COSC341 TUTORIAL 2

The theme of this tutorial is to explore the definition of strings a bit, and to understand the relationships between regular grammars and DFAs.

Recall that a string of length n over an alphabet Σ is actually a function from the set $\{0, 1, 2, \ldots, n-1\}$ (which I'll denote $[n]$) to Σ and that the set of all strings (of any length, including the empty string) over Σ is denoted Σ^*. The notation for the length of a given string w is $|w|$.

1. *If Σ has k elements how many different strings of length n are there over Σ?*

   There are n times we get to choose freely from among k things and these choices are independent, so the number of ways to do is $k \times k \times \cdots \times k$ where there are n factors of k, i.e., k^n.

2. *If $w \in \Sigma^*$ and $m, i + j \leq |w|$ how are:*

   - *the prefix of w of length m,*
   - *the suffix of w of length m, and*
   - *the substring of w starting at position i of length j*

   defined?

   - The prefix, p of w of length m is the map $p : \{0, 1, \ldots, m-1\} \to \Sigma$ where $p(t) = w(t)$ for all $0 \leqslant t < m$.
   - The suffix, s of w of length m is the map $s : \{0, 1, \ldots, m-1\} \to \Sigma$ where $s(t) = w(t + n - m)$ for all $0 \leqslant t < m$.
   - The substring, m of w of length j starting at position i is the map $m : \{0, 1, \ldots, j-1\} \to \Sigma$ where $m(t) = w(i + t)$ for all $0 \leqslant t < j$.

3. *If $w, v \in \Sigma^*$ how are:*

   - *the result of prepending $a \in \Sigma$ to w,*
   - *the result of appending $a \in \Sigma$ to w, and*
   - *the result of concatenating w and v*

   defined?

   - The result of prepending $a \in \Sigma$ to w is the map u where $u(0) = a$, and $u(t) = w(t-1)$ for $1 \leqslant t \leqslant |w|$ (note not $< |w|$ since we're adding one to the length).

- The result of appending $a \in \Sigma$ to w is the map u where $u(t) = w(t)$ for $0 \leqslant t < |w|$ and $u(|w|) = a$.
- The result of concatenting w and v is the map $w \cdot v$ defined for $0 \leqslant t < |w| + |v|$ where:

$$(w \cdot v)(t) = \begin{cases} w(t) & \text{for } 0 \leqslant t < |w|, \\ v(t - |w|) & \text{for } |w| \leqslant t < |w| + |v|. \end{cases}$$

4. *Determine a regular grammar for the language of all strings containing an even number of as (over $\Sigma = \{a, b\}$).*

$$\begin{aligned} S &\to \epsilon \,|\, aO \,|\, bS \\ O &\to aS \,|\, bO. \end{aligned}$$

Basically, S stands for "even number of a's so far" and O for "odd number of a's so far".

5. *Let $\mathbf{A}$ and $\mathbf{B}$ be DFAs over the same alphabet, Σ. Can you describe DFAs that accept:*

- *The complement of the language $L(\mathbf{A})$, i.e., the set of all strings not belonging to $L(\mathbf{A})$,*
- *The intersection of $L(\mathbf{A})$ and $L(\mathbf{B})$, i.e., the set of all strings belonging to both of $L(\mathbf{A})$ and $L(\mathbf{B})$, and*
- *The union of $L(\mathbf{A})$ and $L(\mathbf{B})$, i.e., the set of all strings belonging to at least one of $L(\mathbf{A})$ and $L(\mathbf{B})$.*

For the first part, just interchange all the accepting and non-accepting states. For the second and third part, we can create a product automaton. Consider a new DFA whose states are members of the cross product $\mathbf{A} \times \mathbf{B}$ (a set of all ordered pairs (A, B) where A is from $\mathbf{A}$ and B is from $\mathbf{B}$). The transition from a state (S, T) on letter a is to the state (S', T') where in $\mathbf{A}$, the transition on a is from S is to S' and in $\mathbf{B}$ the transition on a is from T to T'. The start state is the pair consisting of the start states of the respective machines. Finally for the second part the accepting states are all those pairs where both states are accepting, while for the third it's those where either state is accepting. See `https://www.geeksforgeeks.org/cross-product-operation-in-dfa/` for an example of the intersection case.

6. *Repeat the previous question for the languages $L(G)$ and $L(H)$ associated to two right-regular grammars over Σ.*

   Union is "easy" since we can assume that, except for the start symbol the two grammars have no non-terminal symbols in common and then just write down all the rules for both grammars. A little care is needed here if we have productions that lead back to the start symbol in either grammar - in that case we need two "fresh" copies of the start symbol appropriate to each grammar and use those instead.

   Because of the inherent non-determinism of grammars (where we might have multiple rules associated with a particular letter) it's not at all clear how one could realise the intersection of two languages or the complement of one.