Program Semantics

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

Antoine Miné

year 2015–2016

course 03
23 September 2015

Goal

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:

- introduction: classic examples of program semantics
- transition systems
- state semantics (forward and backward)
- trace semantics (finite and infinite)
- relational semantics
- state and trace properties

Flavors of program semantics

Small-step operational semantics of the $\lambda-$calculus

Goal:

Illustrate through a simple example ($\lambda-$calculus) different favors and levels of semantics.

They feature some notion of states and transitions.

$\Longrightarrow$ justifies transition systems as a universal model of semantics

Example: $\lambda-$calcul

syntax: $\lambda-$terms

t	$::=$	x	*(variable)*
	$\mid$	$\lambda x.t$	*(abstraction)*
	$\mid$	$t\ u$	*(application)*

Small-step operational semantics of the $\lambda-$calculus

Small-step operational semantics: (call-by-value)

$$\frac{}{(\lambda x.M)N \rightsquigarrow M[x/N]} \qquad \frac{M \rightsquigarrow M'}{M\ N \rightsquigarrow M'\ N} \qquad \frac{N \rightsquigarrow N'}{M\ N \rightsquigarrow M\ N'}$$

Models program execution as a sequence of term-rewriting $\rightsquigarrow$ exposing each transition (low level).

Big-step operational semantics of the λ−calculus

Big-step operational semantics: (call-by-value)

$$\frac{}{\lambda x.M \Downarrow \lambda x.M} \qquad \frac{M \Downarrow \lambda x.L \quad N \Downarrow V_2 \quad L[x/V_2] \Downarrow V_1}{M\ N \Downarrow V_1}$$

$t \Downarrow u$ associates to a term t its full evaluation u,
abstracting away intermediate steps (higher level).

Denotational semantics of the $\lambda-$calculus

Denotational semantics:

$$
\begin{array}{lcl}
[\![x]\!]_\rho & \stackrel{\text{def}}{=} & \rho(x) \\
[\![t\, u]\!]_\rho & \stackrel{\text{def}}{=} & [\![t]\!]_\rho([\![u]\!]_\rho) \\
[\![\lambda x.t]\!]_\rho & \stackrel{\text{def}}{=} & \lambda v.[\![t]\!]_{\rho[x \mapsto v]}
\end{array}
$$

The semantics $[\![t]\!]_\rho$ of a term t in an environment ρ is given as an element of a Scott domain $\mathcal{D}$.

- $\mathcal{D}$ should satisfy the domain equation: $\mathcal{D} \simeq \mathcal{D} \stackrel{c}{\to} \mathcal{D}_\perp$
  (CPO $\mathcal{D}$ closed by continuous functions from $\mathcal{D}$ to the lifted CPO $\mathcal{D}_\perp$)
- The semantics of a program function is a mathematical function.
  (very high level)

Abstract machine semantics of the $\lambda-$calculus

Krivine abstract machine: (call-by-value)

- variables in $\lambda-$terms are replaced with De Bruijn indices
  ($x \mapsto$ number of nested λ to reach λx)

- $\lambda-$terms are compiled into sequences of instructions:

$$\begin{array}{lcl} \mathcal{I} & \stackrel{\text{def}}{=} & \mathit{Grab} \mid \mathit{Access}(\mathbb{Z}) \mid \mathit{Push}(\mathcal{I}) \mid \mathcal{I};\mathcal{I} \\ [\![\cdot]\!] & \in & t \to \mathcal{I} \\ [\![n]\!] & \stackrel{\text{def}}{=} & \mathit{Access}(n) \\ [\![\lambda N]\!] & \stackrel{\text{def}}{=} & \mathit{Grab}; [\![N]\!] \\ [\![N\ M]\!] & \stackrel{\text{def}}{=} & \mathit{Push}([\![M]\!]); [\![N]\!] \end{array}$$

Abstract machine semantics of the $\lambda-$calculus

- instructions are executed over configurations (C, e, s)
  - C: sequence of instructions to execute
  - e: environment
    s: stack $=$ list of pairs of (C, e) (closures)

  with transitions:
  - $\langle Access(0) \cdot C,\ (C_0, e_0) \cdot e,\ s \rangle \to \langle C_0, e_0, s \rangle$
  - $\langle Access(n+1) \cdot C,\ (C_0, e_0) \cdot e,\ s \rangle \to \langle Access(n), e, s \rangle$
  - $\langle Push(C') \cdot C,\ e,\ s \rangle \to \langle C,\ e,\ (C', e) \cdot s \rangle$
  - $\langle Grab \cdot C,\ e,\ (C_0, e_0) \cdot s \rangle \to \langle C,\ (s_0, e_0) \cdot e,\ s \rangle$

$\Longrightarrow$ very low level. (but very efficient)

Transition systems

Transition systems: definition

Language-neutral formalism to discuss about program semantics.

Transition system: (Σ, τ)

- set of states Σ,

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

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

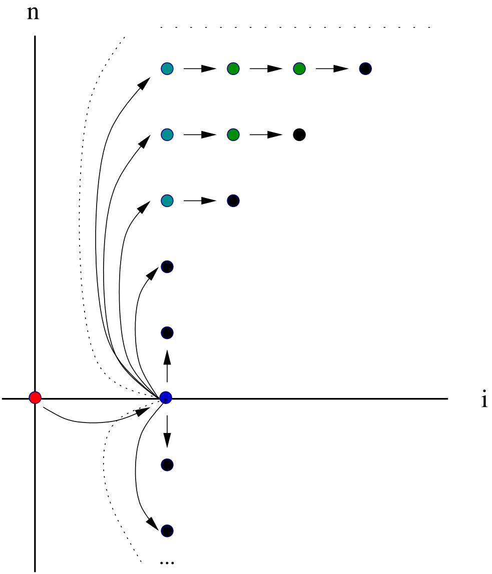
$(\sigma, \sigma') \in \tau$ is noted $\sigma \to \sigma'$:

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

Transition system: example

$i \leftarrow 2;$
$n \leftarrow [-\infty, +\infty];$
while $i < n$ **do**
  if ? **then**
    $i \leftarrow i + 1$

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



From programs to transition systems

Example: on a simple imperative language.

Language syntax

${}^{\ell}stat^{\ell}$	$::=$	${}^{\ell}X \leftarrow expr^{\ell}$	*(assignment)*
	$\mid$	${}^{\ell}\textbf{if}\ expr \bowtie 0\ \textbf{then}\ {}^{\ell}stat^{\ell}$	*(conditional)*
	$\mid$	${}^{\ell}\textbf{while}\ {}^{\ell}expr \bowtie 0\ \textbf{do}\ {}^{\ell}stat^{\ell}$	*(loop)*
	$\mid$	${}^{\ell}stat; {}^{\ell}stat^{\ell}$	*(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,
- $\bowtie \in \{=, \leq, \ldots\}$, 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 $\mathcal{E} \stackrel{\text{def}}{=} \mathbb{V} \to \mathbb{R}$.

From programs to transition systems

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

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

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

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

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

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

State semantics

States and state operators

Initial, final, blocking 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 $\mathcal{B} \stackrel{\text{def}}{=} \{\, \sigma \mid \forall \sigma' \in \Sigma \colon \sigma \not\to \sigma' \,\}$,
- model 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:
$\tau' \stackrel{\text{def}}{=} \tau \cup \{\, (\sigma, \sigma) \mid \sigma \in \mathcal{B} \,\}$. (add self-loops)

Post-image, pre-image

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

- successors: (forward, post-image)
  $$\mathsf{post}_\tau(S) \stackrel{\text{def}}{=} \{\, \sigma' \mid \exists \sigma \in S: \sigma \to \sigma' \,\}$$
- predecessors: (backward, pre-image)
  $$\mathsf{pre}_\tau(S) \stackrel{\text{def}}{=} \{\, \sigma \mid \exists \sigma' \in S: \sigma \to \sigma' \,\}$$

post_τ and pre_τ are complete $\cup-$morphisms in $(\mathcal{P}(\Sigma), \subseteq, \cup, \cap, \emptyset, \Sigma)$.
$(\mathsf{post}_\tau(\cup_{i \in I} S_i) = \cup_{i \in I}\ \mathsf{post}_\tau(S_i),\ \mathsf{pre}_\tau(\cup_{i \in I} S_i) = \cup_{i \in I}\ \mathsf{pre}_\tau(S_i))$

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

We have: $\mathsf{pre}_\tau(S) = \cup \{\, \mathsf{pre}_\tau(\{s\}) \mid s \in S \,\}$ and $\mathsf{post}_\tau(S) = \cup \{\, \mathsf{post}_\tau(\{s\}) \mid s \in S \,\}$.

Dual images

Dual post-images and pre-images:

- $\widetilde{\text{pre}}_\tau(S) \stackrel{\text{def}}{=} \{\, \sigma \mid \forall \sigma' \colon \sigma \to \sigma' \implies \sigma' \in S \,\}$

  (states such that all successors satisfy S)

- $\widetilde{\text{post}}_\tau(S) \stackrel{\text{def}}{=} \{\, \sigma' \mid \forall \sigma \colon \sigma \to \sigma' \implies \sigma \in S \,\}$

  (states such that all predecessors satisfy S)

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

Correspondences between images and dual images

$$\begin{array}{lcl}
\mathrm{post}_\tau(S) & \stackrel{\mathrm{def}}{=} & \{\, \sigma' \mid \exists \sigma \in S \colon \sigma \to \sigma' \,\} \\
\mathrm{pre}_\tau(S) & \stackrel{\mathrm{def}}{=} & \{\, \sigma \mid \exists \sigma' \in S \colon \sigma \to \sigma' \,\} \\
\widetilde{\mathrm{pre}}_\tau(S) & \stackrel{\mathrm{def}}{=} & \{\, \sigma \mid \forall \sigma' \colon \sigma \to \sigma' \implies \sigma' \in S \,\} \\
\widetilde{\mathrm{post}}_\tau(S) & \stackrel{\mathrm{def}}{=} & \{\, \sigma' \mid \forall \sigma \colon \sigma \to \sigma' \implies \sigma \in S \,\}
\end{array}$$

We have the following correspondences:

- inverse

$$\mathrm{pre}_\tau = \mathrm{post}_{(\tau^{-1})} \qquad \mathrm{post}_\tau = \mathrm{pre}_{(\tau^{-1})}$$

$$\widetilde{\mathrm{pre}}_\tau = \widetilde{\mathrm{post}}_{(\tau^{-1})} \qquad \widetilde{\mathrm{post}}_\tau = \widetilde{\mathrm{pre}}_{(\tau^{-1})}$$

