Yale

Layered and Object-Based Game Semantics

Arthur Oliveira Vale [1] Paul-André Melliès [2] Zhong Shao [1] Jérémie Koenig [1] Leo Stefanesco [3]

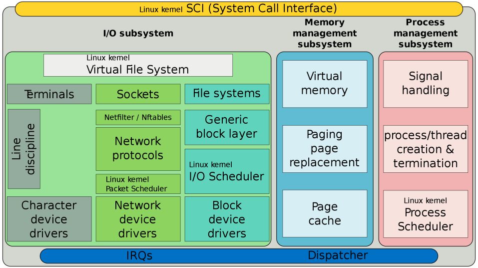
[1]Yale University, USA

[2]CNRS and Université de Paris

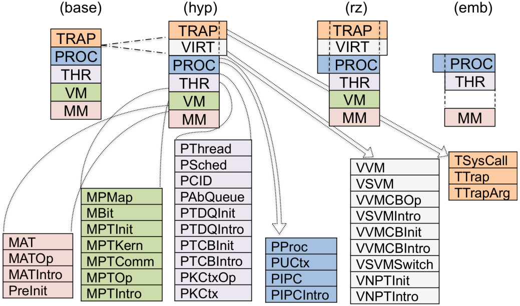
[3]MPI-SWS, Germany

January 21, 2022

Layered Systems



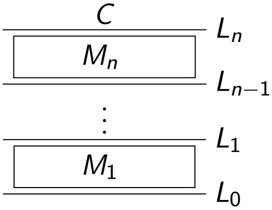
https://commons.wikimedia.org/wiki/File:Simplified_Structure_of_the_Linux_Kernel.svg



Certified Abstraction Layers

- **Layer:**

- **Layering:**



- **Certified Implementation:**

$$L_1 \vdash M : L_2$$

- **Encapsulation:**

???
↑
Abstract State
↑
Concrete State

Modeling State: Global vs Local State

Global State:

- C Memory Model
- ML Reference Types
- Haskell State Monads
- Algebraic Effects (Global)

  ⋮

Local State:

- Algebraic Effects (Local)
- Object-Based Semantics

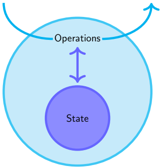
  ⋮

Object-Based Semantics

Lisp and Symbolic Computation, 9, 7–76 (1996)

Global State Considered Unnecessary: An Introduction to Object-Based Semantics

UDAY S. REDDY reddy@cs.uiuc.edu
Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL.

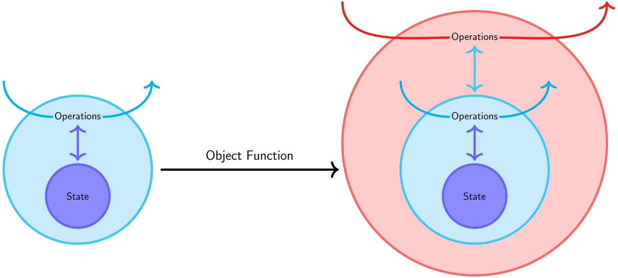


Object-Based Semantics

Lisp and Symbolic Computation, 9, 7–76 (1996)

**Global State Considered Unnecessary:
An Introduction to Object-Based Semantics**

UDAY S. REDDY reddy@cs.uiuc.edu
Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL.



Object-Based Semantics and Certified Abstraction Layers

$$\text{Object Type} \approx \text{Layer Signature}$$
$$\text{Object} \approx \text{Layer Specification}$$
$$\text{Object Function} \approx \text{Layer Implementation}$$
$$??? \approx \text{Certified Layer}$$

How to build compositional semantic models for certifying large heterogenous systems?

Our Contributions

- A novel model of Certified Abstraction Layers.
    - No explicit state
    - Rooted in linear logic
    - Clarifies some aspects of the abstract semantics of CAL
    - Supports an operational account as a game semantics model
    - Supports a denotational account as a domain-theoretic model
- Generalized notion of layer interface that faithfully encapsulates state
- An extension to handle non-determinism in layer interfaces
- An extension to support concurrency

Outline

Outline

Layer Signatures

DEFINITION

An **effect signature** consists of

- A set E of **operations**
- A set $\mathsf{ar}(e)$ to each operation e called the **arity** of e

EXAMPLE

- $\mathsf{Var} := \{\mathsf{get} : 1 \to \mathbb{N}, \mathsf{set} : \mathbb{N} \to 1\}$:

Operations $\mathsf{Var} = \{\mathsf{get}, \mathsf{set}(n) \mid n \in \mathbb{N}\}$

Arities
- $\mathsf{ar}(\mathsf{get}) = \mathbb{N}$
- $\mathsf{ar}(\mathsf{set}(n)) = 1 \cong \{\mathsf{ok}\}$

Layer Signatures

DEFINITION

An **effect signature** consists of

- A set E of **operations**
- A set $\mathsf{ar}(e)$ to each operation e called the **arity** of e

EXAMPLE

- $\mathsf{Var} := \{\mathsf{get} : 1 \to \mathbb{N}, \mathsf{set} : \mathbb{N} \to 1\}$:

Operations $\mathsf{Var} = \{\mathsf{get}, \mathsf{set}(n) \mid n \in \mathbb{N}\}$

Arities
- $\mathsf{ar}(\mathsf{get}) = \mathbb{N}$
- $\mathsf{ar}(\mathsf{set}(n)) = 1 \cong \{\mathsf{ok}\}$

State-Based Layer Specifications

$$\text{Var} := \{\text{get} : 1 \to \mathbb{N}, \text{set} : \mathbb{N} \to 1\}$$

