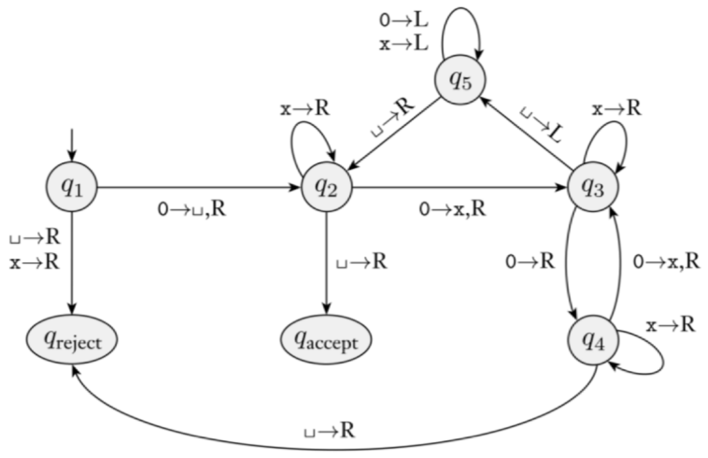
Practice Problem Set 1

Fall 25

1. Turing Machine (attributed to the Sipser book Exercise 3.1)

   Recall the TM example for recognizing powers of 2 we discussed in class:



   Give the sequence of configurations that the TM enters when started on the input string 00.

Solution:

$q_1 00 \Rightarrow \sqcup q_2 0 \Rightarrow \sqcup x q_3 \sqcup \Rightarrow \sqcup q_5 x \sqcup \Rightarrow q_5 \sqcup x \sqcup \Rightarrow \sqcup q_2 x \sqcup \Rightarrow \sqcup x q_2 \sqcup \Rightarrow \sqcup x \sqcup q_{accept}$

2. Turing Machine (attributed to the Sipser book Problem 3.10)

   Say that a *write-once Turing machine* is a single-tape TM that can alter each tape cell at most once (including the input portion of the tape). Show that this variant Turing machine model is equivalent to the ordinary Turing machine model.

   Hint: As a first step consider the case whereby the Turing machine may alter each tape cell at most twice. Use lots of tape.

Solution:

Proof: We first simulate an ordinary Turing machine by a write-twice Turing machine. The write-twice machine simulates a single step of the original machine by copying the entire tape over to a fresh portion of the tape to the right-hand side of the currently used portion. The copying procedure operates character by character, marking a character as it is copied. This procedure alters each tape cell twice: once to write the character for the first time, and again to mark that it has been copied. The position of the original Turing machine's tape head is marked on the tape. When copying the cells at or adjacent to the marked position, the tape content is updated according to the rules of the original Turing machine.

To carry out the simulation with a write-once machine, operate as before, except that each cell of the previous tape is now represented by two cells. The first of these contains the original machine's tape symbol and the second is for the mark used in the copying procedure. The input is not presented to the machine in the format with two cells per symbol, so the very first time the tape is copied, the copying marks are put directly over the input symbols. ■

3. Variable Increment

   Let us extend IMP with a common feature, namely variable increment, which increments the variable (right away) and returns its new value in context:

$$AExp \ ::= \ \cdots \mid {+}{+}Var$$

   Add variable increment to the big-step OS: list all the new rules, as well as the new versions of all the already existing rules that need to change.

Solution:

Variable increment, which is an arithmetic expression, has the side effect of changing program state. So we need to introduce new transition relations of the form $\langle a, \sigma\rangle \Downarrow \langle i, \sigma'\rangle$ or $\langle b, \sigma\rangle \Downarrow \langle t, \sigma'\rangle$.

