Programs and executions
Simple structured, numeric language

- $X \in \mathbb{V}$, where $\mathbb{V}$ is a finite set of program variables
- $\ell \in \mathbb{L}$, where $\mathbb{L}$ is a finite set of control points
- numeric expressions: $\otimes \in \{=, \leq, \ldots\}$, $\diamond \in \{+, -, \times, /\}$
- random inputs: $X \leftarrow [c, c']$

model environment, parametric programs, unknown functions, ...
Example

```
\[aX \leftarrow [-\infty, \infty];
\]
\[b\text{while } \neg cX = 0 \text{ do } dX \leftarrow X - 1 \text{ done } e\]
```

Where:
- control points \( L = \{a, b, c, d, e\} \)
- variables \( V = \{X\} \)

We also define:
- the entry control point: \( a \)
- the exit control point: \( e \)
- the memory states: \( E \overset{\text{def}}{=} \forall \rightarrow \mathbb{Z} \)
- the program states: \( \Sigma \overset{\text{def}}{=} L \times E \) (control and memory state)
Program execution modeled as discrete \textit{transitions} between \textit{states}.

- $\Sigma$: set of \textit{states}
- $\tau \subseteq \Sigma \times \Sigma$: a \textit{transition relation}, written $\sigma \xrightarrow{\tau} \sigma'$, or $\sigma \rightarrow \sigma'$

\Longrightarrow a \text{ form of small-step semantics.}

and also sometimes:
- distinguished set of \textit{initial states} $\mathcal{I} \subseteq \Sigma$
- distinguished set of \textit{final states} $\mathcal{F} \subseteq \Sigma$
- \textit{labeled} transition systems: $\tau \subseteq \Sigma \times \mathcal{A} \times \Sigma$, $\sigma \xrightarrow{a} \sigma'$
  where $\mathcal{A}$ is a set of \textit{labels}, or \textit{actions}
Application: on our programming language

- $\Sigma \overset{\text{def}}{=} L \times E$: a control point and a memory state
  where $E \overset{\text{def}}{=} \text{V} \rightarrow \mathbb{Z}$

- initial states $I \overset{\text{def}}{=} \{ \ell \} \times E$ and
  final states $F \overset{\text{def}}{=} \{ \ell' \} \times E$ for program $\ell \text{ stat } \ell'$

- $\tau$ is defined by structural induction on $\ell \text{ stat } \ell'$ (next slides)

- $\tau$ is non-deterministic
  (several possible successors for $X \leftarrow [a, b]$)
Programs and executions

Transition semantics example

Example

\[a X \leftarrow [-\infty, \infty];\]

\[b \text{while } c X \neq 0 \text{ do } d X \leftarrow X - 1 \text{ done } e\]
From programs to transition relations

Transitions: \( \tau[\ell \text{ stat}^{\ell'}] \subseteq \Sigma \times \Sigma \)

\[
\tau[\ell_1 X \leftarrow e^{\ell_2}] \overset{\text{def}}{=} \{(\ell_1, \rho) \rightarrow (\ell_2, \rho[X \mapsto v]) | \rho \in \mathcal{E}, \ v \in E[e] \rho\}
\]

\[
\tau[\ell_1 \text{ if } e \bowtie 0 \text{ then } \ell_2 s^{\ell_3}] \overset{\text{def}}{=}
\{(\ell_1, \rho) \rightarrow (\ell_2, \rho) | \rho \in \mathcal{E}, \ \exists v \in E[e] \rho: v \bowtie 0 \} \cup
\{(\ell_1, \rho) \rightarrow (\ell_3, \rho) | \rho \in \mathcal{E}, \ \exists v \in E[e] \rho: v \not\bowtie 0 \} \cup \tau[\ell_2 s^{\ell_3}]
\]

\[
\tau[\ell_1 \text{ while } \ell_2 e \bowtie 0 \text{ do } \ell_3 s^{\ell_4} \text{ done}^{\ell_5}] \overset{\text{def}}{=}
\{(\ell_1, \rho) \rightarrow (\ell_2, \rho) | \rho \in \mathcal{E} \} \cup
\{(\ell_2, \rho) \rightarrow (\ell_3, \rho) | \rho \in \mathcal{E}, \ \exists v \in E[e] \rho: v \bowtie 0 \} \cup \tau[\ell_3 s^{\ell_4}] \cup
\{(\ell_4, \rho) \rightarrow (\ell_2, \rho) | \rho \in \mathcal{E} \} \cup
\{(\ell_2, \rho) \rightarrow (\ell_5, \rho) | \rho \in \mathcal{E}, \ \exists v \in E[e] \rho: v \not\bowtie 0 \}
\]

\[
\tau[\ell_1 s^{\ell_1}; \ell_2 s^{\ell_2} s^{\ell_3}] \overset{\text{def}}{=} \tau[\ell_1 s^{\ell_1} \ell_2] \cup \tau[\ell_2 s^{\ell_2} s^{\ell_3}]
\]

(Expression semantics \( E[e] \) on next slide)
Expression semantics

\( E[e] : (\forall \to \mathbb{Z}) \to \mathcal{P}(\mathbb{Z}) \)

- semantics of an expression in a memory state \( \rho \in E \quad \text{def} \quad \forall \to \mathbb{Z} \)
- outputs a set of values in \( \mathcal{P}(\mathbb{Z}) \)
  - divisions by zero return no result (omit error states for simplicity)
  - random inputs lead to several values (non-determinism)
- defined by structural induction

\[
\begin{align*}
E[ [c, c'] ] \rho & \quad \text{def} \quad \{ x \in \mathbb{Z} \mid c \leq x \leq c' \} \\
E[ X ] \rho & \quad \text{def} \quad \{ \rho(X) \} \\
E[ -e ] \rho & \quad \text{def} \quad \{ -v \mid v \in E[e] \rho \} \\
E[ e_1 + e_2 ] \rho & \quad \text{def} \quad \{ v_1 + v_2 \mid v_1 \in E[e_1] \rho, v_2 \in E[e_2] \rho \} \\
E[ e_1 - e_2 ] \rho & \quad \text{def} \quad \{ v_1 - v_2 \mid v_1 \in E[e_1] \rho, v_2 \in E[e_2] \rho \} \\
E[ e_1 \times e_2 ] \rho & \quad \text{def} \quad \{ v_1 \times v_2 \mid v_1 \in E[e_1] \rho, v_2 \in E[e_2] \rho \} \\
E[ e_1 / e_2 ] \rho & \quad \text{def} \quad \{ v_1/v_2 \mid v_1 \in E[e_1] \rho, v_2 \in E[e_2] \rho, v_2 \neq 0 \}
\end{align*}
\]
Another example: $\lambda$–calculus

**Syntax:**

\[
\begin{align*}
 t & ::= x \quad \text{(variable)} \\
 & \quad | \quad \lambda x. t \quad \text{(abstraction)} \\
 & \quad | \quad t \; u \quad \text{(application)}
\end{align*}
\]

**Small-step operational semantics:**

\[
\begin{align*}
(\lambda x. M) N & \rightsquigarrow M[x/N] \\
M & \rightsquigarrow M' \\
N & \rightsquigarrow N'
\end{align*}
\]

Models program execution as a sequence of term-rewriting $\rightsquigarrow$

exposing each transition (low level).

\[
\begin{align*}
\Sigma & \overset{\text{def}}{=} \{\lambda\text{–terms}\} \\
\mathcal{T} & \overset{\text{def}}{=} \rightsquigarrow
\]
Program executions

Intuitive model of executions:

- program traces
  sequences of states encountered during execution
  sequences are possibly unbounded
- a program can have several traces
  due to non-determinism

Trace semantics:

- the domain is $\mathcal{D} \defeq \mathcal{P}(\Sigma^*)$
- the semantics is:
  \[ \mathcal{T}_p(\mathcal{I}) \defeq \{ \sigma_0, \ldots, \sigma_n \mid n \geq 0, \sigma_0 \in \mathcal{I}, \forall i : \sigma_i \rightarrow \sigma_{i+1} \} \]
- actually, execution prefixes observable in finite time
Trace semantics example

Example

\[a \ X \leftarrow [-\infty, \infty];\]
\[b \text{ while } c \ X \neq 0 \text{ do } d \ X \leftarrow X - 1 \text{ done } e\]
Several other choices of semantic are possible:

- reachable states
- relations between input and output
- going backward as well as forward
- . . .

these are all uncomputable concrete semantics

(next course will consider computable approximations)

**Goal:** use abstract interpretation to

- express all these semantics uniformly as fixpoints
  (stay most of the time at the level of transition systems, not program syntax)
- relate these semantics by abstraction relations
- study which semantics to choose for which class of properties
Finite prefix trace semantics
Finite prefix trace semantics

Finite traces

**Finite trace:** finite sequence of elements from $\Sigma$

- $\epsilon$: empty trace (unique)
- $\sigma$: trace of length 1 (assimilated to a state)
- $\sigma_0, \ldots, \sigma_{n-1}$: trace of length $n$

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

Note: we assimilate

- a set if states $S \subseteq \Sigma$ with a set of traces of length 1
- a relation $R \subseteq \Sigma \times \Sigma$ with a set of traces of length 2

so, $I, F, \tau \in \mathcal{P}(\Sigma^*)$
Operations on traces:

- **length**: \( |t| \in \mathbb{N} \) of a trace \( t \in \Sigma^* \)

- **concatenation** \( \cdot \):
  \[
  (\sigma_0, \ldots, \sigma_n) \cdot (\sigma'_0, \ldots, \sigma'_m) \defeq \sigma_0, \ldots, \sigma_n, \sigma'_0, \ldots, \sigma'_m
  \]
  \( \epsilon \cdot t \defeq t \cdot \epsilon \defeq t \)

- **junction** \( \triangleright \):
  \[
  (\sigma_0, \ldots, \sigma_n) \triangleright (\sigma'_0, \sigma'_1 \ldots, \sigma'_m) \defeq \sigma_0, \ldots, \sigma_n, \sigma'_1, \ldots, \sigma'_m
  \]
  when \( \sigma_n = \sigma'_0 \)

  undefined if \( \sigma_n \neq \sigma'_0 \), and for \( \epsilon \)