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

Correspondences between images and dual images

$$\begin{array}{lcl}
\mathrm{post}_\tau(S) & \stackrel{\text{def}}{=} & \{\,\sigma' \mid \exists \sigma \in S \colon \sigma \to \sigma'\,\} \\
\mathrm{pre}_\tau(S) & \stackrel{\text{def}}{=} & \{\,\sigma \mid \exists \sigma' \in S \colon \sigma \to \sigma'\,\} \\
\widetilde{\mathrm{pre}}_\tau(S) & \stackrel{\text{def}}{=} & \{\,\sigma \mid \forall \sigma' \colon \sigma \to \sigma' \implies \sigma' \in S\,\} \\
\widetilde{\mathrm{post}}_\tau(S) & \stackrel{\text{def}}{=} & \{\,\sigma' \mid \forall \sigma \colon \sigma \to \sigma' \implies \sigma \in S\,\}
\end{array}$$

We have the following correspondences:

- Galois connections

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

$$(\mathcal{P}(\Sigma),\subseteq) \xleftarrow[\mathrm{pre}_\tau]{\widetilde{\mathrm{post}}_\tau} \!\!\!\!\xrightarrow{} (\mathcal{P}(\Sigma),\subseteq).$$

proof:

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

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

Reachability state semantics

Forward reachability

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

$$\mathcal{R}(\mathcal{I}) \stackrel{\text{def}}{=} \{\, \sigma \mid \exists n \geq 0, \sigma_0, \ldots, \sigma_n \colon \sigma_0 \in \mathcal{I}, \sigma = \sigma_n, \forall i \colon \sigma_i \to \sigma_{i+1} \,\}$$
$$= \bigcup_{n \geq 0} \mathrm{post}_\tau^n(\mathcal{I})$$

(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:

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

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

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

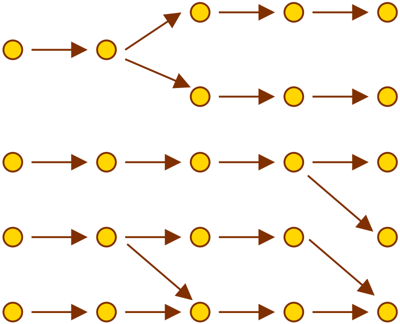
- $F_\mathcal{R}^0(\emptyset) = \emptyset$
- assuming the property at n,

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

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

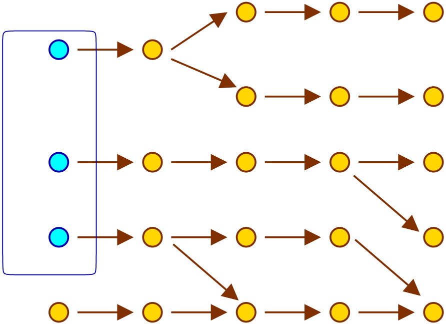
The proof is similar for the alternate form, given that $\text{lfp}_\mathcal{I}\, G_\mathcal{R} = \cup_{n\in\mathbb{N}} G_\mathcal{R}^n(\mathcal{I})$ and $G_\mathcal{R}^n(\mathcal{I}) = F_\mathcal{R}^{n+1}(\emptyset) = \cup_{i\leq n} \text{post}_\tau^i(\mathcal{I})$.

Forward reachability: graphical illustration



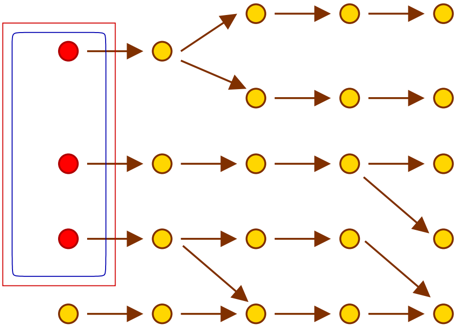
Transition system.

Forward reachability: graphical illustration



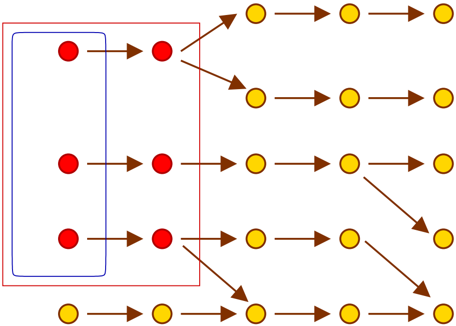
Initial states $\mathcal{I}$.

Forward reachability: graphical illustration



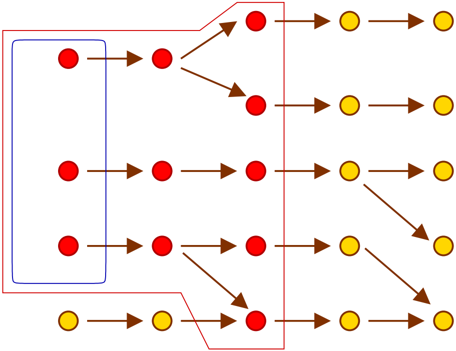
Iterate $F^1_{\mathcal{R}}(\mathcal{I})$.

Forward reachability: graphical illustration



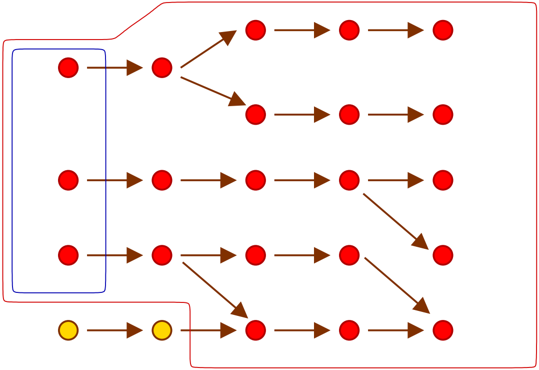
Iterate $F_{\mathcal{R}}^2(\mathcal{I})$.

Forward reachability: graphical illustration



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

Forward reachability: graphical illustration



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

Forward reachability: applications

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

example

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

  - initial states $\mathcal{I}$: $j \in [0, 10]$ at control state $\bullet$,
  - final states $\mathcal{F}$: any memory state at control state $\bullet$,
  - $\Longrightarrow \mathcal{R}(\mathcal{I}) \cap \mathcal{F}$: control at $\bullet$, $i = 100$, and $j \in [0, 110]$.

- Prove the absence of run-time error: $\mathcal{R}(\mathcal{I}) \cap \mathcal{B} \subseteq \mathcal{F}$.

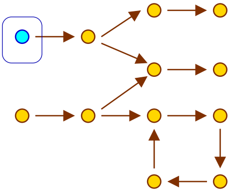
(never block except when reaching the end of the program)

Multiple forward fixpoints

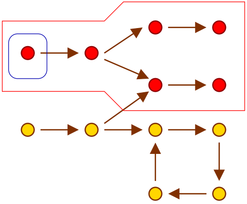
Recall: $\mathcal{R}(\mathcal{I}) = \text{lfp } F_{\mathcal{R}}$ where $F_{\mathcal{R}}(S) \stackrel{\text{def}}{=} \mathcal{I} \cup \text{post}_\tau(S)$.

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

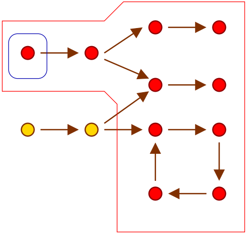
Example:



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 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: $\mathcal{P}(\Sigma) \simeq \mathcal{L} \to \mathcal{P}(\mathcal{E})$.

We have a Galois isomorphism:

$$(\mathcal{P}(\Sigma), \subseteq) \xleftrightarrows[\alpha_{\mathcal{L}}]{\gamma_{\mathcal{L}}} (\mathcal{L} \to \mathcal{P}(\mathcal{E}), \dot{\subseteq})$$

- $X \dot{\subseteq} Y \stackrel{\text{def}}{\iff} \forall \ell \in \mathcal{L}: X(\ell) \subseteq Y(\ell)$
- $\alpha_{\mathcal{L}}(S) \stackrel{\text{def}}{=} \lambda \ell.\{\, \rho \,|\, (\ell, \rho) \in S \,\}$
- $\gamma_{\mathcal{L}}(X) \stackrel{\text{def}}{=} \{\, (\ell, \rho) \,|\, \ell \in \mathcal{L}, \rho \in X(\ell) \,\}$

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 $\alpha_{\mathcal{L}}(\mathcal{R}(\mathcal{I})) : \mathcal{L} \to \mathcal{P}(\mathcal{E})$

- introduce variables: $\mathcal{X}_\ell = (\alpha_{\mathcal{L}}(\mathcal{R}(\mathcal{I})))(\ell) \in \mathcal{P}(\mathcal{E})$,
- decompose the fixpoint equation $F_{\mathcal{R}}(S) = \mathcal{I} \cup \text{post}_\tau(S)$ on $\mathcal{L}$: $\alpha_{\mathcal{L}} \circ F_{\mathcal{R}} \circ \gamma_{\mathcal{L}}$ gives an equation system on $(\mathcal{X}_\ell)_{\ell \in \mathcal{L}}$.

