AAA616: Program Analysis

Lecture 1: Review on Operational Semantics

Hakjoo Oh

2016 Fall

Syntax of While

- Notations for syntactic categories:

n will range over numerals, **Num**,

x will range over variables, **Var**,

a will range over arithmetic expressions, **Aexp**,

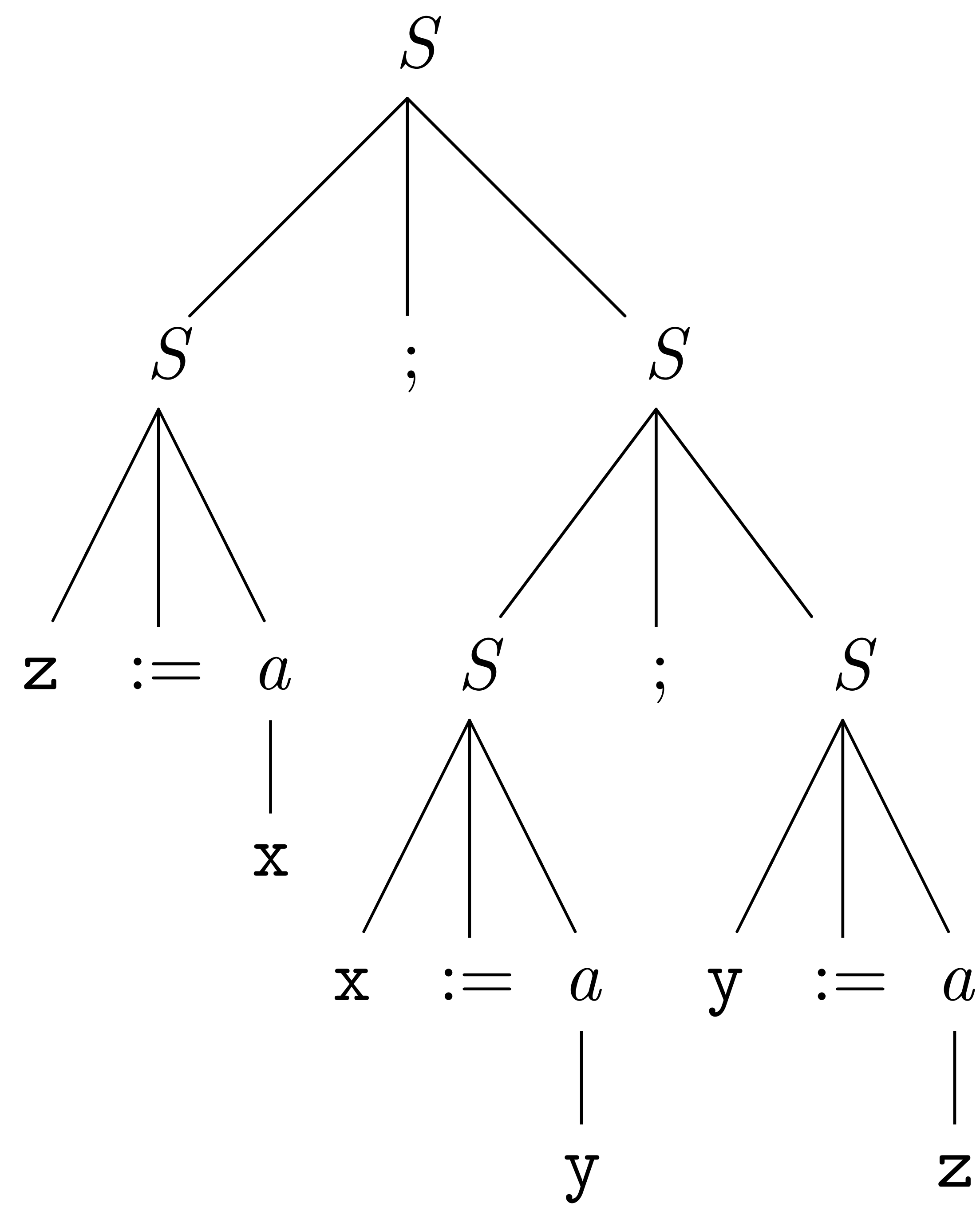
b will range over boolean expressions, **Bexp**, and

S will range over statements, **Stm**.

