Technische Universität München
Winter term 2018/19
Prof. J. Esparza / S. Sickert

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 $(1 + 2 + 1 + 1 + 1 + 1 + 1 = 8$ points)

a. Draw the LazyDFA over the alphabet $\Sigma = \{a, l, s\}$ for the pattern $p = aalsaal$.

b. Give all residuals of the language $L = \{w : w \text{ contains an odd number of } ab\text{'s}\}$ over the alphabet $\Sigma = \{a, b, c\}$ and give for each non-empty residual the lexicographically smallest element of that residual.

c. Give an NFA over the alphabet $\Sigma = \{a, b\}$ that recognises exactly the language of the following MSO formula φ:

$$\varphi = \forall X.\ \mathsf{Even}(X) \rightarrow (\forall x \in X.\ \mathsf{last}(x) \rightarrow Q_a(x))$$

d. Consider the LTL formula $\varphi = \neg p \wedge \mathbf{X}(q\ \mathbf{U}\ r)$ over the set of atomic propositions $Ap = \{p, q, r\}$. Compute the satisfaction sequence $sats(\varphi, \sigma)$ for the computation $\sigma = \{r\}(\{p\}\{q, r\})^\omega$.

e. Give a DRA over the alphabet $\Sigma = 2^{\{p,q\}}$ that recognisises exactly all computations satisfying $\mathbf{FG}(p\ \mathbf{U}\ q)$.

f. Prove or disprove: Every ω-regular language can be recognised by an NRA with a single initial state and a single Rabin pair.

g. Prove or disprove: Let $\Sigma = \{a\}$ be a singleton alphabet. Then any language $L \subseteq \Sigma^\omega$ is ω-regular.

Question 2 $(2 + 2 = 4$ points)

Consider the following regular language L over the alphabet $\Sigma = \{a, b\}$:

$$L = \{w \in \Sigma^* : w \text{ has odd length and does not contain } aa\}$$

a. Draw a DFA D recognising the language L, i.e. $L(D) = L$.

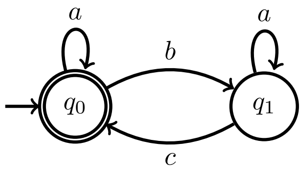
b. Compute the regular expression r recognising the language L, i.e. $L(r) = L$, by applying the algorithm presented in the lecture and document each step of the algorithm.

Question 3 (1 + 3 = 4 points)

Let us define for any language $L \subseteq \Sigma^*$ the following operation:

$$\frac{1}{2}L := \{w \in \Sigma^* : \exists u \in \Sigma^*.\ |w| = |u| \wedge wu \in L\}$$

a. Construct an NFA N' such that $L(N') = \frac{1}{2}L(N)$ where N is the following NFA with $\Sigma = \{a, b, c\}$:



b. Prove that if $L \subseteq \Sigma^*$ is a regular language, then $\frac{1}{2}L$ is also a regular language.

It suffices that you give a construction that takes an NFA $N = (Q, \Sigma, \delta, Q_0, F)$ for L as input and returns an NFA $N' = (Q', \Sigma, \delta', Q'_0, F')$ for $\frac{1}{2}L$ as output. Give definitions for Q', δ', Q'_0 and F' and explain the idea of the construction.

Hint: In the exercise classes you have seen a construction for the language $\sqrt{L} := \{w \in \Sigma^* : ww \in L\}$. You can adapt this construction to obtain a construction for $\frac{1}{2}L$.

Question 4 (2 + 1 + 2 = 5 points)

Consider the following program made of two processes (or agents) sharing a variable x initialised to red, which models a pedestrian traffic light in a *minimalistic way*:

```
while true:                    while true:
  if x = red:                    x ← red
    x ← green
```

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 $\{x = \mathsf{red}, x \neq \mathsf{red}, x \leftarrow \mathsf{red},\ x \leftarrow \mathsf{green}\}$.

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

c. Let p be an atomic proposition that holds if and only if “variable x has value green”. Does $\mathbf{GF}p$ hold for every infinite execution of the program? Does $\mathbf{FG}p$ hold for some infinite execution of the program? Justify your answers.

Question 5 (3 + 3 = 6 points)

Recall that, given a state q of the master automaton for fixed-length languages over an alphabet Σ, and given $a \in \Sigma$, we let q^a denote the unique a-successor of q. Further, we denote by q_ϵ and $q_\emptyset$ the states of the master automaton recognising $\{\epsilon\}$ and $\emptyset$, respectively.

Recall the definition of the symmetric difference of two languages L_1 and L_2:

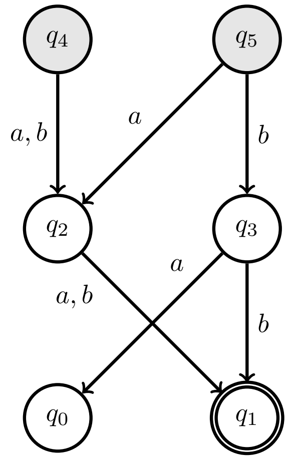
$$L_1 \,\Delta\, L_2 \coloneqq \{w : (w \in L_1 \wedge w \notin L_2) \vee (w \notin L_1 \wedge w \in L_2)\}$$

