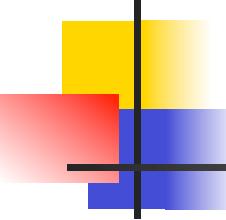


Model for Hoare Logic?

- Seen proof system for Hoare Logic
- What about models?
- Informally, triple modeled by
 - pairs of assignments of program variables to values
 - where executing program starting with initial assignment results in a memory that gives the final assignment
- Calls for alternate definition of execution



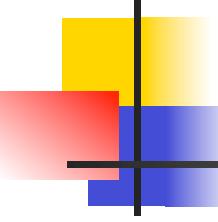
Natural Semantics

- Aka Structural Operational Semantics, aka “Big Step Semantics”
- Provide value for a program by rules and derivations, similar to type derivations
- Rule conclusions look like

$$(C, m) \Downarrow m'$$

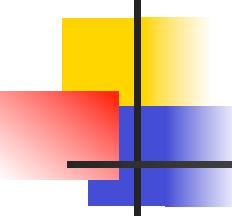
or

$$(E, m) \Downarrow v$$



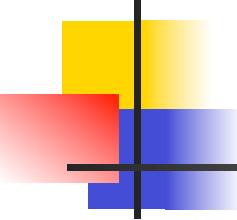
Simple Imperative Programming Language

- $I \in \text{Identifiers}$
- $N \in \text{Numerals}$
- $B ::= \text{true} \mid \text{false} \mid B \& B \mid B \text{ or } B \mid \text{not } B$
 $\mid E < E \mid E = E$
- $E ::= N \mid I \mid E + E \mid E * E \mid E - E \mid - E$
- $C ::= \text{skip} \mid C; C \mid I ::= E$
 $\mid \text{if } B \text{ then } C \text{ else } C \text{ fi} \mid \text{while } B \text{ do } C \text{ od}$



Natural Semantics of Atomic Expressions

- Identifiers: $(I,m) \Downarrow m(I)$
- Numerals are values: $(N,m) \Downarrow N$
- Booleans:
 - $(\text{true},m) \Downarrow \text{true}$
 - $(\text{false },m) \Downarrow \text{false}$



Booleans:

$$\frac{(B, m) \Downarrow \text{false}}{(B \& B', m) \Downarrow \text{false}}$$

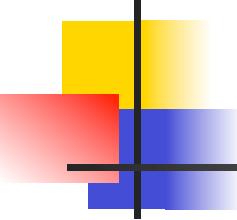
$$\frac{(B, m) \Downarrow \text{true} \quad (B', m) \Downarrow b}{(B \& B', m) \Downarrow b}$$

$$\frac{(B, m) \Downarrow \text{true}}{(B \text{ or } B', m) \Downarrow \text{true}}$$

$$\frac{(B, m) \Downarrow \text{false} \quad (B', m) \Downarrow b}{(B \text{ or } B', m) \Downarrow b}$$

$$\frac{(B, m) \Downarrow \text{true}}{(\text{not } B, m) \Downarrow \text{false}}$$

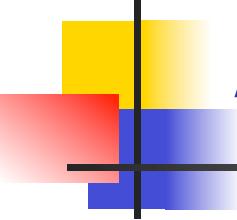
$$\frac{(B, m) \Downarrow \text{false}}{(\text{not } B, m) \Downarrow \text{true}}$$



Relations

$$\frac{(E, m) \Downarrow U \quad (E', m) \Downarrow V \quad U \sim V = b}{(E \sim E', m) \Downarrow b}$$

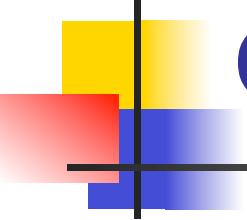
- By $U \sim V = b$, we mean does (the meaning of) the relation \sim hold on the meaning of U and V
- May be specified by a mathematical expression/equation or rules matching U and V



Arithmetic Expressions

$$\frac{(E, m) \Downarrow U \quad (E', m) \Downarrow V \quad U \text{ op } V = N}{(E \text{ op } E', m) \Downarrow N}$$

where N is the specified value for $U \text{ op } V$

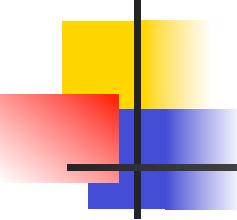


Commands

Skip: $(\text{skip}, m) \Downarrow m$

Assignment:
$$\frac{(E,m) \Downarrow V}{(I ::= E, m) \Downarrow m[I \leftarrow V]}$$