- Abstract syntax:

$$
\begin{array}{lcl}
a & ::= & n \mid x \mid a_1 + a_2 \mid a_1 \star a_2 \mid a_1 - a_2 \\
b & ::= & \texttt{true} \mid \texttt{false} \mid a_1 = a_2 \mid a_1 \leq a_2 \mid \neg b \mid b_1 \wedge b_2 \\
S & ::= & x := a \mid \texttt{skip} \mid S_1 \, ; \, S_2 \mid \texttt{if } b \texttt{ then } S_1 \texttt{ else } S_2 \\
  & \mid & \texttt{while } b \texttt{ do } S
\end{array}
$$

Abstract Syntax Trees



Semantics of Expressions

- The meaning of an expression depends on the state:

$$\mathbf{State} = \mathbf{Var} \rightarrow \mathbf{Z}$$

Semantics of Expressions

- The semantic function for arithmetic expressions:

$$\mathcal{A}\colon \mathbf{Aexp} \to (\mathbf{State} \to \mathbf{Z})$$

$$\begin{array}{rcl}
\mathcal{A}[\![n]\!]s & = & \mathcal{N}[\![n]\!] \\
\mathcal{A}[\![x]\!]s & = & s\ x \\
\mathcal{A}[\![a_1\ +\ a_2]\!]s & = & \mathcal{A}[\![a_1]\!]s + \mathcal{A}[\![a_2]\!]s \\
\mathcal{A}[\![a_1\ \star\ a_2]\!]s & = & \mathcal{A}[\![a_1]\!]s \cdot \mathcal{A}[\![a_2]\!]s \\
\mathcal{A}[\![a_1\ -\ a_2]\!]s & = & \mathcal{A}[\![a_1]\!]s - \mathcal{A}[\![a_2]\!]s
\end{array}$$

Semantics of Expressions

- The semantic function for boolean expressions

$$\mathcal{B}\colon \mathbf{Bexp} \to (\mathbf{State} \to \mathbf{T})$$

$$\begin{aligned}
\mathcal{B}[\![\texttt{true}]\!]s &= \mathbf{tt} \\
\mathcal{B}[\![\texttt{false}]\!]s &= \mathbf{ff} \\
\mathcal{B}[\![a_1 = a_2]\!]s &= \begin{cases} \mathbf{tt} & \text{if } \mathcal{A}[\![a_1]\!]s = \mathcal{A}[\![a_2]\!]s \\ \mathbf{ff} & \text{if } \mathcal{A}[\![a_1]\!]s \neq \mathcal{A}[\![a_2]\!]s \end{cases} \\
\mathcal{B}[\![a_1 \leq a_2]\!]s &= \begin{cases} \mathbf{tt} & \text{if } \mathcal{A}[\![a_1]\!]s \leq \mathcal{A}[\![a_2]\!]s \\ \mathbf{ff} & \text{if } \mathcal{A}[\![a_1]\!]s > \mathcal{A}[\![a_2]\!]s \end{cases} \\
\mathcal{B}[\![\neg\, b]\!]s &= \begin{cases} \mathbf{tt} & \text{if } \mathcal{B}[\![b]\!]s = \mathbf{ff} \\ \mathbf{ff} & \text{if } \mathcal{B}[\![b]\!]s = \mathbf{tt} \end{cases} \\
\mathcal{B}[\![b_1 \wedge b_2]\!]s &= \begin{cases} \mathbf{tt} & \text{if } \mathcal{B}[\![b_1]\!]s = \mathbf{tt} \text{ and } \mathcal{B}[\![b_2]\!]s = \mathbf{tt} \\ \mathbf{ff} & \text{if } \mathcal{B}[\![b_1]\!]s = \mathbf{ff} \text{ or } \mathcal{B}[\![b_2]\!]s = \mathbf{ff} \end{cases}
\end{aligned}$$

Free Variables & Substitution

- Free variables: variables occurring in expressions

$$
\begin{aligned}
\mathrm{FV}(n) &= \emptyset \\
\mathrm{FV}(x) &= \{\, x \,\} \\
\mathrm{FV}(a_1 + a_2) &= \mathrm{FV}(a_1) \cup \mathrm{FV}(a_2) \\
\mathrm{FV}(a_1 \star a_2) &= \mathrm{FV}(a_1) \cup \mathrm{FV}(a_2) \\
\mathrm{FV}(a_1 - a_2) &= \mathrm{FV}(a_1) \cup \mathrm{FV}(a_2)
\end{aligned}
$$

$$
\begin{aligned}
\mathrm{FV}(\texttt{true}) &= \emptyset \\
\mathrm{FV}(\texttt{false}) &= \emptyset \\
\mathrm{FV}(a_1 = a_2) &= \mathrm{FV}(a_1) \cup \mathrm{FV}(a_2) \\
\mathrm{FV}(a_1 \leq a_2) &= \mathrm{FV}(a_1) \cup \mathrm{FV}(a_2) \\
\mathrm{FV}(\neg b) &= \mathrm{FV}(b) \\
\mathrm{FV}(b_1 \wedge b_2) &= \mathrm{FV}(b_1) \cup \mathrm{FV}(b_2)
\end{aligned}
$$

