UUCS > IFIP21 Web > WebLeftBar > Lesbos (24 Oct 2017, BernhardMoeller)

Meeting Information

The 76th meeting will be organised by Lambert Meertens.

Location: Pasiphae hotel, near Skala Kallonís, Lesbos, Greece.

For practical information, go to Lesbos (practical).

Dates

Monday 2017-10-16 to Friday 2017-10-20.

Participants

Name	Function
Roland Backhouse	
Andrew Black	
Nils Anders Danielsson	
Jeremy Gibbons	
Fritz Henglein	
Peter Hoefner	
Cezar Ionescu	
Patrik Jansson	
Johan Jeuring	
Oleg Kiselyov	
Sam Lindley	
Annie Liu	
Clare Martin	
Lambert Meertens	organizer
Bernhard Möller	
Carroll Morgan	chair
Shin-Cheng Mu	
Alberto Pardo	
Matija Pretnar	
Florian Rabe	
Tom Schrijvers	secretary
Doaitse Swierstra	
Wouter Swierstra	
Nicolas Wu	

See also the table of participants on the “practical” page for detailed practical information

Organizational and administrative matters

Friday morning, October 20, 2017.

Membership

These private matters are not recorded in the public version of the minutes.

Formal resolution

The members and observers of WG2.1 present at the 76th meeting, on Lesbos, express their gratitude to Lambert Meertens for adroitly having avoided the group's apparent propensity for natural phenomena on the Grand Scale, both good and bad: we have in the past had volcanic eruptions, solar eclipses --- and, for this meeting, an earthquake. In spite of that last, Lambert organised our meeting in idyllic surroundings where, the cats and dogs notwithstanding, it did not rain a drop; nor did the Minotaur come, unwished for, to visit his mother.

In fact our only problem seemed to be being hugged to death by the tour guide. And of course we learned the true nature of Proof just before, rather than in, the eating of the Pudding.

Next meetings:

- M77 (Jul 2018): Germany.
- M78 (Mar 2019) Taiwan.
- M79 (Dec 2019) Eastern Europe?

Technical presentations in scheduled order

Andrew Black, *The Left-hand of Equals* (Monday, October 16, 2017, 10:45)

Object-oriented languages have long felt the need to apologize for their threatment of equality, both because it seems inevitable that they must have object idenity as well as value equality, and because in an Object-based universe, equality is not symmetric, and thus is not natually an equaivalence relation. This talk is based on an essay presented at Onward 2016, and takes a descriptive historial journey through equality. We end up at a place we did not expect: a single "left-handed" equality relying on trust and grace.

Fritz Henglein, *Smart Contracts are Neither* (Monday, October 16, 2017, 12:00):

Smart contracts are the programmatic power tool behind the surging blockchain/distributed ledger technology and the cryptocurrencies, initial coin offerings, decentralized autonomous organizations, financial contract management, etc., they facilitate. In this nontechnical talk I will give a one-slide introduction to blockchain technology and smart contracts and a (multi-slide) diagrammatic, decidedly noncategorical argument why smart contracts are neither smart nor contracts.

Jeremy Gibbons, *Choice and Other Effects* (Monday, October 16, 2017, 14:25) (slides)

I have some half-baked ideas about combining choice with state, probabilistic choice with nondeterministic, and angelic choice with demonic, about which I would greatly appreciate the group's thoughts---especially the "effects" people, the "Kleene algebraicists", and the "refinement calculators". I'd be happy to harangue people over a beer rather than in a talk if that is deemed preferable.

Patrik Jansson, *VisPar: Visualising Dataflow Graphs from the Par Monad* (Monday, October 16, 2017, 16:03): (VisPar slides, VisPar paper, VisPar source code)

We present a work in progress tool for visualising computations in the Par monad in Haskell. Our contribution is not a revolutionary new idea but rather a modest addition to the set of tools available for making sense of Par-monad computations. We hope to show that it can be useful as a teaching tool by providing visualisations of a few examples from a course on parallel functional programming. I hope that the 2.1 audience can provide some insight into calculi for transforming Par monad computations. A short paper abour VisPar was presented at FHPC 2017 and the code is availabe on github.

Jeremy Gibbons, *Relational Algebra by Way of Adjunctions* (Monday, October 16, 2017, 16:52) (slides)

Bulk types such as sets, bags, and lists are monads, and therefore support a notation for database queries based on comprehensions. This fact is the basis of much work on database query languages. The monadic structure easily explains most of standard relational algebra, such as selections and projections; in particular, the equivalences that arise from the adjunctions underlying these monads provide an elegant mathematical foundation justifying most database query transformations. Most, but not all: monads do not immediately offer an explanation of relational join or grouping, nor of query transformations involving those crucial aspects of relational algebra. But generalization to *graded monads* does complete the explanation; moreover, graded monads also have underlying adjunctions, which again give rise to a body of equivalences that justify the associated query transformations. This is joint work with Ralf Hinze, Nick Wu, and Fritz Henglein; it's a continuation of the work I presented on *Comprehending Monadic Queries* at Meeting 73.

