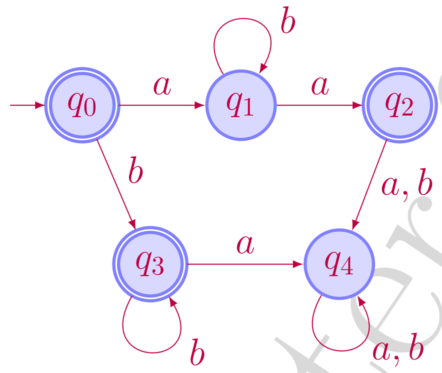
CPSC 313 (Winter 2016) L01

Final Sketch Solutions

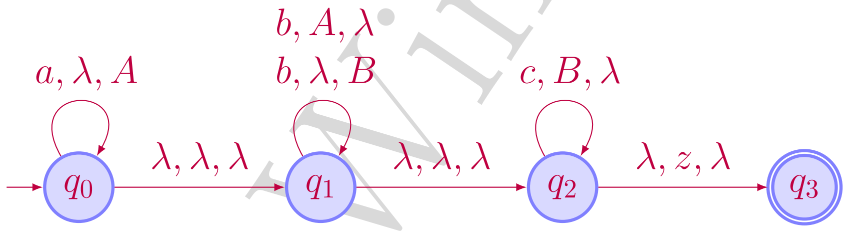
A 180 minutes exam held on April 18, 2016

1. Short answers

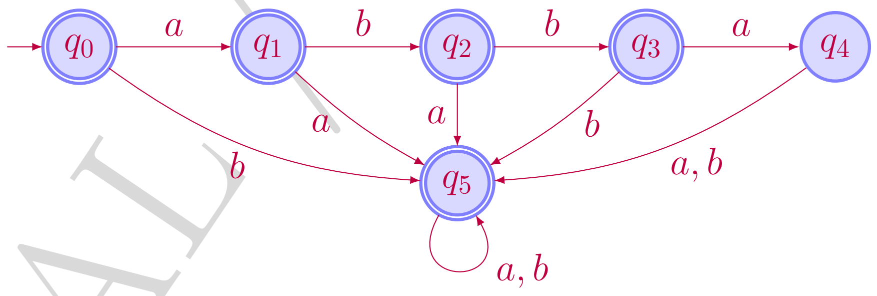
1. Give a minimal DFA for the language $L(\boldsymbol{r})$ where $\boldsymbol{r} = \boldsymbol{a}(\boldsymbol{b}^*)\boldsymbol{a} + \boldsymbol{b}^*$.



2. Give a PDA for the language $L = \{a^i b^j c^k \mid j = i + k \text{ and } i, j, k \geq 0\}$.



3. Give an NFA for the *complement* of the language $L = \{abba\}$ over the alphabet $\Sigma = \{a, b\}$.



4. Give a regular expression that is as simple as possible for the language
$L = \{w \in \{a, b\}^* \mid w \text{ does not contain the substring } aa \text{ nor the substring } bb\}$.

$\boldsymbol{r} = (\boldsymbol{a} + \boldsymbol{\lambda})(\boldsymbol{ba})^*(\boldsymbol{b} + \boldsymbol{\lambda})$

5. Let M be a Turing Machine that halts on all inputs in at most 20 steps. Explain why $L(M)$ is necessarily **regular**.

Whether an input string w is accepted by M or not depends only on the first 20 symbols in w. We first find all strings of length strictly less than 20 that are accepted by M, by simulation. We construct a regular expression $\boldsymbol{r}_<$ corresponding to those strings. We next find all strings of length exactly 20 that are accepted by M, by simulation. We construct a regular expression $\boldsymbol{r_{20}}$ corresponding to those strings. An input string w of length strictly larger than 20 is accepted by M if and only if the prefix consisting of the first 20 symbols in w is accepted by M. The language of M is then $L = L(\boldsymbol{r_<}) \cup L(\boldsymbol{r_{20}})\Sigma^*$.

6. Give a context-free grammar for the language $L = \{a^i b^j c^i d^k e^k \mid i, j, k \geq 0\}$.

   $S \to AD; \quad A \to aAc \mid B; \quad B \to bB \mid \lambda; \quad D \to dDe \mid \lambda;$

7. The following theorem is true, but its "proof" is flawed. Precisely and succinctly identify a flaw in the proof and explain why it is a flaw. If there are more than one flaw, identify and explain them all.

   Theorem 1 *The language $L = \{a^i b^i c^i d^i \mid i \geq 0\}$ is not context-free.*

   Proof Let $L_1 = \{a^i b^j c^j d^i \mid i, j \geq 0\}$. Then L_1 is context-free. Let $L_2 = \{a^i b^j c^i d^j \mid i, j \geq 0\}$. Then L_2 is not context-free, and hence $L_1 \cap L_2$ is neither context-free. But $L_1 \cap L_2$ is equal to L, and hence L is not context-free. $\square$

   There is one flaw: $L_1 \cap L_2$ might be context-free, even if L_2 is not. A counter-example is to pick $L_1 = \{a\}$ and $L_2 = \{a^i b^i c^i \mid i \geq 0\}$.

