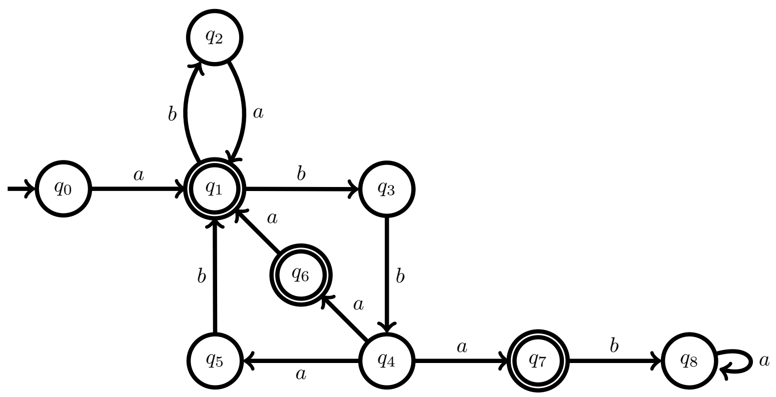
Automata and Formal Languages — Exercise Sheet 14

Exercise 14.1
Let B be the following Büchi automaton:



(a) Execute the emptiness algorithm *NestedDFS* on B.

(b) Recall that *NestedDFS* is a non-deterministic algorithm and different choices of runs may return different lassos. Which lassos of B can be found by *NestedDFS*?

(c) Show that *NestedDFS* is non optimal by exhibiting some search sequence on B.

(d) Execute the emptiness algorithm *OneStack* on B.

(e) Which lassos of B can be found by *OneStack*?

Exercise 14.2
A Büchi automaton is weak if none of its strongly connected components contains both accepting and non-accepting states. Give an emptiness algorithm for weak Büchi automata. What is the complexity of the algorithm?

Solution 14.1

(a) Let us assume that the algorithms always pick states in ascending order with respect to their indices. *dfs1* visits $q_0, q_1, q_2, q_3, q_4, q_5, q_6$, then calls *dfs2* which visits $q6, q1, q2, q3, q4, q5, q6$ and reports "non empty".

(b) Since q_7 does not belong to any lasso, only lassos containing q_1 or q_6 can be found. In every run of the algorithm, *dfs1* blackens q_6 before q_1. The only lasso containing q_6 is: $q_0, q_1, q_3, q_4, q_6, q_1$. Therefore, this is the only lasso that can be found by the algorithm.

(c) The execution given in (a) shows that *NestedDFS* is non optimal since it returns the lasso $q_0, q_1, q_3, q_4, q_6, q_1$ even though the lasso q_0, q_1, q_2, q_1 was already appearing in the explored subgraph.

(d) Let us assume that the algorithms always pick states in ascending order with respect to their indices. The algorithm reports "non empty" after the following execution:

$$\xrightarrow{C.\text{push}(q_0)} \begin{array}{c} C \\ \hline \\ \\ q_0 \end{array} \xrightarrow{C.\text{push}(q_1)} \begin{array}{c} C \\ \hline \\ q_1 \\ q_0 \end{array} \xrightarrow{C.\text{push}(q_2)} \begin{array}{c} C \\ \hline q_2 \\ q_1 \\ q_0 \end{array} \xrightarrow{C.\text{pop}()} \begin{array}{c} C \\ \hline \\ q_1 \\ q_0 \end{array} \xrightarrow{C.\text{pop}()} \begin{array}{c} C \\ \hline \\ \\ q_0 \end{array}$$

(e) All of them. The lasso q_0, q_1, q_2, q_1 is found by the above execution. The lasso $q_0, q_1, q_3, q_4, q_6, q_1$ is found by the following execution:

$$\xrightarrow{C.\text{push}(q_0)} \begin{array}{c} C \\ \hline \\ \\ \\ \\ q_0 \end{array} \xrightarrow{C.\text{push}(q_1)} \begin{array}{c} C \\ \hline \\ \\ \\ q_1 \\ q_0 \end{array} \xrightarrow{C.\text{push}(q_3)} \begin{array}{c} C \\ \hline \\ \\ q_3 \\ q_1 \\ q_0 \end{array} \xrightarrow{C.\text{push}(q_4)} \begin{array}{c} C \\ \hline \\ q_4 \\ q_3 \\ q_1 \\ q_0 \end{array} \xrightarrow{C.\text{push}(q_6)} \begin{array}{c} C \\ \hline q_6 \\ q_4 \\ q_3 \\ q_1 \\ q_0 \end{array} \xrightarrow{C.\text{pop}()} \begin{array}{c} C \\ \hline \\ q_4 \\ q_3 \\ q_1 \\ q_0 \end{array}$$

The lasso $q_0, q_1, q_3, q_4, q_5, q_1$ is found by the following execution:

$$\xrightarrow{C.\text{push}(q_0)} \begin{array}{c} C \\ \hline \\ \\ \\ \\ q_0 \end{array} \xrightarrow{C.\text{push}(q_1)} \begin{array}{c} C \\ \hline \\ \\ \\ q_1 \\ q_0 \end{array} \xrightarrow{C.\text{push}(q_3)} \begin{array}{c} C \\ \hline \\ \\ q_3 \\ q_1 \\ q_0 \end{array} \xrightarrow{C.\text{push}(q_4)} \begin{array}{c} C \\ \hline \\ q_4 \\ q_3 \\ q_1 \\ q_0 \end{array} \xrightarrow{C.\text{push}(q_5)} \begin{array}{c} C \\ \hline q_5 \\ q_4 \\ q_3 \\ q_1 \\ q_0 \end{array}$$

$$\xrightarrow{C.\text{pop}()} \begin{array}{c} C \\ \hline q_4 \\ q_3 \\ q_1 \\ q_0 \end{array} \xrightarrow{C.\text{pop}()} \begin{array}{c} C \\ \hline \\ q_3 \\ q_1 \\ q_0 \end{array} \xrightarrow{C.\text{pop}()} \begin{array}{c} C \\ \hline \\ \\ q_1 \\ q_0 \end{array} \xrightarrow{C.\text{pop}()} \begin{array}{c} C \\ \hline \\ \\ \\ q_0 \end{array}$$

Solution 14.2

The idea is to maintain a set V of the gray vertices: when a dfs meets a gray state r, by the gray-path theorem this means that there is a cycle with r in it, and since we are considering weak Büchi automata it suffices to check if r is gray. The following algorithm works in linear time:

The space complexity is $O(|V|)$, as we maintain two sets S, V that can both contain at most all the nodes of the graph. The time complexity is $O(|V| + |E|)$, same as DFS.

Input: Weak Büchi automaton $B = (Q, \Sigma, \delta, q_0, F)$.
Output: $L_\omega(B) = \emptyset$?

$S, V \leftarrow \emptyset$
`dfs`(q_0)
report “empty”

`dfs`(q) *:*
    S.**add**(q)
    V.**add**(q)
    for $r \in$ **succ**(q) **do**
        if $r \notin S$ **then**
            `dfs`(r)
        else if $r \in V$ and $r \in F$ **then**
            report “non empty”
    V.**remove**(q)