Extra problems - Sheet 2

1. Prove or disprove that the following languages are context-free.

   (a) $\{w\#w' \mid w, w' \in \Sigma^* \text{ and } w \neq w'\}$.

   (b) $\{0^n 1^m \mid n \leq m \leq 2n\}$.

   (c) $a^*b^*c^* - \{a^n b^n c^n \mid n \geq 0\}$.

   (d) $\{w \in \{0,1,2\}^* \mid \#_0(w) = \#_1(w) = \#_2(w)\}$

2. Assume that L is a regular language. Prove or disprove that the following languages are regular.

   (a) EXP $= \{w \mid w^{|w|} \in L\}$.

   (b) EQ-a-b $= \{xy \mid \#_a(x) = \#_b(y)\}$.

   (c) ORCond $= \{a^i b^j \mid$ either $i \geq j$ or i is odd$\}$

   (d) DoubleEXP $= \{w \mid \exists y : |y| = 2^{2^{|w|}} \text{ and } wy \in L\}$

   (e) EQ$' = \{x\$y \mid \#_a(x) = \#_b(y)\}$.

3. Give two languages such that L_1 and L_2 are not regular, but $L_1 \cap L_2$ is regular.

4. Give two languages such that L_1 and L_2 are not regular, but $L_1 \circ L_2$ is regular.

5. Prove or disprove that the following languages are regular.

   (a) $EQ = \{w \cdot w \mid w \in \Sigma^*\}$.

   (b) $Twice = \{w \in \{a,b\}^* \mid \#_a(w) = 2 \cdot \#_b(w)\}$.

   (c) $Prod = \{w \in \{a,b\}^* \mid \#_a(w) \cdot \#_b(w)$ is even$\}$.

   (d) $Len = \{w1^n \mid |w| = n\}$.

   (e) $NEQ = \{0^i 1^j \mid i \neq j\}$.

   (f) $L = \{a^n b^m c^{n-m} \mid n \geq m \geq 0\}$.

6. Let L be a regular language. One of the following languages is regular and the other is not. Give a proof and provide a counterexample, respectively.

   (a) $\{w \in \{a,b\}^* \mid \exists n \geq 0, \exists x \in L, x = w^n\}$

   (b) $\{w \in \{a,b\}^* \mid \exists n \geq 0, \exists x \in L, w = x^n\}$

7. Let L be any language (not necessarily regular) over a unary alphabet, i.e. $L \subseteq \{a\}^*$. Show that L^* is regular.

Tutorial 1

Notation: For $w \in \Sigma^*$ let w^R be the reverse of the string w, i.e. if $|w| = 1$ then $w^R = w$ and for $|w| > 1$ and $w = u \cdot a$ then $w^R = a \cdot u^R$.

1. Give a DFA over the alphabet $\{0, 1\}^*$ such that it accepts strings which are the binary representations of a number which is 3 (mod 5).
2. Give a DFA over the alphabet $\{0, 1\}^*$ such that it accepts strings which are the binary representations of a number which is not 3 (mod 5). i.e. it accepts a string if it is the binary representation of a number which is 0 (mod 5), 1 (mod 5), 2 (mod 5), or 4 (mod 5).
3. Give a DFA over the alphabet $\{a, b\}^*$ that accepts strings which do not contain aaa as a substring.
4. Give a DFA over the alphabet $\{a, b\}^*$ that accepts all strings which contain aab as a substring but do not contain aaa as a substring.
5. Give a DFA which accepts the following language.

   $L = \{w \in \{a, b\}^* \mid w$ contains atleast two a's and at most one b$\}$.
6. Give a DFA which accepts the following language.

   $\{w \in \{a, b\}^* \mid w$ either starts with the letter a or ends with the letter b$\}$.
7. Let L be the language in Question 5 above. What is the language $L \circ L$?
8. Is there any gadget/object that you use in your day-to-day life which can be modelled as a DFA? Write down the DFA for it.

Tutorial 2

Notation: Let $\Sigma = \{a, b\}$. For $w \in \Sigma^*$ let $|w|$ denote the length of w. Let $\#_a(w)$ denote the number of as in w and let $\#_b(w)$ denote the number of bs in w. Let w^R be the reverse of the string w. For a DFA/NFA A, let $L(A)$ denote the language accepted by A.

Definition 0.1. *Given a DFA $A = (Q, \Sigma, q_0, F, \delta)$, let $\delta^* : Q \times \Sigma^* \to Q$ be the function defined inductively as follows:*

for any $q \in Q$, $\delta^*(q, \varepsilon) = q$

for any $q \in Q, w \in \Sigma^*$ and $a \in \Sigma$, $\delta^*(q, wa) = \delta(\delta^*(q, w), a))$.

