Local Local Reasoning: A BI-Hyperdoctrine for Full Ground Store

Miriam Polzer **Sergey Goncharov**

FAU Erlangen-Nürnberg

FoSSaCS/ETAPS, ~~Dublin~~ ~~Luxembourg~~ Internet, March 30, 2020

Outline

- Motivation: Logical Conundrum
- Recap of Full Ground Store Monad
- Full Ground Store BI-Hyperdoctrine
- Further Work

Motivation

Intuitionistic Reasoning
about Shared Mutable Data Structure*

John C. Reynolds
Department of Computer Science
Carnegie Mellon University

July 28, 2000

...we extend Hoare's approach
...programs, to deal with pro-

false.

It is easy to prove

$$\{\mathbf{true}\}$$
$$\mathsf{x} := \mathbf{cons}_2(1,2)\ ;$$
$$\{\mathsf{x} \to 1,2\}$$
$$\mathsf{x} := 3$$
$$\{(\exists \mathsf{x}.\ \mathsf{x} \to 1,2) \wedge \mathsf{x} = 3\}.$$

Here the existentially quantified location is disconnected from the data structures accessible to the computation, and can be eliminated by garbage collection. Of course, one can view garbage collection as a program optimization with no effect on observable computations, so that this example is sound. Nevertheless the fact that $\exists \mathsf{x}.\ \mathsf{x} \to 1,2$ is an "unobservable assertion" is worrisome.

Solutions

- There are two principled approaches towards resolving the conundrum: to treat the relevant effect as a **Bug** or as a **Feature**
- We treat it as a **Feature**
- The conundrum indicates a tradeoff between program semantics and logics: should garbage collection be embedded to the model or verified logically?
- We adopt recently developed monad-based, extensional, computationally adequate model for full ground store[1], and argue that it predetermines the approach to the conundrum

[1] Kammar et al., "A monad for full ground reference cells".

What We Did?

- Did we provide a solution to the conundrum? Not entirelly
- We constructed a BI-hyperdoctrine = semantics of a higher order logic of bunched implication complying with the full ground store monad model
- Defining separation logic requires a connection between programs and assertions – a step to be made (non-trivial, because heap separation generates dangling pointers, while program semantics is type safe)
- Possible solutions to the conundrum were proposed previously[2], but not after the monad was introduced

[2] Calcagno, O'Hearn, and Bornat, "Program logic and equivalence in the presence of garbage collection".

Full Ground Store Monad

Key rules (FGCBV-style[3]):

$$\textbf{(put)}\ \frac{\Gamma \vdash_{\mathsf{v}} \ell\colon\ \mathsf{Ref}_S \qquad \Gamma \vdash_{\mathsf{v}} v\colon\ \mathsf{CType}(S)}{\Gamma \vdash_{\mathsf{c}} \ell := v\colon 1} \qquad \textbf{(get)}\ \frac{\Gamma \vdash_{\mathsf{v}} \ell\colon\ \mathsf{Ref}_S}{\Gamma \vdash_{\mathsf{c}} !\,\ell\colon\ \mathsf{CType}(S)}$$

$$\textbf{(new)}\ \frac{\begin{array}{c}\Gamma, \ell_1\colon\ \mathsf{Ref}_{S_1}, \ldots, \ell_n\colon\ \mathsf{Ref}_{S_n} \vdash_{\mathsf{v}} v_1\colon\ \mathsf{CType}(S_1) \\ \vdots \\ \Gamma, \ell_1\colon\ \mathsf{Ref}_{S_1}, \ldots, \ell_n\colon\ \mathsf{Ref}_{S_n} \vdash_{\mathsf{v}} v_n\colon\ \mathsf{CType}(S_n) \\ \Gamma, \ell_1\colon\ \mathsf{Ref}_{S_1}, \ldots, \ell_n\colon\ \mathsf{Ref}_{S_n} \vdash_{\mathsf{c}} p\colon A\end{array}}{\Gamma \vdash_{\mathsf{c}} \mathsf{letref}\ \ell_1 := v_1, \ldots, \ell_n := v_n\ \mathsf{in}\ p\colon A}$$

Example: $\mathsf{letref}\ \ell_1 := (0, \mathsf{inr}\,\star, \mathsf{inl}\,\ell_2);\ \ell_2 := (1, \mathsf{inl}\,\ell_1, \mathsf{inr}\,\star)\ \mathsf{in}\ \mathsf{ret}\,\ell_1$

[3] Levy, Power, and Thielecke, "Modelling Environments in Call-By-Value Programming Languages".

