Report from Dagstuhl Seminar 13422

Nominal Computation Theory

Edited by

Mikołaj Bojańczyk[1], Bartek Klin[2], Alexander Kurz[3], and Andrew M. Pitts[4]

1 **University of Warsaw, PL,** `bojan@mimuw.edu.pl`
2 **University of Warsaw, PL,** `klin@mimuw.edu.pl`
3 **University of Leicester, GB,** `kurz@mcs.le.ac.uk`
4 **University of Cambridge, GB,** `andrew.pitts@cl.cam.ac.uk`

Abstract

This report documents the program and the outcomes of Dagstuhl Seminar 13422 "Nominal Computation Theory". The underlying theme of the seminar was nominal sets (also known as sets with atoms or Fraenkel-Mostowski sets) and they role and applications in three distinct research areas: automata over infinite alphabets, program semantics using nominal sets and nominal calculi of concurrent processes.

Seminar 13–16. October, 2013 – www.dagstuhl.de/13422
1998 ACM Subject Classification F.1.1 Models of computation – automata, Turing machines, computability theory, relations between models, F.3.2 Semantics of programming languages – denotational semantics, operational semantics, process models, F.4.1 Mathematical logic – logic and constraint programming, mechanical theorem proving, set theory, F.1.2 Modes of computation – alternation and nondeterminism, D.3.3 Language constructs and features – control structures, data types and structures
Keywords and phrases nominal sets, Fraenkel-Mostowski sets
Digital Object Identifier 10.4230/DagRep.3.10.58
Edited in cooperation with Joanna Ochremiak

1 Executive Summary

Mikołaj Bojańczyk
Bartek Klin
Alexander Kurz
Andrew M. Pitts

License Creative Commons BY 3.0 Unported license
© Mikołaj Bojańczyk, Bartek Klin, Alexander Kurz, and Andrew M. Pitts

The short Dagstuhl seminar 13422 "Nominal Computation Theory" took place from October 13th to 16th, 2013. The topic of the seminar was the theory of nominal sets and their applications to Computer Science. The seminar arose from a recent exciting and unexpected confluence of the following three distinct research directions.

- The research in automata theory on automata over infinite alphabets with applications to querying XML and databases.
- The research in program semantics on nominal sets, with many applications to the syntax and semantics of programming language constructs that involve binding, or localising names.

Except where otherwise noted, content of this report is licensed under a Creative Commons BY 3.0 Unported license
Nominal Computation Theory, *Dagstuhl Reports*, Vol. 3, Issue 10, pp. 58–71
Editors: Mikołaj Bojańczyk, Bartek Klin, Alexander Kurz, and Andrew M. Pitts

Dagstuhl Reports
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

- The research in concurrency on nominal calculi (π-calculus, etc) with applications to the automatic verification of process specifications.

In each of these three topics, an important role is played by name (or atom) symmetries and permutations, albeit for a priori different reasons. In the first case they arise from the way automata use registers to store letters, in the second case they are used to define the notion freshness, in the third case they are needed to minimize automata. In all three cases there is a connection with mathematical model theory, which is aimed at studying classes of mathematical structures definable by logical theories. The permutations allowed on atomic names can be usefully understood as automorphisms of a relational structure on those names. Model-theoretic notions such as homogeneity, algebraic closure and oligomorphic groups turn out very useful in describing those relational structures on atoms that yield meaningful theories of nominal sets.

The aim of the seminar was to profit from the excitement created by the confluence described above and to explore new directions with a new mix of research communities from computer science and mathematical logic. The main topics of interest included: automata and complexity theory in nominal sets, verification of nominal automata, symmetry in domain theory, and nominal programming.

The seminar was attended by 30 participants from 8 countries; 20 of them gave presentations, whose abstracts are included in this document. Four of the presentations (A. Pitts, M. Bojańczyk, B. Klin and N. Tzevelekos) were extended tutorials that presented various points of view on the background topics of the meeting. Other speakers presented the current state of the art in the field, with topic varying from mathematical insights into the nature of nominal sets (A. Blass, D. Petrisan), to applications in automata theory (V. Ciancia, T. Colcombet, S. Lasota, T. Suzuki) and computation theory (A. Dawar, S. Toruńczyk), semantics and domain theory (R. Crole, J. Gabbay, S. Loesch, A. Murawski), process calculi (U. Montanari), Petri nets (R. Lazic), logic programming (J. Cheney) and theorem proving (C. Urban).

