

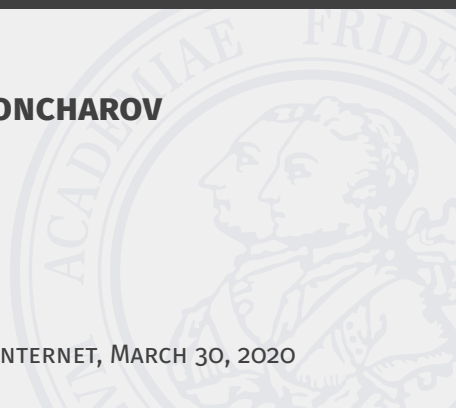
# LOCAL LOCAL REASONING: A BI-HYPERDOCTRINE FOR FULL GROUND STORE



MIRIAM POLZER    **SERGEY GONCHAROV**

FAU ERLANGEN-NÜRNBERG

FOSSACS/ETAPS, ~~DUBLIN~~  LUXEMBOURG INTERNET, MARCH 30, 2020



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

# MOTIVATION

# REYNOLDS' CONUNDRUM

## Intuitionistic Reasoning about Shared Mutable Data Structure\*

John C. Reynolds  
Department of Computer Science  
Carnegie Mellon University

July 28, 2000

raise.

It is easy to prove

```
{true}
x := cons2(1, 2);
{x → 1, 2}
x := 3
{(∃x. x → 1, 2) ∧ 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  $\exists x. x \rightarrow 1, 2$  is an “unobservable assertion” is worrisome.

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

- ❁ 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<sup>1</sup>, and argue that it predetermines the approach to the conundrum

---

<sup>1</sup>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<sup>2</sup>, but not after the monad was introduced

---

<sup>2</sup>Calcagno, O'Hearn, and Bornat, "Program logic and equivalence in the presence of garbage collection".

# **FULL GROUND STORE MONAD**

Key rules (FGCBV-style<sup>3</sup>):

$$\text{(put)} \frac{\Gamma \vdash_v l : \text{Ref}_S \quad \Gamma \vdash_v v : \text{CType}(S)}{\Gamma \vdash_c l := v : 1} \qquad \text{(get)} \frac{\Gamma \vdash_v l : \text{Ref}_S}{\Gamma \vdash_c !l : \text{CType}(S)}$$

$$\begin{array}{c} \Gamma, l_1 : \text{Ref}_{S_1}, \dots, l_n : \text{Ref}_{S_n} \vdash_v v_1 : \text{CType}(S_1) \\ \vdots \\ \Gamma, l_1 : \text{Ref}_{S_1}, \dots, l_n : \text{Ref}_{S_n} \vdash_v v_n : \text{CType}(S_n) \\ \text{(new)} \frac{\Gamma, l_1 : \text{Ref}_{S_1}, \dots, l_n : \text{Ref}_{S_n} \vdash_c p : A}{\Gamma \vdash_c \text{letref } l_1 := v_1, \dots, l_n := v_n \text{ in } p : A} \end{array}$$

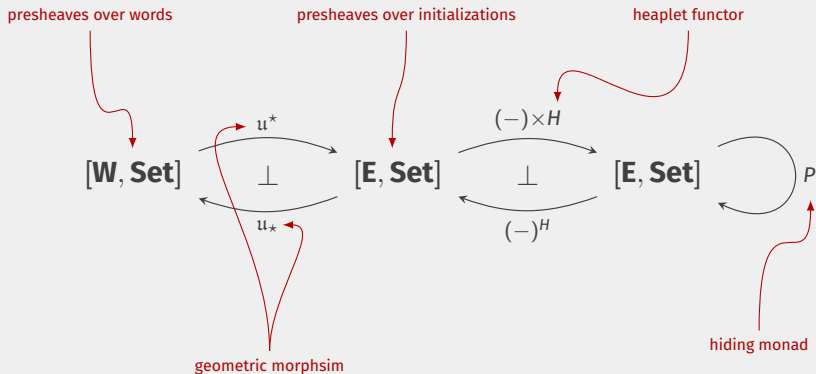
**Example:**  $\text{letref } l_1 := (0, \text{inr } \star, \text{inl } l_2); l_2 := (1, \text{inl } l_1, \text{inr } \star) \text{ in ret } l_1$

---

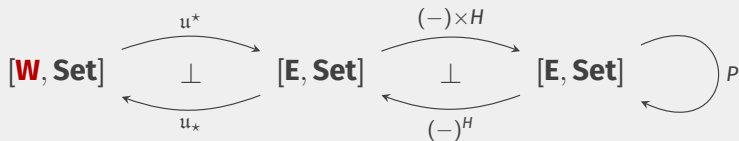
<sup>3</sup>Levy, Power, and Thielecke, “Modelling Environments in Call-By-Value Programming Languages”.



# FULL GROUND STORE MONAD: EXPLODED-VIEW



**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**  $\mathcal{S}$
- Objects of **W** are maps  $w: \mathcal{L} \rightarrow_{\text{fin}} \mathcal{S}$
- Morphism of **W** are type preserving injections  
 $\rho: \text{dom } w \hookrightarrow \text{dom } w'$

# HEAPLETS

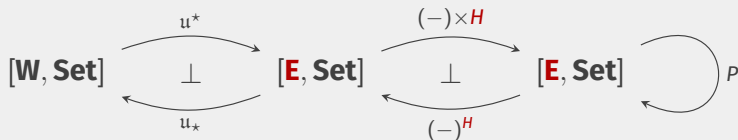
- Postulate a semantics of sorts over each world  
range:  $\mathcal{S} \rightarrow [\mathbf{W}, \mathbf{Set}]$
- A **heap** would send each  $(\ell: S) \in w$  to  $\text{range}(S)(w)$ , and it is not a functor (!)