That is, given a state and a word $w \in \Sigma^*$, δ^* outputs the state in which A ends up, after reading the string w.

1. Given a DFA $A = (Q, \Sigma, q_0, F, \delta)$, come up with another DFA $A' = (Q', \Sigma, q_0', F', \delta')$ such that $L(A) = L(A')$, i.e. they accept the same language and for all $a \in \Sigma$ and for all $q \in Q'$ $|\delta(q, a)| = 1$.

2. Let L_1, L_2 be two regular languages accepted by DFAs $A_1 = (Q_1, \Sigma, q_0^1, F_1, \delta_1)$, $A_2 = (Q_2, \Sigma, q_0^2, F_2, \delta_2)$ respectively. Let $A = (Q, \Sigma, q_0, F, \delta)$ be the automata obtained by the product construction (recall the product construction presented in class). Prove the following statements:

   (a) For any $w \in \Sigma^*$, such that $|w| \leq 1$ and for any $q_1 \in Q_1$ and $q_2 \in Q_2$, prove that $\delta^*((q_1, q_2), w) = (\delta_1(q_1, w), \delta_2(q_2, w))$.

   (b) For any $w \in \Sigma^*$, $a \in \Sigma$ and for any $q_1 \in Q_1$ and $q_2 \in Q_2$, prove that $\delta^*((q_1, q_2), wa) = \delta\left((\delta_1^*(q_1, w), \delta_2^*(q_2, w)), a\right)$.

   (c) Using the two parts above, conclude that for any $w \in \Sigma^*$ and $(q, q') \in Q$, $\delta^*((q, q'), w) = (\delta_1^*(q, w), \delta_2^*(q', w))$.

   (d) Let $F = F_1 \times F_2$. From (2c), conclude that if w is accepted by A_1 and A_2, then w is also accepted by A.

   (e) Let $F = (F_1 \times Q_2) \cup (Q_1 \times F_2)$. From (2c), conclude that if w is accepted by A_1 or A_2, then w is also accepted by A.

3. Let L_1, L_2 be two regular languages. Show that $L_1 - L_2 = \{w \mid w \in L_1 \text{ and } w \notin L_2\}$ is also regular, i.e. give a DFA for $L_1 - L_2$ using the DFA for L_1, L_2.

4. Give an NFA for the following language.

$$L = \{w \cdot c \mid w \in \{0, 1, 2\}^*, c \in \{0, 1, 2\} \text{ and } c \text{ occurs in } w\}$$

5. Let $\Sigma = \{a_1, a_2, \ldots, a_k\}$. Draw an NFA with $k + 1$ states for the following language:

$$L := \{w \mid \exists i \text{ s. t. } 1 \leq i \leq k \text{ and } a_i \text{ does not appear in } w\}.$$

6. Let two-NFA be a non-deterministic finite state automata $A = (Q, \Sigma, q_0, F, \delta)$ in which all the elements are defined exactly like in an NFA, but with a slightly different accepting condiction.

Definition 0.2 (Accepting run)**.** *We say that a non-deterministic finite state automata A has an* accepting run *on a word $w \in (\Sigma \setminus \{\varepsilon\})^*$ if w can be written as $y_1 y_2 \dots y_m$, where $m \geq |w|$, each $y_i \in \Sigma$ and there exists a sequence of states $p_0, p_1, \dots, p_m$ such that*

- $p_0 = q_0$,
- $p_m \in F$,
- *and for every* $0 \leq i \leq m-1$ $\delta(p_i, y_{i+1}) = p_{i+1}$.

We say that a word $w \in (\Sigma \setminus \{\varepsilon\})^*$ is accepted by a twoNFA A if A has exactly two accepting runs on w and A is said to reject w if A has no accepting run on w.

Prove that given any twoNFA A, there exists an NFA, say B, such that $L(A) = L(B)$.

Tutorial 3

Notation: Let $\Sigma = \{a, b\}$. For $w \in \Sigma^*$ let $|w|$ denote the length of w. Let $\#_a(w)$ denote the number of as in w and let $\#_b(w)$ denote the number of bs in w.