Example:

$^{\ell 1}$ $i \leftarrow 2;$
$^{\ell 2}$ $n \leftarrow [-\infty, +\infty];$
$^{\ell 3}$ **while** $^{\ell 4}$ $i < n$ **do**
  $^{\ell 5}$ **if** $[0,1] = 0$ **then**
    $^{\ell 6}$ $i \leftarrow i + 1$
  $^{\ell 7}$
$^{\ell 8}$

$$
\begin{aligned}
\mathcal{X}_1 &= \mathcal{I}_1 \\
\mathcal{X}_2 &= \mathsf{C}[\![\, i \leftarrow 2 \,]\!]\, \mathcal{X}_1 \\
\mathcal{X}_3 &= \mathsf{C}[\![\, n \leftarrow [-\infty, +\infty] \,]\!]\, \mathcal{X}_2 \\
\mathcal{X}_4 &= \mathcal{X}_3 \cup \mathcal{X}_7 \\
\mathcal{X}_5 &= \mathsf{C}[\![\, i < n \,]\!]\, \mathcal{X}_4 \\
\mathcal{X}_6 &= \mathcal{X}_5 \\
\mathcal{X}_7 &= \mathcal{X}_5 \cup \mathsf{C}[\![\, i \leftarrow i + 1 \,]\!]\, \mathcal{X}_6 \\
\mathcal{X}_8 &= \mathsf{C}[\![\, i \geq n \,]\!]\, \mathcal{X}_4
\end{aligned}
$$

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

Forward reachability equation system: construction

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

$$eq({}^{\ell1}X \leftarrow e^{\ell2}) \stackrel{\text{def}}{=} \{\, \mathcal{X}_{\ell2} = \mathsf{C}[\![\, X \leftarrow e \,]\!]\, \mathcal{X}_{\ell1} \,\}$$

$$eq({}^{\ell1}\textbf{if } e \bowtie 0 \textbf{ then } {}^{\ell2}s^{\ell3}) \stackrel{\text{def}}{=}$$
$$\{\, \mathcal{X}_{\ell2} = \mathsf{C}[\![\, e \bowtie 0 \,]\!]\, \mathcal{X}_{\ell1},\ \mathcal{X}_{\ell3} = \mathcal{X}_{\ell3'} \cup \mathsf{C}[\![\, e \not\bowtie 0 \,]\!]\, \mathcal{X}_{\ell1} \,\} \cup eq({}^{\ell2}s^{\ell3'})$$

$$eq({}^{\ell1}\textbf{while } {}^{\ell2}e \bowtie 0 \textbf{ do } {}^{\ell3}s^{\ell4}) \stackrel{\text{def}}{=}$$
$$\{\, \mathcal{X}_{\ell2} = \mathcal{X}_{\ell1} \cup \mathcal{X}_{\ell4'},\ \mathcal{X}_{\ell3} = \mathsf{C}[\![\, e \bowtie 0 \,]\!]\, \mathcal{X}_{\ell2},\ \mathcal{X}_{\ell4} = \mathsf{C}[\![\, e \not\bowtie 0 \,]\!]\, \mathcal{X}_{\ell2} \,\} \cup$$
$$eq({}^{\ell3}s^{\ell4'})$$

$$eq({}^{\ell1}s_1;\, {}^{\ell2}s_2{}^{\ell3}) \stackrel{\text{def}}{=} eq({}^{\ell1}s_1{}^{\ell2}) \cup ({}^{\ell2}s_2{}^{\ell3})$$

where:

- $\mathcal{X}^{\ell3'}$, $\mathcal{X}^{\ell4'}$ are fresh variables storing intermediate results
- $\mathsf{C}[\![\, X \leftarrow e \,]\!]\, \mathcal{X} \stackrel{\text{def}}{=} \{\, \rho[X \mapsto v] \mid \rho \in \mathcal{X},\ v \in \mathsf{E}[\![\, e \,]\!]\, \rho \,\}$
  $\mathsf{C}[\![\, e \bowtie 0 \,]\!]\, \mathcal{X} \stackrel{\text{def}}{=} \{\, \rho \in \mathcal{X} \mid \exists v \in \mathsf{E}[\![\, \rho \,]\!]\, \rho \colon v \bowtie 0 \,\}$

Co-reachability state semantics

Backward reachability

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

$$\mathcal{C}(\mathcal{F}) \stackrel{\text{def}}{=} \{\, \sigma \mid \exists n \geq 0, \sigma_0, \ldots, \sigma_n \colon \sigma = \sigma_0, \sigma_n \in \mathcal{F}, \forall i \colon \sigma_i \to \sigma_{i+1} \,\}$$
$$= \bigcup_{n \geq 0} \mathrm{pre}_\tau^n(\mathcal{F})$$

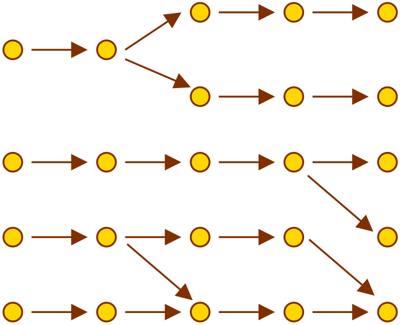
$\mathcal{C}(\mathcal{F})$ can also be expressed in fixpoint form:

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

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

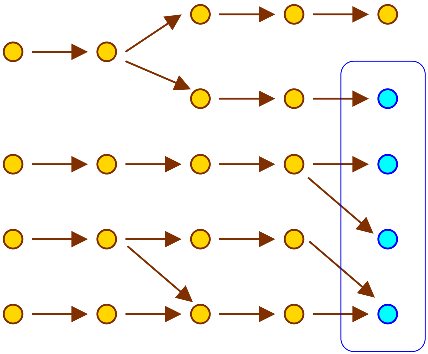
Justification: $\mathcal{C}(\mathcal{F})$ in τ is exactly $\mathcal{R}(\mathcal{F})$ in τ^{-1}.

Backward reachability: graphical illustration



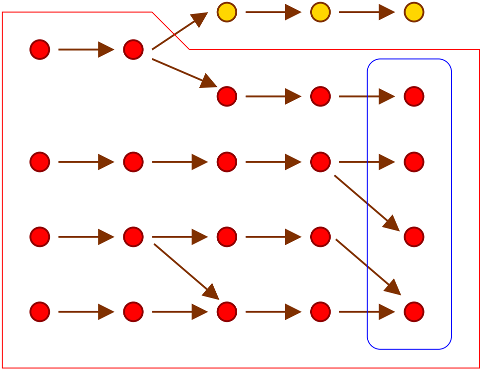
Transition system.

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
  $\Longrightarrow$ abstract debugging [Bour93].

Backward reachability equation system: example

Principle:

Use $(\mathcal{P}(\Sigma), \subseteq) \xleftarrow[\alpha_{\mathcal{L}}]{\gamma_{\mathcal{L}}} (\mathcal{L} \to \mathcal{P}(\mathcal{E}), \dot{\subseteq})$ on $F_{\mathcal{C}}(S) \stackrel{\text{def}}{=} \mathcal{F} \cup \mathrm{pre}_\tau(S)$
to derive an equation system $\alpha_{\mathcal{L}} \circ F_{\mathcal{C}} \circ \gamma_{\mathcal{L}}$.

Example:

$\ell 1$ $i \leftarrow 2$;
$\ell 2$ $n \leftarrow [-\infty, +\infty]$;
$\ell 3$ **while** $\ell 4$ $i < n$ **do**
  $\ell 5$ **if** $[0,1] = 0$ **then**
    $\ell 6$ $i \leftarrow i + 1$
  $\ell 7$
$\ell 8$

$$
\begin{aligned}
\mathcal{X}_1 &= \mathsf{C}[\![\, i \to 2 \,]\!]\, \mathcal{X}_2 \\
\mathcal{X}_2 &= \mathsf{C}[\![\, n \to [-\infty, +\infty] \,]\!]\, \mathcal{X}_3 \\
\mathcal{X}_3 &= \mathcal{X}_4 \\
\mathcal{X}_4 &= \mathsf{C}[\![\, i < n \,]\!]\, \mathcal{X}_5 \cup \mathsf{C}[\![\, i \geq n \,]\!]\, \mathcal{X}_8 \\
\mathcal{X}_5 &= \mathcal{X}_6 \cup \mathcal{X}_7 \\
\mathcal{X}_6 &= \mathsf{C}[\![\, i \to i + 1 \,]\!]\, \mathcal{X}_7 \\
\mathcal{X}_7 &= \mathcal{X}_4 \\
\mathcal{X}_8 &= \mathcal{F}_8
\end{aligned}
$$

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

Pre-condition state semantics

Sufficient preconditions

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

$$\mathcal{S}(\mathcal{Y}) \stackrel{\text{def}}{=} \{\, \sigma \mid \forall n \geq 0, \sigma_0, \ldots, \sigma_n \colon (\sigma = \sigma_0 \land \forall i \colon \sigma_i \to \sigma_{i+1}) \implies \sigma_n \in \mathcal{Y} \,\}$$
$$= \bigcap_{n \geq 0} \widetilde{\text{pre}}_\tau^n(\mathcal{Y})$$

$\mathcal{S}(\mathcal{Y})$ 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:
$F_\mathcal{S}(\cap_{i \in I} A_i) = \cap_{i \in I} F_\mathcal{S}(A_i)$.
By Kleene's theorem in the dual, gfp $F_\mathcal{S} = \cap_{n \in \mathbb{N}} F_\mathcal{S}^n(\Sigma)$.
We would prove by recurrence that $F_\mathcal{S}^n(\Sigma) = \cap_{i < n} \widetilde{\text{pre}}_\tau^i(\mathcal{Y})$.

Sufficient preconditions and reachability

Correspondence with reachability:

We have a Galois connection:

$$(\mathcal{P}(\Sigma), \subseteq) \underset{\mathcal{R}}{\overset{\mathcal{S}}{\leftrightarrows}} (\mathcal{P}(\Sigma), \subseteq)$$

- $\mathcal{R}(\mathcal{I}) \subseteq \mathcal{Y} \iff \mathcal{I} \subseteq \mathcal{S}(\mathcal{Y})$
- so $\mathcal{S}(\mathcal{Y}) = \bigcup \{ X \mid \mathcal{R}(X) \subseteq \mathcal{Y} \}$

  ($\mathcal{S}(\mathcal{Y})$ 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 $\mathcal{R}(\mathcal{I}) = \mathrm{lfp}_{\mathcal{I}}\, G_{\mathcal{R}}$ where $G_{\mathcal{R}}(S) = S \cup \mathrm{post}_\tau(S)$.
Likewise, $\mathcal{S}(\mathcal{Y}) = \mathrm{gfp}_{\mathcal{Y}}\, G_{\mathcal{S}}$ where $G_{\mathcal{S}}(S) = S \cap \widetilde{\mathrm{pre}}_\tau(S)$.

Recall the Galois connection $(\mathcal{P}(\Sigma), \subseteq) \underset{\mathrm{post}_\tau}{\overset{\widetilde{\mathrm{pre}}_\tau}{\leftrightarrows}} (\mathcal{P}(\Sigma), \subseteq)$.

As a consequence $(\mathcal{P}(\Sigma), \subseteq) \underset{G_{\mathcal{R}}}{\overset{G_{\mathcal{S}}}{\leftrightarrows}} (\mathcal{P}(\Sigma), \subseteq)$.

The Galois connection can be lifted to fixpoint operators:

$(\mathcal{P}(\Sigma), \subseteq) \underset{x \mapsto \mathrm{lfp}_x\, G_{\mathcal{R}}}{\overset{x \mapsto \mathrm{gfp}_x\, G_{\mathcal{S}}}{\leftrightarrows}} (\mathcal{P}(\Sigma), \subseteq)$.