  (join two consecutive traces, the common element \( \sigma_n = \sigma'_0 \) is not repeated)
Extension to sets of traces:

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

- $A \triangleright B \overset{\text{def}}{=} \{ a \triangleright b \mid a \in A, b \in B, a \triangleright b \text{ defined} \}$
  - $\Sigma$ is the neutral element for $\triangleright$

\[
egin{align*}
A^0 & \overset{\text{def}}{=} \{\epsilon\} & A^0 & \overset{\text{def}}{=} \Sigma \\
A^{n+1} & \overset{\text{def}}{=} A \cdot A^n & A^{n+1} & \overset{\text{def}}{=} A \triangleright A^n \\
A^* & \overset{\text{def}}{=} \bigcup_{n<\omega} A^n & A^* & \overset{\text{def}}{=} \bigcup_{n<\omega} A^{n+1}
\end{align*}
\]

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

Note: $\cdot$ and $\triangleright$ distribute $\bigcup$ and $\cap$

$(\bigcup_{i \in I} A_i) \triangleright (\bigcup_{j \in J} B_j) = \bigcup_{i \in I, j \in J} (A_i \triangleright B_j)$, etc.
Prefix trace semantics

$T_p(I)$: finite partial execution traces starting in $I$.

$$T_p(I) \overset{\text{def}}{=} \{ \sigma_0, \ldots, \sigma_n \mid n \geq 0, \sigma_0 \in I, \forall i: \sigma_i \rightarrow \sigma_{i+1} \}$$

$$= \bigcup_{n \geq 0} I \downarrow (\tau \downarrow n)$$

(traces of length $n$, for any $n$, starting in $I$ and following $\tau$)

$T_p(I)$ can be expressed in fixpoint form:

$$T_p(I) = \text{lfp } F_p \text{ where } F_p(T) \overset{\text{def}}{=} I \cup T \downarrow \tau$$

($F_p$ appends a transition to each trace, and adds back $I$)

Alternate characterization: $T_p(I) = \text{lfp}_I G_p$ where $G_p(T) = T \cup T \downarrow \tau$.

$G_p$ extends $T$ by $\tau$ and accumulates the result with $T$

(proofs on next slides)
Prefix trace semantics: graphical illustration

\[ I \overset{\text{def}}{=} \{a\} \]
\[ \tau \overset{\text{def}}{=} \{(a, b), (b, b), (b, c)\} \]

**Iterates:** \[ T_p(I) = \text{lfp } F_p \text{ where } F_p(T) \overset{\text{def}}{=} I \cup T \circ \tau. \]

- \( F_p^0(\emptyset) = \emptyset \)
- \( F_p^1(\emptyset) = I = \{a\} \)
- \( F_p^2(\emptyset) = \{a, ab\} \)
- \( F_p^3(\emptyset) = \{a, ab, abb, abc\} \)
- \( F_p^n(\emptyset) = \{a, ab^i, ab^j c \mid i \in [1, n - 1], j \in [1, n - 2]\} \)
- \( T_p(I) = \bigcup_{n \geq 0} F_p^n(\emptyset) = \{a, ab^i, ab^i c \mid i \geq 1\} \)
Finite prefix trace semantics

Prefix trace semantics: proof

proof of: \( T_p(\mathcal{I}) = \text{lfp} \ F_p \) where \( F_p(T) = \mathcal{I} \cup T \twoheadrightarrow \tau \)

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

\[
F_p(\bigcup_{i \in I} T_i) \\
= \mathcal{I} \cup \left( \bigcup_{i \in I} T_i \right) \twoheadrightarrow \tau \\
= \mathcal{I} \cup \left( \bigcup_{i \in I} T_i \twoheadrightarrow \tau \right) = \bigcup_{i \in I} (\mathcal{I} \cup T_i \twoheadrightarrow \tau)
\]

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

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

1. \( F_p^0(\emptyset) = \emptyset \),
2. \( F_p^{n+1}(\emptyset) = \mathcal{I} \cup F_p^n(\emptyset) \twoheadrightarrow \tau \\
= \mathcal{I} \cup \left( \bigcup_{i < n} \mathcal{I} \twoheadrightarrow \tau \twoheadrightarrow i \right) \twoheadrightarrow \tau \\
= \mathcal{I} \twoheadrightarrow \tau \twoheadrightarrow 0 \cup \bigcup_{i < n} \left( \mathcal{I} \twoheadrightarrow \tau \twoheadrightarrow i + 1 \right) \\
= \bigcup_{i < n+1} \mathcal{I} \twoheadrightarrow \tau \twoheadrightarrow i \\
Thus, \( \text{lfp} \ F_p = \bigcup_{n \in \mathbb{N}} F_p^n(\emptyset) = \bigcup_{n \in \mathbb{N}} \bigcup_{i < n} \mathcal{I} \twoheadrightarrow \tau \twoheadrightarrow i = \bigcup_{i \in \mathbb{N}} \mathcal{I} \twoheadrightarrow \tau \twoheadrightarrow i \).

The proof is similar for the alternate form \( T_p(\mathcal{I}) = \text{lfp}_\mathcal{I} \ G_p \) where \( G_p(T) = T \cup T \twoheadrightarrow \tau \) as \( G_p^n(\mathcal{I}) = F_p^{n+1}(\emptyset) = \bigcup_{i \leq n} \mathcal{I} \twoheadrightarrow \tau \twoheadrightarrow i \).
Finite prefix trace semantics

Note: prefix closure

Prefix partial order: \( \preceq \) on \( \Sigma^* \)

\[
x \preceq y \iff \exists u \in \Sigma^*: x \cdot u = y
\]

Note: \((\Sigma^*, \preceq)\) is not a CPO

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

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

\( \rho_p \) is an upper closure operator on \( \mathcal{P}(\Sigma^* \setminus \{\epsilon\}) \).
( monotonic, extensive \( T \subseteq \rho_p(T) \), idempotent \( \rho_p \circ \rho_p = \rho_p \) )

The prefix trace semantics is closed by prefix:

\[
\rho_p(T_p(I)) = T_p(I).
\]

(note that \( \epsilon \not\in T_p(I) \), which is why we disallowed \( \epsilon \) in \( \rho_p \) )
General and restricted trace properties
General properties

General setting:

- given a program \( \text{prog} \in \text{Prog} \)
- its semantics: \( [\cdot] : \text{Prog} \to \mathcal{P}(\Sigma^*) \) is a set of finite traces
- a property \( P \) is the set of correct program semantics
  
i.e., a set of sets of traces \( P \in \mathcal{P}(\mathcal{P}(\Sigma^*)) \)

\( \subseteq \) gives an information order on properties

\( P \subseteq P' \) means that \( P' \) is weaker than \( P \) (allows more semantics)
General and restricted trace properties

General collecting semantics

The collecting semantics \( Col : Prog \rightarrow P(\mathcal{P}(\Sigma^*)) \)
is the strongest property of a program

Hence: \( Col(prog) \overset{\text{def}}{=} \{ \llbracket prog \rrbracket \} \)

Benefit:

- given a program \( prog \) and a property \( P \in \mathcal{P}(\mathcal{P}(\Sigma^*)) \) the verification problem is an inclusion checking:

  \[ Col(prog) \subseteq P \]

- generally, the collecting semantics cannot be computed
  we settle for a weaker property \( S^\# \) that
    - is sound: \( Col(prog) \subseteq S^\# \)
    - implies the desired property: \( S^\# \subseteq P \)
Restricted properties

Reasoning on (and abstracting) $\mathcal{P}(\mathcal{P}(\Sigma^*))$ is hard!

In the following, we use a simpler setting:

- a property is a set of traces $P \in \mathcal{P}(\Sigma^*)$
- the collecting semantics is a set of traces: $Col(prog) \overset{\text{def}}{=} \llbracket prog \rrbracket$
- the verification problem remains an inclusion checking: $\llbracket prog \rrbracket \subseteq P$
- abstraction will over-approximate the set of traces $\llbracket prog \rrbracket$

Example properties:

- state property $P \overset{\text{def}}{=} S^*$ (remain in the set $S$ of safe states)
- maximal execution time: $P \overset{\text{def}}{=} S^{\leq k}$
- ordering: $P \overset{\text{def}}{=} (\Sigma \setminus \{b\})^* \cdot a \cdot \Sigma^* \cdot b \cdot \Sigma^*$ (a occurs before b)
Invariance proof method: find an inductive invariant \( I \)

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

and implies the desired property: \( I \subseteq P \)

Link with the finite prefix trace semantics \( T_p(I) \):

An inductive invariant is a post-fixpoint of \( F_p \): \( F_p(I) \subseteq I \)
where \( F_p(T) \overset{\text{def}}{=} I \cup T \triangledown_{\tau} \).

\( T_p(I) = \text{lfp} F_p \) is the tightest inductive invariant.
Limitations

Our semantics is closed by prefix. It cannot distinguish between:

- non-terminating executions (infinite loops)
- and unbounded executions

⇒ we cannot prove termination and, more generally, liveness (this will be solved using maximal trace semantics later in this course)

Some properties, such as non-interferences, cannot be expressed as sets of traces, we need sets of sets of traces

\[ P \overset{\text{def}}{=} \{ T \in \mathcal{P}(\Sigma^*) \mid \forall \sigma_0, \ldots, \sigma_n \in T : \forall \sigma'_0 : \sigma_0 \equiv \sigma'_0 \implies \exists \sigma'_0, \ldots, \sigma'_m \in T : \sigma'_m \equiv \sigma_m \} \]

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

changing the initial value of \(X\) does not affect the set of final environments up to the value of \(X\)
Forward state reachability semantics

Forward state reachability semantics
State semantics and properties

**Principle:** reason on sets of states instead of sets of traces

- simpler semantic $Col : Prog \rightarrow \mathcal{P}(\Sigma)$
- state properties are also sets of states $P \in \mathcal{P}(\Sigma)$
  $\implies$ sufficient for many purposes
- easier to abstract
- can be seen as an abstraction of traces
  (forgets the ordering of states)
Forward image: \( \text{post}_\tau : \mathcal{P}(\Sigma) \rightarrow \mathcal{P}(\Sigma) \)

\[ \text{post}_\tau(S) \overset{\text{def}}{=} \{ \sigma' \mid \exists \sigma \in S: \sigma \rightarrow \sigma' \} \]

\( \text{post}_\tau \) is a strict, complete \( \cup \)-morphism in \((\mathcal{P}(\Sigma), \subseteq, \cup, \cap, \emptyset, \Sigma)\).

