Hybrid Semantics for Higher-order Store

Bernhard Reus

University of Sussex

Continuity, Computability, Constructivity – From Logic to Algorithms (Nancy 2017)

Motivation (short)

- Soundness for a logic for programs that can store commands, use dynamic memory allocation (heaps)
- with “local reasoning rules” (w.r.t. heap).

Recursion through Store

"Landin's knot"

[c] : dereference c

eval [f] : run code in f

```
let n = [c] in
if n =0 then skip else ( eval [f]; [c] := n-1; eval [it] )
```

assume this code is stored in cell *it*

recursive call "through the store"

- Not so easy to show soundness of typing or verification rules due to circularity.

Higher-order Store

as "infinite type"

recursive domain equations

$$Heap = Rec(Val)$$

records labelled by natural numbers

$$Val = Integers_{\perp} \oplus Com_{\perp}$$

$$Com = Heap \multimap T_{err}(Heap)$$

strict

error monad

continuous

function space

Specifications

Hoare Calculus

$$\frac{}{\{P[e/x]\}\, x \texttt{:=} e\, \{P\}} \qquad \text{(Assignment)}$$

$$\frac{\{P\}\, S_1\, \{I\} \quad \{I\}\, S_2\, \{Q\}}{\{P\}\, S_1 \texttt{;} S_2\, \{Q\}} \qquad \text{(Composition)}$$

$$\frac{\{B \wedge P\}\, T\, \{Q\} \qquad \{\neg B \wedge P\}\, E\, \{Q\}}{\{P\}\, \texttt{if}\, B\, \texttt{then}\, T\, \texttt{else}\, E\, \{Q\}} \qquad \text{(if-then-else)}$$

$$\frac{\{B \wedge I\}\, S\, \{I\}}{\{I\}\, \texttt{while}\, B\, \texttt{do}\, S\, \texttt{od}\, \{I \wedge \neg B\}} \qquad \text{(while)}$$

$$\frac{P \Rightarrow P' \quad \{P'\}\, S\, \{Q'\} \quad Q' \Rightarrow Q}{\{P\}\, S\, \{Q\}} \qquad \text{(Weakening)}$$

Separation Logic

P. W. O'Hearn, H. Yang, and J. C. Reynolds. Local reasoning about programs that alter data structures. In *Proc. of CSL'01*, pages 1–19, Paris, France, 2001.

Reynolds, J.C.: Separation logic: A logic for shared mutable data structures. LICS. (2002) 55–74

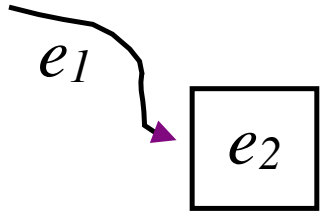
extra assertions

emp

heaplet is empty

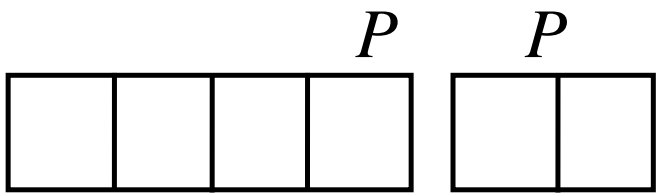
$e_1 \mapsto e_2$

heaplet has exactly one cell e_1 containing e_2



$P_1 \star P_2$

heaplet can be divided into separate heaplets (no aliasing) such that one part fulfils P_1 and the other P_2



Separation Logic

- allows for local reasoning on heaps via *Frame Axiom*
- that says that if a command does not fault with given heap “P”, then neither will it fail, given some extra heap “R”:

$$\{P\}\, e\, \{Q\} \Rightarrow \{P * R\}\, e\, \{Q * R\}$$

means program semantics can't depend on non-allocated addresses

Nested triples

[**Nested Hoare Triples and Frame Rules for Higher-order Stores,** Schwinghammer, Birkedal, Reus, Yang, LMCS 7(3)]

$$x \mapsto \forall a.\ \{a \mapsto _\} \cdot (a)\{emp\}$$

Cell x contains a unary procedure with formal parameter a that deallocates a.

