COMS W3261
Computer Science Theory
Homework #3

Alexander Roth

2014 – 10 – 27

Problems

1. Informally describe a Turing machine that accepts all strings of the form $\{\, a^n b^n c^n \mid n \geq 1 \,\}$. Show the sequence of ID's that your TM goes through starting with the input *aabbcc*.

Solution: Let us construct the TM that will accept the language $\{\, a^n b^n c^n \mid n \geq 1 \,\}$. Initially, it is given a finite sequence of a's, b's, and c's on its tape, preceded and followed by an infinity of blanks. Alternatively, the TM will change an a to an X, a b to a Y, and a c to a Z, until all a's, b's and c's have been matched.

Starting at the left end of the tape, it will change an a to an X, it then moves to the right passing over any a's and Y's it encounters. When it comes upon a b, it will change that into a Y and continue to the right, ignoring any subsequent b's and Z's. When it comes across a c, it transforms the c into a Z and continue to the right. When it reads a B, it will move to left, over any Z's, c's, Y's, b's, and a's it encounters. When it reaches an X, it looks for an a to the immediate right. If that is found, it repeats the process till it accepts. If, instead, the Turing machine reads a Y, it will continue to read down the string to the right. If it reads past the string and reads a blank, it accepts the string, as there all a's, b's and c's have been matched.

The turing machine looks like this:

$$M = (\{q_0, q_1, q_2, q_3, q_4, q_5\}, \{a, b, c\}, \{a, b, c, X, Y, Z, B\}, \delta, q_0, B, \{q_5\}$$

where the transition function is given by the following table:

State	Symbol						
	a	b	c	X	Y	Z	B
q_0	(q_1, X, R)	$-$	$-$	$-$	(q_0, Y, R)	(q_0, Z, R)	(q_5, B, L)
q_1	(q_1, a, R)	(q_2, Y, R)	$-$	$-$	(q_1, Y, R)	$-$	$-$
q_2	$-$	(q_2, b, R)	(q_3, Z, R)	$-$	$-$	(q_2, Z, R)	$-$
q_3	$-$	$-$	(q_3, c, R)	$-$	$-$	$-$	(q_4, B, L)
q_4	(q_4, a, L)	(q_4, b, L)	(q_4, c, L)	(q_0, X, R)	(q_4, Y, L)	(q_4, Z, L)	$-$
q_5	$-$	$-$	$-$	$-$	$-$	$-$	$-$

Thus, for the string $aabbcc$, we have the following sequence:

$q_0aabbcc \vdash Xq_1abbcc \vdash Xaq_1bbcc \vdash XaYq_2bcc \vdash XaYbq_2cc\check{d}ash$
$XaYbZq_3c \vdash XaYbZcq_3B \vdash XaYbZq_4cB \vdash XaYbq_4ZcB \vdash XaYq_4bZcB \vdash$
$Xaq_4YbZcB \vdash Xq_4aYbZcB \vdash q_4XaYbZcB \vdash Xq_0aYbZcBashXXq_1YbZcB \vdash$
$XXYq_1bZcB \vdash XXYYq_2ZcB \vdash XXYYZq_2cB \vdash XXYYZZq_3B \vdash XXYYZq_4ZB \vdash$
$XXYYq_4ZZB \vdash XXYq_4YZZB \vdash XXq_4YYZZB \vdash Xq_4XYYZZB \vdash XXq_0YYZZB \vdash$
$XXYq_0YZZB \vdash XXYYq_0ZZB \vdash XXYYZq_0ZB \vdash XXYYZZq_0B \vdash XXYYZq_5ZB$

Since we have entered state q_5, we accept the string $aabbcc$.

