CMSC 303 Introduction to Theory of Computation, VCU
Spring 2019, Assignment 5 Key

Due: Tuesday, March 19, 2019

Turned in electronically in PDF, PNG or Word format before the start of class

Total marks: 45 marks + 3 marks bonus for typing your solutions in LaTeX.

Unless otherwise noted, the alphabet for all questions below is assumed to be $\Sigma = \{0, 1\}$.

1. [10 marks] This question asks you to examine the formal definitions of a TM and related concepts closely. Based on these definitions, answer the following.

   (a) A *configuration* of a Turing Machine (TM) consists of three things. What are these three things?

   Solution: The location of the tape head, the tape contents, and the state of the machine.

   (b) Can input alphabet Σ contain the blank symbol $\sqcup$? Why or why not?

   Solution: No — otherwise, the TM has no way to figure out where the input ends.

   (c) The tape is infinite. Is the tape alphabet infinite?

   Solution: No, the alphabet is finite.

   (d) Can a Turing machine's head *ever* be in the same location in two successive steps?

   Solution: Yes. This happens if we are at the leftmost cell on the tape and try to move the head left.

   (e) What is the difference between a decidable language and a Turing-recognizable language?

   Solution: A language L is decidable if there exists a TM which halts on any input $x \in \Sigma^*$ and accepts if $x \in L$, or rejects if $x \notin L$. For a recognizable language, only the acceptance condition is required. In other words, if $x \notin L$, the TM can either halt and reject or it can loop forever.

2. [12 marks] This question requires you to read a Turing Machine and determine if a given string is accepted by the TM. For each string below, show the tape at each step and state whether the TM accepts the string or not:

   (a) [3 marks] 0110

   Solution: Rejects

$Tape$	$Configuration$
$\underline{0}$ 1 1 0	q_1 0 1 1 0
$\sqcup$ $\underline{1}$ 1 0	$\sqcup$ q_2 1 1 0
$\sqcup$ 1 $\underline{1}$ 0	$\sqcup$ 1 q_2 1 0
$\sqcup$ 1 1 $\underline{0}$	$\sqcup$ b b q_2 a
$\sqcup$ 1 $\underline{1}$ $\sqcup$	$\sqcup$ 1 q_3 1 $\sqcup$
$\sqcup$ $\underline{1}$ 1 $\sqcup$	$\sqcup$ q_3 1 1 $\sqcup$
$\underline{\sqcup}$ 1 1 $\sqcup$	q_3 $\sqcup$ 1 1 $\sqcup$
$\underline{\sqcup}$ 1 1 $\sqcup$	q_1 $\sqcup$ 1 1 $\sqcup$
$\sqcup$ $\underline{1}$ 1 $\sqcup$	$\sqcup$ q_{reject} 1 1 $\sqcup$

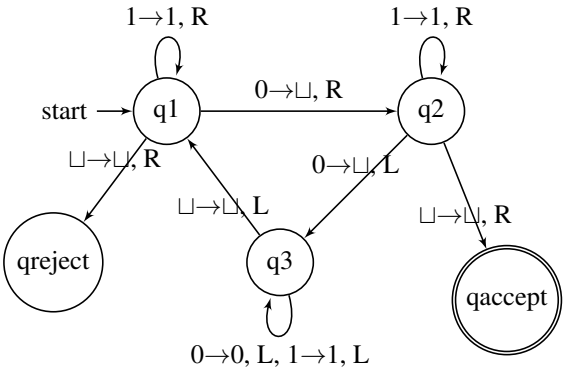

Figure 1: Turing Machine Finite Automata

(b) [3 marks] 010101

Solution: Rejects

$Tape$	$Configuration$
$\underline{0}\ 1\ 0\ 1\ 0\ 1$	$q_1\ 0\ 1\ 0\ 1\ 0\ 1$
$\sqcup\ \underline{1}\ 0\ 1\ 0\ 1$	$\sqcup\ q_2\ 1\ 0\ 1\ 0\ 1$
$\sqcup\ 1\ \underline{0}\ 1\ 0\ 1$	$\sqcup\ 1\ q_2\ 0\ 1\ 0\ 1$
$\sqcup\ \underline{1}\ \sqcup\ 1\ 0\ 1$	$\sqcup\ q_3\ 1\ \sqcup\ 1\ 0\ 1$
$\underline{\sqcup}\ 1\ \sqcup\ 1\ 0\ 1$	$q_3\ \sqcup\ 1\ \sqcup\ 1\ 0\ 1$
$\underline{\sqcup}\ 1\ \sqcup\ 1\ 0\ 1$	$q_1\ \sqcup\ 1\ \sqcup\ 1\ 0\ 1$
$\sqcup\ \underline{1}\ \sqcup\ 1\ 0\ 1$	$\sqcup\ q_{reject}\ 1\ \sqcup\ 1\ 0\ 1$

(c) [3 marks] 110

Solution: Accepts

$Tape$	$Configuration$
$\underline{1}\ 1\ 0$	$q_1\ 1\ 1\ 0$
$1\ \underline{1}\ 0$	$1\ q_1\ 1\ 0$
$1\ 1\ \underline{0}$	$1\ 1\ q_1\ 0$
$1\ 1\ \sqcup\ \underline{\sqcup}$	$1\ 1\ \sqcup\ q_2\ \sqcup$
$1\ 1\ \sqcup\ \sqcup\ \underline{\sqcup}$	$1\ 1\ \sqcup\ \sqcup\ q_{accept}\ \sqcup$

(d) [3 marks] What language does this Turing Machine describe?

