

# Lecture 8 – Trace Properties

## AAA551: Programming Language Theory

Jihyeok Park



2026 Spring

- Trace Semantics
  - Transition Systems
  - Traces
  - Finite Trace Semantics
  - Infinite Trace Semantics
  - Compositionality
- Example
  - Non-Deterministic Imperative Language – NIMP

The goal of **verification** is to show that the **programs semantics**  $\llbracket P \rrbracket$  satisfies a specific **desired property**  $\mathbb{P}$ :

$$\llbracket P \rrbracket \subseteq \mathbb{P}$$

We will discuss classes of properties:

- **State Properties:** “Certain states are never observed”
- **Trace Properties:** “Certain traces are never observed”
  - **Safety Properties:** “Nothing bad happens”
  - **Liveness Properties:** “Something good eventually happens”

## 1. State and Trace Properties

## 2. Safety Properties

- Irreducibility and Finite Bad Prefix
- Prefix Closure and Limit
- Formal Definition of Safety Properties
- Verification by Invariance

## 3. Liveness Properties

- Eventuality
- Formal Definition of Liveness Properties
- Verification by Variance

## 4. Decomposition of Trace Properties

## 1. State and Trace Properties

## 2. Safety Properties

- Irreducibility and Finite Bad Prefix
- Prefix Closure and Limit
- Formal Definition of Safety Properties
- Verification by Invariance

## 3. Liveness Properties

- Eventuality
- Formal Definition of Liveness Properties
- Verification by Variance

## 4. Decomposition of Trace Properties

Consider a state transition system  $\mathcal{T} = (\mathbb{S}, \rightarrow, \mathbb{S}_I)$ :

- $\mathbb{S}$  is a set of states
- $\rightarrow \subseteq \mathbb{S} \times \mathbb{S}$  is a transition relation
- $\mathbb{S}_I \subseteq \mathbb{S}$  is a set of initial states

## Definition (State Property)

A **state property** is a subset of states  $\mathbb{P} \subseteq \mathbb{S}$ .

$\mathbb{P}$  is satisfied if and only if all **reachable states** of  $P$  are belong to  $\mathbb{P}$ :

$$[[P]]_{\mathcal{R}} \subseteq \mathbb{P}$$

where

$$[[P]]_{\mathcal{R}} = \{\sigma_n \in \mathbb{S} \mid \exists \langle \sigma_0, \dots, \sigma_n \rangle \in [[P]]^*. \sigma_0 \in \mathbb{S}_I\}$$

For example, “*the variable  $x$  is always positive*” is a state property.

## Definition (Trace Property)

A **trace property** is a subset of traces  $\mathbb{P} \subseteq \mathbb{S}^\infty$ .

$\mathbb{P}$  is satisfied if and only if all traces of  $P$  are belong to  $\mathbb{P}$ :

$$[[P]]^\infty \subseteq \mathbb{P}$$

Recall that

- $[[P]]^*$  is the **finite trace semantics** of  $P$ .
- $[[P]]^\omega$  is the **infinite trace semantics** of  $P$ .
- $[[P]]^\infty$  is the **finite and infinite trace semantics** of  $P$ .

Note that all state properties are trace properties.

For example, “*the variable  $x$  is eventually becomes zero*” is a trace property but not a state property.

To **verify** a property  $\mathbb{P}$  of a program  $P$ , we need to show that the program semantics  $\llbracket P \rrbracket$  is a subset of the property  $\mathbb{P}$ :

## Definition (Verification)

A program  $P$  **satisfies** a property  $\mathbb{P}$  if and only if:

$$\llbracket P \rrbracket \subseteq \mathbb{P}$$

## Definition (Stronger Property)

A  $\mathbb{P}$  is **stronger** than  $\mathbb{P}'$  if and only if  $\mathbb{P}$  is a subset of  $\mathbb{P}'$ :

$$\mathbb{P} \subseteq \mathbb{P}'$$

If a program satisfies a stronger property, it also satisfies a weaker property:

$$\mathbb{P} \subseteq \mathbb{P}' \quad \text{and} \quad \llbracket P \rrbracket \subseteq \mathbb{P} \implies \llbracket P \rrbracket \subseteq \mathbb{P}'$$