State: $n, m \in S_{\text{Var}} := \mathbb{N}$

Initial State: 0

Transitions: $n \xrightarrow{get.n} n$

$n \xrightarrow{\text{set}(m).\text{ok}} m$

$$\text{FAI} := \{\text{fai} : 1 \to \mathbb{N}\}$$

State: $n \in S_{\text{FAI}} := \mathbb{N}$

Initial State: 0

Transitions: $n \xrightarrow{\text{fai}.n} n + 1$

State-Based Layer Specifications

$$\text{Var} := \{\text{get} : 1 \to \mathbb{N}, \text{set} : \mathbb{N} \to 1\}$$

State: $n, m \in S_{\text{Var}} := \mathbb{N}$

Initial State: 0

Transitions: $n \xrightarrow{get.n} n$

$n \xrightarrow{\text{set}(m).\text{ok}} m$

$$\text{FAI} := \{\text{fai} : 1 \to \mathbb{N}\}$$

State: $n \in S_{\text{FAI}} := \mathbb{N}$

Initial State: 0

Transitions: $n \xrightarrow{\text{fai}.n} n + 1$

Bounded Queue

Signature

$$E_{\mathsf{bq}} := \{\mathsf{enq} : \mathbb{U} \to 1, \mathsf{deq} : 1 \to \mathbb{U}\}$$

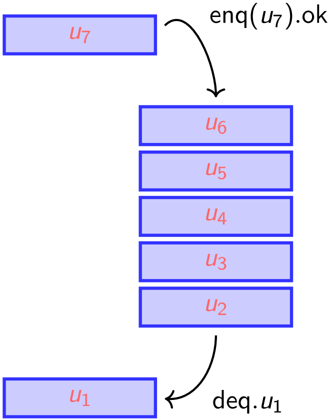
States $S_{\mathsf{bq}} = \mathbb{U}^*$

Initial State ϵ

Transitions

- $|\vec{q}| < N \Rightarrow \vec{q} \xrightarrow{\mathsf{enq}(v).\mathsf{ok}} \vec{q}v$
- $\vec{q} = v\vec{q'} \Rightarrow \vec{q} \xrightarrow{\mathsf{deq}.v} \vec{q'}$

Queue Semantics:



Bounded Queue

Signature

$$E_{bq} := \{\mathsf{enq} : \mathbb{U} \to 1, \mathsf{deq} : 1 \to \mathbb{U}\}$$

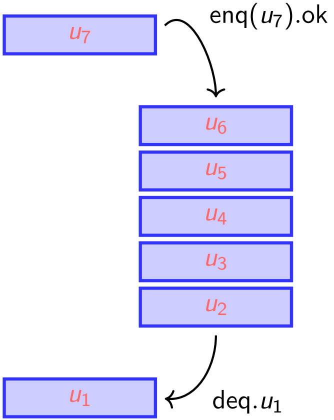
States $S_{bq} = \mathbb{U}^*$

Initial State ϵ

Transitions

- $|\vec{q}| < N \Rightarrow \vec{q} \xrightarrow{\mathsf{enq}(v).\mathsf{ok}} \vec{q}v$
- $\vec{q} = v\vec{q'} \Rightarrow \vec{q} \xrightarrow{\mathsf{deq}.v} \vec{q'}$

Queue Semantics:



Ring Buffer

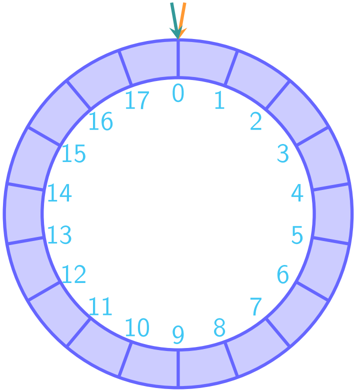
Signature:

$$E_{\mathsf{bq}} := \{\mathsf{set} : \mathbb{N} \times \mathbb{U} \to 1, \mathsf{get} : \mathbb{N} \to \mathbb{U}\} \cup \{\mathsf{fai}_1 : 1 \to \mathbb{N}, \mathsf{fai}_2 : 1 \to \mathbb{N}\}$$

State: $S_{\mathsf{rb}} := \mathbb{U}^{\mathbb{N}} \times \mathbb{N} \times \mathbb{N}$

Initial State: $(\varnothing, 0, 0)$

Initial State:



Ring Buffer

Signature:

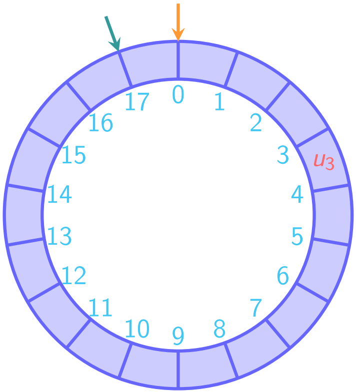
$$E_{bq} := \{\mathsf{set} : \mathbb{N} \times \mathbb{U} \to 1, \mathsf{get} : \mathbb{N} \to \mathbb{U}\} \cup \{\mathsf{fai}_1 : 1 \to \mathbb{N}, \mathsf{fai}_2 : 1 \to \mathbb{N}\}$$

State: $S_{rb} := \mathbb{U}^{\mathbb{N}} \times \mathbb{N} \times \mathbb{N}$

Initial State: $(\varnothing, 0, 0)$

set Semantics:

$$(\varnothing, 0, 17) \xrightarrow{\mathsf{set}(3,u_3).\mathsf{ok}} (\{3 \mapsto u_3\}, 0, 17)$$