Note:
no need to specify that the rest of the heap remains unchanged.
This will be done by the Frame Axiom.

Spec for Iterator

saying f does not have any effects on heap

$$\{\, c \mapsto _ * f \mapsto \{\mathrm{emp}\}\ _\ \{\mathrm{emp}\} * it \mapsto _ \,\}$$
$$[it] := \text{`}C_{it,f,c}\text{'};\ \texttt{eval}\ [it]$$
$$\{\, c \mapsto 0 * f \mapsto \{\mathrm{emp}\}\ _\ \{\mathrm{emp}\} * R_{it} \,\}\ .$$

recursive heap predicate definitions must be contractive

$$R_{it} \equiv \mu X.\ it \mapsto \{c \mapsto _ * f \mapsto \{emp\}\ _\ \{emp\} * X\}$$
$$\overline{\{c \mapsto 0 * f \mapsto \{emp\}\ _\ \{emp\} * X\}}$$

[it] "maintains itself"

Challenges

- recursively defined specifications need exist (semantically)
- need to express local reasoning w.r.t heaps, i.e. various frame rules must be validated (invariants preserved):

Invariants in Sep.Logic

$P * R$ indicates that P and R govern disjoint regions of the heap.

$P \otimes R$ adds invariant to all nested triples in P

$$(x \mapsto \forall a.\ \{a \mapsto _\} \cdot (a)\{emp\}) \ \otimes \ v \mapsto _ \equiv \ x \mapsto \{a \mapsto _ * v \mapsto _\} \cdot (a)\{v \mapsto _\}$$

$$\frac{\Xi; \Gamma \ \vdash \ P}{\Xi; \Gamma \ \vdash \ P \otimes R}$$

$$P \circ R \stackrel{def}{=} (P \otimes R) * R$$

$$\{P\}\, e\, \{Q\} \otimes R \Leftrightarrow \{P \circ R\}\, e\, \{Q \circ R\}$$

$$(P \otimes R') \otimes R \Leftrightarrow P \otimes (R' \circ R)$$

$$(\kappa x.P) \otimes R \Leftrightarrow \kappa x.(P \otimes R) \qquad (\kappa \in \{\forall, \exists\}, x \notin fv(R))$$

$$(P \oplus Q) \otimes R \Leftrightarrow (P \otimes R) \oplus (Q \otimes R) \qquad (\oplus \in \{\Rightarrow, \wedge, \vee, *\})$$

$$P \otimes R \Leftrightarrow P \qquad (P \text{ is one of } \mathit{true},\ \mathit{false},\ \mathit{emp},\ e = e',\ e \mapsto e')$$

Motivation (longer)

- This gives rise to a Kripke semantics:
- world = store invariants
- assertion = world-indexed predicate
- Kripke monotonicity = invariance under store (world) extensions
- leads to **recursively defined worlds involving predicates!**

Motivation (longer)

recursive domain equation

$$W = W \to Pred(Heap)$$

Semantic type of heap assertions then:

$$W \to Pred(Heap)$$

world W =
invariants to be
maintained

Problem

- How to define (or approximate) those recursive worlds involving predicates, in particular for higher-order store?
- Four ways (at least):
  - use Andy Pitts' "recursive properties of recursive domains technique" (but no recursive unfolding of worlds)
  - construct them using complete ultrametric spaces
  - use operational semantics and "approximative solutions" (step-indexing)
  - or: **mix** ultrametric domains and step-indexing approaches

This is what the talk is about

Ultrametric Spaces

Definition

An **ultrametric space** (X, d) satisfies the strong Δ-inequality

$$d(x, z) \;\leq\; \max\{d(x, y), d(y, z)\}$$

A function $f : X_1 \to X_2$ is **non-expansive** if

$$\forall x, y \in X_1 : \quad d_2(f(x), f(y)) \;\leq\; d_1(x, y)$$

CBUlt: category with

- complete, 1-bounded, non-empty ultrametric spaces and
- non-expansive functions