## 1. State and Trace Properties

## 2. Safety Properties

- Irreducibility and Finite Bad Prefix
- Prefix Closure and Limit
- Formal Definition of Safety Properties
- Verification by Invariance

## 3. Liveness Properties

- Eventuality
- Formal Definition of Liveness Properties
- Verification by Variance

## 4. Decomposition of Trace Properties

The informal definition of **safety properties** is:

*“Nothing bad happens”*

For example,

- *“the variable  $x$  is always positive”* is a safety property.  
(**bad thing**: reaching a state where  $x \leq 0$ )
- *“the output of the sort function is always sorted”* is a safety property.  
(**bad thing**: producing an output that is not sorted)

All state properties are safety properties, but there exist safety properties that are not state properties. For example,

- *“output of the sort function should have same elements as the input”* is a safety but not a state property.  
(**bad thing**: producing an output that does not have the same elements as the input)

Let's formalize the definition of safety properties.

Intuitively, a safety property  $\mathbb{P}$  satisfies the following two conditions:

- 1 **Irreducibility**: *“Once a bad thing happens, it cannot be undone.”*

$$\forall \tau, \tau' \in \mathbb{S}^\infty. \quad \tau \preceq \tau' \implies (\tau \notin \mathbb{P} \implies \tau' \notin \mathbb{P})$$

which is equivalent to:

$$\forall \tau, \tau' \in \mathbb{S}^\infty. \quad \tau \preceq \tau' \implies (\tau \in \mathbb{P} \iff \tau' \in \mathbb{P})$$

Thus, a safety property is **prefix closed**:

*“If a trace  $\tau' \in \mathbb{S}^\infty$  is in  $\mathbb{P}$ , then all prefixes of  $\tau'$  are also in  $\mathbb{P}$ .”*

Let's define a **prefix closure operator**  $\text{PCI}$ .

② **Finite Bad Prefix:** *“A bad thing can be observed in a finite time.”*

$$\forall \tau \in \mathbb{S}^\infty. \quad \tau \notin \mathbb{P} \implies (\exists \tau' \in \mathbb{S}^*. \quad \tau' \preceq \tau \quad \wedge \quad \tau' \notin \mathbb{P})$$

which is equivalent to:

$$\forall \tau \in \mathbb{S}^\infty. \quad \tau \in \mathbb{P} \iff (\forall \tau' \in \mathbb{S}^*. \quad (\tau' \preceq \tau \implies \tau' \in \mathbb{P}))$$

In other words,

*“If all finite prefixes of a trace  $\tau \in \mathbb{S}^\infty$  are in  $\mathbb{P}$ , then  $\tau$  is in  $\mathbb{P}$ .”*

We need to defined a **prefix limit operator**  $\text{Lim}$ .

## Definition (Prefix Closure)

The **prefix closure operator**  $\text{PCI} : \mathcal{P}(\mathbb{S}^\infty) \rightarrow \mathcal{P}(\mathbb{S}^*)$  is defined as:

$$\text{PCI}(X) = \{\tau' \in \mathbb{S}^* \mid \exists \tau \in X. \tau' \preceq \tau\}$$

Note that it takes a set of **finite and infinite traces**  $X$ , but returns a set of **finite traces** that are prefixes of the traces in  $X$ .

The PCI operator is:

- **monotone:**  $X \subseteq Y \implies \text{PCI}(X) \subseteq \text{PCI}(Y)$
- **idempotent:**  $\text{PCI}(\text{PCI}(X)) = \text{PCI}(X)$

## Definition (Prefix Limit)

The **prefix limit operator**  $\text{Lim} : \mathcal{P}(\mathbb{S}^\infty) \rightarrow \mathcal{P}(\mathbb{S}^\infty)$  is defined as:

$$\text{Lim}(X) = X \cup \{\tau \in \mathbb{S}^\infty \mid \forall \tau' \in \mathbb{S}^*. (\tau' \preceq \tau \implies \tau' \in X)\}$$

The Lim operator is:

- **monotone:**  $X \subseteq Y \implies \text{Lim}(X) \subseteq \text{Lim}(Y)$
- **extensive:**  $X \subseteq \text{Lim}(X)$
- **idempotent:**  $\text{Lim}(\text{Lim}(X)) = \text{Lim}(X)$