Alberto Pardo, *Extensible Records in Idris* (Tuesday, October 17, 2017, 9:00) (slides)

Extensible records are records structures that can be dynamically extended with new fields. In some languages, extensible records are supported as a primitive, in others they are implemented as a user library, each alternative with its benefits and drawbacks. In this talk we present a library to strongly-typed extensible records in Idris, a functional programming language with dependent types. Like HList, a Haskell library for extensible records, we use heterogeneous lists to represent our records, but now exploiting the power of dependent types.

This is joint work with Gonzalo Waszczuk and Marcos Viera.

Cezar Ionescu, *Folding DAGs* (Tuesday, October 17, 2017, 10:01)

Abstract: Hamana and Fiore presented in [1] the proof of existence of initial algebras for functors associated with a large class of inductive data definitions. I will show how to write the corresponding fold operator for a number of examples, culminating with an inductive type for directed acyclic graphs (DAGs), and an interesting application of fold-fusion in which the DAG represents a function on intervals of floating-point numbers.

[1] Hamana and Fiore, 2011. "A Foundation for GADTs and Inductive Families"

Sam Lindley, *Encoding Products in Simply Typed Lambda Calculus* (Tuesday, October 17, 2017, 11:10)

By analysing the structure of normal forms I shall give a simple proof that there exists no uniform type-respecting local translation of product types into plain simply typed lambda calculus with only base and function types. The same idea extends to coproduct types.

Tom Schrijvers, *Coherent Nested Composition with Disjoint Intersection Types* (Tuesday, October 17, 2017, 11:25)

Calculi with disjoint intersection types support an introduc- tion form for intersections called the merge operator, while retaining a coherent semantics. Disjoint intersections types have a great potential to serve as a foundation to powerful, flexible and yet type-safe and easy to reason OO languages. This paper shows how to significantly extend the expressive power of disjoint intersection types by adding support for nested subtyping and composition, which enables simple forms of fam- ily polymorphism to be expressed in the calculus. The extension with nested subtyping and composition is challenging, for two different rea- sons. Firstly, the subtyping relation that supports such features is non- trivial, especially when it comes to obtaining an algorithmic version. Secondly, the syntactic method used in previous calculi with disjoint in- tersection types to prove coherence is too inflexible, making it hard to extend those calculi with new features (such as nested subtyping). We show how to address the first problem by adapting and extending Baren- dregt, Coppo and Dezani (BCD) subtyping rules for intersections with records and coercions. A sound and complete algorithmic system is ob- tained by using an approach inspired by Pierce's work. To address the second problem we replace the syntactic method to prove coherence, by a semantic proof method based on logical relations. Our work has been fully formalized in Coq, and we have an implementation of our calculus.

This is joint work with Xuan "Jeremy" Bi and Bruno C. d. S. Oliveira.

Nils Anders Danielsson, *Up-to Techniques Using Sized Types* (Tuesday, October 17, 2017, 14:02)

Slides, accompanying paper.

Wouter Swierstra, *Embedding the Refinement Calculus in Coq* (Tuesday, October 17, 2017, 14:46)

The refinement calculus and type theory are both frameworks that support the specification and verification of programs. This paper presents an embedding of the refinement calculus in the interactive theorem prover Coq, clarifying the relation between the two. As a result, refinement calculations can be performed in Coq, enabling the interactive calculation of formally verified programs from their specification.

Johan Jeuring, *Feedback and Hints in Ask-Elle for creative students* (Tuesday, October 17, 2017, 16:02):

In this talk I want to discuss an experiment we performed with students using Ask-Elle, our functional programming tutor. I analyse how many student submissions we can diagnose, and discuss possibilities to increase both the number of programs we can diagnose, and the number of situations in which we can give hints. For this purpose we need program transformation and synthesis techniques, and automatic theorem proving.

Peter Hoefner, *Backwards and Forwards Reasoning in Separation Logic* (Tuesday, October 17, 2017, 16:51):

In this talk I present a framework that allows backward reasoning using weakest preconditions, and forward reasoning using strongest postconditions for separation logic. While the former is derived directly from the standard operators of separation logic, the latter uses a new operator, which I introduce. We implemented the framework in the interactive theorem prover Isabelle/HOL, and enable automation with several interactive proof tactics. I will end the presentation with a couple of open problems we are working on right now.

Matija Pretnar, *Explicit Effect Subytping* (Wednesday, October 18, 2017, 9:02):

To inform the programmer about the effects of a given program, a bottom up inference algorithm that Eff provides is sufficient. However, for an optimizing compiler, we need access to effects of all subterms. It turned out that adapting the current algorithm is overly error-prone as the core language is implicitly typed and subtyping-based.

In hope of remedying the situation, we present an explicitly-typed polymorphic core calculus, which reifies appeals to subtyping in explicit casts with coercions that witness the subtyping proof. We provide a constraint-based inference algorithm that turns an implicitly-typed term into our calculus, and effect erasure, which allows us to targets an ML-like language with or without support for algebraic effects.

The talk will include a brief introduction to algebraic effect handlers.

