Coinduction: Automata, Formal Proof, Companions

Damien Pous
Univ Lyon, CNRS, ENS de Lyon, UCB Lyon 1, LIP, Lyon, France
`http://perso.ens-lyon.fr/damien.pous/`
Damien.Pous@ens-lyon.fr

Abstract

Coinduction is a mathematical tool that is used pervasively in computer science: to program and reason about infinite data-structures, to give semantics to concurrent systems, to obtain automata algorithms. We present some of these applications in automata theory and in formalised mathematics. Then we discuss recent developments on the abstract theory of coinduction and its enhancements.

2012 ACM Subject Classification Theory of computation → Logic

Keywords and phrases Coinduction, Automata, Coalgebra, Formal proofs

Digital Object Identifier 10.4230/LIPIcs.CALCO.2019.4

Category Invited Paper

Funding This work has been funded by the European Research Council (ERC) under the European Union's Horizon 2020 programme (CoVeCe, grant agreement No 678157).

Induction and coinduction

Induction and coinduction are used in two main ways in computer science: to define datatypes and compute with those, and to define predicates and reason about them. The former can be presented using category theory: inductive datatypes such as natural numbers, lists, or trees can be modelled as initial algebras, while coinductive datatypes such as streams, infinite trees, automata, or labelled transitions systems can be modelled as final coalgebras. In contrast, we use lattice theory for predicates: inductive predicates such as reachability or derivability in a proof system are presented as least fixpoints while coinductive predicates such as divergence, bisimilarity, or language equivalence, are presented as greatest fixpoints.

When doing a proof by induction, one has to be careful about two things: 1/ choosing an appropriate induction principle (e.g., simple induction, rank-2 induction, or course of value induction), and 2/ choosing a strong enough invariant. We shall see that the very same observation applies with coinduction.

Automata algorithms

Take for instance algorithms for checking language equivalence of finite deterministic automata. Language equivalence can be characterised as a greatest fixpoint, so that it can be checked using a coinductive algorithm: start from a relation consisting of the pair of initial states, and widen this relation until it becomes a *bisimulation* [5]. This iterative process corresponds to point 2/ above: we iteratively refine an initial guess until we get a proper invariant. Such an algorithm can be made more efficient by choosing a more powerful coinduction principle, e.g., *bisimulations up to equivalence*, or *bisimulations up to congruence* for non-deterministic automata; this is point 1/ above.

Formal proofs and equational theories

Coinduction can be used for many automata algorithms, which in turn can be used in the context of formal proofs, to improve automation and delegate administrative steps to the computer. Indeed, several equational theories can be decided by characterising them in terms of language equivalence. The key example is that of *Kleene algebra*: this (quasi)equational theory admits binary relations and formal languages as free models [13, 15, 4], and is decidable in PSpace using automata algorithms. Therefore, proof obligations corresponding to this fragment can be discharged automatically in proof assistants [6], using so-called *reflexive tactics*. Another important theory is that of Kleene algebra with tests (KAT) [14], which can be decided using automata on guarded strings. This was used successfully to reason about while programs and flowchart schemes in the Coq proof assistant [18].

Other theories include Kleene algebra with converse [3, 10], Kleene allegories [7, 17], and concurrent Kleene algebra [8]. Those respectively require working with downward-closed languages, languages of graphs, and languages of partially ordered multisets (pomsets) [20]. Decidability can be obtained by designing appropriate automata models, and by characterising language equivalence as a greatest fixpoint (a coinductive predicate). In some cases, completeness can be obtained by reasoning about this greatest fixpoint [9]. Those various results are technically involved, so that formalising them in a proof assistant in order to extend the aforementioned reflexive tactics would require an important work.

Theory of enhanced coinduction in complete lattices

For predicates, the theory of coinduction can be expressed in complete lattices, starting from Knaster-Tarski's theorem [12, 25]: every monotone function b in a complete lattice admits a greatest fixpoint νb. Enhancements, or up-to techniques, have been studied by Sangiorgi [23, 24]. Given a function b, the idea is to find other functions b' that are easier to use, and such that $\nu b' = \nu b$. A recent proposal [19] consists in using the function $b' = bt$, where t, the *companion* of b is the largest function f such that $fb \leq bf$. This simple idea greatly simplifies the whole theory: the companion is a closure operator and it intuitively contains all potential enhancements. It moreover makes it possible present coinductive proofs on the fly, without needing to announce the invariant upfront – a important feature when it comes of formalisation in proof assistants [11].

Theory of enhanced coinduction in category theory

Streams of real numbers form the final coalgebra for the functor $BX = \mathbb{R} \times X$. This observation makes it possible to define streams corecursively: it suffices to provide a coalgebra for B. The constant streams, the stream *nat* of natural numbers, and the pointwise addition of streams can be defined in this way. Note that for *nat*, one has to define a coalgebra that provides not only *nat*, but also all its suffixes; this is point 2/ above: one has to provide a strong enough invariant. In slightly more involved situations, one has to use a stronger corecursion principle (point 1/ above). This the case for the convolution product, whose natural definition builds on addition [22]. A standard solution[16, 1] consists in using *distributive laws* $\lambda : FB \Rightarrow BF$, and such techniques were recently implemented in Isabelle/HOL [2]. Given the aforementioned situation in complete lattices, one can naturally ask whether there exists a “largest” such distributive law, a *companion* T for B [21]. If it exists, the companion is a monad; if B preserves the codensity monad of its final sequence, then the companion is that codensity; this is the case for polynomial functors, and in this case the companion can be characterised in terms of *causal algebras* on the final coalgebra. An intriguing open question is whether the finite powerset functor admits a companion.

