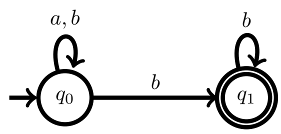
Automata and Formal Languages — Homework 12

Due 27.01.2017

Exercise 12.1
Consider the following Büchi automaton over $\Sigma = \{a, b\}$:



(a) Sketch $\text{dag}(abab^\omega)$ and $\text{dag}((ab)^\omega)$.

(b) Let r_w be the ranking of $\text{dag}(w)$ defined by

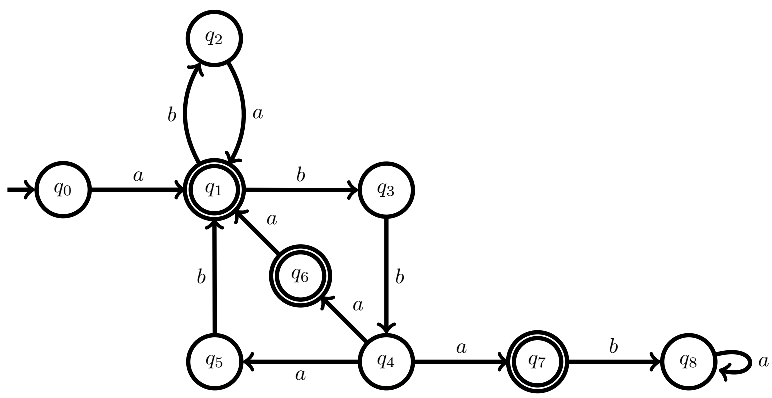
$$r_w(q,i) = \begin{cases} 1 & \text{if } q = q_0 \text{ and } \langle q_0, i\rangle \text{ appears in dag}(w), \\ 0 & \text{if } q = q_1 \text{ and } \langle q_1, i\rangle \text{ appears in dag}(w), \\ \bot & \text{otherwise.} \end{cases}$$

Are r_{abab^ω} and $r_{(ab)^\omega}$ odd rankings?

(c) Show that r_w is an odd ranking if and only if $w \notin L_\omega(B)$.

(d) Build a Büchi automaton accepting $\overline{L_\omega(B)}$ using the construction seen in class. (Hint: by (c), it is sufficient to use $\{0, 1\}$ as ranks.)

Exercise 12.2
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 *TwoStack* on B.

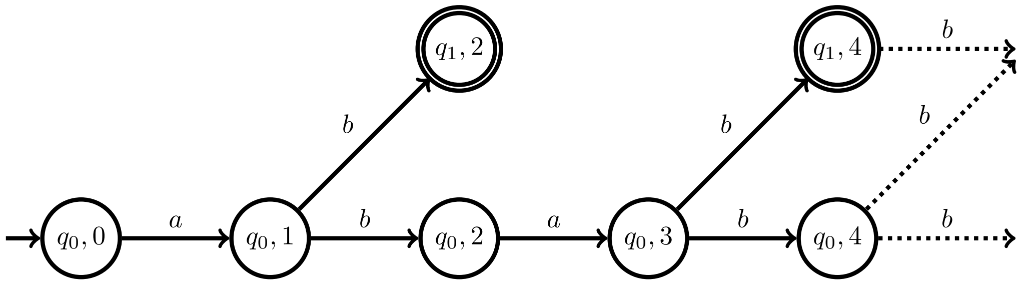
(e) Which lassos of B can be found by *TwoStack*?