\( \text{post}_\tau(\bigcup_{i \in I} S_i) = \bigcup_{i \in I} \text{post}_\tau(S_i) \), \( \text{post}_\tau(\emptyset) = \emptyset \)

Blocking states: \( \mathcal{B} \overset{\text{def}}{=} \{ \sigma \mid \forall \sigma' \in \Sigma: \sigma \not\rightarrow \sigma' \} \)

(states with no successor: valid final states but also errors)

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

\[ \mathcal{R}(\mathcal{I}) \overset{\text{def}}{=} \{ \sigma \mid \exists n \geq 0, \sigma_0, \ldots, \sigma_n: \sigma_0 \in \mathcal{I}, \sigma = \sigma_n, \forall i: \sigma_i \rightarrow \sigma_{i+1} \} \]

\[ = \bigcup_{n \geq 0} \text{post}^n_\tau(\mathcal{I}) \]

(reachable \( \iff \) reachable from \( \mathcal{I} \) in \( n \) steps of \( \tau \) for some \( n \geq 0 \))
\( \mathcal{R}(\mathcal{I}) \) can be expressed in fixpoint form:

\[
\mathcal{R}(\mathcal{I}) = \text{lfp } F_\mathcal{R} \text{ where } F_\mathcal{R}(S) \overset{\text{def}}{=} \mathcal{I} \cup \text{post}_\tau(S)
\]

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

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

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

(proofs on next slide)
Forward state reachability semantics

Fixpoint formulation proof

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

$(\mathcal{P}(\Sigma), \subseteq)$ is a CPO and $\text{post}_\tau$ is continuous, hence $F_{\mathcal{R}}$ is continuous:

$F_{\mathcal{R}}(\bigcup_{i \in I} A_i) = \bigcup_{i \in I} F_{\mathcal{R}}(A_i)$.

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

We prove by recurrence on $n$ that: $\forall n: F_{\mathcal{R}}^n(\emptyset) = \bigcup_{i \leq n} \text{post}^i_\tau(\mathcal{I})$.

(states reachable in less than $n$ steps)

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

\[
F_{\mathcal{R}}^{n+1}(\emptyset) = F_{\mathcal{R}}\left(\bigcup_{i \leq n} \text{post}^i_\tau(\mathcal{I})\right)
= \mathcal{I} \cup \text{post}_\tau\left(\bigcup_{i \leq n} \text{post}^i_\tau(\mathcal{I})\right)
= \mathcal{I} \cup \bigcup_{i \leq n} \text{post}_\tau(\text{post}^i_\tau(\mathcal{I}))
= \mathcal{I} \cup \bigcup_{1 \leq i \leq n+1} \text{post}^i_\tau(\mathcal{I})
= \bigcup_{i \leq n+1} \text{post}^i_\tau(\mathcal{I})
\]

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

The proof is similar for the alternate form, given that $\text{lfp } G_{\mathcal{R}} = \bigcup_{n \in \mathbb{N}} G_{\mathcal{R}}^n(\mathcal{I})$ and

$G_{\mathcal{R}}^n(\mathcal{I}) = F_{\mathcal{R}}^{n+1}(\emptyset) = \bigcup_{i \leq n} \text{post}^i_\tau(\mathcal{I})$. 
Graphical illustration

Transition system.
Forward state reachability semantics

Graphical illustration

Initial states $\mathcal{I}$. 
Iterate $F^1_R(I)$.
Iterate $F^2_R(I)$.
Iterate $F^3_R(I)$. 
Iterate $F^4_R(I)$. 
Iterate $F^5_R(\mathcal{I})$.

$F^6_R(\mathcal{I}) = F^5_R(\mathcal{I}) \implies$ we reached a fixpoint $\mathcal{R}(\mathcal{I}) = F^5_R(\mathcal{I})$. 
Multiple forward fixpoints

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

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

Example:

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

Exercise:
Compute all the fixpoints of $G_\mathcal{R}(S) \overset{\text{def}}{=} S \cup \text{post}_\tau(S)$ on this example.
Infer the set of possible states at program end: $R(I) \cap F$.

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

- initial states $I$: $j \in [0, 10]$ at control point ●,
- final states $F$: any memory state at control point ●,
- $\Rightarrow R(I) \cap F$: control at ●, $i = 100$, and $j \in [0, 110]$.

Prove the absence of run-time error: $R(I) \cap B \subseteq F$.
(never block except when reaching the end of the program)

To ensure soundness, over-approximations are sufficient.
(if $R^\#(I) \supseteq R(I)$, then $R^\#(I) \cap B \subseteq F \Rightarrow R(I) \cap B \subseteq F$)
Forward state reachability semantics

Link with state-based invariance proof methods

**Invariance proof method:** find an inductive invariant \( I \subseteq \Sigma \)

- \( I \subseteq I \) (contains initial states)
- \( \forall \sigma \in I: \sigma \rightarrow \sigma' \implies \sigma' \in I \) (invariant by program transition)
- that implies the desired property: \( I \subseteq P \)

**Link with the state semantics \( R(I) \):**

- if \( I \) is an inductive invariant, then \( F_R(I) \subseteq I \)
  \[
  F_R(I) = I \cup \text{post}_\tau(I) \subseteq I \cup I = I
  \]
  \( \implies \) an inductive invariant is a post-fixpoint of \( F_R \)
- \( R(I) = \text{lfp} F_R \)
  \( \implies \) \( R(I) \) is the tightest inductive invariant
Link with the equational semantics

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

**Grouping by control location:** \( \mathcal{P}(\Sigma) = \mathcal{P}(\mathcal{L} \times \mathcal{E}) \simeq \mathcal{L} \rightarrow \mathcal{P}(\mathcal{E}) \)

We have a Galois isomorphism:

\[
(\mathcal{P}(\Sigma), \subseteq) \leftrightarrow \left( \mathcal{L} \rightarrow \mathcal{P}(\mathcal{E}), \dot{\subseteq} \right)
\]

- \( X \dot{\subseteq} Y \) \( \overset{\text{def}}{\iff} \forall \ell \in \mathcal{L}: X(\ell) \subseteq Y(\ell) \)
- \( \alpha_{\mathcal{L}}(S) \overset{\text{def}}{=} \lambda \ell. \{ \rho \mid (\ell, \rho) \in S \} \)
- \( \gamma_{\mathcal{L}}(X) \overset{\text{def}}{=} \{ (\ell, \rho) \mid \ell \in \mathcal{L}, \rho \in X(\ell) \} \)
- given \( F_{eq} \overset{\text{def}}{=} \alpha_{\mathcal{L}} \circ F_{R} \circ \gamma_{\mathcal{L}} \)
  - we get back an equation system \( \bigwedge_{\ell \in \mathcal{L}} X_{\ell} = F_{eq,\ell}(X_1, \ldots, X_n) \)
- \( \alpha_{\mathcal{L}} \circ \gamma_{\mathcal{L}} = \gamma_{\mathcal{L}} \circ \alpha_{\mathcal{L}} = id \) (no abstraction)

simply reorganize the states by control point
after actual abstraction, partitioning makes a difference (flow-sensitivity)
Example equation system

\[
\ell_1 X \leftarrow [0, 10]; \quad \ell_2 Y \leftarrow 100;
\]

while \( \ell_3 X \geq 0 \) do
\[
\ell_4 X \leftarrow X - 1; \quad \ell_5 Y \leftarrow Y + 10
\]

done \( \ell_6 \)

\[
\{ \begin{align*}
\mathcal{X}_1 &= \mathcal{E} \\
\mathcal{X}_2 &= C[ X \leftarrow [0, 10] ] \mathcal{X}_1 \\
\mathcal{X}_3 &= C[ Y \leftarrow 100 ] \mathcal{X}_2 \cup C[ Y \leftarrow Y + 10 ] \mathcal{X}_5 \\
\mathcal{X}_4 &= C[ X \geq 0 ] \mathcal{X}_3 \\
\mathcal{X}_5 &= C[ X \leftarrow X - 1 ] \mathcal{X}_4 \\
\mathcal{X}_6 &= C[ X < 0 ] \mathcal{X}_3
\end{align*} \}
\]

(atomic command semantics \( C[ \text{com} ] \) on next slide)

- \( \mathcal{X}_i \in \mathcal{P}(\mathcal{E}) \): set of memory states at program point \( i \in \mathcal{L} \)
  
  e.g.: \( \mathcal{X}_3 = \{ \rho \in \mathcal{E} | \rho(X) \in [0, 10], 10\rho(X) + \rho(Y) \in [100, 200] \cap 10\mathbb{Z} \} \)

- \( \mathcal{R} \) corresponds to the smallest solution \( (\mathcal{X}_i)_{i \in \mathcal{L}} \) of the system

- \( I \subseteq \mathcal{E} \) is invariant at \( i \) if \( \mathcal{X}_i \subseteq I \)
Systematic derivation of equations

**Atomic commands:** \( \text{com} \) : \( \mathcal{P}(\mathcal{E}) \rightarrow \mathcal{P}(\mathcal{E}) \)

\( \text{com} \) \( \overset{\text{def}}{=} \{ X \leftarrow \text{exp}, \text{exp} \bowtie 0 \} \): assignments and tests.

