Electronic Notes in Theoretical Informatics And Computer Science

ENTICS
https://entics.episciences.org

Volume 3
Proceedings of
MFPS 2023

Preface

This volume contains the *Proceedings* of the 39h International Conference of Mathematical Foundations of Programming Semantics (MFPS XXXIX), which was held as a hybrid conference with an in-person meeting at Bloomington University in Bloomington, Indian USA and as a virtual event online. The conference took place from June 21 to 23, 2023, and was colocated with CALCO which took place from June 19 to 21. The local organization was chaired by Larry Moss, and involved Devendra, Osanda Illeperuma, Pavel Kovalev, Caleb Schultz Kisby, Darshal Shetty, Zixiu Su, and Cathy Wyss. Many thanks go to the MFPS Organizers (Steering Committee), in particular Michael Mislove, for their guidance and advice.

MFPS conferences are dedicated to the areas of mathematics, logic, and computer science that are related to models of computation in general, and to semantics of programming languages in particular. This is a forum where researchers in mathematics and computer science can meet and exchange ideas.

Topics include, but are not limited to, the following: bio-computation; concurrent qualitative and quantitative distributed systems; process calculi; probabilistic systems; constructive mathematics; domain theory and categorical models; formal languages; formal methods; game semantics; lambda calculus; programming-language theory; quantum computation; security; topological models; logic; type systems; type theory.

MFPS 2023 continued the tradition of having an exciting group of invited speakers, both plenary and in special sessions. We enjoyed listening to Robert Harper and Assia Mahboubi, who gave a plenary talk joint with CALCO, and to Azalea Raad and Alex Simpson, who gave MFPS plenary talks. MFPS and CALCO held a joint special session on Machine-checked mathematics, organized by Assia Mahboubi. Additionally, MFPS held special sessions on Semantics and Compilers, organized by Amal Ahmed, and on Categories of bidirectional processes, organized by Jules Hedges.

The conference received submissions from Australia, Bangladesh, Denmark, France, Greece, Italy Netherlands, Norway, Portugal, Sweden, United Kingdom, and the United States. We are grateful to the program committee members and external reviewers for their work. The accepted and invited papers are published in this volume (in alphabetical order of the names of first authors). We hope you enjoy reading the papers in this volume as much as we enjoyed listening to the talks.

Marie Kerjean and Paul B. Levy
November 2023

Published November 15, 2023 Available Online at
https://doi.org/10.46298.entics.proceedings.mfps39

Program Committee

Henning Basold	LIACS, Leiden University
Andrej Bauer	University of Ljubljana
Robin Cockett	University of Calgary
Liron Cohen	Ben-Gurion University
Adrian Francalanza	University of Malta
Francesco Gavazzo	University of Pisa
Sergey Goncharov	FAU Erlangen-Nürnberg
Tom Hirschowitz	CNRS, Univ. Savoie Mont Blanc
Justin Hsu	Cornell University
Guilhem Jaber	Universié de Nantes
Achim Jung	University of Birmingham
Marie Kerjean (co-chair)	CNRS, Univ. Sorbonne Paris Nord
Vasileios Koutavas	Trinity College Dublin
Neelakantan Krishnaswami	University of Cambridge
Paul Blain Levy (co-chair)	University of Birmingham
Maria Emilia Maietti	University of Padua
Samuel Mimram	École Polytechnique
Alexandre Miquel	University of the Republic (Montevideo)
Michael Mislove	Tulane University
Koko Muroya	RIMS, Kyoto University
Rasmus Ejlers Møgelberg	IT University of Copenhagen
Max New	University of Michigan
Paige North	Utrecht University
Filip Sieczkowski	Heriot-Watt University

Organizers (Steering Committee)

L Andrej Bauer, *University of Ljubljana*

Lars Birkedal, *Aarhus University*

Steve Brookes, *Carnegie Mellon University*

Achim Jung, *University of Birmingham*

Catherine Meadows, *Naval Research Laboratory*

Christine Tasson, *Sorbonne University*

Michael Mislove, *Tulane University*

Joël Ouaknine, *MPI for Software Systems*

Prakash Panangaden, *McGill University*

Alexandra Silva, *Cornell University*

Sam Staton, *University of Oxford*

Justin Hsu, *Cornell*

External Reviewers

Davide Barbarossa, Rafaël Bocquet, Amar Hadzihasanovic, Alex Kavvos, Dylan McDermott, Simon Mirwasser, Eugenio Moggi, Lê Thành Düng Nguyen, Tanjona Ralaivaosaona, Eike Ritter, Gerard Tabone, Alejandro Villoria, Jasmine Xuereb.