Ring Buffer

Signature:

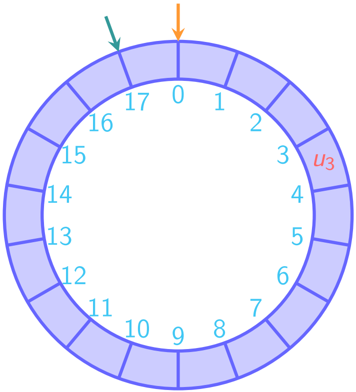
$$E_{bq} := \{\text{set} : \mathbb{N} \times \mathbb{U} \to 1, \text{get} : \mathbb{N} \to \mathbb{U}\} \cup \{\text{fai}_1 : 1 \to \mathbb{N}, \text{fai}_2 : 1 \to \mathbb{N}\}$$

State: $S_{rb} := \mathbb{U}^{\mathbb{N}} \times \mathbb{N} \times \mathbb{N}$

Initial State: $(\varnothing, 0, 0)$

get Semantics:

$$(\{3 \mapsto u_3\}, 0, 17) \xrightarrow{\text{get}(3).u_3} (\{3 \mapsto u_3\}, 0, 17)$$



Ring Buffer

Signature:

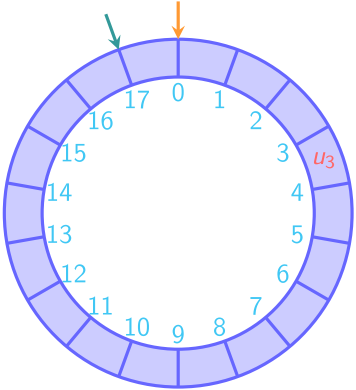
$$E_{bq} := \{\mathsf{set} : \mathbb{N} \times \mathbb{U} \to 1, \mathsf{get} : \mathbb{N} \to \mathbb{U}\} \cup \{\mathsf{fai}_1 : 1 \to \mathbb{N}, \mathsf{fai}_2 : 1 \to \mathbb{N}\}$$

State: $S_{rb} := \mathbb{U}^{\mathbb{N}} \times \mathbb{N} \times \mathbb{N}$

Initial State: $(\varnothing, 0, 0)$

get Semantics:

$$(\{3 \mapsto u_3\}, 0, 17) \xrightarrow{\mathsf{get}(0).u_3}$$



Ring Buffer

Signature:

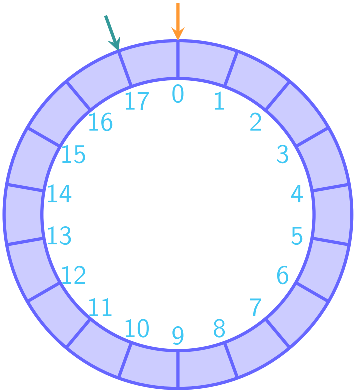
$$E_{\mathsf{bq}} := \{\mathsf{set} : \mathbb{N} \times \mathbb{U} \to 1, \mathsf{get} : \mathbb{N} \to \mathbb{U}\}$$
$$\cup \{\mathsf{fai}_1 : 1 \to \mathbb{N}, \mathsf{fai}_2 : 1 \to \mathbb{N}\}$$

State: $S_{\mathsf{rb}} := \mathbb{U}^{\mathbb{N}} \times \mathbb{N} \times \mathbb{N}$

Initial State: $(\varnothing, 0, 0)$

fai Semantics:

$$(\varnothing, 0, 17) \xrightarrow{\mathsf{fai}_1.0} (\varnothing, 1, 17)$$



Ring Buffer

Signature:

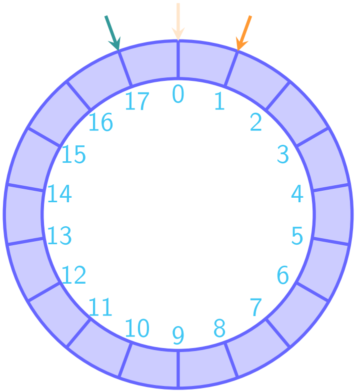
$$E_{bq} := \{\mathsf{set} : \mathbb{N} \times \mathbb{U} \to 1, \mathsf{get} : \mathbb{N} \to \mathbb{U}\} \cup \{\mathsf{fai}_1 : 1 \to \mathbb{N}, \mathsf{fai}_2 : 1 \to \mathbb{N}\}$$

State: $S_{rb} := \mathbb{U}^{\mathbb{N}} \times \mathbb{N} \times \mathbb{N}$

Initial State: $(\varnothing, 0, 0)$

fai Semantics:

$$(\varnothing, 0, 17) \xrightarrow{\mathsf{fai}_1 .0} (\varnothing, 1, 17)$$



Ring Buffer

Signature:

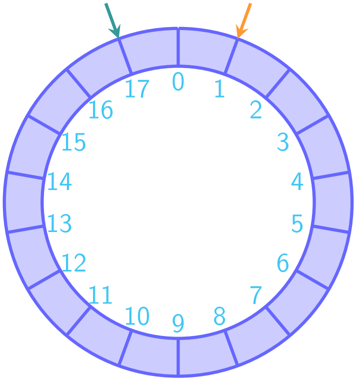
$$E_{\text{bq}} := \{\text{set} : \mathbb{N} \times \mathbb{U} \to 1, \text{get} : \mathbb{N} \to \mathbb{U}\} \cup \{\text{fai}_1 : 1 \to \mathbb{N}, \text{fai}_2 : 1 \to \mathbb{N}\}$$

