CALCO EARLY IDEAS 2017

– a satellite workshop of CALCO 2017 –

Programme Committee

- Giorgio Bacci *Aalborg University, Denmark*
- Marta Bílková *Charles University, Czech Republic*
- Daniela Petrișan (chair) *CNRS, IRIF, Université Paris Diderot – Paris 7, France*
- Jurriaan Rot *Radboud University Nijmegen, Netherlands*
- Dmitriy Traytel *ETH Zürich, Switzerland*
- Ionut Tutu *Royal Holloway University of London, UK*
- Fabio Zanasi *University College London, UK*
- Joost Winter *University of Warsaw, Poland*

Accepted Abstracts

Automata Learning via Logic*

Simone Barlocco[†1] and Clemens Kupke[‡1]

[1]Department of Computer & Information Sciences, University of Strathclyde, Glasgow, Scotland

Abstract

In this talk we will outline a fresh take on Dana Angluin's algorithm for learning a DFA using ideas from coalgebraic modal logic. Our work opens up possibilities for applications of tools & techniques from modal logic to automata learning and vice versa. As main technical result we generalise Angluin's original algorithm from DFAs to coalgebras for an arbitrary finitary set functor T in the following sense: given a (possibly infinite) pointed T-coalgebra that we assume to be regular (ie. having an equivalent finite representation) we can learn its finite representation by asking (i) "logical queries" (corresponding to membership queries) and (ii) making conjectures to which the teacher has to reply with a counter example. This covers (a known variant) of the original L* algorithm and the learning of Mealy machines and Moore machines. Other examples are streams, infinite trees and bisimulation quotients of transition systems.

1 Introduction

Coalgebra studies "generated behaviour" that can be observed when interacting with a system. Much progress has been made thus far to formalise behaviour and to create languages that allow to specify and reason about it. Surprisingly little, however, has been done to formalise the process of using observations to learn how a system works. This has changed thanks to a series of recent work [3, 4] but the connections between coalgebra & learning are still far from being well understood. In this talk we will describe a link between the above mentioned coalgebraic specification languages (aka coalgebraic modal logics) and the well-known L* algorithm [2] for learning deterministic finite automata (DFA). This algorithm constructs a minimal DFA accepting a (at the beginning unknown) regular language by asking a teacher membership queries of the form "is word w in the language?" and by making conjectures for what the language/DFA is, to which the teacher replies with a counterexample in case the conjecture is false. A central role in the algorithm is played by so-called tables that essentially consist of two sets of finite words S and E which correspond to states of the constructed DFA and to tests performed on these states, respectively.

The use of tests shows that the connection to logic is not at all surprising. A bit more surprising - or rather eye-opening - was for us the observation that closed & consistent tables can be best understood using the notion of filtrations from modal logic. We will not discuss this observation in

*This work was partially supported by EPSRC grant EP/N015843/1.

[†]simone.barlocco@strath.ac.uk

[‡]clemens.kupke@strath.ac.uk

detail - instead we will describe a generalisation of the L* algorithm to coalgebras that was made possible by it. Our "L^{co} algorithm" allows - in principle - the learning of regular coalgebras for an arbitrary finitary set functor. We will now briefly describe in what way we generalise Dana Angluin's L* algorithm but will not introduce the details of our algorithm here due to space reasons.

What the algorithm learns

The classical L* algorithm looks very much like a bottom up procedure. It is instructive, however, to think of the algorithm as follows: The regular language $\mathcal{L} \subseteq \Sigma^*$ that we intend to learn can be represented by the infinite automaton

$$\langle o, \delta \rangle : \Sigma^* \to 2 \times (\Sigma^*)^\Sigma$$

where $o(w) = 1$ iff $w \in \mathcal{L}$ and $\delta(w)(a) = w \cdot a$ for all $w \in \Sigma^*$, $a \in \Sigma$. The assumption that $\mathcal{L}$ is a regular language can be rephrased by stating that the pointed coalgebra $(\Sigma^*, \langle o, \delta \rangle, \lambda)$ is bisimilar to a finite well-pointed coalgebra [1]. The aim of the algorithm is to learn this finite well-pointed coalgebra using queries that can be asked concerning the given infinite coalgebra. Our generalisation of Angluin's algorithm looks thus as follows: We are given a (possibly infinite) pointed T-coalgebra (X, γ, x) (corresponding to the language $\mathcal{L}$) for a finitary set functor T and we assume that (X, γ, x) is *regular*, ie., behaviourally equivalent to a finite well-pointed T-coalgebra (Y, δ, y). The goal of the algorithm is to learn (Y, δ, y).

Means to learn

The central device for learning in our setting is logic: we assume that we are provided with an expressive modal language that allows to characterise coalgebras up-to behavioural equivalence. The learner is able to ask two types of queries:

- Logical queries of the form *"is formula φ true at some state $x' \in X$?"*
- Conjectures of the form *"is this the well-pointed T-coalgebra we are looking for?"*

Logical queries are answered truthfully with *Yes* or *No* by the teacher, conjectures are either confirmed or the teacher provides a counterexample in the shape of a formula $\psi \in \mathcal{F}(\Lambda)$ that can be used to distinguish the point of the conjectured coalgebra from the point x of the coalgebra that we are trying to learn.

References

[1] Jirí Adámek, Stefan Milius, Lawrence S. Moss, and Lurdes Sousa. Well-pointed coalgebras. *Logical Methods in Computer Science*, 9(3), 2013.

[2] Dana Angluin. Learning regular sets from queries and counterexamples. *Information and Computation*, 75(2):87 – 106, 1987.

[3] Bart Jacobs and Alexandra Silva. Automata learning: A categorical perspective. volume 8464 of *LNCS*, pages 384–406. Springer, 2014.

[4] Joshua Moerman, Matteo Sammartino, Alexandra Silva, Bartek Klin, and Michał Szynwelski. Learning nominal automata. POPL 2017, 2017.

Lazy Players Matching Pennies

Jason Castiglione
Department of Electrical Engineering
University of Hawaii at Manoa, Honolulu, Hawaii, USA
jcastig@hawaii.edu

June 14, 2017

Abstract

In the matching pennies game there are two opponents, Alice and Bob, and each have a penny. On the cue of a third party, they choose either head or tails and place their penny on the table. Alice wins if her coin matches Bob's coin, and Bob wins if the coins do not match. It is well known that if each opponents' strategy must be declared, the optimal strategy is to play random coin flips. We look at a different take on the game, where we consider the consequences of the players being lazy. The players are aware they cannot win, and would like to put forth a minimal amount of effort in not winning. This is a simply stated problem, yet representing it in terms of coalgebra provides clarity to the model, and the ability to apply coinductive proof techniques.

1 Introduction

One of the essential concepts in game theory is that of equilibrium points in a game. Informally, they can be thought of as fixed points with regards to all possible strategies of a player. A strategy is considered a fixed point when there is no advantage in changing strategies. In Nash [4], non-cooperative games were presented and conditions for the existence of these equilibrium points were proven with the use of the Brouwer fixed point theorem. Nash's seminal thesis focused on players with complete information on potential strategies, and one game. Later on it was of great interest to consider repeated games with incomplete information, i.e., a game was played over and over again, with players having limited information on each others strategies. Aumann and Maschler [2] provide a thorough introduction to this field, and motivation through concrete examples in economics. In Pavlovic [5], it is pointed out that the semantics of computation of fixed points is a natural setting to analyze games which can be viewed as processes. Furthermore Pavlovic, goes on to show that strategies can be viewed as randomized programs. In recent work by Abramsky and Winschel [1], infinite extensive games and subgame perfect equilibria were defined in coalgebraic terms. In earlier work [3], provides methods using trellis based techniques for estimation of information rates for processes with potentially infinite memory. Trellis based techniques have been used in information theory for maximum likelihood decoding of error correction codes. They make use of the sum-product algorithm which can efficiently calculate probabilities of state-based systems. We recognize that players might be in a game where there is no winning strategy to a game, so they might as well sip tea and put forth a minimal amount of effort. The author's goal is to use the trellis based techniques to estimate properties of cost functions for games with lazy players.

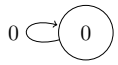


Figure 1: Alice's coalgebra, $a : \mathbb{1} \to \mathbb{1} \times \mathbb{2}$

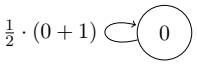


Figure 2: Bob's coalgebra, $b : \mathbb{1} \to \Delta(\mathbb{1} \times \mathbb{2})$

Let us denote random bits drawn from a distribution with probability $p \in [0, 1]$ that the bit is a 1, by **p-random bits**. In playing the matching pennies (MP) game [6], the Nash equilibrium is for each player to play $\frac{1}{2}$-random bits. Each player is assured that they will not benefit by playing the game, nor will they be penalized. This is a little disconcerting, because as a player of games one would like to feel like they win in some regard. Let Bob assume the strategy of playing $\frac{1}{2}$-random bits. Alice can play all 0s and neither win nor lose. Alice and Bob tie on average since the probability that they will agree is $\frac{1}{2}$. In this instance, Alice is a stream coalgebra, and Bob is probabilistic coalgebra, see figs. (1, 2) [1]. In a sense, Alice can tell herself she did better then Bob because she did not put in nearly as much effort. The objective is to put forth the least amount of effort. Thus, we associate a cost of $c = h(p)$ for each p-random bit played, where h is the binary entropy function[2]. Then for n plays Alice has a cost of $n \cdot h(0) = 0$, and Bob has a cost of $n \cdot h(\frac{1}{2}) = n$.

In the next evolution of the game Bob publishes his strategy online. Bob hopes that Alice will get the message that he is onto her and will detect her lazy strategy. Let $a_1, a_2, \ldots$ denote the plays of Alice. Bob implements the following strategy. His first two plays are $\frac{1}{2}$-random bits. Every subsequent play Bob averages the last two plays of Alice, say $p_t = \frac{a_{t-2}+a_{t-1}}{2}$, and then plays a $(1 - p_t)$-random variable at time t. Knowing Bob's fixed strategy, Alice forms a simple strategy. Alice plays $001100110011\ldots$. Bob's resulting plays are; $\frac{1}{2}$-random variables for the first two plays, a 1, $\frac{1}{2}$-random variable, 0, and the last four repeat. Now observe Alice has cost 0, and Bob has cost $2 + \lfloor \frac{n-2}{3} \rfloor$. Interestingly, we see that in this game on average Alice wins $\frac{3n-2}{4}$ out of n trials, i.e., approximately 75% of the time. Thus Bob declared a fixed strategy based on Alice's play, and lost by a significant margin. We notice that Bob's current state depends on the previous two outputs from Alice.

