« Bi-inductive Structural Semantics and its Abstraction »

Patrick Cousot

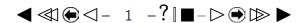
École normale supérieure 45 rue d'Ulm, 75230 Paris cedex 05, France

Patrick.Cousot@ens.fr www.di.ens.fr/~cousot

(joint work with Radhia Cousot)

Departmental Seminar — Department of Computing, Imperial College London

Wednesday July 4th, 2007



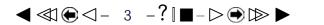
Contents

Motivation	3
Example: semantics of the eager λ -calculus	7
Bi-inductive structural definitions	47
Abstraction	63
Conclusion	66

1. Motivation







Motivation

We look for a formalism to specify abstract program semantics

from definitional semantics ...

to static program analysis algorithms

handling the many different styles of presentations found in the literature (rules, fixpoint, equations, constraints, ...) in a uniform way

 A simple generalization of inductive definitions from sets to posets seems adequate.

On the importance of defining both finite and infinite behaviors

- Example of the *choice operator* $E_1 \mid E_2$ where:

$$E_1 \Longrightarrow a$$
 $E_2 \Longrightarrow b$ termination or $E_1 \Longrightarrow \bot$ $E_2 \Longrightarrow \bot$ non-termination

- The *finite behavior* of $E_1 \mid E_2$ is:

$$a \mid b \Longrightarrow a$$
 $a \mid b \Longrightarrow b$.

- But for the case $\bot | \bot \Longrightarrow \bot$, the *infinite behaviors* of $E_1 | E_2$ depend on the choice method:

Non-deter- ministic	Parallel	Eager	Mixed left- to-right	Mixed right- to-left
$\perp \mid b \Longrightarrow b$	$oxed{\perp \mid b \Longrightarrow b}$			$ot \mid b \Longrightarrow b$
$\perp \mid b \Longrightarrow \perp$		$\perp \mid b \Longrightarrow \perp$	$ig oxedsymbol{\perp} ig b \Longrightarrow oxedsymbol{\perp}$	$ot \mid b \Longrightarrow ot$
$\mid a \mid \bot \Longrightarrow a$	$ a \perp \Longrightarrow a$		$\mid a \mid \bot \Longrightarrow a \mid$	
$ a \perp \Longrightarrow \perp$		$ a \perp \Longrightarrow \perp $	$\mid a \mid \bot \Longrightarrow \bot \mid$	$a\mid \bot\Longrightarrow \bot$

- Nondeterministic: an internal choice is made initially to evaluate E_1 or to evaluate E_2 ;
- Parallel: evaluate E_1 and E_2 concurrently, with an unspecified scheduling, and return the first available result a or b;
- Mixed left-to-right: evaluate E_1 and then either return its result a or evaluate E_2 and return its result b;
- Mixed right-to-left: evaluate E_2 and then either return its result b or evaluate E_1 and return its result a;
- Eager: evaluate both E_1 and E_2 and return either results if both terminate.



Semantics of the Eager λ -calculus

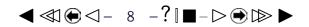
P. Cousot & R. Cousot. Bi-inductive Structural Semantics. SOS 2007, July 9, 2007, Wroclaw, Poland.





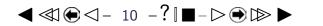


Syntax



Syntax of the Eager λ -calculus

Trace Semantics



Example I: Finite Computation

function argument
$$((\lambda x \cdot x \times x) (\lambda y \cdot y)) ((\lambda z \cdot z) 0)$$

$$\rightarrow \qquad \qquad \text{evaluate function}$$

$$((\lambda y \cdot y) (\lambda y \cdot y)) ((\lambda z \cdot z) 0)$$

$$\rightarrow \qquad \qquad \text{evaluate function, cont'd}$$

$$(\lambda y \cdot y) ((\lambda z \cdot z) 0)$$

$$\rightarrow \qquad \qquad \text{evaluate argument}$$

$$(\lambda y \cdot y) 0$$

$$\rightarrow \qquad \qquad \text{apply function to}$$

$$0 \qquad \text{a value!} \qquad \text{argument}$$

Example II: Infinite Computation

```
function argument
(\lambda \times \cdot \times \times) (\lambda \times \cdot \times \times)
\rightarrow \qquad \text{apply function to argument}
(\lambda \times \cdot \times \times) (\lambda \times \cdot \times \times)
\rightarrow \qquad \text{apply function to argument}
(\lambda \times \cdot \times \times) (\lambda \times \cdot \times \times)
\rightarrow \qquad \text{apply function to argument}
```

... non termination!

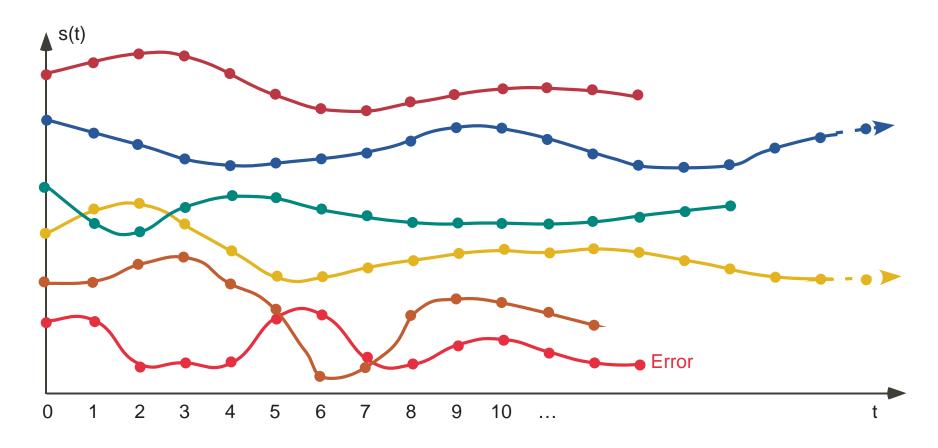


Example III: Erroneous Computation

a runtime error!



Finite, Infinite and Erroneous Trace Semantics





Traces

- $-\mathbb{T}^*$ (resp. \mathbb{T}^+ , \mathbb{T}^ω , \mathbb{T}^∞ and \mathbb{T}^∞) be the set of finite (resp. nonempty finite, infinite, finite or infinite, and nonempty finite or infinite) sequences of terms
- $-\epsilon$ is the empty sequence $\epsilon \cdot \sigma = \sigma \cdot \epsilon = \sigma$.
- $-|\sigma| \in \mathbb{N} \cup \{\omega\}$ is the length of $\sigma \in \mathbb{T}^{\infty}$. $|\epsilon| = 0$.
- $\text{ If } \sigma \in \mathbb{T}^+ \text{ then } |\sigma| > 0 \text{ and } \sigma = \sigma_0 \bullet \sigma_1 \bullet \ldots \bullet \sigma_{|\sigma|-1}.$
- If $\sigma \in \mathbb{T}^{\omega}$ then $|\sigma| = \omega$ and $\sigma = \sigma_0 \bullet \ldots \bullet \sigma_n \bullet \ldots$



Operations on Traces

- For $a \in \mathbb{T}$ and $\sigma \in \mathbb{T}^{\infty}$, we define $a@\sigma$ to be $\sigma' \in \mathbb{T}^{\infty}$ such that $\forall i < |\sigma| : \sigma'_i = a \ \sigma_i$

Example

$$- a = (\lambda y \cdot y)$$

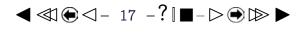
$$- \sigma = ((\lambda z \cdot z) \ 0) \cdot 0$$

$$- a@\sigma =$$

$$(\lambda y \cdot y)@((\lambda z \cdot z) \ 0) \cdot 0 =$$

$$((\lambda y \cdot y) ((\lambda z \cdot z) \ 0)) \cdot ((\lambda y \cdot y) \ 0)$$





Operations on Traces (Cont'd)

- Similarly for $a \in \mathbb{T}$ and $\sigma \in \mathbb{T}^{\infty}$, $\sigma @ a$ is σ' where $\forall i < |\sigma| : \sigma'_i = \sigma_i \ a$

Example

$$-\sigma = ((\lambda x \cdot x \times x) (\lambda y \cdot y)) \cdot ((\lambda y \cdot y) (\lambda y \cdot y)) \cdot (\lambda y \cdot y)$$

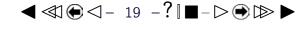
$$-b = ((\lambda z \cdot z) 0)$$

$$-(\sigma @ b)$$

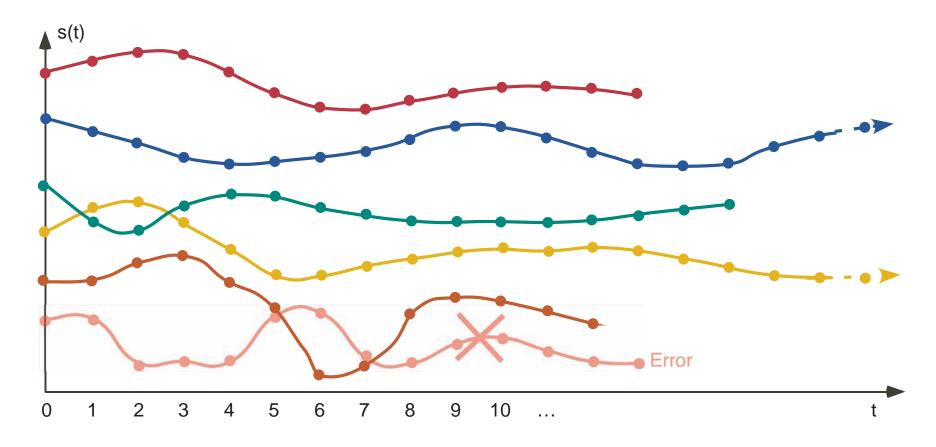
$$= (((\lambda x \cdot x \times x) (\lambda y \cdot y)) \cdot ((\lambda y \cdot y) (\lambda y \cdot y)) \cdot (\lambda y \cdot y) @ ((\lambda z \cdot z) 0))$$

$$= (((\lambda x \cdot x \times x) (\lambda y \cdot y)) ((\lambda z \cdot z) 0)) \cdot (((\lambda y \cdot y) (\lambda y \cdot y)) ((\lambda z \cdot z) 0)) \cdot ((\lambda y \cdot y) ((\lambda z \cdot z) 0))$$