1. Give an NFA (with ε moves) for the following regular expressions. (You may simplify the expression as much as possible.)
   (a) $(aa^* + bb^*)^*$
   (b) $(ab + ba) \cdot (ab + ba) \cdot (ab + ba)$
2. Write the regular expressions corresponding to the following languages.
   (a) $L = \{w \in \{a, b\}^* \mid \#_a(w) = 1 \pmod 2\}$.
   (b) $L = \{w \in \{a, b\}^* \mid$ every other letter is $a\}$
3. Recall the DFA that we constructed in class that accepts strings which are the binary representations of a number which is 0 (mod 3). Give a regular expression corresponding to the language using the conversion from DFA to regular expression presented in Lecture 5, 6.
4. Let $\Sigma = \{a_1, a_2, \ldots, a_k\}$. Recall the NFA with $k+1$ states that we constructed for the following language:

   $$L := \{w \mid \exists i \text{ s. t. } 1 \leq i \leq k \text{ and } a_i \text{ does not appear in } w\}.$$

   Let us call the NFA we constructed to be $A = (Q, \Sigma, q_0, F, \delta)$.
   (a) Create an NFA $A' = (Q', \Sigma, q_0', F', \delta')$ for L such that $|Q'| = |Q| + 1$, $\forall a \in \Sigma$, $\forall q \in Q'$ $|\delta'(q, a)| = 1$.
   (b) Draw a DFA corresponding to L using the subset construction. What is the number of states in the constructed DFA?
   (c) Show that any DFA accepting L must have $2^{\Omega(k)}$ states.
5. A homomorphism on a set Σ is a map from Σ to another set Σ'^* such that each letter in Σ is mapped to a string over Σ'. For example, say $\Sigma = \{0, 1\}$ and a function h is defined as follows $h(0) := aaab$ and $h(1) := aba$ then h is a homomorphism on $\{0, 1\}$. Let L be a regular language. Show that the following language is also regular

   $$h(L) := \{h(w) \mid w \in L\}.$$

6. Let L, L' be two regular languages. Let us define $L||L'$ as follows:

   $$L||L' := \{x_1y_1x_2y_2 \ldots x_ny_n \mid x_1x_2 \ldots x_n, y_1y_2 \ldots y_n \in \Sigma^*, x_1x_2 \ldots x_n \in L \text{ and } y_1y_2 \ldots y_n \in L_2\}$$

   Prove that $L||L'$ is regular by explicitly constructing an NFA/DFA for the language.
7. Prove or disprove that the following languages are regular.
   (a) $EQ = \{w \cdot w \mid w \in \Sigma^*\}$.
   (b) $Twice = \{w \in \{a, b\}^* \mid \#_a(w) = 2 \cdot \#_b(w)\}$.
   (c) $Prod = \{w \in \{a, b\}^* \mid \#_a(w) \cdot \#_b(w)$ is even$\}$.
   (d) $Len = \{w1^n \mid |w| = n\}$.
   (e) $NEQ = \{0^i1^j \mid i \neq j\}$. [This is a slightly hard problem.]

Tutorial 4

Notation: Let $\Sigma = \{a, b\}$. For $w \in \Sigma^*$ let $|w|$ denote the length of w. Let $\#_a(w)$ denote the number of as in w and let $\#_b(w)$ denote the number of bs in w. Let w^R be the reverse of the string w. For a language L, let L^R be equal to $\{w^R \mid w \in L\}$.

1. **Prove or disprove the following**
   (a) $L^*L^* = L^*$
   (b) If $L_1L_2 = L_2L_1$ implies that $L_1 = L_2$.
   (c) $(L^*)^* = L^*$
   (d) If $L_1^* = L_2^*$ then $L_1 = L_2$.
   (e) $(L^*)^R = (L^R)^*$.
   (f) If $L \cdot L = L$ then either $L = \emptyset$ or $\varepsilon \in L$.

2. Consider the language $L = \{w \in \Sigma^* \mid 2^{\text{nd}}$ letter from the end is $a\}$.
   (a) Draw an NFA for L.
   (b) Using the ideas of subset construction draw a DFA for L.
   (c) Using the DFA minimization idea discussed in class, check whether the DFA thus constructed is minimal or not. If it not a minimal DFA then draw the correspoding minimal DFA for it.
   (d) Let $L_k = \{w \in \Sigma^* \mid k^{\text{nd}}$ letter from the end is $a\}$. Prove using pigeon hole principle (or by any other method) that any DFA accepting L_k must have $\Omega(k)$ states.

