

Program Semantics

MPRI 2–6: Abstract Interpretation,
application to verification and static analysis

Antoine Miné

Year 2017–2018

Course 02

20 September 2017

Discuss several **flavors of concrete semantics**:

- **independently** from programming languages (transition systems)
- defined in a **constructive** way (as fixpoints)
- **compare** their expressive power (link by abstractions)

Plan:

- transition systems
- state semantics (forward and backward)
- trace semantics (finite and infinite)
- relational semantics
- state and trace properties

Transition systems

Transition systems: definition

Language-neutral formalism to discuss program semantics.

Transition system: (Σ, τ)

- set of states Σ ,
(memory states, λ -terms, configurations, etc., generally infinite)
- transition relation $\tau \subseteq \Sigma \times \Sigma$.

(Σ, τ) is a general form of small-step operational semantics.

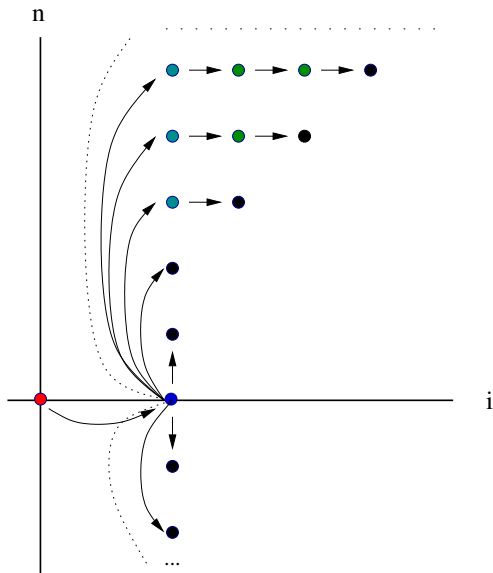
$(\sigma, \sigma') \in \tau$ is noted $\sigma \rightarrow \sigma'$:

starting in state σ , after one execution step, we can go to state σ' .

Transition system: example

```

i ← 2;
n ← [−∞, +∞];
while i < n do
  if ? then
    i ← i + 1
  
```

$$\Sigma \stackrel{\text{def}}{=} \{i, n\} \rightarrow \mathbb{Z}$$


From programs to transition systems

Example: on a simple imperative language.

Language syntax

${}^{\ell}stat^{\ell}$	$::=$	${}^{\ell}X \leftarrow expr^{\ell}$	(assignment)
		${}^{\ell}if\ expr \bowtie 0\ then\ {}^{\ell}stat^{\ell}$	(conditional)
		${}^{\ell}while\ {}^{\ell}expr \bowtie 0\ do\ {}^{\ell}stat^{\ell}$	(loop)
		${}^{\ell}stat; {}^{\ell}stat^{\ell}$	(sequence)

- $X \in \mathbb{V}$, where \mathbb{V} is a finite set of program variables,
- $\ell \in \mathcal{L}$ is a finite set of control labels,
- $\bowtie \in \{=, \leq, \dots\}$, the syntax of $expr$ is left undefined.
(see next course)

Program states: $\Sigma \stackrel{\text{def}}{=} \mathcal{L} \times \mathcal{E}$ are composed of:

- a **control** state in \mathcal{L} ,
- a **memory** state in $\mathcal{E} \stackrel{\text{def}}{=} \mathbb{V} \rightarrow \mathbb{R}$.

From programs to transition systems

Transitions: $\tau[\ell \text{ stat } \ell'] \subseteq \Sigma \times \Sigma$ is defined by induction on the syntax.

Assuming that expression semantics is given as $E[e] : \mathcal{E} \rightarrow \mathcal{P}(\mathbb{R})$.
(see next course)

$$\tau[\ell^1 X \leftarrow e^{\ell^2}] \stackrel{\text{def}}{=} \{ (\ell^1, \rho) \rightarrow (\ell^2, \rho[X \mapsto v]) \mid \rho \in \mathcal{E}, v \in E[e] \rho \}$$

$$\begin{aligned} \tau[\ell^1 \text{if } e \bowtie 0 \text{ then } \ell^2 s^{\ell^3}] &\stackrel{\text{def}}{=} \\ &\{ (\ell^1, \rho) \rightarrow (\ell^2, \rho) \mid \rho \in \mathcal{E}, \exists v \in E[e] \rho: v \bowtie 0 \} \cup \\ &\{ (\ell^1, \rho) \rightarrow (\ell^3, \rho) \mid \rho \in \mathcal{E}, \exists v \in E[e] \rho: v \not\bowtie 0 \} \cup \tau[\ell^2 s^{\ell^3}] \end{aligned}$$

$$\begin{aligned} \tau[\ell^1 \text{while } \ell^2 e \bowtie 0 \text{ do } \ell^3 s^{\ell^4}] &\stackrel{\text{def}}{=} \\ &\{ (\ell^1, \rho) \rightarrow (\ell^2, \rho) \mid \rho \in \mathcal{E} \} \cup \\ &\{ (\ell^2, \rho) \rightarrow (\ell^3, \rho) \mid \rho \in \mathcal{E}, \exists v \in E[e] \rho: v \bowtie 0 \} \cup \\ &\{ (\ell^2, \rho) \rightarrow (\ell^4, \rho) \mid \rho \in \mathcal{E}, \exists v \in E[e] \rho: v \not\bowtie 0 \} \cup \tau[\ell^3 s^{\ell^2}] \end{aligned}$$

$$\tau[\ell^1 s_1; \ell^2 s_2^{\ell^3}] \stackrel{\text{def}}{=} \tau[\ell^1 s_1] \cup \tau[\ell^2 s_2^{\ell^3}]$$

Initial, final, blocking states

Initial and final states:

Transition systems (Σ, τ) are often **enriched** with:

- $\mathcal{I} \subseteq \Sigma$ a set of distinguished **initial** states,
- $\mathcal{F} \subseteq \Sigma$ a set of distinguished **final** states.

(e.g., limit observation to executions starting in an initial state and ending in a final state)

Blocking states \mathcal{B} :

- states with **no successor** $\mathcal{B} \stackrel{\text{def}}{=} \{ \sigma \mid \forall \sigma' \in \Sigma: \sigma \not\rightarrow \sigma' \}$,
- model both correct program termination and program errors, (correct exit, program stuck, unhandled exception, etc.)
- often include (or equal) final states \mathcal{F} .

Note: we can always remove blocking states by completing τ :

$$\tau' \stackrel{\text{def}}{=} \tau \cup \{ (\sigma, \sigma) \mid \sigma \in \mathcal{B} \}. \quad (\text{add self-loops})$$

State semantics

State operators

Post-image, pre-image

Forward and backward images, in $\mathcal{P}(\Sigma) \rightarrow \mathcal{P}(\Sigma)$:

- **successors:** (forward, post-image)

$$\text{post}_\tau(S) \stackrel{\text{def}}{=} \{ \sigma' \mid \exists \sigma \in S : \sigma \rightarrow \sigma' \}$$

- **predecessors:** (backward, pre-image)

$$\text{pre}_\tau(S) \stackrel{\text{def}}{=} \{ \sigma \mid \exists \sigma' \in S : \sigma \rightarrow \sigma' \}$$

post_τ and pre_τ are complete \cup -morphisms in $(\mathcal{P}(\Sigma), \subseteq, \cup, \cap, \emptyset, \Sigma)$.

$$(\text{post}_\tau(\cup_{i \in I} S_i) = \cup_{i \in I} \text{post}_\tau(S_i), \text{pre}_\tau(\cup_{i \in I} S_i) = \cup_{i \in I} \text{pre}_\tau(S_i))$$

post_τ and pre_τ are strict. $(\text{post}_\tau(\emptyset) = \text{pre}_\tau(\emptyset) = \emptyset)$

Dual images

Dual post-images and pre-images:

- $\widetilde{\text{pre}}_{\tau}(S) \stackrel{\text{def}}{=} \{ \sigma \mid \forall \sigma' : \sigma \rightarrow \sigma' \implies \sigma' \in S \}$
(states such that all successors satisfy S)

- $\widetilde{\text{post}}_{\tau}(S) \stackrel{\text{def}}{=} \{ \sigma' \mid \forall \sigma : \sigma \rightarrow \sigma' \implies \sigma \in S \}$
(states such that all predecessors satisfy S)

$\widetilde{\text{pre}}_{\tau}$ and $\widetilde{\text{post}}_{\tau}$ are complete \cap -morphisms and not strict.

$\widetilde{\text{post}}$ is not much used...

Correspondences between images and dual images

We have the following correspondences:

- inverse:**

$$\text{pre}_\tau = \text{post}_{(\tau^{-1})} \quad \text{post}_\tau = \text{pre}_{(\tau^{-1})}$$

$$\widetilde{\text{pre}}_\tau = \widetilde{\text{post}}_{(\tau^{-1})} \quad \widetilde{\text{post}}_\tau = \widetilde{\text{pre}}_{(\tau^{-1})}$$

(where $\tau^{-1} \stackrel{\text{def}}{=} \{(\sigma, \sigma') \mid (\sigma', \sigma) \in \tau\}$)

- Galois connections:**

$$(\mathcal{P}(\Sigma), \subseteq) \xleftrightarrow[\text{post}_\tau]{\widetilde{\text{pre}}_\tau} (\mathcal{P}(\Sigma), \subseteq) \text{ and}$$

$$(\mathcal{P}(\Sigma), \subseteq) \xleftrightarrow[\text{pre}_\tau]{\widetilde{\text{post}}_\tau} (\mathcal{P}(\Sigma), \subseteq).$$

proof:

$$\text{post}_\tau(A) \subseteq B \iff \{\sigma' \mid \exists \sigma \in A: \sigma \rightarrow \sigma'\} \subseteq B \iff (\forall \sigma \in A: \sigma \rightarrow \sigma' \implies \sigma' \in B) \iff (A \subseteq \{\sigma \mid \forall \sigma': \sigma \rightarrow \sigma' \implies \sigma' \in B\}) \iff A \subseteq \widetilde{\text{pre}}_\tau(B);$$

other directions are similar.

Deterministic systems

Determinism:

- (Σ, τ) is **deterministic** if $\forall \sigma \in \Sigma: |\text{post}_\tau(\{\sigma\})| = 1$,
(every state has a single successor, no blocking state)
- most transition systems are **non-deterministic**.
(e.g., effect of input $X \leftarrow [0, 10]$, program termination)

We have the following correspondences:

- If τ is deterministic
then $\text{pre}_\tau = \widetilde{\text{pre}}_\tau$ and $\text{post}_\tau = \widetilde{\text{post}}_\tau$.
- More generally: $\forall S: \mathcal{B} \subseteq \widetilde{\text{pre}}_\tau(S) \subseteq \text{pre}_\tau(S) \cup \mathcal{B}$.

When $\mathcal{B} = \emptyset$ (no blocking state)

then $\widetilde{\text{pre}}_\tau(S) \subseteq \text{pre}_\tau(S)$.

post: reachability state semantics

Forward reachability

$\mathcal{R}(\mathcal{I})$: states **reachable from \mathcal{I}** in the transition system

$$\begin{aligned}\mathcal{R}(\mathcal{I}) &\stackrel{\text{def}}{=} \{ \sigma \mid \exists n \geq 0, \sigma_0, \dots, \sigma_n: \sigma_0 \in \mathcal{I}, \sigma = \sigma_n, \forall i: \sigma_i \rightarrow \sigma_{i+1} \} \\ &= \bigcup_{n \geq 0} \text{post}_{\tau}^n(\mathcal{I})\end{aligned}$$

(reachable \iff reachable from \mathcal{I} in n steps of τ for some $n \geq 0$)

$\mathcal{R}(\mathcal{I})$ can be expressed in **fixpoint form**:

$$\mathcal{R}(\mathcal{I}) = \text{lfp } F_{\mathcal{R}} \text{ where } F_{\mathcal{R}}(S) \stackrel{\text{def}}{=} \mathcal{I} \cup \text{post}_{\tau}(S)$$

($F_{\mathcal{R}}$ shifts S and adds back \mathcal{I})

Alternate characterization: $\mathcal{R} = \text{lfp}_{\mathcal{I}} G_{\mathcal{R}}$ where $G_{\mathcal{R}}(S) \stackrel{\text{def}}{=} S \cup \text{post}_{\tau}(S)$.

($G_{\mathcal{R}}$ shifts S by τ and accumulates the result with S)

(proofs on next slide)

Forward reachability: proof

proof: of $\mathcal{R}(\mathcal{I}) = \text{lfp } F_{\mathcal{R}}$ where $F_{\mathcal{R}}(S) \stackrel{\text{def}}{=} \mathcal{I} \cup \text{post}_{\tau}(S)$

$(\mathcal{P}(\Sigma), \subseteq)$ is a CPO and post_{τ} is continuous, hence $F_{\mathcal{R}}$ is continuous:
 $F_{\mathcal{R}}(\cup_{i \in I} A_i) = \cup_{i \in I} F_{\mathcal{R}}(A_i)$.

By Kleene's theorem, $\text{lfp } F_{\mathcal{R}} = \cup_{n \in \mathbb{N}} F_{\mathcal{R}}^n(\emptyset)$.

We prove by recurrence on n that: $\forall n: F_{\mathcal{R}}^n(\emptyset) = \cup_{i < n} \text{post}_{\tau}^i(\mathcal{I})$.
 (states reachable in less than n steps)

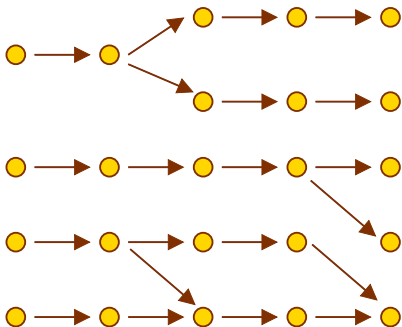
- $F_{\mathcal{R}}^0(\emptyset) = \emptyset$
- assuming the property at n ,

$$\begin{aligned} F_{\mathcal{R}}^{n+1}(\emptyset) &= F_{\mathcal{R}}(\cup_{i < n} \text{post}_{\tau}^i(\mathcal{I})) \\ &= \mathcal{I} \cup \text{post}_{\tau}(\cup_{i < n} \text{post}_{\tau}^i(\mathcal{I})) \\ &= \mathcal{I} \cup \cup_{i < n} \text{post}_{\tau}(\text{post}_{\tau}^i(\mathcal{I})) \\ &= \mathcal{I} \cup \cup_{1 \leq i < n+1} \text{post}_{\tau}^i(\mathcal{I}) \\ &= \cup_{i < n+1} \text{post}_{\tau}^i(\mathcal{I}) \end{aligned}$$

Hence: $\text{lfp } F_{\mathcal{R}} = \cup_{n \in \mathbb{N}} F_{\mathcal{R}}^n(\emptyset) = \cup_{i \in \mathbb{N}} \text{post}_{\tau}^i(\mathcal{I}) = \mathcal{R}(\mathcal{I})$.