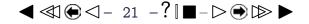
Finite and Infinite Trace Semantics





Bifinitary Trace Semantics $\vec{\mathbb{S}}$ of the Eager λ -calculus [CC92]

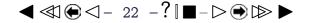
Note: $a[x \leftarrow b]$ is the capture-avoiding substitution of b for all free occurences of x within a. We let FV(a) be the free variables of a. We define the call-by-value semantics of closed terms (without free variables) $\overline{\mathbb{T}} \triangleq \{a \in \mathbb{T} \mid FV(a) = \varnothing\}.$





Bifinitary Trace Semantics \vec{S} of the Eager λ -calculus [CC92]

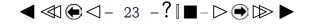
Note: $a[x \leftarrow b]$ is the capture-avoiding substitution of b for all free occurences of x within a. We let FV(a) be the free variables of a. We define the call-by-value semantics of closed terms (without free variables) $\overline{\mathbb{T}} \triangleq \{a \in \mathbb{T} \mid FV(a) = \varnothing\}.$





Bifinitary Trace Semantics \vec{S} of the Eager λ -calculus [CC92]

Note: $a[x \leftarrow b]$ is the capture-avoiding substitution of b for all free occurences of x within a. We let FV(a) be the free variables of a. We define the call-by-value semantics of closed terms (without free variables) $\overline{\mathbb{T}} \triangleq \{a \in \mathbb{T} \mid FV(a) = \varnothing\}.$

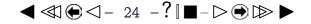




Bifinitary Trace Semantics \vec{S} of the Eager λ -calculus [CC92]

$$\begin{array}{c}
a[x\leftarrow v] \bullet \sigma \in \vec{S} \\
\hline
(\lambda x \cdot a) v \bullet a[x\leftarrow v] \bullet \sigma \in \vec{S} \\
\hline
(\lambda x \cdot a) v \bullet a[x\leftarrow v] \bullet \sigma \in \vec{S} \\
\hline
(\lambda x \cdot a) v \bullet a[x\leftarrow v] \bullet \sigma \in \vec{S} \\
\hline
(\lambda x \cdot a) v \bullet a[x\leftarrow v] \bullet \sigma \in \vec{S} \\
\hline
(\lambda x \cdot a) v \bullet a[x\leftarrow v] \bullet \sigma \in \vec{S} \\
\hline
(\lambda x \cdot a) v \bullet a[x\leftarrow v] \bullet \sigma \in \vec{S} \\
\hline
(a v) \bullet \sigma' \in \vec{S} \\
\hline
(a v) \bullet \sigma'$$

Note: $a[x \leftarrow b]$ is the capture-avoiding substitution of b for all free occurences of x within a. We let FV(a) be the free variables of a. We define the call-by-value semantics of closed terms (without free variables) $\overline{\mathbb{T}} \triangleq \{a \in \mathbb{T} \mid FV(a) = \varnothing\}.$





Non-Standard Meaning of the Rules

The rules

$$\mathcal{R} = \left\{rac{P_i}{C_i} \sqsubseteq igg| i \in \Delta
ight\}$$

define

$$\operatorname{lfp}^{\sqsubseteq}F\llbracket\mathcal{R}
rbracket$$

where the consequence operator is

$$egin{aligned} F[\![\mathcal{R}]\!](T) &= igsqcup \Big\{ C \ \Big| \ P \sqsubseteq T \wedge rac{P}{C} \sqsubseteq \in \mathcal{R} \Big\} \end{aligned}$$

and ...

The Computational Lattice

Given $S, T \in \wp(\mathbb{T}^{\infty})$, we define

$$-S^+ \triangleq S \cap \mathbb{T}^+$$

finite traces

$$-S^{\omega} \triangleq S \cap \mathbb{T}^{\omega}$$

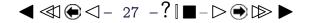
infinite traces

$$-S \sqsubseteq T \triangleq S^+ \subseteq T^+ \land S^\omega \supseteq T^\omega$$
 computational order

$$-\langle \wp(\mathbb{T}^{\infty}), \sqsubseteq, \mathbb{T}^{\omega}, \mathbb{T}^{+}, \sqcup, \sqcap \rangle$$
 is a complete lattice

Bifinitary Trace Semantics $\vec{\mathbb{S}}$ of the Eager λ -calculus [CC92]

Note: $a[x \leftarrow b]$ is the capture-avoiding substitution of b for all free occurences of x within a. We let FV(a) be the free variables of a. We define the call-by-value semantics of closed terms (without free variables) $\overline{\mathbb{T}} \triangleq \{a \in \mathbb{T} \mid FV(a) = \varnothing\}.$





Example

$$\frac{\sigma \bullet \mathsf{v} \in \vec{\mathbb{S}}^+, \ (\mathsf{a} \ \mathsf{v}) \bullet \sigma' \in \vec{\mathbb{S}}}{(\mathsf{a} @ \sigma) \bullet (\mathsf{a} \ \mathsf{v}) \bullet \sigma' \in \vec{\mathbb{S}}} \sqsubseteq, \quad \mathsf{v}, \mathsf{a} \in \mathbb{V} \ .$$

$$-\sigma \cdot \mathbf{v} = ((\lambda \mathbf{z} \cdot \mathbf{z}) \ 0) \cdot 0 \in \in \mathbb{S}^+$$

$$-(a v) \cdot \sigma' = (\lambda y \cdot y) 0 \cdot 0 \in \vec{\mathbb{S}}$$

$$-(a@\sigma) \cdot (a \lor) \cdot \sigma'$$

=

$$((\lambda y \cdot y)@((\lambda z \cdot z) 0) \cdot 0)$$

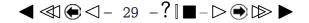
=

$$(\lambda y \cdot y) ((\lambda z \cdot z) 0) \cdot (\lambda y \cdot y) 0 \cdot 0 \in \vec{S}$$



Bifinitary Trace Semantics $\vec{\mathbb{S}}$ of the Eager λ -calculus [CC92]

Note: $a[x \leftarrow b]$ is the capture-avoiding substitution of b for all free occurences of x within a. We let FV(a) be the free variables of a. We define the call-by-value semantics of closed terms (without free variables) $\overline{\mathbb{T}} \triangleq \{a \in \mathbb{T} \mid FV(a) = \varnothing\}.$





Example

$$\frac{\sigma \bullet \mathsf{v} \in \vec{\mathbb{S}}^+, \ (\mathsf{v} \ \mathsf{b}) \bullet \sigma' \in \vec{\mathbb{S}}}{(\sigma @ \mathsf{b}) \bullet (\mathsf{v} \ \mathsf{b}) \bullet \sigma' \in \vec{\mathbb{S}}} \sqsubseteq, \quad \mathsf{v} \in \mathbb{V}$$

$$-\sigma \bullet v = ((\lambda x \cdot x \times x) (\lambda y \cdot y)) \bullet ((\lambda y \cdot y) (\lambda y \cdot y)) \bullet (\lambda y \cdot y) \in \vec{\mathbb{S}}^{+}$$

$$-(v b) \bullet \sigma' = (\lambda y \cdot y) ((\lambda z \cdot z) 0) \bullet (\lambda y \cdot y) 0 \bullet 0 \in \vec{\mathbb{S}}$$

$$-(\sigma @ b) \bullet (v b) \bullet \sigma'$$

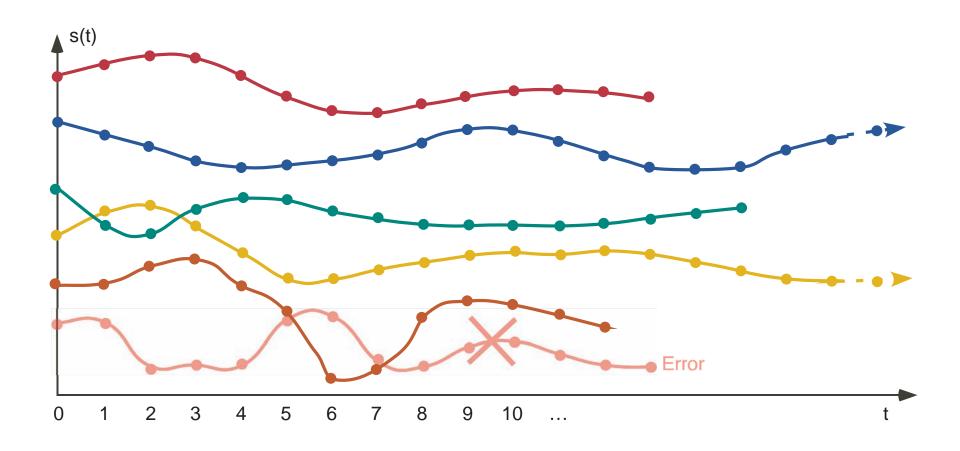
$$= (((\lambda x \cdot x \times x) (\lambda y \cdot y)) \bullet ((\lambda y \cdot y) (\lambda y \cdot y)) @ ((\lambda z \cdot z) 0)) \bullet ((\lambda y \cdot y) ((\lambda z \cdot z) 0)) \bullet ((\lambda y \cdot y) ((\lambda z \cdot z) 0)) \bullet ((\lambda y \cdot y) ((\lambda z \cdot z) 0))$$

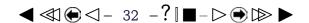
$$= ((\lambda x \cdot x \times x) (\lambda y \cdot y)) ((\lambda z \cdot z) 0) \bullet ((\lambda y \cdot y) (\lambda y \cdot y)) ((\lambda z \cdot z) 0)$$

$$\bullet (\lambda y \cdot y) ((\lambda z \cdot z) 0) \bullet (\lambda y \cdot y) 0 \bullet 0 \in \vec{\mathbb{S}}$$