This is joint work with Amr Hany Saleh, George Karachalias and Tom Schrijvers.

Bernhard Möller, *Formalisation of Geographic Reachability in Space and Time* (Wednesday, October 18, 2017, 10:08) (slidesI, slidesII):

(with David Pätzel)

Time Geography is a framework for describing reachable points in a (static) spatio-temporal environment. While it was originally devised to facilitate reasoning about an individual's or a population's living conditions, it was later adapted to many other applications. However, a formal treatment was missing.

In Part I of the talk we report on the use of the functional programming language Haskell to provide both a formalisation and a corresponding implementation of an existing extension to Time Geography. Extending earlier approaches, it also deals with dynamic environments that contain obstacles and destinations. This allows then expressing compulsions such as "we must reach the airport before the plane leaves" and avoidance of boundaries such as "we must not pass through road X because of construction work". The formalisation also shows that the original approach to Time Geography has to be refined, as some of its concepts have to be adapted before they can be modelled formally. For simplicity, the implementation was restricted to deal with one-dimensional space only.

In Part II we show how to model these notions in relational algebra (and, more abstractly, in modal semirings) with box and diamond operators. We provide a more comprehensive extension of the approach sketched at WG 2.1 Meeting #71. Admissible or undesired regions an be described as Boolean combinations of primitive regions such as the set of all points reachable by forward or backward movement from a given region of starting points. This yields a generalisation in several aspects: the algebraic model abstracts from
- the particular reachability relation between space-time points,
- dimensionality of space,
- shape of barriers.
Ongoing work concerns finding such expressions for the admissible/forbidden region spanned by the union of two regions, which were computed via a fixed point iteration for corresponding closure operators in Part I as well as tackling two-dimensional space as well.

Sam Lindley, *Delimited Continuations, Macro Expressiveness and Effect Handlers* (Thursday, October 19, 2017, 9:02):

I shall compare the expressive power of two different programming abstractions for user-defined computational effects: delimited continuations and effect handlers. I will introduce Danvy and Filinski's flavour of delimited continuations, Felleisen's notion of macro expressiveness, and Plotkin and Pretnar's effect handlers. I will then show that, disregarding types, delimited continuations and effect handlers can macro express one another. Finally, I will consider macro translations in the presence of types. Given simple type systems for delimited continuations and effect handlers neither macro translation preserves typeability. Furthermore, no typeability-preserving macro translation exists from effect handlers to delimited continuations. I speculate that adding various forms of polymorphism is sufficient to type the macro translations.

(Based on joint work with Yannick Forster, Ohad Kammar, and Matija Pretnar.)

Oleg Kiselyov, *λ to SKI Semantically* (Thursday, October 19, 2017, 10:59):

TBD

Shin-Cheng Mu, *Reasoning and Derivation of Monadic Programs: A Case Study of Non-determinism and State* (Thursday, October 19, 2017, 11:39):

Equational reasoning is among the most important tools that functional programming provides us. Curiously, relatively little attention has been paid to reasoning about monadic programs. In this pearl we aim to develop theorems and patterns useful for the derivation of monadic programs, focusing on non-determinism and state. We model the aggregation function in Spark as a monadic program and derive conditions under which it is deterministic. We also investigate the intricate interaction between state and non-determinism, and derive two backtracking algorithms, respectively using local and global states, to solve the n-queens and Sudoku puzzles.

Carroll Morgan, *Probabilistic Coupling* (Thursday, October 19, 2017, 14:01):

TBD

Shin-Cheng Mu, *Fun with Polynomials of Polynomials* (Thursday, October 19, 2017, 14:54):

TBD

Florian Rabe, *Uniformal: A Framework for Defining Programming Languages* (Thursday, October 19, 2017, 15:55):

I presented recent progress on using MMT for the declarative specification of the syntax and semantics of programming languages.

The slides (slides.pdf) were interspersed with examples from the encoding of a concrete example (rabe_encodings.mmt).

Nicolas Wu, *Handlers in Scope with Adjunctions* (Thursday, October 19, 2017, 16:39):

TBD

Doaitse Swierstra, *Lazy Implementation of the ST Monad* (Thursday, October 19, 2017, 17:31):

TBD

Oleg Kiselyov, *Semantics of ORDER BY in SQL* (Friday, October 20, 2017, 11:08):

TBD

Schedule (subject to change)

Sessions will start at 9:00 on Monday and finish at noon on Friday.

Monday, Tuesday, and Thursday, the schedule is as follows:

- session – 9:00 to 10:30
- coffee break – 10:30 to 11:00
- session – 11:00 to 12:30
- lunch – 12:30 to 14:00
- session – 14:00 to 15:30
- coffee break – 15:30 to 16:00
- session – 16:00 to 18:00
- dinner – 19:30 to 21:30

Wednesday and Friday we will have sessions only in the morning:

- session – 9:00 to 10:30
- coffee break – 10:30 to 11:00
- session – 11:00 to 12:30
- lunch (Wednesday) – 12:30 to 13:30

The Friday-morning session before the 10:30 coffee break is attended by members only.