Technische Universität München
Winter term 2017/18
Prof. J. Esparza / Dr. M. Blondin

Automata and Formal Languages — Endterm

- You have 120 minutes to complete the exam.
- Answers must be written in a separate booklet. Do not answer on the exam.
- Please let us know if you need more paper.
- Write your name and Matrikelnummer on every sheet.
- Write with a non-erasable pen. Do not use red or green.
- You are not allowed to use auxiliary means other than pen and paper.
- You can obtain 40 points. You need 17 points to pass.

Question 1 $\quad$ **(2 + 2 + 2 + 3 + 2 = 11 points)**

(a) Let $\varphi = ((\mathbf{F}p)\ \mathbf{U}\ (\mathbf{X}q)) \wedge \neg\mathbf{XGF}q$ be an LTL formula over the set of atomic propositions $AP = \{p, q\}$. Give an ω-word that satisfies φ, and give an ω-word that does not satisfy φ. (Make clear which is which!)

(b) Give a transducer over alphabet $\{0,1\} \times \{0,1\}$ accepting all least significant bit first (lsbf) encodings of pairs $(x, y) \in \mathbb{N} \times \mathbb{N}$ such that $x + 1 \equiv y \pmod 4$. For example, $(01100, 11010)$ should be accepted, and $(1100, 1001)$ should be rejected.

(c) Prove (via a proof) or disprove (via a counterexample) that the following LTL equivalence holds for every formula φ and ψ:

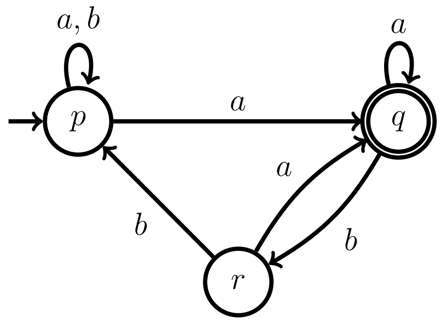
$$\neg\mathbf{GF}\neg\varphi \wedge \neg\mathbf{GF}\psi \ \equiv\ \mathbf{FG}\neg(\varphi \rightarrow \psi).$$

(d) We have seen that testing the inclusion of two fixed-length languages can be achieved by using the algorithms for intersection and equality. It is also possible to test inclusion directly, as sketched in the following algorithm. Fill the blanks of the algorithm and briefly justify your answer.

Note that the alphabet of the master automaton is $\Sigma = \{a_1, a_2, \ldots, a_m\}$.

Input: states q_1, q_2 of the master automaton such that $L(q_1)$ and $L(q_2)$ are of the same length.
Output: $L(q_1) \subseteq L(q_2)$?
`inclusion(`q_1, q_2`)`*:*
    if $G(q_1, q_2)$ is not empty **then return** $G(q_1, q_2)$
    else if $\boxed{\text{Blank 1}}$ **then return** *true*
    else if $\boxed{\text{Blank 2}}$ **then return** *false*
    else
        for $i = 1, \ldots, m$ **do**
            $x_i \leftarrow \boxed{\text{Blank 3}}$
        $G(q_1, q_2) \leftarrow \mathbf{and}(x_1, \ldots, x_m)$
        return $G(q_1, q_2)$

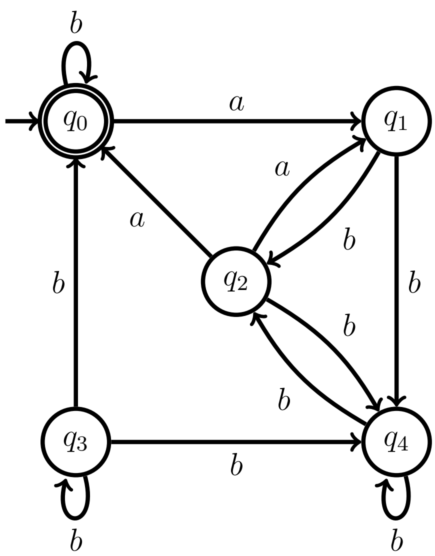
(e) Consider the following Büchi automaton:



Sketch at least 5 levels of $dag(aa(ba)^\omega)$ and $dag(aab^\omega)$. For each dag, say whether there exists an odd ranking. Justify your answers by either giving an odd ranking or by explaining why there is none.