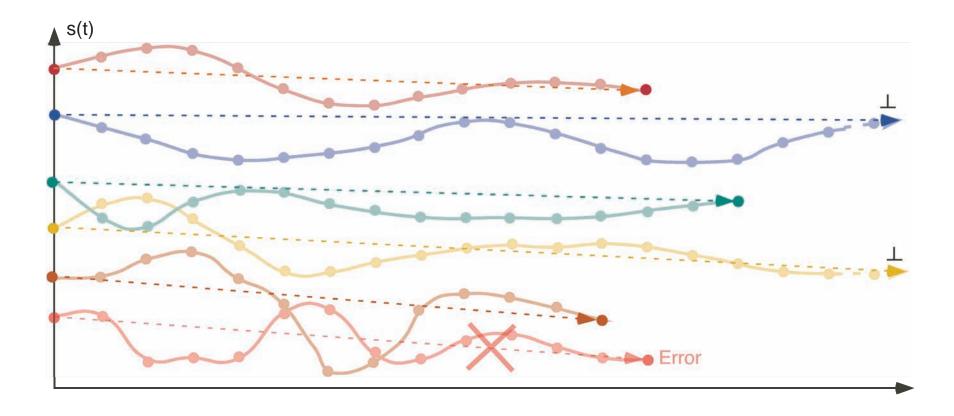
Relational Semantics

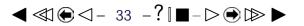
Trace Semantics



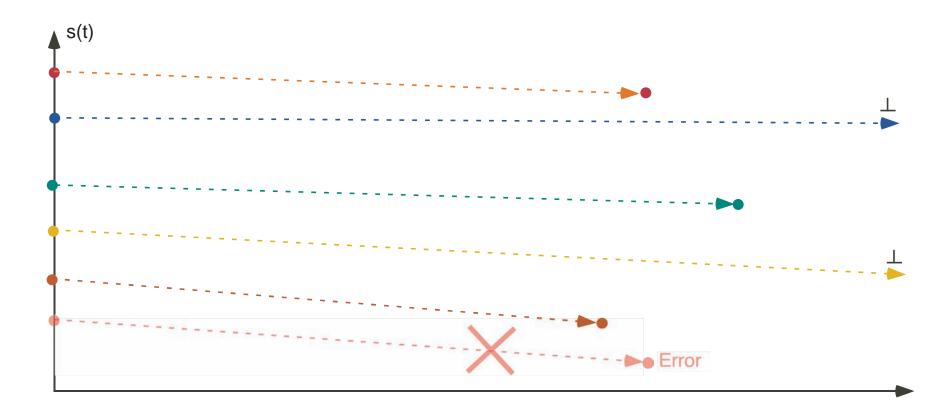


Relational Semantics = α (Trace Semantics)





Relational Semantics





Abstraction to the Bifinitary Relational Semantics of the Eager λ -calculus

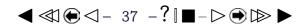
remember the input/output behaviors, forget about the intermediate computation steps

$$lpha(T) \stackrel{ ext{def}}{=} \{lpha(\sigma) \mid \sigma \in T\}$$
 $lpha(\sigma_0 ullet \sigma_1 ullet \dots ullet \sigma_n) \stackrel{ ext{def}}{=} \langle \sigma_0, \ \sigma_n
angle$ $lpha(\sigma_0 ullet \dots ullet \sigma_n ullet \dots) \stackrel{ ext{def}}{=} \langle \sigma_0, \ oldsymbol{\perp}
angle$

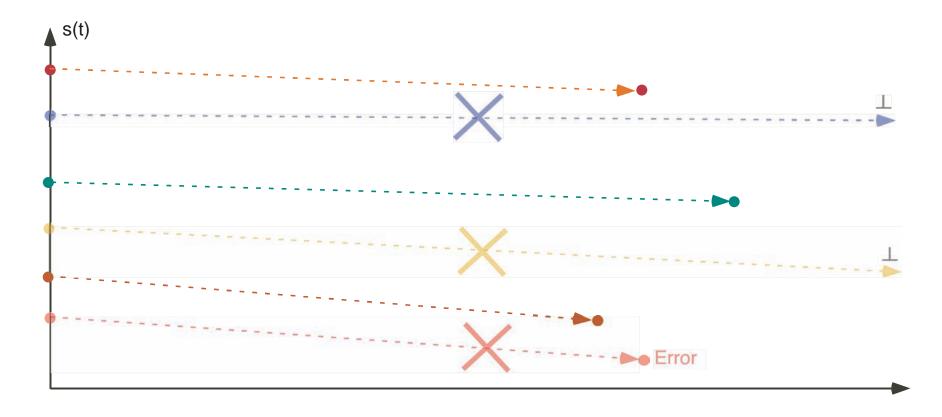
Bifinitary Relational Semantics of the Eager λ -calculus

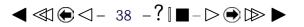
$$\begin{array}{c} \mathsf{v} \Rightarrow \mathsf{v}, \quad \mathsf{v} \in \mathbb{V} \\ \hline \mathsf{a} \Rightarrow \bot \\ \hline \mathsf{a} \mathsf{b} \Rightarrow \bot \\ \hline \\ \mathsf{c}, \quad \mathsf{v} \in \mathbb{V}, \quad r \in \mathbb{V} \cup \{\bot\} \\ \hline \\ \mathsf{a} \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{v} \mathsf{b} \Rightarrow r \\ \hline \\ \mathsf{a} \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{v} \mathsf{b} \Rightarrow r \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{a} \mathsf{v} \Rightarrow r \\ \hline \\ \mathsf{a} \mathsf{b} \Rightarrow \mathsf{r} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{a} \mathsf{v} \Rightarrow r \\ \hline \\ \mathsf{a} \mathsf{b} \Rightarrow \mathsf{r} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{a} \mathsf{v} \Rightarrow r \\ \hline \\ \mathsf{a} \mathsf{b} \Rightarrow \mathsf{r} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{a} \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v}, \quad \mathsf{b} \Rightarrow \mathsf{v} \\ \hline \\ \mathsf{b} \Rightarrow \mathsf{v} \\$$

Natural Semantics



Natural Semantics = α (Relational Semantics)





Abstraction to the Natural Big-Step Semantics of the Eager λ -calculus

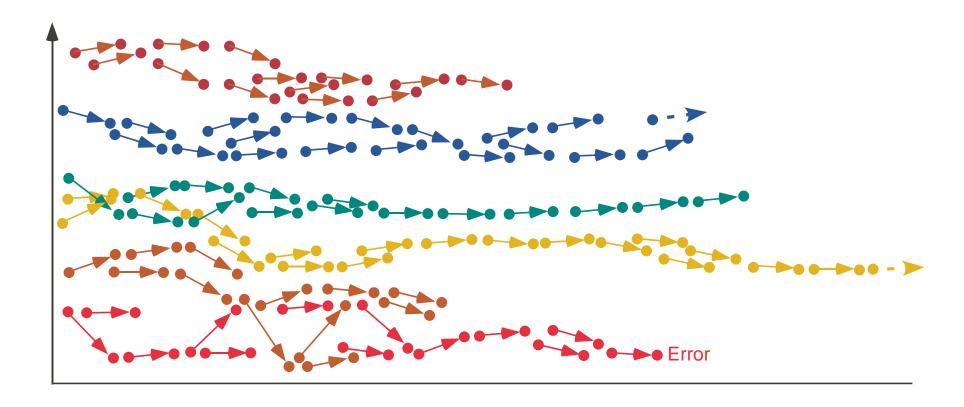
remember the finite input/output behaviors, forget about non-termination

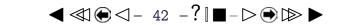
Natural Big-Step Semantics of the Eager λ -calculus [Kah88]

$$egin{aligned} \mathbf{v} &\Longrightarrow \mathbf{v}, \quad \mathbf{v} \in \mathbb{V} \ & \dfrac{\mathbf{a}[\mathbf{x} \leftarrow \mathbf{v}] \Longrightarrow r}{(oldsymbol{\lambda} \mathbf{x} \cdot \mathbf{a}) \quad \mathbf{v} \Longrightarrow r} \subseteq, \quad \mathbf{v} \in \mathbb{V}, \ r \in \mathbb{V} \ & \dfrac{\mathbf{a} \Longrightarrow \mathbf{v}, \quad \mathbf{v} \ \mathbf{b} \Longrightarrow r}{\mathbf{c}} \subseteq, \quad \mathbf{v} \in \mathbb{V}, \ r \in \mathbb{V} \ & \dfrac{\mathbf{b} \Longrightarrow \mathbf{v}, \quad \mathbf{a} \ \mathbf{v} \Longrightarrow r}{\mathbf{a} \ \mathbf{b} \Longrightarrow r} \subseteq, \quad \mathbf{a} \in \mathbb{V}, \ \mathbf{v} \in \mathbb{V}, \ r \in \mathbb{V} \ . \end{aligned}$$

Transition Semantics

Transition Semantics = α (Trace Semantics)





Abstraction to the Transition Semantics of the Eager λ -calculus

remember execution steps, forget about their sequencing

$$egin{aligned} lpha(T) \stackrel{ ext{def}}{=} igcup \{lpha(\sigma) \mid \sigma \in T\} \ & lpha(\sigma_0 ullet \sigma_1 ullet \ldots ullet \sigma_n) \stackrel{ ext{def}}{=} \{\langle \sigma_i, \ \sigma_{i+1}
angle \mid 0 \leqslant i \land i < n\} \ & lpha(\sigma_0 ullet \ldots ullet \sigma_n ullet \ldots) \stackrel{ ext{def}}{=} \{\langle \sigma_i, \ \sigma_{i+1}
angle \mid i \geqslant 0\} \end{aligned}$$