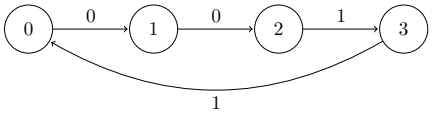


Figure 3: Alice's coalgebra, $a : 4 \to 4 \times \mathbb{2}$

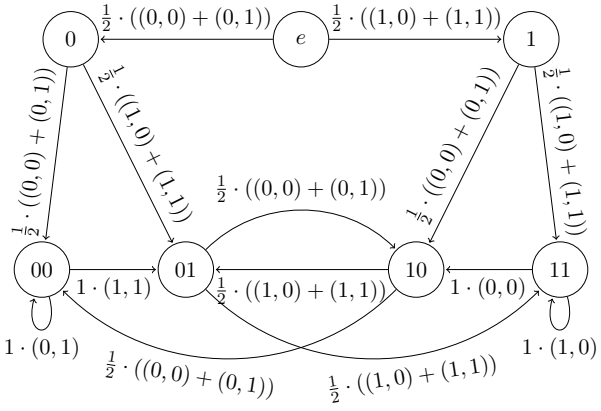


Figure 4: Bob's coalgebra, $b : 7 \to (7 \times \Delta 2)^2$

We are starting to see a game form under the following constraints; one player must fix a strategy and publish it. The published strategy should minimize cost, and not deviate far from

[1] To reduce the number of arrows in a diagram, we collect all arrows into a given state, and write expressions of the form $\alpha \cdot (\ell_1 + \ell_2 + ...)$ to represent label ℓ_i occurs with probability α.

[2] $h(p) = p \log(\frac{1}{p}) + (1 - p) \log(\frac{1}{1-p})$

equal losses and wins. Thus the question arises, does there exist a fixed strategy for Bob based on Alice's actions with cost less then cn that Bob can make public and Alice can not defeat? If Alice has unlimited time, and computational capability, Bob will run into similar problems as before. Of course this doesn't happen in practice, because Alice is limited in compute time. Thus it would be of interest to quantify when can Alice discover Bob's biases in polynomial steps.

References

[1] S. Abramsky and V. Winschel. Coalgebraic analysis of subgame-perfect equilibria in infinite games without discounting. *Mathematical Structures in Computer Science*, 2017.

[2] R.J. Aumann, M. Maschler, and R.E. Stearns. *Repeated Games with Incomplete Information*. MIT Press, 1995.

[3] J. Castiglione and A. Kavcic. Trellis based lower bounds on capacities of channels with synchronization errors. In *2015 IEEE Information Theory Workshop - Fall (ITW), Jeju Island, South Korea, October 11-15, 2015*, pages 24–28, 2015.

[4] J. Nash. Non-cooperative games. *Annals of Mathematics*, 54(2):286–295, 1951.

[5] D. Pavlovic. A semantical approach to equilibria and rationality. In *Proceedings of the 3rd International Conference on Algebra and Coalgebra in Computer Science*, CALCO'09, pages 317–334, Berlin, Heidelberg, 2009. Springer-Verlag.

[6] D. Pavlovic. Testing randomness by Matching Pennies. *ArXiv*, 2015.

Trace semantics via lax comonad

Liang-Ting Chen
Institute of Information Science, Academia Sinica, Taiwan
lxc@iis.sinica.edu.tw

Abstract

This on-going work aims at a succinct approach to generic trace semantics using laxness. The notion of lax comonad is proposed in lieu of an ordinary functor adopted in previous works by Power and Turi [13], Jacobs [7], Hasuo et al. [6], and Cîrstea [4] to name a few.

1 Motivation

Reactive state-based systems with *atomic* transitions can be characterised as coalgebras for some functors, and many works based on this formulation successfully unify and extend classic results elegantly. But, 'the area of coalgebra is still in its infancy' [8]. Take trace semantics for example. A generic approach to finite trace semantics and simulation is addressed in [6] and [5] respectively, whereas infinite trace semantics is discussed only for non-deterministic systems in [13, 7]. From the viewpoint of verification [11, 1], finite trace semantics can only describe safety properties leaving liveness properties untouched. The interpretation of temporal coalgebraic modality is defined by repeatedly unfolding the coalgebra structure [3] while LTL and CTL are given on traces directly. Furthermore, some state-based systems such as timed processes [10] simply cannot be modelled by any functor.

The above discussion leads us to the view of transition systems as *sets of executions* considered by Lamport [12]. Here, an *execution* over an action set $\mathcal{A}$ denotes either a finite or an infinite alternating sequence of states and actions, and a system is viewed as the set of all possible maximal execution fragments initiated from a given state. For example, labelled transition systems with an explicit termination ✓ and simulations are coalgebras and their homomorphisms for the following functors and comonads respectively:

LTS's with ✓	functor (atomic transitions)	comonad (executions)
deterministic	$B_{\mathcal{A}} \coloneqq \{\checkmark\} + \mathcal{A} \times (-) : \mathbf{Set} \to \mathbf{Set}$	$(B_{\mathcal{A}}^{\infty}, \epsilon, \delta) \coloneqq \mathrm{cofree}(B_{\mathcal{A}})$
non-deterministic	$\overline{B_{\mathcal{A}}} \colon \mathbf{Set}_{\mathbb{P}} \to \mathbf{Set}_{\mathbb{P}}$ [13, 7, 5]	?

where $\overline{B_{\mathcal{A}}}$ is the functor lifted to the Kleisli category $\mathbf{Set}_{\mathbb{P}}$ for the powerset monad $\mathbb{P}$ by a canonical distributive law, and $(B_{\mathcal{A}}^{\infty}, \epsilon, \delta)$ the (coalgebraically) cofree comonad over $B_{\mathcal{A}}$ consisting of $B_{\mathcal{A}}^{\infty} \colon X \mapsto X \times (\mathcal{A} \times X)^{\infty}$ the set of executions over $\mathcal{A}$, $\epsilon_X \colon B_{\mathcal{A}}^{\infty} X \to X$ the first projection, and $\delta \colon B_{\mathcal{A}}^{\infty} \to B_{\mathcal{A}}^{\infty} B_{\mathcal{A}}^{\infty}$ which maps a sequence π to the sequence of its tails:

$$\pi = \left(x_0 \xrightarrow{a_0} x_1 \xrightarrow{a_1} \cdots \right) \mapsto \left(\pi \xrightarrow{a_0} \mathsf{tail}(\pi) \xrightarrow{a_1} \mathsf{tail}^2(\pi) \xrightarrow{a_2} \cdots \right).$$

The one-to-one correspondence between deterministic labelled transition systems as atomic transitions and as sets of executions is essentially the coalgebraic cofreeness of $(B^\infty_{\mathcal{A}}, \epsilon, \delta)$. To fill the bottom-right corner of the above table, we demonstrate a lax comonad on the Kleisli category $\mathbf{Set}_{\mathbb{P}}$ reconciling systems of executions in a coalgebraic form.

2 Lax coalgebras of systems of executions

Similar to the previous approaches, let us consider a *lax distributive law* $B^\infty_{\mathcal{A}}\mathbb{P} \xrightarrow{\lambda} \mathbb{P}B^\infty_{\mathcal{A}}$ of the comonad $(B^\infty_{\mathcal{A}}, \epsilon, \delta)$ over the powerset monad $(\mathbb{P}, \{-\}, \bigcup)$ defined by

$$(S, \boldsymbol{\pi}) \mapsto \{\, (s, \pi) \mid s \in S,\ |\pi| = |\boldsymbol{\pi}|,\ p_1^\infty \pi = p_1^\infty \boldsymbol{\pi},\ \text{and for each } i \leq |\pi|\colon\ (p_2^\infty \pi)_i \in (p_2^\infty \boldsymbol{\pi})_i \,\}$$

where $|{-}|$ is the length function, p_1^∞ and p_2^∞ are the lifted projections. Note that the Kleisli category $\mathbf{Set}_{\mathbb{P}} \cong \mathbf{Rel}$ is a 2-category whose hom-categories are posetal.

The distributive law λ defines a *lax comonad* $(\overline{B^\infty_{\mathcal{A}}}, \overline{\epsilon}, \overline{\delta})$ over $\mathbf{Set}_{\mathbb{P}}$ by the following data:

- a functor $\overline{B^\infty_{\mathcal{A}}}\colon \mathbf{Set}_{\mathbb{P}} \to \mathbf{Set}_{\mathbb{P}}$ defined by $X \mapsto B^\infty_{\mathcal{A}} X$ and $f^\dagger \mapsto (\lambda_Y \circ B^\infty_{\mathcal{A}} f)^\dagger$;
- a natural transformation $\overline{\epsilon} \coloneqq (\{-\} \circ \epsilon)^\dagger$ as the counit;
- a lax natural transformation $\overline{\delta} \coloneqq (\{-\} \circ \delta)^\dagger$ as the comultiplication

where $f^\dagger$ denotes the Kleisli morphism $X \to Y$ in $\mathbf{Set}_{\mathbb{P}}$ for a function $f\colon X \to \mathbb{P}Y$.

Definition (see [2])**.** *A lax $\overline{B^\infty_{\mathcal{A}}}$-coalgebra is a morphism $\xi\colon X \to \overline{B^\infty_{\mathcal{A}}} X$ in $\mathbf{Set}_{\mathbb{P}}$ satisfying inequations $\overline{\epsilon}_X \circ \xi \leq id$ and $\overline{\delta}_X \circ \xi \leq \overline{B^\infty_{\mathcal{A}}}\xi \circ \xi$. A lax $\overline{B^\infty_{\mathcal{A}}}$-homomorphism from (X, ξ) to (Y, γ) is a morphism $f\colon X \to Y$ satisfying $\gamma \circ f \leq \overline{B^\infty_{\mathcal{A}}} f \circ \xi$.*

Put differently, a lax coalgebra $\xi\colon X \to \overline{B^\infty_{\mathcal{A}}} X$ is a function $X \to \mathbb{P}B^\infty_{\mathcal{A}} X$ such that every execution $\pi \in \xi(x)$ initiates from x and for every i-th state x_i in the execution π the execution $\mathsf{tail}^i(\pi)$ is contained in $\xi(x_i)$. Every lax homomorphism $f\colon (X, \xi) \to (Y, \gamma)$ is a *simulation on executions*, i.e. a relation $\mathcal{R} \subseteq X \times Y$ such that $x\ \mathcal{R}\ y$ if and only if for every execution $\pi \in \gamma(y)$ there is $\pi' \in \xi(x)$ of the same length with $p_1^\infty(\pi) = p_1^\infty(\pi')$ and $(p_2^\infty \pi)_i\ \mathcal{R}\ (p_2^\infty \pi')_i$ for each $i \leq |\pi|$; also every simulation (on states) is a simulation on executions and vice versa. To summarise, we have the following proposition.

