Languages and Automata

Final exam, 11 January 2019

8:30 – 10h30

This test consists of **5 exercises** over **6 pages**. Explain your approach, and **write your answers to the exercises on a separate folio (double pages) as indicated.** You can score a maximum of 100 points, and each question indicates how many points it is worth. The test is closed book. You are NOT allowed to use a calculator, a computer or a mobile phone. You should answer in English. Please write clearly, and do not forget to put on each page: your name and your student number.

Notation Throughout the test, we denote for any alphabet A, $w \in A^*$ and $a \in A$ by $|w|_a$ the number of a's in w, as it was introduced in the lecture. We denote the length of a word w by $|w|$.

Write your answers to Problems 1 and 2 on a separate folio (double page)

Problem 1.

For each of the following claims, state whether it is true or false. Give a brief explanation (no proofs needed) if it is true, and a counterexample if it is false. **(12pt)**

1. Suppose G is a context-free grammar. If the language $\mathcal{L}(G)$ is regular, then G must be a regular grammar.
2. The language $\{ww^R \mid w \in \{a,b\}^*\}$ is context-sensitive.
3. Let $L, K \subseteq A^*$ be languages over some alphabet A. If L is not regular, then also $L \cap K$ is not regular.

Solution: . . . .

1. False. For instance, the grammar $S \to aaS \mid \lambda$ is not regular, but the generated language is.
2. True. It is context-free (as we saw in the lecture) so also context-sensitive.
3. False. Take $L = \{a^n b^n \mid n \geq 0\}$, which is not regular, and $K = \emptyset$. Then $L \cap K = \emptyset$ is a regular language.

□

Problem 2.

Consider the following three languages over the alphabet $A = \{a, b, c\}$.

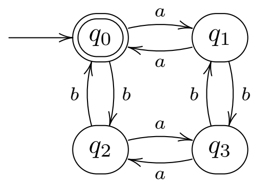
$$L_1 = \{a^i b^j c^k \mid i, j, k \geq 0 \text{ where } i = j \textbf{ or } j = k\}$$
$$L_2 = \{w \in \{a,b\}^* \mid |w|_a \text{ and } |w|_b \text{ are both even}\}$$
$$L_3 = \{a^i b^j c^k \mid i, j, k \geq 0 \text{ where } i = j = k\}$$

One of these languages is regular, one is context-free but not regular, and one is context-sensitive but not context-free.

a) Which language is regular? Show that it is, by giving a deterministic finite automaton which accepts it. Explain your answer. **(10pt)**

Solution: ....

The language L_2 is regular. DFA:

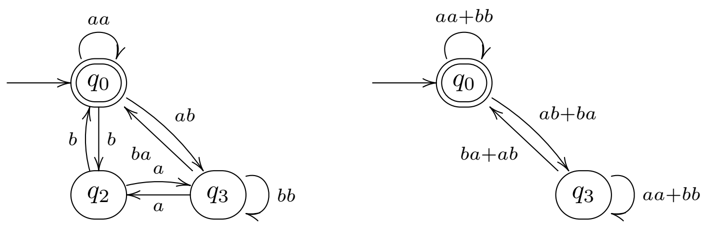


Explanation: while reading an input word, the number of a's read so far is even iff we are in q_0 or q_2, and the number of b's is even iff we are in q_0 or q_1. We should accept when both are even, so in q_0. □

b) Give a regular expression for the language which is regular (from the answer to the previous question). Explain your answer. **(10pt)**

Solution: ....

One possible way (which is not required) is to re-use the answer from the previous exercise, and use the state-elimination method, as follows:



resulting in

$$(aa + bb + (ba + ab)(aa + bb)^*(ab + ba))^*$$

□

c) Pick one of the other languages from the above three and show that it is not regular. **(10pt)**

Solution: ....

For L_1, one can use closure properties: if L_1 is regular, then also $L_1 \cap \mathcal{L}(a^*b^*) = \{a^n b^n \mid n \geq 0\}$ which is not regular, contradiction.

