In this assignment you will give constructive proofs of closure properties of regular languages, and implement them in haskell, similarly to the unionDFA, complementDFA and composeFST operations presented in class.

For each problem, it is advisable that you first write down a mathematical description of the transformation. Once you have worked out the math, you should code it in haskell starting from the template files HW11.hs, HW12.hs and HW13.hs provided on the course webpage, and replacing the ... as directed by the comments. Submit the three files using the bundleHW1 command on ieng6 by the due date.

1 Intersection

Prove that regular languages are closed under intersection by giving a transformation that on input two DFAs for languages $L_1$ and $L_2$, produces a DFA for $L_1 \cap L_2$. Submit your solution as HW11.hs.

2 Prefix removal

For any language $L \subseteq \Sigma^*$ and string $w \in \Sigma^*$, let $w \setminus L = \{ u \in \Sigma^* \mid wu \in L \}$, i.e., the set of all strings in $L$ that begin in $w$, but with the prefix $w$ removed. For example, $ab \setminus \{abaa, a, baba, abab\} = \{aa, ab\}$. Prove that for any $L$ and $w$, if $L$ is regular then also $w \setminus L$ is regular by giving a transformation $T_a(M)$ that on input a DFA $M$ with alphabet $\Sigma$ and a string $w \in \Sigma^*$, outputs a DFA for the language $w \setminus L(M)$. Submit your solution as HW12.hs.

3 Combining FST and DFA

Let $M$ be a DFA and $T$ an FST. Prove that the language

$$L(M \circ T) = \{ w \colon M(T(w)) \text{ accepts} \}$$

is regular by giving a transformation that on input $M$ and $T$, outputs a DFA for the language $L(M \circ T)$. Submit your solution as HW13.hs. You can test your solution by designing a DFA that rejects all strings containing consecutive zeros, composing it with the “nodup0FST” transducer, and checking if the resulting DFA does not accept any string.