Let's say a trace property  $\mathbb{P} \subseteq \mathbb{S}^\infty$  is a safety property if and only if its **limit of prefix closure** is itself:

## Definition (Safety Property)

A trace property  $\mathbb{P} \subseteq \mathbb{S}^\infty$  is a **safety property** if and only if:

$$\text{Safe}(\mathbb{P}) = \mathbb{P}$$

where  $\text{Safe} = \text{Lim} \circ \text{PCI}$

One interesting fact is that we can always construct a safety property from any trace property by applying the Safe operator:

## Theorem

*For a given trace property  $\mathbb{P} \subseteq \mathbb{S}^\infty$ ,  $\text{Safe}(\mathbb{P})$  is a safety property:*

$$\text{Safe}(\text{Safe}(\mathbb{P})) = \text{Safe}(\mathbb{P})$$

We need to prove that Safe is a **idempotent**.

Let's prove that the Safe operator is an upper closure operator.

## Theorem

The  $\text{Safe} : \mathcal{P}(\mathbb{S}^\infty) \rightarrow \mathcal{P}(\mathbb{S}^\infty)$  operator is an **upper closure operator**, which satisfies the following properties:

- **monotone:**  $X \subseteq X' \implies \text{Safe}(X) \subseteq \text{Safe}(X')$
- **extensive:**  $X \subseteq \text{Safe}(X)$
- **idempotent:**  $\text{Safe}(\text{Safe}(X)) = \text{Safe}(X)$

- Safe is **monotone**:

$$X \subseteq X' \implies \text{Safe}(X) \subseteq \text{Safe}(X')$$

Since PCI and Lim are monotone, their composition is also monotone.

- Safe is **extensive**:

$$X \subseteq \text{Safe}(X)$$

- If  $\tau \in X \cap \mathbb{S}^*$ , it is a prefix of itself, and Lim is extensive, so:

$$\tau \in \text{PCI}(X) \subseteq \text{Lim}(\text{PCI}(X)) = \text{Safe}(X)$$

- If  $\tau \in X \cap \mathbb{S}^\omega$ , then all finite prefixes of  $\tau$  are in  $\text{PCI}(X)$ , so  $\tau \in \text{Lim}(\text{PCI}(X)) = \text{Safe}(X)$ .

- Safe is **idempotent**:

$$\text{Safe}(\text{Safe}(X)) = \text{Safe}(X)$$

Let's prove the two directions separately:

- $\text{Safe}(X) \subseteq \text{Safe}(\text{Safe}(X))$  :

Since Safe is extensive, we are done.

- $\text{Safe}(\text{Safe}(X)) \subseteq \text{Safe}(X)$  :

$$\tau \in \text{Safe}(\text{Safe}(\mathbb{P}))$$

$$\implies \forall \tau' \in \mathbb{S}^*. (\tau' \preceq \tau. \implies \tau' \in \text{PCI} \circ \text{Safe}(\mathbb{P})) \quad (\text{by def. of Lim})$$

$$\implies \forall \tau' \in \mathbb{S}^*. (\tau' \preceq \tau. \implies \exists \tau'' \in \text{Safe}(\mathbb{P}). \tau' \preceq \tau'') \quad (\text{by def. of PCI})$$

$$\implies \forall \tau' \in \mathbb{S}^*. (\tau' \preceq \tau. \implies \tau' \in \text{PCI}(\mathbb{P})) \quad (\text{by def. of Lim})$$

$$\implies \tau \in \text{Safe}(\mathbb{P}) \quad (\text{by def. of Lim})$$

To verify that a program  $P$  satisfies a safety property  $\mathbb{P}$ , we need to think about **invariance** (i.e., “what is always true?”) of the program.