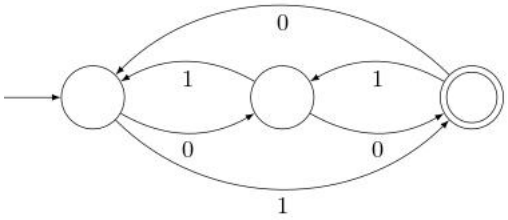
3. **Right congruence**
   Recall that we defined the right congruence property of an equivalence relation. We say that an equivalence relation $\equiv$ on Σ^* has the right congruence property if $\forall x, y \in \Sigma^*$ and $a \in \Sigma$, ($x \equiv y$ if and only if $xa \equiv ya$).
   (a) Prove that if $\equiv$ has the right congruence property then $\forall x, y, z \in \Sigma^*$, ($x \equiv y$ if and only if $xz \equiv yz$).

4. **Pumping Lemma vs. Myhill-Nerode**
   (a) Prove that the pumping lemma cannot be used to prove that the following language is non-regular: $L = \{a^n b^m c^\ell \mid m, n, \ell \geq 0$ and if $n = 1$ then $m = \ell\}$.
   (b) Use the Myhill-Nerode theorem to prove that the language above is not regular.

5. **DFA minimization**
   (a) Build an NFA for the following language:
   $L = \{w \in \{a, c, b, d\}^* \mid |w| > 1$ and the last letter in w does not appear anywhere else in w$\}$
   (b) Prove that the minimum DFA for the above language L must have at least 16 states.

6. **DFA to Regular expression**

(a) Given below is the description of a DFA. Give equivalent regular expression for that DFA.



(b) Let $x \in \Sigma^*$ be any string and $L_x = \{y \mid xy \in L\}$. How many distinct L_x languages are possible for the above DFA?

Tutorial 5

1. Let us assume that we are working over $\Sigma = \{a, b\}$.

   (a) Consider the following language.

   $$L_i = \{w \mid i\text{th letter from the end in } w \text{ is an } a\}.$$

   Let s_i denote the number of states in an NFA that recognizes this language. Show that $s_i = O(i)$.

   (b) Construct a DFA for L_i with $2^{O(s_i)}$ states. Can you give an exact constant hidden under the $O(\cdot)$ in the exponent?

   (c) Argue using the minimization algorithm (or by any other method) that any DFA for L_i must have $2^{\Omega(s_i)}$ states.

   (d) Give a 2DFA for L_i with $O(s_i)$ states. This shows that 2DFAs are exponentially more powerful than DFAs (in terms of the number of states).

2. Let L be a regular language. Give a 2DFA for the following language. $L' = \{w \mid w \cdot w \in L\}$.

3. Let $\Sigma_2 = \left\{ \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \end{bmatrix} \right\}$. For any word over Σ_2^*, think of the bottom and top rows as a string over $\{0,1\}^*$. For example, if $w = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \end{bmatrix}$, then the top row is a string $0 \cdot 0 \cdot 1$ and the bottom row is a string equal to $0 \cdot 1 \cdot 0$.

   Prove that the following language is not regular.

   $$L = \{w \in \Sigma_2^* \mid \text{bottom row of } w \text{ is reverse of top row of } w\}.$$

4. Let $\Sigma = \{0, 1, +, =\}$. Prove that the following language is not regular.

   $$\text{ADD} = \{x = y + z \mid x, y, z \text{ are binary integers and } x \text{ is a sum of } y, z\}.$$

5. Let L be a regular language. Consider the following language:

   $$\text{MID}_L = \{x \cdot z \mid \exists y \text{ s.t. } |x| = |y| = |z| \ \&\ x \cdot y \cdot z \in L\}.$$

   Prove or disprove that if L is regular then MID_L is also regular.

6. Recall the algorithm we studied in class to find equivalent states in a DFA. Let us call it the table filling algorithm.

   (a) Prove that if an entry corresponding to two states (p, q) stays $--$ (blank)when the table filling algorithm terminates, then the two states are equivalent.

   (b) Let $A = (Q, \Sigma, \delta, q_0, F), A' == (Q', \Sigma, \delta', q_0', F')$ be two DFAs. Assume that $Q \cap Q' = \emptyset$, that is, the two DFAs do not have any state name in common. Form a new DFA $A = (Q'', \Sigma, \delta'', q_0'', F'')$, where $Q'' = Q \cup Q'$, $\delta'' = \delta \cup \delta'$, $q_0'' = q_0$ and $F'' = F \cup F'$. If we run the table filling algorithm on A'' and find that q_0 and q_0' are equivalent then in fact $L(A) = L(A')$.

(c) Show that euivalence of states is transitive, i.e. if p, q are equivalent and q, r are equivalent then p, r are also equivalent.

(d) Let $A = (Q, \Sigma, \delta, q_0, F)$ be a DFA. Let $[q] = \{p \mid p \text{ equivalent to } q\}$. We will call this a block of q. Prove that for each $q \in Q$, q belongs to exactly one block.

