# Lecture #7: Denotational Semantics of SIMPL

### CS 4301/6371: Advanced Programming Languages

### February 13, 2024

## 1 Syntax

| | |
|---|---|
| commands | $c ::= \texttt{skip} \mid c_1; c_2 \mid v := a \mid \texttt{if } b \texttt{ then } c_1 \texttt{ else } c_2 \mid \texttt{while } b \texttt{ do } c$ |
| boolean expressions | $b ::= \texttt{true} \mid \texttt{false} \mid a_1 \texttt{<=} a_2 \mid b_1 \texttt{ \&\& } b_2 \mid b_1 \texttt{ || } b_2 \mid !b$ |
| arithmetic expressions | $a ::= n \mid v \mid a_1 + a_2 \mid a_1 - a_2 \mid a_1 * a_2$ |
| variable names | $v$ |
| integer constants | $n$ |

## 2 Denotational Semantics

### 2.1 Semantic Domain

$$\Sigma = v \rightharpoonup \mathbb{Z}$$
$$\mathcal{A} : a \to (\Sigma \rightharpoonup \mathbb{Z})$$
$$\mathcal{B} : b \to (\Sigma \rightharpoonup \{T, F\})$$
$$\mathcal{C} : c \to (\Sigma \rightharpoonup \Sigma)$$

### 2.2 Arithmetic Expressions

Function $\mathcal{A}$ takes a SIMPL arithmetic expression as its input and returns an equivalent mathematical function from stores to integers.

$$\mathcal{A}[\![n]\!] = \{(\sigma, n) \mid \sigma \in \Sigma\}$$
$$\mathcal{A}[\![x]\!] = \{(\sigma, \sigma(x)) \mid \sigma \in \Sigma, \ x \in \sigma^{\leftarrow}\}$$
$$\mathcal{A}[\![a_1 + a_2]\!] = \{(\sigma, n_1 + n_2) \mid (\sigma, n_1) \in \mathcal{A}[\![a_1]\!], \ (\sigma, n_2) \in \mathcal{A}[\![a_2]\!]\}$$
$$\mathcal{A}[\![a_1 - a_2]\!] = \{(\sigma, n_1 - n_2) \mid (\sigma, n_1) \in \mathcal{A}[\![a_1]\!], \ (\sigma, n_2) \in \mathcal{A}[\![a_2]\!]\}$$
$$\mathcal{A}[\![a_1 * a_2]\!] = \{(\sigma, n_1 n_2) \mid (\sigma, n_1) \in \mathcal{A}[\![a_1]\!], \ (\sigma, n_2) \in \mathcal{A}[\![a_2]\!]\}$$

## 2.3   Boolean Expressions

Function $\mathcal{B}$ takes a SIMPL boolean expression as its input and returns an equivalent mathematical function from stores to truth values.

$$\mathcal{B}[\![\texttt{true}]\!] = \{(\sigma, T) \mid \sigma \in \Sigma\}$$

$$\mathcal{B}[\![\texttt{false}]\!] = \{(\sigma, F) \mid \sigma \in \Sigma\}$$

$$\mathcal{B}[\![a_1\texttt{<=}a_2]\!] = \{(\sigma, T) \mid (\sigma, n_1) \in \mathcal{A}[\![a_1]\!],\ (\sigma, n_2) \in \mathcal{A}[\![a_2]\!],\ n_1 \le n_2\} \cup$$
$$\{(\sigma, F) \mid (\sigma, n_1) \in \mathcal{A}[\![a_1]\!],\ (\sigma, n_2) \in \mathcal{A}[\![a_2]\!],\ n_1 > n_2\}$$

$$\mathcal{B}[\![b_1 \texttt{ \&\& } b_2]\!] = \{(\sigma, T) \mid (\sigma, T) \in \mathcal{B}[\![b_1]\!],\ (\sigma, T) \in \mathcal{B}[\![b_2]\!]\} \cup$$
$$\{(\sigma, F) \mid (\sigma, F) \in \mathcal{B}[\![b_1]\!],\ \sigma \in \mathcal{B}[\![b_2]\!]^{\leftarrow}\} \cup$$
$$\{(\sigma, F) \mid (\sigma, F) \in \mathcal{B}[\![b_2]\!],\ \sigma \in \mathcal{B}[\![b_1]\!]^{\leftarrow}\}$$

