CS181-Homework2

February 2018

1

1.1 Exercise

1. a For this problem, I will use proof by contradiction to prove, Let assume that shuffle(L1,L2) and shuffle (L1,$\overline{L2}$) are both regular,and we will use the union closure property such that L_{union} is regular where L_{union} = shuffle(L1,L2) $\cup$ shuffle (L1,$\overline{L2}$), we know that shuffle L1 with L2 or $\overline{L2}$ is just shuffling with any character in Σ. In homework 1, we know if L is regular then L_{alt} is also regular. In our case, L is just our L_{union}, and L_{alt} is just $L_{union_{alt}}$, which consist of L1. L1 regular base on the homework 1; however, it contradicts with the given problem that L1 is not regular, thus, we get contradiction here.

2. b) proof idea: we want to show that if L1 and L2 are regular, then shuffle(L1, L2) is also regular. Because of L1 and L2 are regular, then we need some finite automation M1 recognizes L1, and some finite automation M2 recognizes L2. In order to prove that shuffle(L1, L2) is regular, we also need to construct a finite automation M that recognizes shuffle(L1, L2), we will need to have this M to also accept its input when either M1 and M2 would accept it in order to recognizes shuffle(L1, L2), to do this, we need to keep track a pair of states. If M1 has k1 states, and M2 has k2 states, the pair of states will consists of state from both M1 and m2, which it the product of k1 x k2.

Let M1 recognize L1, such that

$$M1 = \{Q_1, \Sigma_1, \delta_1, q_{1_0}, F_1\}$$

Let M2 recognize L2, such that

$$M2 = \{Q_2, \Sigma_2, \delta_2, q_{2_0}, F_2\}$$

Construct our M to recognizes our shuffle(L1,L2) where M = {Q, Σ, δ, q_0, F }

$$Q = Q_1 X Q_2 X \{k1, k2\}$$

$$q_0 = \{q_{1_0}, q_{2_0}, k1\}$$

in order to have this machine keep track on both M1 and M2 states, our transition function will be:
$\delta((q_{1_i}, q_{2_j}, n), a) = (q_{1_k}, q_{2_j}, k2)$ where,$\delta_1(q_i, k) = q_k$, $n = k1$
$\delta((q_{1_i}, q_{2_j}, n), a) = (q_{1_j}, q_{2_k}, k1)$ where,$\delta_2(q_j, k) = q_k$, $n = k2$
the set of accepting states F:

$$F = \{(q_{1_i}, q_{2_j}, k1) | (q_{1_i} \in F1, q_{2_j} \in F2)\}$$

According to the book, the definition of a NFA accepts a word w if $w = y1...ym$, where y_i is a member of Σ_ϵ and a sequence of states $(r_0, ...r_m)$ exists in $\mathcal{Q}$.

1. $r_0 = q_0$
2. $r_{i+1} \in \delta r_i, y_{y+1}$
3. $r_m \in F$

let's assume our M1, M2 accept input w_1, w_2 respectively, thus, $w_1 = y1_1, y1_2, y1_3....y1_m$ with our states $r1_0, r1_1...r1_m$, similarly to our $w_2 = y2_1, y2_2, y2_3....y2_m$ with our states $r2_0, r2_1...r2_m$.by using our transition function, assume that we have $w = y1_1, y2_1, y1_2, y2_2...$ we will have something like this $(r1_0, r2_0, k1), (r1_1, r2_1, k2), (r1_2, r2_2, k1), (r1_3, r2_3, k2)$ and so on. Thus, this will meet the definition of a NFA accepts a word w, thus M will accept w.