Question 2 $\quad$ (1 + 2 + 2 = 5 points)

Reduce the size of the following NFA by means of the partition refinement algorithm introduced in the course.



(a) Give the blocks of the initial partition.

(b) Describe for each step which block is being refined, which is the splitter, and which are the two blocks resulting from the split.

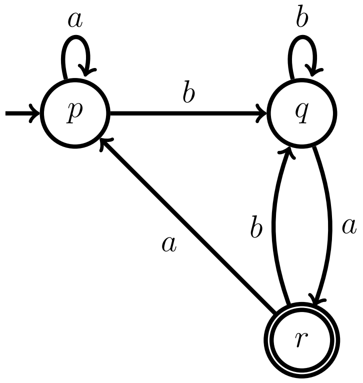
(c) Draw the quotient of the original automaton with respect to the partition computed in (b).

Question 3 **($3+2=5$ points)**

Let Σ be an alphabet. Let $\text{shift}(\varepsilon) = \varepsilon$ and let $\text{shift}(a_1 a_2 \cdots a_n) = a_2 a_3 \cdots a_n a_1$ for every $n \in \mathbb{N}$ and $a_1, a_2, \ldots, a_n \in \Sigma$. For every language $L \subseteq \Sigma^*$, let $\text{shift}(L) = \{\text{shift}(w) : w \in L\}$. For example,

$$\text{shift}(\{\varepsilon, aab, bab, bbb\}) = \{\varepsilon, aba, abb, bbb\}.$$

(a) Give an NFA that recognizes $\text{shift}(L)$ where L is the language recognized by the following DFA:



(b) Sketch, with the help of your solution to (a), a general procedure that, given a DFA recognizing a language L, returns an NFA recognizing $\text{shift}(L)$.

Question 4 **($3+2+2=7$ points)**

Consider the following program made of two processes sharing a variable x initialized to 0:

while *true*:
  if $x = 1$:
    $x \leftarrow 0$
	while *true*:
	  $x \leftarrow 1 - x$

(a) Model the program by constructing a network of three automata (one for each process and one for the variable). Give the alphabet of each automaton. Each alphabet should be a subset of $\{x = 1,\ x \leftarrow 0,\ x \leftarrow 1 - x\}$.

(b) Construct the asynchronous product of the three automata obtained in (a). The alphabet of the automaton should be $\Sigma = \{x = 1,\ x \leftarrow 0,\ x \leftarrow 1 - x\}$.

(c) Let p be an atomic proposition that holds if and only if "variable x has value 1". Does $\mathbf{FG}p$ hold for every infinite execution of the program? Justify your answer.

Question 5 (2 + 2 + 2 = 6 points)
Let $L = \{w \in \{a,b\}^* :$ between any two consecutive b's of w there is an even number of a's$\}$. (In particular, every word with no b's or with a single b belongs to L.)

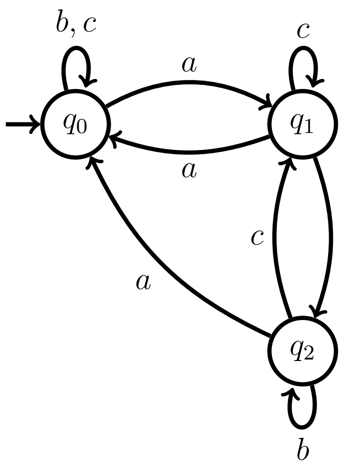
(a) Give the minimal DFA for L.

(b) Give an MSO formula $\text{Even}(x, y)$ that holds if and only if $x \le y$ and $\{x, x+1, \ldots, y\}$ is of even size.

(c) Give an MSO formula φ such that $L(\varphi) = L$. You can make use of the formula $\text{Even}(x, y)$.

For (b) and (c), you can *only* use the standard expressions $Q_a(x)$, $x < y$, $\neg\varphi$, $\varphi_1 \vee \varphi_2$, $\exists x\ \varphi$, and the abbreviations $\forall x\ \varphi$, $\varphi_1 \wedge \varphi_2$, $\varphi_1 \rightarrow \varphi_2$, $\varphi_1 \leftrightarrow \varphi_2$, $x = y$, $x \le y$, $\text{first}(x)$, $\text{last}(x)$, and $y = x + k$ where k is a constant. If you wish to use other abbreviations, you must define them.