- \( \mathcal{C}[ X \leftarrow e ] \mathcal{X} \overset{\text{def}}{=} \{ \rho[X \mapsto v] \mid \rho \in \mathcal{X}, v \in \mathcal{E}[e] \rho \} \)
- \( \mathcal{C}[ e \bowtie 0 ] \mathcal{X} \overset{\text{def}}{=} \{ \rho \in \mathcal{X} \mid \exists v \in \mathcal{E}[\rho] \rho : v \bowtie 0 \} \)

\( \mathcal{C}[ \cdot ] \) are \( \cup \)-morphisms: \( \mathcal{C}[ s ] \mathcal{X} = \bigcup_{\rho \in \mathcal{X}} \mathcal{C}[ s ] \{ \rho \} \), monotonic, continuous

**Systematic derivation of the equation system:** \( \text{eq}(\ell \text{stat}^{\ell'}) \)

by structural induction:

\( \text{eq}(\ell^1 X \leftarrow e^{\ell_2}) \overset{\text{def}}{=} \{ \mathcal{X}_{\ell_2} = \mathcal{C}[ X \leftarrow e ] \mathcal{X}_{\ell_1} \} \)

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

\( \text{eq}(\ell^1 \text{if } e \bowtie 0 \text{ then } \ell^2 s^3) \overset{\text{def}}{=} \)

\( \{ \mathcal{X}_{\ell_2} = \mathcal{C}[ e \bowtie 0 ] \mathcal{X}_{\ell_1} \} \cup \text{eq}(\ell^2 s^3') \cup \{ \mathcal{X}_{\ell_3} = \mathcal{X}_{\ell_3'} \cup \mathcal{C}[ e \bowtie 0 ] \mathcal{X}_{\ell_1} \} \)

\( \text{eq}(\ell^1 \text{while } e \bowtie 0 \text{ do } \ell^3 s^4 \text{ done}^{\ell_5}) \overset{\text{def}}{=} \)

\( \{ \mathcal{X}_{\ell_2} = \mathcal{X}_{\ell_1} \cup \mathcal{X}_{\ell_4}, \mathcal{X}_{\ell_3} = \mathcal{C}[ e \bowtie 0 ] \mathcal{X}_{\ell_2} \} \cup \text{eq}(\ell^3 s^4) \cup \{ \mathcal{X}_{\ell_5} = \mathcal{C}[ e \bowtie 0 ] \mathcal{X}_{\ell_2} \} \)

where: \( \mathcal{X}_{\ell_3}' \) is a fresh variable storing intermediate results
Solving the equational semantics

\[ \forall i \in [1,n] \quad \mathcal{X}_i = F_i(\mathcal{X}_1, \ldots, \mathcal{X}_n) \]

Each \( F_i \) is continuous in \( \mathcal{P}(\mathcal{E})^n \rightarrow \mathcal{P}(\mathcal{E}) \) (complete \( \cup - \)morphism)

aka \( \vec{F} \overset{\text{def}}{=} (F_1, \ldots, F_n) \) is continuous in \( \mathcal{P}(\mathcal{E})^n \rightarrow \mathcal{P}(\mathcal{E})^n \)

By Kleene’s fixpoint theorem, \( \text{lfp} \, \vec{F} \) exists.

By Kleene’s fixpoint theorem

called Jacobi iterations by analogy with linear algebra

The limit of \( (\mathcal{X}_1^k, \ldots, \mathcal{X}_n^k) \) is \( \text{lfp} \, \vec{F} \).

Naïve application of Kleene’s theorem

called Jacobi iterations by analogy with linear algebra
Other iteration techniques exist [Cous92].

**Gauss-Seidl iterations**

\[
\begin{align*}
\chi_1^{k+1} & \overset{\text{def}}{=} F_1(\chi_1^k, \ldots, \chi_n^k) \\
& \quad \ldots \\
\chi_i^{k+1} & \overset{\text{def}}{=} F_i(\chi_1^{k+1}, \ldots, \chi_{i-1}^{k+1}, \chi_i^k, \ldots, \chi_n^k) \\
& \quad \ldots \\
\chi_n^{k+1} & \overset{\text{def}}{=} F_n(\chi_1^{k+1}, \ldots, \chi_{n-1}^{k+1}, \chi_n^k)
\end{align*}
\]

Use new results as soon available.

**Chaotic iterations**

\[
\chi_i^{k+1} \overset{\text{def}}{=} \begin{cases} 
F_i(\chi_1^k, \ldots, \chi_n^k) & \text{if } i = \phi(k + 1) \\
\chi_i^k & \text{otherwise}
\end{cases}
\]

Wrt. a fair schedule \( \phi : \mathbb{N} \rightarrow [1, n] \)

\[\forall i \in [1, n]: \forall N > 0: \exists k > N: \phi(k) = i\]

- Worklist algorithms
- Asynchronous iterations (parallel versions of chaotic iterations)

All give the same limit! (this will not be the case for abstract static analyses...)
**Forward state reachability semantics**

**Alternate view: inductive abstract interpreter**

**Principle:**
- follow the **control-flow** of the program
- replace the global fixpoint with **local fixpoints** (loops)

```
C[X ← e] X \[\text{def} = \{ \rho[X \mapsto v] \mid \rho \in X, \ v \in E[e] \rho \} \]
C[e \times 0] X \[\text{def} = \{ \rho \in X \mid \exists v \in E[\rho] : v \times 0 \} \]
C[s_1; s_2] X \[\text{def} = C[s_2](C[s_1] X) \]
C[if e \times 0 then s] X \[\text{def} = (C[s](C[e \times 0] X)) \cup (C[e \times 0] X) \]
C[while e \times 0 do s done] X \[\text{def} = C[e \times 0](\text{lfp } F) \]
where \( F(Y) \[\text{def} = X \cup C[s](C[e \times 0] Y) \]
```

**informal justification for the loop semantics:**

All the \( C[s] \) functions are continuous, hence the fixpoints exist.
By induction on \( k \), \( F^k(\emptyset) = \bigcup_{i \leq k} (C[s] \circ C[e \times 0]^i)X \)
hence, \( \text{lfp } F = \bigcup_i (C[s] \circ C[e \times 0]^i)X \)
We fall back to a special case of (transfinite) chaotic iteration
that stabilizes loops depth-first.
From finite traces to reachability
Abstracting traces into states

Idea: view state semantics as abstractions of traces semantics.

A state in the state semantics corresponds to any partial execution trace terminating in this state.

We have a Galois embedding between finite traces and states:

\[(\mathcal{P}(\Sigma^*), \subseteq) \xrightarrow{\alpha_p} (\mathcal{P}(\Sigma), \subseteq) \xleftarrow{\gamma_p}\]

- \(\alpha_p(T) \overset{\text{def}}{=} \{ \sigma \in \Sigma | \exists \sigma_0, \ldots, \sigma_n \in T: \sigma = \sigma_n \}\)
  (last state in traces in \(T\))

- \(\gamma_p(S) \overset{\text{def}}{=} \{ \sigma_0, \ldots, \sigma_n \in \Sigma^* | \sigma_n \in S \}\)
  (traces ending in a state in \(S\))

(proof on next slide)
proof of: \((\alpha_p, \gamma_p)\) 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: \(\alpha\) and \(\gamma\) are monotonic, \(\gamma \circ \alpha\) is extensive, and \(\alpha \circ \gamma\) is reductive. Embedding means that, additionally, \(\alpha \circ \gamma = id\).

- \(\alpha_p, \gamma_p\) are \(\cup\)–morphisms, hence monotonic
- \((\gamma_p \circ \alpha_p)(T)\)
  \[= \{ \sigma_0, \ldots, \sigma_n \mid \sigma_n \in \alpha_p(T) \}\]
  \[= \{ \sigma_0, \ldots, \sigma_n \mid \exists \sigma'_0, \ldots, \sigma'_m \in T : \sigma_n = \sigma'_m \}\]
  \[\supseteq T\]
- \((\alpha_p \circ \gamma_p)(S)\)
  \[= \{ \sigma \mid \exists \sigma_0, \ldots, \sigma_n \in \gamma_p(S) : \sigma = \sigma_n \}\]
  \[= \{ \sigma \mid \exists \sigma_0, \ldots, \sigma_n : \sigma_n \in S, \sigma = \sigma_n \}\]
  \[= S\]
We can abstract semantic operators and their least fixpoint.

Recall that:

\[ T_p(I) = \text{lfp} \ F_p \] where \[ F_p(T) \overset{\text{def}}{=} I \cup T \tau \],

\[ R(I) = \text{lfp} \ F_R \] where \[ F_R(S) \overset{\text{def}}{=} I \cup \text{post}_\tau(S) \],

\[ (\mathcal{P}(\Sigma^*), \subseteq) \xrightarrow{\gamma_p} \xleftarrow{\alpha_p} (\mathcal{P}(\Sigma), \subseteq). \]

We have: \[ \alpha_p \circ F_p = F_R \circ \alpha_p \];

by fixpoint transfer, we get: \[ \alpha_p(T_p(I)) = R(I) \].

(proof on next slide)
Abstracting prefix traces into reachability (proof)

proof: of \( \alpha_p \circ F_p = F_R \circ \alpha_p \)

\[
(\alpha_p \circ F_p)(T) \\
= \alpha_p(I \cup T \setminus \tau) \\
= \{ \sigma \mid \exists \sigma_0, \ldots, \sigma_n \in I \cup T \setminus \tau : \sigma = \sigma_n \} \\
= I \cup \{ \sigma \mid \exists \sigma_0, \ldots, \sigma_n \in T \setminus \tau : \sigma = \sigma_n \} \\
= I \cup \{ \sigma \mid \exists \sigma_0, \ldots, \sigma_n \in T : \sigma_n \rightarrow \sigma \} \\
= I \cup \text{post}_\tau(\{ \sigma \mid \exists \sigma_0, \ldots, \sigma_n \in T : \sigma = \sigma_n \}) \\
= I \cup \text{post}_\tau(\alpha_p(T)) \\
= (F_R \circ \alpha_p)(T)
\]
Abstracting traces into states (example)

program

\[
\begin{align*}
&j \leftarrow 0; \\
i &\leftarrow 0; \\
&\textbf{while } i < 100 \textbf{ do} \\
&\quad i \leftarrow i + 1; \\
&\quad j \leftarrow j + [0, 1] \\
&\textbf{done}
\end{align*}
\]

- **prefix trace semantics:**
  \(i\) and \(j\) are increasing and \(0 \leq j \leq i \leq 100\)

- **forward reachable state semantics:**
  \(0 \leq j \leq i \leq 100\)

\[\Rightarrow\] the abstraction **forgets the ordering of states.**
Another state/trace abstraction: ordering abstraction

Another Galois embedding between finite traces and states:

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

- \( \alpha_o(T) \overset{\text{def}}{=} \{ \sigma \mid \exists \sigma_0, \ldots, \sigma_n \in T, i \leq n: \sigma = \sigma_i \} \)
  (set of all states appearing in some trace in \( T \))

- \( \gamma_o(S) \overset{\text{def}}{=} \{ \sigma_0, \ldots, \sigma_n \mid n \geq 0, \forall i \leq n: \sigma_i \in S \} \)
  (traces composed of elements from \( S \))

proof sketch:
\( \alpha_o \) and \( \gamma_o \) are monotonic, and \( \alpha_o \circ \gamma_o = id \).
\( (\gamma_o \circ \alpha_o)(T) = \{ \sigma_0, \ldots, \sigma_n \mid \forall i \leq n: \exists \sigma'_0, \ldots, \sigma'_m \in T, j \leq m: \sigma_i = \sigma'_j \} \supseteq T \).
We have: \( \alpha_o(T_p(I)) = R(I) \).

