COL 352 Intro to Automata and Theory of Comput.
Minor 2, Sem II 2018-19, Max 40, Time 1 hr

Name ___________________________ Entry No. ___________________________ Group

Note (i) Write your answers neatly and precisely in the space provided with each question including back of the sheet. You won't get a second chance to explain what you have written.
(ii) You can quote any result covered in the lectures without proof but any other claim should be formally justified.
(iii) You can make use of Dirichlet theorem - in any sequence \( a + b \cdot i, i > 0 \) where \( a, b \) are relatively prime there are infinitely many primes congruent to \( a \) modulo \( b \).

1. Consider the language \( L = (00 + 11)^+ \). Describe the equivalence classes of strings over \( \{0, 1\} \) of the relation \( R_L \) (Myhill Nerode relation). (5)

   There are five equivalence classes corresponding to the five states of the min state DFA. These are \( \epsilon, (00 + 11)^+, 0(00)^*, 1(11)^*, (01+10)\cdot(0 + 1)^* \). It can be verified that none of the states are equivalent as per \( R_L \) and only the second set of strings correspond to the accepting state.

2. Are the following languages CFL? Justify or prove otherwise. (5 × 2)

   (a) The language PAREN2 consists of all balanced strings over \( (,), [\cdot] \). For example, \( [[[\cdot]]]\) is balanced but \( [[\cdot]] \) is not. In other words, the two distinct parenthesized strings should be individually balanced over the pairs \( (,) \) and \( [\cdot] \) respectively but the balancing cannot be interspersed.

   \[ S \rightarrow (|]|SS|(S)[S] \]

   (b) \( \{0^i| i \text{ is composite}\} \).

   By using pumping lemma on a sufficiently long string \( 0^N = uvwxy \) where \( 1 \leq |v| + |x| \leq n \leq N \), all strings \( 0^{ki} \cdot 0^{N-k} \in L \) Let \( N - k = a \) and \( k = b \). To apply Dirichlet theorem, we need to ensure that \( a, b \) are co-prime. Note that we can choose \( N > n \) where \( n \) is the parameter of the PL. So we can choose a prime \( p > n \) and \( N = p^2 \) since \( N \) must be composite.

3. Describe a procedure to convert a well-formed (valid) regular expression \( r \) into an equivalent CFG \( G \) with some underlying justification. Illustrate this on the r.e. \( (0 \cdot 1 + 1^*)^* \) for all strings over \( \{0, 1\} \). (10)

   Hint: Use the recursive definition of r.e.

   We will do this by induction on the length of the regular expressions. The base cases are the unit length symbols of \( \Sigma \) and \( \epsilon \). So these are the productions of the form \( S \rightarrow a \ a \in \Sigma \)

   Suppose we can represent all regular expr upto length \( n \). Then either

   \begin{itemize}
   \item \( r = r_1 + r + 2 \) where \( |r_i| < n \). Then add a production \( S \rightarrow S_1|S_2 \) where \( r_i \) can be generated by CFGs with start symbol \( S_i \).
   \item \( r = r_1 \cdot r_2 \), then using similar idea \( S \rightarrow S_1 \cdot S_2 \)
   \item \( r = r'^* \) where \( |r'| < n \). Then add the productions \( S \rightarrow SS' \ \epsilon \) where \( S' \) derives \( r' \). \( S \rightarrow SS' \)
   \item \( r = (r') \) where \( |r'| < n \), then add \( S \rightarrow (S) \).
   \end{itemize}

Since all the r.e.s can be constructed recursively, we can obtain a CFG for any given r.e. For the given expression, we can use the above construction in a bottom up fashion using adequate number of variables.
Let $A \to 0$ $B \to 1$ and $C \to A \cdot B$. Thus $C$ derives the r.e. $01$.
Similarly $D \to D \cdot B|\epsilon$ allows $D$ to derive $1^*$. Subsequently let $E \to C|D$ allows $E$ to derive the r.e. $01 + 1^*$. Finally $S \to SE|\epsilon$ derives the original r.e.

Note that we can also add another production $S \to (S)$ to take care of parenthesis.

4. Given a CFL $L$ describe an algorithm to decide if it contains any string NOT of the form $(0 \cdot 1)^i$ for some $i > 0$. (It need not contain all such strings). (6 )

Since $(0 \cdot 1)^i$ is regular so are strings that belong to the complement of this set, say $R$. Assume that the CFL is given in CNF. Then, we can set the parameter of the PL as $n = 2^k$ where $k$ is the number of variables. Using closure property of CFL under intersection with a regular language is a CFL. Therefore $R \cap L$ is a CFL. Now we can use the emptiness testing on $R \cap L$ as an algorithm for the decision problem.

5. Consider the languages
$L_1 = \{(01)^i| i \geq 0\}$, $L_2 = \{0^i \cdot 1^i| i \geq 0\}$ $L_3 = \{0^i1^i2^i| i \geq 0\}$.
Consider the following machine models where $Q$: states $\Sigma$ input alphabet $\Gamma$: tape alphabet/Stack alphabet
$M_1$ ordinary Turing machine $\delta_1 : Q \times \Gamma \to Q \times \Gamma \times \{L, R\}$
$M_2$ A Turing machine that is not allowed to overwrite, with transition function $\delta_2 : Q \times \Gamma \to Q \times \{L, R\}$
$M_3$ A deterministic PDA with two stacks. $\delta_3 : Q \times \Sigma \cup \{\epsilon\} \times \Gamma \times \Gamma \to Q \times \Gamma^* \times \Gamma^*$.

For each Machine model, identify which languages it can recognize. (3 + 3 +3 )

<table>
<thead>
<tr>
<th>Machine</th>
<th>Which languages does it recognize (proof not needed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M_1$</td>
<td>$L_1$ $L_2$ $L_3$</td>
</tr>
<tr>
<td>$M_2$</td>
<td>$L_1$</td>
</tr>
<tr>
<td>$M_3$</td>
<td>$L_1$ $L_2$ $L_3$</td>
</tr>
</tbody>
</table>

The underlying reasoning is that $M_2$ is equivalent to DFA and $M_3$ is equivalent to $M_1$. The proofs are not expected and takes some effort especially the characterization of $M_2$.

---

1Marks will be given only if you correctly identify all the languages for each machine