(e) Using parts $(b), (c), (d)$ above prove that the minimum DFA for any regular language is unique (up to state relabellings).

[Hint: If there are two minimum DFAs one with fewer states than the other say A, A', then take the union DFA A'' and do state elimination for it. As they both accept the same language, their initial states must be equivalent.]

7. Prove that minimal NFA are not unique.

Tutorial 6

1. Recall the function $T_x : Q \cup \{\bowtie\} \to Q \times \{\perp\}$. Recall also the relation we defined on words using T_x. We said that $x \sim y$ if $T_x = T_y$.

   (a) Show that $\sim$ is an equivalence relation.

   (b) Show that $\sim$ satisfies right congruence.

   (c) Let L be a language accepted by a 2DFA. Show that $\sim$ refines L.

   (d) Show that $\sim$ has finite index.

   (e) Using the parts above, conclude that for any language L accepted by a 2DFA, show that $\sim$ is a Myhill-Nerode relation for L.

2. Given a 2DFA A give a DFA A' such that $L(A) = L(A')$. (You may use the Myhill-Nerode defined in Question 1 above.)

3. Show that the following functions are FST computable.

   (a) Input: $w = a^n$
   Output: $w' = a^{n/2}$

   (b) Input: $w = a^n$
   Output: $w' = a^{2n}$

   (c) Input: $w \in \{0,1\}^* \cdot 1 \cdot \{0,1\}^*$
   Output: $w' \in \{0,1\}^*$ such that $\#_1(w') = \#_1(w) - 1$

   (d) Input: $w \in \{0,1\}^*$
   Output: $w' \in \{0,1\}^*$ such that $w' = 1 \cdot w$

   (e) Input: $w \in \{0,1\}^*$
   Output: $w' \in \{0,1\}^*$ such that $\text{bin}(w) = 2 \cdot \text{bin}(w')$

4. Show that the following function is not FST computable.

   Input: $w \in \{0,1\}^*$
   Output: $w' \in \{0,1\}^*$ such that $w' = w^R$

Tutorial 7

1. Give a NPDAs for the following languages
   (a) $\{0^n 1^m \mid \text{either } n = m \text{ or } 2n = m\}$.
   (b) $\{a^n b^m c^k \mid n, m, k \geq 1, n + k = m\}$
   (c) $\{w \in \{0,1\}^* \mid \#_0(w) = \#_1(w)\}$
   (d) $\{a^n b^m c^k \mid k \neq n + m\}$
   (e) $\{w \mid w \in \Sigma^*, w \text{ has odd length and the middle letter is } a\}$
   (f) $\{w\#x \mid w, x \in \Sigma^*, w^R \text{ is a substring of } x\}$
2. Let $N = (Q, \Sigma, \Gamma, \delta, q_0, \bot, F)$ be an NPDA accepting language L. Show that there exists another NPDA $N' = (Q', \Sigma, \Gamma', \delta', q_0', \bot, F')$ accepting the same language L such that $|\Gamma'| = 2$.
3. Let $N = (Q, \Sigma, \Gamma, \delta, q_0, \bot, F)$ be an NPDA accepting language L. Show that there exists another NPDA $N' = (Q', \Sigma, \Gamma', \delta', q_0', \bot, F')$ accepting the same language L such that in one step N' pushes at most 2 symbols on the stack.
4. Let $L_1, L_2 \subseteq \Sigma^*$. Let $L_1/L_2 = \{x \mid \exists y \in L_2, xy \in L_1\}$. Show that if L_1 is a context-free language and L_2 is a regular language then L_1/L_2 is a context-free language.
5. Show that if L is a context-free language and R is a regular language then $L \cap R$ is a context-free language.
6. Let L, L' be two languages. Let us define $L||L'$ as follows:

   $L||L' := \{x_1 y_1 x_2 y_2 \ldots x_n y_n \mid x_1 x_2 \ldots x_n, y_1 y_2 \ldots y_n \in \Sigma^*, x_1 x_2 \ldots x_n \in L \text{ and } y_1 y_2 \ldots y_n \in L'\}$

   Prove that if L is a context-free language and L' is a regular language then $L||L'$ is a context-free language by explicitly constructing an NPDA for the language.