State: $S_{\text{rb}} := \mathbb{U}^{\mathbb{N}} \times \mathbb{N} \times \mathbb{N}$

Initial State: $(\varnothing, 0, 0)$

fai Semantics:

$$(\varnothing, 1, 17) \xrightarrow{\text{fai}_2.17} (\varnothing, 1, 0)$$



Ring Buffer

Signature:

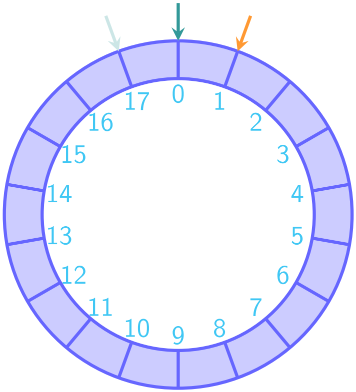
$$E_{bq} := \{\text{set} : \mathbb{N} \times \mathbb{U} \to 1, \text{get} : \mathbb{N} \to \mathbb{U}\}$$
$$\cup \{\text{fai}_1 : 1 \to \mathbb{N}, \text{fai}_2 : 1 \to \mathbb{N}\}$$

State: $S_{rb} := \mathbb{U}^{\mathbb{N}} \times \mathbb{N} \times \mathbb{N}$

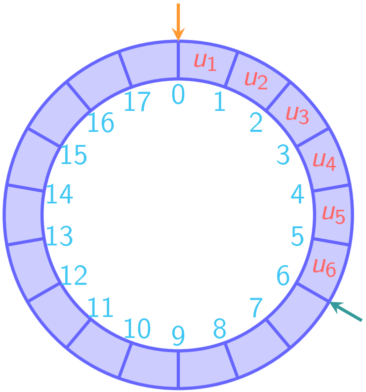
Initial State: $(\varnothing, 0, 0)$

fai Semantics:

$$(\varnothing, 1, 17) \xrightarrow{\text{fai}_2.17} (\varnothing, 1, 0)$$



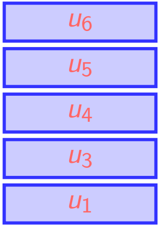
Implementing a Bounded Queue using a Ring Buffer



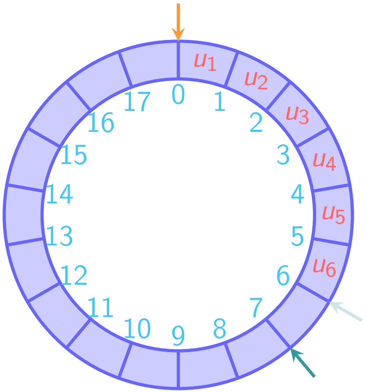
M

```
enq(u) {
  i ← fai₂();
  set(i,u);
  return ok
}
deq() {
  i ← fai₁();
  get(i)
}
```

enq(u_7)



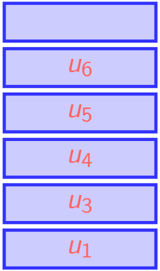
Implementing a Bounded Queue using a Ring Buffer



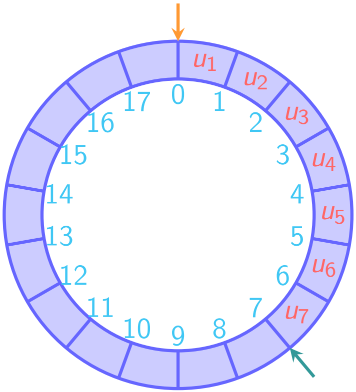
M

```
enq(u) {
  i ← fai₂();
  set(i,u);
  return ok
}
deq() {
  i ← fai₁();
  get(i)
}
```

enq(u_7)



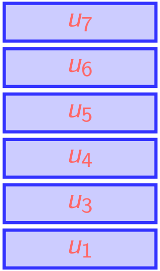
Implementing a Bounded Queue using a Ring Buffer



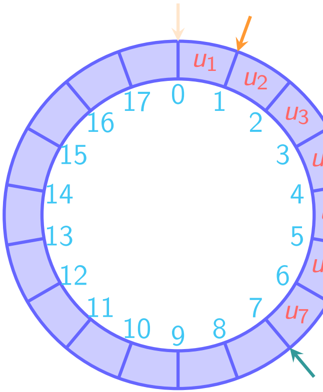
M

```
enq(u) {
  i ← fai₂();
  set(i,u);
  return ok
}
deq() {
  i ← fai₁();
  get(i)
}
```

enq(u_7).ok

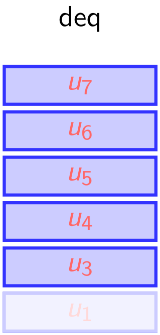


Implementing a Bounded Queue using a Ring Buffer

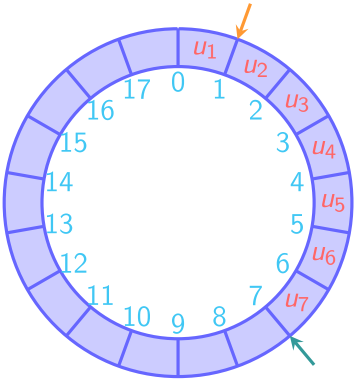


M

```
enq(u) {
  i ← fai₂();
  set(i,u);
  return ok
}
deq() {
  i ← fai₁();
  get(i)
}
```



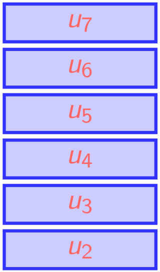
Implementing a Bounded Queue using a Ring Buffer