**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: \( R(I) = \alpha_p(T_p(I)) \) and the fact that the prefix trace semantics is closed by prefix: \( \rho_p(T_p(I)) = T_p(I) \).

We get \( \alpha_o(T_p(I)) = \alpha_p(\rho_p(T_p(I))) = \alpha_p(T_p(I)) = R(I) \).

This is a **direct proof**, not a fixpoint transfer proof (our theorems do not apply . . .)

**alternate proof:** generalized fixpoint transfer

Recall that \( T_p(I) = \text{lfp } F_p \) where \( F_p(T) \overset{\text{def}}{=} I \cup T \triangleleft \tau \) and \( R(I) = \text{lfp } F_R \) where \( F_R(S) \overset{\text{def}}{=} I \cup \text{post}_{\tau}(S) \), but \( \alpha_o \circ F_p = F_R \circ \alpha_o \) does not hold in general, so, fixpoint transfer theorems do not apply directly.

However, \( \alpha_o \circ F_p = F_R \circ \alpha_o \) holds for sets of traces closed by prefix. By induction, the Kleene iterates \( a_n^p \) and \( a_n^R \) involved in the computation of \( \text{lfp } F_p \) and \( \text{lfp } F_R \) satisfy \( \forall n: \alpha_o(a_n^p) = a_n^R \), and so \( \alpha_o(\text{lfp } F_p) = \text{lfp } F_R \).
Backward state and trace semantics
Backward state and trace semantics

Backward state co-reachability

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

\[ C(\mathcal{F}) \overset{\text{def}}{=} \{ \sigma \mid \exists n \geq 0, \sigma_0, \ldots, \sigma_n: \sigma = \sigma_0, \sigma_n \in \mathcal{F}, \forall i: \sigma_i \rightarrow \sigma_{i+1} \} \]

\[ = \bigcup_{n \geq 0} \text{pre}_\tau^n(\mathcal{F}) \]

where \( \text{pre}_\tau(S) \overset{\text{def}}{=} \{ \sigma \mid \exists \sigma' \in S: \sigma \rightarrow \sigma' \} \) \( (\text{pre}_\tau = \text{post}_{\tau^{-1}}) \)

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

\[ C(\mathcal{F}) = \text{lfp} \, F_C \text{ where } F_C(S) \overset{\text{def}}{=} \mathcal{F} \cup \text{pre}_\tau(S) \]

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

Alternate characterization: \( C(\mathcal{F}) = \text{lfp}_\mathcal{F} \, G_C \text{ where } G_C(S) = S \cup \text{pre}_\tau(S) \)
Graphical illustration

Transition system.
Final states $\mathcal{F}$. 
Backward state and trace semantics

Graphical illustration
Graphical illustration
Graphical illustration
States co-reachable from $\mathcal{F}$. 
Application of backward co-reachability

\[ I \cap C(B \setminus F) \]
Initial states that have at least one erroneous execution.

- initial states \( I \): \( i \in [0, 100] \) at \( \bullet \)
- final states \( F \): any memory state at \( \bullet \)
- blocking states \( B \): final, or \( j > 200 \) (assertion failure)
- Over-approximating \( C \) is useful to isolate possibly incorrect executions from those guaranteed to be correct.
- Iterate forward and backward analyses interactively \( \Rightarrow \) abstract debugging [Bour93].
Backward state and trace semantics

Backward co-reachability in equational form

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

**Example:**

```
ℓ₁ j ← 0;
ℓ₂ while ℓ₃ i > 0 do
  ℓ₄ i ← i − 1;
  ℓ₅ j ← j + [0, 10]
ℓ₆
```

```
\begin{align*}
\mathcal{X}_1 &= \leftarrow C[i \rightarrow 0] \mathcal{X}_2 \\
\mathcal{X}_2 &= \mathcal{X}_3 \\
\mathcal{X}_3 &= \leftarrow C[i > 0] \mathcal{X}_4 \cup \leftarrow C[i \leq 0] \mathcal{X}_6 \\
\mathcal{X}_4 &= \leftarrow C[i \leftarrow i - 1] \mathcal{X}_5 \\
\mathcal{X}_5 &= \leftarrow C[j \leftarrow j + [0, 10]] \mathcal{X}_3 \\
\mathcal{X}_6 &= \mathcal{F}
\end{align*}
```

- final states $\{\ell₆\} \times \mathcal{F}$.
- $\leftarrow C[X \leftarrow e] \mathcal{X} \overset{\text{def}}{=} \{ \rho \mid \exists v \in E[e] \rho : \rho[X \leftarrow v] \in \mathcal{X} \}$.
- $\leftarrow C[e \triangleright 0] \mathcal{X} \overset{\text{def}}{=} \{ \rho \in \mathcal{X} \mid \exists v \in E[\rho] \rho : v \triangleright 0 \} = C[e \triangleright 0] \mathcal{X}$

(also possible on control-flow graphs... )
Backward state and trace semantics

Suffix trace semantics

Similarly to the finite prefix trace semantics from $\mathcal{I}$, we can build a suffix trace semantics going backwards from $\mathcal{F}$:

- $\mathcal{T}_s(\mathcal{F}) \overset{\text{def}}{=} \{ \sigma_0, \ldots, \sigma_n \mid n \geq 0, \sigma_n \in \mathcal{F}, \forall i: \sigma_i \rightarrow \sigma_{i+1} \}$
  (traces following $\tau$ and ending in a state in $\mathcal{F}$)

- $\mathcal{T}_s(\mathcal{F}) = \bigcup_{n \geq 0} (\tau \downarrow^n) \mathcal{F}$

- $\mathcal{T}_s(\mathcal{F}) = \text{lfp } F_s$ where $F_s(\mathcal{T}) \overset{\text{def}}{=} \mathcal{F} \cup \tau \downarrow \mathcal{T}$
  ($F_s$ prepends a transition to each trace, and adds back $\mathcal{F}$)

Backward state co-reachability abstracts the suffix trace semantics:

- $\alpha_s(\mathcal{T}_s(\mathcal{F})) = \mathcal{C}(\mathcal{F})$ where $\alpha_s(\mathcal{T}) \overset{\text{def}}{=} \{ \sigma \mid \exists \sigma_0, \ldots, \sigma_n \in \mathcal{T}: \sigma = \sigma_0 \}$

- $\rho_s(\mathcal{T}_s(\mathcal{F})) = \mathcal{T}_s(\mathcal{F})$ where $\rho_s(\mathcal{T}) \overset{\text{def}}{=} \{ u \mid \exists t \in \Sigma^*: t \cdot u \in \mathcal{T}, u \neq \epsilon \}$
  (closed by suffix)
Graphical illustration

\[ F \overset{\text{def}}{=} \{ c \} \]
\[ \tau \overset{\text{def}}{=} \{(a, b), (b, b), (b, c)\} \]

Iterates: \[ T_s(\mathcal{F}) = \operatorname{lfp} F_s \] where \[ F_s(T) \overset{\text{def}}{=} \mathcal{F} \cup \tau \sim T. \]

- \[ F_s^0(\emptyset) = \emptyset \]
- \[ F_s^1(\emptyset) = \mathcal{F} = \{ c \} \]
- \[ F_s^2(\emptyset) = \{ c, bc \} \]
- \[ F_s^3(\emptyset) = \{ c, bc, bbc, abc \} \]
- \[ F_s^n(\emptyset) = \{ c, b^i c, ab^j c \mid i \in [1, n-1], j \in [1, n-2] \} \]
- \[ T_s(\mathcal{F}) = \bigcup_{n \geq 0} F_s^n(\emptyset) = \{ c, b^i c, ab^i c \mid i \geq 1 \} \]
Symmetric finite partial trace semantics
Symmetric finite partial trace semantics

\( \mathcal{T} \): all the finite partial execution traces.
(not necessarily starting in \( \mathcal{I} \) or ending in \( \mathcal{F} \))

\[
\mathcal{T} \overset{\text{def}}{=} \{ \sigma_0, \ldots, \sigma_n \mid n \geq 0, \forall i : \sigma_i \rightarrow \sigma_{i+1} \}
= \bigcup_{n \geq 0} \Sigma \upharpoonright \mathcal{T} \upharpoonright^n
= \bigcup_{n \geq 0} \mathcal{T} \upharpoonright^n \upharpoonright \Sigma
\]

The semantics (and iterates) are forward/backward symmetric:

- \( \mathcal{T} = \mathcal{T}_p(\Sigma) \), hence \( \mathcal{T} = \text{lfp } F_{p*} \) where \( F_{p*}(\mathcal{T}) \overset{\text{def}}{=} \Sigma \cup \mathcal{T} \upharpoonright \mathcal{T} \)
  (prefix partial traces from any initial state)

- \( \mathcal{T} = \mathcal{T}_s(\Sigma) \), hence \( \mathcal{T} = \text{lfp } F_{s*} \) where \( F_{s*}(\mathcal{T}) \overset{\text{def}}{=} \Sigma \cup \mathcal{T} \upharpoonright \mathcal{T} \)
  (suffix partial traces to any final state)

- \( F_{p*}^n(\emptyset) = F_{s*}^n(\emptyset) = \bigcup_{i<n} \Sigma \upharpoonright \mathcal{T} \upharpoonright^i = \bigcup_{i<n} \mathcal{T} \upharpoonright^i \upharpoonright \Sigma = \mathcal{T} \cap \Sigma^{<n} \)
Abstracting partial traces into prefix traces

Prefix traces abstract partial traces as we forget all about partial traces not starting in $\mathcal{I}$.

