TOC - Assignment 2

Nirjhar Nath

BMC202239

(1) Given that L is a regular language. Consider the language $L' = \{w \mid ww \in L\}$. We want to prove that L' is regular.

Since L is regular, $\exists$ a DFA $M = (Q, \Sigma, \delta, q_0, F)$ which recognizes L.

To show that L' is regular, we give an NFA that accepts L'. We construct it as follows:

- The automaton starts from the initial state $q_0 \in Q$ as well as the middle state $q_m \in Q$,
- and parallelly parse two occurences of w, one from q_0 to q_m and another from q_m to q_f where $q_f \in F$ is a final state.

The NFA $M' = (Q', \Sigma, \delta', q_0', F')$, where

$$Q' = Q \times Q \times Q$$

$$q_0' = \{(q_0, q_m, q_m) \mid q_m \in Q\}$$

$$F' = \{(q_m, q_f, q_m) \mid q_m \in Q,\ q_f \in F\}$$

$$\delta' = \{((q_1, q_2, q_m), a, (q_1', q_2', q_m)) \mid \delta(q_1, a) = q_1',\ \delta(q_2, a) = q_2'\}$$

recognizes $L' = \{w \mid ww \in L\}$. $\therefore$ L' is regular. $\blacksquare$

(2) M is a DFA with n states. Let p, q be two states.
Let Q be the set of states of M and F be the set of final states.

Consider the DFA M' with states $Q \times Q$. For a state $(a,b) \in Q \times Q$, define $\hat{\delta}$ by:

$$\hat{\delta}((a,b), s) = (\hat{\delta}(a,s), \hat{\delta}(b,s))$$

We say that two states p, q in M are distinguishable if $(\delta(p,s) \in F$ and $\delta(q,s) \notin F)$ or $(\hat{\delta}(p,s) \notin F$ and $\delta(q,s) \in F)$.

$\therefore$ Two states p, q are distinguishable if

$$\hat{\delta}((p,q), s) \in (F \times F^c) \cup (F^c \times F)$$

from (p,q), we might have to go to all possible states before reaching such ~~as a states~~ a state.

Since there are n^2 states in $Q \times Q$.
$\therefore$ If $x \in \Sigma^*$ is the shortest string that distinguishes between p and q, then we have,

$$|x| \leq n^2$$

&

For the lower bound, we see that if $p \in F$ and $q \notin F$, then ϵ distinguishes p and q, i.e., the lower bound is 0.

$\therefore$ $0 \leq |x| \leq n^2$, as desired.

(3) Let $G = (V, E)$ be a graph in which V corresponds to the set of states in the machine M; E is the set of edges, i.e., two states a, b have an edge between them if one is reachable from the other on reading some inputs.

Emptiness Algorithm

Let $v_0 \in V$ correspond to the ~~start state~~ initial state q_0.

- Start from v_0.
- Mark all vertices that are reachable from ~~q_0~~ v_0. (We can use BFS or DFS for that purpose)
- If $\exists$ a vertex $v \in V$ that is marked and if v corresponds to some $q_f \in F$, where F is the set of final states of the machine, then $L \subseteq \Sigma^* \times \Gamma^*$ is non-empty, otherwise it is empty.

Finiteness Algorithm

- Start from v_0 (as in the previous algorithm).
- Mark every vertex $v \in V$ such that $v_f \in V$ is reachable from v, where v_f corresponds to some final state of the machine M. (We can use BFS or DFS for this).
- If $\exists$ a cycle starting at any of the marked states, then L is infinite, otherwise it is finite.

(4) Given words $u, v \in \Sigma^*$, a shuffle $u \circ v$ of words is any word of length $|u|+|v|$ that can be split into two subsequences that are u and v.

For languages L and L', the shuffle $L \circ L'$ is the set of all $u \circ v$ such that $u \in L$ and $v \in L'$.

Suppose L and L' are both regular. We have to prove that $L \circ L'$ is regular.

Let $M = (Q, \Sigma, \delta, q_0, F)$ and $M' = (Q', \Sigma', \delta', q_0', F')$ be DFAs such that $L(M) = L$ and $L(M') = L'$.