Proposition. *The category having as objects systems of executions and as morphisms simulations is a full subcategory of lax $\overline{B^\infty_{\mathcal{A}}}$-coalgebras.*

Every lax comonad induces a *lax right adjoint* to the forgetful 2-functor from the category of lax coalgebras to its underlying category [2], but a lax adjunction is not necessarily a natural isomorphism between hom-categories but only an adjunction *up to adjointness* [9]. It follows that the coalgebra of traces is a *local terminal* object in the category of lax coalgebras, clarifying the weak finality observed in [7]:

Theorem. *For every $\overline{B^\infty_{\mathcal{A}}}$-coalgebra $(X, \xi^\dagger)$ there exists a greatest lax homomorphism from $(X, \xi^\dagger)$ to the cofree lax coalgebra $(\mathcal{A}^\infty, \widehat{\delta})$ defined by*

$$X \xrightarrow{\xi} \mathbb{P}B^\infty_{\mathcal{A}} X \xrightarrow{\mathbb{P}B^\infty_{\mathcal{A}} k_{\{*\}}} \mathbb{P}B^\infty_{\mathcal{A}}\mathbb{P}\{*\} \xrightarrow{\mathbb{P}\lambda_{\{*\}}} \mathbb{P}^2 B^\infty_{\mathcal{A}}\{*\} \xrightarrow{\bigcup} \mathbb{P}B^\infty_{\mathcal{A}}\{*\} \cong \mathbb{P}\mathcal{A}^\infty.$$

which is the transpose of the constant map $k^\dagger_{\{*\}}\colon X \to \{*\}$ at the singleton $\{*\}$ under the lax adjunction induced by the comonad.

3 Conclusions

The current work is obviously in its very early stage, and the bulk of trace calculation boils down to the construction of a system of executions from a labelled transition system, i.e. constructing a lax $\overline{F^\infty}$-coalgebras from an $\overline{F}$-coalgebra where F^∞ is the coalgebraically cofree comonad over F. Nevertheless, the above approach requires only a monad whose Kleisli category is locally posetal and a lax mixed distributive law, so it seems a more conceptual approach to generic trace semantics and enables us to take continuous hybrid systems into account. In the future, we plan to investigate laxness in full detail, more examples of lax comonad, and path-based coalgebraic temporal logic in this framework.

References

[1] Bowen Alpern and Fred B. Schneider. Recognizing safety and liveness. *Distrib. Comput.*, 2(3):117–126, sep 1987.

[2] Marta C. Bunge. Coherent extensions and relational algebras. *Trans. Am. Math. Soc.*, 197:355–355, 1974.

[3] Corina Cîrstea. A coalgebraic approach to quantitative linear time logics. *Log. Methods Comput. Sci.*, 7(3):1–26, dec 2016.

[4] Corina Cîrstea. From branching to linear time, coalgebraically. *Fundam. Informaticae*, 150(3-4):379–406, mar 2017.

[5] Ichiro Hasuo. Generic forward and backward simulations. In Christel Baier and Holger Hermanns, editors, *CONCUR 2006 Concurr. Theory*, volume 4137 of *Lecture Notes in Computer Science*, pages 406–420. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[6] Ichiro Hasuo, Bart Jacobs, and Ana Sokolova. Generic trace semantics via coinduction. *Log. Methods Comput. Sci.*, 3(4):36, nov 2007.

[7] Bart Jacobs. Trace semantics for coalgebras. *Electron. Notes Theor. Comput. Sci.*, 106(1-4 SPEC. ISS.):167–184, dec 2004.

[8] Bart Jacobs. *Introduction to Coalgebra: Towards Mathematics of States and Observation*, volume 59 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, dec 2016.

[9] C. Barry Jay. Local adjunctions. *J. Pure Appl. Algebr.*, 53(3):227–238, sep 1988.

[10] Marco Kick, A. John Power, and Alex Simpson. Coalgebraic semantics for timed processes. *Inf. Comput.*, 204(4):588–609, apr 2006.

[11] Leslie Lamport. Proving the correctness of multiprocess programs. *IEEE Trans. Softw. Eng.*, SE-3(2):125–143, mar 1977.

[12] Leslie Lamport. Basic concepts. In M. Paul, H. J. Siegert, M. W. Alford, J. P. Ansart, G. Hommel, L. Lamport, B. Liskov, G. P. Mullery, and F. B. Schneider, editors, *Distrib. Syst.*

Methods Tools Specif. An Adv. Course, volume 190 of *Lecture Notes in Computer Science*, pages 7–43. Springer Berlin Heidelberg, 1985.

[13] John Power and Daniele Turi. A coalgebraic foundation for linear time semantics. *Electron. Notes Theor. Comput. Sci.*, 29(1):259–274, 1999.

Towards a categorical understanding of variety theorems for quantitative algebras

William Boshuck[1], Florence Clerc[2] and Prakash Panangaden[2]

[1]John Abbott College
[2]School of Computer Science, McGill University

A *quantitative algebra* (QA) consists of a space equipped with both an algebra structure and an extended metric that are compatible in the sense that the operations of the signature are non-expansive with respect to the metric. They were introduced in [3] as a formalism intended to capture approximate reasoning. To reason about algebras in the usual sense, it is common to use equations $s = t$ which define a congruence over the algebraic structure. In the quantitative case, the analog is *quantitative equations* $s =_\epsilon t$ which should be understood as "s and t are within ϵ of each other". A *Quantitative Equational Theory* (QET) is a set of Horn clauses of such quantitative equations. The authors give examples showing how to characterize some specific metrics (like the Wasserstein distances or the Hausdorff distance) by QETs.

In [4], the authors give analogs of the well-known Birkhoff theorem relying on the idea of c-reflexivity (where c is a cardinal) : a homomorphism $f : A \to B$ of QAs is called *c-reflexive* if for any subset B' of B of size at most c, there exists a subset A' of A such that $f(A') = B'$ and f is an isometry of A'. The very general version they prove is that, given a cardinal c, a class of QAs is c-equational (i.e. it can be described by a set of Horn clauses $\Gamma \vdash s =_\epsilon t$ where Γ is a set of at most c quantitative basic equations) if and only if it is a c-variety (i.e. closed under arbitrary products, subalgebras and c-reflexive homomorphic images).

This quasivariety theorem is surprising as it is not obvious why cardinality issues arise. We are aiming at understanding this work from a categorical viewpoint in order to be able to explain this connection and to reach a more "structural" understanding of the theorem. To do that, we first concentrate our efforts on a variety theorem. Our initial approach relies on the presentation in *Algebraic Theories*, a well-known book by E. Manes [2]. The result we are trying to show is the standard Birkhoff theorem : a class of QAs is a variety (i.e. closed under arbitrary products, subalgebras and homomorphic images) if and only if it can be described by quantitative equations.

In order to prove this we will aim to show that the following four statements are equivalent, given a class $\mathbb{K}$ of QAs :

1. $\mathbb{K}$ is described by quantitative equations,
2. $\mathbb{K}$ is a variety,
3. $\mathbb{K}$ is an *abstract Birkhoff subcategory* (ABS) (*i.e.* it is full and replete, closed under U-split epimorphisms and every free object has a reflection), and

4. $\mathbb{K}$ is isomorphic to the Eilenberg-Moore category $\mathbf{EMet}^{T'}$ where T' is a monad satisfying some more assumptions.

The proof of the equivalence of those four statements goes as follows.

- Proving that $1 \Rightarrow 2$ is done directly using the definition of QET where the Horn clauses have empty left sides.
- The main difficulty of proving $2 \Rightarrow 3$ is designing the reflection for all free algebras. However, this is done for any QA A by first generating the free algebra of all terms TA from A and then quotienting it by a well-chosen pseudometric. Here this pseudometric acts as the quantitative equations that all algebras in $\mathbb{K}$ have to satisfy. However, the choice of this pseudometric is not straight-forward : it starts by defining a set of pseudometrics P_A such that for any pseudometric p in P_A, the quotient of the free algebra TA by p (which we define as a generalization of the well-known notion of quotient by a relation) is in $\mathcal{K}$. We can then take the limit of a functor $F : P_A \to \mathbf{EMet}$ and the metric on this limit is the pseudometric we consider. It is still unclear to us why the sup of all pseudometrics in P_A does not work here.
- The proof of $3 \Rightarrow 4$ is theorem 3.4 of chapter 3 of [2] which states that a category $\mathbb{K}$ is an ABS if and only if there exists a monad T' on **EMet** such that $\mathbb{K}$ and the Eilenberg-Moore category $\mathbf{EMet}^{T'}$ are isomorphic (with some additional conditions).
- Next step of the proof is proving that $4 \Rightarrow 2$. This will conclude the proof of the equivalence of 2, 3 and 4. This step is not yet done.
- To prove that $2 \Rightarrow 1$, we are using the pseudometric we designed in the proof of $2 \Rightarrow 3$. Indeed, there are two equivalent sets of quantitative equations that can be defined from a variety $\mathbb{K}$: the set $\mathcal{U}$ of quantitative equations satisfied by all QAs in $\mathbb{K}$ and the set of quantitative equations satisfied by all QAs of the form TA/α as defined in the proof of $2 \Rightarrow 3$. Proving then that the variety $\mathbb{K}$ is in fact the QET induced by the set $\mathcal{U}$ amounts to proving that if a QA A satisfies the set $\mathcal{U}$ of quantitative equations, then the two QAs A and TA/α are isomorphic. However this last step is not complete yet.

By providing a categorical understanding of this proof, we are hoping to understand the results of [4] and why cardinalities arise. This should help us extend those results to an enriched setting in the future.

This proof and especially the design of the reflection makes it look like it could be related to locally presentable categories. We are thus exploring whether the theory of locally presentable categories [1] is a better route to understanding these results. It seems that this path will make it easy to extend to the case of general Horn clauses which are covered in [4].

Acknowledgements This work was supported by a grant from Natural Sciences and Engineering Research Council of Canada.

References

[1] Jiří Adámek and Jiří Rosicky. *Locally presentable and accessible categories*, volume 189. Cambridge University Press, 1994.

[2] Ernest Manes. *Algebraic theories*, volume 26 of *Graduate Texts in Mathematics*. Springer-Verlag, 1976.

[3] Radu Mardare, Prakash Panangaden, and Gordon Plotkin. Quantitative algebraic reasoning. In *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science*, pages 700–709. ACM, 2016.

[4] Radu Mardare, Prakash Panangaden, and Gordon Plotkin. Variety theorems for quantitative algebras. In *Proceedings of the 32nd Annual ACM/IEEE Symposium on Logic in Computer Science*. ACM, 2017.

A Categorical Semantics for the Probabilistic λ-calculus

Alessandra Di Pierro
Dipartimento di Informatica, Università di Verona, Verona, Italy
alessandra.dipierro@univr.it

Abstract

From a programming language viewpoint, the probabilistic λ calculus formalises several features of the modern description of probabilistic computation and its implementation. We present a categorical semantics that captures a basic feature of probabilistic programming languages, namely continuous linear functionals as both the objects and the result of a computation.

1 The Problem and the Idea

The denotational semantics of programming languages is needed to express the functional meaning of a program and is therefore an essential basis for several program analyses. Although probabilistic programming is today a well-established discipline, there isn't yet a well-established reference model for probabilistic computation similar to the λ-calculus and the Scott denotational domain for classical computation [1].

In this work, we discuss the definition of an abstract semantical model for probabilistic λ-calculus, i.e. the classical λ-calculus extended with a syntactic construct for probabilistic choice. The model we have in mind is essentially a *computational monad* in the sense of [4, 3], that we define over the category of topological vector spaces and continuous linear functionals.

