ANNOUNCEMENT

- Assignment 1
  - Will be out today, due Oct 7
  - Use Latex for writing solutions
  - Academic Honesty

LAST LECTURE

- Deductive Verification using constraint solving
- Imp: A simple imperative language
- Specifying correctness

OPERATIONAL SEMANTICS OF IMP

- In order to formally define the verification problem, i.e. 'the program is verified to satisfy its specification', we will first define Operational Semantics of Imp.
- The operational semantics formally define how the program state evolves during execution.
- A program state (σ, c) consists of two components:
  - $\sigma : V \to \mathbb{R}$ is a valuation of program variables
  - c is the rest of the program to be executed
- Let $S = (\mathbb{R}^{|V|} \times \mathscr{P}) \cup \{Error\}$ be the set of all states
  - $\mathscr{P}$ is the set of all Imp programs.
- A transition $(\sigma_1, \mathrm{c}_1) \hookrightarrow (\sigma_2, \mathrm{c}_2)$ denotes a step taken by the program

TRANSITIONS OF IMP

$$\frac{\sigma_2 = \sigma_1[x \to \sigma_1(e)]}{(\sigma_1, \texttt{x := e}) \hookrightarrow (\sigma_2, \texttt{skip})}$$

TRANSITIONS OF IMP

$$\frac{\sigma_2 = \sigma_1[x \to \sigma_1(e)]}{(\sigma_1, \texttt{x := e}) \hookrightarrow (\sigma_2, \texttt{skip})}$$

NOTATION ALERT:

$f = g[a \to b]$ means:

- $f(a) = b$
- $\forall x \in dom(g)\,.\, x \neq a \to f(x) = g(x)$

TRANSITIONS OF IMP

$$\frac{\sigma_2 = \sigma_1[x \to \sigma_1(e)]}{(\sigma_1, \texttt{x := e}) \hookrightarrow (\sigma_2, \texttt{skip})}$$

NOTATION ALERT:

For $e \in Exp(V)$ and $\sigma \in \mathbb{R}^{|V|}$, $\sigma(e)$ denotes the evaluation of e at σ using the standard interpretations of Arithmetic operators.

TRANSITIONS OF IMP

$$\frac{\sigma_2 = \sigma_1[x \to \sigma_1(e)]}{(\sigma_1, \texttt{x := e}) \hookrightarrow (\sigma_2, \texttt{skip})} \text{[T-ASSIGN]}$$

TRANSITIONS OF IMP

$$\frac{\sigma_2 = \sigma_1[x \to \sigma_1(e)]}{(\sigma_1, \texttt{x := e}) \hookrightarrow (\sigma_2, \texttt{skip})} \text{[T-ASSIGN]}$$

$$\frac{\sigma_2 = \sigma_1[x \to n] \quad n \in \mathbb{R}}{(\sigma_1, \texttt{x := havoc}) \hookrightarrow (\sigma_2, \texttt{skip})} \text{[T-HAVOC]}$$

TRANSITIONS OF IMP

$$\frac{\sigma_2 = \sigma_1[x \rightarrow \sigma_1(e)]}{(\sigma_1, \texttt{x := e}) \hookrightarrow (\sigma_2, \texttt{skip})} \text{[T-ASSIGN]}$$

$$\frac{\sigma_2 = \sigma_1[x \rightarrow n] \quad n \in \mathbb{R}}{(\sigma_1, \texttt{x := havoc}) \hookrightarrow (\sigma_2, \texttt{skip})} \text{[T-HAVOC]}$$

$$\frac{???}{(\sigma_1, \texttt{assume(F)}) \hookrightarrow (\sigma_1, \texttt{skip})} \text{[T-ASSUME]}$$

TRANSITIONS OF IMP

$$\frac{\sigma_2 = \sigma_1[x \rightarrow \sigma_1(e)]}{(\sigma_1, \texttt{x := e}) \hookrightarrow (\sigma_2, \texttt{skip})} \quad \text{[T-ASSIGN]}$$

$$\frac{\sigma_2 = \sigma_1[x \rightarrow n] \quad n \in \mathbb{R}}{(\sigma_1, \texttt{x := havoc}) \hookrightarrow (\sigma_2, \texttt{skip})} \quad \text{[T-HAVOC]}$$

$$\frac{\sigma_1 \vDash F}{(\sigma_1, \texttt{assume(F)}) \hookrightarrow (\sigma_1, \texttt{skip})} \quad \text{[T-ASSUME]}$$