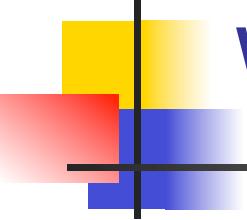
Sequencing:
$$\frac{(C,m) \Downarrow m' \quad (C',m') \Downarrow m''}{(C;C', m) \Downarrow m''}$$



If Then Else Command

$$\frac{(B,m) \Downarrow \text{true} \quad (C,m) \Downarrow m'}{(\text{if } B \text{ then } C \text{ else } C' \text{ fi, } m) \Downarrow m'}$$

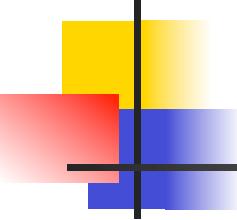
$$\frac{(B,m) \Downarrow \text{false} \quad (C',m) \Downarrow m'}{(\text{if } B \text{ then } C \text{ else } C' \text{ fi, } m) \Downarrow m'}$$



While Command

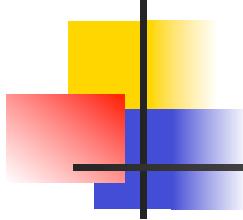
$$\frac{(B,m) \Downarrow \text{false}}{(\text{while } B \text{ do } C \text{ od}, m) \Downarrow m}$$

$$\frac{(B,m) \Downarrow \text{true} \ (C,m) \Downarrow m' \ (\text{while } B \text{ do } C \text{ od}, m') \Downarrow m',}{(\text{while } B \text{ do } C \text{ od}, m) \Downarrow m''}$$



Example: If Then Else Rule

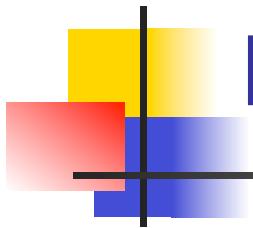
$$(\text{if } x > 5 \text{ then } y := 2 + 3 \text{ else } y := 3 + 4 \text{ fi,} \\ \{x \rightarrow 7\}) \Downarrow ?$$



Example: If Then Else Rule

$$(x > 5, \{x \rightarrow 7\}) \Downarrow ?$$

$$\begin{aligned} & (\text{if } x > 5 \text{ then } y := 2 + 3 \text{ else } y := 3 + 4 \text{ fi,} \\ & \quad \{x \rightarrow 7\}) \Downarrow ? \end{aligned}$$



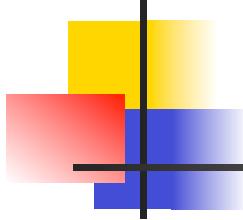
Example: Arith Relation

? > ? = ?

$$\frac{(x, \{x > 7\}) \Downarrow ? \quad (5, \{x > 7\}) \Downarrow ?}{(x > 5, \{x > 7\}) \Downarrow ?}$$

(if $x > 5$ then $y := 2 + 3$ else $y := 3 + 4$ fi,

$\{x > 7\}) \Downarrow ?$



Example: Identifier(s)

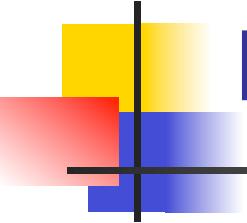
$7 > 5 = \text{true}$

$(x, \{x > 7\}) \Downarrow 7 \quad (5, \{x > 7\}) \Downarrow 5$

$(x > 5, \{x > 7\}) \Downarrow ?$

$(\text{if } x > 5 \text{ then } y := 2 + 3 \text{ else } y := 3 + 4 \text{ fi,}$

$\{x > 7\}) \Downarrow ?$



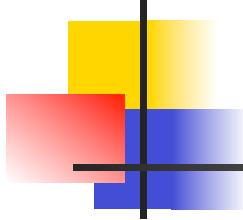
Example: Arith Relation

$$7 > 5 = \text{true}$$

$$\frac{(x, \{x > 7\}) \Downarrow 7 \quad (5, \{x > 7\}) \Downarrow 5}{(x > 5, \{x > 7\}) \Downarrow \text{true}}$$

