# CSC236 Winter 2020
# Assignment #3: formal languages
# Solutions

1. grep and many other software implementations of regular expressions include the question mark , '?', as a special symbol which marks the preceding expression as optional. For example, the regular expression `dog(gy)?` matches the strings 'dog' and 'doggy'.

   Let $\mathcal{REQ}$ be an extension of our familiar language of regular expressions with the question mark operator added. We will formally define the set $\mathcal{REQ}$ by extending the definition of $\mathcal{RE}$ (definition 7.6 in the Vassos course notes) to add the following induction step: If $R \in \mathcal{REQ}$, then $(R)? \in \mathcal{REQ}$.

   (a) Definition 7.7 in the Vassos course notes is a recursive definition of the language denoted by a regular expression $R \in \mathcal{RE}$. Give an extended version of this definition for $\mathcal{REQ}$.

   **Solution:**
   We extend definition 7.7 by adding the following to the induction step: $\mathcal{L}((S)?) = \{\varepsilon\} \cup \mathcal{L}(S)$.

   (b) Show that $\mathcal{REQ}$ has no more expressive power than $\mathcal{RE}$, by proving the following statement: $\forall R_1 \in \mathcal{REQ}, \exists R_2 \in \mathcal{RE}, \mathcal{L}(R_2) = \mathcal{L}(R_1)$. Your proof should use structural induction.

   **Solution:**
   Define $P(R) : \exists R_2 \in \mathcal{RE}, \mathcal{L}(R) = \mathcal{L}(R_2)$.
   I will prove $\forall R \in \mathcal{REQ}, P(R)$ by structural induction.
   The base cases of $\emptyset$, $\varepsilon$, and $a \in \Sigma$ are trivially true, since these expressions denote the same languages in $\mathcal{RE}$ as they do in $\mathcal{REQ}$. (i.e. the equivalent expression is identical.)
   Let $S, T \in \mathcal{REQ}$, and assume $P(S) \wedge P(T)$. Therefore, there exist REs in $\mathcal{RE}$, call them $R_1, R_2$, such that $\mathcal{L}(R_1) = \mathcal{L}(S)$ and $\mathcal{L}(R_2) = \mathcal{L}(T)$. Therefore $(R_1 + R_2) \in \mathcal{RE}$ is equivalent to $(S + T) \in \mathcal{REQ}$. So $P((S + T))$ holds. Similar arguments hold for concatenation and Kleene star.

Lastly, consider the RE $(S)?$. We know from part (a) that $\mathcal{L}((S)?) = \{\varepsilon\} \cup \mathcal{L}(S)$.

The RE $(R_1 + \varepsilon) \in RE$ denotes the same language, by the definition of $+$. So $P((S)?)$ holds, concluding the induction.

2. Given a DFSA $M = (Q, \Sigma, \delta, s, F)$, we will say that $M$ is **frumious** if the following is true:

   $\forall a \in \Sigma, \exists q_1 \in Q, \forall q_2 \in Q, \delta(q_2, a) = q_1$

   (a) Give a short English description of what it means for a DFSA to be frumious.

   **Solution:**
   A DFSA is frumious if and only if its current state is determined entirely by the last character it saw (assuming it has seen a non-zero number of characters).

   (b) If $M$ is frumious, what can we say about the language accepted by $M$, $\mathcal{L}(M)$?

   **Solution:**
   Let $S_a = \{x \in \Sigma^* \mid \exists y \in \Sigma^*, x = ya\}$ for arbitrary $a \in \Sigma$. Then for every $a \in \Sigma$, either $S_a \subseteq \mathcal{L}(M)$, or $S_a \cap \mathcal{L}(M) = \emptyset$.

   In other words, for every symbol $a$, $M$ either accepts all strings ending in $a$, or none of them.

   (c) How many distinct languages over the alphabet $\{0, 1\}$ can be recognized by frumious DFSAs? Briefly explain your answer.

   **Solution:**
   Eight languages. We can recognize each of the following four languages:

   - The empty language
   - Strings ending with 0
   - Strings ending with 1
   - Strings ending with 0 or 1

   And for each of these four, we can also recognize the union of that language with $\{\varepsilon\}$.

3. Suppose $L$ is an infinite regular language. Does it follow that there exists a finite language $S$ such that $L = SS^*$? If yes, prove it. If no, find a counterexample language $L$ and prove that it cannot be formed this way.

**Solution:**

A counterexample is the language $L = \{a\}\{b\}^*$. Suppose for sake of contradiction that $S$ is a finite language such that $SS^* = L$.

$ab$ is a string in $L$, so it must be in $SS^*$. Let $x \in S, y \in S^*$ such that $x \circ y = ab$.

Case 1: $x = \varepsilon$. Then $\varepsilon \in S$, and therefore $\varepsilon \in SS^*$. But the empty string is not in L. $\Rightarrow\Leftarrow$

Case 2: $x = a, y = b$. It follows that $\{a, b\} \subseteq S$. But then $ba \in SS^*$, but $ba$ is not a string in $L$. $\Rightarrow\Leftarrow$

Case 3: $x = ab, y = \varepsilon$. Then $ab \in S$. It follows that $abab \in SS^*$, but $abab$ is not in $L$. $\Rightarrow\Leftarrow$

These are the only concatenations that can form $ab$, and in each case, we are led to a contradiction, so no such $S$ exists.