CSE 105: Automata and Computability Theory **Autumn 2014**

Homework #3 Solutions

ANDREAS PRODROMOU A53049230

Collaborated with: NONE

Solution to Problem 1

a. Proof Idea: I will construct a new DFA that recognizes THREES(L), based on the DFA that recognizes L. I will create five copies of the original DFA. Each copy will denote the number of characters read so far. In other words, if we are currently at DFA copy #0, 1 or 2, it means we have already read exactly zero, one or two characters respectively. Our new δ function will force a jump to the next DFA level when a new character is read. Since the target language recognizes strings of size three, any input string that exceeds this size will bring our computation on DFA #4, where it will remain until the end of the string, based on a special rule in our new δ function. Our accepted states will be those states that are on DFA #3 AND they are live states in the original DFA. A state on DFA #3 is guaranteed to have been reached after reading exactly 3 characters and since it's a live state, we know it can lead to at least one string in L.

Proof:
Let DFA $M = (Q, \Sigma, \delta, q_0, F)$ recognize L.
Construct DFA $M' = (Q', \Sigma, \delta', q_0', F')$ to recognize THREES(L).

1. $Q' = Q \times \{0, 1, 2, 3, 4\}$
2. $q_0' = (q_0, 0)$
3. $\delta'((q, 4), c) = (\delta(q, c), 4) \rightarrow$ Making sure that once something arrives in DFA #4 stays there forever
   $\delta'((q, x), c) = (\delta(q, c), x + 1)$
4. $F = \{(q, 3) \mid q \in LiveStates(M)\}$

Solution to Problem 2

a. Proof Idea: My proof is based on an observation about the target language HASPREFIX(L). From the definition of HASPREFIX(L) it's guaranteed that all the accepted strings will visit at least two accepted states during their computation. The first accepted state will be visited at the end of the "prefix" string x and the last accepted state will be visited at the end of the string xy (Assuming the string xy is accepted). I

will construct an NFA with two copies of the original DFA. For all accepted states of the first DFA, I will add transitions to the second DFA copy. To ensure the last condition of the target language, these new transitions will consume a character (i.e. no epsilon transitions). The accepted states of my resulting NFA will be all the accepted states of the original DFA that are in the second DFA copy. By construction, it is guaranteed that more than one accepted states will be visited before a string is accepted and it also guarantees that the transition between the first and last accepted state visited, is at least a character long.

Proof:
Let DFA $M = (Q, \Sigma, \delta, q_0, F)$ recognize L.
Construct NFA $M' = (Q', \Sigma, \delta', q_0', F')$ to recognize HasPrefix(L).

1. $Q' = Q \times \{B, A\} \rightarrow$ Before and After DFA copies
2. $q_0' = (q_0, B)$
3. $F' = F \times A$
4. $\delta'((q, A), c) = (\delta(q, c), A)$

$$\delta'((q, B), c) = \begin{cases} (\delta(q, c), A) & \text{if } q \in F \\ (\delta(q, c), B) & \text{otherwise} \end{cases}$$