Quiz 4: NFAs and TMs

Collaboration: This quiz is open note but individual. You may use any resources that were provided to you by the course, or that you had recorded as part of your own notes prior to when you first viewed this quiz. You may not discuss this quiz with your cohort-mates, class-mates, tutor, friends, family, or anyone else except the course staff (who will answer clarification questions only).

Problem 1: Non-Deterministic Finite State Automata

Give a non-deterministic finite state automaton which decides each of the languages described below, using no more than the number of states indicated. Give both a drawing and a description.

(a) $\{x \in \{0,1\}^* \mid x$ contains the substring $111\}$, using no more than four states.

(b) Original problem was impossible so it is dropped.

(c) $\{x \in \{0,1\}^* \mid x$ does not contain the substring $1\}$, using no more than one state.

Problem 2: Turing Machine Running Time

Below I have provided a Turing machine which computes the following language: $\{w\#w^R | w \in \{0,1\}^*\}$. Where w^R refers to the reverse of the string w (so if $w = 1011$ then $w^R = 1101$). In other words, this language has all strings over the alphabet $\{0,1,\#\}$ that have the format of: a binary string, then the $\#$ symbol, then the first string reversed. The following strings therefore belong to the language:

– $1011\#1101$

– $00\#00$

– $1\#1$

– $\#$

And then these strings do not belong to the language:

– $1011\#110$

– $01\#01$

– $1\#$

– ε

– $0\#\#0$

– 0110

The behavior of the Turing Machine is as follows (note that this description is of slightly higher specificity than our expectations for Turing Machine "psuedo-code"):

1. Check that the string has exactly one $\#$ character by (see yellow states of the machine):
   (a) After the start of tape character, move right so long as the read head is looking at a 0 or 1
   (b) Move to a new state once you see a $\#$ symbol (halt and return 0 if you get to $\emptyset$ without seeing one)
   (c) if anything other than a 0 or 1 is seen before $\emptyset$, return 0
2. From the "inside out", cross off 0s and 1s from either side of the $\#$ character by (see blue states):

(a) Go to the middle of the input by moving left until the read head is at #

(b) Move left until you find a 0 or 1. Mark that off by overwriting with an x, remember which was seen within the states.

(c) Go to the middle of the input by moving right until the read head is at #

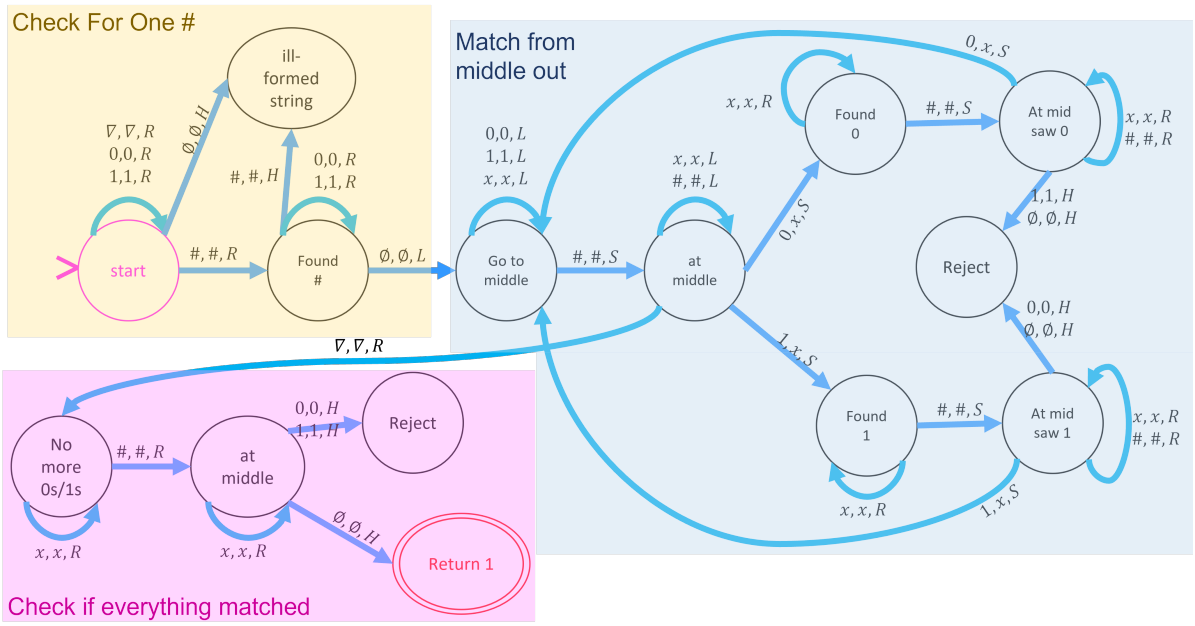
(d) If the firsts non-x character seen matches the character remembered, go back to the first step (a). Otherwise return 0.

3. Check whether all 0s and 1s on both sides of the # character have been crossed off by (see pink states):

(a) If when looking for a 0 or 1 on the left-hand side of # you see the start of the tape, go to the middle of the input (i.e. to the # character)

(b) after the # character, check if the first non-x character is $\emptyset$. If it is, return 1, otherwise return 0.

The following image depicts one potential implementation of the machine described above:



Consider that this machine is given each of the inputs describe below, all of which are $n+1$ characters long. Give a Θ bound on the total number of transitions this machine would take before halting as a function of the input length n. You should informally justify how you came to that conclusion (e.g. "after each 0 we overwrite we need to take one more transition"), but a formal proof is not necessary.

a. A string of n 1s followed by # (e.g. for $n = 4$ this would be 1111#)

b. The # character followed by a string of n 0s (e.g. for $n = 4$ this would be #0000)

c. A string of $\frac{n}{2}$ 1s followed by the # character, then followed by $\frac{n}{2}$ 1s. You may assume n is even. (e.g. for $n = 4$ this would be 11#11)

d. A string of $n+1$ #s (e.g. for $n = 4$ this would be #####)

Problem 3: Describe the TM

Each subproblem below gives a function that can be computed by a Turing machine. Describe, using high level Turing machine "pseudo-code", how a Turing machine could compute the given function. For example, your pseudo-code might have statements like "scan left until you find a 1, then cross it off", or "halt and return 1 if there you get to the start of the tape without seeing a 1". Refer to Problem 1 to see an example.

For all problems, the input alphabet of the Turing machine will be $\{0, 1, \#\}$ where the input string given will be $w\#x$ where w and x are the inputs to the functions below (so, for example, to compute $SameLength(11, 011)$ the Turing machine's input would be $11\#011$). You may use whatever tape alphabet you wish.

(a) for $w \in \{0,1\}^*, x \in \{0,1\}^*$,

$$SameLength(w, x) = \begin{cases} 1 & \text{if the length of } w \text{ matches the length of } x. \\ 0 & \text{otherwise} \end{cases}$$

(b) for $w \in \{0,1\}^*, x \in \{0,1\}^*$,

$$DoubleLength(w, x) = \begin{cases} 1 & \text{if the length of } w \text{ is twice the length of } x. \\ 0 & \text{otherwise} \end{cases}$$