M

```
enq(u) {
  i ← fai₂();
  set(i,u);
  return ok
}
deq() {
  i ← fai₁();
  get(i)
}
```

deq.u_1



Outline

Layer Signatures and Interactions

Every effect signature E has an associated game, which for every $e \in E$ and $v \in \mathrm{ar}(e)$, has a play:

$$\begin{array}{ccc} \color{blue}{env} & & \\ \vdots & & \\ e & \longrightarrow & v \\ & & \vdots \\ & & \color{green}{sys} \end{array}$$

EXAMPLE

$$E_{\mathrm{rb}} := \{\mathrm{set} : \mathbb{N} \times \mathbb{U} \to 1, \mathrm{get} : \mathbb{N} \to \mathbb{U}\} \\ \cup \{\mathrm{fai}_1 : 1 \to \mathbb{N}, \mathrm{fai}_2 : 1 \to \mathbb{N}\}$$

$$\mathrm{set}(i, u) \longrightarrow \mathrm{ok}$$

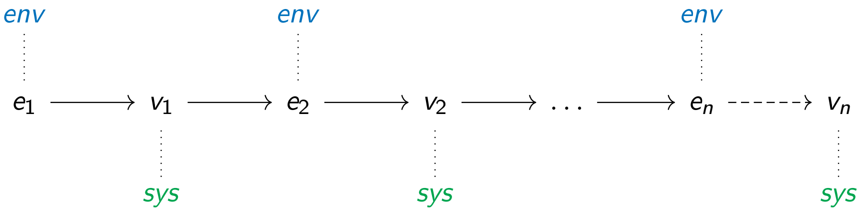
$$\mathrm{get}(i) \longrightarrow u$$

$$\mathrm{fai}_1 \longrightarrow v$$

$$\mathrm{fai}_2 \longrightarrow v$$

The Replay Modality $\dagger E$

The game $\dagger E$ consists of plays:



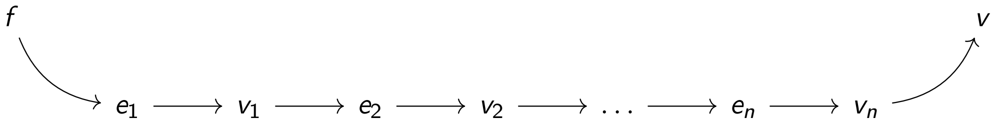
EXAMPLE

A linear logic modality $\dagger E_{\mathsf{rb}}$ describes plays as sequences of E_{rb} interactions:

- $\mathsf{fai}_1 \longrightarrow 0 \longrightarrow \mathsf{fai}_1 \longrightarrow 1 \longrightarrow \mathsf{fai}_1 \longrightarrow 2$
- $\mathsf{fai}_1 \longrightarrow 7 \longrightarrow \mathsf{get}(3) \longrightarrow u_3 \longrightarrow \mathsf{set}(2, u_2) \longrightarrow \mathsf{ok} \longrightarrow \mathsf{fai}_2 \longrightarrow 13$

Implementation Plays: $\dagger E \multimap F$

Plays of $\dagger E \multimap F$ are of the form



Implementations: Example

$$M := \left(\cup_{u \in \mathbb{U}} M^{\mathsf{enq}(u)} \right) \cup M^{\mathsf{deq}}$$

```
enq(u) {
  i ← fai₂();
  set(i,u);
  return ok
}
```

$$\Rightarrow \quad M^{\mathsf{enq}(u)} := \downarrow \left\{ \mathsf{enq}(u) \curvearrowright \mathsf{fai}_2 \longrightarrow i \longrightarrow \mathsf{set}(i,u) \longrightarrow \mathsf{ok} \nearrow \mathsf{ok} \;\middle|\; i \in \mathbb{N} \right\}$$

```
deq() {
  i ← fai₁();
  get(i)
}
```

$$\Rightarrow \quad M^{\mathsf{deq}} := \downarrow \left\{ \mathsf{deq} \searrow \mathsf{fai}_1 \longrightarrow i \longrightarrow \mathsf{get}(i) \longrightarrow u \nearrow u \;\middle|\; i \in \mathbb{N}, u \in \mathbb{U} \right\}$$

Regular Extensions

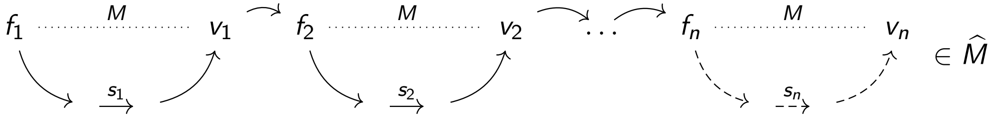
From an implementation:

$$M : \dagger E \multimap F$$

We build its regular extension:

$$\widehat{M} : \dagger E \multimap \dagger F$$

by



In general, approximately:

$$\widehat{M} \approx M^*$$

With local states implementations are stateless!

Implementations as Strategies

An implementation $M : E \to F$ is a **deterministic strategy** of type $\dagger E \multimap F$.
That is, M is a set of plays that is:

- **non-empty**,
- **prefix-closed**,
- **deterministic**

The Structure of the Replay Modality

The replay modality $\dagger -$ is a Comonad:

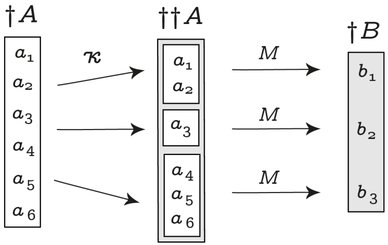
$$\epsilon_A : \dagger A \multimap A$$

$$\kappa_A : \dagger A \multimap \dagger\dagger A$$

The regular extension is just the Kleisli morphism:

$$M : \dagger A \multimap B$$

$$\dagger A \xrightarrow{\widehat{M}} \dagger B = \dagger A \xrightarrow{\kappa_A} \dagger\dagger A \xrightarrow{\dagger M} \dagger B$$



Outline

Layer Specifications from State Transition Systems

How to get rid of state-based specifications?