## Definition (Invariance)

Given transition system  $\mathcal{T} = (\mathbb{S}, \rightarrow, \mathbb{S}_{\mathcal{I}})$  for a program  $P$ ,

$$\llbracket \mathcal{T} \rrbracket^* = \mathbf{lfp}(F_*) = \bigcup_{n \in \mathbb{Z}} F_*^n(\emptyset)$$

where  $\mathcal{I} = \{\epsilon\} \cup \{\langle \sigma \rangle \mid \sigma \in \mathbb{S}\}$  and  $F_* : \mathcal{P}(\mathbb{S}^*) \rightarrow \mathcal{P}(\mathbb{S}^*)$ :

$$F_*(X) = \mathcal{I} \cup \{\langle \sigma_0, \dots, \sigma_n, \sigma' \rangle \mid \langle \sigma_0, \dots, \sigma_n \rangle \in X \wedge \sigma_n \rightarrow \sigma'\}$$

A set of finite traces  $\mathbb{I} \subseteq \mathbb{S}^*$  is an **invariant** if and only if  $\mathbb{I}$  is closed under  $F_*$ :

$$F_*(\mathbb{I}) \subseteq \mathbb{I}$$

```

 $\ell_0$  : (true)
      s = 0;
 $\ell_1$  : (s = 0)
      i = 0;
 $\ell_2$  : (i = 0  $\wedge$  s = 0)
      while (i < n) {
 $\ell_3$  : (0  $\leq$  i < n  $\wedge$  s =  $\sum_{k=0}^{i-1} x[k]$ )
          s = s + x[i];
 $\ell_4$  : (0  $\leq$  i < n  $\wedge$  s =  $\sum_{k=0}^i x[k]$ )
          i = i + 1;
 $\ell_5$  : (1  $\leq$  i  $\leq$  n  $\wedge$  s =  $\sum_{k=0}^{i-1} x[k]$ )
      }
 $\ell_6$  : (i = n  $\wedge$  s =  $\sum_{k=0}^{n-1} x[k]$ )
  
```

Consider a program that computes the sum of an array  $x$  of size  $n$ .

The **safety property** we want to verify is  $s = \sum_{k=0}^{n-1} x[k]$  at the end of the program.

We define the **local invariant** at each program point  $\ell_i \in \mathbb{L}$  as a memory predicate  $m_i$  that holds at  $\ell_i$ .

The **global invariant** is the union of all local invariants:

$$\mathbb{I} = \{ \langle \ell_0, m_0 \rangle, \dots, \langle \ell_6, m_6 \rangle \}$$

### Theorem (Soundness)

If  $\mathbb{I}$  is an invariant of a program  $P$  and  $\mathbb{I}$  is stronger than a safety property  $\mathbb{P}$ , then  $P$  satisfies  $\mathbb{P}$ .

Since  $F_*(\mathbb{I}) \subseteq \mathbb{I}$  and  $F_*$  is monotone, we have:

$$\emptyset \subseteq \mathbb{I} \implies F_*(\emptyset) \subseteq F_*(\mathbb{I}) \subseteq \mathbb{I} \implies \dots F_*^n(\emptyset) \subseteq \mathbb{I}$$

Thus,  $\llbracket P \rrbracket^* = \mathbf{lfp}(F_*) = \bigcup_{n \in \mathbb{Z}} F_*^n(\emptyset) \subseteq \mathbb{I}$ .

Now, we have the following:

$$\begin{aligned} \llbracket P \rrbracket^\infty &= \text{Safe}(\llbracket P \rrbracket^*) && \text{(by def. of Safe)} \\ &\subseteq \text{Safe}(\mathbb{I}) && \text{(since above)} \\ &\subseteq \text{Safe}(\mathbb{P}) && \text{(since Safe is monotone and } \mathbb{I} \subseteq \mathbb{P} \text{)} \\ &= \mathbb{P} && \text{(since } \mathbb{P} \text{ is a safety property)} \end{aligned}$$

## Theorem (Completeness)

*If a program  $P$  satisfies a safety property  $\mathbb{P}$ , then there exists an invariant  $\mathbb{I}$  such that  $\mathbb{I}$  is stronger than  $\mathbb{P}$ .*

The program semantics itself is an invariant that is stronger than the safety property:

$$\mathbb{I} = \llbracket P \rrbracket^*$$

since it is a least fixed point of  $F_*$ , it is closed under  $F_*$ .

Then, since  $P$  satisfies  $\mathbb{P}$ , we have  $\mathbb{I} = \llbracket P \rrbracket^* \subseteq \mathbb{P}$ .

It does **NOT** mean that we can always **algorithmically construct** an **invariant** from a safety property, since the program semantics is typically **undecidable**.