Question 6 (2 + 2 + 2 = 6 points)
Consider the following Muller automaton A, over alphabet $\Sigma = \{a, b, c\}$, with acceptance condition $\{\{q_1, q_2\}\}$:



(a) Describe the ω-language recognized by A. Justify your answer.

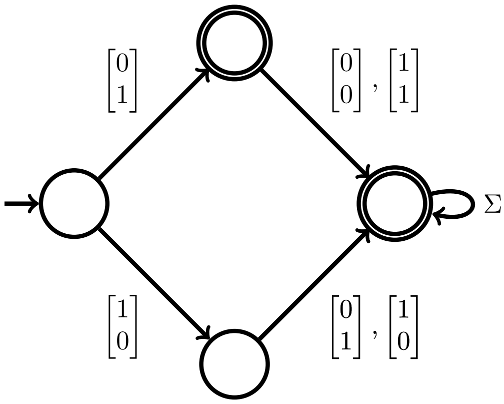
(b) Give an ω-regular expression for the ω-language recognized by A.

(c) Give a (non deterministic) Büchi automaton that recognizes the same ω-language as A. You may use a conversion algorithm seen in class, but it is not mandatory.

Solution 1 $(2+2+2+3+2 = 11$ **points)**

(a) $\emptyset\{q\}\emptyset^\omega$ satisfies φ, and $\emptyset^\omega$ does not satisfy φ.

(b)



(c) The equivalence holds:

$$\begin{aligned}
\neg\mathbf{GF}\neg\varphi \wedge \neg\mathbf{GF}\psi &\equiv \neg(\mathbf{GF}\neg\varphi \vee \mathbf{GF}\psi) && \text{(by de Morgan's law)}\\
&\equiv \neg(\mathbf{GF}(\neg\varphi \vee \psi)) && \text{(by distributivity of } \mathbf{GF} \text{ over disjunction)}\\
&\equiv \neg\mathbf{GF}(\varphi \rightarrow \psi) && \square
\end{aligned}$$

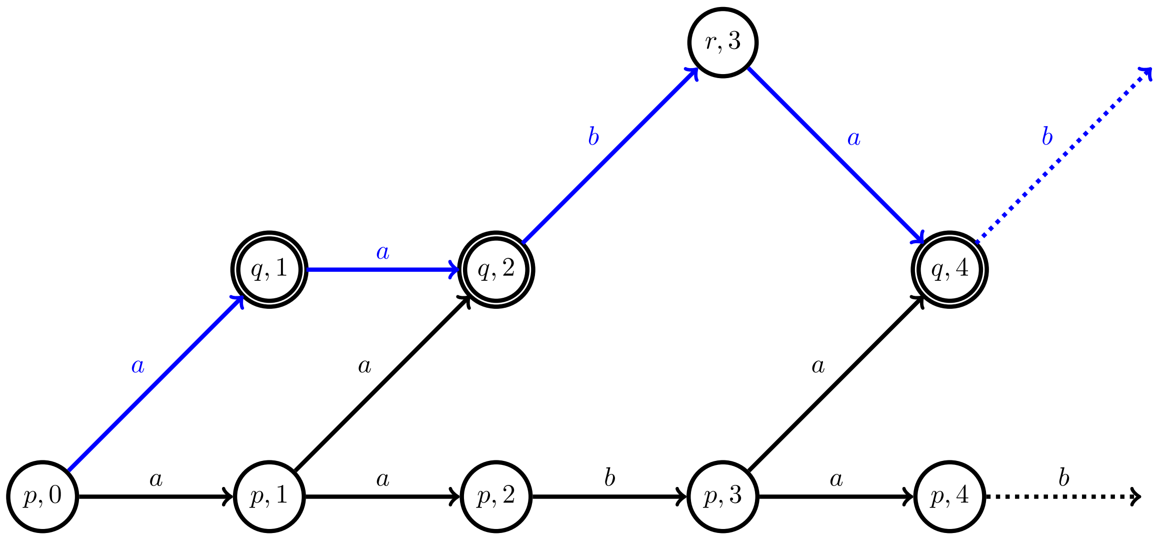
(d)