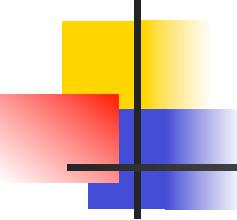
$$\frac{}{\text{(if } x > 5 \text{ then } y := 2 + 3 \text{ else } y := 3 + 4 \text{ fi,}}$$

$$\{x > 7\} \Downarrow ?$$



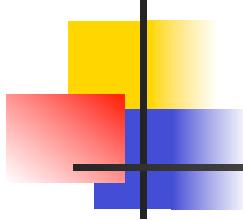
Example: If Then Else Rule

$$\frac{\begin{array}{c} 7 > 5 = \text{true} \\[10pt] \underline{(x, \{x > 7\}) \Downarrow 7 \quad (5, \{x > 7\}) \Downarrow 5} \\[10pt] \underline{(x > 5, \{x -> 7\}) \Downarrow \text{true}} \end{array}}{\begin{array}{c} \overline{(y := 2 + 3, \{x > 7\})} \\[10pt] \Downarrow ? \end{array}} .$$
$$\frac{(\text{if } x > 5 \text{ then } y := 2 + 3 \text{ else } y := 3 + 4 \text{ fi, } \{x -> 7\}) \Downarrow ?}{}$$



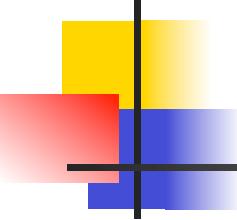
Example: Assignment

$$\frac{\frac{7 > 5 = \text{true} \quad \underline{(x, \{x > 7\}) \Downarrow 7 \quad (5, \{x > 7\}) \Downarrow 5}}{(x > 5, \{x > 7\}) \Downarrow \text{true}} \quad \frac{(2+3, \{x > 7\}) \Downarrow ?}{(y := 2 + 3, \{x > 7\})} \quad \Downarrow ?}{(\text{if } x > 5 \text{ then } y := 2 + 3 \text{ else } y := 3 + 4 \text{ fi, } \{x > 7\}) \Downarrow ?}$$



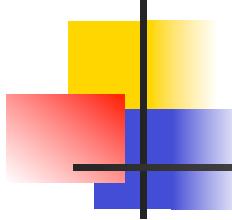
Example: Arith Op

$$\begin{array}{c} ? + ? = ? \\ \hline (2, \{x > 7\}) \Downarrow ? \quad (3, \{x > 7\}) \Downarrow ? \\ \hline 7 > 5 = \text{true} \qquad \qquad \qquad \frac{(2+3, \{x > 7\}) \Downarrow ?}{(y := 2 + 3, \{x > 7\})} \\ \hline \underline{(x, \{x > 7\}) \Downarrow 7} \quad \underline{(5, \{x > 7\}) \Downarrow 5} \qquad \qquad \qquad (y := 2 + 3, \{x > 7\}) \\ \hline \underline{(x > 5, \{x > 7\}) \Downarrow \text{true}} \qquad \qquad \qquad \Downarrow ? \qquad \qquad \cdot \\ \hline (\text{if } x > 5 \text{ then } y := 2 + 3 \text{ else } y := 3 + 4 \text{ fi,} \\ \qquad \qquad \qquad \{x > 7\}) \Downarrow ? \end{array}$$



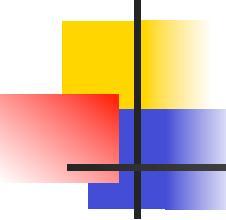
Example: Numerals

$$\frac{\frac{\frac{2 + 3 = 5}{(2, \{x > 7\}) \Downarrow 2 \quad (3, \{x > 7\}) \Downarrow 3}}{\frac{7 > 5 = \text{true}}{(x, \{x > 7\}) \Downarrow 7 \quad (5, \{x > 7\}) \Downarrow 5}} \quad \frac{(2+3, \{x > 7\}) \Downarrow ?}{(y := 2 + 3, \{x > 7\})}}{\frac{(x > 5, \{x > 7\}) \Downarrow \text{true}}{\frac{}{(\text{if } x > 5 \text{ then } y := 2 + 3 \text{ else } y := 3 + 4 \text{ fi}, \\ \{x > 7\}) \Downarrow ?}}}}$$