Constructions

Function spaces $(X_1, d_1) \to (X_2, d_2)$**.**
Non-expansive functions $X_1 \to X_2$ with

$$d(f, g) = \sup\{d_2(f(x), g(x)) \mid x \in X_1\}$$

Scaling $\delta \cdot (X, d')$**.**
Set X with distance function $d(x, x') = \delta \cdot d'(x, x')$

used to enforce contractiveness

Predicate Domain

predicates over *Heap* from earlier

- $\bot = \pi_0(h) \sqsubseteq \ldots \sqsubseteq \pi_k(h) \sqsubseteq \pi_{k+1}(h) \sqsubseteq \ldots \sqsubseteq h$ for all $h \in \mathit{Heap}$, i.e., the π_k's form an increasing chain of approximations of the identity on *Heap*;
- $\pi_j \circ \pi_k = \pi_{\min\{j,k\}}$ for all j, k; in particular, every π_k is idempotent;
- $\bigsqcup_k \pi_k(h) = h$, i.e., every heap is the limit of its approximations.

solve Heap equation in SFP domains

Which Subsets of Heaps shall we use?

uniform: for any $k \in \mathbb{N}$, if $h \in p$ then $\pi_k(h) \in p$.

Uniform Predicates

$UAdm(D) =$ all uniform admissible subsets of (recursive) domain D

closed under sups of ascending chains

$p_{[n]}$ denotes the image of p under π_n.

$$d(p,q) = \begin{cases} 2^{-\max\{i \in \omega \mid p_{[i]} = q_{[i]}\}} & \text{if } p \neq q \\ 0 & \text{otherwise} \end{cases}$$

If D is a (recursive) domain with π_i finite projections as on previous slide then:

Lemma $(UAdm(D), d)$ is a 1-bounded complete ultrametric space.

Recursive Definitions

- ... exist for locally contractive functors on complete ultrametric spaces [America, Rutten, based on Banach, 1989]
- one can always make the functor locally contractive by scaling with a shrinking factor < 1.

The functor $\hat{F}(X, Y) = \frac{1}{2} \cdot F(X, Y)$ is **locally contractive**:

$$d(\hat{F}(f, g), \hat{F}(f', g')) \;\leq\; \tfrac{1}{2} \cdot \max\{d(f, f'), d(g, g')\}$$

Recursive Worlds

Theorem *There exists an ultrametric space W and an isomorphism ι from $\frac{1}{2} \cdot (W \to UAdm)$ to W in $CBUlt$.*

Proof. By an application of America & Rutten's existence theorem for fixed points of locally contractive functors on complete ultrametric spaces

We write Pred for $\frac{1}{2} \cdot (W \to UAdm)$ and $\iota^{-1} : W \cong \mathrm{Pred}$ for the inverse to ι.

- This solves our initial domain equation, however, with an extra "shrinking factor".

top level

Semantic of triples

$$w \models \{p\}\, c\, \{q\},$$

$$\text{if for all } r \in UAdm \text{ and all } h \in Heap:$$

$$h \in p(w) * \iota^{-1}(w)(emp) * r \;\Rightarrow\; c(h) \in \mathrm{Ad}(q(w) * \iota^{-1}(w)(emp) * r)$$

where $\mathrm{Ad}(r)$ denotes the least downward closed and admissible set of heaps containing r.