Define the NFA $M'' = (Q'', \Sigma'', \delta'', q_0'', F'')$, as

$Q'' = Q \times Q'$, $q_0'' = (q_0, q_0')$,

$\Sigma'' = \Sigma \times \Sigma'$, $F'' = F \times F'$

and, $\delta'' : Q'' \times Q'' \to Q''$ such that

$$\delta''((q, q'), a) = \begin{cases} \{(\delta(q,a), q'), (q, \delta(q', a))\} & \forall a \in \Sigma \times \Sigma' \\ \{(\delta(q,a), q')\}, & \forall a \in \Sigma \\ \{(q, \delta'(q', a))\}, & \forall a \in \Sigma' \end{cases}$$

Then $L \circ L'$ is accepted by M'' and hence it is regular.

Claim: Context-free languages are not closed under shuffle.

Proof: Let $L=\{a^n b^n \mid n>0\}$

$L'=\{c^n d^n \mid n>0\}$

We shall use pumping lemma to prove that $L \circ L'$ ~~is not a shuffle~~ is not a context-free language.

For this, assume to the contrary, that $L \circ L'$ is context-free. Then ~~there exist~~ suppose p is the pumping length.

Consider the string $z = a^p c^p b^p d^p$. Decompose z as $z = uvwxy$ where ~~vwxy~~ $|vwx| \le p$, $vx \neq \epsilon$.

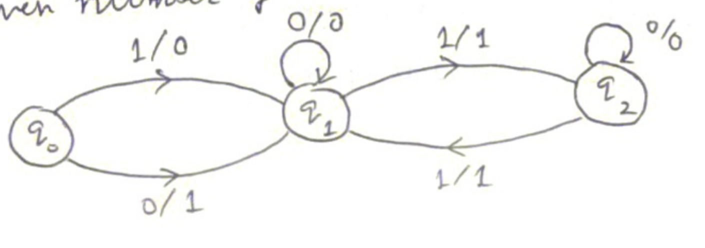
Then, vwx cannot contain both a, b or c, d together.

This is because we can pump v, x two times to get $z_2 = uv^2wx^2y$ then either number of a's $\neq$ number of b's or number of c's $\neq$ number of ~~c~~ d's or both, in the string z_2. $\therefore z_2 \notin L \circ L'$, a contradiction to the ~~pu~~ pumping lemma. $\therefore$ CFL's are not closed under shuffles. ▨

(5) $f : \Sigma^* \to \Sigma^*$ is a <u>reduction</u> from a ~~langu~~ language L to L' if $\forall w \in \Sigma^*$, ~~$w \in L \iff f(w)$~~

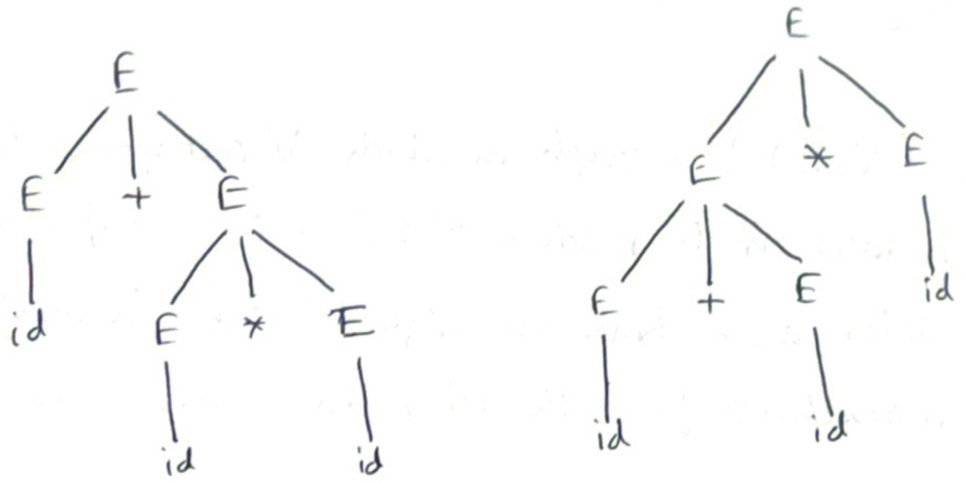
$$w \in L \iff f(w) \in L'.$$

A Mealy machine that computes a reduction from L to L', where L consists of all $w \in \{0,1\}^*$ with odd number of 1's and L' consists of all $w \in \{0,1\}^*$ with an even number of 1's is:



⑥ Given CFG is $E \to E+E \mid E*E \mid (E) \mid id$

The string $id+id*id$ fromed by the given CFG has two parse trees:



∴ The given CFG is an ambiguous grammar.

