Theory of Computation '23
Problem Set 2

Notations. Let $\Sigma = \{a, b\}$. For $w \in \Sigma^\star$, 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.

Problem 1. Let L_1, L_2 be two regular languages. Show that $L1 \setminus L2 = \{w | w \in L_1 \text{ and } w \notin L_2\}$ is also regular. **Solution.** The set $L_1 \setminus L_2$ can be written as $L_1 \cap \bar{L_2}$ (if you do not know this, then you must brush up basic set theory). Now, we know that regular languages are closed under complementing (Tutorial 1). So $\bar{L_2}$ is regular. Further, we can use closure under intersection to claim the result.

Problem 2. Give an NFA for the following regular expressions. (You may simplify the expression as much as possible using one of the algebraic laws listed in Lecture 9 of the Kozen book - equations 9.1-9.13)

1. $(aa^\star + bb^\star)^\star$

   Solution. Observe that the above reg-ex can actually be simplifies to just $(a + b)^\star$. Hence the NFA is trivial now.

2. $(ab + ba) \cdot (ab + ba) \cdot (ab + ba)$ There is no point in simplifying this one. Best is to just build an NFA for $ab + ba$ using the standard technique - first design an NFA for ab, ba separately and applu union closure. Then just concatenate three copies of that.

Problem 4. Write the regular expressions corresponding to the following languages.

1. $L = \{\#_a(w) = 1(\mod 2)\}$.

   Solution. So this one wants you to find a reg-ex for strings that contain an odd number of a's. A nice and standard technique to solve these problems is to write down examples and break down the string in to patterns that are going to be repeated. In this case, since you need an odd number of a's (that is numbers of the form $2k + 1$, for $k \geq 0$), observe that any string with this property can be broken down as follows - begin with zero or more b's, followed by an a (this takes care of the $+1$). Now the following pattern repeats - zero or more b's followed by another a, again followed by zero or more b's and finally an a (this takes care of the $2k$ part). Finally there could be bunch of b's at the end. Hence, now it is easy to see that one possible reg-ex for L is

   $$(b^\star a)(b^\star a b^\star a)^\star b^\star$$

   Again, the solution itself is not so important. What you should remember is the technique. Also note that there could be simpler regular expressions for the above.

2. $L = \{$every other letter in w is $a\}$

   Solution. This one is so easy I won't even bother to write.

3. $L = \{w$ contains an odd number of a's and an even number of b's$\}$

   Solution. This is a tricky one. Here is one possible solution. Observe that any string with the above property can broken in to three possible parts : the first part contains an even number of a's and b's, the second part is just a single a (this makes the number of a's odd) followed by a third part which has zero or more occurrences of bb. Now the problem reduces to figuring out the first part, that is reg-ex for strings with even number of a's and b's. This is slightly simpler than the original task. Here is an observation, any such string can be thought of as zero or more repetition of one of the following patterns - $aa, bb, abba, baab, abab, baba$ (one can formally prove this but let us just be convinced for now). Now we can write the final reg-ex

   $$(aa + bb + abba + baab + abab + baba)^\star a(bb)^\star$$

   which can be simplifed to

   $$(aa + bb + (ab + ba)(ab + ba))^\star a(bb)^\star$$

   (Disclaimer : I am not fully convinced about the correctness of the above to be honest. I will double check and confirm).

Problem 4 Construct a DFA that accepts strings which are the binary representations of numbers which are 0 (mod 3). Give a regular expression corresponding to this DFA using the recursive (inductive) construction done in class (given in Kozen book).