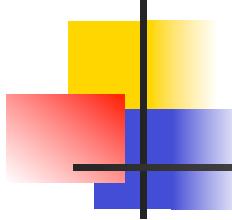
Example: Arith Op

$$\frac{\frac{2 + 3 = 5}{(2, \{x > 7\}) \Downarrow 2 \quad (3, \{x > 7\}) \Downarrow 3} \quad 7 > 5 = \text{true}}{\frac{(2+3, \{x > 7\}) \Downarrow 5}{(y := 2 + 3, \{x > 7\})}} \quad \frac{\underline{(x, \{x > 7\}) \Downarrow 7 \quad (5, \{x > 7\}) \Downarrow 5}}{(x > 5, \{x > 7\}) \Downarrow \text{true}} \quad \Downarrow ?}$$
$$\frac{(if \ x > 5 \text{ then } y := 2 + 3 \text{ else } y := 3 + 4 \text{ fi}, \{x > 7\}) \Downarrow ?}{}$$



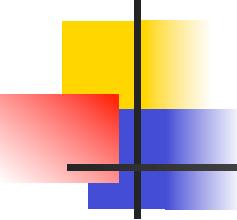
Example: Assignment

$$\frac{\frac{\frac{2 + 3 = 5}{(2, \{x > 7\}) \Downarrow 2 \quad (3, \{x > 7\}) \Downarrow 3}}{\frac{7 > 5 = \text{true}}{(x, \{x > 7\}) \Downarrow 7 \quad (5, \{x > 7\}) \Downarrow 5}} \quad \frac{(2+3, \{x > 7\}) \Downarrow 5}{(y := 2 + 3, \{x > 7\)}}}{\frac{(x > 5, \{x > 7\}) \Downarrow \text{true}}{(y := 2 + 3, \{x > 7\}) \Downarrow \{x > 7, y > 5\}}} \quad \frac{}{(y := 2 + 3, \{x > 7\}) \Downarrow \{x > 7, y > 5\}}}$$
$$\frac{(if \ x > 5 \ then \ y := 2 + 3 \ else \ y := 3 + 4 \ fi, \{x > 7\}) \Downarrow ?}{(y := 2 + 3, \{x > 7\}) \Downarrow \{x > 7, y > 5\}}$$



Example: If Then Else Rule

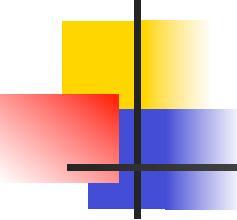
$$\frac{\frac{2 + 3 = 5}{(2, \{x > 7\}) \Downarrow 2 \quad (3, \{x > 7\}) \Downarrow 3} \quad 7 > 5 = \text{true}}{\frac{(2+3, \{x > 7\}) \Downarrow 5}{(y := 2 + 3, \{x > 7\}) \Downarrow \{x > 7, y > 5\}}} \quad \frac{(x, \{x > 7\}) \Downarrow 7 \quad (5, \{x > 7\}) \Downarrow 5}{(x > 5, \{x > 7\}) \Downarrow \text{true}}$$
$$\frac{(if \ x > 5 \ then \ y := 2 + 3 \ else \ y := 3 + 4 \ fi, \{x > 7\}) \Downarrow \{x > 7, y > 5\}}{}$$



Let in Command

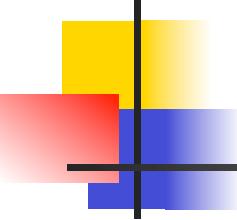
$$\frac{(E, m) \Downarrow v \quad (C, m[I \leftarrow v]) \Downarrow m'}{(\text{let } I = E \text{ in } C, m) \Downarrow m''}$$

Where $m''(y) = m'(y)$ for $y \neq I$ and
 $m''(I) = m(I)$ if $m(I)$ is defined,
and $m''(I)$ is undefined otherwise