$\text{Counter} := \{\text{get} : 1 \to \mathbb{N}, \text{inc} : 1 \to 1\}$

State: $n \in S_{\text{Counter}} := \mathbb{N}$

Initial State: 0

Transitions: $n \xrightarrow{get.n} n$

$n \xrightarrow{\text{inc.ok}} n+1$

Denote the observable behaviors at state q as:

$$(S, \to)\sharp q$$

$$(S_{\text{Counter}}, \to_{\text{Counter}})\sharp 0 = \{\epsilon, \text{get}\cdot 0, \text{inc}\cdot\text{ok}, \text{get}\cdot 0\cdot\text{get}\cdot 0, \text{inc}\cdot\text{ok}\cdot\text{get}\cdot 1, \text{get}\cdot 0\cdot\text{inc}\cdot\text{ok}, \text{inc}\cdot\text{ok}\cdot\text{inc}\cdot\text{ok}, \text{get}\cdot 0\cdot\text{inc}\cdot\text{ok}\cdot\text{get}\cdot 1, \ldots\}$$

Layer Specifications from State Transition Systems

How to get rid of state-based specifications?

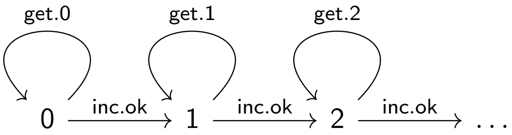
$\text{Counter} := \{\text{get} : 1 \to \mathbb{N}, \text{inc} : 1 \to 1\}$

State: $n \in S_{\text{Counter}} := \mathbb{N}$

Initial State: 0

Transitions: $n \xrightarrow{get.n} n$

$n \xrightarrow{\text{inc.ok}} n + 1$



Denote the observable behaviors at state q as:

$$(S, \to)\sharp q$$

$$(S_{\text{Counter}}, \to_{\text{Counter}})\sharp 0 = \{\epsilon, \text{get} \cdot 0, \text{inc} \cdot \text{ok}, \text{get} \cdot 0 \cdot \text{get} \cdot 0, \text{inc} \cdot \text{ok} \cdot \text{get} \cdot 1, \text{get} \cdot 0 \cdot \text{inc} \cdot \text{ok}, \text{inc} \cdot \text{ok} \cdot \text{inc} \cdot \text{ok}, \text{get} \cdot 0 \cdot \text{inc} \cdot \text{ok} \cdot \text{get} \cdot 1, \ldots\}$$

Layer Specifications from State Transition Systems

How to get rid of state-based specifications?

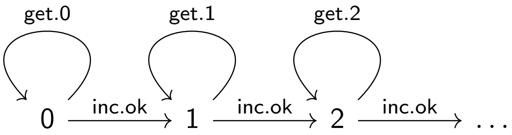
Counter $:= \{\text{get} : 1 \to \mathbb{N}, \text{inc} : 1 \to 1\}$

State: $n \in S_{\text{Counter}} := \mathbb{N}$

Initial State: 0

Transitions: $n \xrightarrow{get.n} n$

$n \xrightarrow{\text{inc.ok}} n + 1$



Denote the observable behaviors at state q as:

$$(S, \to)\sharp q$$

$$(S_{\text{Counter}}, \to_{\text{Counter}})\sharp 0 = \{\epsilon, \text{get} \cdot 0, \text{inc} \cdot \text{ok}, \text{get} \cdot 0 \cdot \text{get} \cdot 0, \text{inc} \cdot \text{ok} \cdot \text{get} \cdot 1, \text{get} \cdot 0 \cdot \text{inc} \cdot \text{ok}, \text{inc} \cdot \text{ok} \cdot \text{inc} \cdot \text{ok}, \text{get} \cdot 0 \cdot \text{inc} \cdot \text{ok} \cdot \text{get} \cdot 1, \ldots\}$$

Bounded Queue Specification: V_{bq}

EXAMPLE (BOUNDED QUEUE)

Signature $E_{\mathsf{bq}} = \{\mathsf{enq} : \mathbb{U} \to 1, \mathsf{deq} : 1 \to \mathbb{U}\}$

States $S_{\mathsf{bq}} = \mathbb{U}^*$

Initial State ϵ

Transitions

- $|\vec{q}| < N \Rightarrow \vec{q} \xrightarrow{\mathsf{enq}(v).\mathsf{ok}} \vec{q}v$
- $\vec{q} = v\vec{q'} \Rightarrow \vec{q} \xrightarrow{\mathsf{deq}.v} \vec{q'}$

We can define a layer specification for E_{bq} as

$$V_{\mathsf{bq}} := (S_{\mathsf{bq}}, \to)\sharp\epsilon$$

Stateless Variable Specification: V_{Var}

EXAMPLE

V_{Var} is the set of plays s of $\dagger\mathsf{Var}$ satisfying:

- $$s = \quad \mathsf{get} \longrightarrow v \longrightarrow \ldots \qquad \Rightarrow v = 0$$
- $$s = \quad \ldots\ \mathsf{get} \longrightarrow v_1 \longrightarrow \mathsf{get} \longrightarrow v_2 \ \ldots \qquad \Rightarrow v_1 = v_2$$
- $$s = \quad \ldots\ \mathsf{set}(n) \longrightarrow \mathsf{ok} \longrightarrow \mathsf{get} \longrightarrow v \ \ldots \qquad \Rightarrow v = n$$

Structure of Trace Sets

- Two sides: system and environment
- Interfaces play as system.