2. Consider the following Turing machine

$$M = (\{A, B, C, D\}, \{a\}, \{a, X, 0, 1, \#\}, \delta, A, \#, \{D\}).$$

Here, we are using # for the blank symbol. The transition function δ is given by the following table:

State	a	X	0	1	#
A	BXL	AXR	$A0R$	$A1R$	$C\#L$
B	BaL	BXL	$A1R$	$B0L$	$A1R$
C	$-$	$C\#L$	$D0R$	$D1R$	$-$
D	$-$	$-$	$-$	$-$	$-$

(a) Show the sequence of ID's that M goes through starting with the input $aaaa$.

Solution:

$Aaaaa \vdash B\#Xaaa \vdash 1AXaaa \vdash 1XAaaa \vdash 1BXXaa \vdash$
$B1XXaa\check{d}ashB\#0XXaa \vdash 1A0XXaa \vdash 10AXXaa \vdash 10XAXaa \vdash$
$10XXAaah10XBXXa \vdash 10BXXXa \vdash 1B0XXXa \vdash 11AXXXa \vdash$
$11XAXXa \vdash 11XXAXa \vdash 11XXXAa \vdash 11XXBXX \vdash 11XBXXX \vdash$
$11BXXXX \vdash 1B1XXXX \vdash B10XXXX \vdash B\#00XXXX \vdash 1A00XXXX \vdash$
$10A0XXXX \vdash 100AXXXX \vdash 100XAXXX \vdash 100XXAXX \vdash 100XXXAX \vdash$
$100XXXXA\# \vdash 100XXXCX\# \vdash 100XXCX\#\# \vdash 100XCX\#\#\# \vdash 100CX\#\#\#\# \vdash$
$10C0\#\#\#\#\# \vdash 100D\#\#\#\#\#$

M halts on the final state D when given the string $aaaa$ as input; thus, $aaaa$ is accepted.

(b) Starting with an input consisting of n a's, $n > 0$, what string will this Turing machine have on its tape after it has halted?

Solution: The string that will be on the tape of the Turing machine will consist of a binary representation of n. For example, a string consisting of 3 a's (aaa) would yield a string on the tape as 11####. Since we now represent the blank symbol as #, the number of # is insignificant.

(c) Briefly explain how the Turing machine does this computation and characterize the role of each state.

Solution: This Turing machine repeatedly finds its rightmost a and replaces that a with an X. It then moves to the left, searching for a #, 0, or 1. If it finds a #, it will write a 1 over the #, start moving to the right, passing over any X's, O's, or 1's until it comes upon another a. This process continues until it reads off the rightmost index of the input string, at which point it reads a # (provided this is a continuous string of a's). It will then convert all the placeholder X's to blank symbols and halt when it reaches a 0 or a 1.

The States

A**:** This state begins the cycle, and breaks it when appropriate. The Turing machine will scan to the right for an a, at which point it will transition to state B after writing an X over that a. It will pass over any 0's or 1's found on the tape. If at any point it reads a blank symbol while in this state, it will transition to state C, and pass over the blank.

B**:** In this state, the Turing machine searches to the left for any 0's or #. If it reaches a 0, it converts it to a 1 and enters state A. If it reads a #, it transforms that into a 1; both allowing the process to repeat. If it moves over a 0, it will remain in state B, but it will change the 0 to a 1.

C**:** At this stage, the Turing machine is reading the blank symbol adjacent to the rightmost input symbol. Thus, it will read over the string, transforming any X it comes across into blanks. Once it reads a 0 or a 1, it transitions to state D.

D**:** The sole purpose of this state is to allow the Turing machine to halt when it has finished its task.

(d) Using big-O notation, how many moves will this Turing machine make on an input consisting of n a's before halting? Briefly justify your answer.

Solution: This Turing machine operates in $O(n^2)$ time. Suppose we have a string of n a's. It will locate the nth a after traversing down the string and adjusting the counter $n - 1$ times. Thus, it will take roughly n times to go through a string of length n. The time taken to adjust the binary string is approximately $n \log(n)$. Finally, there are n moves to transform all X's at the end of the string to blank symbols. However, since we are using Big-O notation, only the n^2 matters.s

3. Let $L = \{\, p \,|\, p$ is a polynomial over a single variable x with an integral root $\}$. (Example: $2x^3 - 9x^2 + 16$ is a polynomial over x with an integral root 4 and would therefore be in L.) Describe at an informal level a Turing machine M such that $L(M) = L$ showing that L is recursively enumerable.

