CSE 105: Automata and Computability Theory **Fall 2015**

Homework #2

Due: Monday, October 19th, 2015, 10:00 AM

All solutions should be submitted using the bundleHW2 command on ieng6. Include also a collaboration disclosure as done in HW1.

Problem 1 (DFA design) For each of the following languages, design a DFA using JFLAP, and submit your solution as a file `HW21a.jff`, `HW21b.jff`, etc. It is highly advisable that you test your automata using JFLAP and/or haskell to make sure they are valid DFAs and they recognize the correct language. All languages are over the alphabet $\{a, b\}$.

(a) The set of all strings that begin with "bbab"

(d) The set of all strings that contain the substring "baa" (the substring should occur as a sequence of contiguous characters)

(c) The set of all strings that contain an even number of "a"s and and odd number of "b"s.

(d) The set of all strings that contain between 2 and 4 "b"s.

Problem 2 (NFA design) Same as problem 1, but this time you should design an NFA. For full credit, give an NFA with the smallest possible number of states. Submit your solution as files `HW22a.jff`, `HW22b.jff`, etc.

(a) The set of all strings such that the 4th chacarter is an "a"

(b) The set of all strings that end with "ba"

(c) The set of all strings such that the 4th chacarter *from the end* is a "b".

(d) The set of all strings that contain the substring "baabb"

Problem 3 (Closure properties) Theorem 1.25 in the textbook proves that regular languages are closed under union. The proof is constructive, i.e., it gives an algorithm that on input two DFAs for languages L_1 and L_2, produces a DFA for $L_1 \cup L_2$. In the class notes on Haskell, you have also seen that the proof is immediately translated into a working computer program that implements the transformation. In this problem you are asked to prove and implement some similar closure properties of regular languages.

(a) 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 $L_1 \cap L_2$. (Assume all languages are over the same alphabet.)

(b) For any language $L \subseteq \Sigma^*$ and symbol $a \in \Sigma$, let $\text{skipLast}(a, L) = \{w \in \Sigma^* \mid wa \in L\}$, i.e., the set of strings in L that end in a, but with that last a removed. Prove that for any L and a, if L is regular then also $\text{skipLast}(a, L)$ is regular by giving a transformation that on input a DFA M with alphabet Σ and a symbol $a \in \Sigma$, outputs a DFA for the language $\text{skipLast}(a, L(M))$.

(c) Similar to part (c), but for the language $\text{skipFirst}(a, L) = \{w \in \Sigma^* \mid aw \in L\}$, i.e., the set of strings in L that start in a, but with that first a removed.

For each part, your solution should consists of a mathematical description of the transformation and a brief explanation of how/why it works, and a haskell program implementing the transformation. The mathematical proof should be typeset and submitted as a pdf file `HW23x.pdf` (for $x = a, b, c$). For the haskell part, start from the template files `HW23x.hs` provided on the course webpage, and modify them as directed.