Exercise: complete the proof sketch.

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

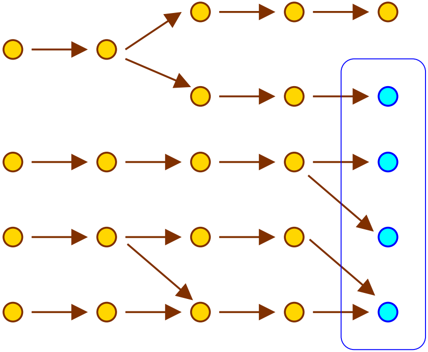
$\bullet$ $i \leftarrow 0$;
while $i < 100$ **do**
  $i \leftarrow i + 1$;
  $j \leftarrow j + [0, 1]$
done $\bullet$

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

  (note that $\mathcal{I} \cap \mathcal{C}(\mathcal{F} \cup (\Sigma \setminus \mathcal{B}))$ gives $\mathcal{I}$)

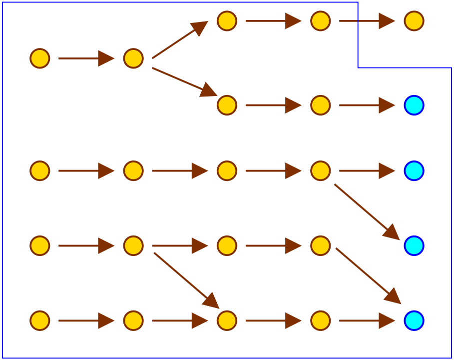
Applications: infer contracts; optimize (hoist) tests;
infer counter-examples.

Sufficient preconditions: graphical illustration



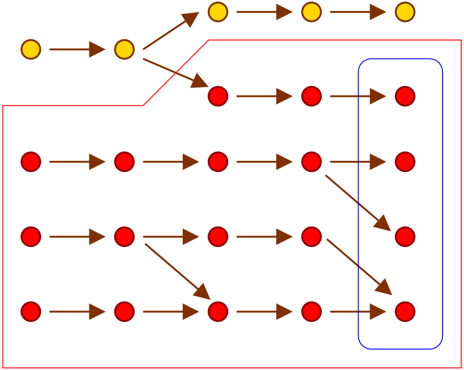
Final states $\mathcal{F}$.

Sufficient preconditions: graphical illustration



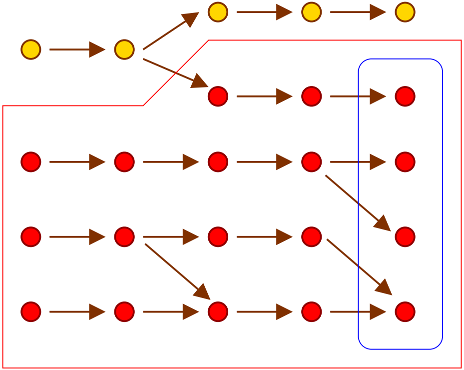
Set of final or non-blocking states $\mathcal{Y} = \mathcal{F} \cup (\Sigma \setminus \mathcal{B})$.

Sufficient preconditions: graphical illustration

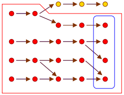


Sufficient preconditions $\mathcal{S}(\mathcal{Y})$.

Sufficient preconditions: graphical illustration



Sufficient preconditions $\mathcal{S}(\mathcal{Y})$.



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

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

Sufficient precondition equation system: example

Principle:

use $(\mathcal{P}(\Sigma), \subseteq) \xleftarrow[\alpha_\mathcal{L}]{\gamma_\mathcal{L}} (\mathcal{L} \to \mathcal{P}(\mathcal{E}), \dot{\subseteq})$ on $F_\mathcal{S}(S) \stackrel{\text{def}}{=} \mathcal{Y} \cap \widetilde{\text{pre}}_\tau(S)$

to derive an equation system $\alpha_\mathcal{L} \circ F_\mathcal{S} \circ \gamma_\mathcal{L}$

Example:

```
ℓ1 i ← 2;
ℓ2 n ← [−∞, +∞];
ℓ3 while ℓ4 i < n do
     ℓ5 if [0, 1] = 0 then
          ℓ6 i ← i + 1
     ℓ7
ℓ8
```

$$
\begin{aligned}
\mathcal{X}_1 &= \overleftarrow{C}[\![\, i \leftarrow 2 \,]\!]\, \mathcal{X}_2 \\
\mathcal{X}_2 &= \overleftarrow{C}[\![\, n \leftarrow [-\infty, +\infty] \,]\!]\, \mathcal{X}_3 \\
\mathcal{X}_3 &= \mathcal{X}_4 \\
\mathcal{X}_4 &= \overleftarrow{C}[\![\, i < n \,]\!]\, \mathcal{X}_5 \cap \overleftarrow{C}[\![\, i \geq n \,]\!]\, \mathcal{X}_8 \\
\mathcal{X}_5 &= \mathcal{X}_6 \cap \mathcal{X}_7 \\
\mathcal{X}_6 &= \overleftarrow{C}[\![\, i \leftarrow i + 1 \,]\!]\, \mathcal{X}_7 \\
\mathcal{X}_7 &= \mathcal{X}_4 \\
\mathcal{X}_8 &= \mathcal{F}_8
\end{aligned}
$$

- "stay in" states $\mathcal{Y} \stackrel{\text{def}}{=} \{ (\ell, \rho) \mid \ell \neq \ell 8 \vee \rho \in \mathcal{F}_8 \}$ for some $\mathcal{F}_8 \subseteq \mathcal{E}$,
- $\overleftarrow{C}[\![\, \cdot \,]\!]$ is the Galois adjoint of $C[\![\, \cdot \,]\!]$.

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)
- $\sigma_0, \ldots, \sigma_{n-1}$: trace of length n
- $\sigma_0, \ldots, \sigma_n, \ldots$: infinite trace (length ω)

Trace sets:

- Σ^n: the set of traces of length n
- $\Sigma^{\leq n} \stackrel{\text{def}}{=} \cup_{i \leq n} \Sigma^i$: the set of traces of length at most n
- $\Sigma^* \stackrel{\text{def}}{=} \cup_{i \in \mathbb{N}} \Sigma^i$: 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

Trace operations

Operations on traces:

- length: $|t| \in \mathbb{N} \cup \{\omega\}$ of a trace $t \in \Sigma^\infty$
- concatenation $\cdot$
  - $(\sigma_0, \ldots, \sigma_n) \cdot (\sigma'_0, \ldots) \stackrel{\text{def}}{=} \sigma_0, \ldots, \sigma_n, \sigma'_0, \ldots$
    (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 $\frown$
  - $(\sigma_0, \ldots, \sigma_n) \frown (\sigma'_0, \sigma'_1 \ldots) \stackrel{\text{def}}{=} \sigma_0, \ldots, \sigma_n, \sigma'_1, \ldots$ 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:

- $A \cdot B \stackrel{\text{def}}{=} \{\, a \cdot b \,|\, a \in A,\ b \in B \,\}$
- $A^\frown B \stackrel{\text{def}}{=} \{\, a^\frown b \,|\, a \in A,\ b \in B,\ a^\frown b \text{ defined} \,\}$
- $A^0 = \{\epsilon\}$ (neutral element for $\cdot$)
  $A^{n+1} \stackrel{\text{def}}{=} A \cdot A^n$,
  $A^\omega \stackrel{\text{def}}{=} A \cdot A \cdot \cdots$
  $A^* \stackrel{\text{def}}{=} \cup_{n<\omega} A^n$,
  $A^\infty \stackrel{\text{def}}{=} \cup_{n\leq\omega} A^n$
- $A^{\frown 0} = \Sigma$ (neutral element for $^\frown$)
  $A^{\frown n+1} \stackrel{\text{def}}{=} A^\frown A^{\frown n}$,
  $A^{\frown\omega} \stackrel{\text{def}}{=} A^\frown A^\frown \cdots$
  $A^{\frown *} \stackrel{\text{def}}{=} \cup_{n<\omega} A^{\frown n}$,
  $A^{\frown\infty} \stackrel{\text{def}}{=} \cup_{n\leq\omega} A^{\frown n}$

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

Distributivity of junction

- $\frown$ distributes over finite and infinite $\cup$:
  $A^\frown(\cup_{i\in I} B_i) = \cup_{i\in I}(A^\frown B_i)$ and
  $(\cup_{i\in I} A_i)^\frown B = \cup_{i\in I}(A_i{}^\frown B)$
  where I can be finite or infinite.

- $\frown$ distributes finite $\cap$ but not infinite $\cap$
  example:
  $\{a^\omega\}^\frown(\cap_{n\in\mathbb{N}}\{\, a^m \mid n \geq m \,\}) = \{a^\omega\}^\frown\emptyset = \emptyset$ but
  $\cap_{n\in\mathbb{N}}(\{a^\omega\}^\frown\{\, a^m \mid n \geq m \,\}) = \cap_{n\in\mathbb{N}}\{a^\omega\} = \{a^\omega\}$

- but, if $A \subseteq \Sigma^*$, then $A^\frown(\cap_{i\in I} B_i) = \cup_{i\in I}(A^\frown B_i)$
  even for infinite I

Note: concatenation $\cdot$ distributes infinite $\cap$ and $\cup$.

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
  $(\{\, \sigma, \sigma' \mid \sigma \to \sigma' \,\})$

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

Finite trace semantics

Prefix trace semantics

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

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

(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:

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

(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 $F_p(T) = \mathcal{I} \cup T \frown \tau$

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 $(\mathcal{P}(\Sigma^*), \subseteq)$:
$F_p(\cup_{i\in I} T_i) = \mathcal{I} \cup (\cup_{i\in I} T_i) \frown \tau = \mathcal{I} \cup (\cup_{i\in I} T_i \frown \tau) = \cup_{i\in I} (\mathcal{I} \cup T_i \frown \tau)$,
hence (Kleene), $\text{lfp}\, F_p = \cup_{n\geq 0} F_p^i(\emptyset)$

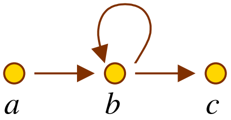
We prove by recurrence on n that $\forall n$: $F_p^n(\emptyset) = \cup_{i<n} \mathcal{I} \frown \tau^{\frown i}$:

- $F_p^0(\emptyset) = \emptyset$,
- $F_p^{n+1}(\emptyset) = \mathcal{I} \cup F_p^n(\emptyset) \frown \tau = \mathcal{I} \cup (\cup_{i<n} \mathcal{I} \frown \tau^{\frown i}) \frown \tau = \mathcal{I} \cup \cup_{i<n} (\mathcal{I} \frown \tau^{\frown i}) \frown \tau = \mathcal{I} \frown \tau^{\frown 0} \cup \cup_{i<n} (\mathcal{I} \frown \tau^{\frown i+1}) = \cup_{i<n+1} \mathcal{I} \frown \tau^{\frown i}$.