Transition Semantics of the Eager λ -calculus [Plo81]

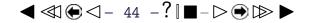
$$((\lambda \times \cdot a) \vee) \longrightarrow a[x \leftarrow v]$$

$$a_0 \longrightarrow a_1$$

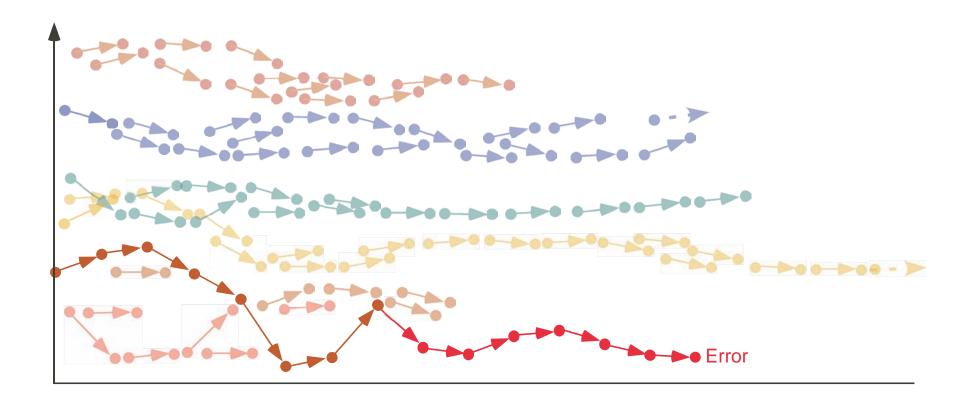
$$a_0 \longrightarrow a_1 \longrightarrow a_1$$

$$b_0 \longrightarrow b_1$$

$$v \mapsto b_0 \longrightarrow v \mapsto b_1$$



Approximation



$$\frac{((\lambda x \cdot x \cdot x) ((\lambda z \cdot z) \cdot 0)) (\lambda y \cdot y) \rightarrow ((\lambda x \cdot x \cdot x) \cdot 0) (\lambda y \cdot y)}{\rightarrow (0 \cdot 0) (\lambda y \cdot y)}$$
 an error!