We add a new rule for variable increment:

$$\frac{\langle x, \sigma\rangle \Downarrow \langle i, \sigma\rangle}{\langle {+}{+}x, \sigma\rangle \Downarrow \langle x \leftarrow i +_{int} 1, \sigma[x \leftarrow i +_{int} 1]\rangle}$$

And all existing rules for arithmetic and boolean expressions now yields not only a value but also an updated program state. We may assume that when multiple operands are involved, the evaluation always start from the leftmost one:

$$\frac{\langle a_1, \sigma\rangle \Downarrow \langle i_1, \sigma''\rangle \quad \langle a_2, \sigma''\rangle \Downarrow \langle i_2, \sigma'\rangle}{\langle a_1 + a_2, \sigma\rangle \Downarrow \langle i_1 +_{int} i_2, \sigma'\rangle}$$

$$\frac{\langle a_1, \sigma\rangle \Downarrow \langle i_1, \sigma''\rangle \quad \langle a_2, \sigma''\rangle \Downarrow \langle i_2, \sigma'\rangle}{\langle a_1 / a_2, \sigma\rangle \Downarrow \langle i, \sigma'\rangle} \text{ where } i_2 \neq 0 \text{ and } i \text{ is the quotient of } i_1 \text{ by } i_2$$

$$\frac{\langle a_1, \sigma\rangle \Downarrow \langle i_1, \sigma''\rangle \quad \langle a_2, \sigma''\rangle \Downarrow \langle i_2, \sigma'\rangle}{\langle a_1 \leq a_2, \sigma\rangle \Downarrow \langle \mathbf{true}, \sigma'\rangle} \text{ where } i_1 \leq_{int} i_2$$

$$\frac{\langle a_1, \sigma\rangle \Downarrow \langle i_1, \sigma''\rangle \quad \langle a_2, \sigma''\rangle \Downarrow \langle i_2, \sigma'\rangle}{\langle a_1 \leq a_2, \sigma\rangle \Downarrow \langle \mathbf{false}, \sigma'\rangle} \text{ where } i_1 >_{int} i_2$$

$$\frac{\langle b_1, \sigma\rangle \Downarrow \langle \mathbf{false}, \sigma'\rangle}{\langle b_1 \ \&\& \ b_2, \sigma\rangle \Downarrow \langle \mathbf{false}, \sigma'\rangle}$$

$$\frac{\langle b_1, \sigma\rangle \Downarrow \langle \mathbf{true}, \sigma''\rangle \quad \langle b_2, \sigma''\rangle \Downarrow \langle t, \sigma'\rangle}{\langle b_1 \ \&\& \ b_2, \sigma\rangle \Downarrow \langle t, \sigma'\rangle}$$

$$\frac{\langle b, \sigma\rangle \Downarrow \langle \mathbf{false}, \sigma'\rangle}{\langle !\ b, \sigma\rangle \Downarrow \langle \mathbf{true}, \sigma'\rangle}$$

$$\frac{\langle b, \sigma\rangle \Downarrow \langle \mathbf{true}, \sigma'\rangle}{\langle !\ b, \sigma\rangle \Downarrow \langle \mathbf{false}, \sigma'\rangle}$$

Also some rules for statements and programs are changed as follow:

$$\frac{\langle a, \sigma\rangle \Downarrow \langle i, \sigma'\rangle}{\langle x := a, \sigma\rangle \Downarrow \langle \sigma'[x \leftarrow i]\rangle}$$

$$\frac{\langle b, \sigma\rangle \Downarrow \langle \mathbf{true}, \sigma''\rangle \quad \langle s_1, \sigma''\rangle \Downarrow \langle \sigma'\rangle}{\langle \mathbf{if}\ (b)\ s_1\ \mathbf{else}\ s_2, \sigma\rangle \Downarrow \langle \sigma'\rangle}$$

$$\frac{\langle b, \sigma\rangle \Downarrow \langle \mathbf{false}, \sigma''\rangle \quad \langle s_2, \sigma''\rangle \Downarrow \langle \sigma'\rangle}{\langle \mathbf{if}\ (b)\ s_1\ \mathbf{else}\ s_2, \sigma\rangle \Downarrow \langle \sigma'\rangle}$$

$$\frac{\langle b, \sigma\rangle \Downarrow \langle \mathbf{false}, \sigma'\rangle}{\langle \mathbf{while}\ (b)\ s, \sigma\rangle \Downarrow \langle \sigma'\rangle}$$

$$\frac{\langle b, \sigma\rangle \Downarrow \langle \mathbf{true}, \sigma''\rangle \quad \langle s\ \mathbf{while}\ b\ s, \sigma''\rangle \Downarrow \langle \sigma'\rangle}{\langle \mathbf{while}\ (b)\ s, \sigma\rangle \Downarrow \langle \sigma'\rangle}$$