Exercise 12.3

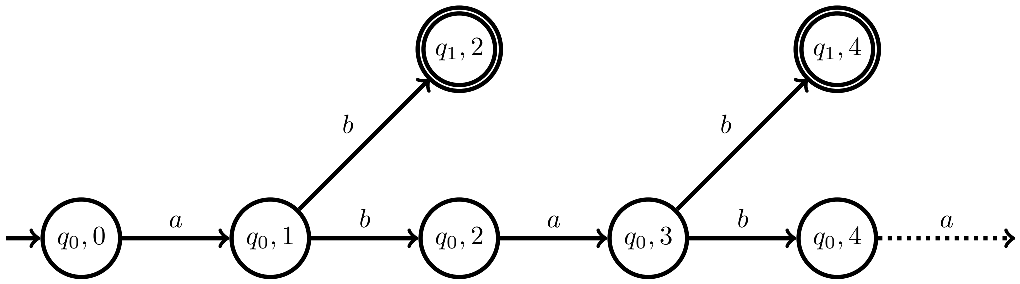
A Büchi automaton is weak if none of its strongly connected components contains both accepting and non-accepting states. Give an emptiness algorithm for weak Büchi automata. What is the complexity of the algorithm?

Solution 12.1

(a) dag($abab^\omega$):



dag($(ab)^\omega$):



(b)
- r *is not* an odd rank for dag($abab^\omega$) since

$$\langle q_0, 0\rangle \xrightarrow{a} \langle q_0, 1\rangle \xrightarrow{b} \langle q_0, 2\rangle \xrightarrow{a} \langle q_0, 3\rangle \xrightarrow{b} \langle q_1, 4\rangle \xrightarrow{b} \langle q_1, 5\rangle \xrightarrow{b} \cdots$$

is an infinite path of dag($abab^\omega$) not visiting odd nodes infinitely often.

- r *is* an odd rank for dag($(ab)^\omega$) since it has a single infinite path:

$$\langle q_0, 0\rangle \xrightarrow{a} \langle q_0, 1\rangle \xrightarrow{b} \langle q_0, 2\rangle \xrightarrow{a} \langle q_0, 3\rangle \xrightarrow{b} \langle q_0, 4\rangle \xrightarrow{a} \langle q_0, 5\rangle \xrightarrow{b} \cdots$$

which only visits odd nodes.

(c) $\Rightarrow$) Let $w \in L_\omega(B)$. We have $w = ub^\omega$ for some $u \in \{a, b\}^*$. This implies that

$$\langle q_0, 0\rangle \xrightarrow{u} \langle q_0, |u|\rangle \xrightarrow{b} \langle q_1, |u| + 1\rangle \xrightarrow{b} \langle q_1, |u| + 2\rangle \xrightarrow{b} \cdots$$

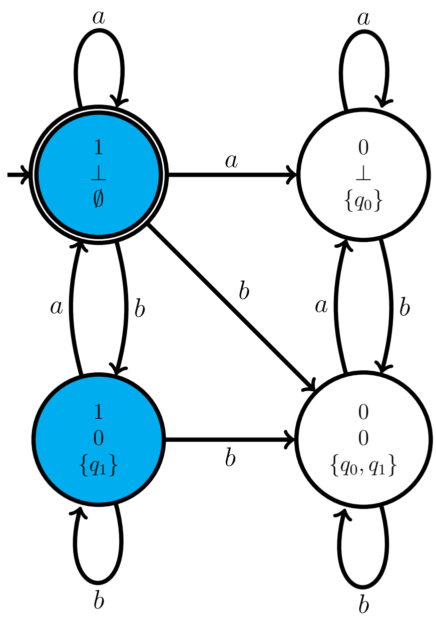
is an infinite path of dag(w). Since this path does not visit odd nodes infinitely often, r is not odd for dag(w).

$\Leftarrow$) Let $w \notin L_\omega(B)$. Suppose there exists an infinite path of dag(w) that does not visit odd nodes infinitely often. At some point, this path must only visit nodes of the form $\langle q_1, i\rangle$. Therefore, there exists $u \in \{a, b\}^*$ such that

$$\langle q_0, 0\rangle \xrightarrow{u} \langle q_1, |u|\rangle \xrightarrow{b} \langle q_1, |u| + 1\rangle \xrightarrow{b} \langle q_1, |u| + 2\rangle \xrightarrow{b} \cdots$$

This implies that $w = ub^\omega \in L_\omega(B)$ which is contradiction.