- Hence **heaplets**:  $\mathcal{H}: \mathbf{W}^{\text{op}} \times \mathbf{W} \rightarrow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

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

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

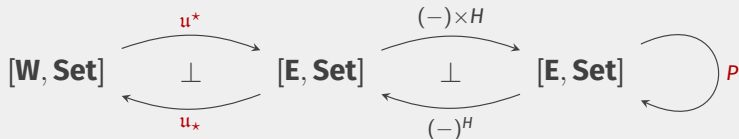
- Ground store case: Every  $\text{range}(S)$  is constant  $\Rightarrow$  heaps form a contravariant functor

# INITIALIZATIONS AND HEAP FUNCTOR



- Objects of  $\mathbf{E}$  are heap layouts  $w \in |\mathbf{W}|$
- Morphism  $\epsilon: w \rightsquigarrow w'$  of  $\mathbf{E}$  (**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  $H: \mathbf{E} \rightarrow \mathbf{Set}$

# HIDING MONAD



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

$$(PX)_W = \int^{\rho: W \rightarrow W' \in W \downarrow u} X_{W'} = \left( \sum_{\rho: W \rightarrow W'} X_{W'} \right) / \sim$$

where  $u: \mathbf{E} \rightarrow \mathbf{W}$  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  $\rightarrow\star$  (**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<sup>4</sup>

---

<sup>4</sup>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  $\Theta$  is the upward closed subfunctor of

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

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.



in  $\mathbf{C}$

Equivalently:  $2$  is a retract of the subobject classifier  $\Omega$  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

$$\begin{array}{ccc} [\hat{\mathbf{E}}, \mathbf{Set}] & \xrightarrow{2^{(-)}} & [\hat{\mathbf{E}}, \mathbf{Set}]^{\text{op}} \\ \hat{p} \downarrow & & \downarrow \hat{u}_* \\ [\mathbf{W}, \mathbf{Set}^{\text{op}}]^{\text{op}} & \xrightarrow{[\mathbf{W}, 2^{(-)}]^{\text{op}}} & [\mathbf{W}, \mathbf{Set}]^{\text{op}} \end{array}$$

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

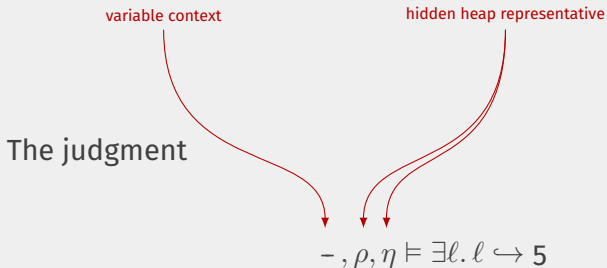
The proof builds on

$$\mathbf{Set} \left( \int^{\rho: W \rightarrow W' \in W \downarrow \hat{u}} XW', 2 \right) \cong \int_{\rho: W \rightarrow W' \in W \downarrow \hat{u}} \mathbf{Set}(XW', 2)$$

# 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



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”:

$s, \rho, \eta \vDash \phi \Rightarrow \psi$  if **for all**  $(\rho, \eta) \sim (\rho', \eta')$  and  
for all  $\eta' \leq \eta''$ ,  
 $s, \rho', \eta'' \vDash \phi$  implies  $s, \rho', \eta'' \vDash \psi$

– different from the standard Kripke semantics

Separating example:

$$\phi = l \vdash \exists l'. \exists x. l \hookrightarrow l' \wedge l' \hookrightarrow x$$

$$\psi = l \vdash \exists l'. l \hookrightarrow l' \wedge l' \hookrightarrow \mathbf{6}$$

⚙ Monotonicity:

