1 Concept Review

In the past weeks, we have examined the finite automaton, a simple computational model with limited memory. We proved that DFAs, NFAs, and regular expressions are equal in computing power and recognize the regular languages. We also showed that the regular languages are closed under union, concatenation, Kleene Star, intersection, difference, complement, and reversal. We then used a counting argument to show that there are indeed languages which are non-regular.

This week in section we will become a little more comfortable with these topics by working with regular expressions, making arguments about countability, and exploring some closure properties of regular languages.

2 Exercises

Exercise 2.1. In Problem Set 1 we proved that regular languages were closed under reversal, using NFAs. Prove that fact using regular expressions.

Exercise 2.2. Describe in plain English the language represented by the following regular expressions.
   
   (a) $a^* \cup b^*$
   
   (b) $(aaa)^*$
   
   (c) $\Sigma^*a\Sigma^*b\Sigma^*a\Sigma^*$

Exercise 2.3. Using the procedure outlined in class, convert the regular expression $(((aa)^*(bb)) \cup ab)^*$ to an equivalent NFA.

Exercise 2.4. Let $L$ be a language over the alphabet $\Sigma = \{a, b\}$. Define $\text{PigLatin}(L) = \{w\sigma : \sigma \in \Sigma, w \in \Sigma^*, \sigma w \in L\}$. Informally, $\text{PigLatin}(L)$ is the language containing all strings in $L$ except that each string has had its first character moved to its end. (For example, $\text{PigLatin}(\{abc, a, aab\}) = \{bca, a, aba\}$.)
Show that if $L$ is regular, then $\text{PigLatin}(L)$ is regular. Specifically, given a DFA for $L$, show how to construct an NFA for $\text{PigLatin}(L)$. (Your proof for this problem should involve finite automata and not regular expressions.)

Exercise 2.5. Prove or disprove the following statements about regular expressions:

1. $L((R \cup S)^*) = L(R^* \cup S^*)$
2. $L((RS \cup R)^*R) = L(R(SR \cup R)^*)$
3. $L((RS \cup R)^*RS) = L((RR^*S)^*)$