Because of the presence in probabilistic computation of numbers expressing the likelihood of each possible continuation, vector spaces provide a natural model for probabilistic behaviours thanks to their structural dependence on a field such as the reals $\mathbb{R}$ or the complex numbers $\mathbb{C}$. Moreover, topology is essential in the infinite dimensional case in order to be able to appropriately define the set of continuous linear functionals, i.e. the topological dual of the vector space which they act on.

Therefore, in order to define a categorical semantics for probabilistic λ-calculus we will consider the category $C = \mathbf{TVect}_{\mathbb{R}}$ of topological vector spaces on the field of the reals $\mathbb{R}$ [2].

2 Probabilistic Computational Monad

We base our definitions on the idea that a program denotes a morphism from the object representing the values of a certain type A to the object of computations of type B. This idea is expressed by the notion of computational monad [4, 3], originally introduced for classical computation. We introduce here an analogous notion for probabilistic computation.

Definition 1 *The probabilistic monad $(\mathcal{T}, \eta, \mu)$ is a monad over the category $\mathbf{TVect}_{\mathbb{R}}$, i.e. an endofunctor equipped with a pair of natural transformations $\mu : \mathcal{T}^2 \to \mathcal{T}$ and $\eta : I \to T$, with I the identity functor.*

The component of μ at an object a is the morphism $\mu_a : \mathcal{T}(\mathcal{T}a) \to \mathcal{T}a$, while, considering that the action of I on the object a is just a, the component of η is given by the morphism $\eta_a : a \to \mathcal{T}a$. The general intuition behind these functors is that $\eta_a : a \to \mathcal{T}a$ gives the inclusion of values into computations, while $\mu_{\mathcal{T}a} : \mathcal{T}^2 a \to \mathcal{T}a$ flattens a computation of a computation into a computation. For the specific case of probabilistic computation, this clearly means lifting classical values to probability distributions and computing the probability distribution associated to a distribution over distributions. Since the space of probability distributions is only a subset of a topological vector space we cannot restrict to such sets but instead consider the topological vector space in which they are included. We then define $\mathcal{T}$ as a covariant 'lifting' functor, i.e. $\mathcal{T}(a) = \mathcal{V}(a)$ with $\mathcal{V}$ a topological vector space constructor, and $\mathcal{T}f(V)$ as the image of V of the continuous linear functional f; moreover, η_a lifts values to vectors and μ_a constructs vectors from linear sum of vectors.

2.1 Semantics of Probabilistic λ-calculus

By using $\mathcal{T}$ as a type constructor we can assign a meaning to the basic types ι and the arrow type $\sigma \to \tau$ by constructing the topological vector space[1] associated to the set of values of that type. Thus the interpretation of ι is the lifting $[\![\iota]\!] = \mathcal{T}(\iota)$ to the associated topological vector space (an object of $\mathbf{TVect}_{\mathbb{R}}$), while $[\![\sigma \to \tau]\!] = \mathcal{T}(\sigma \to \tau)$ are the morphisms[2] between $[\![\sigma]\!]$ and $[\![\tau]\!]$. For this interpretation we are using the fact that $\mathbf{TVect}_{\mathbb{R}}$ is an *additive, monoidal* category; in particular, additivity implies that $Hom([\![\sigma]\!], [\![\tau]\!])$ is an $\mathbb{R}$-topological vector space in its turn.

The probabilistic extension is realised by the inclusion in the calculus of probabilistic terms of type $\sigma_1 + \ldots + \sigma_n$, where σ_i is a basic or arrow type. To define a meaning of this type in our monad we use the monoidal structure of the category $\mathbf{TVect}_{\mathbb{R}}$ and define $[\![\sigma_1 + \ldots + \sigma_n]\!]$ via the co-product $\oplus$ as $[\![\sigma_1]\!] \oplus \ldots \oplus [\![\sigma_n]\!]$.

The probabilistic monad explained here provides the basis for the definition of a denotational semantics of the terms of probabilistic λ-calculus, that we plan to develop in further work.

References

[1] H. P. Barendregt. *The Lambda Calculus*, volume 103 of *Studies in Logic and the Foundations of Mathematics*. North-Holland, revised edition, 1991.

[2] J. Horvát. *Topological Vector Spaces and Distributions*, volume 1. Addison?Wesley, Reading, MA, 1966.

[3] Eugenio Moggi. Computational lambda-calculus and monads. In *Proc. of IEEE Symposium on Logic in Computer Science*, pages 14–23. IEEE Computer Society Press, 1989.

[4] Eugenio Moggi. Notions of computation and monads. *Inform. and Comput.*, 93:55–92, 1989.

[1] Concrete examples of topological vector space are Hilbert spaces and Banach spaces.

[2] A concrete example is given by the bounded linear operators on a Hilbert space.

Institutions for database schemas and datasets

Martin Glauer[1] and Till Mossakowski[2]

[1]Otto von Guericke University, Magdeburg, Germany, glauer@iks.cs.ovgu.de
[2]Otto von Guericke University, Magdeburg, Germany, till@iks.cs.ovgu.de

Abstract

Category-theoretical approaches to database systems have been studied for a long time. Yet, many approaches focus on structural features, e.g. the notions of schema integration and schema merges. In this short paper we present a draft for an institutional description of relational database schemas that also covers the data contained in these schemas.

1 Introduction

Database techniques have a long tradition in computer science and build the foundation for numerous modern applications. There are many category-theoretical approaches tackling several problems that occur when working with databases. Modelling database schemas as categories yields the intuitive notion of a schema merge as pushouts of functors [4]. A similar approach towards schema integrations was definedfor the first time in an institutional setting in [1]. The defined structures are not close to actual relational database structures. Yet, an institutional approach towards database structures yields functionalities not only on a structural, but also on the data level. Institutions were defined in [2] as a framework to cover the vast landscape of logical formalisms. The formalization presented in this paper can be a first step towards institution based logical reasoning on relational databases.

2 Formalization

We present an institution that, in contrast to [4], closely follows the Data Description Language (DDL) and the Data Manipulation Language (DML) of modern SQL-based database systems.

An object Σ of the **signature** category corresponds to a schema formulated in the DDL excluding constraint definitions. It consists of a sef of tables $\mathbb{T}^\Sigma$, a set of columns $\mathbb{C}^\Sigma$, a set of types $\mathbb{S}^\Sigma$, a family of sets of functional symbols $(\mathcal{F}_{\Sigma,w,s})_{w\in(\mathbb{S}^\Sigma)^*,s\in\mathbb{S}^\Sigma}$, and a family of sets of predicate symbols $(\mathcal{P}_{\Sigma,w})_{w\in(\mathbb{S}^\Sigma)^*}$. Additionaly, it contains functions that link tables and columns $\mathrm{col}^\Sigma(\cdot) : \mathbb{T}^\Sigma \to \wp(\mathbb{C}^\Sigma)$ and columns to their types $\tau(\cdot,\cdot) : \{(t,c)|t \in \mathbb{T}^\Sigma, c \in \mathrm{col}(t)\} \to \mathbb{S}^\Sigma$.
In the following we will use the abbreviation $\mathbb{C}^\Sigma\{\mathbb{T}\} := \{(t,c)|t \in \mathbb{T}^\Sigma, c \in \mathrm{col}^\Sigma(t)\}$.

A **signature morphism** $\sigma : \Sigma \to \Sigma'$ consists of functions $\sigma_\mathbb{T} : \mathbb{T}^\Sigma \to \mathbb{T}^{\Sigma'}$, $\sigma_\mathbb{S} : \mathbb{S}^\Sigma \to \mathbb{S}^{\Sigma'}$. σ_{col} maps tables t to functions $(col(t) \to col'(\sigma_\mathbb{T}(t)))$ on their respecive column spaces, $\sigma_\mathcal{F}$ such that all n-ary function symbols $f \in \mathcal{F}_{\Sigma,w_1,\ldots,w_n,s}$ are mapped to $\sigma_\mathcal{F}(f) \in \mathcal{F}_{\Sigma,\sigma_\mathbb{S}(w_1),\ldots,\sigma_\mathbb{S}(w_n),\sigma_\mathbb{S}(s)}$ and $\sigma_\mathcal{P}$ respectively. Additionally, the types of columns must be preserved along $\sigma_\mathbb{S}$.

A **sentence** φ in $Sen(\Sigma)$ consists of a table $t \in \mathbb{T}^\Sigma$ and a constraint where the latter can can be one of the following: A primary key $pk \in \text{col}(t)$, a foreign key $fk \in \text{col}(t) \times \mathbb{C}\{\mathbb{T}\}$, a check constraint ck that is an unquantified first-order formula over the variables $\text{col}(t)$ or a uniqueness constraint $un \in \text{col}(t)$. The translation of these sentences along a signature morphisms is the intuitive application of the corresponding functions.

An object M of the **model** category $\text{Mod}(\Sigma)$ of a signature Σ represents the data stored in each table as well as interpretation of the sorts named in Σ. Each model M consists of non-empty carrier sets M_s for $s \in \mathbb{S}^\Sigma$, a function $(f_{w_1 \ldots w_n s})_M : M_{w_1} \times \ldots \times M_{w_n} \to M_s$ for each function symbol $f \in \mathcal{F}_{w_1,\ldots,w_n,s}$, a relation $(p_{w_1 \ldots w_n})_M \subseteq M_{w_1} \times \ldots \times M_{w_n}$ for each predicate symbol $f \in \mathcal{P}_{w_1,\ldots,w_n}$, a function *data* that maps each table $t \in \mathbb{T}^\Sigma$ to a (multi-)set of functions $(\text{col}(t) \to M_{\tau(t,c)})$.

The **reduct** $M'|_\sigma$ of a model M' in $\text{Mod}(\Sigma')$ against a signature morphism $\sigma : \Sigma \to \Sigma'$ consists of carrier sets $(M'|_\sigma)_s = M'_{\sigma_\mathbb{S}(s)}$. Analogously, function and predicates are obtained by translating their sorts along $\sigma_\mathbb{S}$. The data state of a table depends on the images of its columns:

$$\forall t \in \mathbb{T}^\Sigma, c \in \text{col}(t) : data_{M|_\sigma}(t) = data_M(\sigma_\mathbb{T}(t)) \circ (\sigma_{\text{col}}(t)) \tag{1}$$

For every $M \in \text{Mod}(\Sigma)$ and $\varphi \in Sen(\Sigma)$ the **satisfaction relation** $M \models_\Sigma \varphi$ holds depending on the structure of φ: If $\varphi = (t, c)$ is a uniqueness or a primary key constraint:

$$\forall row, row' \in data_M(t) : \big(row \neq row' \Rightarrow row(c) \neq row(c')\big)$$

If $\varphi = (t, (c, (t', c')))$ is a foreign key constraint:

$$\forall row \in data_M(t), \exists row' \in data_M(t') : row(c) = row'(c')$$

If $\varphi = (t, ck)$ is a check key constraint it is evaluated as a first order formula:

$$\forall row \in data_M(t) : row \models_\Sigma{}^{FOL} ck$$

Whilst the models in the database institution described above represent a possible data state, morphisms of the category of models are the transitions between these states, i.e. statements of the DML like *INSERT*, *UPDATE*, *DELETE*. Similar morphism structures have been used in a categorical approach towards version control systems [3].