- The environment is unpredictable hence non-deterministic:
- The system is deterministic:

Structure of Trace Sets

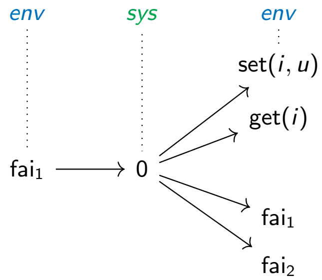
- Two sides: system and environment
- Interfaces play as system.
- The environment is unpredictable hence non-deterministic:

- The system is deterministic:

$$\underset{env}{\mathsf{fai}_1} \longrightarrow \underset{sys}{0} \longrightarrow \underset{env}{\mathsf{fai}_1} \longrightarrow \underset{sys}{1} \in V_{\mathrm{rb}}$$

Structure of Trace Sets

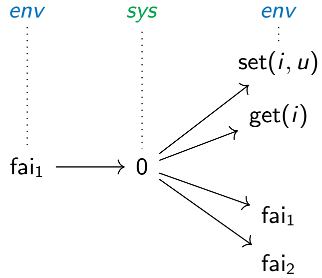
- Two sides: system and environment
- Interfaces play as system.
- The environment is unpredictable hence non-deterministic:



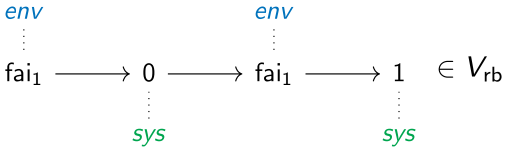
- The system is deterministic: $\underset{env}{\text{fai}_1} \longrightarrow \underset{sys}{0} \longrightarrow \underset{env}{\text{fai}_1} \longrightarrow \underset{sys}{1} \in V_{\text{rb}}$

Structure of Trace Sets

- Two sides: system and environment
- Interfaces play as system.
- The environment is unpredictable hence non-deterministic:



- The system is deterministic:

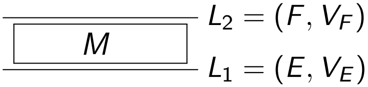


Layer Specifications

An **layer specification** V_E over E is a **deterministic strategy** of type $\dagger E$.
That is, V_E is a set of $\dagger E$ plays that is:

- **non-empty**,
- **prefix-closed**,
- **deterministic**

Layer Interfaces



DEFINITION

A **layer interface** is a pair $L = (E, V_E : \dagger E)$.

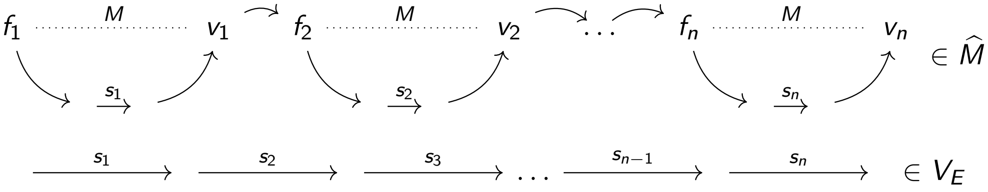
Outline

Certified Implementations

- $M : E_{\text{rb}} \to E_{\text{bq}}$ does not know rb semantics or bq semantics.
- Composing with V_{rb} "gives" the rb semantics to $\widehat{M}$:

$$f_1 \longrightarrow v_1 \longrightarrow f_2 \longrightarrow v_2 \longrightarrow \dots \longrightarrow f_n \longrightarrow v_n \quad \in V_E; \widehat{M}$$

$$\Longleftrightarrow$$



Certified Layer Implementations

DEFINITION

A **certified layer implementation** $L_E \vdash M : V_F$ consists of:

Underlay:	Overlay:	Implementation:
$L_E = (E, V_E)$	$L_F = (F, V_F)$	$M : E \to F$

Correctness:

$$V_F \subseteq V_E ; \widehat{M}$$

EXAMPLE

$$(E_{\mathsf{rb}}, V_{\mathsf{rb}}) \vdash M : (E_{\mathsf{bq}}, V_{\mathsf{bq}})$$

"For every V_{bq} behavior there is some V_{rb} that can be used by M to implement the V_{bq} behavior"

Outline

Non-Deterministic Layer Specifications

A **non-deterministic layer specification** $\mathcal{V}_E$ for E is a non-empty set of $V_E : \dagger E$.

A **non-deterministic layer interface** is pair $(E, \mathcal{V}_E)$.