**Galois connection:**

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

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

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

Similarly for the suffix traces: $\mathcal{T}_s(\mathcal{F}) = \alpha_{\mathcal{F}}(\mathcal{T})$ where $\alpha_{\mathcal{F}}(T) \overset{\text{def}}{=} T \cap (\Sigma^* \cdot \mathcal{F})$ (proof on next slide)
proof

\( \alpha_I \) and \( \gamma_I \) are monotonic. 

\[(\alpha_I \circ \gamma_I)(T) = (T \cup (\Sigma \setminus I) \cdot \Sigma^*) \cap I \cdot \Sigma^*) = T \cap I \cdot \Sigma^* \subseteq T.\]

\[(\gamma_I \circ \alpha_I)(T) = (T \cap I \cdot \Sigma^*) \cup (\Sigma \setminus I) \cdot \Sigma^* = T \cup (\Sigma \setminus I) \cdot \Sigma^* \supseteq T.\]

So, we have a Galois connection.

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

We can also retrieve the result by fixpoint transfer.

\( \mathcal{T} = \text{lfp} \ F_p^* \) where \( F_p^*(T) \overset{\text{def}}{=} \Sigma \cup T \cdot \tau \).

\( \mathcal{T}_p = \text{lfp} \ F_p \) where \( F_p(T) \overset{\text{def}}{=} I \cup T \cdot \tau \).

We have: 

\[(\alpha_I \circ F_{p^*})(T) = (\Sigma \cup T \cdot \tau) \cap (I \cdot \Sigma^*) = I \cup ((T \cdot \tau) \cap (I \cdot \Sigma^*)) = I \cup ((T \cap (I \cdot \Sigma^*)) \cdot \tau) = (F_p \circ \alpha_I)(T).\]
A first hierarchy of semantics

- $\mathcal{R}(\mathcal{I})$ and $\mathcal{C}(\mathcal{F})$: Forward/backward states
- $\mathcal{T}_p(\mathcal{I})$ and $\mathcal{T}_s(\mathcal{F})$: Prefix/suffix traces
- $\mathcal{T}$: Partial finite traces

Symbols:
- $\alpha_p$: Arrow pointing to forward/backward states
- $\alpha_I$: Arrow pointing to $\mathcal{T}_p(\mathcal{I})$
- $\alpha_F$: Arrow pointing to $\mathcal{T}_s(\mathcal{F})$
- $\alpha_T$: Arrow pointing to $\mathcal{T}$
Sufficient precondition state semantics
Sufficient preconditions

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

\[
S(\mathcal{Y}) \overset{\text{def}}{=} \{ \sigma | \forall n \geq 0, \sigma_0, \ldots, \sigma_n : (\sigma = \sigma_0 \land \forall i : \sigma_i \rightarrow \sigma_{i+1}) \implies \sigma_n \in \mathcal{Y} \}
\]

where \( \widetilde{\text{pre}}_\tau(S) \overset{\text{def}}{=} \{ \sigma | \forall \sigma' : \sigma \rightarrow \sigma' \implies \sigma' \in S \} \)

(states such that all successors satisfy \( S \), \( \widetilde{\text{pre}} \) is a complete \( \cap \)-morphism)

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

\[
S(\mathcal{Y}) = \text{gfp} \ F_S \text{ where } F_S(S) \overset{\text{def}}{=} \mathcal{Y} \cap \widetilde{\text{pre}}_\tau(S)
\]

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

\( F_S \) is continuous in the dual CPO \( (\mathcal{P}(\Sigma), \supseteq) \), because \( \widetilde{\text{pre}}_\tau \) is: \( F_S(\bigcap_{i \in I} A_i) = \bigcap_{i \in I} F_S(A_i) \).

By Kleene’s theorem in the dual, \( \text{gfp} F_S = \bigcap_{n \in \mathbb{N}} F_S^n(\Sigma) \).

We would prove by recurrence that \( F_S^n(\Sigma) = \bigcap_{i < n} \widetilde{\text{pre}}_\tau^i(\mathcal{Y}) \).
Final states $\mathcal{F}$.
Goal: when stopping, stop in $\mathcal{F}$
Final states $\mathcal{F}$.

Goal: stay in $\mathcal{Y} = \mathcal{F} \cup (\Sigma \setminus \mathcal{B})$

Iteration $F^0_S(\mathcal{Y})$
Final states $\mathcal{F}$.
Goal: stay in $\mathcal{Y} = \mathcal{F} \cup (\Sigma \setminus B)$
Iteration $F_S^1(\mathcal{Y})$
Final states $\mathcal{F}$.
Goal: stay in $\mathcal{Y} = \mathcal{F} \cup (\Sigma \setminus \mathcal{B})$
Iteration $F^2_S(\mathcal{Y})$
Final states $\mathcal{F}$.
Goal: stay in $\mathcal{Y} = \mathcal{F} \cup (\Sigma \setminus \mathcal{B})$
Iteration $F^3_S(\mathcal{Y})$
Final states $\mathcal{F}$.
Goal: stay in $\mathcal{Y} = \mathcal{F} \cup (\Sigma \setminus \mathcal{B})$
Sufficient preconditions $S(\mathcal{Y})$ to stop in $\mathcal{F}$
Final states $\mathcal{F}$.
Goal: stay in $\mathcal{Y} = \mathcal{F} \cup (\Sigma \setminus \mathcal{B})$
Sufficient preconditions $S(\mathcal{Y})$ to stop in $\mathcal{F}$

Note: $S(\mathcal{Y}) \subsetneq C(\mathcal{F})$
Sufficient preconditions and reachability

Correspondence with reachability:

We have a Galois connection:

\[(\mathcal{P}(\Sigma), \subseteq) \leftrightarrow^{S}_{\mathcal{R}} (\mathcal{P}(\Sigma), \subseteq)\]

- \(\mathcal{R}(\mathcal{I}) \subseteq \mathcal{Y} \iff \mathcal{I} \subseteq \mathcal{S}(\mathcal{Y})\)
  definition of a Galois connection
  all executions from \(\mathcal{I}\) stay in \(\mathcal{Y}\)
  \(\iff\) \(\mathcal{I}\) includes only sufficient pre-conditions for \(\mathcal{Y}\)

- so \(\mathcal{S}(\mathcal{Y}) = \bigcup \{ \mathcal{X} \mid \mathcal{R}(\mathcal{X}) \subseteq \mathcal{Y} \}\)
  by Galois connection property
  \(\mathcal{S}(\mathcal{Y})\) is the largest initial set whose reachability is in \(\mathcal{Y}\)

We retrieve Dijkstra’s weakest liberal preconditions.

(proof sketch on next slide)
proof sketch:

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

We have a Galois connection: $(\mathcal{P}(\Sigma), \subseteq) \xleftrightarrow{\text{post}_\tau} (\mathcal{P}(\Sigma), \subseteq)$.

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

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

The Galois connection can be lifted to fixpoint operators:

$(\mathcal{P}(\Sigma), \subseteq) \xleftrightarrow{x \mapsto \text{lfp}_x \ G_{\mathcal{R}}} (\mathcal{P}(\Sigma), \subseteq)$.
Applications of sufficient preconditions

Initial states such that all executions are correct: $\mathcal{I} \cap S(\mathcal{F} \cup (\Sigma \setminus \mathcal{B}))$.

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

- initial states $\mathcal{I}$: $j \in [0, 10]$ at $\bullet$
- final states $\mathcal{F}$: any memory state at $\bullet$
- blocking states $\mathcal{B}$: either final or $j > 105$ (assertion failure)

$\mathcal{I} \cap S(\mathcal{F} \cup (\Sigma \setminus \mathcal{B}))$: at $\bullet$, $j \in [0, 5]$

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

- application to inferring function contracts
- application to inferring counter-examples
- requires under-approximations to build decidable abstractions but most analyses can only provide over-approximations!

$\implies$ research topic
Research topic

Inferring sound sufficient preconditions requires under-approximations. if \( S(\mathcal{X}) \) is a sufficient precondition, any \( S^{\#}(\mathcal{X}) \subset S(\mathcal{X}) \) is stronger, thus also sufficient

Most works in abstract interpretation only target over-approximations.

The search for effective under-approximations remains an uncharted area.

Applications:

- infer function contracts
  - infer sufficient conditions on the input so that the function has no error
  - infer plausible specifications

- optimization
  - e.g., hoist dynamic checks outside loops when possible
  - replace: \( \text{for } i \text{ in } [0,n] \text{ get}(a,i) \)
  - with: \( \text{if } (\mathcal{X}) \text{ then } \text{for } i \text{ in } [0,n] \text{ unsafe-get}(a,i) \)
  - else \( \text{for } i \text{ in } [0,n] \text{ get}(a,i) \)
  - where \( \mathcal{X} \) ensures no array overflow in the loop

- infer counterexamples
  - infer conditions that ensures program mis-behavior even in the presence of non-determinism
Maximal trace semantics
The need for maximal traces

The partial trace semantics cannot distinguish between:

\[
\begin{align*}
\text{while } a \ 0 = 0 \ \text{do done} & \quad \text{while } a \ [0, 1] = 0 \ \text{do done}
\end{align*}
\]

(we get \( a^* \) for both programs)

**Principle:** restrict the semantics to complete executions only

- keep only executions finishing in a blocking state \( B \)
- add back infinite executions

  the partial semantics took into account infinite execution by including all their finite parts, but we no longer keep them as they are not maximal!