2 Table of Contents

Executive Summary
Mikołaj Bojańczyk, Bartek Klin, Alexander Kurz, and Andrew M. Pitts 58

Overview of Talks

Set-theoretic aspects of orbit-finiteness
Andreas R. Blass 62

Orbit finiteness
Mikołaj Bojańczyk 62

Nominal logic programming
James Cheney 62

A decidable class of (nominal) omega-regular languages over an infinite alphabet
Vincenzo Ciancia 63

On the Use of Guards for Logics with Data
Thomas Colcombet 63

Nominal Lambda Calculus: An Internal Language for FM-Cartesian Closed Categories
Roy L. Crole 64

Symmetric Circuits and Fixed-Point Logics
Anuj Dawar 64

Nominal duality theory for new, forall, and lambda.
Jamie Gabbay 64

Programming in nominal sets: a survey
Bartek Klin 65

A machine-independent characterization of timed languages
Sławomir Lasota 66

Nominal nets
Ranko Lazic 66

Concurrent Domain Theory with Nominal Sets
Steffen Loesch 67

Named Graphs and HD Automata for Network-Conscious pi-Calculus
Ugo Montanari 67

Algorithmic games for full ground references
Andrzej Murawski 68

On Orbit finiteness and Limits in Nominal Sets
Daniela Petrisan 68

Nominal Sets: Introduction and Survey
Andrew M. Pitts 68

Nominal Automata and Regular Expressions
Tomoyuki Suzuki 69

Turing machines with atoms
Szymon Toruńczyk 69

Names, games and automata
Nikos Tzevelekos 70

Nominal Isabelle, or, How Not to be Intimidated by the Variable Convention
Christian Urban 70

Participants 71

3 Overview of Talks

3.1 Set-theoretic aspects of orbit-finiteness

Andreas R. Blass (University of Michigan, US)

License Creative Commons BY 3.0 Unported license
© Andreas R. Blass

I show that a set in a permutation model of set theory is orbit-finite, in the sense of nominal computation theory, if and only if its power set in the permutation model is Dedekind-finite there. I give some equivalent characterizations of Dedekind-finiteness of the power set, and I also discuss some other notions of finiteness in set theory lacking the axiom of choice.

3.2 Orbit finiteness

Mikołaj Bojańczyk (University of Warsaw, PL)

License Creative Commons BY 3.0 Unported license
© Mikołaj Bojańczyk
Joint work of Bojańczyk, Mikołaj; Klin, Bartosz; Lasota, Sławomir; Toruńczyk, Szymon

There are two principal topics of this tutorial:

1. Sets with atoms can be based on an arbitrary logical structure. So instead of a countable set of names with equality, the atoms can be the rational numbers with order, or the integers with successor.
2. In the presence of atoms, one can consider a relaxed version of finiteness, called orbit-finiteness. When that logical structure for the atoms is oligomorphic (which is a notion from model theory that is equivalent to being omega-categorical), then the notion of orbit-finiteness is well behaved in that:
   a. there are several equivalent definitions of orbit-finiteness;
   b. orbit-finite sets are closed under finite products and finitely supported subsets.

Furthermore, under natural effectivity assumptions on the atoms (e.g. a decidable first-order theory), orbit-finite sets can be represented by data structures and manipulated by algorithms.

3.3 Nominal logic programming

James Cheney (University of Edinburgh, GB)

License Creative Commons BY 3.0 Unported license
© James Cheney
Joint work of Cheney, James; Urban, Christian; Momigliano, Alberto
Main reference J. Cheney, C. Urban, "Nominal logic programming," ACM Trans. Program. Lang. Syst. 30, 5, Article 26 (September 2008), 47 pp., 2008.
URL http://dx.doi.org/10.1145/1387673.1387675

Nominal logic programming is an extension to logic programming to incorporate features of nominal logic, such as freshness constraints, names, abstraction, and the new-quantifier. It is suitable for direct translations of many typical formal systems (type systems, functional and concurrency calculi, and languages with generative features such as references). In this talk I present examples of programs in an implemented nominal logic programming language

called alphaProlog, and show how we can search automatically for shallow counterexamples to desired properties of such programs. I also present an overview of the least Herbrand model and least fixed-point semantics of nominal logic programs, and describe the challenges involved in unification modulo equivariance and how to overcome them.

