Xiaohui Bei October 24, 2017

Algorithms and Theory of Computation

MAS714 2017

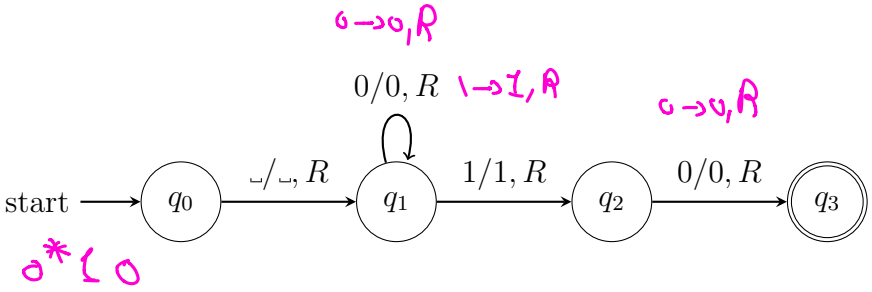
Exercises on Turing Machines

Exercise 1:
Simulate the Lecture 17 Example 1 TM that accepts language $L = \{a^n b^n c^n \mid n \geq 1\}$ on input "aaabbbccc". List the state and the tape content of every step of the run.

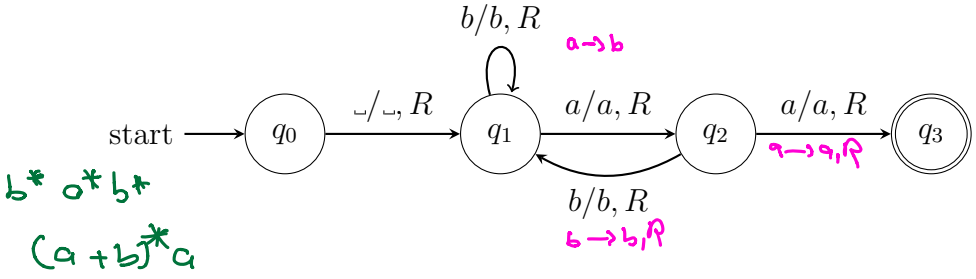
Solution: Skipped.

Exercise 2:
Describe what are the languages accepted by the following Turing machines.
Note: in figures below, "␣" means the blank symbol.

(a).



(b).



Solution:

(a). $L(M) = 0^*10(1+0)^*$

(b). $L(M) = (a+b)^*aa(a+b)^*$

Exercise 3:
As discussed in class, a common operation in Turing-machine programs involves "shifting over". We need to move the contents of each of the cells to the right of the current head position one cell right, and then find our way back to the current head position.
Give the high level description of how to perform this operation.

Solution: On input string w:

(a). We maintain a set of states corresponding to the symbol we just read.

(b). Mark the first symbol with a \$ (assume $\$ \notin \Sigma$) and go on to the state corresponding to that symbol read.

(c). Move the head toward right till the end of the string (till you hit $\sqcup$) while replacing the current symbol with the previous symbol (this information is contained in the state the you are currently in), shifting to the state corresponding to the symbol just read.

(d). After hitting the $\sqcup$ at the end of the input, replace the $\sqcup$ with the last symbol and start going left back to the start position.

(e). Clear the $\sqcup$ symbol and move right one position to place the head right on top of the first symbol of the shifted string.

(f). Move one position backwards without doing anything to the tape to place the head right on top of the original start position, accept.

(g). For any situation not described above, reject.

Exercise 4:
Examine the formal definition of a Turing machine to answer the following questions, and explain your reasoning.

(a). Can a Turing machine ever write the blank symbol on its tape?

(b). Can the tape alphabet Γ be the same as the input alphabet Σ?

(c). Can a Turing machine contain just a single state?

Solution:

(a). Yes. The tape alphabet Γ contains the blank symbol $\sqcup$.

(b). No. Σ never contains $\sqcup$, but Γ always contains $\sqcup$. So they cannot be equal.

(c). No. Any Turing machine must contain two distinct states $q_{\text{accept}}, q_{\text{reject}}$.

Exercise 5:
A *Turing machine with doubly infinite tape* is similar to an ordinary Turing machine, but its tape is infinite to the left as well as to the right. The tape is initially filled with blanks except for the portion that contains the input. Computation is defined as usual except that the head never encounters an end to the tape as it moves leftward. Show that this type of Turing machine recognizes the class of Turing-recognizable languages.

Solution: A TM with doubly infinite tape can simulate an ordinary TM. It marks the left-hand end of the input to detect and prevent the head from moving off of that end. To simulate the doubly infinite tape TM by an ordinary TM, we show how to simulate it with a 2-tape TM, which was already shown to be equivalent in power to an ordinary TM. The first tape of the 2-tape TM is written with the input string, and the second tape is blank. We cut the tape of the doubly infinite tape TM into two parts, at the starting cell of the input string. The portion with the input string and all the blank spaces to its right appears on the first tape of the 2-tape TM. The portion to the left of the input string appears on the second tape, in reverse order.

Exercise 6:
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: 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 contents 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.

Exercise 7:

(a). Show that the set of decidable languages is closed under the union operation.

(b). Show that the set of recursively enumerable languages is closed under the union operation.

Solution:

(a). For any two decidable languages L_1 and L_2, let M_1 and M_2 be the TMs that decide them. We construct a TM M' that decides the union of L_1 and L_2:
On input w:

(a) Run M_1 on w. If it accepts, *accept*.

(b) Run M_2 on w. If it accepts, *accept*. Otherwise, *reject*.

It is easy to check that M' accepts w if either M_1 or M_2 accepts it. If both reject, M' rejects.

(b). For any two r.e. languages L_1 and L_2, let M_1 and M_2 be the TMs that recognize them. We construct a TM M' that recognizes the union of L_1 and L_2:
On input w:

(a) Run M_1 and M_2 alternatively on w step by step. If either accept, *accept*. If both halt and reject, *reject*.

If either M_1 or M_2 accepts w, M' accepts w because the accepting TM arrives to its accepting state after a finite number of steps. Note that if both M_1 and M_2 reject and either of them does so by looping, then M' will loop. By definition this means M' rejects w.