Full Ground Store Monad: Exploded-View

presheaves over words

presheaves over initializations

heaplet functor

$$
\begin{tikzcd}
{[\mathbf{W}, \mathbf{Set}]} \arrow[r, bend left, "\mathfrak{u}^\star"] \arrow[r, phantom, "\perp"] & {[\mathbf{E}, \mathbf{Set}]} \arrow[l, bend left, "\mathfrak{u}_\star"] \arrow[r, bend left, "(-)\times H"] \arrow[r, phantom, "\perp"] & {[\mathbf{E}, \mathbf{Set}]} \arrow[l, bend left, "(-)^H"] \arrow[loop right, "P"]
\end{tikzcd}
$$

hiding monad

geometric morphsim

Slogan: The local (full ground) store monad is just a global store monad transform of the hiding monad sandwiched within a geometric morphism.

$$
\begin{tikzcd}
{[\mathbf{W}, \mathbf{Set}]} \arrow[r, bend left, "\mathfrak{u}^\star"] \arrow[r, phantom, "\perp"] & {[\mathbf{E}, \mathbf{Set}]} \arrow[l, bend left, "\mathfrak{u}_\star"] \arrow[r, bend left, "(-)\times H"] \arrow[r, phantom, "\perp"] & {[\mathbf{E}, \mathbf{Set}]} \arrow[l, bend left, "(-)^H"] \arrow[loop right, "P"]
\end{tikzcd}
$$

- Set of locations $\mathcal{L} \cong \mathbb{N}$
- Set of sorts $\mathcal{S}$
- Objects of **W** are maps $w \colon \mathcal{L} \rightharpoonup_{fin} \mathcal{S}$
- Morphism of **W** are type preserving injections
  $\rho\colon \ \mathrm{dom}\, w \hookrightarrow \mathrm{dom}\, w'$

Heaplets

- Postulate a semantics of sorts over each world range: $\mathcal{S} \to [\mathbf{W}, \mathbf{Set}]$
- A heap would send each $(\ell\colon S) \in w$ to $\mathrm{range}(S)(w)$, and it is not a functor (!)
- Hence heaplets: $\mathcal{H}\colon \mathbf{W}^{\mathrm{op}} \times \mathbf{W} \to \mathbf{Set}$:

$$\mathcal{H}(w^-, w^+) = \prod\nolimits_{(\ell\colon S)\in w^-} \mathrm{range}(S)(w^+)$$

  So, heaps over w are the elements of $\mathcal{H}(w, w)$
- Ground store case: Every $\mathrm{range}(S)$ is constant $\Rightarrow$ heaps form a contravariant functor

Initializations and Heap Functor

$$
\begin{tikzcd}
{[\mathbf{W}, \mathbf{Set}]} \arrow[rr, bend left, "\mathfrak{u}^\star"] \arrow[rr, phantom, "\perp"] & & {[\mathbf{E}, \mathbf{Set}]} \arrow[ll, bend left, "\mathfrak{u}_\star"] \arrow[rr, bend left, "(-)\times H"] \arrow[rr, phantom, "\perp"] & & {[\mathbf{E}, \mathbf{Set}]} \arrow[ll, bend left, "(-)^H"] \arrow[loop right, "P"]
\end{tikzcd}
$$

- Objects of $\mathbf{E}$ are heap layouts $w \in |\mathbf{W}|$
- Morphism $\epsilon\colon w \rightsquigarrow w'$ of $\mathbf{E}$ (initializations) consist of
  - An injection $\rho\colon w \hookrightarrow w'$
  - A heaplet $\eta \in \mathcal{H}(w' \setminus \rho[w], w')$
- $w \mapsto \mathcal{H}(w, w)$ becomes a functor $H\colon \mathbf{E} \to \mathbf{Set}$

