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

Homework #3

Due: Friday, October 30th, 2015, 10:00 PM

All solutions should be submitted using the bundleHW3 command on ieng6, and include a collaboration disclosure as in previuos homeworks. Each of Problems 1 and 2 carries 25% of the points for this problem set. Problem 3 gives the remaining 50%.

Fix an alphabet Σ. For any string $w \in \Sigma^*$ let $even(w)$ be the string obtained by taking the symbols of w at even positions, and $odd(w)$ the string obtained by taking the symbols at odd positions. E.g., $even(abbabaa) = baa$ and $odd(abbabaa) = abba$. Formally, the functions even and odd can be defined by mutual recursion as

$$\begin{aligned} even(\epsilon) &= \epsilon \\ even(aw) &= odd(w) \\ odd(\epsilon) &= \epsilon \\ odd(aw) &= a \cdot even(w) \end{aligned}$$

where $a \in \Sigma$ and $w \in \Sigma^*$. As usual, the functions are extended to languages by

$$\begin{aligned} even(L) &= \{even(w) \colon w \in L\} \\ odd(L) &= \{odd(w) \colon w \in L\} \end{aligned}$$

Problem 1 (Closure Properties) Prove that for any regular language L, $odd(L)$ is also regular by giving a transformation `odd` that on input a DFA for L produces an NFA for $odd(L)$. As usual, 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 `HW31.pdf`. For the haskell part, start from the template file `HW31.hs` provided on the course webpage, and modify it as directed.

Hint: In haskell, if you want to define an NFA with twice as many states as the original automaton, you can use type $(st, Bool)$ *to map each original state* q *to a pair of states* $(q, True)$ *and* $(q, False)$.

Problem 2 (More Closure Properties) Prove that for any regular language L, $even(L)$ is also regular by giving a transformation `even` that on input a DFA for L produces an NFA for $even(L)$. As usual, 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 `HW32.pdf`. For the haskell part, start from the template file `HW32.hs` provided on the course webpage, and modify it as directed.

Problem 3 (Non-regular languages) Prove that there is a language L such that $even(L)$ and $odd(L)$ are both regular, but L is not regular. Your solution should consists of

(a) A clear mathematical description of the language L

(b) A proof that $even(L)$ is regular. You can prove that the language is regular any way you like, e.g., by giving a DFA or a regular expression. But make sure you clearly state your logic, e.g., you could write something like "The language even(L) is the set of string such that ..... This language is regular because it is the language of the following regular expression ...".

(c) A proof that $odd(L)$ is regular. Similar to part (b).

(d) A proof that L is not regular, e.g., by using the pumping lemma, or the closure properties of regular languages, or any combination those techniques.

Submit your solution as `HW33.pdf`.

Backup Problem (half credit) This is a backup problem, in case you find problem 3 too hard. You can submit a solution to this problem as `HW33.pdf` instead of problem 3 for partial credit. If you elect do to do so, start your solution by stating that you are solving the back up problem. If you solved problem 3, you do not need to do this. You should submit the solution to only one of the two problems.

Problem: Let PAL2 be the set of all palyndromes of even length over the alphabet $\{a, b\}$, i.e., the set of all strings $w \in \Sigma^*$ such that the length of w is even, and $w^R = w$, where w^R is the reverse of w. Prove, using the pumping lemma, that the language PAL2 is non-regular.