3.4 A decidable class of (nominal) omega-regular languages over an infinite alphabet

Vincenzo Ciancia (CNR – Pisa, IT)

License Creative Commons BY 3.0 Unported license
© Vincenzo Ciancia
Joint work of Ciancia, Vincenzo; Sammartino, Matteo
Main reference V. Ciancia, M. Sammartino, "A decidable class of (nominal) omega-regular languages over an infinite alphabet," arXiv:1310.3945v1 [cs.FL], 2013.
URL http://arxiv.org/abs/1310.3945v1

We define a class of languages of infinite words over infinite alphabets, and the corresponding automata. The automata used for recognition are a generalisation of deterministic Muller automata to the setting of nominal sets. Remarkably, the obtained languages are determined by their ultimately periodic fragments, as in the classical case. Closure under complement, union and intersection, and decidability of emptiness and equivalence are preserved by the generalisation. This is shown by using finite representations of the (otherwise infinite-state) defined class of automata.

3.5 On the Use of Guards for Logics with Data

Thomas Colcombet (CNRS / University Paris-Diderot, FR)

License Creative Commons BY 3.0 Unported license
© Thomas Colcombet
Joint work of Colcombet, Thomas; Ley, Clemens; Puppis, Gabriele
Main reference T. Colcombet, C. Ley, G. Puppis, "On the Use of Guards for Logics with Data," in Proc. of the 36th Int'l Symp. of Mathematical Foundations of Computer Science (MFCS'11), LNCS, Vol. 6907, pp. 243–255, Springer, 2011.
URL http://dx.doi.org/10.1007/978-3-642-22993-0_24

The notion of orbit finite data monoid (i.e., nominal monoids) was recently introduced by Bojańczyk as an algebraic object for defining recognizable languages of data words. Following Büchi's approach, we introduce the new logic 'rigidly guarded MSO' and show that the data languages definable in this logic are exactly those recognizable by orbit finite data monoids. We also establish, following this time the approach of Schützenberger, McNaughton and Papert, that the first-order variant of this logic defines exactly the languages recognizable by aperiodic orbit finite data monoids. Finally, we give a variant of the logic that captures the larger class of languages recognized by non-deterministic finite memory automata.

3.6 Nominal Lambda Calculus: An Internal Language for FM-Cartesian Closed Categories

Roy L. Crole (University of Leicester, GB)

License Creative Commons BY 3.0 Unported license
© Roy L. Crole
Joint work of Crole, Roy L.; Nebel, Frank

Theories in equational logic (also known as algebraic theories, or theories for equational reasoning) have been studied for many years. Nominal Equational Logic provides judgements both of expression equality, and of freshness of atoms. Just as Equational Logic can be enriched with function types to yield the lambda-calculus, we have developed Nominal Lambda Calculus (NLC) by enriching NEL with (atom-dependent) function types and abstraction types.

We introduce pure NLC and look in detail at some the the rules of the NLC formal system. We show that NLC is sound by giving a categorical semantics, explaining some of the complexities that arise from a dependent type system. We then consider completeness and illustrate that we cannot define a complete categorical semantics for pure NLC. However, by adding in a novel form of dependently typed atom abstraction, we get a system that is complete. We demonstrate some of the rules for such atom abstraction, and outline how the problem with completeness can be solved in a neat manner.

3.7 Symmetric Circuits and Fixed-Point Logics

Anuj Dawar (University of Cambridge, GB)

License Creative Commons BY 3.0 Unported license
© Anuj Dawar
Joint work of Anderson, Matthew; Dawar, Anuj

We study queries on graphs (and other relational structures) defined by families of Boolean circuits that are invariant under permutations of the vertices. In particular, we study circuits that are symmetric, that is, circuits whose invariance is explicitly witnessed by automorphisms of the circuit induced by the permutation of their inputs. We show a close connection between queries defined on structures by uniform families of symmetric circuits and definability in fixed-point logics.

3.8 Nominal duality theory for new, forall, and lambda.

Jamie Gabbay (Heriot-Watt University Edinburgh, GB)

License Creative Commons BY 3.0 Unported license
© Jamie Gabbay
Joint work of Gabbay, Jamie; Gabbay, Michael; Petrisan, Daniela; Litak, Tadeusz
Main reference Murdoch J. Gabbay, "Semantics out of context: nominal absolute denotations for first-order logic and computation," arXiv:1305.6291v2 [cs.LO], 2013.
URL http://arxiv.org/abs/1305.6291v2

Nominal algebra lets us axiomatise substitution and quantifiers, and thus the new-quantifier, first-order logic, and the lambda-calculus. Nominal lattice theory lets us characterise binders

as greatest and least upper bounds subject to freshness conditions; this is possible for "forall" and "exists" and also for "lambda".

