logic.

[1] In the Cold War era, many results were simultaneously and independently discovered on either side of the Iron curtain. Conway's result was no exception; Sergei Maslov's 1968 paper [18] gave a very similar result.

[2] also considered by L. Collatz, as demonstrated by his unpublished notebooks [14].

Congruential functions in (categorical) logic

Both J. Conway's demonstration of computational universality in elementary arithmetic, and Collatz's infuriating problems, concern *congruential functions*. The starting point for these is P. Erdös's notion of an 'exact covering system' – a dissection of the natural numbers into (disjoint) congruence classes $\{A_j\mathbb{N} + B_j\}_{j\in J}$, so $\mathbb{N} = \bigcup_{j\in J} A_j\mathbb{N} + B_j$ and $(A_i\mathbb{N} + B_i) \cap (A_j\mathbb{N} + B_j) = \emptyset$ for $i \neq j$ (i.e. a disjoint finite open cover of $(\mathbb{N}, +)$ w.r.t. the profinite topology). A congruential function is given by applying a linear map conditioned on the congruence class to which the argument belongs. The overall effect is as the union of partial linear maps with disjoint domains, of the form $A_j\mathbb{N}+B_j \mapsto C_j\mathbb{N}+D_j$.

In [12, 15], it is demonstrated how dissections of a countably infinite set into k infinite subsets correspond to concrete realisations of Nivat & Perot's *polycyclic monoids* [19] P_k. Such realisations occur in models of logics and lambda calculii as the *dynamical algebras* (e.g. [6, 8, 9]), and a significant class of congruential functions (including those considered by Collatz and Conway) occur as 'mappings between dynamical algebras'.

A motivating example from logical models

We see these in a model of linear logic that has long been known to have connections with iteration on the natural numbers. It was independently discovered by many authors [12, 1, 11, 2] that J.-Y. Girard's Geometry of Interaction series of papers [8, 9, 10] interpret propositions within the 'symmetric inverse monoid' of partial injective functions on the natural numbers. The model of (multiplicative) conjunction used then a (semi-)monoidal tensor on this monoid. In [13], it is demonstrated – as a generalisation of J. Isbell's argument as quoted in [17] – that no such tensor on a monoid can be strictly associative[3] unless the monoid in question is highly degenerate (i.e. a monoid of abstract scalars, and hence abelian, with tensor and composition coinciding).

Although Girard's conjunction is not strictly associative, it is nevertheless associative *up to canonical isomorphism*; there exists some fixed bijection $\alpha : \mathbb{N} \to \mathbb{N}$ satisfying both naturality and MacLane's pentagon condition.

Naturality $\otimes(f \otimes (g \otimes h)) = ((f \otimes g) \otimes h)\alpha$

MacLane's pentagon condition $\alpha^2 = (\alpha \otimes Id)\alpha(Id \otimes \alpha)$

The 'naturality' property allows us to perform arbitrary re-bracketings, and the 'pentagon', or 'coherence' condition ensures that any two ways of performing the same bracketing are identical.

The *canonical associativity isomorphism* or *associator* for Girard's conjunction is then precisely a congruential function, although Collatz-like problems based on it are disappointingly trivial! However, it does illustrate that congruential functions occur in logical / categorical models. This is not at the level of propositions, connectives, or operators, but rather as the structural morphisms that mediate properties such as associativity.

As a preliminary step, we give an efficient decision procedure for deciding whether a given natural number has a finite or infinite orbit under this congruential function. We then use the above naturality property, and the well-known [7] connection between associativity laws and Thompson's group $\mathcal{F}$ to give a presentation of $\mathcal{F}$ as a group of congruential functions.

Collatz on the associahedron

Our next step is to generalise Girard's conjunction in a natural way to an $\mathbb{N}^+$-indexed coherent family of 'unbiased tensors' on the symmetric inverse monoid of partial injections on the natural numbers. We treat these operadically, and give a 1:1 correspondence with the operad of rooted planar trees (and thus arbitrary facets of Stasheff's associahedra).

We then exhibit natural isomorphisms between these unbiased tensors, giving a notion of coherence based on natural isomorphisms whose components are congruential functions. We describe the simplest possible case explicitly; this results in the canonical associativity isomorphism for Girard's conjunction factoring into two distinct ways of expressing the Original Collatz Conjecture[4] – putting Conway's motivating example firmly within the realm of coherence for categorical logic.