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

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

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

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

Prefix trace semantics: expressive power

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

$\Longrightarrow$ 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:

$$(\mathcal{P}(\Sigma^*), \subseteq) \underset{\alpha_p}{\overset{\gamma_p}{\leftrightarrows}} (\mathcal{P}(\Sigma), \subseteq)$$

- $\alpha_p(T) \stackrel{\text{def}}{=} \{\, \sigma \in \Sigma \mid \exists \sigma_0, \ldots, \sigma_n \in T : \sigma = \sigma_n \,\}$
  (last state in traces in T)
- $\gamma_p(S) \stackrel{\text{def}}{=} \{\, \sigma_0, \ldots, \sigma_n \in \Sigma^* \mid \sigma_n \in S \,\}$
  (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
- $$\begin{aligned}&(\gamma_p \circ \alpha_p)(T)\\ &= \{\, \sigma_0, \ldots, \sigma_n \mid \sigma_n \in \alpha_p(T) \,\}\\ &= \{\, \sigma_0, \ldots, \sigma_n \mid \exists \sigma'_0, \ldots, \sigma'_m \in T \colon \sigma_n = \sigma'_m \,\}\\ &\supseteq T\end{aligned}$$
- $$\begin{aligned}&(\alpha_p \circ \gamma_p)(S)\\ &= \{\, \sigma \mid \exists \sigma_0, \ldots, \sigma_n \in \gamma_p(S) \colon \sigma = \sigma_n \,\}\\ &= \{\, \sigma \mid \exists \sigma_0, \ldots, \sigma_n \colon \sigma_n \in S,\ \sigma = \sigma_n \,\}\\ &= S\end{aligned}$$

Abstracting prefix traces into reachability

Recall that:

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

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 $\alpha_p \circ F_p = F_{\mathcal{R}} \circ \alpha_p$

$$
\begin{aligned}
&(\alpha_p \circ F_p)(T) \\
&= \alpha_p(\mathcal{I} \cup T^\frown \tau) \\
&= \{\, \sigma \mid \exists \sigma_0, \ldots, \sigma_n \in \mathcal{I} \cup T^\frown \tau \colon \sigma = \sigma_n \,\} \\
&= \mathcal{I} \cup \{\, \sigma \mid \exists \sigma_0, \ldots, \sigma_n \in T^\frown \tau \colon \sigma = \sigma_n \,\} \\
&= \mathcal{I} \cup \{\, \sigma \mid \exists \sigma_0, \ldots, \sigma_n \in T \colon \sigma_n \to \sigma \,\} \\
&= \mathcal{I} \cup \mathrm{post}_\tau(\{\, \sigma \mid \exists \sigma_0, \ldots, \sigma_n \in T \colon \sigma = \sigma_n \,\}) \\
&= \mathcal{I} \cup \mathrm{post}_\tau(\alpha_p(T)) \\
&= (F_{\mathcal{R}} \circ \alpha_p)(T)
\end{aligned}
$$

Abstracting traces into states (example)

program

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

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

$\Longrightarrow$ the abstraction forgets the ordering of states.

Prefix closure

Prefix partial order: $\preceq$ on Σ^∞

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

$(\Sigma^\infty, \preceq)$ is a CPO, while $(\Sigma^*, \preceq)$ is not complete.

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

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

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)

Ordering abstraction

Another Galois embedding between finite traces and states:

$$(\mathcal{P}(\Sigma^*), \subseteq) \xleftrightarrows[\alpha_o]{\gamma_o} (\mathcal{P}(\Sigma), \subseteq)$$

- $\alpha_o(T) \stackrel{\text{def}}{=} \{\, \sigma \mid \exists \sigma_0, \ldots, \sigma_n \in T, i \leq n \colon \sigma = \sigma_i \,\}$
  (set of all states appearing in some trace in T)
- $\gamma_o(S) \stackrel{\text{def}}{=} \{\, \sigma_0, \ldots, \sigma_n \mid n \geq 0, \forall i \leq n \colon \sigma_i \in S \,\}$
  (traces composed of elements from S)

proof sketch:

α_o and γ_o are monotonic, and $\alpha_o \circ \gamma_o = id$.
$(\gamma_o \circ \alpha_o)(T) = \{\, \sigma_0, \ldots, \sigma_n \mid \forall i \leq n \colon \exists \sigma'_0, \ldots, \sigma'_m \in T, j \leq m \colon \sigma_i = \sigma'_j \,\}$
$\supseteq T$.

Ordering abstraction

We have: $\alpha_o(\mathcal{T}_p(\mathcal{I})) = \mathcal{R}(\mathcal{I})$.

proof:

We have $\alpha_o = \alpha_p \circ \rho_p$ (i.e.: a state is in a trace if it is the last state of one of its prefix).
Recall the prefix trace abstraction into states: $\mathcal{R}(\mathcal{I}) = \alpha_p(\mathcal{T}_p(\mathcal{I}))$ and the fact that the prefix trace semantics is closed by prefix: $\rho_p(\mathcal{T}_p(\mathcal{I})) = \mathcal{T}_p(\mathcal{I})$.
We get $\alpha_o(\mathcal{T}_p(\mathcal{I})) = \alpha_p(\rho_p(\mathcal{T}_p(\mathcal{I}))) = \alpha_p(\mathcal{T}_p(\mathcal{I})) = \mathcal{R}(\mathcal{I})$.

alternate proof: generalized fixpoint transfer

Recall that $\mathcal{T}_p(\mathcal{I}) = \text{lfp}\, F_p$ where $F_p(T) \stackrel{\text{def}}{=} \mathcal{I} \cup T \frown \tau$ and $\mathcal{R}(\mathcal{I}) = \text{lfp}\, F_\mathcal{R}$ where $F_\mathcal{R}(S) \stackrel{\text{def}}{=} \mathcal{I} \cup \text{post}_\tau(S)$, but $\alpha_o \circ F_p = F_\mathcal{R} \circ \alpha_o$ does not hold in general, so, fixpoint transfer theorems do not apply directly.

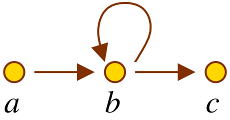
However, $\alpha_o \circ F_p = F_\mathcal{R} \circ \alpha_o$ holds for sets of traces closed by prefix. By induction, the Kleene iterates a_p^n and $a_\mathcal{R}^n$ involved in the computation of $\text{lfp}\, F_p$ and $\text{lfp}\, F_\mathcal{R}$ satisfy $\forall n: \alpha_o(a_p^n) = a_\mathcal{R}^n$, and so $\alpha_o(\text{lfp}\, F_p) = \text{lfp}\, F_\mathcal{R}$.

Suffix trace semantics

Similar results on the suffix trace semantics:

- $\mathcal{T}_s(\mathcal{F}) \stackrel{\text{def}}{=} \{\, \sigma_0, \ldots, \sigma_n \mid n \geq 0, \sigma_n \in \mathcal{F}, \forall i : \sigma_i \to \sigma_{i+1} \,\}$
  (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 $\alpha_s(T) \stackrel{\text{def}}{=} \{\, \sigma \mid \exists \sigma_0, \ldots, \sigma_n \in T : \sigma = \sigma_0 \,\}$
- $\rho_s(\mathcal{T}_s(\mathcal{F})) = \mathcal{T}_s(\mathcal{F})$
  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)
- $\alpha_o(\mathcal{T}_s(\mathcal{F})) = \mathcal{C}(\mathcal{F})$

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 $F_s(T) \stackrel{\text{def}}{=} \mathcal{F} \cup \tau \frown T$.

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

Finite partial trace semantics

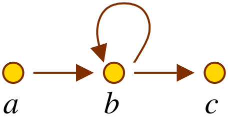
$\mathcal{T}$: all finite partial finite execution traces.

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

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

- $\mathcal{T} = \mathcal{T}_p(\Sigma)$, hence $\mathcal{T} = \mathrm{lfp}\, F_{p*}$ 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 $\mathcal{T} = \mathrm{lfp}\, F_{s*}$ 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 $F_{p*}(T) \stackrel{\text{def}}{=} \Sigma \cup T \frown \tau$.

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

(using $F_{s*}(T) \stackrel{\text{def}}{=} \Sigma \cup \tau \frown T$, 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:

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

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

(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.

$\mathcal{T} = \mathrm{lfp}\, F_{p*}$ where $F_{p*}(T) \overset{\text{def}}{=} \Sigma \cup T \frown \tau$.

$\mathcal{T}_p = \mathrm{lfp}\, F_p$ where $F_p(T) \overset{\text{def}}{=} \mathcal{I} \cup T \frown \tau$.
We have: $(\alpha_\mathcal{I} \circ F_{p*})(T) = (\Sigma \cup T \frown \tau) \cap (\mathcal{I} \cdot \Sigma^*) =$
$\mathcal{I} \cup ((T \frown \tau) \cap (\mathcal{I} \cdot \Sigma^*) = \mathcal{I} \cup ((T \cap (\mathcal{I} \cdot \Sigma^*)) \frown \tau) = (F_p \circ \alpha_\mathcal{I})(T)$.

Maximal 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)

$$\begin{array}{rl}\mathcal{M}_\infty \stackrel{\text{def}}{=} & \{\, \sigma_0, \ldots, \sigma_n \in \Sigma^* \mid \sigma_n \in \mathcal{B}, \forall i < n : \sigma_i \to \sigma_{i+1} \,\} \cup \\ & \{\, \sigma_0, \ldots, \sigma_n, \ldots \in \Sigma^\omega \mid \forall i < \omega : \sigma_i \to \sigma_{i+1} \,\}\end{array}$$

(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 $\mathcal{M}_\infty$.

We consider separately finite and infinite maximal traces.

- Finite traces:

  From the suffix partial trace semantics, recall:

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

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

- Infinite traces:

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

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

(proof on next slide)

Partitioned fixpoint formulation of maximal traces (proof)

proof: of $\mathcal{M}_\infty \cap \Sigma^\omega = \text{gfp}\, G_s$ where $G_s(T) \stackrel{\text{def}}{=} \tau ^\frown T$ in $(\mathcal{P}(\Sigma^\omega), \subseteq)$.