References

1 F. Bartels. *On generalised coinduction and probabilistic specification formats*. PhD thesis, CWI, Amsterdam, April 2004.

2 Jasmin Christian Blanchette, Aymeric Bouzy, Andreas Lochbihler, Andrei Popescu, and Dmitriy Traytel. Friends with Benefits - Implementing Corecursion in Foundational Proof Assistants. In *ESOP*, volume 10201 of *LNCS*, pages 111–140. Springer, 2017. `doi:10.1007/978-3-662-54434-1_5`.

3 S. L. Bloom, Z. Ésik, and G. Stefanescu. Notes on equational theories of relations. *Algebra Universalis*, 33(1):98–126, 1995. `doi:10.1007/BF01190768`.

4 Maurice Boffa. Une remarque sur les systèmes complets d'identités rationnelles. *Informatique Théorique et Applications*, 24:419–428, 1990. URL: `http://archive.numdam.org/article/ITA_1990__24_4_419_0.pdf`.

5 Filippo Bonchi and Damien Pous. Checking NFA equivalence with bisimulations up to congruence. In *POPL*, pages 457–468. ACM, 2013. `doi:10.1145/2429069.2429124`.

6 Thomas Braibant and Damien Pous. Deciding Kleene Algebras in Coq. *LMCS*, 8(1):1–16, 2012. `doi:10.2168/LMCS-8(1:16)2012`.

7 Paul Brunet and Damien Pous. Petri automata for Kleene allegories. In *LICS*, pages 68–79. ACM, 2015. `doi:10.1109/LICS.2015.17`.

8 Paul Brunet, Damien Pous, and Georg Struth. On decidability of Concurrent Kleene Algebra. In *CONCUR*, volume 85 of *LIPIcs*, pages 24:1–24:16. Schloss Dagstuhl, 2017. `doi:10.4230/LIPIcs.CONCUR.2017.28`.

9 Amina Doumane and Damien Pous. Completeness for identity-free Kleene Lattices. In *CONCUR*, volume 118 of *LIPIcs*, pages 18:1–18:17. Schloss Dagstuhl, 2018. `doi:10.4230/LIPIcs.CONCUR.2018.18`.

10 Z. Ésik and L. Bernátsky. Equational properties of Kleene algebras of relations with conversion. *Theoretical Computer Science*, 137(2):237–251, 1995. `doi:10.1016/0304-3975(94)00041-G`.

11 Chung-Kil Hur, Georg Neis, Derek Dreyer, and Viktor Vafeiadis. The power of parameterization in coinductive proof. In *POPL*, pages 193–206. ACM, 2013. `doi:10.1145/2429069.2429093`.

12 B. Knaster. Un théorème sur les fonctions d'ensembles. *Annales de la Société Polonaise de Mathématiques*, 6:133–134, 1928.

13 D. Kozen. A Completeness Theorem for KLeene Algebras and the Algebra of Regular Events. *Information and Computation*, 110(2):366–390, 1994. `doi:10.1006/inco.1994.1037`.

14 D. Kozen. Kleene algebra with tests. *Transactions on Programming Languages and Systems*, 19(3):427–443, May 1997. `doi:10.1145/256167.256195`.

15 Daniel Krob. A Complete System of B-Rational Identities. In *ICALP*, volume 443 of *LNCS*, pages 60–73. Springer, 1990. `doi:10.1007/BFb0032022`.

16 Marina Lenisa, John Power, and Hiroshi Watanabe. Distributivity for endofunctors, pointed and co-pointed endofunctors, monads and comonads. *Electronical Notes in Computer Science*, 33:230–260, 2000. `doi:10.1016/S1571-0661(05)80350-0`.

17 Yoshiki Nakamura. Partial derivatives on graphs for Kleene allegories. In *LiCS*, pages 1–12. IEEE, 2017. `doi:10.1109/LICS.2017.8005132`.

18 Damien Pous. Kleene Algebra with Tests and Coq tools for while programs. In *ITP*, volume 7998 of *LNCS*, pages 180–196. Springer, 2013. `doi:10.1007/978-3-642-39634-2_15`.

19 Damien Pous. Coinduction all the way up. In *LICS*, pages 307–316. ACM, 2016. `doi:10.1145/2933575.2934564`.

20 Damien Pous. On the positive calculus of relations with transitive closure. In *STACS*, volume 96 of *LIPIcs*, pages 3:1–3:16. Schloss Dagstuhl, 2018. `doi:10.4230/LIPIcs.STACS.2018.3`.

21 Damien Pous and Jurriaan Rot. Companions, Codensity, and Causality. In *FoSSaCS*, volume 10203 of *LNCS*. Springer, 2017. Extended version at `https://arxiv.org/abs/1712.08526`. `doi:10.1007/978-3-662-54458-7_7`.

22 Jan J. M. M. Rutten. A coinductive calculus of streams. *Math. Struct. in Comp. Sci.*, 15(1):93–147, 2005. `doi:10.1017/S0960129504004517`.

23 D. Sangiorgi. On the Bisimulation Proof Method. *Math. Struct. in Comp. Sci.*, 8:447–479, 1998. `doi:10.1017/S0960129598002527`.

24 Davide Sangiorgi and David Walker. *The π-calculus: a Theory of Mobile Processes*. Cambridge University Press, 2001.

25 A. Tarski. A Lattice-Theoretical Fixpoint Theorem and its Applications. *Pacific Journal of Mathematics*, 5(2):285–309, June 1955.