- result of *c(h)* must not be *error*
- invariant *w* is maintained by *c ...*
- ... but also by code in assertions *p* and *q*
- frame rule baked in via *r*
- downward closure due to uniformity (partial correctness)
- admissibility so that we can use approximations (next slide

nested i.e. as assertions

Semantic of triples

A triple holds *approximately up to level k*, $w \models_k \{p\}\, d\, \{q\}$, if $w \models \{p\}\, \pi_k; d; \pi_k\, \{q\}$.

$$[\![\{P\}\, e\, \{Q\}]\!]_{\eta,\rho}\, w = \mathrm{Ad}\{h \in \mathit{Heap} \mid rnk(h) > 0 \Rightarrow w \models_{rnk(h)-1} \{[\![P]\!]_{\eta,\rho}\}\ [\![e]\!]_\eta\ \{[\![Q]\!]_{\eta,\rho}\}\}$$

Definition 15 (Rank of a heap)**.** If h is a compact element of *Heap*, then the least n for which $\pi_n(h) = h$ is the *rank* of h, abbreviated $rnk(h)$, otherwise the rank is undefined.

A triple holds *approximately up to level k*, $w \models_k \{p\}\, d\, \{q\}$, if $w \models \{p\}\, \pi_k; d; \pi_k\, \{q\}$.

$w \models \{p\}\, c\, \{q\}$ is equivalent to $\forall k \in \omega.\, w \models_k \{p\}\, c\, \{q\}$.

- needs to be non-expansive in semantics of P and Q
- needs to be non-expansive in the world w (to be in Pred)

Deep Frame Rule

Lemma *There exists a non-expansive map* $\circ : W \times W \to W$ *and a map* $\otimes : \mathrm{Pred} \times W \to \mathrm{Pred}$ *that is non-expansive in its first and contractive in its second argument, satisfying* $q \circ r = \iota(\iota^{-1}(q) \otimes r * \iota^{-1}(r))$ *and* $p \otimes r = \lambda w.p(r \circ w)$ *for all* $p \in \mathrm{Pred}$ *and* $q, r \in W$.

- These are recursively defined operations which exist by Banach's fixpoint theorem.

Sample Rules

$$\frac{\{A\}\, e\, \{B\} \Rightarrow \{P\}\, e'\, \{Q\}}{\{\{A\}\, e\, \{B\} \wedge P\}\, e'\, \{Q\}}$$

$$\frac{k \vdash R[k] \Rightarrow \{P * e \mapsto R[_]\}\, k\, \{Q\}}{\vdash \{P * e \mapsto R[_]\}\, \text{`}\texttt{eval}\,[e]\text{'}\, \{Q\}}$$

such an R necessarily to be recursively defined!

- Which rule(s) is(are) sound?

only 2nd rule is sound

Operational Semantics: Step-indexing

Semantic types and operational semantics

- type soundness with small step semantics usually as *subject reduction* (preservation & progress properties) for imperative lambda calculus credited to *[Felleisen & Wright '91]*
- *semantic types*: interpret types as sets of pairs (index,value), downward closed in first component, give semantics of typing judgments and show
  - semantics of typing rules sound
  - semantics of typing judgment implies safety.

Step-indexing's Origin

- Motto: operational semantics with semantic soundness proofs
- no need to formalise syntactic types and type environments
- by [Appel & McAllester, ToPLaS 2002]
- to simplify type correctness proofs for Proof Carrying Code
- and explicitly avoid metric semantics!

Idea of step-indexing

- index is a natural number that states how many operational steps the given value is guaranteed to be of given type (indexed judgements)

$$\Gamma \vdash e :_k \tau$$

- show this for all k
- provides reasoning principle: induction over natural numbers (the index)
- helpful where there is no induction over type structure (mutable references, higher-order store)

Works well...