From this follow soundness, completeness, representation, and topological duality results for algebraic/lattice-theoretic theories in nominal sets and topological spaces. A great deal of structure is revealed by this, which I will outline.

References

1 Murdoch J. Gabbay. *Semantics out of context: nominal absolute denotations for first-order logic and computation.* http://arxiv.org/abs/1305.6291

2 Murdoch J. Gabbay and Michael J. Gabbay. *Representation and duality of the untyped lambda-calculus in nominal lattice and topological semantics, with a proof of topological completeness.* http://arxiv.org/abs/1305.5968

3 Murdoch J. Gabbay, Tadeusz Litak, Daniela Petrisan. *Stone Duality for Nominal Boolean Algebras with NEW.* http://dx.doi.org/10.1007/978-3-642-22944-2_14

3.9 Programming in nominal sets: a survey

Bartek Klin (University of Warsaw, PL)

License Creative Commons BY 3.0 Unported license
© Bartek Klin

An important benefit of nominal techniques is a neat syntactic presentation of computations over certain infinite data sets, e.g., α-equivalence classes of λ-terms or state spaces of automata over infinite alphabets. A few programming languages have been designed that aim to offer that presentation to the programmer, hiding the necessary symbolic manipulations in the compiler. I briefly describe and compare:

- FreshML [3] and its later version FreshO'Caml [4]: functional languages aimed for computation of data structures with binding over equality atoms,
- Nλ [1]: an experimental functional language, implemented as an extension of Haskell, for computing orbit-finite data structures over arbitrary atom symmetries, with no direct treatment of binding,
- While programs with atoms [2]: an imperative language with a very weak type system, with a purpose very similar to that of Nλ.

References

1 M. Bojańczyk, L. Braud, B. Klin, S. Lasota. *Towards nominal computation.* Proceedings of POPL 2012.

2 M. Bojańczyk, S. Toruńczyk. *Imperative programming in sets with atoms.* Proceedings of FSTTCS 2013.

3 M. J. Gabbay, A. M. Pitts. *A metalanguage for programming with bound names modulo renaming.* Proceedings of MPC, volume 1837 of LNCS, 2000.

4 M. R. Shinwell. *The fresh approach: functional programming with names and binders.* PhD thesis, University of Cambridge, 2005.

3.10 A machine-independent characterization of timed languages

Sławomir Lasota (University of Warsaw, PL)

License Creative Commons BY 3.0 Unported license
© Sławomir Lasota
Joint work of Lasota, Sławomir; Bojańczyk, Mikołaj
Main reference M. Bojańczyk, S. Lasota, "A Machine-Independent Characterization of Timed Languages," in Proc. of the 39th Int'l Colloquium on Automata, Languages, and Programming (ICALP'12), Part II, LNCS, Vol. 7392, pp. 92–103, Springer, 2012.
URL http://dx.doi.org/10.1007/978-3-642-31585-5_12

We use a variant of sets with atoms (known also as Fraenkel-Mostowski sets, or as generalized nominal sets, cf. [5, 6, 3]) as a framework suitable for stating and proving the following two results on timed automata [2]. As atoms, we suitably choose the structure of reals with the order $<$ and $+1$ predicate. The first result is a machine-independent characterization of languages of deterministic timed automata, in the style of Myhill-Nerode theorem. As a second result we distinguish a subclass of automata with atoms, called by us timed register automata, that extends timed automata and is effectively closed under minimization. The class is obtained naturally by restricting to those sets with atoms that are quantifier-free definable, using the vocabulary of atoms, namely $<$ and $+1$. Timed register automata are also related to timed automata with updates [4]. Thus sets with atoms provide an appropriate framework for minimization of timed automata, cf. [7, 1].