Invited plenary talks

Robert Harper (Carnegie Mellon University) *A cost-aware logical framework.* (Joint with CALCO)

The computational view of intuitionistic dependent type theory is as an intrinsic logic of (functional) programs in which types are viewed as specifications of their behavior. Equational reasoning is particularly relevant in the functional case, where correctness can be formulated as equality between two implementations of the same behavior. Besides behavior, it is also important to specify and verify the cost of programs, measured in terms of their resource usage, with respect to both sequential and parallel evaluation. Although program cost can—and has been—verified in type theory using an extrinsic formulation of programs as data objects, what we seek here is, instead, an intrinsic account within type theory itself.

In this talk we discuss Calf, the Cost-Aware Logical Framework, which is an extension of dependent call-by-push-value type theory that provides an intrinsic account of both parallel and sequential resource usage for a variety of problem-specific measures of cost. Thus, for example, it is possible to prove that insertion sort and merge sort are equal as regards behavior, but differ in terms of the number of comparisons required to achieve the same results. But how can equal functions have different cost? To provide an intrinsic account of both intensional and extensional properties of programs, we make use of Sterling's notion of Synthetic Tait Computability, a generalization of Tait's method originally developed for the study of higher type theory.

In STC the concept of a "phase" plays a central role: originally as the distinction between the syntactic and semantic aspects of a computability structure, but more recently applied to the formulation of type theories for program modules and for information flow properties of programs. In Calf we distinguish two phases, the intensional and extensional, which differ as regards the significance of cost accounting—extensionally it is neglected, intensionally it is of paramount importance. Thus, in the extensional phase insertion sort and merge sort are equal, but in the intensional phase they are distinct, and indeed one is proved to have optimal behavior as regards comparisons, and the other not. Importantly, both phases are needed in a cost verification—the proof of the complexity of an algorithm usually relies on aspects of its correctness.

We will provide an overview of Calf itself, and of its application in the verification of the cost and behavior of a variety of programs. So far we have been able to verify cost bounds on Euclid's Algorithm, amortized bounds on batched queues, parallel cost bounds on a joinable form of red-black trees, and the equivalence and cost of the aforementioned sorting methods. In a companion paper at this meeting Grodin and I develop an account of amortization that relates the standard inductive view of instruction sequences with the coinductive view of data structures characterized by the same operations. In ongoing work we are extending the base of verified deterministic algorithms to those taught in the undergraduate parallel algorithms course at Carnegie Mellon, and are extending Calf itself to account for probabilistic methods, which are also used in that course.

Joint work with Harrison Grodin (Carnegie Mellon), Yue Niu (Carnegie Mellon), and Jon Sterling (Cambridge).

Assia Mahboubi (Inria) *Machine-checked computational mathematics.* (Joint with CALCO)

Geared with increasingly fast computer algebra libraries and scientific computing software, computers have become amazing instruments for mathematical guesswork. In fact, computer calculations are even sometimes used to substantiate actual reasoning steps in proofs, later published in major venues of the mathematical literature. Yet surprisingly, little of the now standard techniques available today for verifying critical software (e.g., cryptographic components, airborne commands, etc.) have been applied to the programs used to produce mathematics. In this talk, we propose to discuss this state of affairs.

Azalea Raad (Imperial College London) *Incorrectness Logic for Scalable Bug Detection.*

Incorrectness Logic (IL) has recently been advanced as a logical under-approximate theory for proving the presence of bugs—dual to Hoare Logic, which is an over-approximate theory for proving the absence of bugs. To facilitate scalable bug detection, later we developed incorrectness separation logic (ISL) by marrying the under-approximate reasoning of IL with the local reasoning of separation logic and its frame rule. This locality leads to techniques that are compositional both in code (concentrating on a program component) and in the resources accessed (spatial locality), without tracking the entire global state or the global program within which a component sits. This enables reasoning to scale to large teams and codebases: reasoning can be done even when a global program is not present. We then developed Pulse-X, an automatic program analysis for catching memory safety errors, underpinned by ISL. Using PulseX, deployed at Meta, we found a number of real bugs in codebases such as OpenSSL, which were subsequently confirmed and fixed. We have compared the performance of Pulse-X against the state-of-the-art tool Infer on a number of large programs; our comparison shows that Pulse-X is comparable with Infer in terms of performance, and in certain cases its fix-rate surpasses that of Infer.

