EXAM AUTOMATA THEORY
Friday 24 January 2025, 09:00 - 12:00

This exam consists of eight exercises, where [x pt] indicates how many points can be earned per exercise. A total of 100 points can be earned.
It is important to provide an explanation or motivation when a question asks for it.
A finite automaton in this exam (without further addition), refers to a deterministic finite automaton without Λ-transitions (which is elsewhere called *DFA*).

---

1. [8 pt] Consider the language

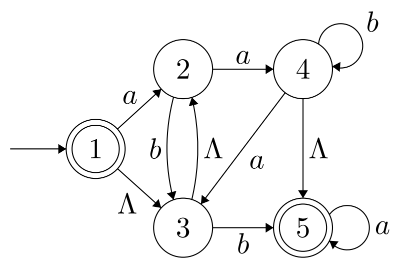
$$L = \{x \in \{a, b\}^* \mid x \text{ contains } bba, \text{ but not } aa\}$$

For example, $abba \in L$ and $bbababa \in L$, but $bbb \notin L$ because there is no bba, and $bbaa \notin L$ because it contains aa. Construct a deterministic finite automaton that accepts L.

---

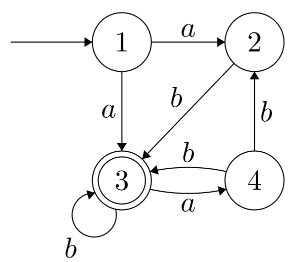
2. [14 pt]

(a) Consider the following non-deterministic finite automaton M_1:



Remove all Λ-transitions from M_1. You only need to provide the resulting automaton.

(b) Consider the following non-deterministic finite automaton M_2:



Remove the non-determinism from M_2 using the subset construction. You only need to provide the resulting automaton. You should omit non-reachable states, but you do not need to minimize the resulting automaton in any other way.

---

3. [15 pt] Consider the languages L_1 described by the regular expression

$$r_1 : (\lambda + b)(a^*b)^*a$$

and L_2 described by the regular expression

$$r_2 : (\lambda + a)(ba + a)^*$$

(a) Does it hold that $L_1 \subseteq L_2$? If yes, no explanation is needed. If no, provide a string which is an element of L_1, but not of L_2.

(b) Does it hold that $L_2 \subseteq L_1$? If yes, no explanation is needed. If no, provide a string which is an element of L_2, but not of L_1.

(c) Construct a non-deterministic finite automaton accepting L_1. The regular expression r_1 should be recognizable in the automaton.

---

4. [10 pt]

Consider the language

$$L = \{x \in \{a, b, c\}^* \mid x = wcw \text{ for some } w \in \{a, b\}^*\}$$

(a) Give the first five strings of L in canonical (shortlex) order.

(b) For each $n \geq 0$, determine the future set L/a^n, i.e., the set of strings z such that $a^n z \in L$. Note that, for all $m \neq n$, it should hold that $L/a^m \neq L/a^n$.

(c) Is L regular? Explain your answer.

---

5. [14 pt] Let
$$L = \{a^i b^j a^k \mid i, j, k \geq 0 \text{ and } j < i + k\}$$
The first five elements in the canonical (shortlex) order of L are a, aa, aaa, aab and aba.

   (a) Consider the context-free grammar G_1 with start variable S and the following productions:

   $$S \to aS \mid XY \qquad X \to aXb \mid a \qquad Y \to bYa \mid A \qquad A \to Aa \mid \Lambda$$

   Intuitively, $S \to aS$ is responsible for additional a's to the left of the string, A is responsible for additional a's to the right of the string, and X and Y yield a's to the left and to the right of the string matching the b's.

   It is given that $L(G_1) \subseteq L$, i.e., G_1 only generates strings that are in L.

      (i) Does it also hold that $L \subseteq L(G_1)$? If yes, then you do not have to explain this. If no, then give a string x in L that is not in $L(G_1)$ and explain why x cannot be generated by G_1.

      (ii) Is G_1 ambiguous? If no, then you do not have to explain this. If yes, then give two different derivation trees for a string $x \in L(G_1)$.