TRANSITIONS OF IMP

$$\frac{\sigma_2 = \sigma_1[x \to \sigma_1(e)]}{(\sigma_1, \texttt{x := e}) \hookrightarrow (\sigma_2, \texttt{skip})}$$ [T-ASSIGN]

$$\frac{\sigma_2 = \sigma_1[x \to n] \quad n \in \mathbb{R}}{(\sigma_1, \texttt{x := havoc}) \hookrightarrow (\sigma_2, \texttt{skip})}$$ [T-HAVOC]

$$\frac{\sigma_1 \vDash F}{(\sigma_1, \texttt{assume(F)}) \hookrightarrow (\sigma_1, \texttt{skip})}$$ [T-ASSUME]

$$\frac{\sigma_1 \vDash F}{(\sigma_1, \texttt{assert(F)}) \hookrightarrow (\sigma_1, \texttt{skip})}$$ [T-ASSERT-TRUE]

$$\frac{\sigma_1 \nvDash F}{(\sigma_1, \texttt{assert(F)}) \hookrightarrow (Error, \texttt{skip})}$$ [T-ASSERT-FALSE]

TRANSITIONS OF IMP

[T-SEQ-1]

$$\frac{(\sigma_1, \mathtt{c}_1) \hookrightarrow (\sigma_2, \mathtt{c}_1')}{(\sigma_1, \mathtt{c}_1; \mathtt{c}_2) \hookrightarrow (\sigma_2, \mathtt{c}_1'; \mathtt{c}_2)}$$

[T-SEQ-2]

$$\frac{}{(\sigma_1, \mathtt{skip}; \mathtt{c}_2) \hookrightarrow (\sigma_1, \mathtt{c}_2)}$$

TRANSITIONS OF IMP

[T-SEQ-1]

$$\frac{(\sigma_1, \mathtt{c}_1) \hookrightarrow (\sigma_2, \mathtt{c}_1')}{(\sigma_1, \mathtt{c}_1; \mathtt{c}_2) \hookrightarrow (\sigma_2, \mathtt{c}_1'; \mathtt{c}_2)}$$

[T-SEQ-2]

$$\frac{}{(\sigma_1, \mathtt{skip}; \mathtt{c}_2) \hookrightarrow (\sigma_1, \mathtt{c}_2)}$$

[T-IF-TRUE]

$$\frac{\sigma_1 \models F}{(\sigma_1, \mathtt{if(F)\ then\ c_1\ else\ c_2}) \hookrightarrow (\sigma_1, \mathtt{c}_1)}$$

[T-IF-FALSE]

$$\frac{\sigma_1 \not\models F}{(\sigma_1, \mathtt{if(F)\ then\ c_1\ else\ c_2}) \hookrightarrow (\sigma_1, \mathtt{c}_2)}$$

TRANSITIONS OF IMP

[T-SEQ-1]

$$\frac{(\sigma_1, \mathtt{c}_1) \hookrightarrow (\sigma_2, \mathtt{c}_1')}{(\sigma_1, \mathtt{c}_1; \mathtt{c}_2) \hookrightarrow (\sigma_2, \mathtt{c}_1'; \mathtt{c}_2)}$$

[T-SEQ-2]

$$\frac{}{(\sigma_1, \mathtt{skip}; \mathtt{c}_2) \hookrightarrow (\sigma_1, \mathtt{c}_2)}$$

[T-IF-TRUE]

$$\frac{\sigma_1 \models F}{(\sigma_1, \mathtt{if(F)\ then}\ \mathtt{c}_1\ \mathtt{else}\ \mathtt{c}_2) \hookrightarrow (\sigma_1, \mathtt{c}_1)}$$

[T-IF-FALSE]

$$\frac{\sigma_1 \not\models F}{(\sigma_1, \mathtt{if(F)\ then}\ \mathtt{c}_1\ \mathtt{else}\ \mathtt{c}_2) \hookrightarrow (\sigma_1, \mathtt{c}_2)}$$

$$\frac{\sigma_1 \models F}{(\sigma_1, \mathtt{while(F)\ do\ c}) \hookrightarrow (\sigma_1, \mathtt{c}; \mathtt{while(F)\ do\ c})}$$

[T-WHILE-TRUE]

TRANSITIONS OF IMP

[T-SEQ-1]

$$\frac{(\sigma_1, \mathtt{c}_1) \hookrightarrow (\sigma_2, \mathtt{c}_1')}{(\sigma_1, \mathtt{c}_1; \mathtt{c}_2) \hookrightarrow (\sigma_2, \mathtt{c}_1'; \mathtt{c}_2)}$$

