

Lecture 3 – Small-Step Operational Semantics

AAA551: Programming Language Theory

Jihyeok Park



2026 Spring

- Programming languages
- Mathematical preliminaries
 - Binary relations and functions
 - Inductively defined sets
- Big-step operational semantics
 - Example: VAE
 - Scala implementation
 - Derivation
- Simple imperative language – IMP
 - Big-step operational semantics of IMP
 - Equivalence of statements
 - Determinism

1. Small-Step Operational Semantics

Example: VAE

Multi-Step Evaluation

Derivation

Big-Step vs. Small-Step Operational Semantics

2. Soundness

Well-Formed Configurations

Progress

Preservation

Termination

Soundness vs. Completeness

3. Simple Imperative Language – IMP

Expressions

Statements

1. Small-Step Operational Semantics

Example: VAE

Multi-Step Evaluation

Derivation

Big-Step vs. Small-Step Operational Semantics

2. Soundness

Well-Formed Configurations

Progress

Preservation

Termination

Soundness vs. Completeness

3. Simple Imperative Language – IMP

Expressions

Statements

There are two main styles of operational semantics:

- **Big-Step Operational Semantics** defines the meaning of a program by specifying how it executes on a machine in **one big step**.

$$\frac{\dots}{\vdash e \Rightarrow n}$$

(An expression e evaluates to n .)

- **Small-Step Operational Semantics** defines the meaning of a program by specifying how it executes on a machine **step-by-step**.

$$e \rightarrow e' \rightarrow e'' \rightarrow \dots \rightarrow n$$

(An expression e is reduced to e' , then to e'' , and so on until n .)

$$e ::= n \mid e + e \mid e * e \mid x := e; e \mid x$$

$$\boxed{\sigma \vdash e \Rightarrow n}$$

$$\frac{}{\sigma \vdash n \Rightarrow n}$$

$$\frac{\sigma \vdash e_1 \Rightarrow n_1 \quad \sigma \vdash e_2 \Rightarrow n_2}{\sigma \vdash e_1 + e_2 \Rightarrow n_1 + n_2} \qquad \frac{\sigma \vdash e_1 \Rightarrow n_1 \quad \sigma \vdash e_2 \Rightarrow n_2}{\sigma \vdash e_1 * e_2 \Rightarrow n_1 \times n_2}$$

$$\frac{\sigma \vdash e_1 \Rightarrow n_1 \quad \sigma[x \mapsto n_1] \vdash e_2 \Rightarrow n_2}{\sigma \vdash x := e_1; e_2 \Rightarrow n_2} \qquad \frac{x \in \text{dom}(\sigma)}{\sigma \vdash x \Rightarrow \sigma(x)}$$

The small-step operational semantics of VAE is defined by the following **small-step evaluation relation** \rightarrow on configurations $\langle \sigma, e \rangle$:

$$\boxed{\langle \sigma, e \rangle \rightarrow \langle \sigma, e \rangle}$$

where a **configuration** $\langle \sigma, e \rangle$ consists of

- 1 an environment $\sigma : \mathbb{X} \rightarrow \mathbb{Z}$ that maps variables to integers, and
- 2 an expression e .

$$\frac{\langle \sigma, e_1 \rangle \rightarrow \langle \sigma', e'_1 \rangle}{\langle \sigma, e_1 + e_2 \rangle \rightarrow \langle \sigma', e'_1 + e_2 \rangle}$$

$$\frac{\langle \sigma, e_1 \rangle \rightarrow \langle \sigma', e'_1 \rangle}{\langle \sigma, e_1 * e_2 \rangle \rightarrow \langle \sigma', e'_1 * e_2 \rangle}$$

$$\frac{\langle \sigma, e_2 \rangle \rightarrow \langle \sigma', e'_2 \rangle}{\langle \sigma, n_1 + e_2 \rangle \rightarrow \langle \sigma', n_1 + e'_2 \rangle}$$

$$\frac{\langle \sigma, e_2 \rangle \rightarrow \langle \sigma', e'_2 \rangle}{\langle \sigma, n_1 * e_2 \rangle \rightarrow \langle \sigma', n_1 * e'_2 \rangle}$$

$$\frac{}{\langle \sigma, n_1 + n_2 \rangle \rightarrow \langle \sigma, n_1 + n_2 \rangle}$$

$$\frac{}{\langle \sigma, n_1 * n_2 \rangle \rightarrow \langle \sigma, n_1 \times n_2 \rangle}$$

$$\frac{\langle \sigma, e_1 \rangle \rightarrow \langle \sigma', e'_1 \rangle}{\langle \sigma, x := e_1; e_2 \rangle \rightarrow \langle \sigma', x := e'_1; e_2 \rangle}$$

$$\overline{\langle \sigma, x := n; e_2 \rangle \rightarrow \langle \sigma[x \mapsto n], e_2 \rangle}$$

$$\frac{x \in \text{dom}(\sigma)}{\langle \sigma, x \rangle \rightarrow \langle \sigma, \sigma(x) \rangle}$$