Alex Simpson (University of Ljubljana) *Probabilistic Programming with Coinductive Data.*

Many stochastic processes can be represented as probabilistically generated coinductive data, produced by programs that utilise a principle of probabilistic corecursion. Mathematically this principle is justified because the categories of deterministic and probabilistic maps enjoy an unusual coincidence of final coalgebras. Based on these observations, I shall present a simple programming language combining recursion, corecursion and probability with a lazy operational semantics. I shall argue that the correctness proof for the operational semantics is most easily carried out in a set theory in which all sets are measurable.

Joint work with Danel Ahman and Léo Soudant.

Special session on machine-checked mathematics (joint with CALCO)

For this session, the following speakers were invited.

Yannick Forster (Inria) *Synthetic Computability in Constructive Type Theory.*

Mathematical practice in most areas of mathematics is based on the assumption that proofs could be made fully formal in a chosen foundation in principle. This assumption is backed by decades of formalising various areas of mathematics in various proof assistants and various foundations. An area that has been largely neglected for computer-assisted and machine-checked proofs is computability theory. This is due to the fact that making computability theory (and its sibling complexity theory) formal is several orders of magnitude more involved than formalising other areas of mathematics, due to the – citing Emil Post – "forbidding, diverse and alien formalisms in which this [...] work of Gödel, Church, Turing, Kleene, Rosser [...] is embodied.". For instance, there have been various approaches of formalising Turing machines, all to the ultimate dissatisfaction of the respective authors, and none going further than constructing a universal machine and proving the halting problem undecidable. Professional computability theorist and teachers of computability theory thus rely on the informal Church Turing thesis to carry out their work and only argue the computability of described algorithms informally.

A way out was proposed in the 1980s by Fred Richman and developed during the last decade by Andrej Bauer: Synthetic computability theory, where one assumes axioms in a constructive foundation which essentially identify all (constructively definable) functions with computable functions. A drawback of the approach is that assuming such an axiom on top of the axiom of countable choice - which is routinely assumed in this branch of constructive mathematics and computable analysis - is that the law of excluded middle, i.e. classical logic, becomes invalid. Computability theory is however dedicatedly classical: Almost all basic results are presented by appeal to classical axioms and even the full axiom of choice.

We observe that a slight foundational shift rectifies the situation: By basing synthetic computability theory in the Calculus of Inductive Constructions, the type theory underlying amongst others the Coq proof assistant, where countable choice is independent and thus not provable, axioms for synthetic computability

are compatible with the law of excluded middle.

I will give an overview over a line of research investigating a synthetic approach to computability theory in constructive type theory, discussing suitable axioms, a Coq library of undecidability proofs, results in the theory of reducibility degrees, a synthetic definition of Kolmogorov complexity, constructive reverse analysis of theorems, and synthetic oracle computability.

Parts of results are in collaboration with Dominik Kirst, Gert Smolka, Felix Jahn, Fabian Kunze, Nils Lauermann, Niklas Mück, and the contributors of the Coq Library of Undecidability Proofs.

Andrei Popescu (University of Sheffield) *On the exquisite pleasure of doing coinduction and corecursion in Isabelle.*

Coinduction is a powerful technique for defining and reasoning about infinite objects and infinite behaviors. I will show how the Isabelle/HOL proof assistant is a natural home for coinduction. I will illustrate Isabelle's compositional infrastructure for codatatypes, coinductive predicates, and corecursive function definitions. As a working example, I will give a formal proof of the equivalence between Knaster-Tarski greatest fixpoints and provability by infinite proof trees – which is a foundational result for coinduction in proof assistants. I will also show how Isabelle facilitates the sound mixture of recursion and corecursion.1n and ample first order differential relations, which is a general technique to find solutions for construction problems.

Special session on semantics and compilers

For this session, the following speakers were invited.