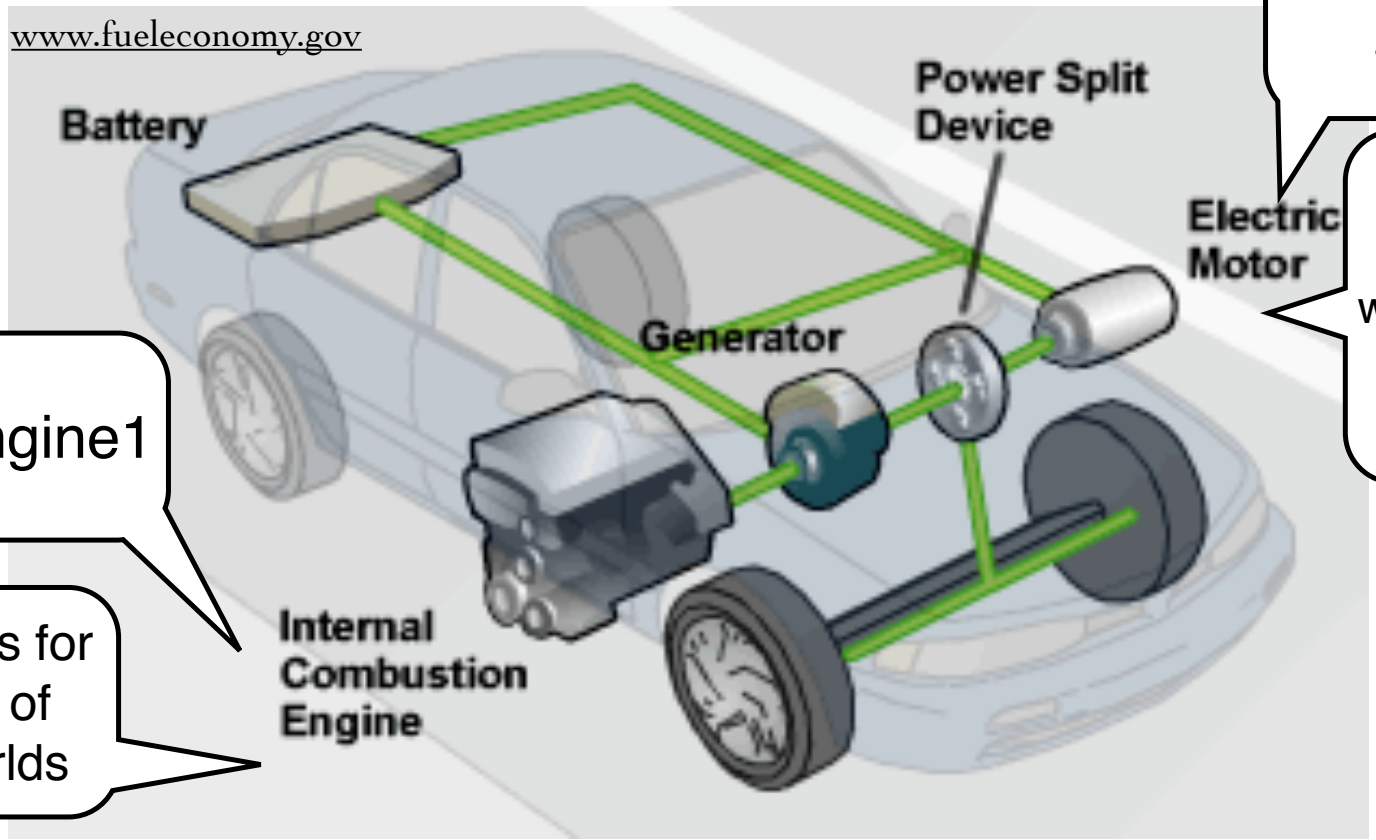
- ... with (higher-order) store (store types or specs can be also indexed)
- ... with recursive types (use iteration level of functional as upper bound for indices)
- ...with binary predicates e.g. for logical relations [Amal Ahmed, Step-Indexed Syntactic Logical Relations, ESOP 2006] or her thesis; Dreyer et al...
- ... with quantified types
- ... with nondeterminism and concurrency

.. but

- for complex nested, recursive types or assertions indexes are cluttered and things become cumbersome.