a. Fill the blanks in the algorithm for computing the state of the master automaton recognising the symmetric difference of the languages recognised by two given states. The alphabet is $\Sigma = \{a_1, a_2, \ldots, a_m\}$.

```
**Input:** states $q_1, q_2$ with same length.
**Output:** state recognizing $L(q_1) \,\Delta\, L(q_2)$.
symmdiff($q_1, q_2$):
    if $G(q_1, q_2)$ is not empty then return $G(q_1, q_2)$
    else if $\boxed{\text{Blank 1}}$ then return $q_\emptyset$
    else if $\boxed{\text{Blank 2}}$ then return $q_\varepsilon$
    else
        for $i = 1, \ldots, m$ do
            $r_i \leftarrow \boxed{\text{Blank 3}}$
        $G(q_1, q_2) \leftarrow$ make$(r_1, \ldots, r_m)$
        return $G(q_1, q_2)$
```

b. Apply the algorithm to compute the state recognising $L(q_4) \,\Delta\, L(q_5)$ to the fragment of the master automaton shown below. Describe the tree of recursive calls, give the result of each call, and draw the resulting automaton. If you use optimisations described in the course, then describe them briefly.

Always recurse into the branch labelled with **a** first and then recurse into the branch labelled with **b**.



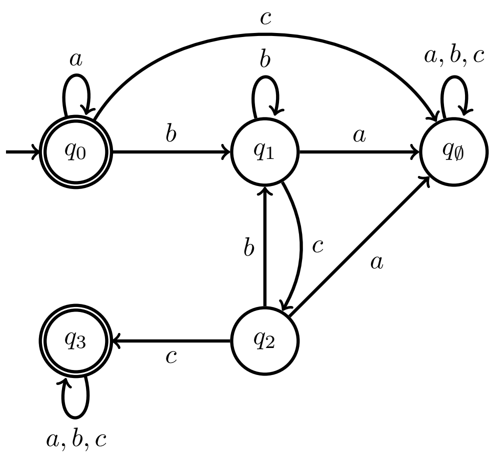
Question 6 $(1+1+2+3+2 = 9$ points)

A Büchi automaton is *weak* if every strongly connected component $S \subseteq Q$ is either contained in the accepting states ($S \subseteq F$) or is disjoint from the accepting states ($S \cap F = \emptyset$).

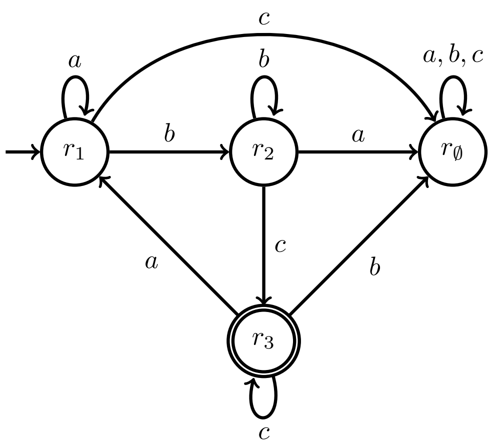
A weak, deterministic Büchi automaton D can be complemented by making accepting states non-accepting and non-accepting states accepting. Formally, let $D = (Q, \Sigma, \delta, q_0, F)$ be a weak DBA. Then the DBA $\overline{D} = (Q, \Sigma, \delta, q_0, Q \setminus F)$ accepts the complement-language, i.e. $L_\omega(\overline{D}) = \overline{L_\omega(D)}$.

Consider now the following two DBAs D_1, D_2:

D_1 :



D_2 :



a. Determine which of D_1 and D_2 are weak.

b. Draw dag$((abbc)^\omega)$ for D_1 and dag$((abbc)^\omega)$ for D_2.

c. Decide if there exists an odd ranking for dag$((abbc)^\omega)$ on D_1 and dag$((abbc)^\omega)$ on D_2. If an odd ranking exists, exhibit it and explain why it is an odd ranking. If none exists, argue why such a ranking does not exist.

d. Prove that this complementation procedure is correct for weak DBAs, i.e. show that $L_\omega(D) = \Sigma^\omega \setminus L_\omega(\overline{D})$ holds for every weak DBA D.

e. Show that the described complementation construction is not correct for weak NBAs. For this, give a weak NBA N such that $L_\omega(N) \neq \Sigma^\omega \setminus L_\omega(\overline{N})$ and an ω-word that belongs to $L_\omega(N) \Delta (\Sigma^\omega \setminus L_\omega(\overline{N}))$.

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

Let $L \subseteq \Sigma^*$ be a language. We define the *infinite infix closure* (infix$_\infty$) of a given language L as:

$$\text{infix}_\infty(L) = \{a_0a_1a_2\cdots \in \Sigma^\omega : \text{there are infinitely many } j \text{ such that } a_ja_{j+1}a_{j+2}\cdots \in L\Sigma^\omega\}$$

For example, let $L = \{ab, cc\} \subseteq \{a, b, c\}^*$ be a language. Then we have $\{(aba)^\omega, abc^\omega\} \subseteq \text{infix}_\infty(L)$ and $\{abb^\omega, abc(cba)^\omega\} \cap \text{infix}_\infty(L) = \emptyset$.

a. Prove that for any regular language L with $L \subseteq \Sigma^*$ the language infix$_\infty(L)$ is ω-regular.

b. Exhibit an ω-regular language $L' \subseteq \Sigma^\omega$ such that there is no regular language $L \subseteq \Sigma^*$ with $L' = \text{infix}_\infty(L)$. Justify your answer.