Hiding Monad

$$\begin{tikzcd}[\mathbf{W}, \mathbf{Set}] \arrow[r, bend left, "\mathfrak{u}^\star"] \arrow[r, phantom, "\perp"] & {[\mathbf{E}, \mathbf{Set}]} \arrow[l, bend left, "\mathfrak{u}_\star"] \arrow[r, bend left, "(-)\times H"] \arrow[r, phantom, "\perp"] & {[\mathbf{E}, \mathbf{Set}]} \arrow[l, bend left, "(-)^H"] \arrow[loop right, "P"]\end{tikzcd}$$

Hidinig monad identifies values that do not depend on unreachable locations:

$$(PX)w = \int^{\rho\colon w \to w' \in w \downarrow \mathfrak{u}} Xw' = \left(\sum\nolimits_{\rho\colon w \to w'} Xw' \right) /\!\sim$$

where $\mathfrak{u}\colon \mathbf{E} \to \mathbf{W}$ is an obvious forgetful functor

Full Ground Store BI-Hyperdoctrine

BI-Hyperdoctrines

- A BI-algebra is a Heyting algebra, which is a commutative monoid $(M, e, \star)$ equipped with a right order-adjoint $-\!\!\star$ (separating implication) to multiplication $\star$ (separating conjunction)
- A BI-Hyperdoctrine is determined by a BI-algebra, which supports quantification and equality
- A standard way to obtain a BI-Hyperdoctrine is by constructing an (internally) complete BI-algebra[4]

[4] Biering, Birkedal, and Torp-Smith, "BI-hyperdoctrines, Higher-order Separation Logic, and Abstraction".

Our Construction

- We coherently upgrade the previous model, so as to deal with partial initializations $\hat{\mathbf{E}}$ and partial heaps $\hat{H}$
- The complete BI-algebra in question Θ is the upward closed subfunctor of

$$[\mathbf{W}^{\mathrm{op}}, 2^{(-)}](\mathbf{W}^{\mathrm{op}} \xrightarrow{\hat{P}\hat{H}} \mathbf{Set}) \colon \mathbf{W}^{\mathrm{op}} \to \mathbf{Set}^{\mathrm{op}}$$

  regarded as a presheaf in $[\mathbf{W}, \mathbf{Set}]$
- Thus two dimensions of locality: via allocation and via separation

Key Observation I

The involved presheave toposes $[\mathbf{C}, \mathbf{Set}]$ are De Morgan, i.e.

$$\begin{tikzcd} \bullet \arrow[r] \arrow[d] & \bullet \arrow[d, dotted] \\ \bullet \arrow[r, dotted] & \bullet \end{tikzcd}$$

in $\mathbf{C}$

Equivalently: 2 is a retract of the subobject classifier Ω in $[\mathbf{C}, \mathbf{Set}]$, but $\Omega \not\cong 2$ (!)

I.e. our toposes are not Boolean, but someone close to Boolean, which plays a key role in constructing $\ominus$

Key Observation II

The diagram

$$\begin{tikzcd}
{[\hat{\mathbf{E}}, \mathbf{Set}]} \arrow[r, "2^{(-)}"] \arrow[d, "\hat{P}"'] & {[\hat{\mathbf{E}}, \mathbf{Set}]^{\mathrm{op}}} \arrow[d, "\hat{\mathfrak{u}}_\star"] \\
{[\mathbf{W}, \mathbf{Set}^{\mathrm{op}}]^{\mathrm{op}}} \arrow[r, "{[\mathbf{W}, 2^{(-)}]^{\mathrm{op}}}"] & {[\mathbf{W}, \mathbf{Set}]^{\mathrm{op}}}
\end{tikzcd}$$

commutes up to isomorphism, hence Θ is an upward closed subfunctor of $\hat{\mathfrak{u}}_\star(2^{\hat{H}})$

The proof builds on

$$\mathbf{Set}\left(\int^{\rho\colon w \to w' \in w\downarrow \hat{\mathfrak{u}}} Xw', 2\right) \cong \int_{\rho\colon w \to w' \in w\downarrow \hat{\mathfrak{u}}} \mathbf{Set}(Xw', 2)$$

What Does not Work

- Keeping initializations total $\Rightarrow$ hidden heaplets are sensitive to modifications of unreachable parts
- Not restricting to upward closed parts $\Rightarrow$ separating conjunction does not have unit (by Yoneda lemma)
- Building a BI-algebra from an internal partial commutative monoid $\Rightarrow$ conflict with irrelevance of the address space structure