**Benefit:**

- avoid confusing prefix of infinite executions with finite executions
- allow reasoning on trace length
- allow reasoning on infinite traces (non-termination, inevitability, liveness)
Infinite traces

Notations:
- $\sigma_0, \ldots, \sigma_n, \ldots$: an infinite trace (length $\omega$)
- $\Sigma^\omega$: the set of all infinite traces
- $\Sigma^\infty \overset{\text{def}}{=} \Sigma^* \cup \Sigma^\omega$: the set of all traces

Extending the operators:
- $(\sigma_0, \ldots, \sigma_n) \cdot (\sigma'_0, \ldots) \overset{\text{def}}{=} \sigma_0, \ldots, \sigma_n, \sigma'_0, \ldots$ (append to a finite trace)
- $t \cdot t' \overset{\text{def}}{=} t$ if $t \in \Sigma^\omega$ (append to an infinite trace does nothing)
- $(\sigma_0, \ldots, \sigma_n) \Leftrightarrow (\sigma'_0, \sigma'_1 \ldots) \overset{\text{def}}{=} \sigma_0, \ldots, \sigma_n, \sigma'_1, \ldots$ when $\sigma_n = \sigma'_0$
- $t \Leftrightarrow t'$ if $t \in \Sigma^\omega$
- prefix: $x \preceq y \iff \exists u \in \Sigma^\omega: x \cdot u = y$ ($\Sigma^\omega, \preceq$) is a CPO

- $\cdot$ distributes infinite $\cup$ and $\cap$
- $\Leftrightarrow$ distributes infinite $\cup$, but not infinite $\cap$

\{a^\omega\} \Leftrightarrow (\bigcap_{n \in \mathbb{N}} \{a^m \mid n \geq m \}) = \{a^\omega\} \Leftrightarrow \emptyset = \emptyset$ but
$\bigcap_{n \in \mathbb{N}} (\{a^\omega\} \Leftrightarrow \{a^m \mid n \geq m \}) = \bigcap_{n \in \mathbb{N}} \{a^\omega\} = \{a^\omega\}$
However $A \Leftrightarrow (\bigcap_{i \in I} B_i) = \bigcup_{i \in I} (A \Leftrightarrow B_i)$ if $A \subseteq \Sigma^*$. 
Maximal traces: $\mathcal{M}_\infty \in \mathcal{P}(\Sigma^\infty)$

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

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

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

**Goal:** we look for a fixpoint characterization of $M_{\infty}$.

We consider separately finite and infinite maximal traces.

- **Finite traces:** already done!

  From the suffix partial trace semantics, recall:
  
  $$M_{\infty} \cap \Sigma^* = T_s(B) = \text{lfp } F_s$$

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

- **Infinite traces:**

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

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

  **Note:** only backward fixpoint formulation of maximal traces exist!

  (proof in following slides)
Maximal trace semantics

Infinite trace semantics: graphical illustration

\[ \begin{align*}
\mathcal{B} & \overset{\text{def}}{=} \{ c \} \\
\tau & \overset{\text{def}}{=} \{ (a, b), (b, b), (b, c) \}
\end{align*} \]

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

- \( G_s^0(\Sigma^\omega) = \Sigma^\omega \)
- \( G_s^1(\Sigma^\omega) = ab\Sigma^\omega \cup bb\Sigma^\omega \cup bc\Sigma^\omega \)
- \( G_s^2(\Sigma^\omega) = abbb\Sigma^\omega \cup bbbb\Sigma^\omega \cup abcc\Sigma^\omega \cup bbcc\Sigma^\omega \)
- \( G_s^3(\Sigma^\omega) = abbb\Sigma^\omega \cup bbbb\Sigma^\omega \cup abbc\Sigma^\omega \cup bbbc\Sigma^\omega \)
- \( G_s^n(\Sigma^\omega) = \{ ab^n t, b^{n+1} t, ab^{n-1} ct, b^n ct \mid t \in \Sigma^\omega \} \)
- \( \mathcal{M}_\infty \cap \Sigma^\omega = \bigcap_{n \geq 0} G_s^n(\Sigma^\omega) = \{ ab^\omega, b^\omega \} \)
Infinite trace semantics: proof

\[ M_\infty \cap \Sigma^\omega = \text{gfp} \ G_s \]
where \( G_s(T) \overset{\text{def}}{=} \tau \vdash T \) in \((\mathcal{P}(\Sigma^\omega), \subseteq)\)

**proof:**

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

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

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

- \( G_s^0(\Sigma^\omega) = \Sigma^\omega = (\tau \vdash^0) \vdash \Sigma^\omega \),

- \( G_s^{n+1}(\Sigma^\omega) = \tau \vdash G_s^n(\Sigma^\omega) = \tau \vdash ((\tau \vdash^n) \vdash \Sigma^\omega) = (\tau \vdash^{n+1}) \vdash \Sigma^\omega \).

\[
\text{gfp} \ G_s = \bigcap_{n \in \mathbb{N}} (\tau \vdash^n) \vdash \Sigma^\omega
\]
\[
= \{ \sigma_0, \ldots \in \Sigma^\omega \mid \forall n \geq 0: \sigma_0, \ldots, \sigma_{n-1} \in \tau \vdash^n \}
\]
\[
= \{ \sigma_0, \ldots \in \Sigma^\omega \mid \forall n \geq 0: \forall i < n: \sigma_i \rightarrow \sigma_{i+1} \}
\]
\[
= M_\infty \cap \Sigma^\omega
\]
Least fixpoint formulation of maximal traces

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

Fixpoint fusion

$\mathcal{M}_\infty \cap \Sigma^*$ is best defined on $(\mathcal{P}(\Sigma^*), \subseteq, \cup, \cap, \emptyset, \Sigma^*)$.
$\mathcal{M}_\infty \cap \Sigma^\omega$ is best defined on $(\mathcal{P}(\Sigma^\omega), \supseteq, \cap, \cup, \Sigma^\omega, \emptyset)$, the dual lattice

(we transform the greatest fixpoint into a least fixpoint!)

We mix them into a new complete lattice $(\mathcal{P}(\Sigma^\infty), \subseteq, \cup, \cap, \bot, \top)$:

- $A \subseteq B \overset{\text{def}}{\iff} (A \cap \Sigma^*) \subseteq (B \cap \Sigma^*) \land (A \cap \Sigma^\omega) \supseteq (B \cap \Sigma^\omega)$
- $A \cup B \overset{\text{def}}{=} ((A \cap \Sigma^*) \cup (B \cap \Sigma^*)) \cup ((A \cap \Sigma^\omega) \cap (B \cap \Sigma^\omega))$
- $A \cap B \overset{\text{def}}{=} ((A \cap \Sigma^*) \cap (B \cap \Sigma^*)) \cup ((A \cap \Sigma^\omega) \cup (B \cap \Sigma^\omega))$
- $\bot \overset{\text{def}}{=} \Sigma^\omega$
- $\top \overset{\text{def}}{=} \Sigma^*$

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

(proof on next slides)
**Fixpoint fusion theorem**

**Theorem:** fixpoint fusion

If \( X_1 = \text{lfp} \, F_1 \) in \((\mathcal{P}(D_1), \sqsubseteq_1)\) and \( X_2 = \text{lfp} \, F_2 \) in \((\mathcal{P}(D_2), \sqsubseteq_2)\) and \( D_1 \cap D_2 = \emptyset \),

then \( X_1 \cup X_2 = \text{lfp} \, F \) in \((\mathcal{P}(D_1 \cup D_2), \sqsubseteq)\) where:

- \( F(X) \overset{\text{def}}{=} F_1(X \cap D_1) \cup F_2(X \cap D_2) \),
- \( A \sqsubseteq B \overset{\text{def}}{\iff} (A \cap D_1) \sqsubseteq_1 (B \cap D_1) \land (A \cap D_2) \sqsubseteq_2 (B \cap D_2) \).

**proof:**

We have:

\[
F(X_1 \cup X_2) = F_1((X_1 \cup X_2) \cap D_1) \cup F_2((X_1 \cup X_2) \cap D_2) = F_1(X_1) \cup F_2(X_2) = X_1 \cup X_2,
\]

hence \( X_1 \cup X_2 \) is a fixpoint of \( F \).

Let \( Y \) be a fixpoint. Then \( Y = F(Y) = F_1(Y \cap D_1) \cup F_2(Y \cap D_2) \), hence, \( Y \cap D_1 = F_1(Y \cap D_1) \) and \( Y \cap D_1 \) is a fixpoint of \( F_1 \). Thus, \( X_1 \sqsubseteq_1 Y \cap D_1 \). Likewise, \( X_2 \sqsubseteq_2 Y \cap D_2 \). We deduce that \( X = X_1 \cup X_2 \sqsubseteq (Y \cap D_1) \cup (Y \cap D_2) = Y \), and so, \( X \) is \( F \)'s least fixpoint.

**note:** we also have \( \text{gfp} \, F = \text{gfp} \, F_1 \cup \text{gfp} \, F_2 \).
Least fixpoint formulation of maximal traces (proof)

We are now ready to finish the proof that $\mathcal{M}_\infty = \text{lfp } F_s$
in $(\mathcal{P}(\Sigma^\infty), \subseteq)$ with $F_s(T) \overset{\text{def}}{=} B \cup \tau \circ T$

proof:
We have:

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

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

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

Note: a greatest fixpoint formulation in $(\Sigma^\infty, \subseteq)$ also exists!
Abstracting maximal traces into partial traces
Abstracting maximal traces into partial traces

Finite and infinite partial trace semantics

Two steps to go from maximal to finite partial traces:

- add all partial traces
- remove infinite traces \((\text{in this order!})\)

Partial trace semantics \(\mathcal{T}_\infty\)

all finite and infinite sequences of states linked by the transition relation \(\tau:\)

\[
\mathcal{T}_\infty \overset{\text{def}}{=} \{ \sigma_0, \ldots, \sigma_n \in \Sigma^* \mid \forall i < n: \sigma_i \rightarrow \sigma_{i+1} \} \cup \\
\{ \sigma_0, \ldots, \sigma_n, \ldots \in \Sigma^\omega \mid \forall i < \omega: \sigma_i \rightarrow \sigma_{i+1} \}
\]

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

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

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

\[\text{proof: similar to the proof of } \mathcal{M}_\infty = \text{lfp } F_S.\]
Finite trace abstraction

Finite partial traces $\mathcal{T}$ are an abstraction of all partial traces $\mathcal{T}_\infty$
(forget about infinite executions)

We have a Galois embedding:

\[(\mathcal{P}(\Sigma^\infty), \sqsubseteq) \overset{\gamma_*}{\longrightarrow} (\mathcal{P}(\Sigma^*), \subseteq)\]

- $\sqsubseteq$ is the fused ordering on $\Sigma^* \cup \Sigma^\omega$:
  \[A \sqsubseteq B \iff (A \cap \Sigma^*) \subseteq (B \cap \Sigma^*) \land (A \cap \Sigma^\omega) \supseteq (B \cap \Sigma^\omega)\]

- $\alpha_*(\mathcal{T}) \overset{\text{def}}{=} \mathcal{T} \cap \Sigma^*$
  (remove infinite traces)

- $\gamma_*(\mathcal{T}) \overset{\text{def}}{=} \mathcal{T}$
  (embedding)

- $\mathcal{T} = \alpha_*(\mathcal{T}_\infty)$
  (proof on next slide)
Finite trace abstraction (proof)

proof:

We have Galois embedding because:

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

Recall that $T_\infty = \text{lfp } F_{s*}$ in $(\mathcal{P}(\Sigma^\infty), \subseteq)$ and $T = \text{lfp } F_{s*}$ in $(\mathcal{P}(\Sigma^*), \subseteq)$, where $F_{s*}(T) \overset{\text{def}}{=} \Sigma \cup T \circ \tau$.