We can define the **multi-step evaluation** relation \rightarrow^* as the reflexive and transitive closure of \rightarrow :

$$\overline{\langle \sigma, e \rangle \rightarrow^* \langle \sigma, e \rangle}$$

$$\frac{\langle \sigma, e \rangle \rightarrow^* \langle \sigma', e' \rangle \quad \langle \sigma', e' \rangle \rightarrow \langle \sigma'', e'' \rangle}{\langle \sigma, e \rangle \rightarrow^* \langle \sigma'', e'' \rangle}$$

$$\begin{aligned}
 \langle \emptyset, x := 1; \{y := 2; x + y\} \rangle &\rightarrow \langle \sigma_0, y := 2; x + y \rangle \\
 &\rightarrow \langle \sigma_1, x + y \rangle \\
 &\rightarrow \langle \sigma_1, 1 + y \rangle \\
 &\rightarrow \langle \sigma_1, 1 + 2 \rangle \\
 &\rightarrow \langle \sigma_1, 3 \rangle
 \end{aligned}$$

where

$$\begin{aligned}
 \sigma_0 &= [x \mapsto 1] \\
 \sigma_1 &= [x \mapsto 1, y \mapsto 2]
 \end{aligned}$$

We can also write the above derivation as a single multi-step evaluation:

$$\langle \emptyset, x := 1; \{y := 2; x + y\} \rangle \rightarrow^* \langle \sigma_1, 3 \rangle$$

It means that the expression $x := 1; \{y := 2; x + y\}$ evaluates to 3.

Are big-step and small-step operational semantics equivalent?

No! Consider the following expression:

$$x := 1; \{ x := 2; x \} + x$$

- It evaluates to 3 under the big-step operational semantics, but
- it evaluates to 4 under the small-step operational semantics.

Why?

$$\frac{\overline{\dots} \quad \overline{[x \mapsto 1] \vdash x := 2; x \Rightarrow 2} \quad \overline{[x \mapsto 1] \vdash x \Rightarrow 1}}{\overline{[x \mapsto 1] \vdash \{x := 2; x\} + x \Rightarrow 3}}$$

vs.

$$\langle [x \mapsto 1], \{x := 2; x\} + x \rangle \rightarrow \langle [x \mapsto 2], x + x \rangle \rightarrow^* \langle [x \mapsto 2], 4 \rangle$$

To make big-step and small-step operational semantics equivalent, we need to use **substitution** instead of **environments** in the small-step operational semantics:

$$\overline{\langle x := n; e_2 \rangle \rightarrow \langle e_2[x \mapsto n] \rangle}$$

where $e[x \mapsto y]$ is the expression obtained by substituting x with y in e .

With substitution, the above expression evaluates to 3 under the small-step operational semantics as well:

$$\langle x := 1; \{x := 2; x\} + x \rangle \rightarrow \langle \{x := 2; x\} + 1 \rangle \rightarrow \langle 2 + 1 \rangle \rightarrow 3$$

This demonstrates that representing **lexical scope** through substitution is often more intuitive than using environments when defining small-step operational semantics.

We will discuss substitution in more detail in the next lecture.

1. Small-Step Operational Semantics

Example: VAE

Multi-Step Evaluation

Derivation

Big-Step vs. Small-Step Operational Semantics

2. Soundness

Well-Formed Configurations

Progress

Preservation

Termination

Soundness vs. Completeness

3. Simple Imperative Language – IMP

Expressions

Statements

Let's prove the **soundness** of the small-step operational semantics of VAE.