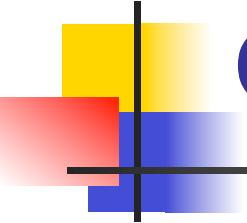
Example

$$\frac{\begin{array}{c} (x, \{x > 5\}) \Downarrow 5 \quad (3, \{x > 5\}) \Downarrow 3 \\ \hline (x+3, \{x > 5\}) \Downarrow 8 \end{array}}{(5, \{x > 17\}) \Downarrow 5 \quad (x := x + 3, \{x > 5\}) \Downarrow \{x > 8\}}$$
$$\frac{(5, \{x > 17\}) \Downarrow 5 \quad (x := x + 3, \{x > 5\}) \Downarrow \{x > 8\}}{(\text{let } x = 5 \text{ in } (x := x + 3), \{x > 17\}) \Downarrow ?}$$



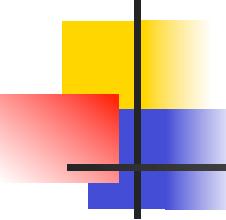
Example

$$\frac{\begin{array}{c} (x,\{x>5\}) \Downarrow 5 \quad (3,\{x>5\}) \Downarrow 3 \\ \hline (x+3,\{x>5\}) \Downarrow 8 \\ \hline (5,\{x>17\}) \Downarrow 5 \quad (x:=x+3,\{x>5\}) \Downarrow \{x>8\} \end{array}}{(\text{let } x = 5 \text{ in } (x:=x+3), \{x > 17\}) \Downarrow \{x>17\}}$$



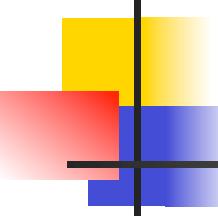
Comment

- Simple Imperative Programming Language introduces variables *implicitly* through assignment
- The let-in command introduces scoped variables *explicitly*
- Clash of constructs apparent in awkward semantics



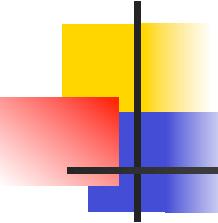
Interpretation Versus Compilation

- A **compiler** from language L1 to language L2 is a program that takes an L1 program and for each piece of code in L1 generates a piece of code in L2 of same meaning
- An **interpreter** of L1 in L2 is an L2 program that executes the meaning of a given L1 program
- Compiler would examine the body of a loop once; an interpreter would examine it every time the loop was executed



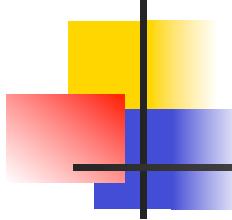
Interpreter

- An *Interpreter* represents the operational semantics of a language L1 (source language) in the language of implementation L2 (target language)
- Built incrementally
 - Start with literals
 - Variables
 - Primitive operations
 - Evaluation of expressions
 - Evaluation of commands/declarations



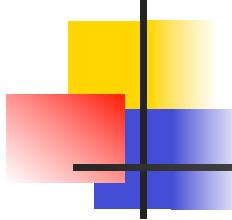
Interpreter

- Takes abstract syntax trees as input
 - In simple cases could be just strings
- One procedure for each syntactic category (nonterminal)
 - eg one for expressions, another for commands
- If Natural semantics used, tells how to compute final value from code
- If Transition semantics used, tells how to compute next “state”
 - To get final value, put in a loop



Natural Semantics Example

- $\text{compute_exp}(\text{Var}(v), m) = \text{look_up } v \text{ in } m$
- $\text{compute_exp}(\text{Int}(n), _) = \text{Num}(n)$
- ...
- $\text{compute_com}(\text{IfExp}(b, c_1, c_2), m) =$
 if $\text{compute_exp}(b, m) = \text{Bool}(\text{true})$
 then $\text{compute_com}(c_1, m)$
 else $\text{compute_com}(c_2, m)$



Natural Semantics Example

- $\text{compute_com}(\text{While}(b,c), m) =$
 if $\text{compute_exp}(b,m) = \text{Bool}(\text{false})$
 then m
 else $\text{compute_com}(\text{While}(b,c), \text{compute_com}(c,m))$

- May fail to terminate - exceed stack limits
- Returns no useful information then