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

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

We have a Galois connection:

\[
(\mathcal{P}(\Sigma^\infty \setminus \{\epsilon\}), \subseteq) \overset{\gamma \leq}{\iff} (\mathcal{P}(\Sigma^\infty \setminus \{\epsilon\}), \subseteq)
\]

- \(\alpha \leq(T) \overset{\text{def}}{=} \{ t \in \Sigma^\infty \setminus \{\epsilon\} | \exists u \in T : t \leq u \}\)
  (set of all non-empty prefixes of traces in \(T\))

- \(\gamma \leq(T) \overset{\text{def}}{=} \{ t \in \Sigma^\infty \setminus \{\epsilon\} | \forall u \in \Sigma^\infty \setminus \{\epsilon\} : u \leq t \Rightarrow u \in T \}\)
  (traces with non-empty prefixes in \(T\))

proof:

\(\alpha \leq\) and \(\gamma \leq\) are monotonic.

\((\alpha \leq \circ \gamma \leq)(T) = \{ t \in T | \rho_p(t) \subseteq T \} \subseteq T\) \hspace{1em} (prefix-closed trace sets).

\((\gamma \leq \circ \alpha \leq)(T) = \rho_p(T) \supseteq T).\)
Finite and infinite partial traces $T_\infty$ are an abstraction of maximal traces $M_\infty$: $T_\infty = \alpha_\preceq (M_\infty)$.

proof:
Firstly, $T_\infty$ and $\alpha_\preceq (M_\infty)$ coincide on infinite traces. Indeed, $T_\infty \cap \Sigma^\omega = M_\infty \cap \Sigma^\omega$ and $\alpha_\preceq$ does not add infinite traces, so: $T_\infty \cap \Sigma^\omega = \alpha_\preceq (M_\infty) \cap \Sigma^\omega$.

We now prove that they also coincide on finite traces. Assume $\sigma_0, \ldots, \sigma_n \in \alpha_\preceq (M_\infty)$, then $\forall i < n: \sigma_i \rightarrow \sigma_{i+1}$, so, $\sigma_0, \ldots, \sigma_n \in T_\infty$.
Assume $\sigma_0, \ldots, \sigma_n \in T_\infty$, then it can be completed into a maximal trace, either finite or infinite, and so, $\sigma_0, \ldots, \sigma_n \in \alpha_\preceq (M_\infty)$.

Note: no fixpoint transfer applies here.
Abstracting maximal traces into partial traces

Enriched hierarchy of semantics

\[ R(I) \]
\[ \alpha_p \]
\[ T_p(I) \]
\[ \alpha_I \]
\[ T \]
\[ \alpha_* \]
\[ T_\infty \]
\[ \alpha_\preceq \]
\[ M_\infty \]
\[ C(F) \]
\[ \alpha_p \]
\[ T_s(F) \]
\[ \alpha_F \]

forward/backward states
prefix/suffix finite traces
partial finite traces
partial traces
maximal traces

See [Cous02] for more semantics in this diagram.
Safety and liveness trace properties
Maximal 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 \( P' \overset{\text{def}}{=} P \cup ((\Sigma \setminus \mathcal{I}) \cdot \Sigma^\infty) \)

Examples:

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

Idea: a safety property $P$ models that "nothing bad ever occurs"

- $P$ is provable by exhaustive testing;
  
  (observe the prefix trace semantics: $T_p(I) \subseteq P$)

- $P$ is disprovable by finding a single finite execution not in $P$.

Examples:

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

- ordering: $P \overset{\text{def}}{=} \Sigma^\infty \setminus \left( (\Sigma \setminus \{a\})^* \cdot b \cdot \Sigma^\infty \right)$,
  
  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 $P \overset{\text{def}}{=} \Sigma^*$ is not a safety property.
  disproof requires exhibiting an infinite execution
Safety and liveness trace properties

Definition of safety properties

**Reminder:** finite prefix abstraction (simplified to allow $\epsilon$)

$$(\mathcal{P}(\Sigma^\infty), \subseteq) \xrightarrow{\alpha_* \preceq} (\mathcal{P}(\Sigma^*), \subseteq) \xrightarrow{\gamma_* \preceq} (\mathcal{P}(\Sigma^\omega), \subseteq)$$

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

The associated upper closure $\rho_* \preceq \overset{\text{def}}{=} \gamma \preceq \circ \alpha \preceq$ is:

$$\rho_* \preceq = \lim \circ \rho_p$$ where:

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

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

Examples and counter-examples:

- state property \( P \overset{\text{def}}{=} S^\infty \) for \( S \subseteq \Sigma \):
  \[ \rho_p(S^\infty) = \lim(S^\infty) = S^\infty \] safety;

- termination \( P \overset{\text{def}}{=} \Sigma^* \):
  \[ \rho_p(\Sigma^*) = \Sigma^* \text{, but } \lim(\Sigma^*) = \Sigma^\infty \neq \Sigma^* \] not safety;

- even number of steps \( P \overset{\text{def}}{=} (\Sigma^2)^\infty \):
  \[ \rho_p((\Sigma^2)^\infty) = \Sigma^\infty \neq (\Sigma^2)^\infty \] not safety.
Proving safety properties

Proving that a program satisfies a safety property $P$ is equivalent to proving that its finite prefix abstraction does

$$T_p(I) \subseteq P$$

proof sketch:

Soundness. Using the Galois connection between $M_\infty$ and $T$, we get:

$$M_\infty \cap (I \cdot \Sigma^\infty) \subseteq \rho_{\star \leq} (M_\infty \cap (I \cdot \Sigma^\infty)) = \gamma_{\star \leq} (\alpha_{\star \leq} (M_\infty \cap (I \cdot \Sigma^\infty))) = \gamma_{\star \leq} (\alpha_{\star \leq} (M_\infty) \cap (I \cdot \Sigma^\star)) = \gamma_{\star \leq} (T \cap (I \cdot \Sigma^\star)) = \gamma_{\star \leq} (T_p(I)).$$

As $T_p(I) \subseteq P$, we have, by monotony, $\gamma_{\star \leq} (T_p(I)) \subseteq \gamma_{\star \leq} (P) = P$.

Hence $M_\infty \cap (I \cdot \Sigma^\infty) \subseteq P$.

Completeness. $T_p(I)$ provides an inductive invariant for $P$. 
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:** $P \overset{\text{def}}{=} \Sigma^*$,
- **inevitability:** $P \overset{\text{def}}{=} \Sigma^* \cdot a \cdot \Sigma^\infty$,
  ($a$ eventually occurs in all executions)
- state properties are **not** liveness properties.
**Definition:**  \( P \in \mathcal{P}(\Sigma^\infty) \) is a liveness property if \( \rho_{\mathcal{P}}(P) = \Sigma^\infty \).

**Examples and counter-examples:**

- **termination** \( P \overset{\text{def}}{=} \Sigma^* \):
  \[
  \rho_p(\Sigma^*) = \Sigma^* \text{ and } \lim(\Sigma^*) = \Sigma^\infty \implies \text{liveness};
  \]

- **inevitability:** \( P \overset{\text{def}}{=} \Sigma^* \cdot a \cdot \Sigma^\infty \)
  \[
  \rho_p(P) = P \cup \Sigma^* \text{ and } \lim(P \cup \Sigma^*) = \Sigma^\infty \implies \text{liveness};
  \]

- **state property** \( P \overset{\text{def}}{=} S^\infty \text{ for } S \subseteq \Sigma \):
  \[
  \rho_p(S^\infty) = \lim(S^\infty) = S^\infty \neq \Sigma^\infty \text{ if } S \neq \Sigma \implies \text{not liveness};
  \]

- **maximal execution time** \( P \overset{\text{def}}{=} \Sigma^{\leq k} \):
  \[
  \rho_p(\Sigma^{\leq k}) = \lim(\Sigma^{\leq k}) = \Sigma^{\leq k} \neq \Sigma^\infty \implies \text{not liveness};
  \]

- **the only property which is both safety and liveness is** \( \Sigma^\infty \).
Proving liveness properties

**Variance proof method:** (informal definition)

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

**Example:** termination proof

- find $f : \Sigma \rightarrow S$ where $(S, \sqsubseteq)$ is **well-ordered**;
  
  ($f$ is called a “ranking function”)

- $\sigma \in \mathcal{B} \implies f = \min S$;

- $\sigma \rightarrow \sigma' \implies f(\sigma') \sqsubseteq f(\sigma)$.

($f$ counts the number of steps remaining before termination)
Trace topology

A topology on a set can be defined as:
– either a family of open sets (closed under union)
– or family of closed sets (closed under intersection)

**Trace topology:** on sets of traces in $\Sigma^\infty$

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

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

- $\rho(x) \overset{\text{def}}{=} \cap \{ c \in C \mid x \subseteq c \}$ (upper closure operator in $(\mathcal{P}(X), \subseteq)$)
- on our trace topology, $\rho = \rho^* \triangleleft$.

**Dense sets:**

- $x \subseteq X$ is dense if $\rho(x) = X$;
- on our trace topology, dense sets are liveness properties.
Theorem: decomposition on a topological space

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

proof:

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

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

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

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
Bibliography