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

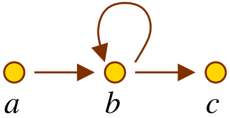
By Kleene's theorem in the dual: $\text{gfp}\, G_s = \cap_{n \in \mathbb{N}} G_s^n(\Sigma^\omega)$.

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

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

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

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 $G_s(T) \stackrel{\text{def}}{=} \tau \frown T$.

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

Least fixpoint formulation of maximal traces

Idea: To get a fixpoint formulation for whole $\mathcal{M}_\infty$,
merge finite and infinite maximal trace 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 $(\Sigma^\omega, \supseteq, \cap, \cup, \Sigma^\omega, \emptyset)$.

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

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

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

(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:

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

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)

proof: of $\mathcal{M}_\infty = \text{lfp } F_s$ where $F_s(T) \stackrel{\text{def}}{=} \mathcal{B} \cup \tau \frown T$.

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 $G_s(T) \stackrel{\text{def}}{=} \tau \frown T$,
- 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 $(\mathcal{P}(\Sigma^\infty), \sqsubseteq)$, we have:

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

Greatest fixpoint formulation of finite maximal traces

Actually, a fixpoint formulation in $(\Sigma^\infty, \subseteq)$ also exists.

Alternate fixpoint for finite maximal traces:

We saw that $\mathcal{M}_\infty \cap \Sigma^* = \text{lfp } F_s$
where $F_s(T) \stackrel{\text{def}}{=} \mathcal{B} \cup \tau \frown T$ in $(\mathcal{P}(\Sigma^*), \subseteq)$.

Additionally, we have $\mathcal{M}_\infty \cap \Sigma^* = \text{gfp } F_s$ in $(\mathcal{P}(\Sigma^*), \subseteq)$.

(F_s has a unique fixpoint in $(\mathcal{P}(\Sigma^*), \subseteq)$.)

(proof on next slide)

Greatest fixpoint formulation of finite maximal traces

proof: of $\mathcal{M}_\infty \cap \Sigma^* = \mathrm{gfp}\, F_s$ where $F_s(T) \stackrel{\text{def}}{=} \mathcal{B} \cup \tau \frown T$.

F_s is continuous in the dual $(\mathcal{P}(\Sigma^*), \supseteq)$: $F_s(\cap_{i\in I} A_i) = \cap_{i\in I} F_s(A_i)$.
By Kleene's theorem in the dual $(\mathcal{P}(\Sigma^*), \supseteq)$, we get: $\mathrm{gfp}\, F_s = \cap_{n\in\mathbb{N}} F_s^n(\Sigma^*)$.

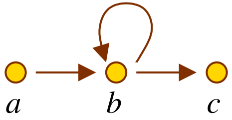
We prove by recurrence on n that $\forall n$: $F_s^n(\Sigma^*) = (\cup_{i<n} \tau^{\frown i} \frown \mathcal{B}) \cup (\tau^{\frown n} \frown \Sigma^*)$: i.e., $F_s^n(\Sigma^*)$ are the maximal finite traces of length at most $n-1$, and the partial traces of length exactly n followed by any sequence of states:

- $F_s^0(\Sigma^*) = \Sigma^* = \tau^{\frown 0} \frown \Sigma^*$
- $F_s(F_s^n(\Sigma^*)) = \mathcal{B} \cup (\tau \frown F_s^n(\Sigma^*))$
  $= \mathcal{B} \cup \tau \frown ((\cup_{i<n} \tau^{\frown i} \frown \mathcal{B}) \cup (\tau^{\frown n} \frown \Sigma^*))$
  $= \mathcal{B} \cup (\cup_{i<n} \tau \frown \tau^{\frown i} \frown \mathcal{B}) \cup (\tau \frown \tau^{\frown n} \frown \Sigma^*)$
  $= \mathcal{B} \cup (\cup_{1<i<n+1} \tau^{\frown i} \frown \mathcal{B}) \cup (\tau^{\frown n+1} \frown \Sigma^*)$
  $= (\cup_{i<n+1} \tau^{\frown i} \frown \mathcal{B}) \cup (\tau^{\frown n+1} \frown \Sigma^*)$

We get:

$\cap_{n\in\mathbb{N}} F_s^n(\Sigma^*) = \cap_{n\in\mathbb{N}} (\cup_{i<n} \tau^{\frown i} \frown \mathcal{B}) \cup (\tau^{\frown n} \frown \Sigma^*) = \cup_{n\in\mathbb{N}} \tau^{\frown n} \frown \mathcal{B} = \mathcal{M}_\infty \cap \Sigma^*$.

Greatest fixpoint of finite traces: graphical illustration



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

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

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

- $F_s^0(\Sigma^*) = \Sigma^*$
- $F_s^1(\Sigma^*) = \{c\} \cup ab\Sigma^* \cup bb\Sigma^* \cup bc\Sigma^*$
- $F_s^2(\Sigma^*) = \{bc, c\} \cup abb\Sigma^* \cup bbb\Sigma^* \cup abc\Sigma^* \cup bbc\Sigma^*$
- $F_s^3(\Sigma^*) = \{abc, bbc, bc, c\} \cup abbb\Sigma^* \cup bbbb\Sigma^* \cup abbc\Sigma^* \cup bbbc\Sigma^*$
- $F_s^n(\Sigma^*) = \{\, ab^ic, b^jc \mid i \in [1, n-2], j \in [0, n-1] \,\} \cup$
  $\{\, ab^nt, b^{n+1}t, ab^{n-1}ct, b^nct \mid t \in \Sigma^* \,\}$
- $\mathcal{M}_\infty \cap \Sigma^* = \cap_{n\geq 0} F_s^n(\Sigma^*) = \{\, ab^ic, b^jc \mid i \geq 1, j \geq 0 \,\}$

Greatest fixpoint formulation of maximal traces

From:

- $\mathcal{M}_\infty \cap \Sigma^* = \text{gfp } F_s$ in $(\mathcal{P}(\Sigma^*), \subseteq)$ where $F_s(T) \stackrel{\text{def}}{=} \mathcal{B} \cup \tau \frown T$
- $\mathcal{M}_\infty \cap \Sigma^\omega = \text{gfp } G_s$ in $(\mathcal{P}(\Sigma^\omega), \subseteq)$ where $G_s(T) \stackrel{\text{def}}{=} \tau \frown T$

we deduce: $\mathcal{M}_\infty = \text{gfp } F_s$ in $(\mathcal{P}(\Sigma^\infty), \subseteq)$.

proof: similar to $\mathcal{M}_\infty = \text{lfp } F_s$ in $(\mathcal{P}(\Sigma^\infty), \sqsubseteq)$, by fixpoint fusion.

Partial trace semantics

Finite and infinite partial trace semantics

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

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

$$\begin{aligned}\mathcal{T}_\infty \stackrel{\text{def}}{=} & \{\, \sigma_0, \ldots, \sigma_n \in \Sigma^* \mid \forall i < n \colon \sigma_i \to \sigma_{i+1} \,\} \cup \\ & \{\, \sigma_0, \ldots, \sigma_n, \ldots \in \Sigma^\omega \mid \forall i < \omega \colon \sigma_i \to \sigma_{i+1} \,\}\end{aligned}$$

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

Fixpoint form similar to $\mathcal{M}_\infty$:

- $\mathcal{T}_\infty = \operatorname{lfp} F_{s*}$ in $(\mathcal{P}(\Sigma^\infty), \sqsubseteq)$ where $F_{s*}(T) \stackrel{\text{def}}{=} \Sigma \cup \tau \frown T$,
- $\mathcal{T}_\infty = \operatorname{gfp} F_{s*}$ in $(\mathcal{P}(\Sigma^\infty), \subseteq)$.

proof: similar to the proofs of $\mathcal{M}_\infty = \operatorname{gfp} F_s$ and $\mathcal{M}_\infty = \operatorname{lfp} F_s$.

Finite trace abstraction

Finite partial traces $\mathcal{T}$ are an abstraction of all partial traces $\mathcal{T}_\infty$.

We have a Galois embedding:

$$(\mathcal{P}(\Sigma^\infty), \sqsubseteq) \underset{\alpha_*}{\overset{\gamma_*}{\leftrightarrows}} (\mathcal{P}(\Sigma^*), \subseteq)$$

- $\sqsubseteq$ is the fused ordering on $\Sigma^* \cup \Sigma^\omega$:
  $A \sqsubseteq B \overset{\text{def}}{\iff} (A \cap \Sigma^*) \subseteq (B \cap \Sigma^*) \land (A \cap \Sigma^\omega) \supseteq (B \cap \Sigma^\omega)$
- $\alpha_*(T) \overset{\text{def}}{=} T \cap \Sigma^*$
  (remove infinite traces)
- $\gamma_*(T) \overset{\text{def}}{=} T$
  (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 $(\alpha_* \circ \gamma_*)(T) = T \cap \Sigma^* = T$,
- $(\gamma_* \circ \alpha_*)(T) = T \cap \Sigma^* \sqsupseteq T$, as we only remove infinite traces.

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

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

Finite trace abstraction (proof)

alternate proof:

It is also possible to use the characterizations $\mathcal{T}_\infty = \mathrm{gfp}\, F_{s*}$ in $(\mathcal{P}(\Sigma^\infty), \subseteq)$ and $\mathcal{T} = \mathrm{gfp}\, F_{s*}$ in $(\mathcal{P}(\Sigma^*), \subseteq)$, and use a fixpoint transfer theorem for greatest fixpoints. Similarly to the fixpoint transfer for least fixpoints, this theorem uses the constructive version of Tarski's theorem, but in the dual: $\mathcal{T}_\infty$ is the limit of transfinite iterations $a_0 = \Sigma^\infty$, $a_{n+1} = F_{s*}(a_n)$, and $a_n = \cap\{\, a_m \mid m < n \,\}$ for transfinite ordinals, while $\mathcal{T}$ is the limit of a similar iteration from $a'_0 = \Sigma^*$. We conclude by noting that $a'_0 = \alpha_*(a_0)$, $\alpha_* \circ F_{s*} = F_{s*} \circ \alpha_*$, and α_* is co-continuous: $\alpha_*(\cap_{i \in I} T_i) = \cap_{i \in I} \alpha_*(T_i)$.

Note that, while the adjoint of α_* for $\sqsubseteq$ was $\gamma_*(T) \stackrel{\text{def}}{=} T$, the adjoint for $\subseteq$ is $\gamma'_*(T) \stackrel{\text{def}}{=} T \cup \Sigma^\omega$.