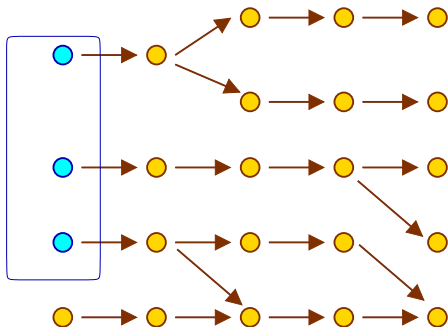
The proof is similar for the alternate form, given that $\text{lfp}_{\mathcal{I}} G_{\mathcal{R}} = \cup_{n \in \mathbb{N}} G_{\mathcal{R}}^n(\mathcal{I})$ and
 $G_{\mathcal{R}}^n(\mathcal{I}) = F_{\mathcal{R}}^{n+1}(\emptyset) = \cup_{i \leq n} \text{post}_{\tau}^i(\mathcal{I})$.

Forward reachability: graphical illustration



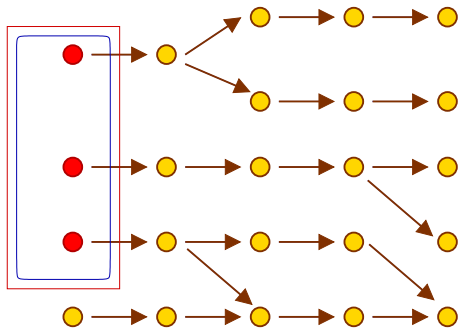
Transition system.

Forward reachability: graphical illustration



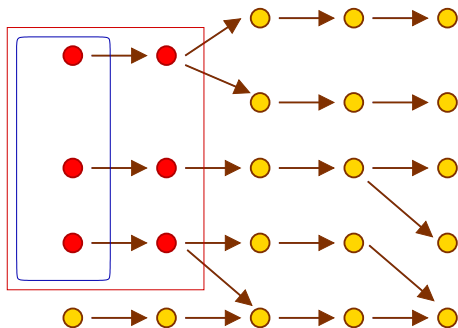
Initial states \mathcal{I} .

Forward reachability: graphical illustration



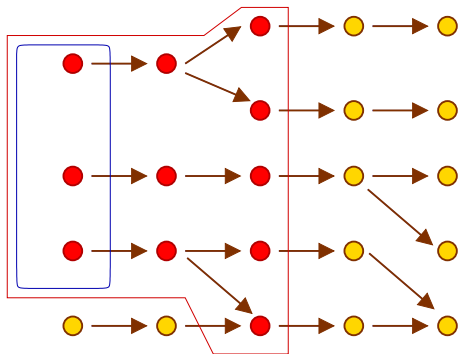
Iterate $F_{\mathcal{R}}^1(\mathcal{I})$.

Forward reachability: graphical illustration



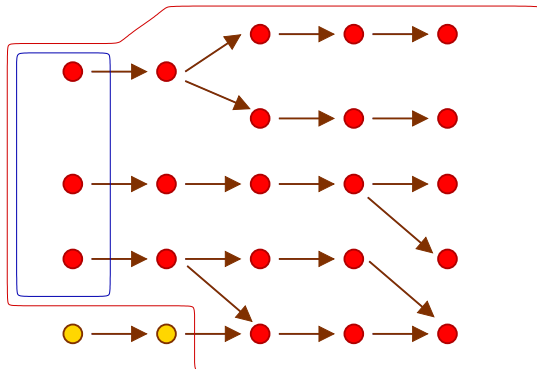
Iterate $F_{\mathcal{R}}^2(\mathcal{I})$.

Forward reachability: graphical illustration



Iterate $F_{\mathcal{R}}^3(\mathcal{I})$.

Forward reachability: graphical illustration



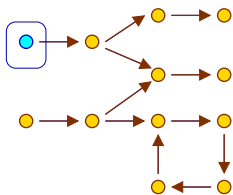
States reachable from \mathcal{I} : $\mathcal{R}(\mathcal{I}) = F_{\mathcal{R}}^5(\mathcal{I})$.

Multiple forward fixpoints

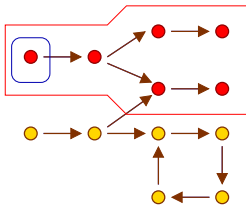
Recall: $\mathcal{R}(\mathcal{I}) = \text{lfp } F_{\mathcal{R}}$ where $F_{\mathcal{R}}(S) \stackrel{\text{def}}{=} \mathcal{I} \cup \text{post}_{\tau}(S)$.

Note that $F_{\mathcal{R}}$ may have **several** fixpoints.

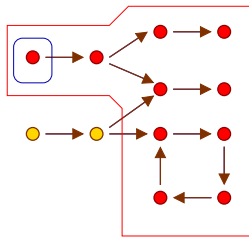
Example:



Initial state \mathcal{I}



$\mathcal{R}(\mathcal{I}) = \text{lfp } F_{\mathcal{R}}$



$\text{gfp } F_{\mathcal{R}}$

Exercise:

Compute all the fixpoints of $G_{\mathcal{R}}(S) \stackrel{\text{def}}{=} S \cup \text{post}_{\tau}(S)$ on this example.

Forward reachability: applications

- Infer the set of possible states at program end: $\mathcal{R}(\mathcal{I}) \cap \mathcal{F}$.

example

```

•  $i \leftarrow 0;$ 
  while  $i < 100$  do
     $i \leftarrow i + 1;$ 
     $j \leftarrow j + [0, 1]$ 
  done •
  
```

- initial states \mathcal{I} : $j \in [0, 10]$ at control state •,
 - final states \mathcal{F} : any memory state at control state •,
 - $\implies \mathcal{R}(\mathcal{I}) \cap \mathcal{F}$: control at •, $i = 100$, and $j \in [0, 110]$.
- Prove the absence of run-time error: $\mathcal{R}(\mathcal{I}) \cap \mathcal{B} \subseteq \mathcal{F}$.
(never block except when reaching the end of the program)

Reachability as an equation system

By partitioning forward reachability wrt. control states, we retrieve the **equation system** form of program semantics.

Control state partitioning

We assume $\Sigma \stackrel{\text{def}}{=} \mathcal{L} \times \mathcal{E}$; note that: $\mathcal{P}(\Sigma) \simeq \mathcal{L} \rightarrow \mathcal{P}(\mathcal{E})$.

We have a Galois **isomorphism**:

$$(\mathcal{P}(\Sigma), \subseteq) \begin{array}{c} \xleftarrow{\gamma_{\mathcal{L}}} \\ \xrightarrow{\alpha_{\mathcal{L}}} \end{array} (\mathcal{L} \rightarrow \mathcal{P}(\mathcal{E}), \dot{\subseteq})$$

- $X \dot{\subseteq} Y \stackrel{\text{def}}{\iff} \forall l \in \mathcal{L}: X(l) \subseteq Y(l)$
- $\alpha_{\mathcal{L}}(S) \stackrel{\text{def}}{=} \lambda l. \{ \rho \mid (l, \rho) \in S \}$
- $\gamma_{\mathcal{L}}(X) \stackrel{\text{def}}{=} \{ (l, \rho) \mid l \in \mathcal{L}, \rho \in X(l) \}$
- simply reorganize the states by control location!

Note that: $\alpha_{\mathcal{L}} \circ \gamma_{\mathcal{L}} = \gamma_{\mathcal{L}} \circ \alpha_{\mathcal{L}} = id$. (no abstraction)

Forward reachability equation system example

Idea:

compute $F_{eq} \stackrel{\text{def}}{=} \alpha_{\mathcal{L}} \circ F_{\mathcal{R}} \circ \gamma_{\mathcal{L}}$, where F_{eq} operates in $\mathcal{L} \rightarrow \mathcal{P}(\mathcal{E})$
 we get an equation system $\forall \ell \in \mathcal{L}: \mathcal{X}_{\ell} = F_{eq, \ell}(\mathcal{X}_1, \dots, \mathcal{X}_n)$

Example:

$\ell 1$	$i \leftarrow 2;$	$\mathcal{X}_1 = \mathcal{I}_1$
$\ell 2$	$n \leftarrow [-\infty, +\infty];$	$\mathcal{X}_2 = C[[i \leftarrow 2]] \mathcal{X}_1$
$\ell 3$	while $\ell 4$ $i < n$ do	$\mathcal{X}_3 = C[[n \leftarrow [-\infty, +\infty]]] \mathcal{X}_2$
	$\ell 5$ if $[0, 1] = 0$ then	$\mathcal{X}_4 = \mathcal{X}_3 \cup \mathcal{X}_7$
	$\ell 6$ $i \leftarrow i + 1$	$\mathcal{X}_5 = C[[i < n]] \mathcal{X}_4$
	$\ell 7$	$\mathcal{X}_6 = \mathcal{X}_5$
$\ell 8$		$\mathcal{X}_7 = \mathcal{X}_5 \cup C[[i \leftarrow i + 1]] \mathcal{X}_6$
		$\mathcal{X}_8 = C[[i \geq n]] \mathcal{X}_4$

- initial states $\mathcal{I} \stackrel{\text{def}}{=} \{(\ell 1, \rho) \mid \rho \in \mathcal{I}_1\}$ for some $\mathcal{I}_1 \subseteq \mathcal{E}$,
- $C[[\cdot]] : \mathcal{P}(\mathcal{E}) \rightarrow \mathcal{P}(\mathcal{E})$ model assignments and tests (see next slide).

We get the **strongest invariant** at each program point.

Forward reachability equation system construction

We derive the **equation system** $eq(\ell \text{ stat}^{\ell'})$
 from the program syntax $\ell \text{ stat}^{\ell'}$ by **structural induction**:

$$eq(\ell^1 X \leftarrow e^{\ell^2}) \stackrel{\text{def}}{=} \{ \mathcal{X}_{\ell^2} = C[[X \leftarrow e]] \mathcal{X}_{\ell^1} \}$$

$$eq(\ell^1 \text{if } e \bowtie 0 \text{ then } \ell^2 s^{\ell^3}) \stackrel{\text{def}}{=} \\ \{ \mathcal{X}_{\ell^2} = C[[e \bowtie 0]] \mathcal{X}_{\ell^1}, \mathcal{X}_{\ell^3} = \mathcal{X}_{\ell^3'} \cup C[[e \not\bowtie 0]] \mathcal{X}_{\ell^1} \} \cup eq(\ell^2 s^{\ell^3'})$$

$$eq(\ell^1 \text{while } \ell^2 e \bowtie 0 \text{ do } \ell^3 s^{\ell^4}) \stackrel{\text{def}}{=} \\ \{ \mathcal{X}_{\ell^2} = \mathcal{X}_{\ell^1} \cup \mathcal{X}_{\ell^4'}, \mathcal{X}_{\ell^3} = C[[e \bowtie 0]] \mathcal{X}_{\ell^2}, \mathcal{X}_{\ell^4} = C[[e \not\bowtie 0]] \mathcal{X}_{\ell^2} \} \cup eq(\ell^3 s^{\ell^4'})$$

$$eq(\ell^1 s_1; \ell^2 s_2^{\ell^3}) \stackrel{\text{def}}{=} eq(\ell^1 s_1^{\ell^2}) \cup (\ell^2 s_2^{\ell^3})$$

where:

- $\mathcal{X}^{\ell^3'}$, $\mathcal{X}^{\ell^4'}$ are fresh variables storing intermediate results
- $C[[X \leftarrow e]] \mathcal{X} \stackrel{\text{def}}{=} \{ \rho[X \mapsto v] \mid \rho \in \mathcal{X}, v \in E[[e]] \rho \}$
 $C[[e \bowtie 0]] \mathcal{X} \stackrel{\text{def}}{=} \{ \rho \in \mathcal{X} \mid \exists v \in E[[\rho]] \rho : v \bowtie 0 \}$

\cup -morphisms in a complete lattice \implies a smallest solution exists

pre: co-reachability state semantics

Backward reachability

$\mathcal{C}(\mathcal{F})$: states **co-reachable from** \mathcal{F} in the transition system:

$$\begin{aligned} \mathcal{C}(\mathcal{F}) &\stackrel{\text{def}}{=} \{ \sigma \mid \exists n \geq 0, \sigma_0, \dots, \sigma_n : \sigma = \sigma_0, \sigma_n \in \mathcal{F}, \forall i : \sigma_i \rightarrow \sigma_{i+1} \} \\ &= \bigcup_{n \geq 0} \text{pre}_\tau^n(\mathcal{F}) \end{aligned}$$

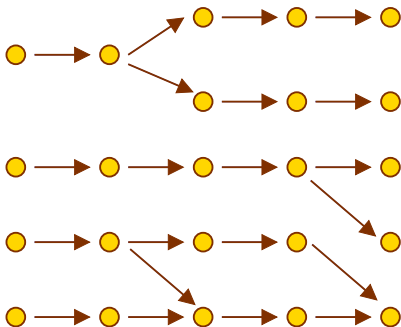
$\mathcal{C}(\mathcal{F})$ can also be expressed in **fixpoint form**:

$$\mathcal{C}(\mathcal{F}) = \text{lfp } F_{\mathcal{C}} \text{ where } F_{\mathcal{C}}(S) \stackrel{\text{def}}{=} \mathcal{F} \cup \text{pre}_\tau(S)$$

Alternate characterization: $\mathcal{C}(\mathcal{F}) = \text{lfp}_{\mathcal{F}} G_{\mathcal{C}}$ where $G_{\mathcal{C}}(S) = S \cup \text{pre}_\tau(S)$

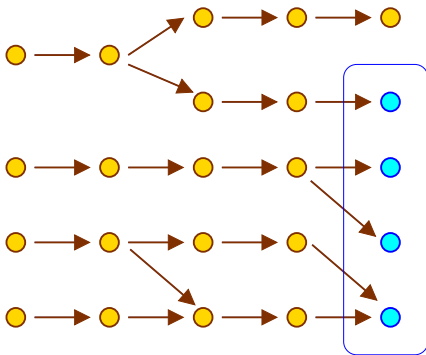
Justification: $\mathcal{C}(\mathcal{F})$ in τ is exactly $\mathcal{R}(\mathcal{F})$ in τ^{-1} .