EXAMPLE (NON-DETERMINISTIC rb INITIALIZATION)

- **Bounded queue:** $\mathcal{V}_{\mathsf{bq}} := \{V_{\mathsf{bq}}\}$
- **Ring buffer:**

Deterministic:

$$V_{\mathsf{rb}} := L_{\mathsf{rb}} \sharp (\varnothing, 0, 0)$$

Non-Deterministic:

$$\mathcal{V}_{\mathsf{rb}} := \{L^S_{\mathsf{rb}} \sharp (f, c, c) \mid f \in \mathbb{U}^N, c < N\}$$

Non-Deterministic Certified Abstraction Layers

DEFINITION

A **non-deterministic certified abstraction layer** $L_E \vdash M : L_F$ consists of:

Underlay:	Overlay:	Implementation:
$L_E = (E, \mathcal{V}_E)$	$L_F = (F, \mathcal{V}_F)$	$M : E \to F$

Correctness:

$$\forall V_E \in \mathcal{V}_E. \quad \exists V_F \in \mathcal{V}_F. \quad (E, V_E) \vdash M : (F, V_F)$$

EXAMPLE

$$(E_{\mathsf{rb}}, \mathcal{V}_{\mathsf{rb}}) \vdash M : (E_{\mathsf{bq}}, \mathcal{V}_{\mathsf{bq}})$$

Outline

Introduction

Layer Signatures: E

Implementations: $M : \dagger E \multimap F$

Layer Interfaces: $L = (E, V_E)$

Certified Layer Implementations: $L_1 \vdash M : L_2$

Non-Deterministic Layer Specifications: $L = (E, \mathcal{V}_E)$

Concurrent Layers: $L = (E, R, V_E)$

Conclusion

Certified Concurrent Layers

- A **concurrent layer interface** is a triple (E, R, V_E) where R is a **coherent congruence** between plays of $\dagger E$.
- R is essentially a "determinism preserving" equational theory:

$$s \cdot e_1 \cdot v_1 \cdot e_2 \cdot v_2 \cdot t \quad R \quad s \cdot e_2 \cdot v_2 \cdot e_1 \cdot v_1 \cdot t$$

- Implementations work on equivalence classes under coherent congruences:

$$M : \dagger_R E \multimap \dagger_S F$$

$$[s]_R \mapsto [t]_S$$

Check paper for details!

Equational Theories and †-Coalgebras

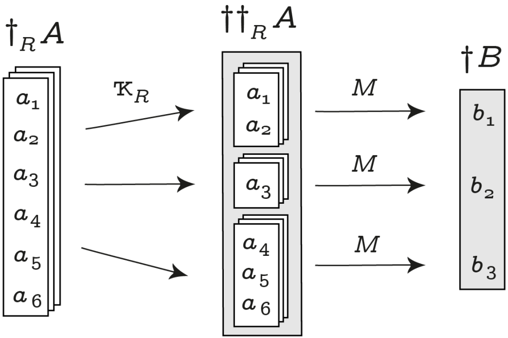
Given equational theory $R \in \text{Rel}(\dagger A, \dagger A)$ we construct a †-Coalgebra:

$$(\dagger_R A, \kappa_R : \dagger_R \multimap \dagger\dagger_R)$$

Generalized regular extension:

$$M : \dagger_R A \multimap B$$

$$\dagger_R A \xrightarrow{\widehat{M}} \dagger B = \dagger_R A \xrightarrow{\kappa_A} \dagger\dagger_R A \xrightarrow{\dagger M} \dagger B$$



Certified Concurrent Layers

This allows us to:

- Reasoning up to the equational theory R
- Express independent products of layers

$$(E, R, V_E) \otimes (F, S, V_F) := (E \uplus F, R \otimes S, V_E \bullet V_F)$$

- Express specific patterns of state sharing (e.g. lock-based)
- Express the implementation of synchronization primitives (e.g. ticket lock)

Conclusion

- We have a novel way of building models of CAL
  - No explicit state
  - Rooted in linear logic
  - Extensible
- Promising direction in compositional semantics for reasoning about large systems.
- $$\text{New CAL [POPL '22]} \xrightarrow{\text{Semantics}} \text{DeepSEA [OOPSLA '19]} \xrightarrow{\text{Compilation}} \text{CompCertO [PLDI '21]}$$

Thank you!

Conclusion

- We have a novel way of building models of CAL
  - No explicit state
  - Rooted in linear logic
  - Extensible
- Promising direction in compositional semantics for reasoning about large systems.
- New CAL [POPL '22] $\xrightarrow{\text{Semantics}}$ DeepSEA [OOPSLA '19] $\xrightarrow{\text{Compilation}}$ CompCertO [PLDI '21]

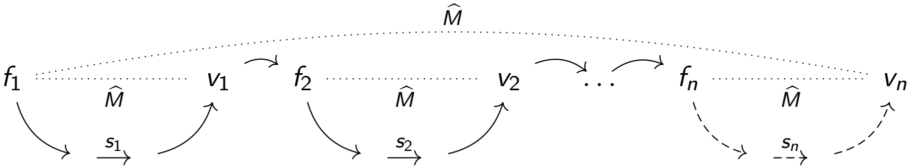
Thank you!

Layer Implementations are Regular Maps

Implementations are regular maps:

$$\widehat{M} : \dagger E \multimap \dagger F$$

Regular maps are linear maps that are "replayable":



THEOREM (REDDY)

$$\dagger E \longrightarrow_{\text{Reg}} \dagger F \cong \dagger E \multimap F$$

Layer Specifications are Stateful

$$V_{\mathsf{Counter}} = \{\epsilon, \mathsf{get}, \mathsf{inc}, \mathsf{get}\cdot 0, \mathsf{inc}\cdot\mathsf{ok}, \mathsf{inc}\cdot\mathsf{ok}\cdot\mathsf{get}, \mathsf{inc}\cdot\mathsf{ok}\cdot\mathsf{get}\cdot 1, \ldots\}$$

State as Past

$$S_E = \text{plays of } V_E \qquad \text{Initial: } \epsilon \qquad s \xrightarrow{e.v} s\cdot e\cdot v \iff s\cdot e\cdot v \in V_E$$