The 1:1 correspondence between (operadic composites of) unbiased tensors, and rooted planar trees, allows us to consider Stasheff's associahedra as commuting categorical diagrams of congruential functions – with arrows between arbitrary facets, rather than simply the 1-skeleton of vertices and edges. We give the diagram arising from the fourth associahedron explicitly (containing, of course, MacLane's pentagon as a sub-diagram). Finally, we demonstrate that Conway's motivation — the amusical permutation, from the Original Collatz Conjecture — occurs in the commuting diagrams derived from arbitrary-dimensional associahedra.

[3] We emphasise that this does not contradict MacLane's well-known theorem; his strictification procedure results in a (monoidally) *equivalent category*, and the property of being a monoid (i.e. only having a single object) is certainly not a categorical property of the type that would be preserved by such an equivalence.

[4] and thus factoring an almost trivial problem into two different ways of expressing the same undecided, and possibly undecidable, problem.

Additional considerations

It is - by far - more common to study categories with a single (binary) monoidal tensor, rather than those equipped with an entire $\mathbb{N}^+$–indexed family of unbiased tensors. A significant reason for this is given by T. Leinster in [16], where is is shown that MacLane's strictification procedure is similarly applicable in the unbiased setting. This, of course, reduces everything to the usual and well-understood case of a category with a single strictly associative (binary) (semi-)monoidal tensor.

If time permits, we will briefly discuss the 'strictified' version of the above framework.

References

[1] S. Abramsky. Retracing some paths in process algebra. In U. Montanari and V. Sassone, editors, *CONCUR '96: Concurrency Theory*, pages 1–17. Springer Berlin Heidelberg, 1996.

[2] S. Abramsky, E. Haghverdi, and P. Scott. Geometry of interaction and linear combinatory algebras. *Mathematical Structures in Computer Science*, 12 (5), 2002.

[3] Serge Burckel. Functional equations associated with congruential functions. *Theor. Comput. Sci.*, 123:397–406, 1994.

[4] J. H. Conway. *FRACTRAN: A Simple Universal Programming Language for Arithmetic*, pages 4–26. Springer New York, 1987.

[5] John Conway. Unpredictable iterations. *Proc. 1972 Number Theory*, pages 49–52, 1972.

[6] V. Danos and L. Regnier. Local and asynchronous beta reduction. In *Proceedings of the Eighth Annual IEEE Symp. on Logic in Computer Science*, 1993.

[7] P. Dehornoy. The structure group for the associativity identity. *J. Pure Appl. Algebra*, 111, 1-3:59–82, 1996.

[8] J.-Y. Girard. Geometry of interaction 1. In *Proceedings Logic Colloquium '88*, pages 221–260. North-Holland, 1988.

[9] J.-Y. Girard. Geometry of interaction 2: deadlock-free algorithms. In *Conference on Computer Logic*, volume 417 of *Lecture Notes in Computer Science*, pages 76–93. Springer, 1988.

[10] J.-Y. Girard. Geometry of interaction 3: Accommodating the additives. In *In: Advances in Linear Logic, LNS 222,CUP, 329–389*, pages 329–389. Cambridge University Press, 1995.

[11] E. Haghverdi. *A categorical approach to linear logic, geometry of proofs and full completeness.* PhD thesis, University of Ottawa, 2000.

[12] P. Hines. *The algebra of self-similarity and its applications.* PhD thesis, University of Wales, Bangor, 1997.

[13] P. Hines. Coherence and strictification for self-similarity. *Journal of Homotopy & Related Structures*, 2016.

[14] J. Lagarias. The $3x + 1$ problem and its generalisations. *Amer. Math. Monthly*, 92:3–23, 1985.

[15] M. V. Lawson. *Inverse semigroups: the theory of partial symmetries.* World Scientific, Singapore, 1998.

[16] Tom Leinster. General Operads and Multicategories. *arXiv Mathematics e-prints*, page math/9810053, October 1998.

[17] S. MacLane. *Categories for the working mathematician.* Springer-Verlag, New York, second edition, 1998.

[18] Sergei Ju. Maslov. On e. l. post's "tag problem". In B. M. Budak, editor, *Eleven Problems on Logic, Algebra, Analysis, and Topology*, A.M.S. Translations. American Mathematical Society, 1971.

[19] M. Nivat and J. Perrot. Une généralisation du monöide bicyclique. *Comptes Rendus de l'Académie des Sciences de Paris*, 27:824–827, 1970.