The Abstract Semantics are Correct by Calculational Design

$v \mapsto v, v \in \mathbb{V}$	$a[x \leftarrow v] \Longrightarrow \sigma$ $v \in V$	
	$(\lambda \times a) \vee \mapsto (\lambda \times a) \vee \sigma$	
	$\mathbf{a} \mapsto \sigma \cdot \mathbf{v}, \mathbf{v} \cdot \mathbf{b} \mapsto \sigma'$ $\mathbf{a} \cdot \mathbf{b} \mapsto (\sigma \otimes \mathbf{b}) \cdot \sigma'$ $\mathbf{v} \in V, \sigma \in T^+$	
$\begin{array}{c} b \longmapsto \sigma \\ \hline \vdots \\ \end{array} ; \mathbf{a} \in \mathbb{V}, \sigma \in T^\omega \end{array} .$	$b \mapsto \sigma \cdot v$, $a v \mapsto \sigma'$ $a b \mapsto (\pi \otimes \sigma) \cdot \sigma'$ $a b \mapsto (\pi \otimes \sigma) \cdot \sigma'$ $a \mapsto a \cdot v \in V, \sigma \in T^+$.	
a b ⊨⇒ a@σ	$a b \mapsto (a @ \sigma) \cdot \sigma'$	
5.4 Abstraction into the big-step calculus	relational semantics of the call-by-value λ -	
5.4.1 Relational abstraction of tr	aces	
The relational abstraction of sets of	of traces is	
$\alpha \in \rho(\mathbb{T}^{\infty}) \mapsto \rho(\mathbb{T} \times (\mathbb{T} \cup \{\bot\}))$	·})) (4)	
	$ =n\} \cup \{\langle \sigma_0, \perp \rangle \mid \sigma \in S \land \sigma = \omega\}$	
$\gamma \in \wp(\mathbb{T} \times (\mathbb{T} \cup \{\bot\})) \mapsto \wp(\mathbb{T} \cup \{\bot\})$		
	$, \sigma_{n-1} \rangle \in T) \vee (\sigma = \omega \wedge \langle \sigma_0, \perp \rangle \in T) \}$	
so that		
$\langle \wp(\mathbb{T}^{\infty}), \subseteq \rangle \stackrel{\gamma}{\longleftarrow_{\alpha}}$	$\equiv \langle \rho(\mathbb{T} \times (\mathbb{T} \cup \{\bot\})), \subseteq \rangle$. (5)	
Proof		
$\alpha(S) \subseteq T$		
	$i\} \cup \{ \langle \sigma_0, \perp \rangle \mid \sigma \in S \land \sigma = \omega \} \subseteq T$	
	(def. α)	
\iff $\forall \sigma \in S^+ : \langle \sigma_0, \sigma_{ \sigma -1} \rangle \in T^+$	$\land \forall \sigma \in S : (\sigma_0, \perp) \in I$ $\{\text{def. } \subseteq, S^+ \triangleq S \cap T^+, \text{ and } S^{\omega} \triangleq S \cap T^{\omega}\}$	
$\iff S^+ \subseteq \{\sigma \mid \sigma = n \wedge \langle \sigma_0, \sigma_{n-1} \rangle$	$_1\rangle \in T\} \wedge S^{\omega} \subseteq \{\sigma \mid \sigma = \omega \wedge \langle \sigma_0, \perp \rangle \in T\}$	
	18	
\hat{l} def. \subseteq , T^+	$\triangleq T \cap (\mathbb{T} \times \mathbb{T})$, and $T^{\omega} \triangleq T \cap (\sigma \mathbb{T} \times \{\bot\})$	
\iff $S \subseteq \gamma(T)$	$(S = S^+ \cup S^\omega \text{ and def. } \gamma(T))$	
CAR DICK TO THE		
5.4.2 Bifinitary relational seman	$\alpha : S \triangleq \alpha(\overline{S}) \in p(\mathbb{T} \times (\mathbb{T} \cup \{\bot\})) \text{ is the}$	
relational abstraction of the trace final value or \perp in case of diverger	e semantics mapping an expression to its	
5.4.3 Fixpoint big-step bifinitary	relational semantics	
The bifinitary relational semantics fixpoint form as $\mathbf{lfp}^{\mathbb{G}} \vec{F}$ where the b $\wp(\mathbb{T} \times (\mathbb{T} \cup \{\bot\}))$ is	$s \ \stackrel{<}{=} \ \stackrel{<}{=} \ \alpha(\overrightarrow{\mathbb{S}}) = \alpha(\operatorname{Ifp}^{\subseteq} \overrightarrow{F})$ can be defined in ig-step transformer $\overrightarrow{F} \in \wp(\mathbb{T} \times (\mathbb{T} \cup \{\bot\})) \mapsto$	
$\vec{F}(T) \triangleq \{\langle v, v \rangle \mid v \in V\} \cup$	(6)	
	$r \in V \land (a[x \leftarrow v], r) \in T\} \cup$ (6)	
$\{\langle (a\ b),\ ot angle \ \ \langle a,\ ot$		
	$\in T^+ \wedge \mathbf{v} \in \mathbb{V} \wedge \langle (\mathbf{v} \ \mathbf{b}), \ r \rangle \in T \} \cup \\$	
$\{\langle (a\ b),\ \bot\rangle\mid a\in \mathbb{N} \}$ $\{\langle (a\ b),\ r\rangle\mid a,v\in \mathbb{N} \}$	$\langle \wedge \langle \mathbf{b}, \perp \rangle \in T \} \cup$ $\forall \wedge \langle \mathbf{b}, \mathbf{v} \rangle \in T^{+} \wedge \langle (\mathbf{a} \mathbf{v}), r \rangle \in T \}$.	
Lemma 10 $\alpha(\vec{F}(S)) = \vec{F}(\alpha(S))$		
	sm, so we calculate $\alpha(\vec{F}(S))$ by cases.	
$\alpha(\{v \in \mathbb{T}^{\infty} \mid v \in \mathbb{V}\})$		
$= \ \{\langle v,v\rangle \mid v \in \mathbb{V}\}$	(def. α and $ \mathbf{v} =1$)	
$= \alpha(\{(\lambda \times \cdot a) \times \cdot a[x \leftarrow v] \cdot \sigma \mid v \in \alpha(\{(\lambda \times \cdot a) \times \cdot a[x \leftarrow v] \cdot \sigma \mid v \in a[x \leftarrow v] \cdot \sigma $	$V \land a[x \leftarrow v] \cdot \sigma \in S^{+}) \cup \alpha(\{(\lambda x \cdot a) v \cdot a[x \leftarrow$	
$v] \cdot \sigma \mid v \in V \land a[x \leftarrow v] \cdot \sigma \in S$	$^{\omega}$ }) $_{\tilde{\zeta}}(S = S^+ \cup S^{\omega} \text{ and } \alpha \text{ preserves lubs}_{\tilde{\zeta}})$	
= $\{\langle (\mathbf{\lambda} \times \mathbf{\cdot} \mathbf{a}) \mathbf{v}, r \rangle \mid \mathbf{v} \in V \land \langle \mathbf{a} \mathbf{x} \cdot \mathbf{v} \rangle \land \langle \mathbf{a} \mathbf{x} \leftarrow \mathbf{v} \rangle, \perp \rangle \in \alpha(S)^{\omega} \}$	$\leftarrow v$, r $\rangle \in a(S)^+$ $\} \cup \{\langle (\lambda x \cdot a) v, \bot \rangle \mid v \in$	
	(def. α)	
$= \{ \langle (\mathbf{A} \times \mathbf{a}) \ \mathbf{v}, \ r \rangle \mid \mathbf{v} \in \mathbb{V} \land \langle \mathbf{a} \mathbf{x} + \mathbf{v} \rangle \}$ $\hat{\mathbf{c}} \text{ def. } T$	$- v , r \in \alpha(S)$ $+ \triangleq T \cap (\mathbb{T} \times \mathbb{T}) \text{ and } T^{\omega} \triangleq T \cap (\mathbb{T} \times \{\bot\})$	
	19	

```
= \{\langle (\sigma_0 | \mathbf{b}), \perp \rangle \mid \sigma \in S^{\omega} \}
                                                                                                                                                                                                                                                                   7def. α and @ i
                                                                                                                                                                                                                                      lS \subseteq \mathbb{T}^{\infty} \text{ so } \sigma_0 \in \mathbb{T}^{\mathbb{N}}
      = \{\langle (a \ b), \ \bot \rangle \mid \langle a, \ \bot \rangle \in \alpha(S) \}
      = \alpha(\{(\sigma \otimes \mathbf{b}) \cdot (\mathbf{v} \ \mathbf{b}) \cdot \sigma' \mid \sigma \cdot \mathbf{v} \in S^+ \wedge \mathbf{v} \in V \wedge (\mathbf{v} \ \mathbf{b}) \cdot \sigma' \in S\})
    =\begin{array}{ll} &\alpha(\{(\sigma \otimes \mathbf{b}).(\mathbf{v} \ \mathbf{b}).\sigma' \ | \ \sigma.\mathbf{v} \in S^+ \land \mathbf{v} \in \mathbb{V} \land (\mathbf{v} \ \mathbf{b}).\sigma' \in S^+ \}) \cup \alpha(\{(\sigma \otimes \mathbf{b}).(\mathbf{v} \ \mathbf{b}).\sigma' \ | \ \sigma.\mathbf{v} \in S^+ \land \mathbf{v} \in \mathbb{V} \land (\mathbf{v} \ \mathbf{b}).\sigma' \in S^- \})\\ &(S=S^+ \cup S^o \ \text{and} \ \alpha \ \text{preserves lubs}) \end{array}
      = \{ ((\sigma_0 \mathsf{\,b}), r) \mid \sigma \cdot \mathsf{v} \in S^+ \land \mathsf{v} \in \forall \land (\langle \mathsf{v} \mathsf{\,b}), r \rangle \in \alpha(S)^+ \} \cup \{ (\langle \sigma \mathsf{\,b}), \bot \rangle \mid \sigma \cdot \mathsf{v} \in S^+ \land \mathsf{v} \in \forall \land (\langle \mathsf{v} \mathsf{\,b}), \bot \rangle \mid \sigma \cdot \mathsf{v} \in S^+ \land \mathsf{v} \in \forall \land (\langle \mathsf{v} \mathsf{\,b}), \bot \rangle \in \alpha(S)^+ \} 
      = \{ \langle (\sigma_0 \mathsf{b}), r \rangle \mid (\sigma_0, \mathsf{v}) \in \alpha(S)^+ \land \mathsf{v} \in V \land ((\mathsf{v} \mathsf{b}), r) \in \alpha(S) \}
                                                                                                 (def. T^+ \triangleq T \cap (\mathbb{T} \times \mathbb{T}), T^{\omega} \triangleq T \cap (\mathbb{T} \times \{\bot\}), and \alpha)
    = \{ \langle (\mathbf{a} \ \mathbf{b}), \, r \rangle \mid \langle \mathbf{a}, \, \mathbf{v} \rangle \in \alpha(S)^+ \wedge \mathbf{v} \in \mathbb{V} \wedge \langle (\mathbf{v} \ \mathbf{b}), \, r \rangle \in \alpha(S) \}
    — α({a@σ | a ∈ V ∧ σ ∈ S<sup>ω</sup>})
    = \{\langle (a \sigma_0), \perp \rangle \mid a \in V \land \sigma \in S^{\omega} \}
         = \ \{ \langle (\mathbf{a} \ \sigma_0), \ \bot \rangle \mid \mathbf{a} \in \mathbb{V} \land \langle \sigma_0, \ \bot \rangle \in \alpha(S) \} \ \text{$\widehat{\mathbf{c}}$ def. $\alpha$ and $T^\omega \triangleq T \cap (\mathbb{T} \cup \{\bot\})$} \}
    = {((a b), ⊥) | a ∈ V ∧ (b, ⊥) ∈ α(S)}
γ (a b), ⊥ | a ∈ V ∧ (b, ⊥) ∈ α(S)}
γ (a b), ⊥ | a ∈ V ∧ (b, ⊥) ∈ α(S)}
      \alpha(\{(a@\sigma),(av),\sigma'|a,v\in V \land \sigma,v\in S^+ \land (av),\sigma'\in S\})
      = \alpha(\{(\mathbf{a}\otimes\sigma)\cdot(\mathbf{a}\ \mathbf{v})\cdot\sigma'\mid \mathbf{a},\mathbf{v}\in\mathbb{V}\wedge\sigma\cdot\mathbf{v}\in S^+\wedge(\mathbf{a}\ \mathbf{v})\cdot\sigma'\in S^+\})\cup\alpha(\{(\mathbf{a}\otimes\sigma)\cdot(\mathbf{a}\ \mathbf{v})\cdot\sigma'\mid \mathbf{a},\mathbf{v}\in\mathbb{V}\wedge\sigma\cdot\mathbf{v}\in S^+\wedge(\mathbf{a}\ \mathbf{v})\cdot\sigma'\in S^+\})}\\ (\mathbf{a}\ \mathbf{v})\cdot\sigma'\mid \mathbf{a},\mathbf{v}\in\mathbb{V}\wedge\sigma\cdot\mathbf{v}\in S^+\wedge(\mathbf{a}\ \mathbf{v})\cdot\sigma'\in S^+\rangle)\\ (S=S^+\cup S^\circ\ \mathrm{and}\ \alpha\ \mathrm{preserves}\ \mathrm{lubs})
    = \{((\mathbf{a} \circ \sigma_0), r) \mid \mathbf{a}, \mathbf{v} \in \mathbb{V} \land (\sigma_0, \mathbf{v}) \in \alpha(S)^+ \land ((\mathbf{a} \ \mathbf{v}), r) \in \alpha(S)^+\} \cup \{((\mathbf{a} \circ \sigma_0), \mathbf{v}) \in \alpha(S)^+ \land ((\mathbf{a} \ \mathbf{v}), \bot) \in \alpha(S)^+\} \cup \{((\mathbf{a} \circ \sigma_0), \mathbf{v}) \in \alpha(S)^+ \land ((\mathbf{a} \ \mathbf{v}), \bot) \in \alpha(S)^+\}
      = \{\langle (a b), r \rangle \mid a, v \in V \land \langle b, v \rangle \in \alpha(S) \land \langle (a v), r \rangle \in \alpha(S) \}
                                                                                                                   T^{\omega} \triangleq T \cap (\mathbb{T} \cup \{\bot\}) \text{ and } S \subseteq \mathbb{T}^{\infty} \text{ so } \sigma_0 \in \mathbb{T}
      Hence, we have the commutation property \alpha(\vec{F}(S)) = \vec{F}(\alpha(S)) when defining
      Theorem 11 \widetilde{S} \triangleq \alpha(\widetilde{S}) = \alpha(y_p^{-c} \widetilde{F}) = y_p^{-c} \widetilde{F}.

PROOF By the fixpoint fusion theorem [7, Th. 9] and the asynchronous fix-
         PROOF By the Exponen use on incorem [7, 11. 3] and the separation of Equation point iteration theorem [8, Th. 3.3.10] for \overline{S}^c, the fixpoint definition of \overline{S} can be written in the form (S^+\triangleq S\cap (\mathbb{T}\times\mathbb{T}), S^o\triangleq S\cap (\mathbb{T}\times\{\bot\}) so S^+\cap S^o=\varnothing)
                           \begin{cases} \widetilde{\mathbb{S}} &= \widetilde{\mathbb{S}}^+ \cup \widetilde{\mathbb{S}}^{\omega} \\ \widetilde{\mathbb{S}}^+ &= \widetilde{F} \left( \widetilde{\mathbb{S}}^+ \right) = \operatorname{lfp}^{\mathbb{S}} \widetilde{F}^+ & \text{where} \quad \widetilde{F}^+(S) \triangleq \widetilde{F} \left( S^+ \right) \end{cases}
                             \mathbb{S}^{\omega} = F'(\mathbb{S}^+ \cup \mathbb{S}^{\omega}) = \mathbf{gfp}^{\mathbb{C}} \, F^{\omega} \quad \text{where} \quad F^{\omega}(S) \triangleq F'(\mathbb{S}^+ \cup S^{\omega}) \; .
      We have \alpha(\vec{S}) = \alpha(\vec{S}^+ \cup \vec{S}^\omega) = \alpha(\vec{S}^+) \cup \alpha(\vec{S}^\omega) and prove that \alpha(\vec{S}^+) = \vec{S}^+ and \alpha(\vec{S}^-) = \vec{S}^- so \alpha(\vec{S}) = \vec{S}^+ \cup \vec{S}^\omega = \vec{S}.
  To prove that \alpha(\tilde{\mathbb{S}}^+) = \alpha(\mathbf{Hp}^{\mathbb{C}}\,\tilde{F}^+) is equal to \mathbf{Hp}^{\mathbb{C}}\,\tilde{F}^+ = \tilde{\mathbb{S}}^+, we observe that a preserves \cup and \alpha \circ \tilde{F}^+ = \tilde{F}^+ \circ \alpha by Lem. 10 so \alpha(\mathbf{Hp}^{\mathbb{C}}\,\tilde{F}^+) = \mathbf{Hp}^{\mathbb{C}}\,\tilde{F}^+ by [7, \text{ Th. } 3].
         We must prove that \alpha(\vec{\mathbb{S}}^\omega) = \alpha(\mathbf{gfp}^{\subseteq} \vec{F}^\omega) is equal to \mathbf{gfp}^{\subseteq} \vec{F}^\omega = \vec{\mathbb{S}}^\omega.
we must prove that \alpha(S)^{-1} = \alpha(g_0^{-1} F)^{-1} is equal to g_0^{-1} F = 2G_0^{-1} F. To prove that \alpha(g_0^{-1} F)^{-1} = g_0^{-1} F^{-1} when k^{-1} + G = 0 and \overline{K}^{-1}, \delta \in O be the respective transfinite interates of F^{0} and F^{0} from X^{0} = F^{0} and F^{0}. If (A) by the A of X^{0} = F^{0} and X^{0} = F^{0}
           To prove that gfp<sup>□</sup> F̄<sup>ω</sup> ⊆ α(gfp<sup>□</sup> F̄<sup>ω</sup>), we show that ∀(a, ⊥) ∈ gfp<sup>□</sup> F̄<sup>ω</sup>
         \exists \sigma \in gfp^{\mathbb{S}} \vec{F}^{\omega} : \sigma_0 = a. To do so for any (a, \perp) \in gfp^{\mathbb{S}} \vec{F}^{\omega}, we prove by transfinite induction on \delta that
                           \forall \delta \in \mathcal{O} > 0: \forall \langle \mathsf{a}, \ \bot \rangle \in \mathsf{gfp}^{^{\mathbb{G}}} \, \overrightarrow{F}^{^{\omega}}: \exists \sigma \in \mathbb{T}^{\omega}: \sigma_0 = \mathsf{a} \wedge \sigma \in \bigcap \, X^{\beta} \, .
      Assume by induction hypothesis, that \exists \sigma \in \Gamma^\omega : \sigma_0 = a \wedge \forall \eta \in \Omega : 0 < \eta < \delta : \sigma \in \bigcap_{1 \leq c_0} X^\beta. We have \sigma \in \bigcap_{1 \leq c_0} \bigcap_{1 \leq c_0} X^\beta = \bigcap_{1 \leq c_0} X^\beta : dv we must show that \exists \sigma \in \Gamma^\omega : \sigma_0 = a \wedge \sigma \in X^\delta = \bar{F}^\omega \cap \bigcap_{1 \leq c_0} X^\beta. Because the iterates X^\delta, \delta \in O are decreasing, this implies \exists \sigma \in \Gamma^\omega : \sigma_0 = a \wedge \sigma \in \bigcap_{1 \leq c_0} X^\beta.
    It remains to show, by structural case analysis on a, that if \sigma \in S: \sigma_0 = a.
```

```
then \exists \sigma' \in \tilde{F}(S) : \sigma'_0 = a where S = \bigcap_{S \in S} X^S.

— If a \in V then (a, \perp) \notin \mathfrak{g} \mathfrak{h}^{\Sigma} \tilde{F}''.

— If a = (\lambda x \cdot x') \vee_V \vee_S \in V then (a, \perp) \in \mathfrak{g} \mathfrak{h}^{\Sigma} \tilde{F}'' = \tilde{F}''(\mathfrak{g} \mathfrak{h}^{\Sigma} \tilde{F}'') so by (6), (\sigma' | x - v'), \perp 1 \in \mathfrak{g} \mathfrak{h}^{\Sigma} \tilde{F}''. By induction on \delta, we have \exists \sigma' \in \mathbb{F}^{\times} : \sigma'_0 = \pi' | x - v' | of \sigma' \in \mathbb{F}_{0,N} X^S so that, by (0), (\lambda x \cdot y | x') = (-v') \sigma' \in \mathbb{F}_{0,N} X^S is (\beta x \cdot y | x') = (-v') \sigma' \in \mathbb{F}_{0,N} X^S).
```