Consider a data state M and two different, concurrent manipulations (i.e. morphisms) $m_1 : M \to M_1$, $m_2 : M \to M_1$. If there is a pushout $m'_1 : M_1 \to M^*$, $m'_2 : M_2 \to M^*$ it is possible to merge both changes directly into a single data state M^*.

3 Conclusion

Whilst current approaches focus mainly on the schematic transformations of databases, the presented institution also covers the behavior on the data level. This allows the definition of formal foundations for collaborative database systems. These foundations can be used to evaluate existing collaborative database systems with respect to formal correctness or develop new such systems.

References

[1] Suad Alagić and Philip A Bernstein. A model theory for generic schema management. In *International Workshop on Database Programming Languages*, pages 228–246. Springer, 2001.

[2] Joseph A Goguen and Rod M Burstall. Institutions: Abstract model theory for specification and programming. *Journal of the ACM (JACM)*, 39(1):95–146, 1992.

[3] Samuel Mimram and Cinzia Di Giusto. A categorical theory of patches. *Electronic notes in theoretical computer science*, 298:283–307, 2013.

[4] Patrick Schultz, David I Spivak, Christina Vasilakopoulou, and Ryan Wisnesky. Algebraic databases. *arXiv preprint arXiv:1602.03501*, 2016.

Coalgebraic operations on bi-infinite streams

Joshua Holland
University of Southampton

Work by Bonchi, Sobociński and Zanasi [1, 2, 3] establishes string diagrams as a category theoretic formalisation of signal flow graphs (SFGs), a notion dating back to Shannon. They are given an operational semantics as streams, and in [4] a generalisation to bi-infinite streams is given. However, the underlying theory is quite mathematical. We seek a coalgebraic universe to interpret these semantics, defining delay, advance in time, copying and adding such streams using the properties of final coalgebras. A coalgebraic approach to bi-infinite streams was previously explored at CALCO-jnr by Silva [7], including Definition 1 and Theorem 2, although the focus was on identifying the correct approach rather than the application to SFGs motivating this work.

We write B for the set of bi-infinite streams $A^{\mathbb{Z}} = \{f : \mathbb{Z} \to A\}$ on some fixed set A.

1 Initial setup

One can consider a bi-infinite stream as a pair of streams growing in parallel, one into the future and one into the past.

$$\ldots, x_{-3}, x_{-2}, x_{-1}, x_0, x_1, x_2, \ldots \leftrightsquigarrow \begin{matrix} x_0, & x_1, & x_2, & \ldots \\ x_{-1}, & x_{-2}, & x_{-3}, & \ldots \end{matrix}$$

This motivates the "bi-infinite stream functor" $X \mapsto (A \times A) \times X$, and the following definition.

Definition 1. *The coalgebra $b = \langle \mathrm{head}_b, \mathrm{tail}_b \rangle : B \to (A \times A) \times B$ is defined by*

$$\mathrm{head}_b(f) = (f(-1), f(0)) \qquad \mathrm{tail}_b(f)(n) = \begin{cases} f(n+1) & \text{if } n \geq 0 \\ f(n-1) & \text{if } n < 0 \end{cases}$$

Theorem 2. *B is isomorphic to the final coalgebra $(A \times A)^{\mathbb{N}}$, and so also final.*

We now define alternative coalgebras on B and use finality to define shift operators forwards and backwards in time. Let $s = \langle \mathrm{head}_s, \mathrm{tail}_s \rangle$ and $s^{-1} = \langle \mathrm{head}_{s^{-1}}, \mathrm{tail}_{s^{-1}} \rangle$ where

$$\mathrm{head}_s(f) = (f(0), f(1)) \qquad \mathrm{tail}_s(f)(n) = \begin{cases} f(n+1) & \text{if } n \geq 1 \\ f(n-1) & \text{if } n < 1 \end{cases}$$

$$\mathrm{head}_{s^{-1}}(f) = (f(-2), f(-1)) \qquad \mathrm{tail}_{s^{-1}}(f)(n) = \begin{cases} f(n+1) & \text{if } n \geq -1 \\ f(n-1) & \text{if } n < -1 \end{cases}$$

and denote by σ and σ^{-1} the corresponding anamorphisms $(B, s) \to (B, b)$ and $(B, s^{-1}) \to (B, b)$. We can show that they are indeed inverse, as expected.

Theorem 3. $\sigma \circ \sigma^{-1} = \mathrm{id} = \sigma^{-1} \circ \sigma$

2 The coalgebraic universe of operations

We would like to show that a composite of anamorphisms $B \to B$ is again an anamorphism, in order to have a well-behaved category of coalgebraic operations. For coalgebras (B, t) and (B, u) with anamorphisms ϕ and ψ respectively, we would require the following to commute:

$$\begin{array}{ccccc} B & \xrightarrow{\phi} & B & \xrightarrow{\psi} & B \\ {\scriptstyle t}\downarrow & & & & \downarrow{\scriptstyle b} \\ A \times A \times B & \xrightarrow[A \times A \times \phi]{} & A \times A \times B & \xrightarrow[A \times A \times \psi]{} & A \times A \times B \end{array}$$

But this does not necessarily follow from ϕ and ψ being coalgebra homomorphisms. We therefore identify a class of anamorphisms which *do* compose well as functions.

Definition 4. *An anamorphism $\phi : B^m \to B$ is* simple *if it arises from a coalgebra where* $\mathrm{head}(\mathbf{f}) = (\alpha_1^\phi(\mathbf{f}), \alpha_2^\phi(\mathbf{f}))$ *and* $\mathrm{tail}(\mathbf{f}) = (\mathrm{tail}_b(f_1), \ldots, \mathrm{tail}_b(f_m))$ *for some functions $\alpha_1^\phi, \alpha_2^\phi : B^m \to A$. We write* $\mathrm{tail}_{\mathrm{sim}}$ *for this definition of* tail. *$\mathbb{X}_{\mathrm{sim}}$ is the category whose objects are natural numbers and where an arrow $m \to n$ is the product of n simple anamorphisms $B^m \to B$.*

Theorem 5. *$\mathbb{X}_{\mathrm{sim}}$ is a category with finite products.*

Proof. If $\phi = \prod_{i=1}^m \phi_i : B^k \to B^m$, $\psi = \prod_{j=1}^n \psi_j : B^m \to B^n$ for simple anamorphisms ϕ_i, ψ_j, their composite is $\prod_{j=1}^n (\psi\phi)_j$, where $\alpha_l^{(\psi\phi)_j} = \alpha_l^{\psi_j} \circ \phi$. The identity at n is $\prod_{i=1}^n \pi_{n,i}$, where each $\pi_{n,i}$ is the projection arising from $\mathrm{head}_{\pi_{n,i}}(f_1, \ldots, f_n) = (f_i(-1), f_i(0))$.

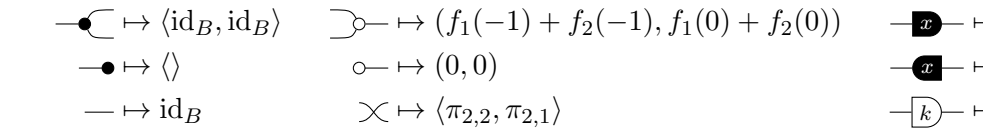
The terminal object is 0; the empty product is a unique arrow $n \to 0$. The product of m and n is $m + n$ (so $\mathbb{X}_{\mathrm{sim}}$ is a prop [6]). For arrows $B^m \xleftarrow{\phi} B^k \xrightarrow{\psi} B^n$, $\langle \phi, \psi \rangle : k \to m + n$ is $\langle \phi_1, \ldots, \phi_m, \psi_1, \ldots, \psi_n \rangle$. □

Definition 6. *$\mathbb{X}$ is the category whose objects are natural numbers, and where an arrow $m \to n$ is a functional composite of products of (not necessarily simple) anamorphisms, which maps $B^m \to B^n$.*

Note there is an embedding $\mathbb{X}_{\mathrm{sim}} \hookrightarrow \mathbb{X}$. Exactly as for $\mathbb{X}_{\mathrm{sim}}$, we have:

Theorem 7. *$\mathbb{X}$ is a category with finite products.*

As a step towards the application of signal flow graphs as in [4], we give a prop morphism $\mathsf{Mat}\,\mathsf{k}[x, x^{-1}] \to \mathbb{X}$ (fixing A as the field k); we must describe where each generator is mapped, and check that all the equations hold; this check is routine and omitted. Round brackets () denote the simple anamorphism from the coalgebra whose head mapping is as given, and angle brackets $\langle\rangle$ the product of the given anamorphisms.



3 Further questions

We might ask for a better characterisation of when composing products of anamorphisms gives another product of anamorphisms, which would allow a nicer definition of $\mathbb{X}$. However, it is not clear how to define a tail operation for a composite. Another interesting question is which anamorphisms are simple. The same anamorphism may arise from more than one coalgebra structure, and it would be good to have a better understanding of when this occurs. Further investigation is also merited into whether we can have an alternative universe as a category enriched in final coalgebras, as in [5].

References

[1] Filippo Bonchi, Paweł Sobociński, and Fabio Zanasi. Full Abstraction for Signal Flow Graphs. In *Proceedings of the 42nd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages - POPL '15*, pages 515–526, New York, New York, USA, 2015. ACM Press. doi:10.1145/2676726.2676993.

[2] Filippo Bonchi, Paweł Sobociński, and Fabio Zanasi. Interacting Hopf algebras. *Journal of Pure and Applied Algebra*, 221(1):144–184, mar 2017. arXiv:1403.7048, doi:10.1016/j.jpaa.2016.06.002.

[3] Filippo Bonchi, Paweł Sobociński, and Fabio Zanasi. The Calculus of Signal Flow Diagrams I: Linear relations on streams. *Information and Computation*, 252(v):2–29, feb 2017. doi:10.1016/j.ic.2016.03.002.

[4] Brendan Fong, Paweł Sobociński, and Paolo Rapisarda. A categorical approach to open and interconnected dynamical systems. In *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science - LICS '16*, volume 1, pages 495–504, New York, New York, USA, 2016. ACM Press. arXiv:1510.05076, doi:10.1145/2933575.2934556.

[5] Sava Krstić, John Launchbury, and Duško Pavlović. Categories of Processes Enriched in Final Coalgebras. In Furio Honsell and Marino Miculan, editors, *Foundations of Software Science and Computation Structures: 4th International Conference, FOSSACS 2001 Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2001 Genova, Italy, April 2–6, 2001 Proceedings*, pages 303–317. Springer Berlin Heidelberg, Berlin, Heidelberg, 2001. doi:10.1007/3-540-45315-6_20.

[6] Saunders Mac Lane. Categorical Algebra. *Bulletin of the American Mathematical Society*, 71(1):40–106, 1965. doi:10.1090/S0002-9904-1965-11234-4.

