
HW 7 – Operational Semantics

CS 421 – Summer 2013

Revision 1.0

Assigned July 10, 2013

Due July 16, 2013, 11:59 PM

Extension None

1 Change Log

1.0 Initial Release.

2 Turn-In Procedure

Your answers to the following questions are to be submitted electronically via the `handin` script as though an MP.

3 Objectives and Background

The purpose of this HW is to test your understanding of

- The difference between natural semantics and structural operational semantics.
- How to create rules for natural semantics.
- How to create rules for structural operational semantics.

Another purpose of HWs is to provide you with experience answering non-programming written questions of the kind you may experience on the midterms and final.

4 Problem

(20 pts) Recall the syntax of the Simple Imperative Programming Language discussed in class (with parentheses added for clarity):

$I \in \text{Identifiers}$

$N \in \text{Numerals}$

$B ::= \text{true} \mid \text{false} \mid B \ \& \ B \mid B \ \text{or} \ B \mid \text{not } B \mid E < E \mid E = E$

$E ::= N \mid I \mid E + E \mid E * E \mid E - E \mid - E$

$C ::= \text{skip} \mid C; C \mid I := E \mid \text{if } B \text{ then } C \text{ else } C \text{ fi} \mid \text{while } B \text{ do } C \text{ od} \mid (C)$

Suppose we want to add for-loops to this language. We've added appropriate syntax to the grammar:

$C ::= \dots \mid \text{for } I \text{ from } E \text{ to } E \text{ do } C \text{ od}$

The intended semantics of `for I from E to E' do C od` is as follows:

1. Evaluate E and assign its value to I in the memory.
2. Check whether I is equal to E' ; if so, return the current memory, and if not, proceed to the next step.
3. Execute C , possibly modifying the memory.
4. Add 1 to the value of I in memory.
5. Return to step 2.

Now, it's your turn: give semantic rules to implement the for-loop.

Hint: it might make your life easier if you can figure out a way to combine step 4 and step 1.

1. (10 points) Give natural (big-step) semantics rules for the for-loop.
2. (10 points) Give structural operational (small-step) semantics rules for the for-loop. As with the while-loop, you will most likely need to define it partially in terms of other constructs.
Hint: Think about which parts you can evaluate at the outset, and which parts need to be re-evaluated each time the loop is executed.

Answer:

Keep in mind that there are many different possible solutions.

1.

$$\frac{(E, m) \Downarrow v \quad (I = E', m[I \leftarrow v]) \Downarrow \text{true}}{(\text{for } I \text{ from } E \text{ to } E' \text{ do } C \text{ od}, m) \Downarrow m[I \leftarrow v]}$$

$$\frac{(E, m) \Downarrow v \quad (I = E', m[I \leftarrow v]) \Downarrow \text{false} \quad (C, m[I \leftarrow v]) \Downarrow m' \quad (\text{for } I \text{ from } I+1 \text{ to } E' \text{ do } C \text{ od}, m') \Downarrow m''}{(\text{for } I \text{ from } E \text{ to } E' \text{ do } C \text{ od}, m) \Downarrow m''}$$

2.

$$\frac{(E, m) \rightarrow (E'', m)}{(\text{for } I \text{ from } E \text{ to } E' \text{ do } C \text{ od}, m) \rightarrow (\text{for } I \text{ from } E'' \text{ to } E' \text{ do } C \text{ od}, m)}$$

$$(\text{for } I \text{ from } v \text{ to } E' \text{ do } C \text{ od}, m) \rightarrow (\text{if } I = E' \text{ then skip else } (C; \text{for } I \text{ from } I+1 \text{ to } E' \text{ do } C \text{ od}), m[I \leftarrow v])$$