Hybrid approach

Hybrid approach



Engine 2

operational semantics with semantic rules (step-indexed)

Engine1

CBUIt domains for construction of recursive worlds

step-indexing using ultrametrically constructed worlds

Uniform Step-indexed Predicates:

Uniform ('step-indexed') predicates $p \subseteq \mathbb{N} \times \mathit{Val}$

uniformity $\quad (n, v) \in p \;\wedge\; j \leq n \implies (j, v) \in p$

For such $p, q \in \mathit{UPred}(\mathit{Val})$ Appel & McAllester, TOPLAS, 2001 define

approximation $\quad p_{[n]} = \{\, (j, v) \in p \mid j < n \,\}$

distance $\quad d(p, q) = \inf\{\, 2^{-n} \mid p_{[n]} = q_{[n]} \,\}$

Lemma For any set A, $\mathit{UPred}(A)$ is a 1-bounded complete ultrametric space.

Interpretation of Sep. Logic

Pred is *BI*-algebra

$$emp(w) = \{(n, []) \mid n \in \mathbb{N}\}$$

$$(p * q)(w) = \{(n, h) \mid \exists h_1, h_2.\, h = h_1 \cdot h_2 \wedge (n, h_1) \in p(w) \wedge (n, h_2) \in q(w)$$

$$(p \mathrel{-\!\!*} q)(w) = \{(n, h) \mid \forall m \leq n.\, ((m, h') \in p(w) \wedge h \# h') \Rightarrow (m, h \cdot h') \in q(w)\}$$

Recursive Worlds

$$W \cong \frac{1}{2}(W \to UPred(H))$$

$$Pred = \frac{1}{2}(W \to UPred(H))$$

$$i : Pred \to W$$

UPred where *H= Heap*

iso

- We will write $UPred$ for $UPred(H)$

Interpretation of nested triples

Now say that $w \models_n (p, C, q)$ holds iff: For all $r \in UPred$, all $m < n$ and all heaps h, if $(m, h) \in p(w) * i^{-1}(w)(\mathsf{emp}) * r$, then:

1. $(C, h) \in \mathrm{Safe}_m$.

   (C,h) does not abort in *m* or fewer steps

2. For all $k \leq m$ and all $h' \in H$, if $(C, h) \rightsquigarrow_k (\mathsf{skip}, h')$, then $(m - k, h') \in q(w) * i^{-1}(w)(\mathsf{emp}) * r$.

$$[\![\{P\}e\{Q\}]\!]_\eta w = \begin{cases} \{(n, h) \mid w \models_n ([\![P]\!]_\eta, C, [\![Q]\!]_\eta)\} & \text{if } [\![e]\!]_\eta = \text{`}C\text{`} \\ \emptyset & \text{otherwise} \end{cases}$$