Tutorial 8

1. Convert the following grammars into Chomsky Normal form.

   (a)
   $$\begin{aligned} S &\to ASB \mid \varepsilon \\ A &\to aAS \mid a \\ B &\to SBS \mid A \mid bb \end{aligned}$$

   (b)
   $$\begin{aligned} S &\to AAA \mid b \\ A &\to Aa \mid B \\ B &\to \varepsilon \end{aligned}$$

   (c)
   $$\begin{aligned} S &\to aAa \mid bBb \mid \varepsilon \\ A &\to C \mid a \\ B &\to C \mid b \\ C &\to CDC \mid \varepsilon \\ D &\to A \mid B \mid ab \end{aligned}$$

2. Give equivalent NPDAs for the following grammars.

   (a)
   $$\begin{aligned} S &\to aAA \\ A &\to aS \mid bS \mid a \end{aligned}$$

   (b)
   $$\begin{aligned} S &\to aSbb \mid T \\ T &\to bTaa \mid S \mid \varepsilon \end{aligned}$$

   (c)
   $$\begin{aligned} S &\to ABS \mid AB \\ A &\to aA \mid a \\ B &\to bA \end{aligned}$$

3. Prove or disprove the following: any context-free language L that does not contain ε, there is a grammar $G = (V, T, P, S)$ such that G generates L and every production rule in G has form $A \to BCD$ or $A \to a$, where $A, B, C, D \in V$ and $a \in T$.

4. Prove that any grammar $G = (V, T, P, S)$ that has only the following types of rules generates a regular language: $A \to aB$ or $A \to a$ or $A \to \varepsilon$, where $A, B \in V$ and $a \in T$.

5. Prove that any grammar $G = (V, T, P, S)$ that has only the following types of rules generates a regular language: $A \to Ba$ or $A \to a$ or $A \to \varepsilon$, where $A, B \in V$ and $a \in T$.

6. A deterministic PDA (DPDA), $D = (Q, \Sigma, \Gamma, \delta, q_0, \bot, F)$ is a PDA such that

   - For any $q \in Q, a \in \Sigma$ or $a = \varepsilon$, and $X \in \Gamma$, $|\delta(q, a, X)| \leq 1$.
   - Moreover, if $\delta(q, a, X)$ is non-empty for some $a \in \Sigma$ then $\delta(q, \varepsilon, X)$ is empty.

   The languages accepted by DPDAs are called deterministic CFLs (DCFLs).

   (a) Show that REG $\subseteq$ DCFL.

   (b) Show that there is a language L that is accepted by a DPDA but is not regular.

   (c) Recall the operator $||$ defined in previous tutorials. Prove or disprove that if L is a DCFL and R is a regular language then $L||R$ is a DCFL.

   (d) Show that there is a language L that is accepted by an NPDA but is not DCFL.

Tutorial 9

1. Let $M = (\{q_0, q_1, q_2, q_f\}, \{0, 1\}, \{0, 1\&\}, \delta, q_0, \{q_f\})$. Describe the behavior of the TM with the following δ transition.

   (a) $\delta(q_0, 0) = (q_1, 0, R)$, $\delta(q_0, 1) = (q_1, 0, R)$, $\delta(q_1, 0) = (q_1, 0, R)$, $\delta(q_1, 1) = (q_2, 1, R)$, $\delta(q_2, 0) = (q_2, 0, R)$, $\delta(q_2, 1) = (q_1, 0, R)$.

   (b) $\delta(q_0, 0) = (q_1, 1, R)$, $\delta(q_1, 1) = (q_2, 0, L)$, $\delta(q_2, 1) = (q_0, 1, R)$, $\delta(q_1, \&) = (q_f, \&, R)$.

2. Give Turing machines (possibly informal discussion as discussed in class) for the following languages. Once you design a TM for one of the parts, you may use that as a subroutine in the subsequent parts of the question. (See for example the description of how to use TMs as subroutines in the textbook by Hopcroft and Ullman.) Assume that the input alphabet is $\Sigma = \{0, 1, \#\}$.

   (a) Given a word $x\#$, where $x \in \{0, 1\}^*$ on the tape, generate $x\#1^{|x|}$ on the tape.

   (b) Given a word $x\#y\#$, where $x, y \in \{0, 1\}^*$ on the tape, generate $x\#y\#1^{max\{|x|,|y|\}}$ on the tape.

   (c) Given a word $x\#$, where $x \in \{0, 1\}^*$ on the tape, generate $x\#x$ on the tape.

   (d) Given a word $x\#$, where $x \in \{0, 1\}^*$ on the tape, generate $x\#x^R$ on the tape.

   (e) Given a word $x\#$, where $x \in \{0, 1\}^*$ on the tape, generate $x^R\#$ on the tape.