Backward reachability: graphical illustration



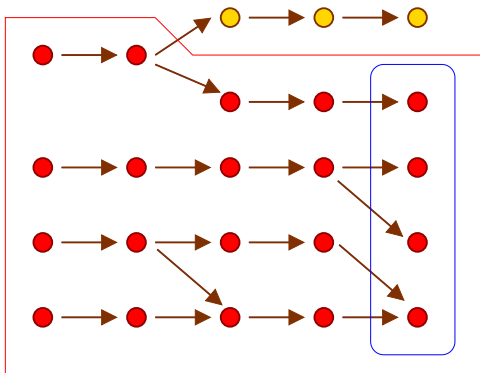
Transition system.

Backward reachability: graphical illustration



Final states \mathcal{F} .

Backward reachability: graphical illustration



States co-reachable from \mathcal{F} .

Backward reachability: applications

- $\mathcal{I} \cap \mathcal{C}(\mathcal{B} \setminus \mathcal{F})$

Initial states that have **at least one** erroneous execution.

program

```

•  $j \leftarrow 0;$ 
  while  $i > 0$  do
     $i \leftarrow i - 1;$ 
     $j \leftarrow j + [0, 10]$ 
  done •

```

- initial states \mathcal{I} : $i \in [0, 100]$ at •
- final states \mathcal{F} : any memory state at •
- blocking states \mathcal{B} : final, or $j > 200$ at any location
- $\mathcal{I} \cap \mathcal{C}(\mathcal{B} \setminus \mathcal{F})$: at •, $i > 20$

- $\mathcal{I} \cap (\Sigma \setminus \mathcal{C}(\mathcal{B}))$

Initial states that necessarily cause the program to loop.

- **Iterate** forward and backward analyses interactively
 \implies abstract debugging [Bour93].

Backward reachability equation system

Principle:

As before, reorganize transitions by label $\ell \in \mathcal{L}$,
to get an equation system on $(\mathcal{X}_\ell)_\ell$, with $\mathcal{X}_\ell \subseteq \mathcal{E}$

Example:

$\ell 1 \quad i \leftarrow 2;$ $\ell 2 \quad n \leftarrow [-\infty, +\infty];$ $\ell 3 \quad \mathbf{while} \quad \ell 4 \quad i < n \quad \mathbf{do}$ $\quad \ell 5 \quad \mathbf{if} \quad [0, 1] = 0 \quad \mathbf{then}$ $\quad \quad \ell 6 \quad i \leftarrow i + 1$ $\quad \ell 7$ $\ell 8$	$\mathcal{X}_1 = C[i \rightarrow 2] \mathcal{X}_2$ $\mathcal{X}_2 = C[n \rightarrow [-\infty, +\infty]] \mathcal{X}_3$ $\mathcal{X}_3 = \mathcal{X}_4$ $\mathcal{X}_4 = C[i < n] \mathcal{X}_5 \cup C[i \geq n] \mathcal{X}_8$ $\mathcal{X}_5 = \mathcal{X}_6 \cup \mathcal{X}_7$ $\mathcal{X}_6 = C[i \rightarrow i + 1] \mathcal{X}_7$ $\mathcal{X}_7 = \mathcal{X}_4$ $\mathcal{X}_8 = \mathcal{F}_8$
---	---

- final states $\mathcal{F} \stackrel{\text{def}}{=} \{(\ell 8, \rho) \mid \rho \in \mathcal{F}_8\}$ for some $\mathcal{F}_8 \subseteq \mathcal{E}$,
- $C[X \rightarrow e] \mathcal{X} \stackrel{\text{def}}{=} \{\rho \mid \exists v \in E[e] \rho: \rho[X \mapsto v] \in \mathcal{X}\}$.

$\widetilde{\text{pre}}$: pre-condition state semantics

Sufficient preconditions

$\mathcal{S}(\mathcal{Y})$: states with executions **staying in \mathcal{Y}** .

$$\begin{aligned} \mathcal{S}(\mathcal{Y}) &\stackrel{\text{def}}{=} \{ \sigma \mid \forall n \geq 0, \sigma_0, \dots, \sigma_n : (\sigma = \sigma_0 \wedge \forall i : \sigma_i \rightarrow \sigma_{i+1}) \implies \sigma_n \in \mathcal{Y} \} \\ &= \bigcap_{n \geq 0} \widetilde{\text{pre}}_{\tau}^n(\mathcal{Y}) \end{aligned}$$

$\mathcal{S}(\mathcal{Y})$ can be expressed in **fixpoint form**:

$$\mathcal{S}(\mathcal{Y}) = \text{gfp } F_{\mathcal{S}} \text{ where } F_{\mathcal{S}}(S) \stackrel{\text{def}}{=} \mathcal{Y} \cap \widetilde{\text{pre}}_{\tau}(S)$$

proof sketch: similar to that of $\mathcal{R}(\mathcal{I})$, in the dual.

$F_{\mathcal{S}}$ is continuous in the dual CPO $(\mathcal{P}(\Sigma), \supseteq)$, because $\widetilde{\text{pre}}_{\tau}$ is:

$$F_{\mathcal{S}}(\bigcap_{i \in I} A_i) = \bigcap_{i \in I} F_{\mathcal{S}}(A_i).$$

By Kleene's theorem in the dual, $\text{gfp } F_{\mathcal{S}} = \bigcap_{n \in \mathbb{N}} F_{\mathcal{S}}^n(\Sigma)$.

We would prove by recurrence that $F_{\mathcal{S}}^n(\Sigma) = \bigcap_{i < n} \widetilde{\text{pre}}_{\tau}^i(\mathcal{Y})$.

Sufficient preconditions and reachability

Correspondence with reachability:

We have a **Galois connection**:

$$(\mathcal{P}(\Sigma), \subseteq) \begin{matrix} \xleftarrow{\mathcal{S}} \\ \xrightarrow{\mathcal{R}} \end{matrix} (\mathcal{P}(\Sigma), \subseteq)$$

- $\mathcal{R}(\mathcal{I}) \subseteq \mathcal{Y} \iff \mathcal{I} \subseteq \mathcal{S}(\mathcal{Y})$
 definition of a Galois connection
 all executions from \mathcal{I} stay in \mathcal{Y}
 $\iff \mathcal{I}$ includes only sufficient pre-conditions for \mathcal{Y}
- so $\mathcal{S}(\mathcal{Y}) = \bigcup \{X \mid \mathcal{R}(X) \subseteq \mathcal{Y}\}$
 by Galois connection property
 $\mathcal{S}(\mathcal{Y})$ is the largest initial set whose reachability is in \mathcal{Y}

We retrieve Dijkstra's **weakest liberal preconditions**.

(proof sketch on next slide)

Sufficient preconditions and reachability (proof)

proof sketch:

Recall that $\mathcal{R}(\mathcal{I}) = \text{lfp}_{\mathcal{I}} G_{\mathcal{R}}$ where $G_{\mathcal{R}}(S) = S \cup \text{post}_{\tau}(S)$.

Likewise, $\mathcal{S}(\mathcal{Y}) = \text{gfp}_{\mathcal{Y}} G_{\mathcal{S}}$ where $G_{\mathcal{S}}(S) = S \cap \widetilde{\text{pre}}_{\tau}(S)$.

Recall the Galois connection $(\mathcal{P}(\Sigma), \subseteq) \xleftrightarrow[\text{post}_{\tau}]{\widetilde{\text{pre}}_{\tau}} (\mathcal{P}(\Sigma), \subseteq)$.

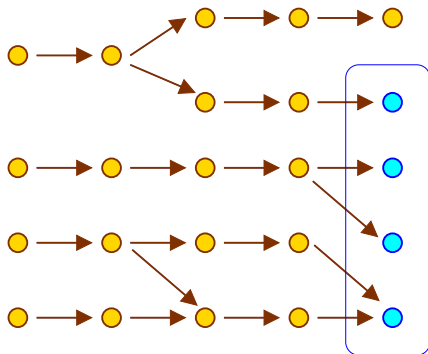
As a consequence $(\mathcal{P}(\Sigma), \subseteq) \xleftrightarrow[G_{\mathcal{R}}]{G_{\mathcal{S}}} (\mathcal{P}(\Sigma), \subseteq)$.

The Galois connection can be lifted to fixpoint operators:

$(\mathcal{P}(\Sigma), \subseteq) \xleftrightarrow[x \mapsto \text{lfp}_x G_{\mathcal{R}}]{x \mapsto \text{gfp}_x G_{\mathcal{S}}} (\mathcal{P}(\Sigma), \subseteq)$.

Exercise: complete the proof sketch.

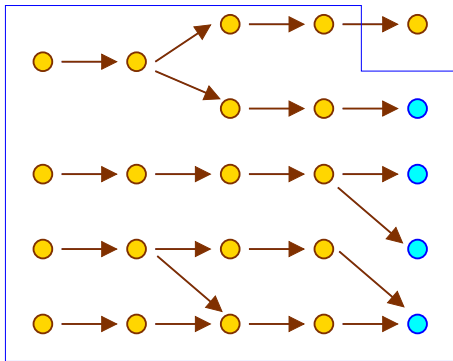
Sufficient preconditions: graphical illustration



Final states \mathcal{F} .

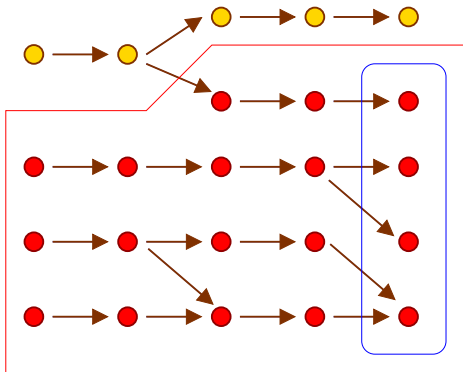
Goal: when stopping, stop in \mathcal{F} .

Sufficient preconditions: graphical illustration



Goal: avoid stopping in a non-final state (i.e., error state)
 but passing through a non-blocking state is not (yet) an error
 \implies consider $\mathcal{Y} = \mathcal{F} \cup (\Sigma \setminus \mathcal{B})$.

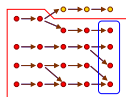
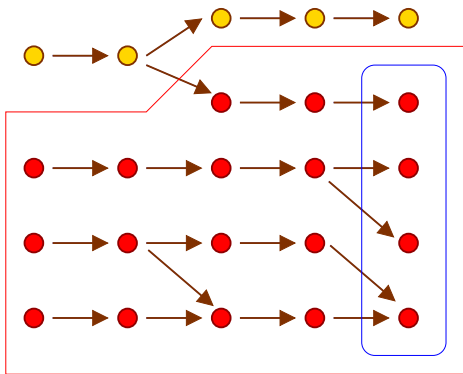
Sufficient preconditions: graphical illustration



Sufficient preconditions $\mathcal{S}(\mathcal{Y})$ to stop in \mathcal{F} .

(without forcing the program to stop at all)

Sufficient preconditions: graphical illustration



Sufficient preconditions $\mathcal{S}(\mathcal{Y})$ to stop in \mathcal{F} .

(without forcing the program to stop at all)

$\mathcal{C}(\mathcal{F})$

Note: $\mathcal{S}(\mathcal{Y}) \subset \mathcal{C}(\mathcal{F})$

Sufficient preconditions: application

Initial states such that **all executions** are correct:

$$\mathcal{I} \cap \mathcal{S}(\mathcal{F} \cup (\Sigma \setminus \mathcal{B})).$$

(the only blocking states reachable from initial states are final states)

program

```

•  $i \leftarrow 0;$ 
  while  $i < 100$  do
     $i \leftarrow i + 1;$ 
     $j \leftarrow j + [0, 1]$ 
  done •
  
```

- initial states \mathcal{I} : $j \in [0, 10]$ at •
- final states \mathcal{F} : any memory state at •
- blocking states \mathcal{B} : final, or $j > 105$ at any location
- $\mathcal{I} \cap \mathcal{S}(\mathcal{F} \cup (\Sigma \setminus \mathcal{B}))$: at •, $j \in [0, 5]$
(note that $\mathcal{I} \cap \mathcal{C}(\mathcal{F} \cup (\Sigma \setminus \mathcal{B}))$ gives \mathcal{I})

Applications: infer contracts;
optimize (hoist) tests;
infer counter-examples.

Trace semantics

Traces and trace operations

Sequences, traces

Trace: sequence of elements from Σ

- ϵ : empty trace (unique)
- σ : trace of length 1 (assimilated to a state)
- $\sigma_0, \dots, \sigma_{n-1}$: trace of length n
- $\sigma_0, \dots, \sigma_n, \dots$: infinite trace (length ω)

Trace sets:

- Σ^n : the set of traces of length n
- $\Sigma^{\leq n} \stackrel{\text{def}}{=} \bigcup_{i \leq n} \Sigma^i$: the set of traces of length at most n
- $\Sigma^* \stackrel{\text{def}}{=} \bigcup_{i \in \mathbb{N}} \Sigma^i$: the set of finite traces
- Σ^ω : the set of infinite traces
- $\Sigma^\infty \stackrel{\text{def}}{=} \Sigma^* \cup \Sigma^\omega$: the set of all traces

Traces of a transition system

Execution traces:

Non-empty sequences of states linked by the transition relation τ .

