HARIKRISHNAN GOPAL JANAKIRAMAN
ID:- 01372580

(1)

91.502 Foundations of Computer Science

Fall 2013

Take-home Final

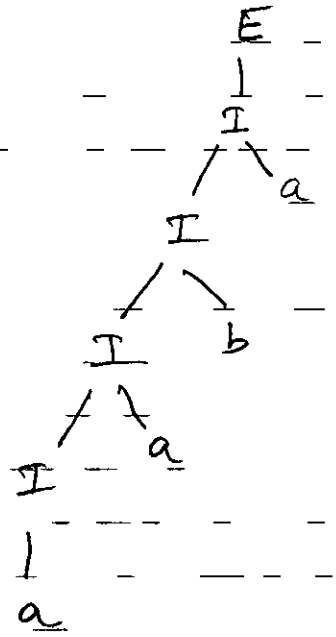
1) $S \rightarrow AB$
$A \rightarrow aA \mid a$
$B \rightarrow bAB$

$S \rightarrow AB \rightarrow aAB \rightarrow aaB \rightarrow aabAB \rightarrow aababB$

$L(G_1) = \{ a^i b^j a^k B \mid i, j, k > 0 \}$

2) $S \rightarrow AB$
$A \rightarrow AA \mid a$
$B \rightarrow bA$

(a)



(b) $S \rightarrow AB \rightarrow AAB \rightarrow aAB \rightarrow aaB \rightarrow aabA \rightarrow aaba$

(c) $S \rightarrow AB \rightarrow AbA \rightarrow AbAA \rightarrow AbAa \rightarrow Abaa \rightarrow abaa$

(2)

3) (a) Let L be the complement of the language

$\{a^n b^n : n \geq 0\}$

$\Rightarrow L = \{a^n b^m : n \neq m\} \cup \{(a \cup b)^* ba (a \cup b)^*\}$

Let Leftmost language be L_1 and Rightmost Language be L_2

The CFG for L_1 is

$S_1 \rightarrow aS_1b \mid X \mid Z$

$X \rightarrow aX \mid a$

$Z \rightarrow Zb \mid b$

The CFG for L_2 is

$S_2 \rightarrow YbaY$

$Y \rightarrow YY \mid a \mid b \mid \varepsilon$

CFG that generate $L = L_1 \cup L_2$ is

$S \rightarrow S_1 \mid S_2$
$S_1 \rightarrow aS_1b \mid X \mid Z$
$S_2 \rightarrow YbaY$
$X \rightarrow aX \mid a$
$Z \rightarrow Zb \mid b$
$Y \rightarrow YY \mid a \mid b \mid \varepsilon$

(3)

4) i)
$S \rightarrow TT \mid U$
$T \rightarrow 0T \mid T0 \mid \#$
$U \rightarrow 0U00 \mid \#$

a) L(G) in Plain language.

The language which is generated by L = L(G) is the set of strings that either are composed by the concatenation of 3 (or) more arbitrary length of zeros (or) strings of the form $0^k \# 0^{2k}$ for $k \geq 0$. A word w in L is of the form $w = 0^{k_1} \# 0^{k_2} \# \ldots \# 0^{k_4}$ for $i \geq 3$ (where $k_j \geq 0$ for all j) (or) the form $w = 0^k \# 0^{2k}$ for $k \geq 0$.

b) L(G) is not Regular.

To prove, : For contradiction, assuming L is regular. There exists a Pumping length $p > 0$ such that for any word $w \in L$ & $|w| \geq p$, There by Splitting the word w into 3 pieces as $w = xyz$, $|y| > 0$, $|xy| \leq p$ for any integer $i \geq 0$.

$w' = xy^iz$ belongs to a language L. For our case, Considering the word $w = 0^p \# 0^{2p}$. w belongs to L & $|w| \geq p$, thereby it must satisfy the theorem conditions.

Partitioning the word w into 3 Pieces xyz such that y is composed only by 0's from the leftmost sequence of 0's.

④

Thereby, $y = 0^k$ for $k > 0$. Pumping down the word Q, such that $Q' = 0^{p-k} \# 0^{2p}$ do not belong to L because $k > 0$. Any Possible Partition of Q causing no Pumping up for all $i > 0$. Overall we concluded with the Contradiction.

Thereby, L should be non-regular.

3(b) $\{ x_1 \# x_2 \# \ldots \# x_k : k \geq 1 \}$, each $x_i \in \{a,b\}^*$, and for some i and j, $x_i = x_j^R$ }.

The CFG for the given language is

$$P \rightarrow Q \mid R \# Q \# R \mid R \# Q \mid Q \# R$$
$$Q \rightarrow aQa \mid bQb \mid \# \mid \#R\#$$
$$R \rightarrow aR \mid bR \mid \#R \mid \varepsilon$$

5) CFG to Chomsky Normal Form:-

$$A \rightarrow BAB \mid B \mid \varepsilon$$
$$B \rightarrow 00 \mid \varepsilon$$

Assuming that A is a start variable, thereby $A \rightarrow \varepsilon$.