Input: states q_1, q_2 of the master automaton and of the same length.
Output: $L(q_1) \subseteq L(q_2)$?
`inclusion`(q_1, q_2) *:*
  if $G(q_1, q_2)$ is not empty **then return** $G(q_1, q_2)$
  else if $\boxed{(q_1 = \emptyset \vee q_2 = q_\varepsilon)}$ **then return** *true*
  else if $\boxed{(q_1 = q_\varepsilon \wedge q_2 = q_\emptyset)}$ **then return** *false*
  else
    for $i = 1, \ldots, m$ **do**
      $x_i \leftarrow \boxed{\texttt{inclusion}(q_1^{a_i}, q_2^{a_i})}$
    $G(q_1, q_2) \leftarrow \mathbf{and}(x_1, \ldots, x_m)$
    return $G(q_1, q_2)$

The algorithm works due to the following identity:

$$L_1 \subseteq L_2 \iff \begin{cases} L_1 = \emptyset \vee L_2 = \{\varepsilon\} & \text{if length}(L_1) = 0 \text{ or length}(L_2) = 0,\\ \bigwedge_{1 \leq i \leq m} L_1^{a_i} \subseteq L_2^{a_i} & \text{otherwise.} \end{cases}$$

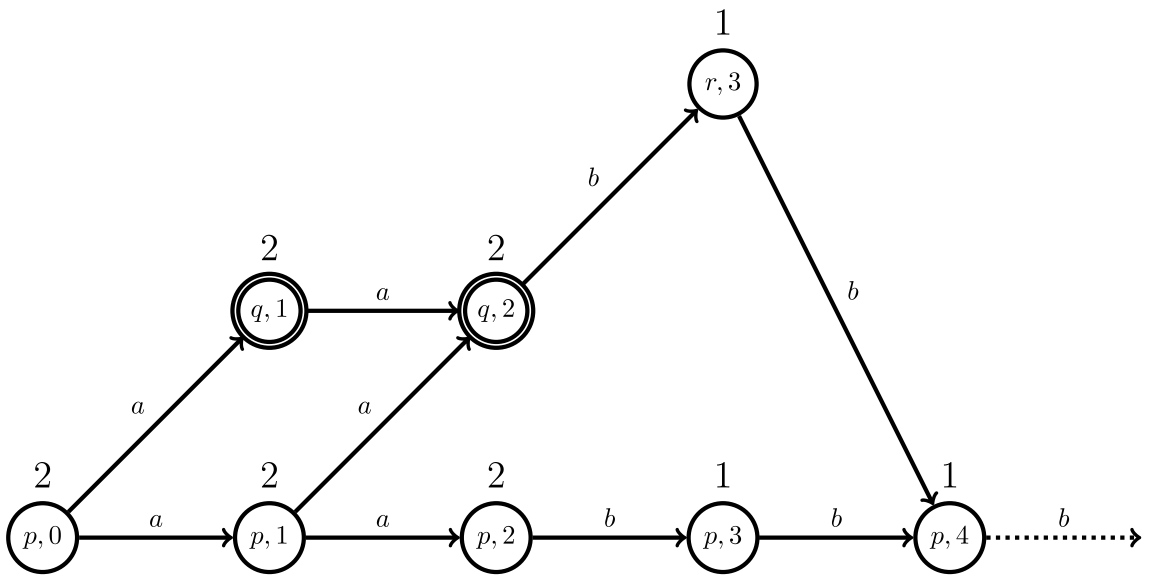
(e) $dag(aa(ba)^\omega)$:



The above dag has no odd ranking since it contains a path (coloured in blue) with infinitely many accepting states.

In more details, assume for the sake of contradiction that there exists an odd ranking r. Since r is non decreasing along the blue path, it must stabilize to some rank $n \in \mathbb{N}$. Since r is odd, n must be odd. Since the blue path contains infinitely many occurrences of q, r must give rank n to a node of the form (q, i). This is a contradiction since the rank of an accepting state must be even.

$dag(aab^\omega)$:



The above dag has an odd ranking, e.g. the ranking that assigns 2 to all nodes of the three first layers and 1 to all other nodes.

