LOCAL LOCAL REASONING: A BI-HYPERDOCTRINE FOR FULL GROUND STORE

MIRIAM POLZER  SERGEY GONCHAROY
FAU ERLANGEN-NÜRNBERG

FoSSaCS/ETAPS, DUBLIN LUXEMBOURG INTERNET, MARCH 30, 2020
Motivation: Logical Conundrum
Recap of Full Ground Store Monad
Full Ground Store BI-Hyperdoctrine
Further Work
Motivation
raise.

It is easy to prove

\[
\begin{align*}
\{ \text{true} \} \\
x := \text{cons}_2(1, 2) ; \\
\{x \to 1, 2\} \\
x := 3 \\
\{(\exists x. x \to 1, 2) \land x = 3\}.
\end{align*}
\]

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 \(\exists x. x \to 1, 2\) is an “unobservable assertion” is worrisome.
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 entirely.
- 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$^3$):

(put) \[ \frac{\Gamma \vdash v \cdot \text{Ref}_S \quad \Gamma \vdash v \cdot \text{CType}(S) \quad \Gamma \vdash_c \ell := v \cdot 1}{\Gamma \vdash_c \ell := v \cdot 1} \]

(get) \[ \frac{\Gamma \vdash v \cdot \text{Ref}_S \quad \Gamma \vdash v \cdot \text{CType}(S)}{\Gamma \vdash_c !\ell \cdot \text{CType}(S)} \]

\[
\begin{align*}
\Gamma, \ell_1: \text{Ref}_{S_1}, \ldots, \ell_n: \text{Ref}_{S_n} \vdash v_1: \text{CType}(S_1) \\
\vdots \\
\Gamma, \ell_1: \text{Ref}_{S_1}, \ldots, \ell_n: \text{Ref}_{S_n} \vdash v_n: \text{CType}(S_n)
\end{align*}
\]

(new) \[ \frac{\Gamma, \ell_1: \text{Ref}_{S_1}, \ldots, \ell_n: \text{Ref}_{S_n} \vdash p \cdot A \quad \Gamma \vdash_c \text{letref} \ell_1 := v_1, \ldots, \ell_n := v_n \text{ in } p \cdot A}{\Gamma \vdash_c \text{letref} \ell_1 := v_1, \ldots, \ell_n := v_n \text{ in } p \cdot A} \]

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

---

$^3$Levy, Power, and Thielecke, “Modelling Environments in Call-By-Value Programming Languages”.
**Slogan:** The **local** (full ground) store monad is just a **global** store monad transform of the hiding monad sandwiched within a geometric morphism.
Set of locations $\mathcal{L} \cong \mathbb{N}$
Set of sorts $S$
Objects of $W$ are maps $w : \mathcal{L} \rightarrow_{\text{fin}} S$
Morphism of $W$ are type preserving injections $
\rho : \text{dom } w \leftrightarrow \text{dom } w'$
Heaplets

Postulate a semantics of sorts over each world
range: \( S \rightarrow [W, \text{Set}] \)

A heap would send each \((\ell: S) \in w\) to \(\text{range}(S)(w)\), and it is not a functor (!)

Hence heaplets: \( \mathcal{H}: W^{\text{op}} \times W \rightarrow \text{Set}: \)

\[
\mathcal{H}(w^-, w^+) = \prod_{(\ell: S) \in w^-} \text{range}(S)(w^+) 
\]

So, heaps over \( w \) are the elements of \( \mathcal{H}(w, w) \)

Ground store case: Every \( \text{range}(S) \) is constant \( \Rightarrow \) heaps form a contravariant functor
Initializations and Heap Functor

- Objects of $E$ are heap layouts $w \in |W|$
- Morphism $\epsilon : w \sim w'$ of $E$ (initializations) consist of
  - An injection $\rho : w \hookrightarrow w'$
  - A heaplet $\eta \in \mathcal{H}(w' \setminus \rho[w], w')$
- $w \mapsto \mathcal{H}(w, w)$ becomes a functor $H : E \to \text{Set}$
Hiding monad identifies values that do not depend on unreachable locations:

$$(PX)w = \int_{\rho: w \to w' \in W \downarrow u} Xw' = \left(\sum_{\rho: w \to w'} Xw'\right) / \sim$$

where $u: E \to W$ is an obvious forgetful functor
Full Ground Store
BI-Hyperdoctriline
A BI-algebra is a Heyting algebra, which is a commutative monoid \((M, e, \star)\) equipped with a right order-adjoint \(\leadsto \star\) (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”. 
We coherently upgrade the previous model, so as to deal with partial initializations $\hat{E}$ and partial heaps $\hat{H}$.

The complete BI-algebra in question $\Theta$ is the upward closed subfunctor of

$$[[W^{\text{op}}, 2^{(-)}](W^{\text{op}} \xrightarrow{\hat{P}H} \textbf{Set}) : W^{\text{op}} \to \text{Set}^{\text{op}}]$$

regarded as a presheaf in $[W, \textbf{Set}]$.

Thus two dimensions of locality: via allocation and via separation.
The involved presheave toposes $[\mathcal{C}, \textbf{Set}]$ are De Morgan, i.e.