[T-SEQ-2]

$$\frac{}{(\sigma_1, \mathtt{skip}; \mathtt{c}_2) \hookrightarrow (\sigma_1, \mathtt{c}_2)}$$

[T-IF-TRUE]

$$\frac{\sigma_1 \vDash F}{(\sigma_1, \mathtt{if(F)\ then}\ \mathtt{c}_1\ \mathtt{else}\ \mathtt{c}_2) \hookrightarrow (\sigma_1, \mathtt{c}_1)}$$

[T-IF-FALSE]

$$\frac{\sigma_1 \nvDash F}{(\sigma_1, \mathtt{if(F)\ then}\ \mathtt{c}_1\ \mathtt{else}\ \mathtt{c}_2) \hookrightarrow (\sigma_1, \mathtt{c}_2)}$$

$$\frac{\sigma_1 \vDash F}{(\sigma_1, \mathtt{while(F)\ do\ c}) \hookrightarrow (\sigma_1, \mathtt{c}; \mathtt{while(F)\ do\ c})}$$

[T-WHILE-TRUE]

$$\frac{\sigma_1 \nvDash F}{(\sigma_1, \mathtt{while(F)\ do\ c}) \hookrightarrow (\sigma_1, \mathtt{skip})}$$

[T-WHILE-FALSE]

EXAMPLE

```
assume(i = 0 ∧ n ≥ 0);
while(i < n) do
  i := i + 1;
assert(i = n);
```

$(\{\mathtt{i} \mapsto 0, \mathtt{n} \mapsto 2\}, \mathtt{assume(i=0} \wedge \mathtt{n} \geq \mathtt{0);}\ldots)$

$\hookrightarrow (\{\mathtt{i} \mapsto 0, \mathtt{n} \mapsto 2\}, \mathtt{skip;}\ldots)$ [T-SEQ-1, T-ASSUME]

$\hookrightarrow (\{\mathtt{i} \mapsto 0, \mathtt{n} \mapsto 2\}, \mathtt{while(i < n)\ do\ i:=i+1;}\ldots)$ [T-SEQ-2]

$\hookrightarrow (\{\mathtt{i} \mapsto 0, \mathtt{n} \mapsto 2\}, \mathtt{i:=i+1;while(i < n)\ do\ i:=i+1;}\ldots)$ [T-WHILE-TRUE]

$\hookrightarrow (\{\mathtt{i} \mapsto 1, \mathtt{n} \mapsto 2\}, \mathtt{while(i < n)\ do\ i:=i+1;}\ldots)$ [T-SEQ-1, T-ASSIGN, T-SEQ-2]

$\hookrightarrow (\{\mathtt{i} \mapsto 1, \mathtt{n} \mapsto 2\}, \mathtt{i:=i+1;while(i < n)\ do\ i:=i+1;}\ldots)$ [T-WHILE-TRUE]

$\hookrightarrow (\{\mathtt{i} \mapsto 2, \mathtt{n} \mapsto 2\}, \mathtt{while(i < n)\ do\ i:=i+1;}\ldots)$ [T-SEQ-1, T-ASSIGN, T-SEQ-2]

$\hookrightarrow (\{\mathtt{i} \mapsto 2, \mathtt{n} \mapsto 2\}, \mathtt{assert(i=n);})$ [T-WHILE-FALSE,T-SEQ-2]

$\hookrightarrow (\{\mathtt{i} \mapsto 2, \mathtt{n} \mapsto 2\}, \mathtt{skip;})$ [T-ASSERT-TRUE]

REACHABILITY AND VERIFICATION

- Let $T \subseteq S \times S$ be the set of transitions ($\hookrightarrow$) defined in the previous slides.
    - Is T finite?
    - Is T defined for a specific program c or for any program?
- Given a program c, a sequence of transitions $(\sigma_0, \mathsf{c}) \hookrightarrow (\sigma_1, \mathsf{c}_1)\ldots \hookrightarrow (\sigma_n, \mathsf{c}_n)$ is called an execution of c.
    - A program state σ is called reachable if there exists an execution $(\sigma_0, \mathsf{c}) \hookrightarrow \ldots \hookrightarrow (\sigma, \mathsf{c}_n)$ which ends in the state σ.
- Verification Problem: Is $(Error, \mathsf{c}')$ reachable for some c′?
    - Program c is called safe if the error state is not reachable.
    - What about the initial state?