Homework 3

Md Ali A20439433
CS 536: Science of Programming

February 10, 2021

Exercise 1. Translate the program below into our programming language
$p = 1; k = x = 0;\ while(k++ < n)\{++x; p* = x;\}$

Proof. $p := 1$
$k := 0$
$x := 0$
$while\, k < n$
do
$x := x + 1$
$p := p * x$
$k := k + 1$
od

□

Exercise 2. Let $S \equiv\ if\ x > 0\ then\ x := x * z\ else\ if\ y > 0\ then\ y := y * z\ fi\ fi$

a. Evaluate $\langle S, \{x = 2, y = 6, z = 4\}\rangle$ to completion, using step-by-step operational semantics.

b. Evaluate $\langle S, \{x = -2, y = 8, z = 5\}\rangle$ to completion, using step-by-step operational semantics.

c. Evaluate $\langle S, \{x = -1, y = -2, z = 6\}\rangle$ to completion, using step-by-step operational semantics.

Proof. Below are all three parts

a. Taking the program S executing in a state with $x = 2$, $y = 6$, and $z = 4$, we get the following.

Since $x > 0$, we following $x = 2 \cdot 4$, where $x = 8$, so we get that $(x := x * z, \{x = 2, y = 6, z = 4\}) \rightarrow (E, \{x = 8, y = 6, z = 4\})$

b. Taking the program S executing in a state with $x = -2$, $y = 8$, and $z = 5$, so we get the following.

Since $x < 0$ which is false, we have to look at the second condition and that come to $y > 0$, so we ge tthat $y = 8 \cdot 5 = 40$, so we get that $(y := y * z, \{x = -2, y = 8, z = 5\}) \rightarrow (E, \{x = -2, y = 40, z = 5\})$

c. In this specific case, we can see that none of the condition are met since $x < 0$ and $y < 0$ in the state where $x = -1$, $y = -2$, and $z = 6$ so we get the expression below.

$(SKIP, \{x = -1, y = -2, z = 6\}) \rightarrow (E, \{x = -1, y = -2, z = 6\})$

□

Exercise 3. Let $W \equiv$ *while* $k \neq n$ *do* S *od* where $S \equiv k := k + 1; x := x + k * k$. Let $\sigma_0 = \{k = 0, x = 1, n = 4\}$. Evaluate $\langle W, \sigma_0 \rangle$ to completion. Show all configuration of the form $\langle W, state \rangle$ and the final $\langle E, state \rangle$. You can use $\rightarrow^n$ to skip other configurations if you like, or you can show them.

Proof. We will take the given of $W \equiv$ *while* $k \neq n$ *do* S *od* where $S \equiv k := k + 1; x := x + k * k$. Let $\sigma_0 = \{k = 0, x = 1, n = 4\}$. Now we will evaluate this to the end of completion as shown below.

$\langle W, \sigma_0 \rangle = (while\ k \neq n\ do\ S\ od)$
$\rightarrow \langle S; W, \sigma_0[k \mapsto 1][x \mapsto 2] \rangle$
$\rightarrow \langle S; W, \sigma_1 \rangle$
$\rightarrow \langle S; W, \sigma_1[k \mapsto 2][x \mapsto 6] \rangle$
$\rightarrow \langle S; W, \sigma_2 \rangle$
$\rightarrow \langle S; W, \sigma_2[k \mapsto 3][x \mapsto 15] \rangle$
$\rightarrow \langle S; W, \sigma_3 \rangle$
$\rightarrow \langle S; W, \sigma_3[k \mapsto 4][x \mapsto 31] \rangle$
$\rightarrow \langle S; W, \sigma_4 \rangle$, where $\sigma_4 = \{k = 4, x = 31, n = 4\}$

□

Exercise 4. Give the denotational semantics ($M(S, ...) =?$) of the configurations in problem 2a-2c.

Proof. We are given $S \equiv$ *if* $x > 0$ *then* $x := x * z$ *else if* $y > 0$ *then* $y := y * z$ *fi fi*. Looking at each case below.