Eliminating the $B \rightarrow \varepsilon$ and ~~changing~~ the right side of A to obtain $A \rightarrow BAB \mid BA \mid AB \mid A \mid B \mid \varepsilon$.

And also, Eliminating unit rules, we get $A \rightarrow A \mid B$ and adding $A \rightarrow 00$. At last, Replacing $A, B \rightarrow 00$ by $A, B \rightarrow PQ$, $P \rightarrow 0$, $Q \rightarrow 0$, and replacing $A \rightarrow BAB$ by $A \rightarrow YB$, $Y \rightarrow BA$

(5)

b) Algorithmic descriptions of Turing machines :-

a) $\{w \mid w$ contains twice as many 0's as 1's$\}$

step-1:- Scan the tape and mark the first 1 which has not been marked. If no unmarked 1's are found go to step 5. Else move the head back to the start of the tape.

step-2:- Scan the tape until an unmarked 0 is found, mark the 0, if no 0's are found then reject it.

step-3:- Scan the tape once more until an unmarked zero is found, mark the 0, if no 0's are found then reject it.

step-4:- move the head back to the start of the tape and go to step-1.

step-5:- move the head back to the start of the tape. Scan the tape to see if any unmarked 0's are found. If none are found accept, else reject it.

b) $\{w : w$ does not contain twice as many 0's as 1's$\}$.

step-1:- Scan the tape and mark the first 1 which has not been marked. If no unmarked 1's are found go to step 5. Else move the head back to the start of the tape.

step-2:- Scan the tape until an unmarked 0 is found, mark the 0, if no 0's are found accept.

Step-3:- Scan the tape once more until an unmarked zero is found, mark the zero, if no 0's are found accept it.

Step-4:- Move the head back to the start of the tape and go to step 1.

Step-5:- Move the head back to the start of the tape. Scan the tape to see if any unmarked 0's are found. If none are found reject it, else accept it.

7) 000111 - FALSE
1000 1 - FALSE
01 01 - FALSE
0000 - FALSE
B - TRUE

(a) 0110011 → 1011
(b) 001110 → 0110
(c) 100100 → 000

M does:-

Turing Machine M, starts with state q_1, and accepts inputs 0, 1, B. Furthermore, it has states q_1, q_H (HALT) until in state q_1, it receives tape Symbol B, the system will not go to halt state and the system goes on looping stage.

(7)

18) a) $\{\langle M\rangle : M$ is a TM that accepts 6 or more different inputs$\}$

- It is not Turing-recognizable.

We cannot recognize a set of language themselves. The reason is that the typical language, being infinite, cannot be considered as finite-length string that could be input to a TM.

Rather we must recognize the Turing machines that accepts those languages; the TM code itself is finite, even if the language it accepts is infinite.

b) $\{\langle M\rangle : M$ is a TM that accepts 5 or fewer different inputs$\}$.

- It is Turing-Recognizable.

Step-1:- The string w could be the lexicographically or string of not more than 5 different input in Σ^*.

Step-2:- R counts the number of different input M makes and if its count not exceeding 5, then R accepts.

Step-3:- Else, if M reaches 6 or more inputs, R erases w, and generates the lexicographically 5 different inputs and repeats the above steps.

Step-4:- If R has simulated M on the lexicographically last 6 different input string and even on the string M has halted in 6 (or) more input, then R Rejects.

9) Undecidable subset of $\{a\}^*$

Solution:-

There are two reasonable ways to solve this problem. The method I used is diagonalization. Let $M_1, M_2, \ldots$ be a list of all Turing machines. Since there are only countably many TM's, we know there is such a list (we don't even need to be able to obtain (M_i) given i, since we are not going to show an algorithm for anything).

We define our diagonalizing languages as $D = \{a^k \mid a^k \notin L(M_k)\}$. This is clearly a subset of $L(a^*)$, since it only contains strings of the form a^k. We also know that D does not equal $L(M_n)$ for any n, because those two sets disagree about whether to include a^n. Since the list $M_1, M_2, \ldots$ was exhaustive, D is not the language of any TM. So not only is D undecidable, it is also unrecognizable.

10) a) Turing machine (TM) is a 7-tuple $m =$

$(Q, \Sigma, T, \delta, q_0, B, F)$ where

Q: a finite set of states of the finite control.
Σ: finite set of input symbols.
T: set of tape symbols, with Σ being a subset
δ: transition function $\delta(q, X) = (P, Y, D)$ where

(9)

- q : the state in Q;
- X : a tape symbol being scanned.
- P : the next state in Q.
- Y : the tape symbol written on the cell being scanned used to replace X.
- D : either L (Left) or R (Right) telling the move direction of the tapehead.

q_0 - Start state in Q.

B - the Blank Symbol in Γ, not in Σ

F - the set of final (or) accepting states.

Designing a Turing machine to divide by length of string by three without remainder.