- can be **finite** (in $\mathcal{P}(\Sigma^*)$) or **infinite** (in $\mathcal{P}(\Sigma^\omega)$)
- can be anchored at initial states, or final states, or none

Atomic traces:

- \mathcal{I} : initial states \simeq set of traces of length 1
- \mathcal{F} : final states \simeq set of traces of length 1
- τ : transition relation \simeq set of traces of length 2
 $(\{\sigma, \sigma' \mid \sigma \rightarrow \sigma'\})$

(as $\Sigma \simeq \Sigma^1$ and $\Sigma \times \Sigma \simeq \Sigma^2$)

Trace operations

Operations on traces:

- **length:** $|t| \in \mathbb{N} \cup \{\omega\}$ of a trace $t \in \Sigma^\infty$
- **concatenation** \cdot
 - $(\sigma_0, \dots, \sigma_n) \cdot (\sigma'_0, \dots) \stackrel{\text{def}}{=} \sigma_0, \dots, \sigma_n, \sigma'_0, \dots$
(append to a finite trace)
 - $t \cdot t' \stackrel{\text{def}}{=} t$ if $t \in \Sigma^\omega$ (append to an infinite trace does nothing)
 - $\epsilon \cdot t \stackrel{\text{def}}{=} t \cdot \epsilon \stackrel{\text{def}}{=} t$ (ϵ is neutral)
- **junction** \frown
 - $(\sigma_0, \dots, \sigma_n) \frown (\sigma'_0, \sigma'_1, \dots) \stackrel{\text{def}}{=} \sigma_0, \dots, \sigma_n, \sigma'_1, \dots$ when $\sigma_n = \sigma'_0$
undefined if $\sigma_n \neq \sigma'_0$
 - $\epsilon \frown t$ and $t \frown \epsilon$ are undefined
 - $t \frown t' \stackrel{\text{def}}{=} t$, if $t \in \Sigma^\omega$

Trace operations (cont.)

Extension to sets of traces:

- $A \cdot B \stackrel{\text{def}}{=} \{a \cdot b \mid a \in A, b \in B\}$
 $\{\epsilon\}$ is the neutral element for \cdot
- $A \frown B \stackrel{\text{def}}{=} \{a \frown b \mid a \in A, b \in B, a \frown b \text{ defined}\}$
 Σ is the neutral element for \frown

A^0	$\stackrel{\text{def}}{=}$	$\{\epsilon\}$	$A \frown^0$	$\stackrel{\text{def}}{=}$	Σ
A^{n+1}	$\stackrel{\text{def}}{=}$	$A \cdot A^n$	$A \frown^{n+1}$	$\stackrel{\text{def}}{=}$	$A \frown A \frown^n$
A^ω	$\stackrel{\text{def}}{=}$	$A \cdot A \cdot \dots$	$A \frown^\omega$	$\stackrel{\text{def}}{=}$	$A \frown A \frown \dots$
A^*	$\stackrel{\text{def}}{=}$	$\bigcup_{n < \omega} A^n$	$A \frown^*$	$\stackrel{\text{def}}{=}$	$\bigcup_{n < \omega} A \frown^n$
A^∞	$\stackrel{\text{def}}{=}$	$\bigcup_{n \leq \omega} A^n$	$A \frown^\infty$	$\stackrel{\text{def}}{=}$	$\bigcup_{n \leq \omega} A \frown^n$

Note: $A^n \neq \{a^n \mid a \in A\}$, $A \frown^n \neq \{a \frown^n \mid a \in A\}$ when $|A| > 1$

Finite prefix trace semantics

Prefix trace semantics

$\mathcal{T}_p(\mathcal{I})$: partial, finite **execution traces** starting in \mathcal{I} .

$$\begin{aligned}\mathcal{T}_p(\mathcal{I}) &\stackrel{\text{def}}{=} \{ \sigma_0, \dots, \sigma_n \mid n \geq 0, \sigma_0 \in \mathcal{I}, \forall i: \sigma_i \rightarrow \sigma_{i+1} \} \\ &= \bigcup_{n \geq 0} \mathcal{I} \frown (\tau \frown^n)\end{aligned}$$

(traces of length n , for any n , starting in \mathcal{I} and following τ)

$\mathcal{T}_p(\mathcal{I})$ can be expressed in **fixpoint form**:

$$\mathcal{T}_p(\mathcal{I}) = \text{lfp } F_p \text{ where } F_p(T) \stackrel{\text{def}}{=} \mathcal{I} \cup T \frown \tau$$

(F_p appends a transition to each trace, and adds back \mathcal{I})

(proof on next slide)

Prefix trace semantics: proof

proof of: $\mathcal{T}_p(\mathcal{I}) = \text{lfp } F_p$ where $F_p(T) = \mathcal{I} \cup T \hat{\ } \tau$

Similar to the proof of $\mathcal{R}(\mathcal{I}) = \text{lfp } F_{\mathcal{R}}$ where $F_{\mathcal{R}}(S) \stackrel{\text{def}}{=} \mathcal{I} \cup \text{post}_{\tau}(S)$.

F_p is continuous in a CPO $(\mathcal{P}(\Sigma^*), \subseteq)$:

$$\begin{aligned} & F_p(\cup_{i \in I} T_i) \\ &= \mathcal{I} \cup (\cup_{i \in I} T_i) \hat{\ } \tau \\ &= \mathcal{I} \cup (\cup_{i \in I} T_i \hat{\ } \tau) = \cup_{i \in I} (\mathcal{I} \cup T_i \hat{\ } \tau) \end{aligned}$$

hence (Kleene), $\text{lfp } F_p = \cup_{n \geq 0} F_p^n(\emptyset)$

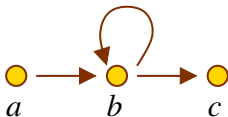
We prove by recurrence on n that $\forall n: F_p^n(\emptyset) = \cup_{i < n} \mathcal{I} \hat{\ } \tau \hat{\ } i$:

- $F_p^0(\emptyset) = \emptyset,$
- $$\begin{aligned} & F_p^{n+1}(\emptyset) \\ &= \mathcal{I} \cup F_p^n(\emptyset) \hat{\ } \tau \\ &= \mathcal{I} \cup (\cup_{i < n} \mathcal{I} \hat{\ } \tau \hat{\ } i) \hat{\ } \tau \\ &= \mathcal{I} \cup \cup_{i < n} (\mathcal{I} \hat{\ } \tau \hat{\ } i) \hat{\ } \tau \\ &= \mathcal{I} \hat{\ } \tau \hat{\ } 0 \cup \cup_{i < n} (\mathcal{I} \hat{\ } \tau \hat{\ } i+1) \\ &= \cup_{i < n+1} \mathcal{I} \hat{\ } \tau \hat{\ } i \end{aligned}$$

Thus, $\text{lfp } F_p = \cup_{n \in \mathbb{N}} F_p^n(\emptyset) = \cup_{n \in \mathbb{N}} \cup_{i < n} \mathcal{I} \hat{\ } \tau \hat{\ } i = \cup_{i \in \mathbb{N}} \mathcal{I} \hat{\ } \tau \hat{\ } i$.

Note: we also have $\mathcal{T}_p(\mathcal{I}) = \text{lfp}_{\mathcal{I}} G_p$ where $G_p(T) = T \cup T \hat{\ } \tau$.

Prefix trace semantics: graphical illustration



$$\mathcal{I} \stackrel{\text{def}}{=} \{a\}$$

$$\tau \stackrel{\text{def}}{=} \{(a, b), (b, b), (b, c)\}$$

Iterates: $\mathcal{T}_p(\mathcal{I}) = \text{lfp } F_p$ where $F_p(T) \stackrel{\text{def}}{=} \mathcal{I} \cup T \cap \tau$.

- $F_p^0(\emptyset) = \emptyset$
- $F_p^1(\emptyset) = \mathcal{I} = \{a\}$
- $F_p^2(\emptyset) = \{a, ab\}$
- $F_p^3(\emptyset) = \{a, ab, abb, abc\}$
- $F_p^n(\emptyset) = \{a, ab^i, ab^j c \mid i \in [1, n-1], j \in [1, n-2]\}$
- $\mathcal{T}_p(\mathcal{I}) = \cup_{n \geq 0} F_p^n(\emptyset) = \{a, ab^i, ab^j c \mid i \geq 1\}$

Prefix trace semantics: expressive power

The prefix trace semantics is the collection of **finite observations** of program executions.

⇒ Semantics of **testing**.

Limitations:

- no information on **infinite** executions,
(we will add infinite traces later)
- can bound maximal execution time: $\mathcal{T}_p(\mathcal{I}) \subseteq \Sigma^{\leq n}$
but cannot bound **minimal execution time**.
(we will consider maximal traces later)

Abstracting traces into states

Idea: view state semantics as abstractions of traces semantics.

We have a **Galois embedding** between finite traces and states:

$$(\mathcal{P}(\Sigma^*), \subseteq) \begin{array}{c} \xleftarrow{\gamma_p} \\ \xrightarrow{\alpha_p} \end{array} (\mathcal{P}(\Sigma), \subseteq)$$

- $\alpha_p(T) \stackrel{\text{def}}{=} \{\sigma \in \Sigma \mid \exists \sigma_0, \dots, \sigma_n \in T : \sigma = \sigma_n\}$
(last state in traces in T)
- $\gamma_p(S) \stackrel{\text{def}}{=} \{\sigma_0, \dots, \sigma_n \in \Sigma^* \mid \sigma_n \in S\}$
(traces ending in a state in S)

(proof on next slide)

Abstracting traces into states (proof)

proof of: (α_p, γ_p) forms a Galois embedding.

Instead of the definition $\alpha(c) \subseteq a \iff c \subseteq \gamma(a)$, we use the alternate characterization of Galois connections: α and γ are monotonic, $\gamma \circ \alpha$ is extensive, and $\alpha \circ \gamma$ is reductive.

Embedding means that, additionally, $\alpha \circ \gamma = id$.

- α_p, γ_p are \cup -morphisms, hence monotonic
- $(\gamma_p \circ \alpha_p)(T)$

$$= \{ \sigma_0, \dots, \sigma_n \mid \sigma_n \in \alpha_p(T) \}$$

$$= \{ \sigma_0, \dots, \sigma_n \mid \exists \sigma'_0, \dots, \sigma'_m \in T : \sigma_n = \sigma'_m \}$$

$$\supseteq T$$
- $(\alpha_p \circ \gamma_p)(S)$

$$= \{ \sigma \mid \exists \sigma_0, \dots, \sigma_n \in \gamma_p(S) : \sigma = \sigma_n \}$$

$$= \{ \sigma \mid \exists \sigma_0, \dots, \sigma_n : \sigma_n \in S, \sigma = \sigma_n \}$$

$$= S$$

Abstracting prefix traces into reachability

Recall that:

- $\mathcal{T}_p(\mathcal{I}) = \text{lfp } F_p$ where $F_p(T) \stackrel{\text{def}}{=} \mathcal{I} \cup T \hat{\cap} \tau$,
- $\mathcal{R}(\mathcal{I}) = \text{lfp } F_{\mathcal{R}}$ where $F_{\mathcal{R}}(S) \stackrel{\text{def}}{=} \mathcal{I} \cup \text{post}_{\tau}(S)$,
- $(\mathcal{P}(\Sigma^*), \subseteq) \xleftarrow{\gamma_p} \xrightarrow{\alpha_p} (\mathcal{P}(\Sigma), \subseteq)$.

We have: $\alpha_p \circ F_p = F_{\mathcal{R}} \circ \alpha_p$;

by fixpoint transfer, we get: $\alpha_p(\mathcal{T}_p(\mathcal{I})) = \mathcal{R}(\mathcal{I})$.

(proof on next slide)

Abstracting prefix traces into reachability (proof)

proof: of $\alpha_p \circ F_p = F_{\mathcal{R}} \circ \alpha_p$

$$\begin{aligned}
 & (\alpha_p \circ F_p)(T) \\
 &= \alpha_p(\mathcal{I} \cup T \hat{\ } \tau) \\
 &= \{ \sigma \mid \exists \sigma_0, \dots, \sigma_n \in \mathcal{I} \cup T \hat{\ } \tau : \sigma = \sigma_n \} \\
 &= \mathcal{I} \cup \{ \sigma \mid \exists \sigma_0, \dots, \sigma_n \in T \hat{\ } \tau : \sigma = \sigma_n \} \\
 &= \mathcal{I} \cup \{ \sigma \mid \exists \sigma_0, \dots, \sigma_n \in T : \sigma_n \rightarrow \sigma \} \\
 &= \mathcal{I} \cup \text{post}_{\tau}(\{ \sigma \mid \exists \sigma_0, \dots, \sigma_n \in T : \sigma = \sigma_n \}) \\
 &= \mathcal{I} \cup \text{post}_{\tau}(\alpha_p(T)) \\
 &= (F_{\mathcal{R}} \circ \alpha_p)(T)
 \end{aligned}$$

Abstracting traces into states (example)

program

```
j ← 0;  
i ← 0;  
while i < 100 do  
  i ← i + 1;  
  j ← j + [0, 1]  
done
```

- prefix trace semantics:
 i and *j* are **increasing** and $0 \leq j \leq i \leq 100$
- forward reachable state semantics:
 $0 \leq j \leq i \leq 100$

⇒ the abstraction **forgets the ordering of states.**

Prefix closure

Prefix partial order: \preceq on Σ^∞

$$x \preceq y \stackrel{\text{def}}{\iff} \exists u \in \Sigma^\infty : x \cdot u = y$$

(Σ^∞, \preceq) is a CPO, while (Σ^*, \preceq) is not complete.

Prefix closure: $\rho_p : \mathcal{P}(\Sigma^\infty) \rightarrow \mathcal{P}(\Sigma^\infty)$

$$\rho_p(T) \stackrel{\text{def}}{=} \{u \mid \exists t \in T : u \preceq t, u \neq \epsilon\}$$

ρ_p is an upper closure operator on $\mathcal{P}(\Sigma^\infty \setminus \{\epsilon\})$.