Alternatively, for L_1 we find the infinite collection of words $W = \{a, aa, aaa, \ldots\} = \mathcal{L}(aa^*)$, and show that, given $a^i, a^j \in W$ with $i \neq j$, a^i and a^j are distinguishable. Indeed, for such a^i, a^j, take b^i, then $a^i b^i \in L_1$ but $a^j b^i \notin L_1$ (note that it is important that $i, j > 0$, otherwise one of them might be equal to the number of c's).

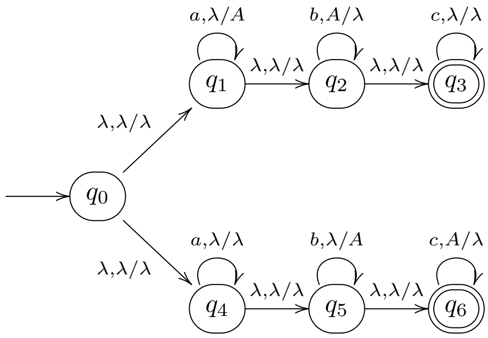
For L_3, take the infinite collection $V = \mathcal{L}(a^*)$. Given two arbitrary words $a_i, a_j \in V$ with $i \neq j$, they are distinguished by $b^i c^i$: we have $a^i b^i c^i \in L_3$, but $a^j b^i c^i \notin L$. Since there is an infinite collection of distinguishable words, L_3 is not regular.

□

d) Which language is context-free but not regular? Give a pushdown automaton recognising this language. Explain your answer. **(10pt)**

Solution: ....

L_1. Here's a PDA:



Explanation: in q_0, we make a choice: either the same number of a's and b's (by going up to q_1) or the same number of b's and c's (by going down to q_4). Of course, these could all be equal; then the choice doesn't matter. Now, in q_1 we read a's while putting the same number of symbols on the stack, proceeding in q_2 to read b's while popping elements from the stack. Finally in q_3 we can read an arbitrary number of c's. But note that the word will only be accepted in q_3 if the stack is empty, which means the number of b's was equal to the number of a's. The path down via q_4 is similar, but reading first (in q_4) an arbitrary number of a's, then reading b's while putting them on the stack (in q_5), and finally reading c's while popping from the stack (in q_6).

□

Write your answers to Problems 3 and 4 on a separate folio (double page)

Problem 3.

Let $A = \{a, b, c\}$.

a) Define, by induction, a function $f\colon A^* \to A^*$ that, given a word w, replaces every letter a in w by aca, every letter b by $babc$ and that removes every c. **(5pt)**
So, for instance, $f(abca) = acababcaca$ and $f(ccbba) = babcbabcaca$.

Solution: ....

We define f by:

$$\begin{aligned} f(\lambda) &= \lambda \\ f(aw) &= acaf(w) \\ f(bw) &= babcf(w) \\ f(cw) &= f(w) \end{aligned}$$

□

b) Show, by induction, that for all words $w \in A^*$: $|f(w)|_c = |w|_a + |w|_b$. **(8pt)**

Solution: ....

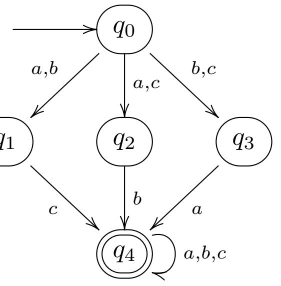
We prove this by induction on w.

- Base case. We have $|f(\lambda)|_c = |\lambda|_c = 0 = |\lambda|_a + |\lambda|_b$ as needed.
- Induction step.
  Suppose $|f(w)|_c = |w|_a + |w|_b$ holds for some $w \in A^*$. We prove that for any $x \in A$ it also holds for xw, that is, $|f(xw)|_c = |xw|_a + |xw|_b$. We proceed with a case distinction:
  - $x = a$:
  $$|f(aw)|_c = |acaf(w)|_c = 1 + |f(w)|_c \overset{IH}{=} 1 + |w|_a + |w|_b = |aw|_a + |aw|_b$$
  - $x = b$:
  $$|f(aw)|_c = |babcf(w)|_c = 1 + |f(w)|_c \overset{IH}{=} 1 + |w|_a + |w|_b = |aw|_a + |aw|_b$$
  - $x = c$:
  $$|f(cw)|_a = |f(w)|_a \overset{IH}{=} |w|_a + |w|_b = |cw|_b + |cw|_c$$