$s, \rho, \eta \models \phi$  and  $\eta \leq \eta'$  imply  $s, \rho, \eta' \models \phi$

⚙ Shrinkage: If  $\eta' \leq \eta$ , and  $\eta'$  contains all cells reachable from  $s$  and  $\rho$  then

$s, \rho, \eta \models \phi$  implies  $s, \rho, \eta' \models \phi$

---

<sup>5</sup>Calcagno, O'Hearn, and Bornat, "Program logic and equivalence in the presence of garbage collection".

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

---

<sup>6</sup>Goncharov and Schröder, “A Relatively Complete Generic Hoare Logic for Order-Enriched Effects”.

<sup>7</sup>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.<sup>8</sup>

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

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

$\text{let } \ell := \text{new } v \text{ in } \text{ret } \star = \text{ret } \star$

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

---

<sup>8</sup>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  $s, \rho', \eta'' \models \psi$
- $s, \rho, \eta \models \phi(\mathbf{v})$  if  $s, \rho, ((\llbracket \Gamma \vdash_{\mathbf{v}} \mathbf{v} : \mathbf{A} \rrbracket_{w'} \circ \llbracket \rho \rrbracket) s, \eta) \models \phi$
- $s, \rho, (a, \eta) \models x. \phi$  if  $a = (X\rho)b$  and  $(s, b), \rho, \eta \models \phi$
- $s, \rho, \eta \models \ell \hookrightarrow \mathbf{v}$  if  $\eta = (w'' \subseteq w', \delta \in \mathcal{H}(w'', w'))$  and  
 $\delta(r : S) = ((\llbracket \Gamma \vdash_{\mathbf{v}} \mathbf{v} : \mathbf{CType}(S) \rrbracket_{w'} \circ \llbracket \rho \rrbracket) s$   
where  $((\llbracket \Gamma \vdash_{\mathbf{v}} \ell : \mathbf{Ref}_S \rrbracket_{w'} \circ \llbracket \rho \rrbracket) s = (r : S) \in w''$

# THE SEMANTICS

- $s, \rho, \eta \models v = u$   
if  $(\llbracket \Gamma \vdash_v v : A \rrbracket_{w''} \circ \llbracket \rho' \circ \llbracket \rho \rrbracket (s) = (\llbracket \Gamma \vdash_v u : A \rrbracket_{w''} \circ \llbracket \rho' \circ \llbracket \rho \rrbracket (s)$   
for some  $\rho' : w' \rightarrow w''$
- $s, \rho, \eta \models \phi \star \psi$  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  
 $s, \rho, (w_2 \subseteq w', \mathcal{H}(w_2 \subseteq w_1 \uplus w_2, w')\eta) \models \psi$
- $s, \rho, \eta \models \phi \rightarrow \psi$  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  $\llbracket \hat{u}\epsilon \circ \rho \rrbracket s, \text{id}_{w''}, (a, \hat{H}\epsilon \circ \eta) \models \phi$  for some  
 $\epsilon : w' \rightsquigarrow w'', a \in \underline{A}w''$
- $s, \rho, \eta \models \forall \phi$  if  $\llbracket \hat{u}\epsilon \circ \rho \rrbracket s, \text{id}_{w''}, (a, \hat{H}\epsilon \circ \eta) \models \phi$  for all  
 $\epsilon : w' \rightsquigarrow w'', a \in \underline{A}w''$