(monotonic, extensive $T \subseteq \rho_p(T)$, idempotent $\rho_p \circ \rho_p = \rho_p$)

The prefix trace semantics is closed by prefix:

$$\rho_p(\mathcal{T}_p(\mathcal{I})) = \mathcal{T}_p(\mathcal{I}).$$

(note that $\epsilon \notin \mathcal{T}_p(\mathcal{I})$, which is why we disallowed ϵ in ρ_p)

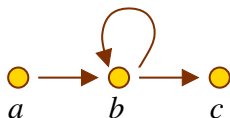
Finite suffix trace semantics

Suffix trace semantics

Similar results on the **suffix** trace semantics,
going backwards from \mathcal{F} :

- $\mathcal{T}_s(\mathcal{F}) \stackrel{\text{def}}{=} \{ \sigma_0, \dots, \sigma_n \mid n \geq 0, \sigma_n \in \mathcal{F}, \forall i: \sigma_i \rightarrow \sigma_{i+1} \}$
(traces following τ and ending in a state in \mathcal{F})
- $\mathcal{T}_s(\mathcal{F}) = \bigcup_{n \geq 0} \tau \frown^n \frown \mathcal{F}$
- $\mathcal{T}_s(\mathcal{F}) = \text{lfp } F_s$ where $F_s(T) \stackrel{\text{def}}{=} \mathcal{F} \cup \tau \frown T$
(F_s prepends a transition to each trace, and adds back \mathcal{F})
- $\alpha_s(\mathcal{T}_s(\mathcal{F})) = \mathcal{C}(\mathcal{F})$
where $\alpha_s(T) \stackrel{\text{def}}{=} \{ \sigma \mid \exists \sigma_0, \dots, \sigma_n \in T: \sigma = \sigma_0 \}$
- $\rho_s(\mathcal{T}_s(\mathcal{F})) = \mathcal{T}_s(\mathcal{F})$
where $\rho_s(T) \stackrel{\text{def}}{=} \{ u \mid \exists t \in \Sigma^\infty: t \cdot u \in T, u \neq \epsilon \}$
(closed by suffix)

Suffix trace semantics: graphical illustration



$$\mathcal{F} \stackrel{\text{def}}{=} \{c\}$$

$$\tau \stackrel{\text{def}}{=} \{(a, b), (b, b), (b, c)\}$$

Iterates: $\mathcal{T}_s(\mathcal{F}) = \text{lfp } F_s$ where $F_s(T) \stackrel{\text{def}}{=} \mathcal{F} \cup \tau \cap T$.

- $F_s^0(\emptyset) = \emptyset$
- $F_s^1(\emptyset) = \mathcal{F} = \{c\}$
- $F_s^2(\emptyset) = \{c, bc\}$
- $F_s^3(\emptyset) = \{c, bc, bbc, abc\}$
- $F_s^n(\emptyset) = \{c, b^i c, ab^j c \mid i \in [1, n-1], j \in [1, n-2]\}$
- $\mathcal{T}_s(\mathcal{F}) = \bigcup_{n \geq 0} F_s^n(\emptyset) = \{c, b^i c, ab^j c \mid i \geq 1\}$

Finite partial trace semantics

Finite partial trace semantics

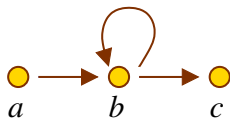
\mathcal{T} : all finite partial finite execution traces.

(not necessarily starting in \mathcal{I} or ending in \mathcal{F})

$$\begin{aligned} \mathcal{T} &\stackrel{\text{def}}{=} \{ \sigma_0, \dots, \sigma_n \mid n \geq 0, \forall i: \sigma_i \rightarrow \sigma_{i+1} \} \\ &= \bigcup_{n \geq 0} \Sigma \frown \tau \frown^n \\ &= \bigcup_{n \geq 0} \tau \frown^n \frown \Sigma \end{aligned}$$

- $\mathcal{T} = \mathcal{T}_p(\Sigma)$, hence $\mathcal{T} = \text{lfp } F_{p^*}$ where $F_{p^*}(T) \stackrel{\text{def}}{=} \Sigma \cup T \frown \tau$
(prefix partial traces from any initial state)
- $\mathcal{T} = \mathcal{T}_s(\Sigma)$, hence $\mathcal{T} = \text{lfp } F_{s^*}$ where $F_{s^*}(T) \stackrel{\text{def}}{=} \Sigma \cup \tau \frown T$
(suffix partial traces to any final state)
- $F_{p^*}^n(\emptyset) = F_{s^*}^n(\emptyset) = \bigcup_{i < n} \Sigma \frown \tau \frown^i = \bigcup_{i < n} \tau \frown^i \frown \Sigma = \mathcal{T} \cap \Sigma^{<n}$
- $\mathcal{T}_p(\mathcal{I}) = \mathcal{T} \cap (\mathcal{I} \cdot \Sigma^*)$ (restricted initial states)
- $\mathcal{T}_s(\mathcal{F}) = \mathcal{T} \cap (\Sigma^* \cdot \mathcal{F})$ (restricted final states)

Partial trace semantics: graphical illustration



$$\tau \stackrel{\text{def}}{=} \{(a, b), (b, b), (b, c)\}$$

Iterates: $\mathcal{T}(\Sigma) = \text{lfp } F_{p^*}$ where $F_{p^*}(T) \stackrel{\text{def}}{=} \Sigma \cup T \cap \tau$.

- $F_{p^*}^0(\emptyset) = \emptyset$
- $F_{p^*}^1(\emptyset) = \Sigma = \{a, b, c\}$
- $F_{p^*}^2(\emptyset) = \{a, b, c, ab, bb, bc\}$
- $F_{p^*}^3(\emptyset) = \{a, b, c, ab, bb, bc, abb, abc, bbb, bbc\}$
- $F_{p^*}^n(\emptyset) = \{ab^i, ab^j c, b^i c, b^k \mid i \in [0, n-1], j \in [1, n-2], k \in [1, n]\}$
- $\mathcal{T} = \cup_{n \geq 0} F_{p^*}^n(\emptyset) = \{ab^i, ab^j c, b^i c, b^j \mid i \geq 0, j > 1\}$

(using $F_{s^*}(T) \stackrel{\text{def}}{=} \Sigma \cup \tau \cap T$, we get the exact same iterates)

Abstracting partial traces to prefix traces

Idea: anchor partial traces at initial states \mathcal{I} .

We have a Galois connection:

$$(\mathcal{P}(\Sigma^*), \subseteq) \xrightleftharpoons[\alpha_{\mathcal{I}}]{\gamma_{\mathcal{I}}} (\mathcal{P}(\Sigma^*), \subseteq)$$

- $\alpha_{\mathcal{I}}(T) \stackrel{\text{def}}{=} T \cap (\mathcal{I} \cdot \Sigma^*)$ (keep only traces starting in \mathcal{I})
- $\gamma_{\mathcal{I}}(T) \stackrel{\text{def}}{=} T \cup ((\Sigma \setminus \mathcal{I}) \cdot \Sigma^*)$ (add all traces not starting in \mathcal{I})

We then have: $\mathcal{T}_p(\mathcal{I}) = \alpha_{\mathcal{I}}(\mathcal{T})$.

(similarly $\mathcal{T}_s(\mathcal{F}) = \alpha_{\mathcal{F}}(\mathcal{T})$ where $\alpha_{\mathcal{F}}(T) \stackrel{\text{def}}{=} T \cap (\Sigma^* \cdot \mathcal{F})$)

(proof on next slide)

Abstracting partial traces to prefix traces (proof)

proof

$\alpha_{\mathcal{I}}$ and $\gamma_{\mathcal{I}}$ are monotonic.

$$(\alpha_{\mathcal{I}} \circ \gamma_{\mathcal{I}})(T) = (T \cup (\Sigma \setminus \mathcal{I}) \cdot \Sigma^*) \cap \mathcal{I} \cdot \Sigma^* = T \cap \mathcal{I} \cdot \Sigma^* \subseteq T.$$

$$(\gamma_{\mathcal{I}} \circ \alpha_{\mathcal{I}})(T) = (T \cap \mathcal{I} \cdot \Sigma^*) \cup (\Sigma \setminus \mathcal{I}) \cdot \Sigma^* = T \cup (\Sigma \setminus \mathcal{I}) \cdot \Sigma^* \supseteq T.$$

So, we have a Galois connection.

A direct proof of $\mathcal{T}_p(\mathcal{I}) = \alpha_{\mathcal{I}}(\mathcal{T})$ is straightforward, by definition of \mathcal{T}_p , $\alpha_{\mathcal{I}}$, and \mathcal{T} .

We can also retrieve the result by fixpoint transfer.

$$\mathcal{T} = \text{lfp } F_{p^*} \text{ where } F_{p^*}(T) \stackrel{\text{def}}{=} \Sigma \cup T \hat{\cap} \tau.$$

$$\mathcal{T}_p = \text{lfp } F_p \text{ where } F_p(T) \stackrel{\text{def}}{=} \mathcal{I} \cup T \hat{\cap} \tau.$$

$$\text{We have: } (\alpha_{\mathcal{I}} \circ F_{p^*})(T) = (\Sigma \cup T \hat{\cap} \tau) \cap (\mathcal{I} \cdot \Sigma^*) =$$

$$\mathcal{I} \cup ((T \hat{\cap} \tau) \cap (\mathcal{I} \cdot \Sigma^*)) = \mathcal{I} \cup ((T \cap (\mathcal{I} \cdot \Sigma^*)) \hat{\cap} \tau) = (F_p \circ \alpha_{\mathcal{I}})(T).$$

Maximal finite and infinite trace semantics

Maximal traces

Maximal traces: $\mathcal{M}_\infty \in \mathcal{P}(\Sigma^\infty)$

- sequences of states linked by the transition relation τ ,
- start in any state ($\mathcal{I} = \Sigma$),
- either finite and **stop in a blocking state** ($\mathcal{F} = \mathcal{B}$),
- or **infinite**.

maximal traces cannot be “extended”
by adding a new transition in τ at their end

$$\mathcal{M}_\infty \stackrel{\text{def}}{=} \left\{ \sigma_0, \dots, \sigma_n \in \Sigma^* \mid \sigma_n \in \mathcal{B}, \forall i < n: \sigma_i \rightarrow \sigma_{i+1} \right\} \cup \left\{ \sigma_0, \dots, \sigma_n, \dots \in \Sigma^\omega \mid \forall i < \omega: \sigma_i \rightarrow \sigma_{i+1} \right\}$$

(can be anchored at \mathcal{I} and \mathcal{F} as: $\mathcal{M}_\infty \cap (\mathcal{I} \cdot \Sigma^\infty) \cap ((\Sigma^* \cdot \mathcal{F}) \cup \Sigma^\omega)$)

Partitioned fixpoint formulation of maximal traces

Goal: we look for a fixpoint characterization of \mathcal{M}_∞ .

We consider separately **finite** and **infinite** maximal traces.

- **Finite traces:** already done!

From the **suffix partial trace semantics**, recall:

$$\mathcal{M}_\infty \cap \Sigma^* = \mathcal{T}_s(\mathcal{B}) = \text{lfp } F_s$$

recall that $F_s(T) \stackrel{\text{def}}{=} \mathcal{B} \cup \tau \frown T$ in $(\mathcal{P}(\Sigma^*), \subseteq)$...

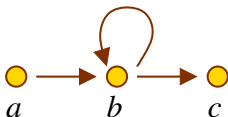
- **Infinite traces:**

Additionally, we will prove: $\mathcal{M}_\infty \cap \Sigma^\omega = \text{gfp } G_s$

where $G_s(T) \stackrel{\text{def}}{=} \tau \frown T$ in $(\mathcal{P}(\Sigma^\omega), \subseteq)$.

(proof in following slides)

Infinite trace semantics: graphical illustration



$$\mathcal{B} \stackrel{\text{def}}{=} \{c\}$$

$$\tau \stackrel{\text{def}}{=} \{(a, b), (b, b), (b, c)\}$$

Iterates: $\mathcal{M}_\infty \cap \Sigma^\omega = \text{gfp } G_s$ where $G_s(T) \stackrel{\text{def}}{=} \tau \frown T$.

- $G_s^0(\Sigma^\omega) = \Sigma^\omega$
- $G_s^1(\Sigma^\omega) = ab\Sigma^\omega \cup bb\Sigma^\omega \cup bc\Sigma^\omega$
- $G_s^2(\Sigma^\omega) = abb\Sigma^\omega \cup bbb\Sigma^\omega \cup abc\Sigma^\omega \cup bbc\Sigma^\omega$
- $G_s^3(\Sigma^\omega) = abbb\Sigma^\omega \cup bbbb\Sigma^\omega \cup abbc\Sigma^\omega \cup bbbc\Sigma^\omega$
- $G_s^n(\Sigma^\omega) = \{ab^n t, b^{n+1} t, ab^{n-1} ct, b^n ct \mid t \in \Sigma^\omega\}$
- $\mathcal{M}_\infty \cap \Sigma^\omega = \bigcap_{n \geq 0} G_s^n(\Sigma^\omega) = \{ab^\omega, b^\omega\}$

Infinite trace semantics: proof

$$\mathcal{M}_\infty \cap \Sigma^\omega = \text{gfp } G_s$$

where $G_s(T) \stackrel{\text{def}}{=} \tau \frown T$ in $(\mathcal{P}(\Sigma^\omega), \subseteq)$

proof:

G_s is continuous in $(\mathcal{P}(\Sigma^\omega), \supseteq)$: $G_s(\bigcap_{i \in I} T_i) = \bigcap_{i \in I} G_s(T_i)$.

By Kleene's theorem in the dual: $\text{gfp } G_s = \bigcap_{n \in \mathbb{N}} G_s^n(\Sigma^\omega)$.

We prove by recurrence on n that $\forall n: G_s^n(\Sigma^\omega) = \tau \frown^n \Sigma^\omega$:

- $G_s^0(\Sigma^\omega) = \Sigma^\omega = \tau \frown^0 \Sigma^\omega$,
- $G_s^{n+1}(\Sigma^\omega) = \tau \frown G_s^n(\Sigma^\omega) = \tau \frown (\tau \frown^n \Sigma^\omega) = \tau \frown^{n+1} \Sigma^\omega$.

$$\begin{aligned} \text{gfp } G_s &= \bigcap_{n \in \mathbb{N}} \tau \frown^n \Sigma^\omega \\ &= \{ \sigma_0, \dots \in \Sigma^\omega \mid \forall n \geq 0: \sigma_0, \dots, \sigma_{n-1} \in \tau \frown^n \Sigma^\omega \} \\ &= \{ \sigma_0, \dots \in \Sigma^\omega \mid \forall n \geq 0: \forall i < n: \sigma_i \rightarrow \sigma_{i+1} \} \\ &= \mathcal{M}_\infty \cap \Sigma^\omega \end{aligned}$$

Least fixpoint formulation of maximal traces

Idea: To get a **least fixpoint** formulation for whole \mathcal{M}_∞ ,
merge finite and infinite maximal trace least fixpoint forms.