- **Soundness:** Every configuration evaluates to an integer:

$$\forall \sigma \in \Sigma. \forall e \in \mathbb{E}. \exists \sigma' \in \Sigma. \exists n' \in \mathbb{Z}. \langle \sigma, e \rangle \rightarrow^* \langle \sigma', n' \rangle$$

Is it true? **No!** Consider the following configuration:

$$\langle \emptyset, \mathbf{x} \rangle \not\rightarrow$$

where $\langle \sigma, e \rangle \not\rightarrow$ is shorthand for:

$$\neg(\exists \sigma' \in \Sigma. \exists e' \in \mathbb{E}. \langle \sigma, e \rangle \rightarrow \langle \sigma', e' \rangle)$$

We can fix the soundness issue by restricting the evaluation to **well-formed** configurations.

Definition (Well-Formed Expressions)

A configuration $\langle \sigma, e \rangle$ is **well-formed** if and only if $\text{fvs}(e) \subseteq \text{dom}(\sigma)$

where $\text{fvs}(e)$ is the set of **free variables** in e defined as follows:

$$\begin{array}{ll} \text{fvs}(n) & \triangleq \emptyset \\ \text{fvs}(e_1 + e_2) & \triangleq \text{fvs}(e_1) \cup \text{fvs}(e_2) \\ \text{fvs}(e_1 * e_2) & \triangleq \text{fvs}(e_1) \cup \text{fvs}(e_2) \\ \text{fvs}(x := e_1; e_2) & \triangleq \text{fvs}(e_1) \cup (\text{fvs}(e_2) \setminus \{x\}) \\ \text{fvs}(x) & \triangleq \{x\} \end{array}$$

With the well-formed configurations, we can restore the soundness of the small-step operational semantics of VAE:

Theorem (Soundness)

Every well-formed configuration evaluates to an integer:

$$\begin{aligned} \forall \sigma \in \Sigma. \forall e \in \mathbb{E}. \\ fvs(e) \subseteq dom(\sigma) \implies \\ \exists \sigma' \in \Sigma. \exists n' \in \mathbb{Z}. \langle \sigma, e \rangle \rightarrow^* \langle \sigma', n' \rangle \end{aligned}$$

We will prove with the following three properties:

- **Progress:** Every well-formed configuration either is an integer or can take a step.
- **Preservation:** Every step preserves well-formedness of configurations.
- **Termination:** Every configuration terminates in a finite steps.

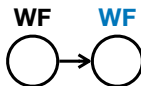
WF



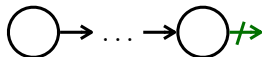
Progress

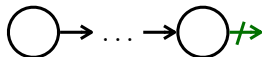
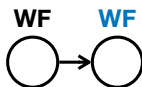
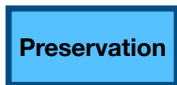
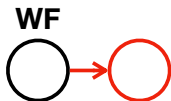


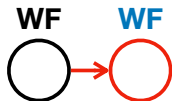
Preservation



Termination



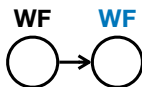




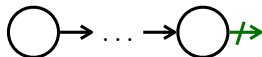
Progress

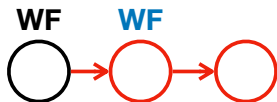


Preservation



Termination

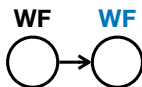




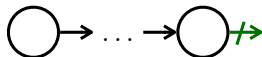
Progress

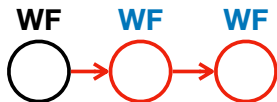


Preservation



Termination

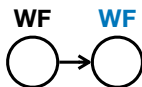




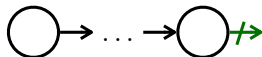
Progress



Preservation



Termination

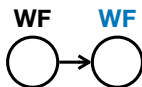




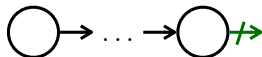
Progress



Preservation



Termination

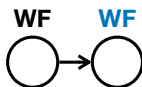




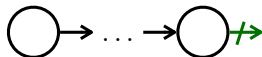
Progress



Preservation



Termination



Theorem (Progress)

Every well-formed configuration either is an integer or can take a step:

$$\forall \sigma \in \Sigma. \forall e \in \mathbb{E}. \\ fvs(e) \subseteq \text{dom}(\sigma) \implies \\ e \in \mathbb{Z} \vee (\exists \sigma' \in \Sigma. \exists e' \in \mathbb{E}. \langle \sigma, e \rangle \rightarrow \langle \sigma', e' \rangle)$$

Let's prove it by structural induction on expressions.