□

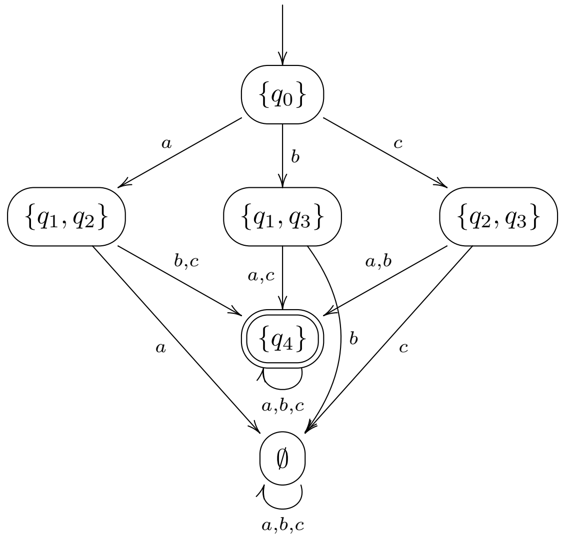
Problem 4.

Consider the following NFA M over the alphabet $A = \{a, b, c\}$:



Use the construction from the lecture to give a deterministic finite automaton M' over A such that $\mathcal{L}(M) = \mathcal{L}(M')$. Clearly mark how the states of M' correspond to the states of M. **(10pt)**

Solution: ....

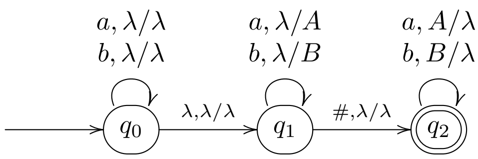


□

Write your answers to Problem 5 on a separate folio (double page)

Problem 5.

Consider the following pushdown automaton M over the alphabet $A = \{a, b, \#\}$, with stack alphabet $\Gamma = \{A, B\}$.



a) Show that the word $abab\#ba$ is accepted, but the word $ab\#ab$ is not accepted. **(5pt)**

Solution: ....

Here's an accepting computation for $abab\#ba$:

$$(q_0, abab\#ba, \lambda) \to (q_0, bab\#ba, \lambda) \to (q_0, ab\#ba, \lambda) \to (q_1, ab\#ba, \lambda) \to (q_1, b\#ba, A)$$
$$\to (q_1, \#ba, BA) \to (q_2, ba, BA) \to (q_2, a, A) \to (q_2, \lambda, \lambda)$$

Since q_2 is a final state and the stack is empty, the word is accepted.

For $ab\#ab$, we have several possibilities:

$$(q_0, ab\#ab, \lambda) \to (q_0, b\#ab, \lambda) \to (q_0, \#ab, \lambda) \to (q_1, \#ab, \lambda) \to (q_2, ab, \lambda)$$

and here we are stuck; or

$$(q_0, ab\#ab, \lambda) \to (q_0, b\#ab, \lambda) \to (q_1, b\#ab, \lambda) \to (q_1, \#ab, B) \to (q_2, ab, B)$$

and we are stuck again; or

$$(q_0, ab\#ab, \lambda) \to (q_1, ab\#ab, \lambda) \to (q_1, b\#ab, A) \to (q_1, \#ab, BA) \to (q_2, ab, BA)$$

and again we are stuck. These are all possible computations starting from $ab\#ab$ with an empty stack, so this word is indeed not accepted. □

b) Give a precise description of the language $\mathcal{L}(M)$ accepted by M using set-notation. **(10pt)**

Solution: ....

$$\begin{aligned} \mathcal{L}(M) &= \{uw\#w^R \mid u, w \in \{a,b\}^*\} \\ &= \{w\#v \mid w \text{ ends with the reverse of } v\} \end{aligned}$$

□

c) Give a context-free grammar G such that $\mathcal{L}(M) = \mathcal{L}(G)$. **(10pt)**

Solution: ....

$$\begin{array}{lcl} S & \to & aS \mid bS \mid T \\ T & \to & aTa \mid bTb \mid \# \end{array}$$

□