Example: Existential Quantification

variable context

hidden heap representative

The judgment

$$- \, , \rho, \eta \vDash \exists \ell.\, \ell \hookrightarrow 5$$

is always valid, because it's always possible to extend the heap, so that it contains a reference to 5

Example: Implication

The clause for implication is “two-dimensional”:

$$
\begin{aligned}
s, \rho, \eta \vDash \phi \Rightarrow \psi \quad \text{if} \quad & \text{for all} \quad (\rho, \eta) \sim (\rho', \eta') \quad \text{and} \\
& \text{for all} \quad \eta' \leq \eta'', \\
& s, \rho', \eta'' \vDash \phi \quad \text{implies} \quad s, \rho', \eta'' \vDash \psi
\end{aligned}
$$

– different from the standard Kripke semantics

Separating example:

$$
\begin{aligned}
\phi &= \ell \vdash \exists \ell'.\, \exists x.\, \ell \hookrightarrow \ell' \wedge \ell' \hookrightarrow x \\
\psi &= \ell \vdash \exists \ell'.\, \ell \hookrightarrow \ell' \wedge \ell' \hookrightarrow 6
\end{aligned}
$$

Further Properties[5]

- Monotonicity:

$$s, \rho, \eta \vDash \phi \quad \text{and} \quad \eta \leq \eta' \quad \text{imply} \quad s, \rho, \eta' \vDash \phi$$

- Shrinkage: If $\eta' \leq \eta$, and η' contains all cells reachable from s and ρ then

$$s, \rho, \eta \vDash \phi \quad \text{implies} \quad s, \rho, \eta' \vDash \phi$$

[5] Calcagno, O'Hearn, and Bornat, "Program logic and equivalence in the presence of garbage collection".

Future Work

- Connect program semantics (total heaps) and logic (partial heaps)
- Frame rule, Hoare logic, soundness, relative completeness[6]
- Dynamic logic (box and diamond modalities)
- Partiality as ownership? $\implies$ Concurrent separations logic
- Completeness w.r.t. intensional models[7]

[6] Goncharov and Schröder, “A Relatively Complete Generic Hoare Logic for Order-Enriched Effects”.

[7] Jung et al., “Iris from the ground up: A modular foundation for higher-order concurrent separation logic”.

Questions?

Biering, Bodil, Lars Birkedal, and Noah Torp-Smith. "BI-hyperdoctrines, Higher-order Separation Logic, and Abstraction". In: *ACM Trans. Program. Lang. Syst.* 29.5 (Aug. 2007).

Calcagno, Cristiano, Peter O'Hearn, and Richard Bornat. "Program logic and equivalence in the presence of garbage collection". In: *Theoretical Computer Science* 298.3 (2003). Foundations of Software Science and Computation Structures, pp. 557 –581.

Goncharov, Sergey and Lutz Schröder. "A Relatively Complete Generic Hoare Logic for Order-Enriched Effects". In: *Proc. 28th Annual Symposium on Logic in Computer Science (LICS 2013)*. IEEE, 2013, pp. 273–282.

Jung, Ralf et al. "Iris from the ground up: A modular foundation for higher-order concurrent separation logic". In: *Journal of Functional Programming* 28 (2018), e20.

Kammar, Ohad et al. “A monad for full ground reference cells”. In: *32nd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2017, Reykjavik, Iceland, June 20-23, 2017*. 2017, pp. 1–12.

Levy, Paul Blain, John Power, and Hayo Thielecke. “Modelling Environments in Call-By-Value Programming Languages”. In: *Inf. & Comp* 185 (2002), p. 2003.

Program Equivalences for Local Store

Monad by Kammar et al:[8]

- "Full ground": storable values depend on the current heap layout.
- Important program properties: Irrelevance of order of memory allocation, unused cells are garbage collected.

$$\mathsf{let}\ \ell := \mathsf{new}\, v;\ \ell' := \mathsf{new}\, w\ \mathsf{in}\ p \;=\; \mathsf{let}\ \ell' := \mathsf{new}\, w;\ \ell := \mathsf{new}\, v\ \mathsf{in}\ p$$

$$\mathsf{let}\ \ell := \mathsf{new}\, v\ \mathsf{in}\ \mathsf{ret}\,\star \;=\; \mathsf{ret}\,\star$$

$$\mathsf{let}\ \ell := \mathsf{new}\, v\ \mathsf{in}\ (\mathsf{if}\ \ell = \ell'\ \mathsf{then\ true\ else\ false}) \;=\; \mathsf{false}$$

[8] Kammar et al., "A monad for full ground reference cells".