- $\boxed{e = n}$: e is an integer, so the progress property holds.
- $\boxed{e = x}$: Since $fvs(e) = \{x\} \subseteq \text{dom}(\sigma)$, we have $x \in \text{dom}(\sigma)$. Thus, $\langle \sigma, x \rangle \rightarrow \langle \sigma, \sigma(x) \rangle$.

$$\frac{x \in \text{dom}(\sigma)}{\langle \sigma, x \rangle \rightarrow \langle \sigma, \sigma(x) \rangle}$$

- $e = e_1 + e_2$: We know that $\langle \sigma, e_1 \rangle$ and $\langle \sigma, e_2 \rangle$ are well-formed:

$$\begin{aligned} \text{fvs}(e_1) \subseteq \text{fvs}(e_1 + e_2) \subseteq \text{dom}(\sigma) \\ \text{fvs}(e_2) \subseteq \text{fvs}(e_1 + e_2) \subseteq \text{dom}(\sigma) \end{aligned}$$

because $\text{fvs}(e_1 + e_2) \triangleq \text{fvs}(e_1) \cup \text{fvs}(e_2)$.

- $e_1 \notin \mathbb{Z}$: By the I.H., we have $\langle \sigma, e_1 \rangle \rightarrow \langle \sigma', e'_1 \rangle$ for some $\sigma' \in \Sigma$ and $e'_1 \in \mathbb{E}$. Thus, $\langle \sigma, e_1 + e_2 \rangle \rightarrow \langle \sigma', e'_1 + e_2 \rangle$.

$$\frac{\langle \sigma, e_1 \rangle \rightarrow \langle \sigma', e'_1 \rangle}{\langle \sigma, e_1 + e_2 \rangle \rightarrow \langle \sigma', e'_1 + e_2 \rangle}$$

- $e_1 \in \mathbb{Z} \wedge e_2 \notin \mathbb{Z}$: Similarly, we can use I.H. for $\langle \sigma, e_2 \rangle$.

$$\frac{\langle \sigma, e_2 \rangle \rightarrow \langle \sigma', e'_2 \rangle}{\langle \sigma, n_1 + e_2 \rangle \rightarrow \langle \sigma', n_1 + e'_2 \rangle}$$

- $e_1 \in \mathbb{Z} \wedge e_2 \in \mathbb{Z}$: Since there is no premise, it can always take a step.

$$\frac{}{\langle \sigma, n_1 + n_2 \rangle \rightarrow \langle \sigma, n_1 + n_2 \rangle}$$

- $e = e_1 * e_2$: Similar to the case of addition.
- $e = (x := e_1; e_2)$: We know that $\langle \sigma, e_1 \rangle$ is well-formed:

$$\text{fvs}(e_1) \subseteq \text{fvs}(x := e_1; e_2) \subseteq \text{dom}(\sigma)$$

because $\text{fvs}(x := e_1; e_2) \triangleq \text{fvs}(e_1) \cup (\text{fvs}(e_2) \setminus \{x\})$.

- $e_1 \notin \mathbb{Z}$: By the I.H., we have $\langle \sigma, e_1 \rangle \rightarrow \langle \sigma', e'_1 \rangle$ for some $\sigma' \in \Sigma$ and $e'_1 \in \mathbb{E}$. Thus, $\langle \sigma, x := e_1; e_2 \rangle \rightarrow \langle \sigma', x := e'_1; e_2 \rangle$.

$$\frac{\langle \sigma, e_1 \rangle \rightarrow \langle \sigma', e'_1 \rangle}{\langle \sigma, x := e_1; e_2 \rangle \rightarrow \langle \sigma', x := e'_1; e_2 \rangle}$$

- $e_1 \in \mathbb{Z}$: Since there is no premise, it can always take a step.

$$\overline{\langle \sigma, x := n; e_2 \rangle \rightarrow \langle \sigma[x \mapsto n], e_2 \rangle}$$

Theorem (Preservation)

Every step preserves well-formedness of configurations:

$$\forall \sigma, \sigma' \in \Sigma. \forall e, e' \in \mathbb{E}.$$

$$fvs(e) \subseteq dom(\sigma) \wedge \langle \sigma, e \rangle \rightarrow \langle \sigma', e' \rangle \implies fvs(e') \subseteq dom(\sigma')$$

Let's prove it by induction on the derivation of inference rules.