Lemma 1.12

Let s and s' be two states satisfying that $s\ x = s'\ x$ for all x in $\mathrm{FV}(a)$. Then $\mathcal{A}[\![a]\!]s = \mathcal{A}[\![a]\!]s'$.

Free Variables & Substitution

- Substitutions: replacing each occurrence of a variable with another expression

$$
\begin{aligned}
n[y \mapsto a_0] &= n \\
x[y \mapsto a_0] &= \begin{cases} a_0 & \text{if } x = y \\ x & \text{if } x \neq y \end{cases} \\
(a_1 + a_2)[y \mapsto a_0] &= (a_1[y \mapsto a_0]) + (a_2[y \mapsto a_0]) \\
(a_1 \star a_2)[y \mapsto a_0] &= (a_1[y \mapsto a_0]) \star (a_2[y \mapsto a_0]) \\
(a_1 - a_2)[y \mapsto a_0] &= (a_1[y \mapsto a_0]) - (a_2[y \mapsto a_0])
\end{aligned}
$$

- Substitution for states:

$$
(s[y \mapsto v])\ x = \begin{cases} v & \text{if } x = y \\ s\ x & \text{if } x \neq y \end{cases}
$$

- Property of substitution:

$$
\mathcal{A}[\![a[y \mapsto a_0]]\!]s = \mathcal{A}[\![a]\!](s[y \mapsto \mathcal{A}[\![a_0]\!]s]) \text{ for all states } s.
$$

Operational Semantics

- Operational semantics is concerned about how to execute programs and not merely what the results of execution are.
- Two different approaches:
  - big-step operational semantics (natural semantics)
  - small-step operational semantics (structural operational semantics)
- In both cases, the semantics is defined by a transition system:
  - configurations
  - transition relation

Big-Step Operational Semantics

$$\langle S,\ s\rangle \rightarrow s'$$

$[\text{ass}_{\text{ns}}]$	$\langle x := a,\ s\rangle \rightarrow s[x \mapsto \mathcal{A}[\![a]\!]s]$
$[\text{skip}_{\text{ns}}]$	$\langle \texttt{skip},\ s\rangle \rightarrow s$
$[\text{comp}_{\text{ns}}]$	$\dfrac{\langle S_1,\ s\rangle \rightarrow s',\ \langle S_2,\ s'\rangle \rightarrow s''}{\langle S_1;S_2,\ s\rangle \rightarrow s''}$
$[\text{if}^{\text{tt}}_{\text{ns}}]$	$\dfrac{\langle S_1,\ s\rangle \rightarrow s'}{\langle \texttt{if}\ b\ \texttt{then}\ S_1\ \texttt{else}\ S_2,\ s\rangle \rightarrow s'}$ if $\mathcal{B}[\![b]\!]s = \mathbf{tt}$
$[\text{if}^{\text{ff}}_{\text{ns}}]$	$\dfrac{\langle S_2,\ s\rangle \rightarrow s'}{\langle \texttt{if}\ b\ \texttt{then}\ S_1\ \texttt{else}\ S_2,\ s\rangle \rightarrow s'}$ if $\mathcal{B}[\![b]\!]s = \mathbf{ff}$
$[\text{while}^{\text{tt}}_{\text{ns}}]$	$\dfrac{\langle S,\ s\rangle \rightarrow s',\ \langle \texttt{while}\ b\ \texttt{do}\ S,\ s'\rangle \rightarrow s''}{\langle \texttt{while}\ b\ \texttt{do}\ S,\ s\rangle \rightarrow s''}$ if $\mathcal{B}[\![b]\!]s = \mathbf{tt}$
$[\text{while}^{\text{ff}}_{\text{ns}}]$	$\langle \texttt{while}\ b\ \texttt{do}\ S,\ s\rangle \rightarrow s$ if $\mathcal{B}[\![b]\!]s = \mathbf{ff}$