Fixpoint fusion

$\mathcal{M}_\infty \cap \Sigma^*$ is best defined on $(\Sigma^*, \subseteq, \cup, \cap, \emptyset, \Sigma^*)$.

$\mathcal{M}_\infty \cap \Sigma^\omega$ is best defined on $(\Sigma^\omega, \supseteq, \cap, \cup, \Sigma^\omega, \emptyset)$, the **dual lattice**
(we transform the greatest fixpoint into a least fixpoint!)

We mix them into a **new** complete lattice $(\Sigma^\infty, \sqsubseteq, \sqcup, \sqcap, \perp, \top)$:

- $A \sqsubseteq B \stackrel{\text{def}}{\iff} (A \cap \Sigma^*) \subseteq (B \cap \Sigma^*) \wedge (A \cap \Sigma^\omega) \supseteq (B \cap \Sigma^\omega)$
- $A \sqcup B \stackrel{\text{def}}{=} ((A \cap \Sigma^*) \cup (B \cap \Sigma^*)) \cup ((A \cap \Sigma^\omega) \cap (B \cap \Sigma^\omega))$
- $A \sqcap B \stackrel{\text{def}}{=} ((A \cap \Sigma^*) \cap (B \cap \Sigma^*)) \cup ((A \cap \Sigma^\omega) \cup (B \cap \Sigma^\omega))$
- $\perp \stackrel{\text{def}}{=} \Sigma^\omega$
- $\top \stackrel{\text{def}}{=} \Sigma^*$

In this lattice, $\mathcal{M}_\infty = \text{lfp } F_s$ where $F_s(T) \stackrel{\text{def}}{=} \mathcal{B} \cup T \cap T$.

(proof on next slides)

Fixpoint fusion theorem

Theorem: fixpoint fusion

If $X_1 = \text{lfp } F_1$ in $(\mathcal{P}(\mathcal{D}_1), \sqsubseteq_1)$ and $X_2 = \text{lfp } F_2$ in $(\mathcal{P}(\mathcal{D}_2), \sqsubseteq_2)$

and $\mathcal{D}_1 \cap \mathcal{D}_2 = \emptyset$,

then $X_1 \cup X_2 = \text{lfp } F$ in $(\mathcal{P}(\mathcal{D}_1 \cup \mathcal{D}_2), \sqsubseteq)$ where:

- $F(X) \stackrel{\text{def}}{=} F_1(X \cap \mathcal{D}_1) \cup F_2(X \cap \mathcal{D}_2)$,
- $A \sqsubseteq B \iff (A \cap \mathcal{D}_1) \sqsubseteq_1 (B \cap \mathcal{D}_1) \wedge (A \cap \mathcal{D}_2) \sqsubseteq_2 (B \cap \mathcal{D}_2)$.

proof:

We have:

$F(X_1 \cup X_2) = F_1((X_1 \cup X_2) \cap \mathcal{D}_1) \cup F_2((X_1 \cup X_2) \cap \mathcal{D}_2) = F_1(X_1) \cup F_2(X_2) = X_1 \cup X_2$,
hence $X_1 \cup X_2$ is a fixpoint of F .

Let Y be a fixpoint. Then $Y = F(Y) = F_1(Y \cap \mathcal{D}_1) \cup F_2(Y \cap \mathcal{D}_2)$, hence,
 $Y \cap \mathcal{D}_1 = F_1(Y \cap \mathcal{D}_1)$ and $Y \cap \mathcal{D}_1$ is a fixpoint of F_1 . Thus, $X_1 \sqsubseteq_1 Y \cap \mathcal{D}_1$. Likewise,
 $X_2 \sqsubseteq_2 Y \cap \mathcal{D}_2$. We deduce that $X = X_1 \cup X_2 \sqsubseteq (Y \cap \mathcal{D}_1) \cup (Y \cap \mathcal{D}_2) = Y$, and so, X
is F 's least fixpoint.

note: we also have $\text{gfp } F = \text{gfp } F_1 \cup \text{gfp } F_2$.

Least fixpoint formulation of maximal traces (proof)

We are now ready to finish the proof that $\mathcal{M}_\infty = \text{lfp } F_s$
 where $F_s(T) \stackrel{\text{def}}{=} \mathcal{B} \cup \tau \cap T$

proof:

We have:

- $\mathcal{M}_\infty \cap \Sigma^* = \text{lfp } F_s$ in $(\mathcal{P}(\Sigma^*), \subseteq)$,
- $\mathcal{M}_\infty \cap \Sigma^\omega = \text{lfp } G_s$ in $(\mathcal{P}(\Sigma^\omega), \supseteq)$ where $G_s(T) \stackrel{\text{def}}{=} \tau \cap T$,
- in $\mathcal{P}(\Sigma^\infty)$, we have

$$F_s(A) = (F_s(A) \cap \Sigma^*) \cup (F_s(A) \cap \Sigma^\omega) = F_s(A \cap \Sigma^*) \cup G_s(A \cap \Sigma^\omega).$$

So, by fixpoint fusion in $(\mathcal{P}(\Sigma^\infty), \subseteq)$, we have:

$$\mathcal{M}_\infty = (\mathcal{M}_\infty \cap \Sigma^*) \cup (\mathcal{M}_\infty \cap \Sigma^\omega) = \text{lfp } F_s.$$

Note: a greatest fixpoint formulation in $(\Sigma^\infty, \supseteq)$ also exists!

Abstracting maximal traces into partial traces

Finite and infinite partial trace semantics

Idea: complete the partial traces \mathcal{T} with infinite traces.

\mathcal{T}_∞ : all finite and infinite sequences of states
linked by the transition relation τ :

$$\mathcal{T}_\infty \stackrel{\text{def}}{=} \left\{ \sigma_0, \dots, \sigma_n \in \Sigma^* \mid \forall i < n: \sigma_i \rightarrow \sigma_{i+1} \right\} \cup \left\{ \sigma_0, \dots, \sigma_n, \dots \in \Sigma^\omega \mid \forall i < \omega: \sigma_i \rightarrow \sigma_{i+1} \right\}$$

(partial finite traces do not necessarily end in a blocking state)

Fixpoint form similar to \mathcal{M}_∞ :

$\mathcal{T}_\infty = \text{lfp } F_{s*}$ in $(\mathcal{P}(\Sigma^\infty), \sqsubseteq)$ where $F_{s*}(T) \stackrel{\text{def}}{=} \Sigma \cup \tau \cap T$,

proof: similar to the proof of $\mathcal{M}_\infty = \text{lfp } F_s$.

Finite trace abstraction

Finite partial traces \mathcal{T} are an **abstraction** of all partial traces \mathcal{T}_∞ .

We have a **Galois embedding**:

$$(\mathcal{P}(\Sigma^\infty), \sqsubseteq) \begin{array}{c} \xleftarrow{\gamma_*} \\ \xrightarrow{\alpha_*} \end{array} (\mathcal{P}(\Sigma^*), \subseteq)$$

- \sqsubseteq is the fused ordering on $\Sigma^* \cup \Sigma^\omega$:

$$A \sqsubseteq B \stackrel{\text{def}}{\iff} (A \cap \Sigma^*) \subseteq (B \cap \Sigma^*) \wedge (A \cap \Sigma^\omega) \supseteq (B \cap \Sigma^\omega)$$

- $\alpha_*(T) \stackrel{\text{def}}{=} T \cap \Sigma^*$

(remove infinite traces)

- $\gamma_*(T) \stackrel{\text{def}}{=} T$

(embedding)

- $\mathcal{T} = \alpha_*(\mathcal{T}_\infty)$

(proof on next slide)

Finite trace abstraction (proof)

proof:

We have Galois embedding because:

- α_* and γ_* are monotonic,
- given $T \subseteq \Sigma^*$, we have $(\alpha_* \circ \gamma_*)(T) = T \cap \Sigma^* = T$,
- $(\gamma_* \circ \alpha_*)(T) = T \cap \Sigma^* \sqsubseteq T$, as we only remove infinite traces.

Recall that $\mathcal{T}_\infty = \text{lfp } F_{S^*}$ in $(\mathcal{P}(\Sigma^\infty), \sqsubseteq)$ and $\mathcal{T} = \text{lfp } F_{S^*}$ in $(\mathcal{P}(\Sigma^*), \subseteq)$, where $F_{S^*}(T) \stackrel{\text{def}}{=} \Sigma \cup T \cap \tau$.

As $\alpha_* \circ F_{S^*} = F_{S^*} \circ \alpha_*$ and $\alpha_*(\emptyset) = \emptyset$, we can apply the fixpoint transfer theorem to get $\alpha_*(\mathcal{T}_\infty) = \mathcal{T}$.

Prefix abstraction

Idea: complete **maximal** traces by adding (non-empty) **prefixes**.

We have a Galois connection:

$$(\mathcal{P}(\Sigma^\infty \setminus \{\epsilon\}), \subseteq) \begin{matrix} \xleftarrow{\gamma_\preceq} \\ \xrightarrow{\alpha_\preceq} \end{matrix} (\mathcal{P}(\Sigma^\infty \setminus \{\epsilon\}), \subseteq)$$

- $\alpha_\preceq(T) \stackrel{\text{def}}{=} \{t \in \Sigma^\infty \setminus \{\epsilon\} \mid \exists u \in T : t \preceq u\}$

(set of all non-empty prefixes of traces in T)

-

$$\gamma_\preceq(T) \stackrel{\text{def}}{=} \{t \in \Sigma^\infty \setminus \{\epsilon\} \mid \forall u \in \Sigma^\infty \setminus \{\epsilon\} : u \preceq t \implies u \in T\}$$

(traces with non-empty prefixes in T)

proof:

α_\preceq and γ_\preceq are monotonic.

$$(\alpha_\preceq \circ \gamma_\preceq)(T) = \{t \in T \mid \rho_p(t) \subseteq T\} \subseteq T \quad (\text{prefix-closed trace sets}).$$

$$(\gamma_\preceq \circ \alpha_\preceq)(T) = \rho_p(T) \supseteq T.$$

Abstraction from maximal traces to partial traces

Finite and infinite **partial traces** \mathcal{T}_∞ are an **abstraction** of **maximal traces** \mathcal{M}_∞ : $\mathcal{T}_\infty = \alpha_{\preceq}(\mathcal{M}_\infty)$.

proof:

Firstly, \mathcal{T}_∞ and $\alpha_{\preceq}(\mathcal{M}_\infty)$ coincide on infinite traces. Indeed, $\mathcal{T}_\infty \cap \Sigma^\omega = \mathcal{M}_\infty \cap \Sigma^\omega$ and α_{\preceq} does not add infinite traces, so: $\mathcal{T}_\infty \cap \Sigma^\omega = \alpha_{\preceq}(\mathcal{M}_\infty) \cap \Sigma^\omega$.

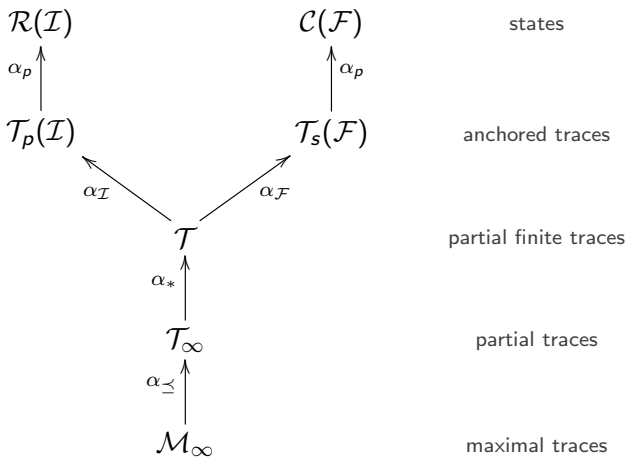
We now prove that they also coincide on finite traces. Assume

$\sigma_0, \dots, \sigma_n \in \alpha_{\preceq}(\mathcal{M}_\infty)$, then $\forall i < n: \sigma_i \rightarrow \sigma_{i+1}$, so, $\sigma_0, \dots, \sigma_n \in \mathcal{T}_\infty$.

Assume $\sigma_0, \dots, \sigma_n \in \mathcal{T}_\infty$, then it can be completed into a maximal trace, either finite or infinite, and so, $\sigma_0, \dots, \sigma_n \in \alpha_{\preceq}(\mathcal{M}_\infty)$.

Note: no fixpoint transfer applies here.

(Partial) hierarchy of semantics



Relational semantics

Big-step semantics

Finite big-step semantics

Pairs of states linked by a sequence of transitions in τ .

$$BS \stackrel{\text{def}}{=} \{ (\sigma_0, \sigma_n) \in \Sigma \times \Sigma \mid n \geq 0, \exists \sigma_1, \dots, \sigma_{n-1} : \forall i < n : \sigma_i \rightarrow \sigma_{i+1} \}$$

(symmetric and transitive closure of τ)

Fixpoint form:

$$BS = \text{lfp } F_B$$

$$\text{where } F_B(R) \stackrel{\text{def}}{=} id \cup \{ (\sigma, \sigma'') \mid \exists \sigma' : (\sigma, \sigma') \in R, \sigma' \rightarrow \sigma'' \}.$$

Relational abstraction

Relational abstraction: allows skipping intermediate steps.

We have a Galois embedding:

$$(\mathcal{P}(\Sigma^*), \subseteq) \begin{array}{c} \xleftarrow{\gamma_{io}} \\ \xrightarrow{\alpha_{io}} \end{array} (\mathcal{P}(\Sigma \times \Sigma), \subseteq)$$

- $\alpha_{io}(T) \stackrel{\text{def}}{=} \{(\sigma, \sigma') \mid \exists \sigma_0, \dots, \sigma_n \in T : \sigma = \sigma_0, \sigma' = \sigma_n\}$
(first and last state of a trace in T)
- $\gamma_{io}(R) \stackrel{\text{def}}{=} \{\sigma_0, \dots, \sigma_n \in \Sigma^* \mid \exists (\sigma, \sigma') \in R : \sigma = \sigma_0, \sigma' = \sigma_n\}$
(traces respecting the first and last states from R)

proof sketch:

γ_{io} and α_{io} are monotonic.

$$(\gamma_{io} \circ \alpha_{io})(T) = \{\sigma_0, \dots, \sigma_n \mid \exists \sigma'_0, \dots, \sigma'_m \in T : \sigma_0 = \sigma'_0, \sigma_n = \sigma'_m\}.$$

$$(\alpha_{io} \circ \gamma_{io})(R) = R.$$

Finite big-step semantics as an abstraction

The finite big-step semantics is an **abstraction** of the finite trace semantics: $\mathcal{BS} = \alpha_{io}(\mathcal{T})$.

proof sketch: by fixpoint transfer.