   (f) Given a word $x\#y\#$, where $x, y \in \{0, 1\}^*$ on the tape, generate $x\#y\#z$ on the tape, such that $\text{bin}(z) = \text{bin}(x) + \text{bin}(y)$.

3. Give the full formal description (using the state diagram or by describing the δ transition) of a deterministic TM that accepts all the words in language $\{a^n b^n c^n \mid n \geq 0\}$ and rejects everything else.

4. Give the full formal description (using the state diagram or by describing the δ transition) of a deterministic TM that accepts all the words in language $\{a^n \mid n \geq 1$ and n is a power of $2\}$ and rejects everything else.

5. Give an example of a deterministic TM that does not halt (loops forever) on at least some input.

Tutorial 10

1. Give Turing machines (with full formal descriptions) for the following languages. Once you design a TM for one of the parts, you may use that as a subroutine in the subsequent parts of the question. (See for example the description of how to use TMs as subroutines in the textbook by Hopcroft and Ullman.)

   (a) Given a word w on the first tape, copy the word on the second tape and make the first tape completely blank.

   (b) Given a word $w \in \{a, b\}^*$ on the first tape, keep only every alternate letter of w on that tape. That is, suppose $abbab$ is written on the tape, you should finally end up with abb on the same tape. You may design a 2-tape TM for this.

   (c) Given a word $w = w_1 w_2 \dots w_{2n}$ over the alphabet $\{a, b\}$ on the tape, output the word $w_1 w_3 \dots w_{2n-1} w_2 w_4 \dots w_{2n}$ on the same tape. You may design a 2-tape TM for this.

   (d) Given a word $w \in \{a, b, \overline{a}, \overline{b}\}^*$ design a single tape TM that accepts if and only if the following two conditions are satisfied:

   - exactly 3 positions in w come from $\{\overline{a}, \overline{b}\}$ and all the others are from $\{a, b\}$,
   - and the values at those positions are either $\overline{b}\overline{a}\overline{a}$ or $\overline{a}\overline{b}\overline{a}$.

   (e) Given a word $w \in \{a, b, \overline{a}, \overline{b}, \#\}^*$ design a single tape TM that accepts if and only if the word satisfies all the following three conditions:

   - exactly 3 positions in w come from $\{\overline{a}, \overline{b}\}$ and all the others are from $\{a, b, \#\}$,
   - every 4th letter in the word w is $\#$, i.e. $w = x_1 \# x_2 \# \dots x_n \#$, where $x_i = x_{i_1} x_{i_2} x_{i_3}$ (i.e. $|x_i| = 3$),
   - and if a letter with overline appears in a certain block, say x_i, at a position j, then in no other block, say $x_{i'}$ where $i' \neq i$, does the letter with overline appear in that position, i.e. if there exist $i \in [n]$ and $j \in \{1, 2, 3\}$ such that $x_{i_j} \in \{\overline{a}, \overline{b}\}$ then for any $i \neq i'$ $x_{i'_j} \in \{a, b\}$.

   (f) Given a word $w \in \{a, b, \overline{a}, \overline{b}, \#\}^*$ design a single tape TM that does the following:

   - checks that the three conditions in part (1e) above are satisfied,
   - if they are not satisfied then rejects and halts,
   - if they are satisfied then updates the word as follows: (i) if $x_{i_j} = \overline{a}$ then changes $x_{i_j} = a$, (ii) if $\left(x_{i_j} = \overline{b} \text{ AND } i > 1\right)$ then overwrites $x_{(i-1)_j}$ with $\overline{b}$ and x_{i_j} with b, (iii) makes no updates in all the other cases.

   (g) Given 1^n on the first tape, output n in binary on the second tape.

   (h) Given $w \in \{0, 1\}^n$ on the first tape, output the number represented by w in unary on the second tape.

2. Prove that for any 3-tape TM there is an equivalent 1-tape TM. You may use various subparts from the Question 1 above as subroutines.

Tutorial 11

1. Prove that any language accepted by an NPDA is Turing decidable.
2. Prove that the following language is decidable: $\{\langle M\rangle \mid M \text{ does not move left on any input}\}$.
3. Prove that for any non-deterministic TM there is an equivalent deterministic TM.
4. In class we proved that Halt $= \{(M, w) \mid M \text{ halts on } w\}$ is undecidable by giving a reduction (from A_{TM}). Prove that Halt is undecidable by giving a proof by the diagonalization argument.
5. Prove that $\overline{\text{Halt}}$ is not Turing recognizable.
6. A function $f : \Sigma^* \to \Gamma^*$ is called a **computable function** if there is a single tape TM M such that on every input $w \in \Sigma^*$ it halts with exactly $f(w)$ on the tape.