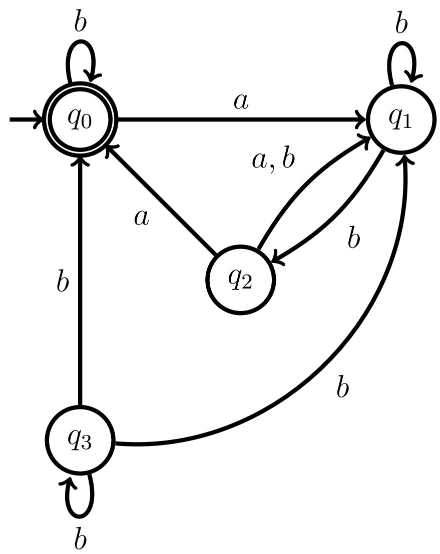
Solution 2 (1 + 2 + 2 = 5 points)

(a) $\{q_0\}$ and $\{q_1, q_2, q_3, q_4\}$

(b)

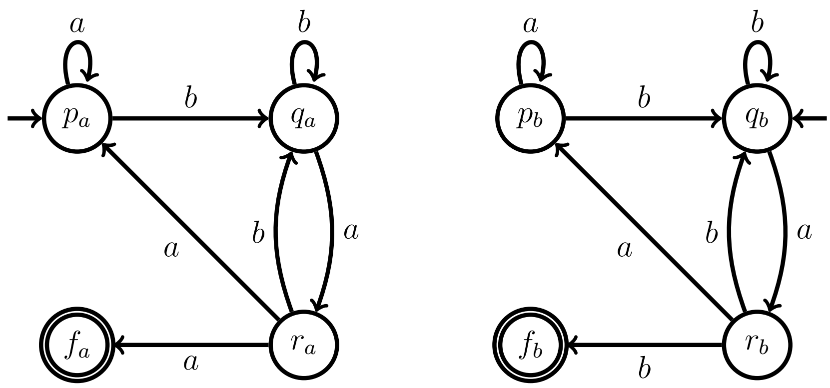
Iter.	Block to split	Splitter	New partition
0	—	—	$\{q_0\}, \{q_1, q_2, q_3, q_4\}$
1	$\{q_1, q_2, q_3, q_4\}$	$(a, \{q_0\})$	$\{q_0\}, \{q_2\}, \{q_1, q_3, q_4\}$
2	$\{q_1, q_3, q_4\}$	$(b, \{q_0\})$	$\{q_0\}, \{q_1, q_4\}, \{q_2\}, \{q_3\}$
3	none, partition is stable	—	—

(c)



Solution 3 (3 + 2 = 5 points)

(a)



(b) The NFA is obtained as follows. Intuitively, the automaton guesses the first letter σ, reads the remaining letters, and then accepts by reading σ. In more details, given a DFA A, we construct a copy A_σ for each letter $\sigma \in \Sigma$. Each state originally final in A_σ is made non final and has a new transition labeled by σ to a new final state f_σ. Every state of A reachable in one step by reading a letter σ is made initial in A_σ. If A accepts ε, we also add an extra initial and accepting state r.

Formally, the automaton $A' = (Q', \Sigma, \delta', Q'_0, F')$ obtained from $A = (Q, \Sigma, \delta, q_0, F)$ is as follows:

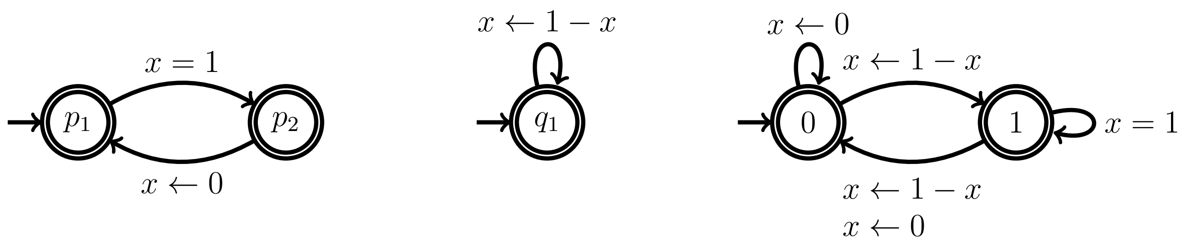
$$\begin{aligned} Q' &= \{q_\sigma : q \in Q, \sigma \in \Sigma\} \cup \{f_\sigma : \sigma \in \Sigma\} \cup \{r\}, \\ Q'_0 &= \{q_\sigma : \sigma \in \Sigma, q_0 \xrightarrow{\sigma} q\} \cup \{r\}, \\ F' &= \begin{cases} \{f_\sigma : \sigma \in \Sigma\} \cup \{r\} & \text{if } q_0 \in F, \\ \{f_\sigma : \sigma \in \Sigma\} & \text{otherwise.} \end{cases} \end{aligned}$$