Prefix abstraction

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

We have a Galois connection:

$$(\mathcal{P}(\Sigma^\infty \setminus \{\epsilon\}), \subseteq) \xleftarrow[\alpha_\preceq]{\gamma_\preceq} (\mathcal{P}(\Sigma^\infty \setminus \{\epsilon\}), \subseteq)$$

- $\alpha_\preceq(T) \stackrel{\text{def}}{=} \{\, t \in \Sigma^\infty \setminus \{\epsilon\} \mid \exists u \in T \colon t \preceq u \,\}$
  (set of all non-empty prefixes of traces in T)
- $\gamma_\preceq(T) \stackrel{\text{def}}{=} \{\, t \in \Sigma^\infty \setminus \{\epsilon\} \mid \forall u \in \Sigma^\infty \setminus \{\epsilon\} \colon u \preceq t \implies u \in T \,\}$
  (traces with non-empty prefixes in T)

proof:

$\alpha_\preceq$ and $\gamma_\preceq$ are monotonic.

$(\alpha_\preceq \circ \gamma_\preceq)(T) = \{\, t \in T \mid \rho_p(t) \subseteq T \,\} \subseteq T$ (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 $\mathcal{T}_\infty$ are an abstraction of maximal traces $\mathcal{M}_\infty$: $\mathcal{T}_\infty = \alpha_{\preceq}(\mathcal{M}_\infty)$.

proof:

Firstly, $\mathcal{T}_\infty$ and $\alpha_{\preceq}(\mathcal{M}_\infty)$ coincide on infinite traces. Indeed, $\mathcal{T}_\infty \cap \Sigma^\omega = \mathcal{M}_\infty \cap \Sigma^\omega$ and $\alpha_{\preceq}$ 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 $\sigma_0, \ldots, \sigma_n \in \alpha_{\preceq}(\mathcal{M}_\infty)$, then $\forall i < n$: $\sigma_i \to \sigma_{i+1}$, so, $\sigma_0, \ldots, \sigma_n \in \mathcal{T}_\infty$.
Assume $\sigma_0, \ldots, \sigma_n \in \mathcal{T}_\infty$, then it can be completed into a maximal trace, either finite or infinite, and so, $\sigma_0, \ldots, \sigma_n \in \alpha_{\preceq}(\mathcal{M}_\infty)$.

Note: no fixpoint transfer applies here.

Finite prefix abstraction

We can abstract directly from maximal traces $\mathcal{M}_\infty$ to finite partial traces $\mathcal{T}$.

Consider the following Galois connection:

$$(\mathcal{P}(\Sigma^\infty \setminus \{\epsilon\}), \subseteq) \xleftrightarrows[\alpha_{*\preceq}]{\gamma_{*\preceq}} (\mathcal{P}(\Sigma^* \setminus \{\epsilon\}), \subseteq)$$

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

  (set of all non-empty prefixes of traces T)
- $\gamma_{*\preceq}(T) \stackrel{\text{def}}{=} \{\, t \in \Sigma^\infty \setminus \{\epsilon\} \,|\, \forall u \in \Sigma^* \setminus \{\epsilon\}: u \preceq t \implies u \in T \,\}$

  (traces with non-empty prefixes in T)

We have $\mathcal{T} = \alpha_{*\preceq}(\mathcal{M}_\infty)$.

(proof on next slide)

Finite prefix abstraction (proof)

proof:

$\alpha_{*\preceq}$ and $\gamma_{*\preceq}$ are monotonic.

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

$(\gamma_{*\preceq} \circ \alpha_{*\preceq})(T) = \rho_p(T) \cup \{\, t \in \Sigma^\omega \mid \forall u \in \Sigma^*: u \preceq t \implies u \in \rho_p(T) \,\} \supseteq T$.

As $\alpha_{*\preceq} = \alpha_* \circ \alpha_{\preceq}$,

we have: $\alpha_{*\preceq}(\mathcal{M}_\infty) = \alpha_*(\alpha_{\preceq}(\mathcal{M}_\infty)) = \alpha_*(\mathcal{T}_\infty) = \mathcal{T}$.

Remarks:

- $\gamma_{*\preceq} \circ \alpha_{*\preceq} \neq id$

  it closes trace sets by limits of finite traces.
- $\gamma_{*\preceq} \neq \gamma_{\preceq} \circ \gamma_*$

  this is because $\gamma_*(T) \stackrel{\text{def}}{=} T$ is the adjoint of α_* in $(\mathcal{P}(\Sigma^\infty), \sqsubseteq)$, while we need to compose $\alpha_{\preceq}$ with the adjoint of α_* in $(\mathcal{P}(\Sigma^\infty), \subseteq)$, which is

  $\gamma'_*(T) \stackrel{\text{def}}{=} T \cup \Sigma^\omega$.

(Partial) hierarchy of semantics

$$
\begin{tikzcd}
\mathcal{R}(\mathcal{I}) & & \mathcal{C}(\mathcal{F}) & \text{(states)} \\
\mathcal{T}_p(\mathcal{I}) \arrow[u, "\alpha_o"] & & \mathcal{T}_s(\mathcal{F}) \arrow[u, "\alpha_o"'] & \text{(anchored traces)} \\
& \mathcal{T} \arrow[ul, "\alpha_{\mathcal{I}}"] \arrow[ur, "\alpha_{\mathcal{F}}"'] & & \text{(partial finite traces)} \\
& \mathcal{T}_\infty \arrow[u, "\alpha_*"] & & \text{(partial traces)} \\
& \mathcal{M}_\infty \arrow[u, "\alpha_\preceq"] & & \text{(maximal traces)}
\end{tikzcd}
$$

Relational semantics

Big-step semantics

Finite big-step semantics

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

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

(symmetric and transitive closure of τ)

Fixpoint form:

$\mathcal{BS} = \mathrm{lfp}\ F_B$
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:

$$(\mathcal{P}(\Sigma^*), \subseteq) \xleftarrow[\alpha_{io}]{\gamma_{io}} \!\!\!\!\to\!\!\!\!\to (\mathcal{P}(\Sigma \times \Sigma), \subseteq)$$

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

proof sketch:

γ_{io} and α_{io} are monotonic.
$(\gamma_{io} \circ \alpha_{io})(T) = \{ \sigma_0, \ldots, \sigma_n \mid \exists \sigma'_0, \ldots, \sigma'_m \in T : \sigma_0 = \sigma'_0, \sigma_n = \sigma'_m \}$.
$(\alpha_{io} \circ \gamma_{io})(R) = R$.

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 $\mathcal{T} = \mathrm{lfp}\, F_{p*}$ where $F_{p*}(T) \stackrel{\text{def}}{=} \Sigma \cup T \frown \tau$.
Moreover, $F_B(R) \stackrel{\text{def}}{=} id \cup \{\, (\sigma, \sigma'') \,|\, \exists \sigma' : (\sigma, \sigma') \in R, \sigma' \to \sigma'' \,\}$.
Then, $\alpha_{io} \circ F_{p*} = F_B \circ \alpha_{io}$ because $\alpha_{io}(\Sigma) = id$ and
$\alpha_{io}(T \frown \tau) = \{\, (\sigma, \sigma'') \,|\, \exists \sigma' : (\sigma, \sigma') \in \alpha_{io}(T) \wedge \sigma' \to \sigma'' \,\}$.
By fixpoint transfer: $\alpha_{io}(\mathcal{T}) = \mathrm{lfp}\, F_B$.

We have a similar result using $F_{s*}(T) \stackrel{\text{def}}{=} \Sigma \cup \tau \frown T$ and
$F'_B(R) \stackrel{\text{def}}{=} id \cup \{\, (\sigma, \sigma'') \,|\, \exists \sigma' : (\sigma', \sigma'') \in R \wedge \sigma \to \sigma' \,\}$.

Finite big-step semantics (example)

program

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

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

Denotational semantics

Denotational semantics (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',
- $(\sigma, \spadesuit)$: non-terminating execution starting in σ.

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

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

- $\alpha_d(T) \stackrel{\text{def}}{=} \alpha_{io}(T \cap \Sigma^*) \cup \{\, (\sigma, \spadesuit) \mid \exists t \in \Sigma^\omega \colon \sigma \cdot t \in T \,\}$
- $\gamma_d(R) \stackrel{\text{def}}{=} \gamma_{io}(R \cap (\Sigma \times \Sigma)) \cup \{\, \sigma \cdot t \mid (\sigma, \spadesuit) \in R, t \in \Sigma^\omega \,\}$

  (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 $\mathcal{M}_\infty$, separate terminating and non-terminating behaviors, and use a fixpoint fusion theorem.

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

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

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

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

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

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

Denotational fixpoint semantics (proof cont.)

proof sketch: for infinite executions

We have $\alpha_d \circ G_s = G_d \circ \alpha_d$ in $\mathcal{P}(\Sigma^\omega) \to \mathcal{P}(\Sigma \times \{\spadesuit\})$, where
$G_d(R) \stackrel{\text{def}}{=} \{ (\sigma, \sigma'') \mid \exists \sigma' : \sigma \to \sigma' \wedge (\sigma', \sigma'') \in R \}$.

The fixpoint theorem for gfp we used in the alternate proof of $\mathcal{T} = \alpha_*(\mathcal{T}_\infty)$ does not apply here because α_d is not co-continuous: $\alpha_d(\cap_{i \in I} S_i) = \cap_{\in I} \alpha_d(S_i)$ does not hold; consider for example: $I = \mathbb{N}$ and $S_i = \{ a^n b^\omega \mid n > i \}$: $\cap_{i \in \mathbb{N}} S_i = \emptyset$, but $\forall i : \alpha_d(S_i) = \{(a, \spadesuit)\}$.