References

1 R. Alur, C. Courcoubetis, N. Halbwachs, D. L. Dill, and H. Wong-Toi. Minimization of timed transition systems. In *CONCUR*, pages 340–354, 1992.
2 R. Alur and D. L. Dill. A theory of timed automata. *Theor. Comput. Sci.*, 126(2):183–235, 1994.
3 M. Bojańczyk, B. Klin, and S. Lasota. Automata with group actions. In *Proc. LICS'11*, pages 355–364, 2011.
4 P. Bouyer, C. Dufourd, E. Fleury, and A. Petit. Updatable timed automata. *Theor. Comput. Sci.*, 321(2-3):291–345, 2004.
5 M. Gabbay. Foundations of nominal techniques: logic and semantics of variables in abstract syntax. *Bulletin of Symbolic Logic*, 17(2):161–229, 2011.
6 M. Gabbay and A. M. Pitts. A new approach to abstract syntax with variable binding. *Formal Asp. Comput.*, 13(3-5):341–363, 2002.
7 M. Yannakakis and D. Lee. An efficient algorithm for minimizing real-time transition systems. *Formal Methods in System Design*, 11(2):113–136, 1997.

3.11 Nominal nets

Ranko Lazic (University of Warwick, GB)

License Creative Commons BY 3.0 Unported license
© Ranko Lazic

I discussed variants of 'nominal nets', i.e. Petri nets in which tokens are labelled by names, focussing on relative expressiveness and computational complexity of coverability and reachability problems. I also mentioned connections with other formalisms, including logics and automata on words and trees over infinite alphabets. The main message was that too many questions are open, and some of them seem difficult.

3.12 Concurrent Domain Theory with Nominal Sets

Steffen Loesch (University of Cambridge, GB)

License Creative Commons BY 3.0 Unported license
© Steffen Loesch
Joint work of Lösch, Steffen; Pitts, Andrew M.; Winskel, Glynn

HOPLA by Nygaard and Winskel is a concurrent metalanguage with a fully abstract domain theory. This talk introduces HOPLA and shows how it can be extended with nominal sets, in order to obtain a more expressive metalanguage.

3.13 Named Graphs and HD Automata for Network-Conscious pi-Calculus

Ugo Montanari (University of Pisa, IT)

License Creative Commons BY 3.0 Unported license
© Ugo Montanari
Joint work of Montanari, Ugo; Sammartino, Matteo
Main reference U. Montanari, M. Sammartino, "Network Conscious Pi-calculus: A Concurrent Semantics," in Proc. of the 28th Conf. on the Mathematical Foundations of Programming Semantics (MFPS'12), ENTCS, Vol. 286, pp. 291–306, Elsevier, 2012.
URL http://dx.doi.org/10.1016/j.entcs.2012.08.019

The semantics of name-passing calculi is often defined employing coalgebraic models over permutation algebras/nominal sets or over presheaf categories. Both these elegant theories lack finiteness properties, hence they are not apt for verification purposes. Coalgebras over named sets, called history-dependent (HD) automata, are better suited for the purpose due to locality of names. The three models are equivalent. Named sets are generalised by the categorical model of families, that is, free coproduct completions, indexed by symmetries, whenever certain conditions are satisfied.

In the talk we survey various notions of HD automata introduced in the last 18 years and we outline the symmetry indexed family construction. Then we observe that traditional process calculi usually abstract away from network details, modeling only communication over shared channels. They, however, seem inadequate to describe new network architectures, such as Software Defined Networks, where programs are allowed to manipulate the infrastructure. Thus we present the Network Conscious pi-calculus (NCPi), a proper extension of the pi-calculus with an explicit notion of network: network links and nodes are both represented as names. Finally, we show that the categorical model of NCPi, equipped with graphs rather than sets of names as resources, actually satisfies the conditions mentioned above, thus admitting an HD automaton construction, suitable for verification.

3.14 Algorithmic games for full ground references

Andrzej Murawski (University of Warwick, GB)

License Creative Commons BY 3.0 Unported license
© Andrzej Murawski
Joint work of Murawski, Andrzej; Tzevelekos, Nikos
Main reference A. S. Murawski, N. Tzevelekos, "Algorithmic Games for Full Ground References," in Proc. of the 39th Int'l Colloquium on Automata, Languages, and Programming (ICALP'12), Part II, LNCS, Vol. 7392, pp. 312–324, Springer, 2012.
URL http://dx.doi.org/10.1007/978-3-642-31585-5_30

We present a full classification of decidable and undecidable cases for contextual equivalence in a finitary ML-like language equipped with full ground storage (both integers and reference names can be stored). The simplest undecidable type is $\mathsf{unit} \to \mathsf{unit} \to \mathsf{unit}$. At the technical level, our results marry game semantics with automata-theoretic techniques developed to handle infinite alphabets. On the automata-theoretic front, we show decidability of the emptiness problem for register pushdown automata extended with fresh-symbol generation.