## 1. State and Trace Properties

## 2. Safety Properties

- Irreducibility and Finite Bad Prefix
- Prefix Closure and Limit
- Formal Definition of Safety Properties
- Verification by Invariance

## 3. Liveness Properties

- Eventuality
- Formal Definition of Liveness Properties
- Verification by Variance

## 4. Decomposition of Trace Properties

The informal definition of **liveness properties** is:

*“Something good eventually happens”*

For example,

- *“the variable  $x$  eventually becomes zero”* is a liveness property.  
(**good thing**: reaching a state where  $x = 0$ )
- *“a program always terminates”* is a liveness property.  
(**good thing**: reaching a terminal state)

How to formalize the following **eventuality** of liveness properties?

*“Something good eventually happens”*

We can formalize it as follows:

$$\forall \tau \in \mathbb{S}^*. \quad \exists \tau' \in \mathbb{P}. \quad \tau \preceq \tau'$$

By definition of PCI, the above is equivalent to:

$$\forall \tau \in \mathbb{S}^*. \quad \tau \in \text{PCI}(\mathbb{P})$$

which means that **(1)**  $\text{PCI}(\mathbb{P}) = \mathbb{S}^*$ .

If we apply  $\text{Lim}$  to both sides, we have **(2)**  $\text{Lim} \circ \text{PCI}(\mathbb{P}) = \text{Lim}(\mathbb{S}^*) = \mathbb{S}^\infty$ .

Finally, **(3)**  $\mathbb{P} = \mathbb{P} \cup \emptyset = \mathbb{P} \cup (\mathbb{S}^\infty \setminus \text{Lim} \circ \text{PCI}(\mathbb{P})) = \mathbb{P} \cup (\mathbb{S}^\infty \setminus \text{Safe}(\mathbb{P}))$ .

In summary, we have the following three equivalent definitions of liveness properties:

## Definition (Liveness Property)

A trace property  $\mathbb{P} \subseteq \mathbb{S}^\infty$  is a **liveness property** if and only if it satisfies one of the following equivalent conditions:

- 1  $\text{PCI}(\mathbb{P}) = \mathbb{S}^*$
- 2  $\text{Lim} \circ \text{PCI}(\mathbb{P}) = \mathbb{S}^\infty$
- 3  $\text{Live}(\mathbb{P}) = \mathbb{P}$

where  $\text{Live} : \mathcal{P}(\mathbb{S}^\infty) \rightarrow \mathcal{P}(\mathbb{S}^\infty)$  is defined as:

$$\text{Live}(\mathbb{P}) = \mathbb{P} \cup (\mathbb{S}^\infty \setminus \text{Safe}(\mathbb{P}))$$

## Theorem

The  $\text{Live} : \mathcal{P}(\mathbb{S}^\infty) \rightarrow \mathcal{P}(\mathbb{S}^\infty)$  operator satisfies the following properties:

- **idempotent:**  $\text{Live}(\text{Live}(X)) = \text{Live}(X)$
- **extensive:**  $X \subseteq \text{Live}(X)$

Note that Live is **not monotone** while Safe is monotone.

Since Live is idempotent, the following is true:

## Theorem

For a given trace property  $\mathbb{P} \subseteq \mathbb{S}^\infty$ ,  $\text{Live}(\mathbb{P})$  is a **liveness property**:

$$\text{Live}(\text{Live}(\mathbb{P})) = \text{Live}(\mathbb{P})$$

To verify that a program  $P$  satisfies a liveness property  $\mathbb{P}$ , we need to consider a **ranking function** that maps a state to a well-founded set, which measures how far the program is from reaching a good state.

## Definition (Ranking Function)

For a given transition system  $\mathcal{T} = (\mathbb{S}, \rightarrow, \mathbb{S}_I)$  for a program  $P$ , a **ranking function** is a function  $\phi : \mathbb{S} \rightarrow W$  satisfying the following condition:

- $(W, \sqsubseteq)$  is **well-founded ordering**,  
which means that all non-empty subset of  $W$  has a minimal element:

$$\forall S \subseteq W. \quad S \neq \emptyset \implies \exists m \in S. \quad \forall x \in S. \quad \neg(x \sqsubset m)$$

- All transitions decrease the rank:

$$\forall \sigma, \sigma' \in \mathbb{S}. \quad \sigma \rightarrow \sigma' \implies \phi(\sigma) \sqsupset \phi(\sigma')$$

Let's define a ranking function for the sum of an array program.