Solution: We shall construct a Turing machine M that takes as input the polynomial p. It will move along the tape and in the process set x to successive values (e.g., 0, 1, -1, 2, ...). M will evaluate p over these different values of x. If at any point p evaluates to 0, M will halt and accept. However, if p has a root that is not integral, M will never halt.

4. Post's Correspondence Problems.

   (a) Is PCP with a single-symbol alphabet decidable? Briefly justify your answer.

   Solution: The PCP with a single-symbol alphabet is decidable. Since we are using only a single-symbol alphabet, all that matters is the length of each string in the pair. Thus, we can define A_i and B_i to be two strings that form the pairs for the PCP problem where i represents the number of pairs in the string. We can assume that there must be at least two pairs in the problem; otherwise, the solution would be decidable if $A = B$ and for no other solutions. Therefore, we have a four cases we must account for:

   Case 1 $|A_i| = |B_i|$ for at least one pair in the problem. Thus, this pair provides a solution to the problem.

   Case 2 $|A_i| > |B_i|$ for all pairs in the problem. Thus, the length of strings formed for A will always be longer than those strings that are formed for B. From this scenario, we can see that there is no solution. Without loss of generality, we can switch A_i with B_i and find similar results (i.e., there is a string with a longer length in B than A for all pairs; thus, there is no solution).

   Case 3 $|A_i| > |B_i|$ for at least one pair in the problem, and $|B_j| > |A_j|$ for at least one other pair. We can find some solution to this case by

   (b) Is PCP with a two-symbol alphabet decidable? Briefly justify your answer.

   Solution: The PCP with a two-symbol alphabet is undecidable. Suppose we have the original PCP with an n-sized alphabet and a modified PCP with a binary alphabet. We shall reduce the first PCP to the instance of PCP over the binary alphabet. We apply the homomorphism that maps a symbol α_i as 1^i0, where i is the length of 1's formed from the index of the symbol. That is, for any instance of the PCP over the n-sized alphabet, we have a mapping to the binary alphabet. From this, we have reduced the PCP over an n-sized alphabet can be reduced to a binary PCP. Since the n-sized PCP is undecidable, we know that the binary PCP is undecidable.

5. What class of languages can a Turing machine recognize if it

   (a) Has only two working states, and one accepting state from which it never makes any transitions?

Solution: This language is context-free. We can restrict the tape of this Turing machine to behave like the stack of a PDA. For each input symbol the machine reads in, the symbol will be added to the subsequent stack, which is represented by the tape to the left of the input head; that is, we have the input head read only left to right. The Turing machine will halt when the final state is reached without regards to what is on the stack like a PDA accepting by final state.

   (b) Never overprints a different symbol on the input tape? That is, if in the transition function for the Turing machine whenever (q, Y, D) is in $\delta(p, X)$, then $Y = X$.

Solution: This language is regular. Since it cannot overwrite any of the symbols, we know that it cannot have any memory of past actions; that is, it behaves like a DFA. This machine will only act on the next input without regard to what it previous saw, until it reaches some accepting state, when it will then halt.

   (c) Has only $\{0, 1, B\}$ as tape symbols?

Solution: This language is recursively enumerable. We can represent any Turing machine with this alphabet by using some homomorphism that maps it s alphabet to a set of binary strings generated from $\{0, 1, B\}$. Thus, we can encode any other language in binary.

   (d) Never moves its input head left?

Solution: This language is regular. Under this restriction, the Turing machine behaves like a DFA. That is, it maintains no memory of what it previously saw, and acts on the next input it reads from the tape, until it reaches some halting state. Thus, it is regular.

Give a brief one or two-sentence justification for each of your answers.