3.15 On Orbit finiteness and Limits in Nominal Sets

Daniela Petrisan (University of Leicester, GB)

License Creative Commons BY 3.0 Unported license
© Daniela Petrisan
Joint work of Kurz, Alexander; Petrisan, Daniela; Severi, Paula; de Vries, Fer-Jan
Main reference A. Kurz, D. Luan Petrisan, P. Severi, F.-J. de Vries, "Nominal Coalgebraic Data Types with Applications to Lambda Calculus," Logical Methods in Computer Science, 9(4:20); available also as arXiv:1311.1395v2 [cs.LO].
URL http://dx.doi.org/10.2168/LMCS-9(4:20)2013
URL http://arxiv.org/abs/1311.1395v2

In this talk we discuss presentations of limits in nominal sets and its relevance to infinitary data types involving binders. To this end we introduce a notion of bound variable relative to a map on nominal sets and we define a map to be safe when in the fiber above each element there exists an element with a maximal number of bound variables. We also introduce maps with orbit-finite fibers as maps for which the inverse image of each element is a finitely presentable (or orbit-finite) nominal subset. These maps have very nice closure properties, are safe maps, and allow us to represents limits of quotients in Nom as a quotient of a pullback computed in Set. As an application, one can find representatives of infinitary terms up to alpha-equivalence.

3.16 Nominal Sets: Introduction and Survey

Andrew M. Pitts (University of Cambridge, GB)

License Creative Commons BY 3.0 Unported license
© Andrew M. Pitts
Main reference A. M. Pitts, "Nominal Sets: Names and Symmetry in Computer Science," Cambridge Tracts in Theoretical Computer Science, Vol. 57, Cambridge University Press, 2013.
URL http://www.cambridge.org/us/academic/subjects/computer-science/programming-languages-and-applied-logic/nominal-sets-names-and-symmetry-computer-science

Names and constructs that bind names are ubiquitous in formal languages in general and programming languages in particular. Nominal sets provide a mathematical theory of

structures involving names, based on some simple, but subtle ideas going back to the symmetric models of set theory with atoms of Fraenkel and Mostowski. The theory was introduced by Gabbay and Pitts in 1999 and has since been developed and applied to programming language semantics, machine-assisted theorem proving and the design of functional and logical metaprogramming languages. This work uses the so-called equality symmetry; more recently Bojańczyk et al have considered more general symmetries and their application to automata theory.

This tutorial introduced the theory of nominal sets, mainly from a category- theoretic rather than set-theoretic perspective; it surveyed some of its applications, concentrating on the case of the equality symmetry; and it highlighted some potential areas for further research.

3.17 Nominal Automata and Regular Expressions

Tomoyuki Suzuki (Academy of Science – Prague, CZ)

License Creative Commons BY 3.0 Unported license
© Tomoyuki Suzuki
Joint work of Kurz, Alexander; Suzuki, Tomoyuki; Tuosto, Emilio
Main reference A. Kurz, T. Suzuki, E. Tuosto, "Nominal Regular Expressions for Languages over Infinite Alphabets," arXiv:1310.7093v1 [cs.FL], 2013.
URL http://arxiv.org/abs/1310.7093v1

We discuss regular expressions to abstractly model and study properties of resource-aware computations. Inspired by nominal techniques – as those popular in process calculi – we extend classical regular expressions with names (to model computational resources) and suitable operators (for allocation, deallocation, scoping of, and freshness conditions on resources). We discuss classes of such nominal regular expressions, show how such expressions have natural interpretations in terms of languages over infinite alphabets, and give Kleene theorems to characterise their formal languages in terms of nominal automata. In the end, we also discuss our ongoing works.

3.18 Turing machines with atoms

Szymon Toruńczyk (University of Warsaw, PL)

License Creative Commons BY 3.0 Unported license
© Szymon Toruńczyk
Joint work of Toruńczyk, Szymon; Klin, Bartek; Lasota, Sławomir; Bojańczyk, Mikołaj
Main reference M. Bojańczyk, B. Klin, S. Lasota, S. Toruńczyk, "Turing Machines with Atoms," in Proc. of the 28th IEEE/ACM Symp. on Logic in Computer Science (LICS'13), pp. 183–192, IEEE CS; available as pre-print from the author's webpage.
URL http://dx.doi.org/10.1109/LICS.2013.24
URL http://www.mimuw.edu.pl/~bojan/papers/atomturing.pdf

We study Turing machines over sets with atoms, also known as nominal sets. Our main result is that deterministic machines are weaker than nondeterministic ones; in particular, P≠NP in sets with atoms. Our main construction is closely related to the Cai-Fürer-Immerman graphs used in descriptive complexity theory.