Ranking function:

	$\phi : \mathbb{S} \quad \rightarrow \quad \mathbb{N}$
$\ell_0 : \text{ s } = 0;$	$\langle \ell_0, m \rangle \mapsto 3n + 6$
$\ell_1 : \text{ i } = 0;$	$\langle \ell_1, m \rangle \mapsto 3n + 5$
$\ell_2 : \text{ while } (i < n) \{$	$\langle \ell_2, m \rangle \mapsto 3n + 4$
$\ell_3 : \quad \text{ s } = \text{ s } + \text{ x}[\text{i}];$	$\langle \ell_3, m \rangle \mapsto 3(n - m(i)) + 3$
$\ell_4 : \quad \text{ i } = \text{ i } + 1;$	$\langle \ell_4, m \rangle \mapsto 3(n - m(i)) + 2$
$\ell_5 : \}$	$\langle \ell_5, m \rangle \mapsto 3(n - m(i)) + 4$
$\ell_6 :$	$\langle \ell_6, m \rangle \mapsto 0$

We skip the soundness and completeness of verification by variance.

## 1. State and Trace Properties

## 2. Safety Properties

- Irreducibility and Finite Bad Prefix
- Prefix Closure and Limit
- Formal Definition of Safety Properties
- Verification by Invariance

## 3. Liveness Properties

- Eventuality
- Formal Definition of Liveness Properties
- Verification by Variance

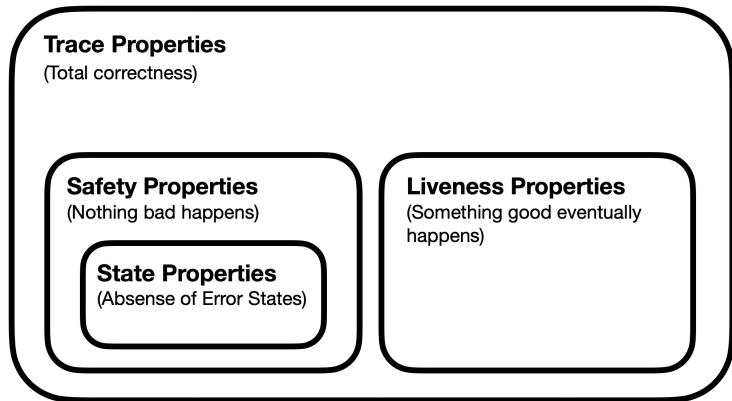
## 4. Decomposition of Trace Properties

## Theorem

*Any trace property  $\mathbb{P}$  can be decomposed into the intersection of a safety property  $\text{Safe}(\mathbb{P})$  and a liveness property  $\text{Live}(\mathbb{P})$ :*

$$\mathbb{P} = \text{Safe}(\mathbb{P}) \cap \text{Live}(\mathbb{P})$$

$$\begin{aligned} \text{Safe}(\mathbb{P}) \cap \text{Live}(\mathbb{P}) &= \text{Safe}(\mathbb{P}) \cap (\mathbb{P} \cup (\mathbb{S}^\infty \setminus \text{Safe}(\mathbb{P}))) \\ &\text{(by def. of Live)} \\ &= (\text{Safe}(\mathbb{P}) \cap \mathbb{P}) \cup (\text{Safe}(\mathbb{P}) \cap (\mathbb{S}^\infty \setminus \text{Safe}(\mathbb{P}))) \\ &\text{(by distributive law of } \cap \text{ over } \cup) \\ &= \mathbb{P} \cup \emptyset \\ &\text{(by Safe is extensive: } \mathbb{P} \subseteq \text{Safe}(\mathbb{P})) \\ &= \mathbb{P} \end{aligned}$$



The proof of any trace property can be decomposed into:

- proof of a safety property  $\text{Safe}(\mathbb{P})$  by invariance
- proof of a liveness property  $\text{Live}(\mathbb{P})$  by variance

- Abstract Machines

Jihyeok Park  
jihyeok\_park@korea.ac.kr  
<https://plrg.korea.ac.kr>