We use instead a fixpoint transfer based on Tarksi's theorem.
We have $\text{gfp } G_s = \cup \{ X \mid X \subseteq G_s(X) \}$.
Thus, $\alpha_d(\text{gfp } G_s) = \alpha_d(\cup \{ X \mid X \subseteq G_s(X) \}) = \cup \{ \alpha_d(X) \mid X \subseteq G_s(X) \}$ 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 $\mathcal{DS}$:

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

(quite different from the big-step semantics)

Denotational semantics (functional form)

Note: denotational semantics are often presented as functions, not relations

This is possible using the following Galois isomorphism:

$$(\mathcal{P}(\Sigma \times (\Sigma \cup \{\spadesuit\})), \sqsubseteq^*) \xleftarrow[\alpha_{df}]{\gamma_{df}} (\Sigma \to \mathcal{P}(\Sigma \cup \{\spadesuit\}), \dot{\sqsubseteq}^*)$$

- $\alpha_{df}(R) \stackrel{\text{def}}{=} \lambda\sigma.\{\,\sigma' \mid (\sigma, \sigma') \in R\,\}$
- $\gamma_{df}(f) \stackrel{\text{def}}{=} \{\,(\sigma, \sigma') \mid \sigma' \in f(\sigma)\,\}$
- $f \mathrel{\dot{\sqsubseteq}^*} f \stackrel{\text{def}}{\iff} \forall\sigma\text{:}\ \ (f(\sigma) \cap \Sigma \subseteq g(\sigma) \cap \Sigma) \wedge (\spadesuit \in g(\sigma) \implies \spadesuit \in f(\sigma))$

We get that: $\alpha_{df}(\mathcal{DS}) = \mathrm{lfp}\ F'_d$ where
$F'_d(f) \stackrel{\text{def}}{=} (\alpha_{df} \circ F_d \circ \gamma_{df})(f) = (\lambda\sigma.\{\,\sigma \mid \sigma \in \mathcal{B}\,\}) \mathbin{\dot{\cup}} (f \circ \mathrm{post}_\tau)$.
(proof by fixpoint transfer, as $F'_d \circ \alpha_{df} = F_d \circ \alpha_{df}$)

Another part of the hierarchy of semantics

$$
\begin{tikzcd}
 & & \mathcal{BS} & \text{(big-step semantics)} \\
\text{(partial finite traces)} & \mathcal{T} \arrow[ur, "\alpha_{io}"'] & & \\
\text{(partial traces)} & \mathcal{T}_\infty \arrow[u, "\alpha_*"] & \mathcal{DS} & \text{(denotational semantics)} \\
\text{(maximal traces)} & \mathcal{M}_\infty \arrow[u, "\alpha_\preceq"] \arrow[ur, "\alpha_d"'] & &
\end{tikzcd}
$$

See [Cou82] for more semantics in this diagram.

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 $I \subseteq \Sigma$

- $\mathcal{I} \subseteq I$

  (contains initial states)

- $\forall \sigma \in I: \sigma \to \sigma' \implies \sigma' \in I$

  (invariant by program transition)

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

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

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

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

Hoare logic proof method

Idea:

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

$$\frac{}{\{P[e/X]\}\, X \leftarrow e\, \{P\}} \qquad \frac{\{P\}\, stat_1\, \{R\} \quad \{R\}\, stat_2\, \{Q\}}{\{P\}\, stat_1; stat_2\, \{Q\}}$$

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

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

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

Equivalent to an invariant proof, partitioned by program location.
Any post-fixpoint of $\alpha_{\mathcal{L}} \circ F_{\mathcal{R}} \circ \gamma_{\mathcal{L}}$ gives valid Hoare triples.
$\alpha_{\mathcal{L}}(\mathcal{R}(\mathcal{I})) = \text{lfp}(\alpha_{\mathcal{L}} \circ F_{\mathcal{R}} \circ \gamma_{\mathcal{L}})$ gives the tightest Hoare triple.

Weakest liberal precondition proof methods

Idea: Start with a postcondition $\mathcal{F} \in \mathcal{P}(\Sigma)$
and compute preconditions backwards $P \Rightarrow wlp(stat, Q)$

- $wlp(X \leftarrow e, Q) \stackrel{\text{def}}{=} Q[e/X]$
- $wlp((stat_1; stat_2), Q) \stackrel{\text{def}}{=} wlp(stat_1, wlp(stat_2, Q))$
- $wlp(\textbf{if } b \textbf{ then } stat, Q) \stackrel{\text{def}}{=} (b \Rightarrow wlp(stat, Q)) \wedge (\neg b \Rightarrow Q)$
- $wlp(\textbf{while } b \textbf{ do } stat, Q) \stackrel{\text{def}}{=}$
  $I \wedge ((I \wedge b) \Rightarrow wlp(stat, I)) \wedge ((I \wedge \neg b) \Rightarrow Q)$
  (where the loop invariant I is generally provided by the user)

($P \Rightarrow wlp(stat, Q)$ is equivalent to $\{P\}\ stat\ \{Q\}$)

Link with the state semantics $\mathcal{S}(\mathcal{Y})$:

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

Equivalent to sufficient preconditions, partitioned by location:
any pre-fixpoint of $\alpha_{\mathcal{L}} \circ F_{\mathcal{S}} \circ \gamma_{\mathcal{L}}$ gives valid liberal preconditions;
$\alpha_{\mathcal{L}}(\mathcal{S}(\mathcal{F})) = \text{gfp}(\alpha_{\mathcal{L}} \circ F_{\mathcal{R}} \circ \gamma_{\mathcal{L}})$ gives the weakest liberal preconditions while inferring loop invariants!

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)

$$(\mathcal{P}(\Sigma^\infty), \subseteq) \xleftarrow[\alpha_{*\preceq}]{\gamma_{*\preceq}} \!\!\!\!\!\rightarrow (\mathcal{P}(\Sigma^*), \subseteq)$$

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

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

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

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

Definition of safety properties (examples)

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

Examples and counter-examples:

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

  $\rho_p(S^\infty) = \lim(S^\infty) = S^\infty \implies$ safety;

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

  $\rho_p(\Sigma^*) = \Sigma^*$, but $\lim(\Sigma^*) = \Sigma^\infty \neq \Sigma^* \implies$ not safety;

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

  $\rho_p((\Sigma^2)^\infty) = \Sigma^\infty \neq (\Sigma^2)^\infty \implies$ 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)
- $\forall \sigma_0, \ldots, \sigma_n \in I \colon \sigma_n \to \sigma_{n+1} \implies \sigma_0, \ldots, \sigma_n, \sigma_{n+1} \in I$

  (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 $F_p(T) \stackrel{\text{def}}{=} \mathcal{I} \cup T ^\frown \tau$.
$\mathcal{T}_p(\mathcal{I}) = \text{lfp}\, F_p$ 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 $\mathcal{M}_\infty$ and $\mathcal{T}$, we get:
$\mathcal{M}_\infty \cap (\mathcal{I} \cdot \Sigma^\infty) \subseteq \rho_{*\preceq}(\mathcal{M}_\infty \cap (\mathcal{I} \cdot \Sigma^\infty)) = \gamma_{*\preceq}(\alpha_{*\preceq}(\mathcal{M}_\infty \cap (\mathcal{I} \cdot \Sigma^\infty))) =$
$\gamma_{*\preceq}(\alpha_{*\preceq}(\mathcal{M}_\infty) \cap (\mathcal{I} \cdot \Sigma^*)) = \gamma_{*\preceq}(\mathcal{T} \cap (\mathcal{I} \cdot \Sigma^*)) = \gamma_{*\preceq}(\mathcal{T}_p(\mathcal{I}))$.
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, $P = \gamma_{*\preceq}(P)$, so, $\gamma_{*\preceq}(\mathcal{T}_p(\mathcal{I})) \subseteq \gamma_{*\preceq}(P) = P$, 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:

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

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 $P \stackrel{\text{def}}{=} S^\infty$:
  $\Rightarrow$ find a partial execution containing a state in $\Sigma \setminus S$;
- disproving an order property $P \stackrel{\text{def}}{=} \Sigma^\infty \setminus ((\Sigma \setminus \{a\})^* \cdot b \cdot \Sigma^\infty)$
  $\Rightarrow$ 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 $\rho_{*\preceq}(P) = \Sigma^\infty$.

Examples and counter-examples:

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

  $\rho_p(\Sigma^*) = \Sigma^*$ and $\lim(\Sigma^*) = \Sigma^\infty \implies$ liveness;

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

  $\rho_p(P) = P \cup \Sigma^*$ and $\lim(P \cup \Sigma^*) = \Sigma^\infty \implies$ liveness;

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

  $\rho_p(S^\infty) = \lim(S^\infty) = S^\infty \neq \Sigma^\infty$ if $S \neq \Sigma \implies$ not liveness;

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

  $\rho_p(\Sigma^{\leq k}) = \lim(\Sigma^{\leq k}) = \Sigma^{\leq k} \neq \Sigma^\infty \implies$ 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 $f : \Sigma \to \mathcal{S}$ where $(\mathcal{S}, \sqsubseteq)$ is well-ordered;

  (f is called a "ranking function")

- $\sigma \in \mathcal{B} \implies f = \min \mathcal{S}$;
- $\sigma \to \sigma' \implies f(\sigma') \sqsubset f(\sigma)$.

(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

Topology on X, defined by

- a family $\mathcal{C} \subseteq \mathcal{P}(X)$ of closed sets
  - $c, c' \in \mathcal{C} \implies c \cup c' \in \mathcal{C}$ (closed by finite unions)
  - $C \subseteq \mathcal{C} \implies \cap \{ c \mid c \in C \} \in \mathcal{C}$ (closed by intersections)
- open sets $\mathcal{O}$ are derived from closed sets:
  $\mathcal{O} \stackrel{\text{def}}{=} \{ X \setminus c \mid c \in \mathcal{C} \}$

  (closed by unions and finite intersections)

  (we can alternatively define a topology by $\mathcal{O}$, and derive $\mathcal{C}$ from $\mathcal{O}$)

Definition: we define a topology on traces by setting:

- $X \stackrel{\text{def}}{=} \Sigma^\infty$
- $\mathcal{C} \stackrel{\text{def}}{=} \{ P \in \mathcal{P}(\Sigma^\infty) \mid P \text{ is a safety property} \}$

Closure and density

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

- $\rho(x) \stackrel{\text{def}}{=} \cap \{\, c \in \mathcal{C} \mid x \subseteq c \,\}$;
  (ρ is an upper closure operator in $(\mathcal{P}(X), \subseteq)$)
  ($\rho(x) = x \iff x \in \mathcal{C}$)
- 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:
$\rho(x) \cap (x \cup (X \setminus \rho(x))) = (\rho(x) \cap x) \cup (\rho(x) \cap (X \setminus \rho(x))) = \rho(x) \cap x = x$ 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?
  $\forall \sigma_0, \ldots, \sigma_n \in \mathcal{M}_\infty : \forall \sigma'_0 : \sigma_0 \equiv \sigma'_0 \implies$
  $\exists \sigma'_0, \ldots, \sigma'_m \in \mathcal{M}_\infty : \sigma'_m \equiv \sigma_m$
  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.