3.19 Names, games and automata

Nikos Tzevelekos (Queen Mary University of London, GB)

License Creative Commons BY 3.0 Unported license
© Nikos Tzevelekos
URL http://www.tzevelekos.org/talks/Dagstuhl_Oct13.pdf

Names constitute a pervasive feature in programming languages. They appear in every computational scenario where entities of specific kinds can be created at will and, moreover, in such a manner that newly created entities are always fresh, i.e. distinct from any other created thus far. For example, references, objects and exceptions in languages like ML or Java can be seen as names. The behaviour of languages which feature names is in general very subtle due to issues of privacy, visibility and flow of names, and the ensuing notion of local state.

This talk is about formal reasoning techniques for programs with names which have emerged in the last years. We will focus on a specific such formalism, called game semantics, which models computation as a formal interaction (a game) between a program and its environment. We moreover see how these models can be given algorithmic representations by means of abstract machines operating on infinite alphabets of names.

3.20 Nominal Isabelle, or, How Not to be Intimidated by the Variable Convention

Christian Urban (King's College London, GB)

License Creative Commons BY 3.0 Unported license
© Christian Urban
Main reference C. Urban, "Nominal Techniques in Isabelle/HOL," J. of Automatic Reasoning, 40(4):327–356, 2008.
URL http://dx.doi.org/10.1007/s10817-008-9097-2

If researchers in programming languages want to formalise and check their work in a theorem prover, they need to deal with binders, renaming of bound variables, capture-avoiding substitution, etc. This is very often a major problem in formal proofs. In informal proofs one often assumes a variable convention and thus side-steps all these problems. In the talk I will show how strong induction principles can be derived that have the variable convention already built-in. However, I will also show that this convention is in general an unsound reasoning principle and requires restrictions in order to be safe. The aim of this work is to provide all proving technology necessary for reasoning conveniently about programming languages. This work has previously been reported in

- C. Urban, Nominal Techniques in Isabelle/HOL. In Journal of Automatic Reasoning, 2008, Vol. 40(4), 327–356.
- C. Urban and C. Kaliszyk, General Bindings and Alpha-Equivalence in Nominal Isabelle. Journal of Logical Methods in Computer Science, Volume 8 (2:14), 2012

Participants

- Andreas R. Blass
University of Michigan, US
- Mikołaj Bojańczyk
University of Warsaw, PL
- James Cheney
University of Edinburgh, GB
- Vincenzo Ciancia
CNR – Pisa, IT
- Thomas Colcombet
CNRS / Univ. Paris-Diderot, FR
- Roy L. Crole
University of Leicester, GB
- Anuj Dawar
University of Cambridge, GB
- Jamie Gabbay
Heriot-Watt University
Edinburgh, GB
- Fabio Gadducci
University of Pisa, IT
- Tomasz Gogacz
University of Wroclaw, PL
- Bartek Klin
University of Warsaw, PL
- Alexander Kurz
University of Leicester, GB
- Sławomir Lasota
University of Warsaw, PL
- Ranko Lazic
University of Warwick, GB
- Steffen Lösch
University of Cambridge, GB
- Justus Matthiesen
University of Cambridge, GB
- Stefan Milius
Univ. Erlangen-Nürnberg, DE
- Ugo Montanari
University of Pisa, IT
- Andrzej Murawski
University of Warwick, GB
- Joanna Ochremiak
University of Warsaw, PL
- Daniela Petrisan
University of Leicester, GB
- Andrew M. Pitts
University of Cambridge, GB
- Luc Segoufin
ENS – Cachan, FR
- Alexandra Silva
Radboud Univ. Nijmegen, NL
- Ian Stark
University of Edinburgh, GB
- Tomoyuki Suzuki
Academy of Science – Prague, CZ
- Szymon Toruńczyk
University of Warsaw, PL
- Emilio Tuosto
University of Leicester, GB
- Nikos Tzevelekos
Queen Mary University of
London, GB
- Christian Urban
King's College London, GB