University of Massachusetts Lowell — Comp 3010: Organization of Programming Languages

HW 3

Name: Rowan Sharwood
UML ID: 02090573
Collaborators: None

Make sure that the remaining pages of this assignment do not contain any identifying information.

$$\frac{\langle a, \sigma\rangle \longrightarrow \langle a', \sigma\rangle}{\langle x := a, \sigma\rangle \longrightarrow \langle x := a', \sigma\rangle} \qquad \frac{}{\langle x := n, \sigma\rangle \longrightarrow \langle \textsf{skip}, \sigma[x \mapsto n]\rangle}$$

$$\frac{\langle c_1, \sigma\rangle \longrightarrow \langle c_1', \sigma'\rangle}{\langle c_1; c_2, \sigma\rangle \longrightarrow \langle c_1'; c_2, \sigma'\rangle} \qquad \frac{}{\langle \textsf{skip}; c_2, \sigma\rangle \longrightarrow \langle c_2, \sigma\rangle}$$

$$\frac{\langle b, \sigma\rangle \longrightarrow \langle b', \sigma\rangle}{\langle \textsf{if } b \textsf{ then } c_1 \textsf{ else } c_2, \sigma\rangle \longrightarrow \langle \textsf{if } b' \textsf{ then } c_1 \textsf{ else } c_2, \sigma\rangle}$$

$$\frac{}{\langle \textsf{if true then } c_1 \textsf{ else } c_2, \sigma\rangle \longrightarrow \langle c_1, \sigma\rangle} \qquad \frac{}{\langle \textsf{if false then } c_1 \textsf{ else } c_2, \sigma\rangle \longrightarrow \langle c_2, \sigma\rangle}$$

$$\frac{}{\langle \textsf{while } b \textsf{ do } c, \sigma\rangle \longrightarrow \langle \textsf{if } b \textsf{ then } (c; \textsf{while } b \textsf{ do } c) \textsf{ else skip}, \sigma\rangle}$$

1 Small-Step Semantics (40 points)

(a) Give the inference rules for division

$$\frac{\langle a_1, \sigma\rangle \longrightarrow \langle a_1', \sigma\rangle}{\langle a_1/a_2, \sigma\rangle \longrightarrow \langle a_1'/a_2, \sigma\rangle}$$

$$\frac{\langle a_2, \sigma\rangle \longrightarrow \langle a_2', \sigma\rangle}{\langle n/a_2, \sigma\rangle \longrightarrow \langle n/a_2', \sigma\rangle}$$

$$\frac{}{\langle n/m, \sigma\rangle \longrightarrow \langle p, \sigma\rangle}\; m \neq 0 \quad p = n/m$$

(b) Give the inference rules for do-while

$$\frac{}{\langle \textbf{do } c \textbf{ while } b, \sigma\rangle \longrightarrow \langle c; \textbf{if } b \textbf{ then do } c \textbf{ while } b \textbf{ else skip}, \sigma\rangle}$$

2 Implementation in OCaml (5 points)

(a) (Answer in OCaml.)

(b) Try running the large-step interpreter you wrote in HW2 (the "eval_c" function) on some example programs, and then run the small-step interpreter you just wrote (the "multistep_c" function) on the same files. What can you say about the behavior of the two interpreters in relation to each other?

When I ran the same example programs using both the large-step interpreter `eval_c` and the small-step interpreter `multistep_c` , they both produced the same final state for all the programs that terminate. The variable values at the end were identical, which shows that both interpreters are consistent with each other in terms of the final result. The main difference is that the large-step interpreter evaluates a command directly to its final state in one recursive evaluation and just computes the result. The small-step interpreter, on the other hand, executes the program one step at a time. It models the actual execution process more explicitly, moving through assignments, conditionals, and loops piece by piece. Both interpreters have similar functions for terminating programs, but the small-step interpreter makes the execution process more detailed and visible, while the large-step interpreter jumps straight to the final answer.