1.2 Exercise

2.a Let say that all strings in the language L2 fall into two cases, case 1 is when $i = 0$ and case 2 when i != 0. We need to show that both cases satisfy our Pumping Lemma and it can give us any length $p' >= 2$ from L1.

case 1: let set $i = 0$, then we will only get b^p where p is prime and it is greater or equal to our p'. In this case, our string $w = xyz$ where $x = \epsilon$, $y = b$, z = b.. the only criteria we need to prove in order to meet our pumping lemma is $xy^iz \in L2$. This is true because the given $b*$ is a subset of L2. Thus, this string will still be in our language.

case 2: let set $i! = 0$, then our string w will consist of some numbers of a and some numbers of b. Since $|w| >= p'$, we will have $x = \epsilon$, $y = a$, z = other chars in this case. The only criteria we need to prove in order to meet our pumping lemma is $xy^iz \in L2$. This is true, because we don't change the number of b when we pump up or down to construct the new string and the number will still remain prime. The only thing we can change is the single a in y, we can only pump up or down where the $i >= 0$ in this case. w will still be in our L2 language.

2.b Let M = {Q, Σ, δ, q_0, F } be a DFA that recognizes L. We assign the pumping length p to be the number of states of M. We show that any string s in L of length at least p may be broken into the three pieces xyz satisfying our three condition. If s in L has length at least p, consider the sequence of states that M goes through when computing with input s. Let says it starts with q1 the start state, then goes to say, q3, then q20, then q9 and so on, until it reaches the end of s in the state q13, with s in L, we know that M accepts s, so q13 in this case is an accept state.

Let n be the length of s, the sequence of states, q1,q3,q20,q9......q13 has length n+1, because we say that n is at least p we know that n+1 is greater than p , the number of state of M, therefore, the sequence must contains a repeat state by pigeonhole principle.

2.c we can prove this by contradiction, let assume that L2 is regular, so if L2 is regular, then we guaranteed that L2 meet our Pumping Lemma. We will have the length denote as p'.

Let define p* = (prime number which greater or equal than p'), then we can construct our string $w = a^{p^*}b^{p^*}$, for $w = xyz$, this will satisfy constraint given that $|y| >= p$, let look at the partition for w, $x = a^{p^*}, y = b^{p^*}$ and $c = \epsilon$. As we discussed in class,L_{prime} is not a regular, thus in our case, $y = b^{p^*}$ where p* = (prime number which greater or equal than p'), this w in this case can't be regular, thus it contradicts with our assumption that L2 is regular.

1.3 Exercise

Let consider the language $L = 0 * 21*$. In this problem, our language L is a regular, we will construct a very simple DFA M. In order to construct a string w in L that correspond w' in $L_{\frac{1}{3}-\frac{1}{3}}$, ultimately, we want to use one of our closure properties union along with proof by contradiction to prove that if L is regular, then $L_{\frac{1}{3}-\frac{1}{3}}$ is not regular. We will consider the following

let's consider the number of 0s in w and the number of 1s in $w = 3n + 2$ where n is greater and equal to 1.

in this pattern, we will consider three parts where w in L correspond to w' in $L_{\frac{1}{3}-\frac{1}{3}}$.

part 1: let consider 2 fall into the begin part then we will have
$w' = 0^k 21^{2n+1-k}$ where $k <= n$
part 2 : let consider 2 fall into the middle part then we don't need to consider the middle part, because it will just add to the loop: $w = 0^{n+1}1^{n+1}$
part 3 : similarly to the part 1 except that now fall into the ending part, then we will have something like this: $w' = 0^{2n+1-k}21^k$ where $k <= n$

Now, we can use proof by contradiction, let assume that if L is regular, then $L_{\frac{1}{3}-\frac{1}{3}}$ is regular, we already know that our machine M recognizes L, so L is regular. So if $L_{\frac{1}{3}-\frac{1}{3}}$ is regular then we can use our union closure property such that

$$L \cup L_{\frac{1}{3}-\frac{1}{3}} = 0*21 \cup 0^{n+1}1^{n+1}$$

We know that $L_{\frac{1}{3}-\frac{1}{3}}$ is regular, then must be the case that three cases we have above are also regular. In the class, we proved that $L = x$ where $x = 0^n1^n$ is not regular by our Pumping Lemma. We can also prove $0^{n+1}1^{n+1}$ is not regular by Pumping Lemma, because in our case, we just the numbers of 0s and 1s will still be the same. Thus, it contradicts we our assumption that $L_{\frac{1}{3}-\frac{1}{3}}$ is regular.