- $e = (x := n; e_2)$: See the following inference rule:

$$\overline{\langle \sigma, x := n; e_2 \rangle \rightarrow \langle \sigma[x \mapsto n], e_2 \rangle}$$

We can show that $fvs(e_2) \subseteq dom(\sigma[x \mapsto n])$ as follows:

$$\begin{aligned} fvs(e_2) &\subseteq fvs(e_2) \setminus \{x\} \cup \{x\} \\ &= fvs(x := n; e_2) \cup \{x\} \\ &\subseteq dom(\sigma) \cup \{x\} \\ &= dom(\sigma[x \mapsto n]) \end{aligned}$$

- $\boxed{e = e_1 + e_2}$: We know that $\langle \sigma, e_1 \rangle$ and $\langle \sigma, e_2 \rangle$ are well-formed:

$$\text{fvs}(e_1) \subseteq \text{fvs}(e_1 + e_2) \subseteq \text{dom}(\sigma)$$

$$\text{fvs}(e_2) \subseteq \text{fvs}(e_1 + e_2) \subseteq \text{dom}(\sigma)$$

because $\text{fvs}(e_1 + e_2) \triangleq \text{fvs}(e_1) \cup \text{fvs}(e_2)$.

$$\frac{\langle \sigma, e_1 \rangle \rightarrow \langle \sigma', e'_1 \rangle}{\langle \sigma, e_1 + e_2 \rangle \rightarrow \langle \sigma', e'_1 + e_2 \rangle}$$

Thus, I.H. tells us that $\text{fvs}(e'_1) \subseteq \text{dom}(\sigma')$.

We know $\text{dom}(\sigma) \subseteq \text{dom}(\sigma')$ because all small-step evaluation rules only add new bindings to environments.

Thus, $\langle \sigma', e'_1 + e_2 \rangle$ is well-formed because:

$$\text{fvs}(e'_1 + e_2) = \text{fvs}(e'_1) \cup \text{fvs}(e_2) \subseteq \text{dom}(\sigma') \cup \text{dom}(\sigma) \subseteq \text{dom}(\sigma')$$

- We can prove the rest of the cases similarly.

Theorem (Termination)

Every configuration terminates in a finite steps:

$$\forall \sigma, \sigma' \in \Sigma. \forall e, e' \in \mathbb{E}. \exists k \in \mathbb{N}. \langle \sigma, e \rangle \rightarrow^k \langle \sigma', e' \rangle \nrightarrow$$

We can prove it by showing that each small-step evaluation strictly decreases the **size** of expressions:

$$\begin{aligned} \text{size}(n) &\triangleq 0 \\ \text{size}(e_1 + e_2) &\triangleq 1 + \text{size}(e_1) + \text{size}(e_2) \\ \text{size}(e_1 * e_2) &\triangleq 1 + \text{size}(e_1) + \text{size}(e_2) \\ \text{size}(x := e_1; e_2) &\triangleq 1 + \text{size}(e_1) + \text{size}(e_2) \\ \text{size}(x) &\triangleq 1 \end{aligned}$$

Based on the induction on the derivation of inference rules, we can show that $\text{size}(e) > \text{size}(e')$ whenever $\langle \sigma, e \rangle \rightarrow \langle \sigma', e' \rangle$.

- **Soundness:** Every well-formed configuration evaluates to an integer.
- **Completeness:** Every configuration that evaluates to an integer is well-formed.

We have proved the soundness of the small-step operational semantics of VAE, but we have not proved its completeness.

Is it complete? **No!** Consider the following configuration evaluating to 2:

$$\langle \emptyset, \{x := 1; x\} + x \rangle \rightarrow \langle [x \mapsto 1], x + x \rangle \rightarrow^* \langle [x \mapsto 1], 2 \rangle$$

But, this configuration is not well-formed:

$$\text{fvs}(\langle x := 1; x \rangle + x) = \{x\} \not\subseteq \text{dom}(\emptyset) = \emptyset$$

It happens because $\text{fvs}(-)$ follows lexical scope, while the small-step operational semantics of VAE with environments does not.

1. Small-Step Operational Semantics

Example: VAE

Multi-Step Evaluation

Derivation

Big-Step vs. Small-Step Operational Semantics

2. Soundness

Well-Formed Configurations

Progress

Preservation

Termination

Soundness vs. Completeness

3. Simple Imperative Language – IMP

Expressions

Statements

Expressions $e ::= n \mid x \mid e + e \mid e * e \mid e < e \mid \text{true} \mid \text{false}$

Statements $s ::= \text{skip}$
 $\quad \mid x := e$
 $\quad \mid s; s$
 $\quad \mid \text{if } e \text{ then } s \text{ else } s$
 $\quad \mid \text{while } e \text{ do } s$

Values $v ::= n \mid \text{true} \mid \text{false}$

where $n \in \mathbb{Z}$, and $x \in \mathbb{X}$.