and

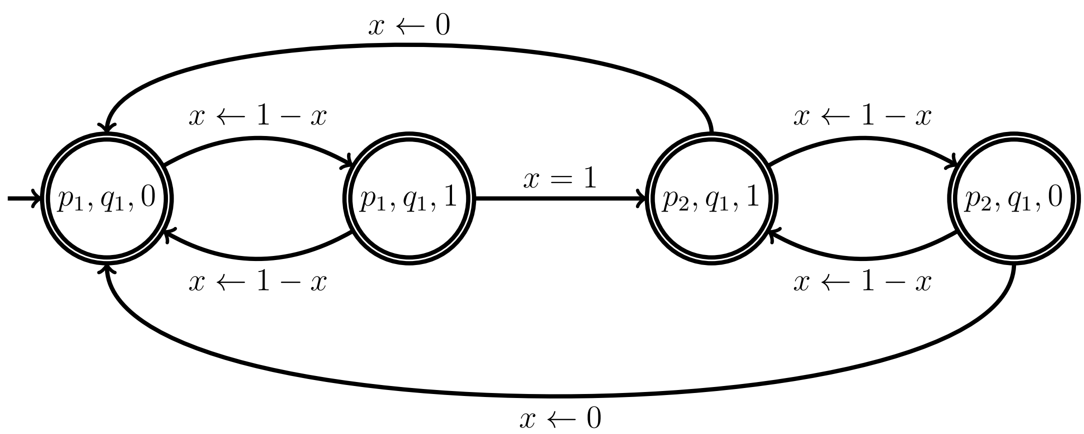
$$\delta(q_\sigma, a) = \begin{cases} \{r_\sigma : q \xrightarrow{a} r\} & \text{if } \sigma \neq a, q \in Q, \\ \{r_\sigma : q \xrightarrow{a} r\} \cup \{f_\sigma\} & \text{if } \sigma = a, q \in Q. \end{cases}$$

Solution 4 (3 + 2 + 2 = 7 points)

(a)



(b)

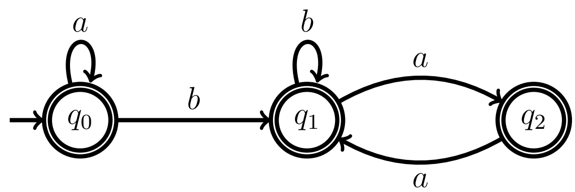


(c) No. $\mathbf{FG}p$ means that eventually x has value 1 permanently. This does not hold, e.g., for the following infinite execution:

$$(p_1, q_1, 0) \xrightarrow{x \leftarrow 1-x} (p_1, q_1, 1) \xrightarrow{x \leftarrow 1-x} (p_1, q_1, 0) \xrightarrow{x \leftarrow 1-x} \cdots.$$

Solution 5 (2 + 2 + 2 = 6 points)

(a)



(b)

$$\begin{aligned} \text{Even}(x, y) = (x \leq y) \wedge \exists E\ [(y \notin E) \wedge \\ (\forall i\ [(i \in E) \leftrightarrow (i = x \vee (i \leq y \wedge \exists j\ (i = j + 2 \wedge j \in E)))])] \end{aligned}$$

(c) $\forall x, y\ [(x < y \land Q_b(x) \land Q_b(y) \land \text{SurroundA}(x, y)) \to \text{Even}(x, y)]$ where

$$\text{SurroundA}(x, y) = \forall i\ [(x < i \land i < y) \to Q_a(i)]$$

Solution 6 (2 + 2 + 2 = 6 points)

(a) Infinite words with finitely many a's, infinitely many b's *and* c's, and an odd number of a's.

(b) $(a(b+c)^*a + b + c)^*a(b^*cc^*b)^\omega$

(c)