a. $\langle S, \{x = 2, y = 6, z = 4\} \rangle$

Taking $M(S, \sigma) = M(S, \{x = 2, y = 6, z = 4\}$
Then we get $M(x := x * z, \{x = 2, y = 6, z = 4\})$ since $\sigma(x > 0) = T$
In conclusion, $M(S, \sigma) = M(S, \{x = 8, y = 6, z = 4\})$

b. $\langle S, \{x = -2, y = 8, z = 5\}\rangle$

Taking $M(S, \sigma) = M(S, \{x = -2, y = 8, z = 5\}$
Then we get $M(y := y * z, \{x = -2, y = 8, z = 5\})$ since $\sigma(x > 0) = F$ but $\sigma(y > 0) = T$
In conclusion, $M(S, \sigma) = M(S, \{x = -2, y = 40, z = 5\})$

c. $\langle S, \{x = -1, y = -2, z = 6\}\rangle$

Taking $M(S, \sigma) = M(S, \{x = -1, y = -2, z = 6\}$
Then we get $M(SKIP, \{x = -1, y = -2, z = 6\})$ since $\sigma(x > 0) = F$ and $\sigma(y > 0) = F$
In conclusion, $M(S, \sigma) = M(S, \{x = -1, y = -2, z = 6\})$

□

Exercise 5. Take the W from problem 3. What is the set of σ such that $\langle W, \sigma\rangle \to^* \langle E, \perp\rangle$?

Proof. We can take the state $\sigma = \{k = 0, x = 1, n = -1\}$, this will lead the loop to diverge because k will increase in each iteration of the loop and is already greater than n, in other words it will never be equal to n evne after an infinite amount of loops so

$\sigma_k \models k \neq n$, we get $M(W, \sigma) = \{\perp_d\}$

□

Exercise 6. Let $S \equiv x := b[m-2]/sqrt(k)$ and let $\sigma = \{m = \alpha, k = \gamma, b = \beta\}$. Let δ be the length of b, so $\beta(0), ..., \beta(\delta - 1)$ are the values of $b[0], b[1], ...$ Describe the set of all σ that cause $M(S, \sigma) = \{\perp_e\}$. (As in class, divide by zero and square root of a negative number cause errors.)

Proof. With the execution of the program, we can get the result of a successful termination, divergence of the loop, or a run time error. With a successful termination, this means that we were able to run the program successfully. In the terms of the divergence of the loop, this is most likely to occur if the loop is already inside and the value of the variable in the loop is within the body with no particular way of satisfying the condition, hence causing an infinite loop, e.g. divergence of the loop. Lastly, the run time error in this specific case can be achieved by having the square root of a negative integer or an index being out of range or having the error of divisibility of zero.

We are given $S \equiv x := b[m-2]/sqrt(k)$, $\sigma = \{m = \alpha, k = \gamma, b = \beta\}$, and $\delta = size(b)$.

We will take all the cases that can set values of σ that causes a run time error such as $M(S, \sigma) = \{\perp_e\}$.

Case 1: Let's take the value of k in σ to be less than zero, such as $k < 0$.
If the value of k is less than or equal to zero, then $M(S, \sigma) = \{\perp_e\}$, due to the fact that the square root of the negative values causes the run time error since the square root of a negative value will be in the set of $\in \mathbb{I}$.

Case 2: Take that the value of k is equal to zero. Hence $k = 0$, making the $sqrt(0) = 0$, which cause the division by zero, which might cause a run time error or a possible unanticipated termination. In either case, in mathematics the division by zero leads to an undefined value.

Case 3: Take the value of $(m - 2)$ that is not in the range of index of b, in other words $(m - 2) < 0$ and $(m - 2) \geq \delta$, where δ is $size(b)$. So take that $(m - 2) < 0$, making $m < 2$, the value at the index of b will be the negative integer, which cause the index to be out of range. Likewise, if we take that $(m - 2) \geq \delta$, meaning that $m \geq \delta + 2$, making the index value greater or equal to the size of the array which will also result in a value of an index to be out of range.

Taking the set of all $\sigma = \{m = \alpha, k = \gamma, b = \beta\}$ that causes a run time error of $M(S, \sigma) = \{\perp_e\}$ are the following

1. $M(S, \sigma) = \{\perp_e\}$, if $k < 0$, due to the fact that $sqrt(k)$ is invalid if value of k is negative.

2. $M(S, \sigma) = \{\perp_e\}$, if $k = 0$, due to the fact that $sqrt(0) = 0$, hence this will cause an error cause by the division of zero.

3. $M(S, \sigma) = \{\perp_e\}$, if $m < 2$ then the index of the array will be out of bounds and causing a run time error.

4. $M(S, \sigma) = \{\perp_e\}$, if $m \geq \delta + 2$, where $\delta = size(b)$ making it where the index of the array will be out of bounds causing a run time error.

To summarize the statements above we can say that for $\sigma = \{m = \alpha, k = \gamma, b = \beta\}$ such that $M(S, \sigma) = \{\perp_e\}$ will be $\{\sigma \in \Sigma | \sigma(m) < 2 \cup \sigma(m) \geq \sigma(\delta) + 2 \cup \sigma(k) \leq 0\}$, where Σ is the set of all states.

$\square$