Exercises on Regular Expressions and Finite Automata

Throughout this exercise sheet, assume that the alphabet is $\Sigma = \{a, b\}$.

1. Recall that (basic) regular expressions are formed from 0 (matches nothing), 1 (matches just the empty word), alternation $(r + s)$, concatenation (rs), and iteration (r^*). Write regular expressions that accept exactly the following languages:

   (a) words that consist of an even number of a's followed by an odd number of b's,

   (b) words of an even length,

   (c) words that contain exactly three a's,

   (d) words that contain no more than three a's,

   (e) words that do not contain two consecutive a's,

   (f) words where every a is immediately followed by a b, and

   (g) words that do not end with ba.

2. Using Thompson's algorithm or otherwise, convert the regular expression $(b+ab)^*(1+a+aa)(b+ba)^*$ into an NFA.

3. Consider an NFA with three states, named s_1, s_2, and s_3. The start state is s_1 and the accepting state is s_3. The a-transitions are $s_2 \to s_1$ and $s_2 \to s_3$. The b-transitions are $s_1 \to s_2$, $s_2 \to s_3$, and $s_3 \to s_1$. There is an ϵ-transition from s_1 to s_3.

   (a) Draw this NFA.

   (b) Using the subset construction or otherwise, convert this NFA into a DFA.

   (c) DFAs with fewer states can be more efficiently simulated. Can you simplify your DFA into a DFA that accepts the same language but requires fewer states?

   (d) Describe (in simple English) the language that your DFA accepts.

4. Let us write Σ^* for the language that contains every word made up from characters in the alphabet Σ. This is sometimes called the *universal language over* Σ. Draw a DFA that accepts the language Σ^*.

5. Suppose you are given an NFA that accepts a language L_1, and another NFA that accepts a language L_2. Explain how to construct a new NFA that accepts the language $L_1 \cup L_2$ (that is, all words that are in *either* language).

6. For any language L, we shall define its *complement* as $\overline{L} = \Sigma^* \setminus L$; that is, the set of all words that are not in L.

   (a) Draw a DFA that accepts the language $\{a, ba, ab\}$.

   (b) Now draw a DFA that accepts the complement of that language.

   (c) Suppose you are given a DFA that accepts a language L. Explain how to construct a new DFA that accepts the language $\overline{L}$.

7. Suppose you are given an NFA that accepts a language L_1, and another NFA that accepts a language L_2. Explain how to construct a new NFA that accepts the language $L_1 \cap L_2$ (that is, all words that are in *both* languages). *[Hint: bear De Morgan's law in mind!]*

8. For any language L, we shall write L^{-1} for the *reverse* language of L. L^{-1} accepts the same words as L, but each word has its sequence of characters reversed. For instance, ab is accepted by L^{-1} if and only if ba is accepted by L.

   (a) Draw a DFA that accepts $\{ba, baa\}$.

   (b) Draw an NFA that accepts the reverse language of $\{ba, baa\}$.

   (c) Suppose you are given a DFA that accepts the language L. Explain how to construct an NFA that accepts the language L^{-1}.