   (b) Consider the context-free grammar G_2 with start variable S and the following productions:

   $$S \to A \mid T \qquad A \to aA \mid a \qquad T \to aTa \mid X \mid Y \qquad X \to aXb \mid aab \qquad Y \to bYa \mid baa$$

   Intuitively, A is responsible for strings consisting of only a's, T is responsible for strings containing at least one b, $T \to aTa$ adds a's on both sides of the string, and X and Y yield a's to the left and to the right of the string matching the b's.

   It is given that $L(G_2) \subseteq L$, i.e., G_2 only generates strings that are in L.

      (i) Does it also hold that $L \subseteq L(G_2)$? If yes, then you do not have to explain this. If no, then give a string x in L that is not in $L(G_2)$ and explain why x cannot be generated by G_2.

      (ii) Is G_2 ambiguous? If no, then you do not have to explain this. If yes, then give two different derivation trees for a string $x \in L(G_2)$.

---

6. [10 pt] Let $L = \{a^i b^j \mid 0 \leq i \leq j \leq 2i\}$, so the number of b's is between the number of a's and twice the number of a's (inclusive). Let G be the context-free grammar with start variable (and only variable) S, and the following productions:
$$S \to aSb \mid aSbb \mid \Lambda$$
In homework 3, you were asked to prove that any string $x \in L$ can be generated by G. Now, you are asked to prove the converse, i.e., that any string generated by G is indeed in L.

   To be concrete, let $x \in L(G)$, i.e., $x \in \{a, b\}^*$ and $S \Rightarrow^* x$. Use induction on the length (the number of steps) of the derivation of x to prove that x is in L.

---

7. [15 pt] Let

$$L = \{a^i b^j c^k \mid \; i, j, k \geq 0 \text{ and } j < i + k\}$$

Note that strings in L end with c^k, whereas in question 5, the strings end with a^k.

Draw a pushdown automaton M, such that $L(M) = L$. This pushdown automaton must be based directly on the properties of the language. It should, therefore, not be the result of a standard construction for, for example, converting a context-free grammar into a pushdown automaton.

Try to ensure that M is deterministic and does not contain any Λ-transitions. If you do not succeed in this, you can still earn most of the points.

Also explain how M uses its states and stack (symbols) to accept precisely the right language.

---

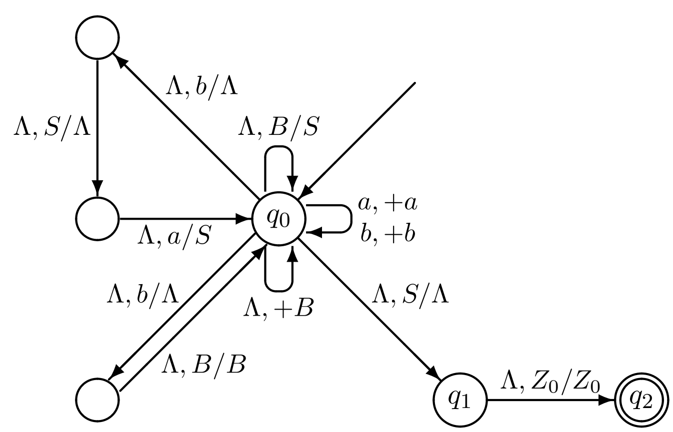
8. [14 pt] Let G be the context-free grammar with start variable S and the following productions:

$$S \to aSb \mid B \qquad B \to Bb \mid \Lambda$$

(a) Draw the non-deterministic top-down pushdown automaton $NT(G)$.

(b) Draw a derivation tree in G for the string *abb*.

(c) The non-deterministic *bottom-up* pushdown automaton $NB(G)$ looks like this:



Here, as usual, Z_0 is the initial stack symbol of $NB(G)$.

Carry out a successful computation in $NB(G)$ for input $x = abb$, i.e., a computation resulting in acceptance of x. Present this computation in a table of the following form:

state	stack (reversed)	remaining input	action
q_0	Z_0	abb	...
...	...	...	...

In the table, you may perform a reduction in one step, even if it actually requires a sequence of transitions of $NB(G)$.

`end of exam`