An equivalent unambiguous grammar for the language is as follows:

$$E \to F \mid E+F$$
$$F \to G \mid F*G$$
$$G \to (E) \mid id$$

In this grammar, all productions are left recursive for operators that have the same precedence.
∴ The parse tree only grows on the left side.
For operators that have different precedence, the operators with lower precedence will be at higher levels in the parse tree than those with higher precedence.

Thus, the grammar we gave is unambiguous. ▨

(7) G is the grammar $S \to aS \mid aSbS \mid \epsilon$.

for any $x \in L(G)$, we shall prove the given statement by induction on the number of steps in the derivation $S \Rightarrow x$:

When number of steps in the derivation $S \Rightarrow x$ is 1, then only the empty string ϵ can be derived, in which the number of a's = 0 = number of b's.

Thus the statement is true for 1 step.

Hypothesis: Suppose the statement is true for all strings with k steps.

Induction Step (We shall prove it for k+1 steps.) Then for a string $x \in L(G)$ with $k+1$ steps in the derivation $S \Rightarrow x$, we divide it into two cases as below:

- The first step is $S \Rightarrow aSbS$:

  Then every prefix of every string generated by aS has more a's than b's, and every prefix of every string generated from aSb has number of a's $\geq$ number of b's.
  By Induction Hypothesis, every prefix of every string generated from the second S of $aSbS$ has number of a's $\geq$ number of b's. So the induction step holds true for derivations with first step $S \Rightarrow aSbS$.

- The first step is $S \Rightarrow aS$

  Then every prefix of every string generated by aS has more a's than b's. $\therefore$ The induction step holds true for this case too.

$\therefore$ $L(G)$ consists of all strings x over a, b s.t. every prefix of x has no. of a's $\geq$ number of b's.

(8) Given language is $L = \{a^i b^{i^2} \mid i \geq 0\}$. Assume, to the contrary, that L is context-free. Then it satisfies the conditions of the pumping lemma:

$\exists\, p$, called the pumping length, such that for any string $z \in L$, $|z| \geq p$, we can write $z = uvwxy$ such that $vx \neq \epsilon$, $|vwx| \leq p$ and $uv^i w x^i y \in L\ \forall i \geq 0$.

Take the string $z = a^p b^{p^2} \in L$. Decompose it as $z = uvwxy$ such that $vx \neq \epsilon$ and $|vwx| \leq p$. Then,

- if either v or x or both contain a and b, then the string $uv^2wx^2y \notin L$ as there will be an a that comes after a b, a contradiction.
- if vx consists of only a's, then $uv^0wx^0y \notin L$ as the number of a's will be less than p but the number of b's will be equal to p^2, a contradiction.
- if vx consists of only b's then the same argument as above works.
- if v contains a's and x contains b's. Then, z will be of the form

$$z = a^{p-k_1-k_2} a^{k_1} a^{k_2} b^{k_3} b^{k_4} b^{p^2-k_3-k_4} = uvwxy$$

such that $k_1+k_2+k_3+k_4 \leq n \Rightarrow k_1+k_4 \leq n$.

Now consider the string $uv^2wx^2y = a^{p-k_1-k_2} a^{2k_1} a^{k_2} b^{k_3} b^{2k_4} b^{p^2-k_3-k_4}$

$$= a^p a^{k_1} b^{k_4} b^{p^2}$$

$\therefore\ k_4 \leq p - k_1$. But for $uv^2wx^2y \in L$, we should have

$$(p+k_1)^2 = p^2 + p - k_1$$

But since $p^2+p-k_1 < p^2+2pk_1+k_1^2$, we get a contradiction. Thus, L is not context-free. $\blacksquare$

(9) Consider the string $z = 01^n1^n001^n \in L = \{ww^Rw \mid w \in \{a,b\}^*\}$.