Amal Ahmed (Northeastern University) *Semantic Intermediate Representations for Safe Language Interoperability.*

Designers of typed programming languages commonly prove meta-theoretic properties such as type soundness for at least a core of their language. But any practical language implementation must provide some way of interoperating with code written in other languages – usually via a foreign-function interface (FFI) – which opens the door to new, potentially unsafe behaviors that aren't accounted for in the original type soundness proofs. Despite the prevalence of interoperability in practical software, principled foundations for the end-to-end design, implementation, and verification of interoperability mechanisms have been largely neglected.

In this talk, I'll advocate a proof technique for ensuring soundness of practical languages, which implement interoperability using glue code that mediates interaction between languages after compilation to a common lower-level intermediate representation (IR). The technique involves building a semantic intermediate representation: a semantic model of source-language types as relations on terms of the lower-level IR. Semantic IRs can be used to guide the design and implementation of sound FFIs and to verify that the IR glue code used to implement conversions ensures type soundness. I'll conclude with some avenues of future work using semantic IRs as a basis for principled design of language interoperability.

Jérémie Koenig (Yale University) *Three dimensions of compositionality in CompCert semantics.*

Compositional models have been designed for a broad range of language features, but using compositional semantics to formulate and establish compiler correctness theorems remains challenging. In this context, horizontal composition of program components interacts with vertical composition of compiler phases, as both must be compatible with the refinement relation used to formulate compositional semantics preservation.

In this talk I will outline how this challenge plays out in the context of the certified compiler CompCert. By borrowing concepts and tools from higher category theory, we can give a systematic account of the way composition principles interact with each other, and give string diagram algebras for various models. I will then explain how the CompCertO model can accomodate spatial as well as horizontal and vertical compositionality, extending its capabilities as a verification framework. This is work in progress with Yu Zhang, Zhong Shao and Yuting Wang.

Max New (University of Michigan) *Compiling with Call-by-push-value.*

Paul Levy demonstrated that call-by-push-value (CBPV) provides a natural way to decompose both the operational and denotational semantics of call-by-value and call-by-name into more fundamental primitive notions of value and computation type. In this talk we discuss how this same call-by-push-value structure naturally arises in stack-based implementation techniques for compilation of functional programming languages. We show how several standard compilation passes (continuation-passing style, closure conversion) amount to the correctness of Church-encoding in impredicative polymorphic CBPV. Further we relate CBPV to existing calculi for compilation: ANF, Monadic form, SSA, CPS and speculate on potential advantages and disadvantages of incorporating CBPV structure explicitly into compiler intermediate languages.

Special session on categories of bidirectional processes

For this session, the following speakers were invited.

Jules Hedges (University of Strathclyde) *Introduction to categorical cybernetics.*

The term "categorical cybernetics" refers to two things: a general theory of categories of optics and related constructions, and a lot of specific examples. These tend to arise in topics that historically were called "cybernetics" (before that term drifted beyond recognition) - AI, control theory, game theory, systems theory. Specific examples of "things that compose optically" are derivatives (well known as backprop), exact and approximate Bayesian inverses, payoffs in game theory, values in control theory and reinforcement learning, updates of data (the original setting for lenses), and updates of state machines. I'll do a gentle tour through these, emphasising their shared structure and the field we're developing to study it.

Valeria de Paiva (Topos Institute) *Lenses and Dialectica constructions*

Dialectica categories were first introduced in my doctoral work to provide an internal characterization of Godel's Dialectica Interpretation. Many years later it was realized that Dialectica category morphisms have the same types as lenses, for a specific definition of lenses. In this short talk, we explain how the motivation for lenses, a special case of bidirectional transformations, in terms of connections of computing systems might suggest uses for Dialectica versions of lenses.

Mario Román (Tallinn University of Technology) *Optics: the Algebra of Monoidal Decomposition.*

Optics have gained popularity in recent years. They appear in differential categories, compositional game theory, bidirectional accessors, process dependency and causality. At the same time, they remain a bit mysterious: why are they so effective? We claim that, apart from their usual monoidal tensor ($\otimes$), optics have a second, hidden, monoidal tensor ($\triangleleft$). This tensor structure is the crucial ingredient for some applications of optics, but it hides in plain sight because it is non-representable: the tensor of two objects is not again an object, but only a presheaf. Once we can clearly see this second tensor, optics become a non-representable duoidal algebra that is the universal such one over a monoidal category in a precise sense.

This talk is based on *The Produoidal Algebra of Process Decomposition*, which is recent joint work with Matt Earnshaw and James Hefford.