The Semantics

- $s, \rho, \eta \models \top$
- $s, \rho, \eta \models \phi \wedge \psi$ if $s, \rho, \eta \models \phi$ and $s, \rho, \eta \models \psi$
- $s, \rho, \eta \models \phi \vee \psi$ if $s, \rho, \eta \models \phi$ or $s, \rho, \eta \models \psi$
- $s, \rho, \eta \models \phi \Rightarrow \psi$ if for all $(\rho, \eta) \sim (\rho', \eta')$ and $\eta' \leq \eta''$,

  $s, \rho', \eta'' \models \phi$ implies $s, \rho', \eta'' \models \psi$
- $s, \rho, \eta \models \phi(v)$ if $s, \rho, ((\llbracket \Gamma \vdash_{\mathsf{v}} v \colon A \rrbracket_{w'} \circ \underline{\Gamma}\rho)s, \eta) \models \phi$
- $s, \rho, (a, \eta) \models x.\, \phi$ if $a = (X\rho)b$ and $(s, b), \rho, \eta \models \phi$
- $s, \rho, \eta \models \ell \hookrightarrow v$ if $\eta = (w'' \subseteq w', \delta \in \mathcal{H}(w'', w'))$ and

  $\delta(r \colon S) = (\llbracket \Gamma \vdash_{\mathsf{v}} v \colon \mathrm{CType}(S) \rrbracket_{w'} \circ \underline{\Gamma}\rho)s$

  where $(\llbracket \Gamma \vdash_{\mathsf{v}} \ell \colon \mathrm{Ref}_S \rrbracket_{w'} \circ \underline{\Gamma}\rho)s = (r \colon S) \in w''$

The Semantics

- $s, \rho, \eta \models v = u$
  if $\quad ([\![\Gamma \vdash_{\mathsf{v}} v \colon A]\!]_{w''} \circ \underline{\Gamma}\rho' \circ \underline{\Gamma}\rho)(s) = ([\![\Gamma \vdash_{\mathsf{v}} u \colon A]\!]_{w''} \circ \underline{\Gamma}\rho' \circ \underline{\Gamma}\rho)(s)$

  for some $\rho' \colon w' \to w''$
- $s, \rho, \eta \models \phi \star \psi$ if for suitable $w_1, w_2, \eta \in \mathcal{H}(w_1 \uplus w_2, w')$,

  $s, \rho, (w_1 \subseteq w', \mathcal{H}(w_1 \subseteq w_1 \uplus w_2, w')\eta) \models \phi$ and

  $s, \rho, (w_2 \subseteq w', \mathcal{H}(w_2 \subseteq w_1 \uplus w_2, w')\eta) \models \psi$
- $s, \rho, \eta \models \phi \mathrel{-\!\!\star} \psi$ if for all $(\rho', \eta_1) \sim (\rho, \eta)$ and for all η_2 such that $\eta_1 \cdot \eta_2$ is defined,

  $s, \rho', \eta_2 \models \phi$ implies $s, \rho', \eta_1 \cdot \eta_2 \models \psi$
- $s, \rho, \eta \models \exists \phi$ if $\underline{\Gamma}(\hat{\mathfrak{u}}\epsilon \circ \rho)s, \mathrm{id}_{w''}, (a, \hat{H}\epsilon \circ \eta) \models \phi$ for some $\epsilon \colon w' \rightsquigarrow w''$, $a \in \underline{A}w''$
- $s, \rho, \eta \models \forall \phi$ if $\underline{\Gamma}(\hat{\mathfrak{u}}\epsilon \circ \rho)s, \mathrm{id}_{w''}, (a, \hat{H}\epsilon \circ \eta) \models \phi$ for all $\epsilon \colon w' \rightsquigarrow w''$, $a \in \underline{A}w''$