We have $\mathcal{T} = \text{lfp } F_{p*}$ where $F_{p*}(T) \stackrel{\text{def}}{=} \Sigma \cup T \cap \tau$.

Moreover, $F_B(R) \stackrel{\text{def}}{=} id \cup \{(\sigma, \sigma'') \mid \exists \sigma' : (\sigma, \sigma') \in R, \sigma' \rightarrow \sigma''\}$.

Then, $\alpha_{io} \circ F_{p*} = F_B \circ \alpha_{io}$ because $\alpha_{io}(\Sigma) = id$ and $\alpha_{io}(T \cap \tau) = \{(\sigma, \sigma'') \mid \exists \sigma' : (\sigma, \sigma') \in \alpha_{io}(T) \wedge \sigma' \rightarrow \sigma''\}$.

By fixpoint transfer: $\alpha_{io}(\mathcal{T}) = \text{lfp } F_B$.

We have a similar result using $F_{s*}(T) \stackrel{\text{def}}{=} \Sigma \cup \tau \cap T$ and

$F'_B(R) \stackrel{\text{def}}{=} id \cup \{(\sigma, \sigma'') \mid \exists \sigma' : (\sigma', \sigma'') \in R \wedge \sigma \rightarrow \sigma'\}$.

Finite big-step semantics (example)

program

```
 $i \leftarrow [0, +\infty];$   
while  $i > 0$  do  
   $i \leftarrow i - [0, 1];$   
done
```

Finite big-step semantics \mathcal{BS} : $\{(\rho, \rho') \mid 0 \leq \rho'(i) \leq \rho(i)\}$.

Relational denotational semantics

Denotational semantics (in relation form)

In the **denotational semantics**, we forget all the intermediate steps and only **keep the input / output relation**:

- $(\sigma, \sigma') \in \Sigma \times \mathcal{B}$: **finite** execution starting in σ , stopping in σ' ,
- $(\sigma, \circlearrowleft)$: **non-terminating** execution starting in σ .

(\neq big-step semantics: we no longer include (σ, σ') if σ' is not blocking!)

Construction by abstraction: of the maximal trace semantics \mathcal{M}_∞ .

$$(\mathcal{P}(\Sigma^\infty), \subseteq) \xleftrightarrow[\alpha_d]{\gamma_d} (\mathcal{P}(\Sigma \times (\Sigma \cup \{\circlearrowleft\})), \subseteq)$$

- $\alpha_d(T) \stackrel{\text{def}}{=} \alpha_{io}(T \cap \Sigma^*) \cup \{(\sigma, \circlearrowleft) \mid \exists t \in \Sigma^\omega: \sigma \cdot t \in T\}$
- $\gamma_d(R) \stackrel{\text{def}}{=} \gamma_{io}(R \cap (\Sigma \times \Sigma)) \cup \{\sigma \cdot t \mid (\sigma, \circlearrowleft) \in R, t \in \Sigma^\omega\}$
(extension of $(\alpha_{io}, \gamma_{io})$ to infinite traces)

The denotational semantics is $\mathcal{DS} \stackrel{\text{def}}{=} \alpha_d(\mathcal{M}_\infty)$.

Denotational fixpoint semantics

Idea: as \mathcal{M}_∞ , separate terminating and non-terminating behaviors, and use a fixpoint fusion theorem.

We have: $\mathcal{DS} = \text{lfp } F_d$

in $(\mathcal{P}(\Sigma \times (\Sigma \cup \{\circ\})), \sqsubseteq^*, \sqcup^*, \sqcap^*, \perp^*, \top^*)$, where

- $\perp^* \stackrel{\text{def}}{=} \{(\sigma, \circ) \mid \sigma \in \Sigma\}$
- $\top^* \stackrel{\text{def}}{=} \{(\sigma, \sigma') \mid \sigma, \sigma' \in \Sigma\}$
- $A \sqsubseteq^* B \iff ((A \cap \top^*) \subseteq (B \cap \top^*)) \wedge ((A \cap \perp^*) \supseteq (B \cap \perp^*))$
- $A \sqcup^* B \stackrel{\text{def}}{=} ((A \cap \top^*) \cup (B \cap \top^*)) \cup ((A \cap \perp^*) \cap (B \cap \perp^*))$
- $A \sqcap^* B \stackrel{\text{def}}{=} ((A \cap \top^*) \cap (B \cap \top^*)) \cup ((A \cap \perp^*) \cup (B \cap \perp^*))$
- $F_d(R) \stackrel{\text{def}}{=} \{(\sigma, \sigma) \mid \sigma \in \mathcal{B}\} \cup \{(\sigma, \sigma'') \mid \exists \sigma' : \sigma \rightarrow \sigma' \wedge (\sigma', \sigma'') \in R\}$

Denotational fixpoint semantics (proof)

proof:

We cannot use directly a fixpoint transfer on $\mathcal{M}_\infty = \text{lfp } F_s$ in $(\mathcal{P}(\Sigma^\infty), \sqsubseteq)$ because our Galois connection (α_d, γ_d) uses the \subseteq order, not \sqsubseteq !

Instead, we use fixpoint transfer separately on finite and infinite executions, and then apply fixpoint fusion.

Recall that $\mathcal{M}_\infty \cap \Sigma^* = \text{lfp } F_s$ in $(\mathcal{P}(\Sigma^*), \subseteq)$ where $F_s(T) \stackrel{\text{def}}{=} B \cup T \cap T$
 and $\mathcal{M}_\infty \cap \Sigma^\omega = \text{gfp } G_s$ in $(\mathcal{P}(\Sigma^\omega), \subseteq)$ where $G_s(T) \stackrel{\text{def}}{=} \cup T \cap T$.

For finite execution, we have $\alpha_d \circ F_s = F_d \circ \alpha_d$ in $\mathcal{P}(\Sigma^*) \rightarrow \mathcal{P}(\Sigma \times \Sigma)$.

We can apply directly fixpoint transfer and get that: $\mathcal{DS} \cap (\Sigma \times \Sigma) = \text{lfp } F_d$.

(proof continued on next slide)

Denotational fixpoint semantics (proof cont.)

proof (continued): proof sketch for infinite executions

We have $\alpha_d \circ G_s = G_d \circ \alpha_d$ in $\mathcal{P}(\Sigma^\omega) \rightarrow \mathcal{P}(\Sigma \times \{\circ\})$, where

$$G_d(R) \stackrel{\text{def}}{=} \{(\sigma, \sigma'') \mid \exists \sigma' : \sigma \rightarrow \sigma' \wedge (\sigma', \sigma'') \in R\}.$$

A candidate proof would be to apply a fixpoint transfer theorem to $\mathcal{M}_\infty \cap \Sigma^\omega = \text{gfp } G_s$, in the dual, replacing lfp with gfp, and \cup with \cap .

However, the proof of the theorem, which required α to be continuous, would require α to be co-continuous in the dual, i.e., $\alpha_d(\bigcap_{i \in I} S_i) = \bigcap_{i \in I} \alpha_d(S_i)$.

This does not hold. Consider for example: $I = \mathbb{N}$ and $S_i = \{a^n b^\omega \mid n > i\}$:

$$\bigcap_{i \in \mathbb{N}} S_i = \emptyset, \text{ but } \forall i : \alpha_d(S_i) = \{(a, \circ)\}.$$

We use instead a fixpoint transfer based on Tarski's theorem.

We have $\text{gfp } G_s = \bigcup \{X \mid X \subseteq G_s(X)\}$.

Thus, $\alpha_d(\text{gfp } G_s) = \alpha_d(\bigcup \{X \mid X \subseteq G_s(X)\}) = \bigcup \{\alpha_d(X) \mid X \subseteq G_s(X)\}$ as α_d is a complete \cup morphism. The proof is finished by noting that the commutation $\alpha_d \circ G_s = G_d \circ \alpha_d$ and the Galois embedding (α_d, γ_d) imply that $\{\alpha_d(X) \mid X \subseteq G_s(X)\} = \{\alpha_d(X) \mid \alpha_d(X) \subseteq G_d(\alpha_d(X))\} = \{Y \mid Y \subseteq G_d(Y)\}$.

(the complete proof can be found in [\[Cous02\]](#))

Denotational semantics (example)

```
program
```

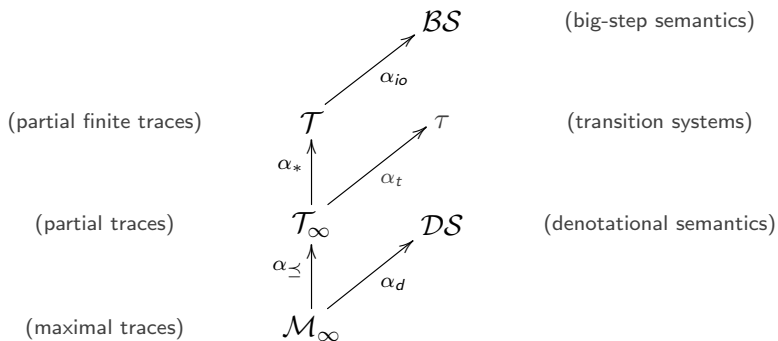
```
 $i \leftarrow [0, +\infty];$   
while  $i > 0$  do  
   $i \leftarrow i - [0, 1];$   
done
```

Denotational semantics \mathcal{DS} :

$$\{(\rho, \rho') \mid \rho(i) \geq 0 \wedge \rho'(i) = 0\} \cup \{(\rho, \circlearrowleft) \mid \rho(i) \geq 0\}.$$

(quite different from the big-step semantics)

Another part of the hierarchy of semantics



See [Cou82] for more semantics in this diagram.

Note: we show transition systems as an abstraction of the partial trace semantics this is left as exercise (see assignment).

Properties and proofs

State properties

State properties

State property: $P \in \mathcal{P}(\Sigma)$.

Verification problem: $\mathcal{R}(\mathcal{I}) \subseteq P$.

(all the states reachable from \mathcal{I} are in P)

Examples:

- absence of blocking: $P \stackrel{\text{def}}{=} \Sigma \setminus \mathcal{B}$,
- the variables remain in a safe range,
- dangerous program locations cannot be reached.

Invariance proof method

Invariance proof method: find an inductive invariant $I \subseteq \Sigma$

- $I \subseteq I$
(contains initial states)
- $\forall \sigma \in I: \sigma \rightarrow \sigma' \implies \sigma' \in I$
(invariant by program transition)

that implies the desired property: $I \subseteq P$.

Link with the state semantics $\mathcal{R}(I)$:

Given $F_{\mathcal{R}}(S) \stackrel{\text{def}}{=} I \cup \text{post}_{\mathcal{T}}(S)$, we have $F_{\mathcal{R}}(I) \subseteq I$
 $\implies I$ is a post-fixpoint of $F_{\mathcal{R}}$.

Recall that $\mathcal{R}(I) = \text{lfp } F_{\mathcal{R}}$
 $\implies \mathcal{R}(I)$ is the tightest inductive invariant.

Link with Hoare logic

Hoare logic: proof method where we

- annotate program points with **local state invariants** in $\mathcal{P}(\Sigma)$
- use logic rules to prove their correctness

$$\frac{}{\{P[e/X]\} X \leftarrow e \{P\}} \quad \frac{\{P\} \text{stat}_1 \{R\} \quad \{R\} \text{stat}_2 \{Q\}}{\{P\} \text{stat}_1; \text{stat}_2 \{Q\}}$$

$$\frac{\{P \wedge b\} \text{stat} \{Q\} \quad P \wedge \neg b \Rightarrow Q}{\{P\} \text{if } b \text{ then } \text{stat} \{Q\}} \quad \frac{\{P \wedge b\} \text{stat} \{P\}}{\{P\} \text{while } b \text{ do } \text{stat} \{P \wedge \neg b\}}$$

$$\frac{\{P\} \text{stat} \{Q\} \quad P' \Rightarrow P \quad Q \Rightarrow Q'}{\{P'\} \text{stat} \{Q'\}}$$

Link with the state semantics $\mathcal{R}(\mathcal{I})$:

Recall the equation system $\forall \ell \in \mathcal{L}: \mathcal{X}_\ell = F_{eq,\ell}(\mathcal{X}_1, \dots, \mathcal{X}_n)$ obtained by partitioning reachability $F_{\mathcal{R}}$ by control point $(\mathcal{P}(\Sigma), \subseteq) \xleftrightarrow[\alpha_{\mathcal{L}}]{\gamma_{\mathcal{L}}} (\mathcal{L} \rightarrow \mathcal{P}(\mathcal{E}), \dot{\subseteq})$.

- any **post-fixpoint** of F_{eq} gives **valid** Hoare triples
- **lfp** F_{eq} gives the **tightest** Hoare triples

Trace properties

Trace properties

Trace property: $P \in \mathcal{P}(\Sigma^\infty)$

Verification problem: $\mathcal{M}_\infty \cap (\mathcal{I} \cdot \Sigma^\infty) \subseteq P$

(or, equivalently, as $\mathcal{M}_\infty \subseteq P'$ where $P' \stackrel{\text{def}}{=} P \cup ((\Sigma \setminus \mathcal{I}) \cdot \Sigma^\infty)$)

Examples:

- **termination**: $P \stackrel{\text{def}}{=} \Sigma^*$,
- **non-termination**: $P \stackrel{\text{def}}{=} \Sigma^\omega$,
- any **state property** $S \subseteq \Sigma$: $P \stackrel{\text{def}}{=} S^\infty$,
- **maximal execution time**: $P \stackrel{\text{def}}{=} \Sigma^{\leq k}$,
- **minimal execution time**: $P \stackrel{\text{def}}{=} \Sigma^{\geq k}$,
- **ordering**, e.g.: $P \stackrel{\text{def}}{=} (\Sigma \setminus \{b\})^* \cdot a \cdot \Sigma^* \cdot b \cdot \Sigma^\infty$.
(a and b occur, and a occurs before b)

Safety properties

Idea: a safety property P models that “nothing bad ever occurs”

- P is provable by exhaustive testing;
(observe the prefix trace semantics: $\mathcal{T}_p(\mathcal{I}) \subseteq P$)
- P is disprovable by finding a single finite execution not in P .