→ Starting at the left end of the input.

→ move to the right for every characters until it finds a blank symbol.

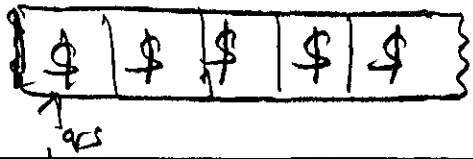
→ move left and replace every string with $. The number of replaced symbol provides the length.

→ Now it reaches the starting state.

→ Divide the length by 3 and discard the remainder and go to halt state.

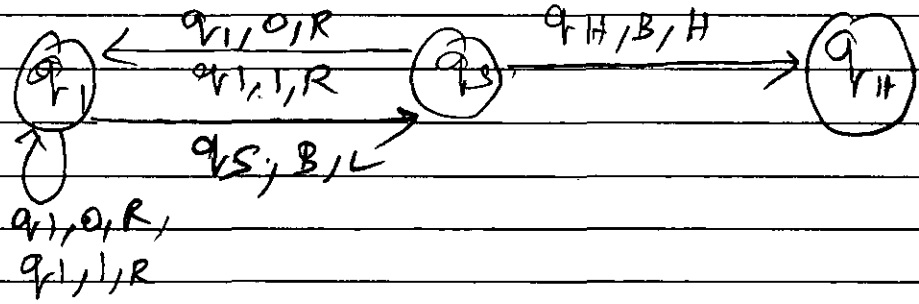
(10)

Tape



State	0	1	B
q_0	$q_1, 0, R$	$q_1, 1, R$	q_H, B, H
q_1	$q_1, 0, R$	$q_1, 1, R$	q_5, B, L

q_0 0011 → 0$q_1$11 → 00$q_1$1 → 001q_1 → 0011B



(b) Designing a Turing machine with arbitrary replacing $ when repeated more than three times.

→ Starting at the left end of the tape

→ move to the right side in tape 0 for every character until it finds a blank symbol.

→ During each move, the following steps are taken

(i) For every new character, a new tape is created
eg:- For 1, tape 1 is created,
For 2, tape 2 is created,
For 1, data one is added to the tape 1.

(11)

(i) tape1...tapen is created, in any of those tape, whose value is updated more than B times, that data is considered arbitrary value.

→ Then the tape 0 is replaced with the arbitrary value by moving to the left side. The number of replaced symbol provides the length.

→ Now it reaches the starting symbol.

→ Divide the length by 3 & discard the remainder and go to the halt state.

(c) TO generalize the division "by three" to any value.

→ starting at the left end of the input.

→ move to the right for every characters until it finds a blank symbol.

→ move left and replace every string with $. the number of replaced symbol provides the length.

→ now it reaches the starting state.

→ Divide the length by randomly chosen value but not greater than length and discard the remainder.

→ Go to the halt state.

(12)

11) $n^2 = O(n)$ FALSE
$3^n = 2^{O(n)}$ FALSE
$2^{2n} = O(2^n)$ FALSE

12) b) NP is closed under union, Intersection, concatenation and kleene star.

Intersection:-
M = "on Input w:
1. Run M_1 on w. If M_1 rejected then reject.
2. Else run M_2 on w. If M_2 rejected then reject.
3. Else accept.

so M is a Poly-time nondeterministic decider for $L_1 \cap L_2$.

Union:-

1. Run M_1 on w. If M_1 accepted then accept.
2. Else run M_2 on w. If M_2 accepted then accept.
3. Else reject.

so M is a polytime nondeterministic decider for $L_1 \cup L_2$.

concatenation:-

1. Non deterministically Split w into w_1, w_2 such that $w = w_1 w_2$.
2. Run M_1 on w_1. If M_1 rejected then reject.
3. Else run M_2 on w_2. If M_2 rejected then reject.

4. Else accept.

So M is a poly-time nondeterministic decider for $L_1 \circ L_2$

Kleenestar:-

1. If $w = \varepsilon$ then accept.
2. Nondeterministically select a number m such that $1 \leq m \leq |w|$.
3. Nondeterministically split w into m pieces such that $w = w_1 w_2 \ldots w_m$.
4. For all i, $1 \leq i \leq m$, run M_1 on w_i. If M_1 rejected then reject.
5. Else (M_1 accepted all w_i, $1 \leq i \leq m$). accept.

So M is a poly-time nondeterministic decider for L_1^*.

12 (a) Operations on:-

Intersection:- $L_1 \cap L_2$

Union:- $L_1 \cup L_2$

Complement:- $\overline{L} = -L$.

Concatenation of L_1 and L_2 = $\{x_1 x_2 : x_1 \text{ in } L_1 \text{ and } x_2 \text{ in } L_2\}$

Closure (or) kleenestar of L

$L^* = \{\varepsilon\} \cup L \cup L^2 \cup L^3 \cup \ldots$

$L^* = L^0 \cup L^1 \cup L^2 \cup \ldots \cup L^n$

where L^n is L concatenated n times with itself.