![Diagram](image)

in $\mathcal{C}$

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

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

\[
\begin{array}{ccc}
[W, \text{Set}]^\text{op} & \xrightarrow{[\mathcal{W}, 2(-)]^\text{op}} & [W, \text{Set}]^\text{op} \\
\phi & \downarrow & \downarrow \\
[W, \text{Set}]^\text{op} & \xrightarrow{[\mathcal{W}, 2(-)]^\text{op}} & [W, \text{Set}]^\text{op}
\end{array}
\]

commutes up to isomorphism, hence \( \Theta \) is an upward closed subfunctor of \( \hat{u}_*(2\hat{H}) \)

The proof builds on

\[
\text{Set} \left( \int \rho: w \to w' \in w \downarrow \hat{u} \quad Xw', 2 \right) \cong \int \rho: w \to w' \in w \downarrow \hat{u} \quad \text{Set}(Xw', 2)
\]
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
The judgment

\[ -, \rho, \eta \models \exists \ell. \ell \rightarrow 5 \]

is always valid, because it’s always possible to extend the heap, so that it contains a reference to 5
The clause for implication is “two-dimensional”:

\[ s, \rho, \eta \models \phi \Rightarrow \psi \quad \text{if} \quad \text{for all} \quad (\rho, \eta) \sim (\rho', \eta') \quad \text{and} \]
\[ \text{for all} \quad \eta' \leq \eta'', \]
\[ s, \rho', \eta'' \models \phi \quad \text{implies} \quad s, \rho', \eta'' \models \psi \]

– different from the standard Kripke semantics

Separating example:

\[ \phi = \ell \vdash \exists \ell'. \exists x. \ell \leftrightarrow \ell' \land \ell' \leftrightarrow x \]

\[ \psi = \ell \vdash \exists \ell'. \ell \leftrightarrow \ell' \land \ell' \leftrightarrow 6 \]
Further Properties\textsuperscript{5}

- Monotonicity:

\[ s, \rho, \eta \models \phi \quad \text{and} \quad \eta \leq \eta' \quad \text{imply} \quad s, \rho, \eta' \models \phi \]

- Shrinkage: If \( \eta' \leq \eta \), and \( \eta' \) contains all cells reachable from \( s \) and \( \rho \) then

\[ s, \rho, \eta \models \phi \quad \text{implies} \quad s, \rho, \eta' \models \phi \]

\textsuperscript{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? \(\Rightarrow\) 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?


Monad by Kammar et al:\textsuperscript{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.

\[
\begin{align*}
\text{let } \ell := \text{new } v; \ell' := \text{new } w \text{ in } p &= \text{let } \ell' := \text{new } w; \ell := \text{new } v \text{ in } p \\
\text{let } \ell := \text{new } v \text{ in } \text{ret} \star &= \text{ret} \star \\
\text{let } \ell := \text{new } v \text{ in } (\text{if } \ell = \ell' \text{ then true else false}) &= \text{false}
\end{align*}
\]
The Semantics

\( s, \rho, \eta \models T \)

\( s, \rho, \eta \models \phi \land \psi \) if \( s, \rho, \eta \models \phi \) and \( s, \rho, \eta \models \psi \)

\( s, \rho, \eta \models \phi \lor \psi \) 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 \( s, \rho, (([\Gamma \vdash v : A]_{w'} \circ \Gamma_\rho)s, \eta) \models \phi \)

\( s, \rho, (a, \eta) \models x. \phi \) if \( a = (X_\rho)b \) and \( (s, b), \rho, \eta \models \phi \)

\( s, \rho, \eta \models \ell \mapsto v \) if \( \eta = (w'' \subseteq w', \delta \in \mathcal{H}(w'', w')) \) and

\( \delta(r : S) = ([\Gamma \vdash v : \text{CType}(S)]_{w'} \circ \Gamma_\rho)s \)

where \((\Gamma \vdash \ell : \text{Ref}_S]_{w'} \circ \Gamma_\rho)s = (r : S) \in w'' \)
The Semantics

- \( s, \rho, \eta \models v = u \)
  if \((\Gamma \vdash v : A)_{w''} \circ \Gamma\rho' \circ \Gamma\rho)(s) = (\Gamma \vdash u : A)_{w''} \circ \Gamma\rho' \circ \Gamma\rho)(s)\)
  for some \( \rho' : w' \rightarrow w'' \)

- \( s, \rho, \eta \models \phi \star \psi \) if for suitable \( w_1, w_2, \eta \in H(w_1 \sqcup w_2, w') \),
  \( s, \rho, (w_1 \subseteq w', H(w_1 \subseteq w_1 \sqcup w_2, w')\eta) \models \phi \) and
  \( s, \rho, (w_2 \subseteq w', H(w_2 \subseteq w_1 \sqcup w_2, w')\eta) \models \psi \)

- \( s, \rho, \eta \models \phi \rightarrow \psi \) if for all \((\rho', \eta_1) \sim (\rho, \eta)\) and for all \( \eta_2 \) 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 \( \Gamma(\hat{\epsilon} \circ \rho)s, \text{id}_{w''}, (a, \hat{H} \epsilon \circ \eta) \models \phi \) for some \( \epsilon : w' \leadsto w'', a \in Aw'' \)

- \( s, \rho, \eta \models \forall \phi \) if \( \Gamma(\hat{\epsilon} \circ \rho)s, \text{id}_{w''}, (a, \hat{H} \epsilon \circ \eta) \models \phi \) for all \( \epsilon : w' \leadsto w'', a \in Aw'' \)