Example

Example 2.1

Let us first consider the statement of Chapter 1:

$$(\texttt{z:=x; x:=y}); \texttt{y:=z}$$

Let s_0 be the state that maps all variables except $\texttt{x}$ and $\texttt{y}$ to $\mathbf{0}$ and has $s_0\ \texttt{x} = \mathbf{5}$ and $s_0\ \texttt{y} = \mathbf{7}$. Then an example of a derivation tree is

$$\dfrac{\dfrac{\langle \texttt{z:=x},\ s_0\rangle \to s_1 \qquad \langle \texttt{x:=y},\ s_1\rangle \to s_2}{\langle \texttt{z:=x; x:=y},\ s_0\rangle \to s_2} \qquad \langle \texttt{y:=z},\ s_2\rangle \to s_3}{\langle (\texttt{z:=x; x:=y}); \texttt{y:=z},\ s_0\rangle \to s_3}$$

where we have used the abbreviations:

$$\begin{aligned} s_1 &= s_0[\texttt{z}\mapsto\mathbf{5}] \\ s_2 &= s_1[\texttt{x}\mapsto\mathbf{7}] \\ s_3 &= s_2[\texttt{y}\mapsto\mathbf{5}] \end{aligned}$$

Properties

- The execution either terminates or loops:
  - *terminates* if and only if there is a state s' such that $\langle S, s \rangle \to s'$ and
  - *loops* if and only if there is *no* state s' such that $\langle S, s \rangle \to s'$.

- The semantics is deterministic:

$$\langle S, s \rangle \to s' \text{ and } \langle S, s \rangle \to s'' \quad \text{imply} \quad s' = s''$$

The Semantic Function

$$\mathcal{S}_{\text{ns}}\colon \mathbf{Stm} \to (\mathbf{State} \hookrightarrow \mathbf{State})$$

$$\mathcal{S}_{\text{ns}}[\![S]\!]s = \begin{cases} s' & \text{if } \langle S,\, s\rangle \to s' \\ \underline{\text{undef}} & \text{otherwise} \end{cases}$$

Small-Step Operational Semantics

$$\langle S,\ s\rangle \Rightarrow \gamma$$

$[\text{ass}_{\text{sos}}]$	$\langle x := a,\ s\rangle \Rightarrow s[x \mapsto \mathcal{A}[\![a]\!]s]$
$[\text{skip}_{\text{sos}}]$	$\langle \texttt{skip},\ s\rangle \Rightarrow s$
$[\text{comp}^1_{\text{sos}}]$	$\dfrac{\langle S_1,\ s\rangle \Rightarrow \langle S_1',\ s'\rangle}{\langle S_1;S_2,\ s\rangle \Rightarrow \langle S_1';S_2,\ s'\rangle}$
$[\text{comp}^2_{\text{sos}}]$	$\dfrac{\langle S_1,\ s\rangle \Rightarrow s'}{\langle S_1;S_2,\ s\rangle \Rightarrow \langle S_2,\ s'\rangle}$
$[\text{if}^{\text{tt}}_{\text{sos}}]$	$\langle \texttt{if}\ b\ \texttt{then}\ S_1\ \texttt{else}\ S_2,\ s\rangle \Rightarrow \langle S_1,\ s\rangle$ if $\mathcal{B}[\![b]\!]s = \mathbf{tt}$
$[\text{if}^{\text{ff}}_{\text{sos}}]$	$\langle \texttt{if}\ b\ \texttt{then}\ S_1\ \texttt{else}\ S_2,\ s\rangle \Rightarrow \langle S_2,\ s\rangle$ if $\mathcal{B}[\![b]\!]s = \mathbf{ff}$
$[\text{while}_{\text{sos}}]$	$\langle \texttt{while}\ b\ \texttt{do}\ S,\ s\rangle \Rightarrow$ $\langle \texttt{if}\ b\ \texttt{then}\ (S;\ \texttt{while}\ b\ \texttt{do}\ S)\ \texttt{else}\ \texttt{skip},\ s\rangle$

The Semantic Function

$$\mathcal{S}_{\text{sos}}\colon \mathbf{Stm} \to (\mathbf{State} \hookrightarrow \mathbf{State})$$

$$\mathcal{S}_{\text{sos}}[\![S]\!]s = \begin{cases} s' & \text{if } \langle S,\, s\rangle \Rightarrow^* s' \\ \underline{\text{undef}} & \text{otherwise} \end{cases}$$

Equivalence

Theorem 2.26

For every statement S of **While**, we have $\mathcal{S}_{\mathrm{ns}}[\![S]\!] = \mathcal{S}_{\mathrm{sos}}[\![S]\!]$.