[7] Alexandra Silva. A coalgebraic view on bi-infinite streams. In *Calco-jnr*, Bergen, Norway, 2007. URL: http://www.alexandrasilva.org/files/calco-jnr2007.pdf.

Bisimulation Games on Coalgebras*

Barbara König[1] Christina Mika[1]

[1]Universität Duisburg-Essen, Germany

1 Introduction

The theory of coalgebras [9] knows several ways to define behavioural equivalence or bisimilarity [11]. Another, complementary, view is to characterize behavioural equivalence via a coalgebraic modal logic [10, 8]. In concurrency theory a third perspective is quite common, where behavioural equivalence is defined via attacker-defender games [12]. Such games are useful both for theoretical reasons, see for instance the role of games in the Van Benthem/Rosen theorem [7], or for didactical purposes. The game starts with two tokens on two states and the attacker tries to make a move that cannot be imitated by the defender. If the defender is always able to match the move of the attacker we can infer that the two initial states are behaviorally equivalent. If the states are not equivalent, a strategy for the attacker can be derived from a distinguishing modal logic formula.

Such games are common for standard labelled transition systems, but have been studied for other types of transition systems only to a lesser extent. For probabilistic transition systems there is a game characterization by Desharnais et al. [5], where the players can make moves to sets of states, rather than take a transition to a single state. Furthermore, in [4] a general theory of games is introduced in order to characterize process equivalences of the linear/branching time spectrum.

To our knowledge such games have not been thoroughly studied in coalgebra. We are mainly aware of [3], which describes a coalgebraic game based on the bisimulation relation.

The aim of our work is to start from the classical bisimulation game and the probabilistic version and generalize these kinds of games, in order to better understand the underlying mechanisms. This generalization allows us, given a new type of system characterized by an endofunctor, to automatically derive the corresponding game. This can for instance be useful in practice in order to demonstrate to the user that two states are (not) behaviourally equivalent.

Our game can be played for coalgebras based on arbitrary **Set**-endofunctors that preserve weak pullbacks, however we require a condition that is interestingly connected to the functor having a separating set of (monotone) predicate liftings, known from coalgebraic modal logics [10]. The proofs of our game characterization of behavioural equivalence are available in [6].

2 The Game

We fix an endofunctor $F\colon \mathbf{Set} \to \mathbf{Set}$, intuitively describing the branching type. A *coalgebra*, describing a transition system of this branching type is given by a function $\alpha\colon X \to FX$ [9].

*Partially supported by the DFG project BEMEGA.

Two states $x, y \in X$ are considered to be *behaviourally equivalent* ($x \sim y$) if there exists a coalgebra homomorphism f from α to a coalgebra $\beta\colon Y \to FY$ (i.e., a function $F\colon X \to Y$ with $\beta \circ f = Ff \circ \alpha$) such that $f(x) = f(y)$.

Since in our version of the game a player moves to a set of states, instead of a single state – as in the probabilistic game of [5] – we have to characterize such sets, for which we will use *predicates*. A predicate for a set X is a function $p\colon X \to 2$, where $2 = \{0, 1\}$. We write $p_1 \leq p_2$ if $p_1(x) \leq p_2(x)$ for all $x \in X$.

We assume the standard order $\leq = \{(0,0), (0,1), (1,1)\}$ on 2. We need to *lift preorders* wrt. F, using standard relation lifting. According to [1] relation lifting preserves preorders whenever F preserves weak pullbacks.

Definition 1 *Let $\leq$ be a preorder on X, i.e. $\leq \subseteq X \times X$. We define the preorder $\leq^F \subseteq FX \times FX$ as follows: for $t_1, t_2 \in FX$ we have that $t_1 \leq^F t_2$ if some $t \in F(\leq)$ exists such that $F\pi_i(t) = t_i$. Here $\pi_i\colon \leq \to X$ with $i \in \{1, 2\}$ are the usual projections.*

The rules of the game are as follows: at the beginning of a game, there are two marked states x, y. The aim of Player 1 (attacker) is to prove that the two states are not behaviourally equivalent. Player 2 (defender) tries to answer all moves of Player 1 in order to prove that they are equivalent.

- **Initial situation:** We are given a coalgebra $\alpha\colon X \to FX$ and want to know if two states $x, y \in X$ are behaviourally equivalent.
- **Step 1:** Player 1 chooses one of the states x, y and a predicate $p_1\colon X \to 2$.
- **Step 2:** Player 2 has to answer with a predicate $p_2\colon X \to 2$, which satisfies the condition $Fp_1 \circ \alpha(x) \leq^F Fp_2 \circ \alpha(y)$ (if Player 1 chose x) or $Fp_1 \circ \alpha(y) \leq^F Fp_2 \circ \alpha(x)$ (if Player 1 chose y).
- **Step 3:** Player 1 chooses p_i with $i \in \{1, 2\}$ and some state $x' \in X$ with $p_i(x') = 1$.
- **Step 4:** Player 2 chooses some state $y' \in X$ with $p_{\neg i}(y') = 1$.

After one round the game continues with the initial situation for the pair x', y'. Player 2 wins the game if the game continues forever or Player 1 can not make a move (in Step 3). In the other case, i.e. Player 2 gets stuck at Step 2 or Step 4, Player 1 wins.

The following theorem shows under which conditions the game faithfully characterizes behavioural equivalence. Apart from weak pullback preservation, the required conditions are equivalent to the functor having a separating set of monotone predicate liftings [10].

Theorem 2 *Assume that F preserves weak pullback, the preorder $\leq^F$ is anti-symmetric*[1] *and that the collection $(Fp\colon FX \to F2)_{\{p\colon X \to 2\}}$ of functions (indexed over all predicates p) is jointly monomorphic.*[2] *Then, given a coalgebra $\alpha\colon X \to FX$ and two states $x, y \in X$, it holds that $x \sim y$ iff Player 2 has a winning strategy for (x, y) in the game described above.*

[1] An order $\leq$ on X is anti-symmetric if for all $x, y \in X$ $x \leq y$ and $y \leq x$ implies $x = y$.

[2] A collection $(f_i)_{i \in I}$ of functions $f_i\colon X \to Y$ is jointly monomorphic if for $x_1, x_2 \in X$ with $x_1 \neq x_2$ there exists at least one index $i \in I$ such that $f_i(x_1) \neq f_i(x_2)$.

3 Future Work

An issue that we did not explain in this short note, but that we have already worked out, is the derivation of a winning strategy for Player 1, in the case where $x \not\sim y$. This is done by using a formula φ of coalgebraic modal logic that distinguishes x, y, i.e., for which $x \models \varphi$, $y \not\models \varphi$.

Our next steps will be to generalize this line of work in two dimensions: first, we will work on metric games in order for verify whether two states have a behavioural distance $d(x, y) \leq \varepsilon$. This game will be based on the Kantorovich lifting for metrics under the functor F [2]. Another dimension will be to incorporate implicit branching and trace semantics, by working in Kleisli categories instead of **Set**.

References

[1] Adriana Balan and Alexander Kurz. Finitary functors: From **Set** to **Preord** and **Poset**. In *Proc. of CALCO '11*, pages 85–99. Springer, 2011. LNCS 6859.

[2] Paolo Baldan, Filippo Bonchi, Henning Kerstan, and Barbara König. Behavioral metrics via functor lifting. In *Proc. of FSTTCS '14*, volume 29 of *LIPIcs*. Schloss Dagstuhl, 2014.

[3] A. Baltag. Truth-as-simulation: Towards a coalgebraic perspective on logic and games. Technical Report SEN-R9923, Centrum voor Wiskunde en Informatica (CWI), November 1999.

[4] Xin Chen and Yuxin Deng. Game characterizations of process equivalences. In *Proc. of APLAS '08*, pages 107–121. Springer, 2008. LNCS 5356.

[5] J. Desharnais, F. Laviolette, and M. Tracol. Approximate analysis of probabilistic processes: Logic, simulation and games. In *Proc. of QEST '08*, pages 264–273. IEEE, 2008.

[6] Barbara König and Christina Mika. Bisimulation Games on Coalgebras. `http://arxiv.org/abs/1705.10165`, 2017.

[7] Martin Otto. Elementary proof of the van Benthem-Rosen characterisation theorem. Technical Report 2342, Department of Mathematics, Technische Universität Darmstadt, 2004.

[8] D. Pattinson. Coalgebraic modal logic: soundness, completeness and decidability of local consequence. *Theoretical Computer Science*, 309(1):177 – 193, 2003.

[9] J.J.M.M. Rutten. Universal coalgebra: a theory of systems. *Theoretical Computer Science*, 249:3–80, 2000.

[10] L. Schröder. Expressivity of coalgebraic modal logic: The limits and beyond. *Theoretical Computer Science*, 390(2):230 – 247, 2008.

[11] Sam Staton. Relating coalgebraic notions of bisimulation. In *Proc. of CALCO '09*, pages 191–205. Springer, 2009. LNCS 5728.

[12] C. Stirling. Bisimulation, modal logic and model checking games. *Logic Journal of the IGPL*, 7(1):103 – 124, 1999.

Finite automata and non-commutative analogue of Λ-rings *

Takeo Uramoto

Research Center of Pure and Applied Mathematics,
Graduate School of Information Sciences, Tohoku University
takeo.uramoto@tohoku.ac.jp

Abstract

This work is a continuation of our previous work [8] on an axiomatic reformulation of *Eilenberg variety theory* [6], a branch in formal language theory that concerns a systematic classification of regular languages, finite monoids and deterministic finite automata (DFAs). In that work, we introduced the class of *semi-galois categories* and studied their general structure, where we particularly showed that every semi-galois category (with finitely generated fundamental monoid) is essentially equivalent to a local variety of DFAs (over a finite alphabet). Continuing this line, the current work reports a progress on a problem addressed in [7]; and provides a construction of semi-galois categories from (integer rings $\mathcal{O}_K$ of) number fields K (or more generally Dedekind domains with finite residue fields), which gives, on one hand, a non-commutative extension of some result in [4, 5], and on the other hand, an arithmetic semantics of Eilenberg variety theory.

1 Contribution

In our previous work [8] on an axiomatization of Eilenberg variety theory [6], the class of *semi-galois categories* was introduced, where particularly proved was that every semi-galois category $\langle \mathscr{C}, \mathrm{F} \rangle$ is equivalent to the category $\mathscr{B}_f M$ of finite M-sets for some profinite monoid M called the *fundamental monoid* of the semi-galois category, denoted $\pi_1(\mathscr{C}, \mathrm{F})$. This implies that, if (and only if) the fundamental monoid $\pi_1(\mathscr{C}, \mathrm{F})$ is topologically generated by a set A, the semi-galois category $\mathscr{C}$ is equivalent to the category $\mathscr{C}_{\mathcal{V}_A}$ consisting of those DFAs which accept languages in a fixed local variety $\mathcal{V}_A$ of regular languages over A. By this correspondence, the concept of semi-galois categories was proved equivalent to the classical concept called local varieties in Eilenberg theory (cf. §5, [7]).