We will define two forms of small-step operational semantics for IMP:

$$\boxed{\langle \sigma, e \rangle \rightarrow \langle \sigma, v \rangle} \quad \boxed{\langle \sigma, s \rangle \rightarrow \langle \sigma, s \rangle}$$

where $\sigma : \mathbb{X} \rightarrow \mathbb{V}$.

$$\frac{x \in \text{dom}(\sigma)}{\langle \sigma, x \rangle \rightarrow \langle \sigma, \sigma(x) \rangle}$$

$$\frac{\langle \sigma, e_1 \rangle \rightarrow \langle \sigma, e'_1 \rangle}{\langle \sigma, e_1 + e_2 \rangle \rightarrow \langle \sigma, e'_1 + e_2 \rangle}$$

$$\frac{\langle \sigma, e_2 \rangle \rightarrow \langle \sigma, e'_2 \rangle}{\langle \sigma, n_1 + e_2 \rangle \rightarrow \langle \sigma, n_1 + e'_2 \rangle}$$

$$\frac{}{\langle \sigma, n_1 + n_2 \rangle \rightarrow \langle \sigma, n_1 + n_2 \rangle}$$

$$\frac{\langle \sigma, e_1 \rangle \rightarrow \langle \sigma, e'_1 \rangle}{\langle \sigma, e_1 * e_2 \rangle \rightarrow \langle \sigma, e'_1 * e_2 \rangle}$$

$$\frac{\langle \sigma, e_2 \rangle \rightarrow \langle \sigma, e'_2 \rangle}{\langle \sigma, n_1 * e_2 \rangle \rightarrow \langle \sigma, n_1 * e'_2 \rangle}$$

$$\frac{}{\langle \sigma, n_1 * n_2 \rangle \rightarrow \langle \sigma, n_1 \times n_2 \rangle}$$

$$\frac{\langle \sigma, e_1 \rangle \rightarrow \langle \sigma, e'_1 \rangle}{\langle \sigma, e_1 < e_2 \rangle \rightarrow \langle \sigma, e'_1 < e_2 \rangle}$$

$$\frac{\langle \sigma, e_2 \rangle \rightarrow \langle \sigma, e'_2 \rangle}{\langle \sigma, n_1 < e_2 \rangle \rightarrow \langle \sigma, n_1 < e'_2 \rangle}$$

$$\frac{}{\langle \sigma, n_1 < n_2 \rangle \rightarrow \langle \sigma, n_1 < n_2 \rangle}$$

$$\frac{\langle \sigma, e \rangle \rightarrow \langle \sigma, e' \rangle}{\langle \sigma, x := e \rangle \rightarrow \langle \sigma, x := e' \rangle} \quad \frac{}{\langle \sigma, x := v \rangle \rightarrow \langle \sigma[x \mapsto v], \text{skip} \rangle}$$

$$\frac{\langle \sigma, s_1 \rangle \rightarrow \langle \sigma, s'_1 \rangle}{\langle \sigma, s_1; s_2 \rangle \rightarrow \langle \sigma, s'_1; s_2 \rangle} \quad \frac{}{\langle \sigma, \text{skip}; s_2 \rangle \rightarrow \langle \sigma, s_2 \rangle}$$

$$\frac{\langle \sigma, e \rangle \rightarrow \langle \sigma, e' \rangle}{\langle \sigma, \text{if } e \text{ then } s_1 \text{ else } s_2 \rangle \rightarrow \langle \sigma, \text{if } e' \text{ then } s_1 \text{ else } s_2 \rangle}$$

$$\frac{}{\langle \sigma, \text{if true then } s_1 \text{ else } s_2 \rangle \rightarrow \langle \sigma, s_1 \rangle}$$

$$\frac{}{\langle \sigma, \text{if false then } s_1 \text{ else } s_2 \rangle \rightarrow \langle \sigma, s_2 \rangle}$$

$$\frac{}{\langle \sigma, \text{while } e \text{ do } s \rangle \rightarrow \langle \sigma, \text{if } e \text{ then } (s; \text{while } e \text{ do } s) \text{ else skip} \rangle}$$

- Evaluation Context and Lambda Calculus

Jihyeok Park

jihyeok_park@korea.ac.kr

<https://plrg.korea.ac.kr>