— If $(s', \perp) \in \operatorname{glp}^{\circ} \overrightarrow{F} \subseteq \bigcap_{\beta < \delta} X^{\beta}$ then, by induction hypothesis on δ , we have $\exists \sigma' \in \mathbb{T}^{\omega} : \sigma'_{\delta} = a' \wedge \sigma' \in \bigcap_{\beta < \delta} X^{\beta}$ so that, by (c), σ' 0b $\in \overrightarrow{F}(\bigcap_{\beta < \delta} X^{\beta}) = X^{\beta}$ is such that $\sigma'_{\delta} = (s', b) = a$ by definition of 0.

— If $(p', \psi) \in \mathbb{S}^+ = \alpha(\mathbb{S}^+)$, $\psi \in \mathbb{V}$, and $((\psi \mathbf{b}), \bot) \in \mathfrak{glp}^+ \widetilde{F}^{\omega}$ then, by induction hypothesis on δ , we have $\widetilde{\sigma} \in \mathbb{T}^+ : \mathfrak{g}_{\sigma} = (\psi \mathbf{b}) \wedge \sigma' \in \Gamma_{be,\sigma} X^* \cup \mathbb{B}$ definition (d) δ , there exists $\in \mathbb{T}^+ : \varepsilon \in \mathbb{S}^+ \wedge \mathbb{K} = n \wedge (\mathfrak{g}_{\sigma}, \mathfrak{g}_{\sigma}) = (\mathcal{Y}, \psi)$ princing by definition (d) of \widetilde{F} that $\widetilde{\sigma}^+ = (\mathfrak{C} \mathfrak{b})_F \circ \widetilde{F} (\Gamma_{be,\sigma} X^0) = X^2$ where, by definition, $(\varepsilon + \varepsilon, -1) \in \mathbb{K}$ and $(\varepsilon + \varepsilon, -1) \in \mathbb{K}$ and $(\varepsilon + \varepsilon, -1) \in \mathbb{K}$.

■ If $a' \in \mathbb{V}$ and $(b, \perp) \in \mathfrak{gfp}^{G} \tilde{F}^{\omega'}$ then by induction hypothesis on δ , $\exists \sigma' \in \mathbb{T}^{\omega} : \sigma_0 = b \land \sigma' \in \Pi_{S \subset S} X^{S}$ proving by definition (c) of \tilde{F} that $\sigma = a' \otimes \sigma' \in \tilde{F}$ ($\Pi_{S \subset S} X^{S}) = X^{S}$ with $\sigma_0 = (a' \otimes \sigma')_0 = (a' \sigma'_0) = (a' b) = a$.

If $\mathbf{x}', \mathbf{v} \in \mathbb{V}$, $(\mathbf{b}, \mathbf{v}) \in \widehat{\mathbb{S}}^+ = \alpha(\widehat{\mathbb{S}}^+)$, and $\langle (\mathbf{z}', \mathbf{v}), \perp \rangle \in g\mathbf{f}^{\circ}\widehat{F}''$ then, by induction hypothesis on δ , we have $\widehat{\mathbf{b}}' \in \Gamma^* : \phi_{\delta}' = \langle \mathbf{z}', \mathbf{v} \rangle \wedge \sigma' \in \Gamma_{b,o} X^{\delta}$. By definition (4) of α , there exists $\mathbf{c} \in \Gamma^* : \mathbf{c} \in \widehat{\mathbb{S}}^+ \wedge |\mathbf{c}| = n \wedge \langle \mathbf{o}_0, \mathbf{c}_{n-1} \rangle = \langle \mathbf{b}, \mathbf{v} \rangle$, proving by definition (4) of \widehat{F} that $(\mathcal{D}^0\mathbf{c}_i) \cdot \mathbf{r}' \in \widehat{F}(\Gamma_{b,o} X^{\delta}) = X^{\delta}$ with $\sigma_0 = (\mathcal{D}^0\mathbf{c}_i)_0 = (\mathcal{D}^*\mathbf{c}_i)_0 = (\mathcal{D}^*\mathbf$

5.4.4 Rule-based biq-step bifinitary relational semantics

The big-step bifinitary relational semantics \Rightarrow is defined as $\mathbf{a} \Rightarrow r \triangleq \langle \mathbf{a}, r \rangle \in \alpha(S[\mathbf{a}])$ where $\mathbf{a} \in \mathbb{T}$ and $r \in \mathbb{T} \cup \{\bot\}$. It is

$$\begin{array}{lll} \mathbf{v} \Rightarrow \mathbf{v}, & \mathbf{v} \in \mathbb{V} & = \frac{\mathbf{a}[\mathbf{v} - \mathbf{v}] \Rightarrow \mathbf{r}}{(\mathbf{A}\mathbf{x} \cdot \mathbf{a}) \mathbf{v} \Rightarrow \mathbf{r}}, & \mathbf{v} \in \mathbb{V}, \ \mathbf{r} \in \mathbb{V} \cup \{\bot\} \\ \\ \mathbf{a} \Rightarrow \mathbf{b} \Rightarrow \bot & = \frac{\mathbf{a} \Rightarrow \mathbf{v}, \quad \mathbf{v} \ \mathbf{b} \Rightarrow \mathbf{r}}{\mathbf{a} \ \mathbf{b} \Rightarrow \mathbf{r}}, & \mathbf{v} \in \mathbb{V}, \ \mathbf{r} \in \mathbb{V} \cup \{\bot\} \\ \\ \\ \mathbf{b} \Rightarrow \mathbf{b} \Rightarrow \bot & = \mathbf{a} \Rightarrow \mathbf{v}, & \mathbf{b} \Rightarrow \mathbf{r} & = \mathbf{c}, & \mathbf{c} \in \mathbb{V}, \ \mathbf{v} \in \mathbb{V}, \ \mathbf{r} \in \mathbb{V} \cup \{\bot\} \\ \\ \\ \mathbf{a} \ \mathbf{b} \Rightarrow \mathbf{b} \Rightarrow \bot & = \mathbf{a} \in \mathbb{V}, \ \mathbf{v} \in \mathbb{V}, \ \mathbf{v} \in \mathbb{V}, \ \mathbf{v} \in \mathbb{V} \cup \{\bot\} \\ \\ \end{array}$$

Again this should neither be understood as a structural induction (since $a|x \mapsto y \neq (\lambda x \cdot a)$ v) nor as action induction (because of infinite behaviors). The abstraction of $|D|^2 = T \cap (1 \times 1)$ yields the classical antural semantics |T| (where all rules with \bot are eliminated and \sqsubseteq becomes \subseteq in the remaining nones). The abstraction of $|T|^2 = T \cap (x \perp 1)$ yields the divergence semantics (beeping only the rules with \bot , \sqsubseteq is \supseteq , and $\Rightarrow \bot$ is written $a \stackrel{\infty}{\Longrightarrow}$ in [18].