Solution: Any string containing exactly one 0.

3. [10 marks] This question gets you to practice describing TM's at a semi-low level. Let

$$L = \left\{ ww^R \mid w \in \{0,1\}^* \right\}$$

Recall that w^R means w written backwards. Give an implementation-level description of a TM that decides L. By *implementation-level description*, we mean a description similar to Example 3.11 in the text (i.e. describe how the machine's head would move around, whether the head might mark certain tape cells, etc.... Please do *not* draw a full state diagram (for your sake and for ours)).

Solution:

(a) If the tape is empty, halt and accept.

(b) Start at the left-most tape symbol.

(c) Read the symbol and mark it, scan to the right-most non-empty tape symbol that is not marked. If there are no unmarked symbols, reject.

(d) Check the unmarked symbol found to see if it is the same symbol. If not, reject.

(e) Mark this symbol.

(f) Scan left to the left-most unmarked symbol and repeat Step C. If there are no more unmarked symbols, accept.

4. [8 marks] This question investigates a variant of our standard TM model from class.

   (a) [6 marks] Consider a TM that, when it moves, left, automatically moves to the leftmost cell. We call this a TM a *Left-Start TM*. Describe how a Left-Start TM can simulate a standard TM. (Hint - Use the fact that you have the ability to mark cells as you move through the tape and change the content of cells.)

   Solution:

   i. Move right through the TM as you would normally do so.
   ii. When you need to move left, mark the cell you are currently at.
   iii. Move left to the start cell, replace the symbol at the start cell with a $.
   iv. Move right and replace the value in the current cell with the value in the previous cell (shift the contents of the tape one cell to the right). If you encounter a marked symbol, leave the mark at the current cell, while shifting the contents one cell to the right. Continue until you have rewritten each non-empty symbol one cell to the right.
   v. Move left to the start cell.
   vi. Move right to the marked cell. This will now be one symbol to the left.

   (b) [2 mark] What does this imply about the sets of languages recognized by both models and about the models themselves?

   Solution: This implies that Turing Machines are robust and that small changes in the machines do not impact the power or ability of the Turing Machine to recognize a given language

5. [5 marks] This question allows you to explore variants of the computational models we've defined in class. Describe how Turing Machines and PDAs are different. How can you equate a Turing Machine to a PDA? (Hint - What would a PDA need to be equivalent to a Turing Machine)

   Solution: PDAs have only one stack to use for memory. This limits the power of the language to what can be pushed and popped from a stack. A Turing Machine has a tape, where the position on the tape can easily change and the contents of the tape can change. A PDA can have one additional stack and can then have the same power as the Turing Machine. Below is the proof:

   Show that a two-stack PDAs are at least as powerful as TMs by showing how to simulate a TM, $M = (Q_M, \Sigma, \Gamma_M, \delta_M, q_0, q_{accept}, q_{reject})$, with two stack PDA, $P = (Q_M \cup Q_P, \Sigma, \Gamma_M \cup \$, \Gamma_M \cup \$, q_0', q_{accept})$. Note that the states of P have all the states of M plus extra ones needed for the simulation and a new start state. Also note there are two stack alphabets, that have a special symbol $ in the TM tape alphabet to mark the bottom of the stack.
   P = "on input $w = w_1 w_2 \ldots w_n$, with each $w_i \in \Sigma$:

   (a) Push a $ on each stack.

   (b) Read in w onto the first stack. The reverse of the input string is now on the stack. Note that since we have read the whole input string, every other transition of P will now be $\epsilon-$transitions.

   (c) Pop each symbol on the first stack and push it on the second stack until the $ is on top of the first stack and move the machine to state q_0. Now the second stack contains w in the proper order. We will now use the second stack as the top being the tape head and everything under it is what is to the right of the tape head on the tape of M. The first stack will be used to store what is to the left of the tape head on the tape of M.

(d) Now we will simulate M's transition by showing how a left and right transition are accomplished with P:

i. Left transition: If $\delta_M(q_i, w_i) = (q_i', w_i', \text{L})$, then we have w_i on top of the second stack.

A. If \$ is on top of the first stack, reject, since M would hang by moving off the left edge of the tape.

B. Otherwise, pop w_i from the top of the second stack and push w_i' onto the second stack.

C. Pop the symbol off the top of the first stack and push it onto the second stack (The symbol that was to the left of the tape head is now under the tape head), switching the state of the machine to q_i'.

D. If q_i' is the accept state, then P accepts w. If q_i' is the reject state, then P rejects w. Otherwise return to (d).

ii. Right transition: If $\delta_M(q_i, w_i) = (q_i', w_i', R)$, then we have w_i on top of the second stack.

A. Pop w_i from the top of the second stack and push w_i' onto the first stack. (Now the tape head is on the symbol to the right of the previous head position)

B. If \$ is now on top of the second stack, push $\sqcup$ onto the second stack (The tape head moved right from the last non-blank symbol of the tape). In either case switch the state of the machine to q_i'.

C. If q_i' is the accept state, then P accepts w. If q_i' is the reject state, then P rejects w. Otherwise return to (d)."

Since P uses its stacks to simulate the transitions of M and accepts and rejects as M does, we see that two-stack PDAs are at least as powerful as TMs.