   We say that a language $L \subseteq \Sigma^*$ is **reducible** to language $L' \subseteq \Gamma^*$, which we denote by $L \leq L'$, if there is a computable function $f : \Sigma^* \to \Gamma^*$, where for every w, $w \in L \Leftrightarrow f(w) \in L'$. The function f of this form is called a **reduction**. Prove or disprove the following statements:

   (a) If $L \leq L'$ and L' is decidable then L is also decidable.
   (b) If $L \leq L'$ and L is undecidable then L' is also undecidable.
   (c) If $L \leq L'$ and L' is a CFL then so is L.
   (d) If L is Turing recognizable and $L \leq \overline{L}$ then L is decidable.
   (e) L is Turing recognizable if and only if $A \leq A_{TM}$.
   (f) There is a laguage L such that L is undecidable and $L \leq \overline{L}$.
7. In this problem we will define the Post Correspondence Problem (PCP) and prove that it is undecidable. This problem and its variants are very useful in proving undecidibility of many interesting languages.

   Definition 0.1. *A domino over Σ consists of two strings $\left[\frac{u}{\ell}\right]$, where both $u, v \in \Sigma^*$. Given a domino $w = \left[\frac{u}{\ell}\right]$, u is called the upper(w) and ℓ is called the lower(w).*

   Definition 0.2. *Given a collection of dominos $d_1, d_2, \ldots, d_t$ over Σ. For any n $x_1x_2 \ldots x_n \in \{d_1, d_2, \ldots, d_t\}^n$ is said to be a match if upper(x_1) $\cdot$ upper(x_2) $\ldots \cdot$ upper(x_n) equals lower(x_1) $\cdot$ lower(x_2) $\ldots \cdot$ lower(x_n).*

   Suppose $d_1 = \left[\frac{a}{ab}\right]$ and $d_2 = \left[\frac{ba}{a}\right]$.

   (a) Give an $x \in \{d_1, d_2\}^*$ such that x is a match.
   (b) Give an $x \in \{d_1, d_2\}^*$ such that x is not a match.

   Definition 0.3. *The Post Correspondence Problem can be stated as follows:*

   Given: *a collection of dominos $w_1, w_2, \ldots, w_t$*
   Output: *$x \in w_1 \cdot \{w_1, w_2, \ldots, w_t\}^*$ such that x is a match.*

   Prove that PCP is undecidable by filling in the details below. The overall proof strategy is as follows: we will create dominos over Γ to encode one valid step of TM M on input $w \in \Sigma^*$. We will then prove that there is a match if and only if M accepts w.

(a) Let d_1 be the domino which checks the initial configuration.

[Hint: $d_1 = \left[\cdot \frac{\not\in}{\not\in q_0 w_1 w_2 \cdots w_n \not\in}\right]$]

(b) Design a domino d_2 to check the correctness of the following move: $(p, b, R) \in \delta(q, a)$.

[Hint: Recall the notion of a configuration and how it changes due to such a transition. Encode the upperd_2 so that it encodes the relevant information regarding the configuration before this transition takes place and encode the lowerd_2 so that it encodes the relevant information regarding the configuration after this transition takes place.]

(c) Design a domino d_3 to check the correctness of the following move: $(p, b, L) \in \delta(q, a)$.

(d) Add a domino d_4 which allows parts of the tape to stay the same (wherever the tape head is not present.)

(e) Add a dominos to handle halting after reaching q_{acc}.

(f) Finally, add a domino to handle parts of the tape which remain after the TM halts.

(g) Let D denote the set of dominos which we designed above. Show that if M accepts w then there exists a string $x \in D^*$ such that it is a match.

(h) Show that if M does not accept w then there is no match for any string over D^*.

(i) Observe that above parts together show that PCP is undecidable.

Tutorial 12

1. Give a $O(n \log n)$ time TM for the following problem.

   Given: 1^n
   Output: binary representation of n

2. Give a $O(n^2)$ time TM for the following problem.

   Given: 1^n
   Output: binary representation of n^2

3. Show that SAT, 3Color are in NP.

4. Show that Min, Verify-SAT are in L.

5. Show that given a directed graph G, whether it has a cycle or not is in NL.

6. Specify the relationships between the following complexity classes: (containment, strict containment, incomparable)

   (a) P. NP, EXP, NEXP, NL
   (b) SPACE(n), TIME(n), TIME(2^n), SPACE(2^n), SPACE($\log n$)

7. State true or false with justification: Either P is strictly contained in NP or EXP properly contains NP.