Observe that both the maximal trace semantics of Sec. 5.3.1 and the above bifinitary relational semantics of Sec. 5.4 define the semantics of a term that "goes wrong" as empty.

23

Bi-inductive Structural Definitions

P. Cousot & R. Cousot. Bi-inductive Structural Semantics. SOS 2007, July 9, 2007, Wroclaw, Poland.





Syntax

- $-\ell,\ell_1,\ldots,\ell_n\in\mathbb{L}$ language
- $-\ell ::= \ell_1, \ldots, \ell_n$ derivation relation
- The "syntactic subcomponent" relation \prec on \mathbb{L} :

$$\ell' \prec \ell \triangleq \ell ::= \ell_1, \ldots, \ell', \ldots \ell_n$$

is

- irreflexive
- finite left images $(\forall \ell \in \mathbb{L} : |\{\ell' \in \mathbb{L} \mid \ell' \prec \ell\}| \in \mathbb{N})$
- well-founded
- Example: a, b, ... ::= $x \mid \lambda x \cdot a \mid a \text{ b defines } a \prec \lambda x \cdot a$, $a \prec a \text{ b and } b \prec a \text{ b}$.

Semantic domains

For each "syntactic component" $\ell \in \mathbb{L}$, we consider a semantic domain

$$\langle \mathcal{D}_{\ell}, \sqsubseteq_{\ell}, \perp_{\ell}, \sqcup_{\ell} \rangle$$

which is assumed to be a directed complete partial order (dcpo).

Variables

- To write definitions we use variables $X_{\ell}, Y_{\ell}, \ldots$ ranging over the semantic domains \mathcal{D}_{ℓ} of syntactic components $\ell \in \mathbb{L}$.

Transformers

- For derivations $\ell ::= \ell_1, \ldots, \ell_n$ we consider transformers

$$F_\ell^i \in \mathcal{D}_\ell imes \mathcal{D}_{\ell_1} \ldots imes \mathcal{D}_{\ell_n} \longmapsto \mathcal{D}_\ell$$

When n=0, we have $F_\ell^i\in\mathcal{D}_\ell\longmapsto\mathcal{D}_\ell$

 The transformers are assumed to be <u>□</u>_ℓ-monotone in their first parameter ²

 $egin{array}{lll} 2 & orall i \in \Delta_\ell, \ \ell_1, \ldots, \ell_n \ ee \ \ell, \ X, Y \ \in \ \mathcal{D}_\ell, X_1 \ \in \ \mathcal{D}_{\ell_1}, \ldots, X_n \ \in \ \mathcal{D}_{\ell_n} \! \colon \ X \ \sqsubseteq_\ell \ Y \implies F^i_\ell(X, X_1, \ldots, X_n) \ \sqsubseteq_\ell \ F^i_\ell(Y, X_1, \ldots, X_n). \end{array}$

Alternatives

- For each "syntactic component" $\ell \in \mathbb{L}$, we let Δ_{ℓ} be indexed sequences (totally ordered sets) of alternatives/definition cases.
- Given a set S,

$$pprox egin{array}{ll} \langle oldsymbol{x_i}, \ oldsymbol{i} \in oldsymbol{\Delta_\ell}
ightarrow oldsymbol{\delta_\ell} \ & \prod_{oldsymbol{i} \in oldsymbol{\Delta_\ell}} oldsymbol{x_i} \in oldsymbol{\Delta_\ell} \ & oldsymbol{i} \in oldsymbol{\Delta_\ell} \ & oldsymbol{i} \in oldsymbol{\Delta_\ell} \end{array}$$

indexed sequence cartesian product

Join

- For each "syntactic component" $\ell \in \mathbb{L}$, the join

$$\gamma_{\!\ell} \in (\Delta_{\ell} \longmapsto \mathcal{D}_{\ell}) \longmapsto \mathcal{D}_{\ell}$$

is used to gather alternatives in formal definitions

- The join operator is assumed to be componentwise <u>L</u>_ℓ-monotone³
- $-\bigvee_{i\in arDelta_\ell} X_i riangleq \gamma_\ell (\prod_{i\in arDelta_\ell} X_i), ext{ for short}$
- If the order of presentation of the alternatives is irrelevant Δ_{ℓ} is a set and the join is associative, commutative, and \sqsubseteq_{ℓ} monotone

$$\overline{\ \ ^3 \ orall \langle X_i, \ i \in \Delta_\ell
angle : orall \langle Y_i, \ i \in \Delta_\ell
angle : (orall i \in \Delta_\ell : X_i \sqsubseteq_\ell Y_i) \Longrightarrow igwedge_\ell (\prod_{i \in \Delta_\ell} X_i) \sqsubseteq_\ell igwedge_\ell (\prod_{i \in \Delta_\ell} Y_i).$$

Fixpoint definitions

A fixpoint definition for all $\ell \in \mathbb{L}$ such that $\ell ::= \ell_1, \ldots, \ell_n$ has the form

$$\mathcal{S}_f\llbracket\ell
rbracket = \mathsf{Ifp}^{\sqsubseteq_\ell} \; oldsymbol{\lambda} \, X oldsymbol{\cdot} igwedge_{\ell}^i F^i_\ell(X, \mathcal{S}_f\llbracket\ell_1
rbracket, \ldots, \mathcal{S}_f\llbracket\ell_n
rbracket) \; .$$

where $\mathsf{Ifp}^{\sqsubseteq}$ is the partially defined \sqsubseteq -least fixpoint operator on a poset $\langle P, \sqsubseteq \rangle$.

Lemma $1 \,\, orall \ell \in \mathbb{L} : \mathcal{S}_f\llbracket \ell
rbracket \,\,\, is \,\, well \,\, defined.$



Fixpoint definitions, particular cases

- without fixpoint:

$$\bigvee_{i\in\Delta_{\ell}}F_{\ell}^{i}(\mathcal{S}_{f}\llbracket\ell_{1}\rrbracket,\ldots,\mathcal{S}_{f}\llbracket\ell_{n}\rrbracket)=\operatorname{Ifp}^{\sqsubseteq_{\ell}}\boldsymbol{\lambda}X\cdot\bigvee_{i\in\Delta_{\ell}}F_{\ell}^{i}(\mathcal{S}_{f}\llbracket\ell_{1}\rrbracket,\ldots,\mathcal{S}_{f}\llbracket\ell_{n}\rrbracket)$$

– and without join:

$$F_\ell^i(\mathcal{S}_f\llbracket\ell_1
rbracket, \mathcal{S}_f\llbracket\ell_n
rbracket) = \operatorname{Ifp}^{\sqsubseteq_\ell} oldsymbol{\lambda} \, X oldsymbol{\cdot} igwedge_{\ell}^{i'}(\mathcal{S}_f\llbracket\ell_1
rbracket, \mathcal{S}_f\llbracket\ell_n
rbracket).$$

Example 1: fixpoint big-step maximal trace semantics

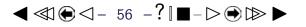
The bifinitary trace semantics $\vec{\mathbb{S}} \in \wp(\overline{\mathbb{T}}^{\infty})$ is

$$ec{\mathbb{S}} riangleq \mathsf{lfp}^{oxtsyle}ec{F}$$

where $\vec{F} \in \wp(\overline{\mathbb{T}}^{\infty}) \mapsto \wp(\overline{\mathbb{T}}^{\infty})$ is

$$\vec{F}(S) \triangleq \{ \mathbf{v} \in \overline{\mathbb{T}}^{\infty} \mid \mathbf{v} \in \mathbb{V} \} \cup \\ \{ (\boldsymbol{\lambda} \times \cdot \mathbf{a}) \times \mathbf{a} [\mathbf{x} \leftarrow \mathbf{v}] \cdot \boldsymbol{\sigma} \mid \mathbf{v} \in \mathbb{V} \wedge \mathbf{a} [\mathbf{x} \leftarrow \mathbf{v}] \cdot \boldsymbol{\sigma} \in S \} \cup \\ \{ \boldsymbol{\sigma} \otimes \mathbf{b} \mid \boldsymbol{\sigma} \in S^{\omega} \} \cup \\ \{ (\boldsymbol{\sigma} \otimes \mathbf{b}) \cdot (\mathbf{v} \ \mathbf{b}) \cdot \boldsymbol{\sigma}' \mid \boldsymbol{\sigma} \neq \boldsymbol{\epsilon} \wedge \boldsymbol{\sigma} \cdot \mathbf{v} \in S^{+} \wedge \mathbf{v} \in \mathbb{V} \wedge (\mathbf{v} \ \mathbf{b}) \cdot \boldsymbol{\sigma}' \in S \} \cup \\ \{ \mathbf{a} \otimes \boldsymbol{\sigma} \mid \mathbf{a} \in \mathbb{V} \wedge \boldsymbol{\sigma} \in S^{\omega} \} \cup \\ \{ (\mathbf{a} \otimes \boldsymbol{\sigma}) \cdot (\mathbf{a} \ \mathbf{v}) \cdot \boldsymbol{\sigma}' \mid \mathbf{a}, \mathbf{v} \in \mathbb{V} \wedge \boldsymbol{\sigma} \neq \boldsymbol{\epsilon} \wedge \boldsymbol{\sigma} \cdot \mathbf{v} \in S^{+} \wedge (\mathbf{a} \ \mathbf{v}) \cdot \boldsymbol{\sigma}' \in S \} . \text{ (f)}$$

We have $\mathbb{L} = \{\bullet\}$ (no structural induction), $\Delta_{\bullet} \triangleq \{a, b, c, d, e, f\}$ where $\vec{F}_{\bullet}^{i}(S)$, $i \in \Delta_{\bullet}$ is defined by equation (i). The join operator is chosen in binary form as $\Upsilon_{\bullet} \triangleq \cup$.