Examples:

- any **state property**: $P \stackrel{\text{def}}{=} S^\infty$ for $S \subseteq \Sigma$,
- **ordering**: $P \stackrel{\text{def}}{=} \Sigma^\infty \setminus ((\Sigma \setminus \{a\})^* \cdot b \cdot \Sigma^\infty)$,
(no b can appear without an a before,
but we can have only a , or neither a nor b)
(not a state property)
- but **termination** $P \stackrel{\text{def}}{=} \Sigma^*$ is **not** a safety property.
(disproving requires exhibiting an *infinite* execution)

Definition of safety properties

Reminder: finite prefix abstraction (simplified to allow ϵ)

$$(\mathcal{P}(\Sigma^\infty), \subseteq) \begin{array}{c} \xleftarrow{\gamma_{*\underline{\preceq}}} \\ \xrightarrow{\alpha_{*\underline{\preceq}}} \end{array} (\mathcal{P}(\Sigma^*), \subseteq)$$

- $\alpha_{*\underline{\preceq}}(T) \stackrel{\text{def}}{=} \{t \in \Sigma^* \mid \exists u \in T : t \underline{\preceq} u\}$
- $\gamma_{*\underline{\preceq}}(T) \stackrel{\text{def}}{=} \{t \in \Sigma^\infty \mid \forall u \in \Sigma^* : u \underline{\preceq} t \implies u \in T\}$

The associated upper closure $\rho_{*\underline{\preceq}} \stackrel{\text{def}}{=} \gamma_{*\underline{\preceq}} \circ \alpha_{*\underline{\preceq}}$ is:

$\rho_{*\underline{\preceq}} = \text{lim} \circ \rho_p$ where:

- $\rho_p(T) \stackrel{\text{def}}{=} \{u \in \Sigma^\infty \mid \exists t \in T : u \underline{\preceq} t\}$,
- $\text{lim}(T) \stackrel{\text{def}}{=} T \cup \{t \in \Sigma^\omega \mid \forall u \in \Sigma^* : u \underline{\preceq} t \implies u \in T\}$.

Definition: $P \in \mathcal{P}(\Sigma^\infty)$ is a **safety property** if $P = \rho_{*\underline{\preceq}}(P)$.

Definition of safety properties (examples)

Definition: $P \subseteq \mathcal{P}(\Sigma^\infty)$ is a **safety property** if $P = \rho_{*\underline{\prec}}(P)$.

Examples and counter-examples:

- state property $P \stackrel{\text{def}}{=} S^\infty$ for $S \subseteq \Sigma$:

$$\rho_P(S^\infty) = \text{lim}(S^\infty) = S^\infty \implies \text{safety};$$

- termination $P \stackrel{\text{def}}{=} \Sigma^*$:

$$\rho_P(\Sigma^*) = \Sigma^*, \text{ but } \text{lim}(\Sigma^*) = \Sigma^\infty \neq \Sigma^* \implies \text{not safety};$$

- even number of steps $P \stackrel{\text{def}}{=} (\Sigma^2)^\infty$:

$$\rho_P((\Sigma^2)^\infty) = \Sigma^\infty \neq (\Sigma^2)^\infty \implies \text{not safety}.$$

Proving safety properties

Invariance proof method: find an **inductive invariant** I

- set of **finite** traces $I \subseteq \Sigma^*$
- $\mathcal{I} \subseteq I$
(contains traces reduced to an initial state)
- $\forall \sigma_0, \dots, \sigma_n \in I: \sigma_n \rightarrow \sigma_{n+1} \implies \sigma_0, \dots, \sigma_n, \sigma_{n+1} \in I$
(invariant by program transition)

and implies the desired property: $I \subseteq P$.

Link with the finite prefix trace semantics $\mathcal{T}_p(\mathcal{I})$:

An inductive invariant is a **post-fixpoint** of F_p : $F_p(I) \subseteq I$

where $F_p(T) \stackrel{\text{def}}{=} \mathcal{I} \cup T \cap \tau$.

$\mathcal{T}_p(\mathcal{I}) = \text{lfp } F_p$ is the **tightest inductive invariant**.

Correctness of the invariant method for safety

Soundness:

if P is a safety property and an inductive invariant I exists
then: $\mathcal{M}_\infty \cap (\mathcal{I} \cdot \Sigma^\infty) \subseteq P$

proof:

Using the Galois connection between \mathcal{M}_∞ and \mathcal{T} , we get:

$$\begin{aligned} \mathcal{M}_\infty \cap (\mathcal{I} \cdot \Sigma^\infty) &\subseteq \rho_{*\underline{\leq}}(\mathcal{M}_\infty \cap (\mathcal{I} \cdot \Sigma^\infty)) = \gamma_{*\underline{\leq}}(\alpha_{*\underline{\leq}}(\mathcal{M}_\infty \cap (\mathcal{I} \cdot \Sigma^\infty))) = \\ &\gamma_{*\underline{\leq}}(\alpha_{*\underline{\leq}}(\mathcal{M}_\infty) \cap (\mathcal{I} \cdot \Sigma^*)) = \gamma_{*\underline{\leq}}(\mathcal{T} \cap (\mathcal{I} \cdot \Sigma^*)) = \gamma_{*\underline{\leq}}(\mathcal{T}_P(\mathcal{I})). \end{aligned}$$

Using the link between invariants and the finite prefix trace semantics, we have:

$$\mathcal{T}_P(\mathcal{I}) \subseteq I \subseteq P.$$

As P is a safety property, $P = \gamma_{*\underline{\leq}}(P)$, so, $\gamma_{*\underline{\leq}}(\mathcal{T}_P(\mathcal{I})) \subseteq \gamma_{*\underline{\leq}}(P) = P$, and so,

$$\mathcal{M}_\infty \cap (\mathcal{I} \cdot \Sigma^\infty) \subseteq P.$$

Completeness: an inductive invariant always exists

proof: $\mathcal{T}_P(\mathcal{I})$ provides an inductive invariant.

Disproving safety properties

Proof method:

A safety property P can be **disproved** by constructing a **finite prefix of execution** that does not satisfy the property:

$$\mathcal{M}_\infty \cap (\mathcal{I} \cdot \Sigma^\infty) \not\subseteq P \implies \exists t \in \mathcal{T}_p(\mathcal{I}): t \notin P$$

proof:

By contradiction, assume that no such trace exists, i.e., $\mathcal{T}_p(\mathcal{I}) \subseteq P$.

We proved in the previous slide that this implies $\mathcal{M}_\infty \cap (\mathcal{I} \cdot \Sigma^\infty) \subseteq P$.

Examples:

- disproving a **state property** $P \stackrel{\text{def}}{=} S^\infty$:
 \implies find a partial execution containing a state in $\Sigma \setminus S$;
- disproving an **order property** $P \stackrel{\text{def}}{=} \Sigma^\infty \setminus ((\Sigma \setminus \{a\})^* \cdot b \cdot \Sigma^\infty)$
 \implies find a partial execution where b appears and not a .

Liveness properties

Idea: liveness property $P \in \mathcal{P}(\Sigma^\infty)$

Liveness properties model that “something good eventually occurs”

- P cannot be proved by testing
(if nothing good happens in a prefix execution,
it can still happen in the rest of the execution)
- disproving P requires exhibiting an infinite execution not in P

Examples:

- **termination:** $P \stackrel{\text{def}}{=} \Sigma^*$,
- **inevitability:** $P \stackrel{\text{def}}{=} \Sigma^* \cdot a \cdot \Sigma^\infty$,
(a eventually occurs in all executions)
- state properties are **not** liveness properties.

Definition of liveness properties

Definition: $P \in \mathcal{P}(\Sigma^\infty)$ is a **liveness property** if $\rho_{*\cup}(P) = \Sigma^\infty$.

Examples and counter-examples:

- termination $P \stackrel{\text{def}}{=} \Sigma^*$:

$$\rho_P(\Sigma^*) = \Sigma^* \text{ and } \lim(\Sigma^*) = \Sigma^\infty \implies \text{liveness};$$

- inevitability: $P \stackrel{\text{def}}{=} \Sigma^* \cdot a \cdot \Sigma^\infty$

$$\rho_P(P) = P \cup \Sigma^* \text{ and } \lim(P \cup \Sigma^*) = \Sigma^\infty \implies \text{liveness};$$

- state property $P \stackrel{\text{def}}{=} S^\infty$ for $S \subseteq \Sigma$:

$$\rho_P(S^\infty) = \lim(S^\infty) = S^\infty \neq \Sigma^\infty \text{ if } S \neq \Sigma \implies \text{not liveness};$$

- maximal execution time $P \stackrel{\text{def}}{=} \Sigma^{\leq k}$:

$$\rho_P(\Sigma^{\leq k}) = \lim(\Sigma^{\leq k}) = \Sigma^{\leq k} \neq \Sigma^\infty \implies \text{not liveness};$$

- the only property which is both safety and liveness is Σ^∞ .

Proving liveness properties

Variance proof method: (informal definition)

Find a **decreasing quantity** until something good happens.

Example: termination proof

- find $f : \Sigma \rightarrow \mathcal{S}$ where $(\mathcal{S}, \sqsubseteq)$ is **well-ordered**;
(f is called a “ranking function”)
- $\sigma \in \mathcal{B} \implies f = \min \mathcal{S}$;
- $\sigma \rightarrow \sigma' \implies f(\sigma') \sqsubset f(\sigma)$.

(f counts the number of steps remaining before termination)

Disproving liveness properties

Property:

If P is a liveness property, then $\forall t \in \Sigma^*: \exists u \in P: t \preceq u$.

proof:

By definition of liveness, $\rho_{*\preceq}(P) = \Sigma^\infty$, so $t \in \rho_{*\preceq}(P) = \lim(\alpha_p(P))$.

As $t \in \Sigma^*$ and \lim only adds infinite traces, $t \in \alpha_p(P)$.

By definition of α_p , $\exists u \in P: t \preceq u$.

Consequence:

- liveness cannot be disproved by testing.

Trace topology

A topology on a set can be defined as:

- either a family of open sets (closed under union)
- or family of closed sets (closed under intersection)

Trace topology: on sets of traces in Σ^∞

- the **closed sets** are: $\mathcal{C} \stackrel{\text{def}}{=} \{ P \in \mathcal{P}(\Sigma^\infty) \mid P \text{ is a safety property} \}$
- the open sets can be derived as $\mathcal{O} \stackrel{\text{def}}{=} \{ \Sigma^\infty \setminus c \mid c \in \mathcal{C} \}$

Topological closure: $\rho : \mathcal{P}(X) \rightarrow \mathcal{P}(X)$

- $\rho(x) \stackrel{\text{def}}{=} \bigcap \{ c \in \mathcal{C} \mid x \subseteq c \}$ (upper closure operator in $(\mathcal{P}(X), \subseteq)$)
- on our trace topology, $\rho = \rho_{*\preceq}$.

Dense sets:

- $x \subseteq X$ is dense if $\rho(x) = X$;
- on our trace topology, dense sets are **liveness properties**.

Decomposition theorem

Theorem: decomposition on a topological space

Any set $x \subseteq X$ is the **intersection** of a **closed** set and a **dense** set.

proof:

We have $x = \rho(x) \cap (x \cup (X \setminus \rho(x)))$. Indeed:

$$\rho(x) \cap (x \cup (X \setminus \rho(x))) = (\rho(x) \cap x) \cup (\rho(x) \cap (X \setminus \rho(x))) = \rho(x) \cap x = x \text{ as } x \subseteq \rho(x).$$

- $\rho(x)$ is closed
- $x \cup (X \setminus \rho(x))$ is dense because:

$$\begin{aligned} \rho(x \cup (X \setminus \rho(x))) &\supseteq \rho(x) \cup \rho(X \setminus \rho(x)) \\ &\supseteq \rho(x) \cup (X \setminus \rho(x)) \\ &= X \end{aligned}$$

Consequence: on trace properties

Every trace property is the **conjunction** of a **safety** property and a **liveness** property.

proving a trace property can be decomposed into a soundness proof and a liveness proof

Beyond trace properties

Some verification problems cannot be expressed as $\mathcal{M}_\infty \subseteq P$

Examples:

- **Program equivalence**

Do two programs (Σ, τ_1) and (Σ, τ_2) have the exact same executions?
i.e., $\mathcal{M}_\infty[\tau_1] = \mathcal{M}_\infty[\tau_2]$

- **Non-interference**

Does changing the initial value of X change its final value?

$\forall \sigma_0, \dots, \sigma_n \in \mathcal{M}_\infty: \forall \sigma'_0: \sigma_0 \equiv \sigma'_0 \implies$

$\exists \sigma'_0, \dots, \sigma'_m \in \mathcal{M}_\infty: \sigma'_m \equiv \sigma_m$

where $(\ell, \rho) \equiv (\ell', \rho') \iff \ell = \ell' \wedge \forall V \neq X: \rho(V) = \rho'(V)$

New verification problem: $\mathcal{M}_\infty \in H$ where $H \in \mathcal{P}(\mathcal{P}(\Sigma^\infty))$

- generalizes trace properties: $\mathcal{M}_\infty \subseteq P$ reduces to $\mathcal{M}_\infty \in \mathcal{P}(P)$;
- program equivalence is $\mathcal{M}_\infty[\tau_1] \in \{\mathcal{M}_\infty[\tau_2]\}$; etc.

Reading assignment: hyperproperties.

Bibliography

Bibliography

[Bour93] **F. Bourdoncle**. *Abstract debugging of higher-order imperative languages*. In PLDI, 46-55, ACM Press, 1993.

[Cous02] **P. Cousot**. *Constructive design of a hierarchy of semantics of a transition system by abstract interpretation*. In Theoretical Comp. Sc., 277(1-2):47-103.

[Plot81] **G. Plotkin**. *The origins of structural operational semantics*. In J. of Logic and Algebraic Prog., 60:60-61, 1981.