Assume to the contrary that L is context-free.

Decompose z as $z = uvwxy$ such that $|vwx| \leq n$, ~~vwx~~ $vx \neq \epsilon$. Then vx can contain 0 or 1 or 2 0's otherwise after pumping, it won't be of the form ww^Rw.

If vx has 0 0's, then pumping v,x 0 times, we ~~get~~ get the string uv^0wx^0y. If we write this string in the form ww^Rw, it is not possible that the number of 1's in the parts w, w^R, w will be the same, which is a contradiction to the pumping lemma.

If vx has 1 0's, then again pumping 0 times, the string uv^0wx^0y cannot be partitioned into ~~w,w~~ ww^Rw such that the number of 1's in the parts w, w^R, w are the same, a contradiction

Also if vx has 2 0's, a similar argument gives a contradiction.

$\therefore$ L does not satisfy the conditions of the pumping lemma.

$\therefore$ L is not context-free.

Claim: L^c is context-free

Proof: Consider $L' = \{xyz \mid |x| = |y| = |z|,\ x \neq y^R\}$.
Then L' is generated by the following grammar:

$$S \to T \mid U$$
$$T \to Va \mid [illegible]\ Vb$$
$$V \to \Sigma V \Sigma\Sigma \mid bX$$
$$X \to \Sigma\Sigma X \Sigma \mid a$$
$$U \to Wa \mid Wb$$
$$W \to \Sigma W \Sigma\Sigma \mid aY$$
$$Y \to \Sigma\Sigma Y \Sigma \mid b$$

$\therefore L'$ is a CFL.

Now consider $L'' = \{xyz \mid |x| = |y| = |z|,\ z \neq y^R\}$

~~Then L'' is generated by the following grammar:~~

Thus, clearly L'' is the reverse of L'. Since CFL's are closed under reverses, $\therefore$ L'' is also a CFL.

~~[illegible]~~

Let $L''' = \{w \mid |w| \not\equiv 0 \pmod 3\}$

Then L''' is generated by the following grammar:

$$S \to \Sigma\Sigma T \mid \Sigma T$$
$$T \to T \Sigma T \Sigma T \Sigma T \mid \epsilon$$

$\therefore$ L''' is a CFL.

~~Likewise~~ Also, $L' \cup L''$ is a CFL, since CFL's are closed under union.

Now, observe that $L^c = (L' \cup L'') \cup L'''$
which is again a union of two CFL's.

$\therefore$ L^c is context-free. ☐

(10) We need to show that $L = \{ucv^R \mid u, v \in \{a,b\}^*$ and v is not a prefix of $u\}$ is context-free.

The following grammar generates L:

$$S \to S_1 \mid S_2$$
$$S_1 \to aS_1a \mid bS_1b \mid aTb \mid bTa \mid Kb \mid Ka$$
$$T \to UcU$$
$$U \to aU \mid bU \mid \epsilon$$
$$S_2 \to cbU \mid caU$$
$$K \to cU$$

$\therefore$ L is context-free.

~~Claim: $L^c = \{ucv^R \mid u, v \in \{a,b\}^*$ and v is a prefix of $u\}$.~~
~~Proof: L^c, hence, consists of all strings w such that the number of w's in~~

Claim: $L^c = \{ucv^R \mid u, v \in \{a,b\}^*$ and v is a prefix of $u\} \cap$
$\{w \mid$ number of c's in w is 0 or more than 1$\}$.
is also context-free.

Proof:
The following grammar ~~for~~ generates L^c:

$$S \to S_1 \mid S_2 \mid S_3$$
$$S_1 \to aS_1a \mid bS_1b \mid Uc$$
$$U \to aU \mid bU \mid \epsilon$$
$$S_2 \to U$$
$$S_3 \to UccKU$$
$$K \to cK \mid \epsilon$$

Thus, L^c is context-free. ▨