One of the primary and expected merits of such axiomatization was that it would allow us to place Eilenberg theory in a wider context, apart from the original context of formal language theory. In this respect, we posed at LICS'16 (cf. §7, [7]) a problem of constructing concrete semi-galois categories, particularly from connected schemes; the current contribution reports a progress on this matter.

Our result has a preceding work pioneered by Borger and de Smit [4, 5], where they proved that the category $\mathscr{I}^{\mathrm{c}}_{\Lambda}(K)$ of Λ-*rings* that are finite étale over a number field K and have *integral models* (or *descend to* $\mathbb{F}_1$ [1]) is canonically equivalent to the category $\mathscr{B}_f DR(K)$ of finite $DR(K)$-sets,

*This work was partially supported by JSPS Kakenhi Grant Number JP16K21115.

where $DR(K)$ denotes the *Deligne-Ribet monoid*. In their proof, they showed that the action of the absolute galois group G_K of K on finite étale integral Λ-rings are forced to be abelian; and thanks to this fact, they could apply class field theory to relate finite étale integral Λ-rings with the Deligne-Ribet monoid $DR(K)$, establishing the above-mentioned equivalence. This equivalence means that the Deligne-Ribet monoid $DR(K)$ classifies finite étale integral Λ-rings over K in the same way as the absolute galois group G_K classifies finite étale algebras over K; and made a connection between class field theory and the study of Λ-rings.

Our result is a non-commutative extension of this categorical equivalence, where for a certain number-theoretic reason we drop the commutativity assumption of Λ-structures from the original definition of Λ-rings [2, 3]. As one can see in [4, 5], the commutativity assumption of Λ-structures is the source of the phenomenon that the actions of G_K on integral Λ-rings are forced to be abelian; and indeed this phenomenon was the key to apply class field theory to prove the categorical equivalence $\mathscr{I}_\Lambda^c(K) \sim \mathscr{B}_f DR(K)$. But it is also of primary interest whether there exists a profinite monoid that classifies those finite étale Λ-rings over K whose field components are non-abelian extensions over K in general. We prove that the non-abelianization of Λ-structures does not spoil the existence of such classifying profinite monoid.

Apparently it is possible to drop the commutativity assumption of Λ-structures from the original definition of Λ-rings; but the issue is that this non-commutativization makes it impossible to simulate the same argument as [4, 5] due to the lack of general non-abelian class field theory. We cope with this issue using the axiomatic characterization of semi-galois categories; in other words, the existence of classifying profinite monoid as $\mathscr{I}_\Lambda^c(K) \sim \mathscr{B}_f DR(K)$ itself can be proved without any reference to class field theory. The main results include: (I) the category $\mathscr{I}_\Lambda(K)$ of finite étale integral Λ-rings over K (in our non-commutative sense) is equivalent to the category $\mathscr{B}_f M_K$ of finite M_K-sets for some profinite monoid M_K; (II) M_K is topologically generated by the infinite set P_K of prime ideals of the integer rings $\mathcal{O}_K$ of K— therefore, each element of M_K can be represented by profinite words $u \in \widehat{P_K^*}$ over the infinite alphabet P_K; and (III) the Deligne-Ribet monoid $DR(K)$ is isomorphic to the maximal abelian quotient M_K^{ab} of our profinite monoid M_K.

Acknowledgements. I am grateful to Go Yamashita for discussions and his continuous encouragements, and in particular, to Isamu Iwanari for introducing to me the papers [4, 1]. I am also grateful to anonymous reviewers, whose comments improved this manuscript.

References

[1] J. Borger. Λ-rings and the field with one element. arXiv:0906.3146.

[2] J. Borger. The basic geometry of Witt vectors, I: The affine case. *Algebra and Number Theory*, 5(2):231–285, 2011a.

[3] J. Borger. The basic geometry of Witt vectors, II: Spaces. *Mathematische Annalen*, 351: 877–933, 2011b.

[4] J. Borger and B. de Smit. Galois theory and integral models of Λ-rings. *Bull. London Math. Soc.*, 40:439–446, 2008.

[5] J. Borger and B. de Smit. Lambda actions of rings of integers, 2011. arXiv:1105.4462.

[6] S. Eilenberg. Automata, Languages and Machines, vol.B. Academic Press, New York, 1976.

[7] T. Uramoto. Semi-galois categories I: The classical Eilenberg variety theory (extended version of [8]). arXiv:1512.04389v4.

[8] T. Uramoto. Semi-galois categories I: The classical Eilenberg variety theory. In *Proc. of LICS'16*, pages 545–554. ACM, 2016.

A completeness result for finite bisimulations up-to congruence*

Joost Winter[1]

[1]University of Warsaw, MIMUW, Banacha 2/Warsaw, Poland
June 14, 2017

Abstract

In this short paper, we will present conditions, in the context of λ-bialgebras for distributive laws between monads and endofunctors on **Set**, under which finite bisimulations up-to congruence are *complete* with respect to behavioural equivalence on freely and finitely generated λ-bialgebras. This completeness result can be seen as a continuation of the work in [13], in which a similar completeness result for λ-bisimulations (or bisimulations up-to context) was presented.

Bisimulation up-to techniques have been extensively studied in recent years, for example in [4], [10], [5], [12], [6], and many other sources. A large benefit of using up-to techniques is that we can often suffice by using smaller relations than we would need if we would use ordinary bisimulations: in some cases, states of coalgebras or bialgebras can be linked by finite bisimulations up-to, but only by infinite ordinary bisimulations. So far, most of the work on bisimulation up-to has focused on *soundness*, however in this paper, continuing the work in [13], the focus is completeness.

Both the present work and the work in [13] require a setting of an endofunctor F on **Set** preserving weak pullbacks, a monad (T, η, μ) on **Set**, and a distributive law of the monad (T, η, μ) over F. In [13], the main result showed that if finitely generated (T, η, μ)-algebras are preserved under taking kernel pairs, then behaviourally equivalent states in freely and finitely generated λ-bialgebras are linked by finite λ-bisimulations. In this paper, we show that if every finitely generated (T, η, μ)-algebra is finitely presented[1], then behaviourally equivalent states in freely and finitely generated λ-bialgebras are linked by finite bisimulations up-to congruence (on a potentially larger, but still freely and finitely generated, λ-bialgebra). We note that there are known cases where the latter precondition holds, but the former does not.

A comprehensive and general introduction to distributive laws and λ-bialgebras can be found in [2], however, all of the concepts used here can also be found, presented more concisely and geared towards a result similar to the one in this paper, in [13]. Some of the applications presented in this paper apply to weighted automata, a general introduction to which can be found in [7]. Various techniques for bisimulation up-to, including bisimulations up-to congrence and up-to context, are extensively treated in e.g. [10], [5], [12], and [6], and bisimulation up-to congruence also plays a big role in [4]. Finally, the work in this paper can most probably be related to recent work [9], in which the notion of a proper semiring has been extended categorically to that of a proper functor in a category of algebras.

*This work was supported by the Polish National Science Centre (NCN) grant 2012/07/E/ST6/03026.

[1]We presently also assume that F has a final coalgebra, but the author conjectures that this condition can be removed.

We define a *congruence* on a category $\mathbf{C}$ as an internal equivalence relation on $\mathbf{C}$: see e.g. [1, Definition 3.9] (there simply called *equivalence relation*) for a definition.

Definition 1. Let F be an endofunctor on $\mathbf{Set}$, let (T, η, μ) be a monad on $\mathbf{Set}$, and let λ be a distributive law of the monad (T, η, μ) over the endofunctor F. Given a λ-bialgebra (X, α, δ), a relation R on X is a *bisimulation up-to congruence* on (X, α, δ) if and only if the least congruence containing R is a bisimulation on the coalgebra (X, δ).

Note that, as a result of the soundness of bisimulation, from this definition it *directly* follows that bisimulations up-to congruence are sound. We observe that our definition in fact coincides with the definition given in [12] whenever F preserves weak pullbacks:

Proposition 2. *If F preserves weak pullbacks, this definition of bisimulation up-to congruence coincides with that in [12].*

We can now state the main result, which can be seen as a completeness result for finite bisimulations up-to congruence (in a similar manner to how the main result of [13] was a completeness result for finite λ-bisimulations):

Theorem 3. *Let F be a* $\mathbf{Set}$*-endofunctor preserving weak pullbacks, such that a final F-coalgebra exists, let (T, η, μ) be a monad on* $\mathbf{Set}$ *such that finitely generated and finitely presented (T, η, μ)-algebras coincide, and let λ be a distributive law of the monad (T, η, μ) over the endofunctor F.*

Given a finite FT-coalgebra (X, δ), if states $x, y \in TX$ are behaviourally equivalent with respect to the λ-bialgebra $(TX, \mu_X, \hat{\delta})$ (with $\hat{\delta} = F\mu_X \circ \lambda_{TX} \circ T\delta$), then there is a finite FT-coalgebra (Y, γ) such that (X, δ) is a sub-FT-coalgebra of (Y, γ), and a finite bisimulation up-to congruence R on the λ-bialgebra $(TY, \mu_Y, \hat{\gamma})$ with $(x, y) \in R$.

This gives us the following variant of the main result in [8]. This is a proper extension of the decidability result for Noetherian semirings because $\mathbb{N}$ is an example of a semiring that is not Noetherian but that has the property of finitely generated semimodules being finitely presented, as a result of the fact that $\mathbb{N}$-semimodules are precisely commutative monoids and Rèdei's Theorem, see [11]. For a definition of an effectively presentable semiring, see [8].

Corollary 4. *Let S be a semiring such that every left-S-semimodule that is finitely generated is also finitely presented, and let S be moreover effectively presentable. Then equivalence of S-weighted automata is decidable.*

For the tropical semiring $\mathbb{T}$, with carrier $\mathbb{N} \cup \infty$, addition given by min and multiplication given by $+$, we obtain the following corollary, using the undecidability of equivalence $\mathbb{T}$-weighted automata:

Corollary 5. *Not all finitely generated $\mathbb{T}$-semimodules are finitely presented.*

Similarly, using the undecidability of equivalence of context-free languages (and their bialgebraic presentation in [3]), we obtain:

Corollary 6. *Not all finitely generated idempotent semirings are finitely presented.*

We leave as future work establishing more precise relationships between the notions of 'finitely generated algebras are finitely presented', 'completeness of finite bisimulation up-to congruence', and the notion of proper semirings (and the more general notion of proper functors, recently introduced in [9]).

References

[1] J. Adámek, J. Rosický, and E. M. Vitale. *Algebraic Theories: A Categorical Introduction to General Algebra*. Cambridge Tracts in Mathematics, 2010.

[2] Falk Bartels. *On Generalized Coinduction and Probabilistic Specification Formats*. PhD thesis, Vrije Universiteit Amsterdam, 2004.

[3] Marcello M. Bonsangue, Helle H. Hansen, Alexander Kurz, and Jurriaan Rot. Presenting distributive laws. In Reiko Heckel and Stefan Milius, editors, *CALCO*, volume 8089 of *Lecture Notes in Computer Science*, pages 95–109. Springer, 2013.