Example 2: fixpoint small-step maximal trace semantics

- The small-step maximal trace semantics $\xrightarrow{\infty}$ of a transition relation \longrightarrow is

Junction ; of set of traces:

$$S \ dagger T riangleq S^\omega \cup \{\sigma_0 ullet \ldots ullet \sigma_{|\sigma|-2} ullet \sigma' \mid \sigma \in S^+ \land \sigma_{|\sigma|-1} = \sigma'_0 \land \sigma' \in T\}$$

- Small-step transformer $\vec{f} \in \wp(\overline{\mathbb{T}}^{\infty}) \mapsto \wp(\overline{\mathbb{T}}^{\infty})$: $\vec{f}(T) \triangleq \{ \mathsf{v} \in \overline{\mathbb{T}}^{\infty} \mid \mathsf{v} \in \mathbb{V} \} \cup \xrightarrow{2} \S T \tag{1}$
- Small-step maximal trace semantics $\stackrel{\infty}{\longrightarrow}$ in fixpoint form: $\stackrel{\infty}{\longrightarrow} = |\mathbf{fp}|^{\sqsubseteq} \vec{f}$.
- The big-step and small-step trace semantics are the same

$$\vec{\mathbb{S}} = \stackrel{\infty}{\longrightarrow} .$$

Constraint-based definitions

A constraint-based definition has the form:

 $\langle S_e \llbracket \ell \rrbracket, \ell \in \mathbb{L} \rangle$ is the componentwise \sqsubseteq_{ℓ} -least $\langle X_{\ell}, \ell \in \mathbb{L} \rangle$ satisfying the system of constraints (inequations)

$$\left\{egin{array}{l} igg|_{\ell} F^i_\ell(X_\ell,\prod_{\ell' < \ell} X_{\ell'}) \sqsubseteq_\ell X_\ell \ i \in \Delta_\ell \ \ell \in \mathbb{L} \end{array}
ight.$$

Rule-based definitions

 A rule-based definition is a sequence of rules of the form

$$egin{aligned} rac{X_{\ell}}{F_{\ell}^{i}(X_{\ell},\prod_{\ell' \prec \ell}\mathcal{S}_{r}\llbracket \ell'
rbracket)} arthing \ell \in \mathbb{L}, i \in \Delta_{\ell} \end{aligned}$$

where the premise and conclusion are elements of the $\langle \mathcal{D}_{\ell}, \sqsubseteq_{\ell} \rangle$ cpo.

– If F_{ℓ}^{i} does not depend upon the premise X_{ℓ} , it is an axiom

Rule-based definitions in logical form

$$egin{aligned} X_\ell &\sqsubseteq_\ell \mathcal{S}_r\llbracket\ell
rbracket \ F_\ell^i(X_\ell, \prod_{\ell' \prec \ell} \mathcal{S}_r\llbracket\ell'
rbracket) &\sqsubseteq_\ell \mathcal{S}_r\llbracket\ell
rbracket \ F_\ell^i(X_\ell, \prod_{\ell' \prec \ell} \mathcal{S}_r\llbracket\ell'
rbracket) & = \ell \end{aligned} \qquad \ell \in \mathbb{L}, \,\, X_\ell \in \mathcal{D}_\ell, i \in \Delta_\ell$$

To make thejoin γ explicit, we can write

$$egin{aligned} X_\ell &\sqsubseteq_\ell \mathcal{S}_r\llbracket\ell
rbracket \ rac{iggriup_\ell}{iggriup_\ell} F_\ell^i(X_\ell, \prod_{\ell' \prec \ell} \mathcal{S}_r\llbracket\ell'
rbracket) iggriup_\ell \mathcal{S}_r\llbracket\ell
rbracket \ \mathcal{S}_r\llbracket\ell
rbracket \end{bmatrix}} arepsilon_\ell \in \mathbb{L}, \,\, X_\ell \in \mathcal{D}_\ell \,\,. \end{aligned}$$



Proofs

- A $D \in \mathcal{D}_{\ell}$ is *provable* if and only if it has a *proof* that is a transfinite sequence $^{4}D_{0}, \ldots, D_{\lambda}$ of elements of \mathcal{D}_{ℓ} such that
 - $D_0 = \perp_{\ell}$, $D_{\lambda} = D$ and
 - $\text{ for all } 0 < \delta \leqslant \lambda, \, D_{\delta} \sqsubseteq_{\ell} \bigvee_{i \in \Delta_{\ell}} F_{\ell}^{i}(\bigsqcup_{\ell} D_{\beta}, \prod_{\ell' \prec \ell} \mathcal{S}_{r}\llbracket \ell' \rrbracket).$
- The *meaning* of a rule-based definition is

$$\mathcal{S}_r\llbracket\ell
rbracket = ig|_{oldsymbol{\ell}} \{D \in \mathcal{D}_{oldsymbol{\ell}} \mid D \; \textit{is provable} \} \; .$$

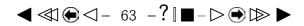
⁴ In the classical case [Acz77], the fixpoint operator is continuous whence proofs are finite.



4. Abstraction







Kleenian abstraction

$$-\langle \mathcal{D}, \sqsubseteq, \perp, \sqcup \rangle, \langle \mathcal{D}^{\sharp}, \sqsubseteq^{\sharp}, \perp^{\sharp}, \sqcup^{\sharp} \rangle$$
 dcpos

$$- F \in \mathcal{D} \mapsto \mathcal{D}, F^{\sharp} \in \mathcal{D}^{\sharp} \mapsto \mathcal{D}^{\sharp}$$
 monotone

$$-\alpha \in \mathcal{D} \mapsto \mathcal{D}^{\sharp}$$
 strict and continuous on chains of \mathcal{D}

 $-\alpha \circ F = F^{\sharp} \circ \alpha$, commutation condition

$$\Longrightarrow lpha(\operatorname{lfp}^{\sqsubseteq} F) = \operatorname{lfp}^{\sqsubseteq^{\sharp}} F^{\sharp}$$

OK for abstracting finite behaviors, not infinite ones



Tarskian abstraction

$$-\langle \mathcal{D}, \sqsubseteq, \perp, \sqcup \rangle, \langle \mathcal{D}^{\sharp}, \sqsubseteq^{\sharp}, \perp^{\sharp}, \sqcup^{\sharp} \rangle$$
 dcpos

$$- F \in \mathcal{D} \mapsto \mathcal{D}, F^{\sharp} \in \mathcal{D}^{\sharp} \mapsto \mathcal{D}^{\sharp}$$
 monotone

$$-\alpha \in \mathcal{D} \mapsto \mathcal{D}^{\sharp}$$
 preserves meets

$$-F^{\sharp}\circ\alpha\sqsubseteq^{\sharp}\alpha\circ F$$
, semi-commutation condition

$$egin{array}{lll} -orall y\in \mathcal{D}^{\sharp}: (F^{\sharp}(y)\mathrel{\sqsubseteq^{\sharp}} y) &\Longrightarrow (\exists x\in \mathcal{D}: lpha(x)=y\land F(x)\mathrel{\sqsubseteq} x \end{array}$$

$$\implies lpha(\mathsf{lfp}^{\sqsubseteq}F) = \mathsf{lfp}^{\sqsubseteq^{\sharp}}F^{\sharp}$$

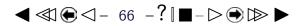
OK for abstracting infinite behaviors, not finite ones \Rightarrow abstract by parts.



5. Conclusion







Requirements

- Both convergence/termination and divergence/nonterminating behaviors are needed in static strictness analysis [Myc80], safety & security analysis, typing [Cou97, Ler06], etc;
- Such static analyzes must be proved correct with respect to a semantics chosen at an appropriate level of abstraction (small-step/big-step trace/relational/natural semantics);

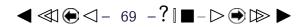
Requirements satisfaction

The bifinite extension of OS should satisfy the need for formal finite and infinite semantics, at various levels of abstraction and using various equivalent presentations (fixpoints, equational, constraints and inference rules) needed in static program analysis.

THE END



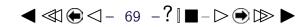




THE END, THANK YOU







Bibliography

- [Acz77] P. Aczel. An introduction to inductive definitions. In J. Barwise, editor, *Handbook of Mathematical Logic*, volume 90 of *Studies in Logic and the Foundations of Mathematics*, pages 739–782. Elsevier, 1977.
- [CC92] P. Cousot and R. Cousot. Inductive definitions, semantics and abstract interpretation. In 19th POPL, pages 83–94, Albuquerque, NM, US, 1992. ACM Press.
- [Cou97] P. Cousot. Types as abstract interpretations, invited paper. In 24th POPL, pages 316–331, Paris, FR, Jan. 1997. ACM Press.
- [Kah88] G. Kahn. Natural semantics. In K. Fuchi and M. Nivat, editors, *Programming of Future Generation Computers*, pages 237–258. Elsevier, 1988.
- [Ler06] X. Leroy. Coinductive big-step operational semantics. In P. Sestoft, editor, *Proc.* 15th ESOP '2006, Vienna, AT, LNCS 3924, pages 54–68. Springer, 27–28 Mar. 2006.
- [Myc80] A. Mycroft. The theory and practice of transforming call-by-need into call-by-value. In B. Robinet, editor, *Proc.* 4th Int. Symp. on Programming, Paris, FR, 22–24 Apr. 1980, LNCS 83, pages 270–281. Springer, 1980.



[Plo81] G.D. Plotkin. A structural approach to operational semantics. Technical Report DAIMI FN-19, Aarhus University, DK, Sep. 1981.