$$\mathcal{B}[\![b_1 \texttt{ || } b_2]\!] = \{(\sigma, T) \mid (\sigma, T) \in \mathcal{B}[\![b_1]\!],\ \sigma \in \mathcal{B}[\![b_2]\!]^{\leftarrow}\} \cup$$
$$\{(\sigma, T) \mid (\sigma, T) \in \mathcal{B}[\![b_2]\!],\ \sigma \in \mathcal{B}[\![b_1]\!]^{\leftarrow}\} \cup$$
$$\{(\sigma, F) \mid (\sigma, F) \in \mathcal{B}[\![b_1]\!],\ (\sigma, F) \in \mathcal{B}[\![b_2]\!]\}$$

$$\mathcal{B}[\![\texttt{!}b]\!] = \{(\sigma, T) \mid (\sigma, F) \in \mathcal{B}[\![b]\!]\} \cup$$
$$\{(\sigma, F) \mid (\sigma, T) \in \mathcal{B}[\![b]\!]\}$$

## 2.4   Commands

Function $\mathcal{C}$ takes a SIMPL command as input and returns an equivalent mathematical function from stores to stores.

$$\mathcal{C}[\![\texttt{skip}]\!] = \{(\sigma, \sigma) \mid \sigma \in \Sigma\} = \iota_{\Sigma \to \Sigma}$$

$$\mathcal{C}[\![c_1; c_2]\!] = \{(\sigma, \mathcal{C}[\![c_2]\!](\mathcal{C}[\![c_1]\!](\sigma))) \mid \sigma \in \Sigma\} = \mathcal{C}[\![c_2]\!] \circ \mathcal{C}[\![c_1]\!]$$

$$\mathcal{C}[\![v := a]\!] = \{(\sigma, \sigma[v \mapsto n]) \mid (\sigma, n) \in \mathcal{A}[\![a]\!]\}$$

$$\mathcal{C}[\![\texttt{if } b \texttt{ then } c_1 \texttt{ else } c_2]\!] = \{(\sigma, \mathcal{C}[\![c_1]\!](\sigma)) \mid (\sigma, T) \in \mathcal{B}[\![b]\!]\} \cup$$
$$\{(\sigma, \mathcal{C}[\![c_2]\!](\sigma)) \mid (\sigma, F) \in \mathcal{B}[\![b]\!]\}$$

$$\mathcal{C}[\![\texttt{while } b \texttt{ do } c]\!] = \bigcup_{i \ge 0} \Gamma^i(\bot_{\Sigma \to \Sigma}) = \mathit{fix}(\Gamma)$$

$$\text{where} \quad \Gamma(f) = \{(\sigma, (f \circ \mathcal{C}[\![c]\!])(\sigma)) \mid (\sigma, T) \in \mathcal{B}[\![b]\!]\} \cup$$
$$\{(\sigma, \sigma) \mid (\sigma, F) \in \mathcal{B}[\![b]\!]\}$$

Here, $\bot_{\Sigma \to \Sigma} = \{\}$. That is, $\bot_{\Sigma \to \Sigma}$ is the partial function from stores to stores that is undefined for all inputs. Function $\Gamma$ takes a partial function as input and returns a partial function as output. So $\Gamma$ has type $\Gamma : (\Sigma \rightharpoonup \Sigma) \to (\Sigma \rightharpoonup \Sigma)$. The operator $\mathit{fix}()$ denotes the *least fixed point* of its argument. That is, $\mathit{fix}(\Gamma)$ yields the smallest set $f$ such that $\Gamma(f) = f$. So the result of $\mathit{fix}(\Gamma)$ is a function of type $\Sigma \rightharpoonup \Sigma$.