[4] Filippo Bonchi and Damien Pous. Checking NFA equivalence with bisimulations up to congruence. In *POPL*, pages 457–468. ACM, 2013.

[5] Filippo Bonchi, Daniela Petrisan, Damien Pous, and Jurriaan Rot. Coinduction up-to in a fibrational setting. In *CSL-LICS*, pages 20:1–20:9. ACM, 2014.

[6] Filippo Bonchi, Daniela Petrisan, Damien Pous, and Jurriaan Rot. A general account of coinduction up-to. *Acta Inf.*, 54(2):127–190, 2017.

[7] Jean Berstel and Christophe Reutenauer. *Noncommutative Rational Series with Applications*. Cambridge University Press, 2011.

[8] Zoltán Ésik and Andreas Maletti. Simulation vs. equivalence. In Hamid R. Arabnia, George A. Gravvanis, and Ashu M. G. Solo, editors, *FCS*, pages 119–124. CSREA Press, 2010.

[9] Stefan Milius. Proper functors and their rational fixed point, 2017. To be presented at *CALCO 2017*.

[10] Jurriaan Rot, Filippo Bonchi, Marcello M. Bonsangue, Damien Pous, Jan Rutten, and Alexandra Silva. Enhanced coalgebraic bisimulation, 2013. Submitted to *Mathematical Structures in Computer Science*.

[11] L. Rèdei. *The Theory of Finitely Generated Commutative Semigroups*. Pergamon, Oxford-Edinburgh-New York, 1965.

[12] Jurriaan Rot. *Enhanced Coinduction*. PhD thesis, Leiden University, 2015.

[13] Joost Winter. A completeness result for finite λ-bisimulations. In *FoSSaCS 2015*, pages 117–132, 2015.

A Coalgebraic Paige-Tarjan Algorithm

Thorsten Wißmann
Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany

This abstract reports on joint work with U. Dorsch, S. Milius and L. Schröder on a generic partition refinement algorithm for coalgebraic systems $\xi : X \to HX$, for $H : \mathcal{C} \to \mathcal{C}$ an endofunctor on a regular category $\mathcal{C}$ (full version at `http://arxiv.org/abs/1705.08362`). We compute a quotient coalgebra of ξ that is simple, i.e. that has no proper quotient. In Set this means that all behaviouraly equivalent states of X are identified.

We characterize sufficient conditions on H in order to show that the instantiation of the algorithm to sorted sets can be implemented to run in $\mathcal{O}((n+m)\log n)$ steps, where n is the number of states and m the number of edges in a suitable encoding of ξ.

1 Algorithm in a category

Assumption. *Let $\mathcal{C}$ be regular and $H : \mathcal{C} \to \mathcal{C}$ mono-preserving, and fix a $\xi : X \to HX$.*

Notation. *In the following, we use standard notation: regular epimorphisms are denoted by $\twoheadrightarrow$, the kernel of $f : A \to B$ by $\mathsf{ker}(f)$, and its image by $A/\mathsf{ker}(f)$. The morphism in the universal property of the product is denoted by $\langle \ldots \rangle$. We use $\chi_S : X \to 2$ to denote the characteristic function of $S \subseteq X$.*

The generic algorithm is parametric in a select routine that maps a chain of regular epis $E \twoheadrightarrow F \twoheadrightarrow G$ to a morphism $k : F \to K$. This routine decides which of the new information to use in the upcoming refinement step. The algorithm constructs two refining sequences Q and P of kernels on X. Invariantly, P_i is finer than Q_i, because it unravels the coalgebra structure one more step. Then $\mathsf{select}(X \twoheadrightarrow X/P_i \twoheadrightarrow X/Q_i)$ tells how to refine from Q_i to Q_{i+1}. Initially, Q_0 identifies all behaviours.

Algorithm. *Let $q_i : X \to K_i$ with $q_0 : X \xrightarrow{!} 1$ and iterate*

1. $Q_i = \mathsf{ker}(\langle q_j \rangle_{j \le i})$ *2.* $P_i = \mathsf{ker}(H\langle q_j \rangle_{j \le i} \cdot \xi)$ *3. if $Q_i = P_i$ then stop, else:*

4. $k_{i+1} \leftarrow \mathsf{select}(X \twoheadrightarrow X/P_i \twoheadrightarrow X/Q_i)$ *5.* $q_{i+1} \leftarrow X \twoheadrightarrow X/P_i \overset{k_{i+1}}{\twoheadrightarrow} K_{i+1}$ *6.* $i \leftarrow i+1$.

One can show that this construction yields a morphism $\xi/_i : X/P_i \to H(X/Q_i)$.

Theorem (Correctness)**.** *If $Q_i = P_i$, then $\xi/_i$ is a simple quotient coalgebra of ξ.*

If select always returns the identity on X/P_i, this means that all the information is directly used for the next refinement step. Then $P_i \cong Q_i$ and we obtain the final chain algorithm of [3]. The low complexity in sorted sets comes from the idea of *processing the smaller half*, which is realized by the following select function in Set:

Example. *For $E \overset{f}{\twoheadrightarrow} F \overset{g}{\twoheadrightarrow} G$, if there is some $x \in E$ whose f equivalence class $S \subseteq E$ is at most half the size of its gf equivalence class $C \subseteq E$, let* select *return $\langle \chi_{f[S]}, \chi_{f[C]} \rangle : F \to 2 \times 2$.*

2 Optimizations in sorted sets

Q_{i+1} can be computed incrementally by $Q_{i+1} = Q_i \cap \ker q_{i+1}$. In order to compute P_{i+1} incrementally, too, we need some more assumptions on H and select.

Definition. *H is* zippable *if $\langle H(A+!), H(! + B)\rangle : H(A + B) \to H(A + 1) \times H(1 + B)$ is a monomorphism.*

Examples of zippable functors include polynomial functors, and zippable functors are closed under products, coproducts, and subfunctors. Monoid-valued functors $M^{(-)}$ for a commutative monoid M are also zippable, and thus also the finitary powerset $\mathcal{P}_\mathrm{f} \cong \mathbb{B}^{(-)}$, the bag functor $\mathcal{B}_\mathrm{f} \cong \mathbb{N}^{(-)}$, and the finite distribution functor $\mathcal{D} \subseteq \mathbb{R}^{(-)}$. However, $\mathcal{P}_\mathrm{f}\mathcal{P}_\mathrm{f}$ is not zippable, and so zippable functors are not closed under quotients and composition. We remedy the latter by working in sorted sets.

Definition. *We say that $a : D \to A$ and $b : D \to B$ are* jointly transitive *if $\ker a \cup \ker b$ is a kernel. In Set, this means that for each $x \in D$ either the a-equivalence class is included in its b-equivalence class or the other way around.*

As a standalone result, we have for zippable Set-functors H that if a and b are jointly transitive, then $\ker\langle Ha, Hb\rangle = \ker H\langle a, b\rangle$. Applying this result to the setup of the algorithm, we have that if $\mathsf{select}(f, g)$ and g are jointly transitive, then $P_{i+1} = P_i \cap \ker(Hq_{i+1} \cdot \xi)$.

3 Implementing Kernel Intersections Efficiently

We define an interface that needs to be implemented for each functor H, in order to compute P_{i+1} as the above intersection. The interface requires the functor to be represented in terms of edges with labels L, i.e. a map $HY \to \mathcal{B}_\mathrm{f}(L \times Y)$, where $\mathcal{B}_\mathrm{f}$ denotes multisets. Furthermore, the interface contains an abstract set of weights W and a measure-like map $w : \mathcal{P}_\mathrm{f}Y \to HY \to W$. Intuitively, $w(s, t)$ is the weight of the set s in the flat term t; e.g. for the powerset functor $\mathcal{P}_\mathrm{f}$, $w(s, t)$ counts the elements in $s \cap t$ and for distributions $\mathcal{D}$, $w(s, t)$ denotes the accumulated probability of the set s in the distribution t.

Functor:	$G^{(-)}$ group valued	$\mathcal{B}_\mathrm{f}$	$\mathcal{D}$	$\mathcal{P}_\mathrm{f}$	Polynomial H_Σ
Labels L:	G	$\mathbb{N}$	$[0, 1]$	1	$\mathbb{N}$
Weights W:	$G^{(2)} = G \times G$	$\mathcal{B}_\mathrm{f}2$	$\mathcal{D}2$	$\mathbb{N}$	$H_\Sigma 2$
$w(C)$, $C \subseteq Y$:	$G\chi_C$	$\mathcal{B}_\mathrm{f}\chi_C$	$\mathcal{D}\chi_C$	$\lvert C \cap (-)\rvert$	$H_\Sigma\chi_C$

The core part of the interface is a function $\mathsf{update} : \mathcal{B}_\mathrm{f}L \times W \to W \times H(2 \times 2) \times W$, which incrementally computes the weights of $S \subseteq C \subseteq X$ and the value of $H\langle\chi_S, \chi_C\rangle \cdot \xi(x)$ in order to obtain the refinement P_{i+1}.

Theorem. *For n states and m edges, the algorithm runs in $\mathcal{O}((m + n) \cdot \log n)$.*

Examples.

1. *For $H = \mathcal{P}_\mathrm{f}$, we obtain the classical algorithm by Paige and Tarjan [4], with the same complexity $\mathcal{O}((m + n) \cdot \log n)$.*
2. *For $H = \mathbb{R}^{(-)}$ we obtain the Markov chain lumping algorithm by Valmari and Franceschinis [5], with the same complexity $\mathcal{O}((m + n) \cdot \log n)$.*

3. *For polynomial functors Σ we obtain an $\mathcal{O}(n \cdot s \cdot \log n)$ algorithm, where s is the maximal arity in Σ; for the case $2 \times (-)^A$, this is the complexity of Hopcroft's algorithm [2].*

4. *For simple (resp. general) Segala systems $\mathcal{P}_\mathrm{f}(A \times -) \cdot \mathcal{D}$ (resp. $\mathcal{P}_\mathrm{f} \cdot \mathcal{D}(A \times -)$), we instantiate to two-sorted sets to obtain a zippable functor. Then our algorithm runs in $\mathcal{O}(m \cdot \log m)$ improving the complexity from [1].*

References

[1] Christel Baier, Bettina Engelen, and Mila Majster-Cederbaum. Deciding bisimilarity and similarity for probabilistic processes. *J. Comput. Syst. Sci.*, 60:187–231, 2000.

[2] John Hopcroft. An $n \log n$ algorithm for minimizing states in a finite automaton. In *Theory of Machines and Computations*, pages 189–196. Academic Press, 1971.

[3] Barbara König and Sebastian Küpper. Generic partition refinement algorithms for coalgebras and an instantiation to weighted automata. In *IFIP TCS 2014*.

[4] Robert Paige and Pobert E. Tarjan. Three partition refinement algorithms. *SIAM J. Comput.*, 16(6):973–989, 1987.

[5] Antti Valmari and Giuliana Franceschinis. Simple $\mathcal{O}(m \log n)$ time Markov chain lumping. In *TACAS 2010*, volume 6015 of *LNCS*, pages 38–52. Springer.