$$
\begin{aligned}
[\![\mathsf{false}]\!]_\eta w &= \emptyset \\
[\![\mathsf{true}]\!]_\eta w &= \mathbb{N} \times H \\
[\![P \wedge Q]\!]_\eta w &= [\![P]\!]_\eta w \cap [\![Q]\!]_\eta w \\
[\![P \vee Q]\!]_\eta w &= [\![P]\!]_\eta w \cup [\![Q]\!]_\eta w \\
[\![P \Rightarrow Q]\!]_\eta w &= \{(n,h) \mid \forall m \leq n.\, (m,h) \in [\![P]\!]_\eta w \Rightarrow (m,h) \in [\![Q]\!]_\eta w \\
[\![\forall x.P]\!]_\eta w &= \textstyle\bigcap_{v \in V} [\![P]\!]_{\eta[x \mapsto v]} w \\
[\![\exists x.P]\!]_\eta w &= \textstyle\bigcup_{v \in V} [\![P]\!]_{\eta[x \mapsto v]} w \\
[\![\mathsf{int}(e)]\!]_\eta w &= \begin{cases} \mathbb{N} \times H & \text{if } [\![e]\!]_\eta = m \text{ for some } m \in \mathbb{Z} \\ \emptyset & \text{otherwise} \end{cases} \\
[\![e_1 = e_2]\!]_\eta w &= \begin{cases} \mathbb{N} \times H & \text{if } [\![e_1]\!]_\eta = [\![e_2]\!]_\eta \\ \emptyset & \text{otherwise} \end{cases} \\
[\![e_1 \leq e_2]\!]_\eta w &= \begin{cases} \mathbb{N} \times H & \text{if } [\![e_1]\!]_\eta = m_1 \text{ and } [\![e_2]\!]_\eta = m_2 \text{ where } m_1 \leq m_2 \\ \emptyset & \text{otherwise} \end{cases} \\
[\![e_1 \mapsto e_2]\!]_\eta w &= \begin{cases} \{(n, [m \mapsto [\![e_2]\!]_\eta]) \mid n \in \mathbb{N}\} & \text{if } [\![e_1]\!]_\eta = m \text{ for some } m \in \mathbb{Z} \\ \emptyset & \text{otherwise} \end{cases} \\
[\![\mathsf{emp}]\!]_\eta w &= \mathbb{N} \times \{[\,]\} \\
[\![P * Q]\!]_\eta w &= [\![P]\!]_\eta w * [\![Q]\!]_\eta w \\
[\![P \mathrel{-\!\!*} Q]\!]_\eta w &= [\![P]\!]_\eta w \mathrel{-\!\!*} [\![Q]\!]_\eta w \\
[\![\{P\}e\{Q\}]\!]_\eta w &= \begin{cases} \{(n,h) \mid w \models_n ([\![P]\!]_\eta, C, [\![Q]\!]_\eta)\,\} & \text{if } [\![e]\!]_\eta = \text{`}C\text{`} \\ \emptyset & \text{otherwise} \end{cases} \\
[\![P \otimes Q]\!]_\eta w &= \Big([\![P]\!]_\eta \otimes i([\![Q]\!]_\eta)\Big)\, w
\end{aligned}
$$

Why this is the “hybrid” model of interest?

- this works as expected whereas
- the ultrametric domain theoretic model [Birkedal, Reus et al. LMCS 2011] needs to use ranks to make Hoare triple specifications non-expansive. This “contaminates” the semantics of the logic (strange implications hold and one does not get a BI algebra).

$$[\![\{P\}\, e\, \{Q\}]\!]_{\eta,\rho}\, w = \mathrm{Ad}\{h \in Heap \mid rnk(h) > 0 \Rightarrow w \models_{rnk(h)-1} \{[\![P]\!]_{\eta,\rho}\}\ [\![e]\!]_{\eta}\ \{[\![Q]\!]_{\eta,\rho}\}\}$$

“Applications”

- soundness of ML type system [Birkedal et al]
- storable locks [Buisse, Birkedal]
- antiframe rule [Schwinghammer et al.]
- Coq formalisations [Varming et al.]
- internal version in the topos of trees [Birkedal, Møgelberg et al.]
- built a prover implementing the rules (proved sound):

`www.sussex.ac.uk/informatics/crowfoot`

Collaborators

Lars Birkedal & Kristian Støvring & Jacob Thamsborg,
ITU Copenhagen
Billiejoe Charlton & Ben Horsfall,
University of Sussex
Jan Schwinghammer,
Universität Saarbrücken
Hongseok Yang,
Oxford
Francois Pottier,
INRIA Rocquencourt

Thank you!

Propaganda: BLC ’17

September 8-9th (PhD day 7th)

Home > British Logic Colloquium 2017 > Homepage

British Logic Colloquium 2017

About the event | Registration | Programme | Travel

Welcome to British Logic Colloquium 2017

The British Logic Colloquium will take place at the University of Sussex on 8-9 September 2017, with the BLC PhD day on 7 September.

This is the annual meeting of the British Logic Colloquium. The scope of the BLC includes mathematical and philosophical logic as well as logic in computer science and applications of logic.

http://www.sussex.ac.uk/blc17/