(d) By (c), for every $w \in \{a, b\}^\omega$, if dag(w) has an odd ranking, then it has one ranging over 0 and 1. Therefore, it suffices to execute *CompNBA* with rankings ranging over 0 and 1. We obtain the following Büchi automaton:



★ Actually, by (c), it is sufficient to only explore the blue states as they correspond to the family of rankings $\{r_w : w \in \Sigma^\omega\}$.

Solution 12.2

(a) Let us assume that the algorithms always pick states in ascending order with respect to their indices. *dfs1* visits $q0, q1, q2, q3, q4, q5, q6$, 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:

$\xrightarrow[V.\text{push}(q_0)]{C.\text{push}(q_0)}$

C	V
q_0	q_0

$\xrightarrow[V.\text{push}(q_1)]{C.\text{push}(q_1)}$

C	V
q_1	q_1
q_0	q_0

$\xrightarrow[V.\text{push}(q_2)]{C.\text{push}(q_2)}$

C	V
q_2	q_2
q_1	q_1
q_0	q_0

$\xrightarrow{C.\text{pop}()}$

C	V
	q_2
q_1	q_1
q_0	q_0

$\xrightarrow{C.\text{pop}()}$

C	V
	q_2
	q_1
q_0	q_0

(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:

$\xrightarrow[V.\text{push}(q_0)]{C.\text{push}(q_0)}$

C	V
q_0	q_0

$\xrightarrow[V.\text{push}(q_1)]{C.\text{push}(q_1)}$

C	V
q_1	q_1
q_0	q_0

$\xrightarrow[V.\text{push}(q_3)]{C.\text{push}(q_3)}$

C	V
q_3	q_3
q_1	q_1
q_0	q_0

$\xrightarrow[V.\text{push}(q_4)]{C.\text{push}(q_4)}$

C	V
q_4	q_4
q_3	q_3
q_1	q_1
q_0	q_0

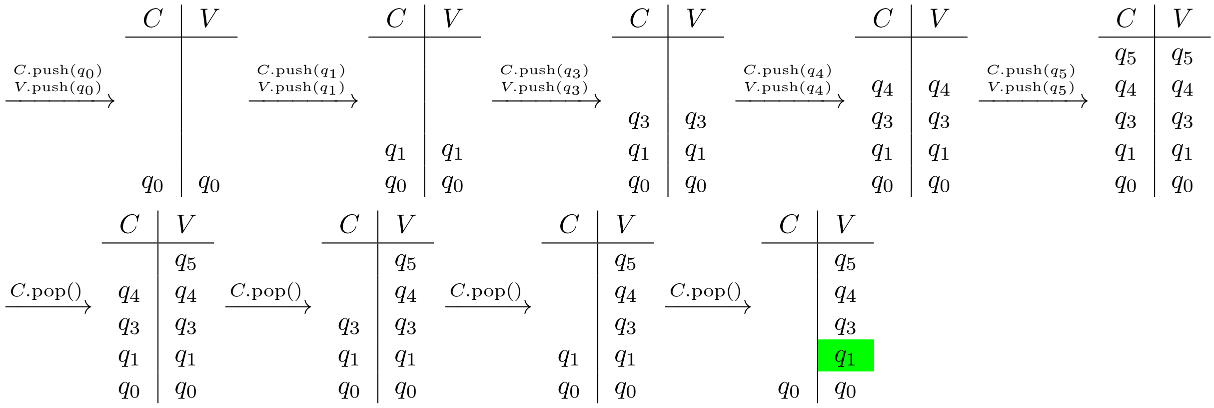
$\xrightarrow[V.\text{push}(q_6)]{C.\text{push}(q_6)}$

C	V
q_6	q_6
q_4	q_4
q_3	q_3
q_1	q_1
q_0	q_0

$\xrightarrow{C.\text{pop}()}$

C	V
	q_6
q_4	q_4
q_3	q_3
q_1	q_1
q_0	q_0

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



Solution 12.3

The following algorithm works in linear time:

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)