2. A language not context-free

Show that the language

$$L = \{a^i b^j c^k a^i b^j c^k \mid i \geq 0, j \geq 1, k = 1\}$$

is not context-free. You may, without proof, use the Pumping Lemma and the closure properties for the context-free languages.

Use the pumping lemma for context-free languages. Let $w = a^m b^m c a^m b^m c$, where m is as in the pumping lemma. Then $w \in L$ and $|w| \geq m$, and we can thus pump on w. Next, consider *all* of the possible decompositions of w into substrings u, v, x, y, z in turn.

3. Reduction

Let

$$\mathsf{Even} = \{\langle M \rangle \mid M \text{ is a TM and } L(M) \text{ contains only strings of even length}\}.$$

Show that $\mathsf{Halt} \leqslant_m \overline{\mathsf{Even}}$. That is, show that the halting problem reduces to the *complement* of the language Even.

The input to the reduction is $\langle M, w \rangle$, and the output of the reduction is $\langle M' \rangle$.

We construct a Turing Machine M' that ignores its input z, and instead does as follows. If the input $\langle M, w \rangle$ is not the proper encoding of a Turing Machine M and string w, then M' immediately halts and rejects. If the input $\langle M, w \rangle$ *is* the proper encoding of a Turing Machine M and string w, then M' simulates M on w. If M halts on w, the Turing Machine M' stops the simulation and accepts.

By construction, $L(M') = \Sigma^*$ if M halts on w, and $L(M') = \emptyset$ if M does *not* halt on w. Thus $L(M')$ contains a string of odd length if and only if M halts on w.

4. True or False

Answer True or False. The languages are over the alphabet $\Sigma = \{0, 1\}$.

	Question	True	False
1	If L is recursive, then so is the complement $\overline{L}$	✓	
2	If $L \notin \text{co-RE}$ then $L \in \text{RE} \cup \text{REC}$		✓
3	If L_1 is finite and $L_2 \in \text{RE}$ then $L_1 \cap L_2 \in \text{co-RE}$	✓	
4	$L = \{\langle M \rangle \mid M \text{ is a TM and } L(M) = \emptyset\}$ is infinite	✓	
5	If L is context-free then L is recursive	✓	
6	$L = \{\langle M \rangle \mid M \text{ is DFA that accepts an infinite language}\}$ is recursive	✓	
7	There is a recursive language L that is not regular	✓	
8	$\text{Halt} \leqslant_m \overline{\text{Equal}}$	✓	
9	$\text{Empty} \not\leqslant_m \text{Valid}$	✓	
10	$\text{Member} \leqslant_m \text{Member}$	✓	

5. Language classification (incl. Bonus)

Answer **exactly one** of four possible answers REC, RE, co-RE, None to each of the following languages. Answer REC if the language is recursive, answer RE if the language is recursive enumerable but not recursive, answer co-RE if the language is co-recursive enumerable but not recursive, and answer None if the language is none of the first three possible answers. No justification required. Below, M, M_1 and M_2 denote Turing Machines. The languages are over the alphabet $\Sigma = \{0, 1\}$.

	Question	REC	RE	co-RE	None
1	$\{\langle M, w\rangle \mid M$ is a TM that either loops on w or accepts $w\}$			✓	
2	$\{\langle M\rangle \mid M$ is a TM that has an odd number of final states$\}$	✓			
3	$\{\langle M, w\rangle \mid L(M) = \{w\}\}$				✓
4	$\{w \mid w = \langle M\rangle$ for some TM M and $w \in L(M)\}$		✓		
5	$\{\langle M\rangle \mid M$ is a TM that halts in a non-final state on some input$\}$		✓		
6	$\{\langle M\rangle \mid L(M) = L(\mathbf{1}^*)\}$				✓
7	$\{w \mid w$ is rejected by some TM $M\}$	✓			
8	$\{\langle M_1, M_2\rangle \mid L(M_1) \cap L(M_2) \neq \emptyset\}$		✓		
9	$\{\langle M, w_1, w_2\rangle \mid M$ is a TM that halts on w_1 and accepts $w_2\}$		✓		
10	$L = \{10110\}$	✓			
11	$\overline{\mathsf{Empty}} \cup \mathsf{Finite}$	✓			
12	$\{\langle G\rangle \mid G$ is a context-free grammar and $L(G) \neq \emptyset\}$	✓			

6. Reversal of context-free languages (Bonus)

The context-free languages are closed under reversal.

Theorem 2 *If L is context-free, then the reverse language L^R is also context-free.*

It is possible to prove this theorem using context-free grammars, by replacing each rule $A \to w$ with its reversal $A \to w^R$, where w^R denote string w reversed. Consider we instead want to prove the above theorem using PDAs. Here is our first flawed proof idea.

Proof [Flawed idea] Let $M = (Q, \Sigma, \Gamma, \delta, q_0, z, F)$ be a PDA such that $L(M) = L$. We construct a new PDA $M' = (Q, \Sigma, \Gamma, \delta', q_0, z, F)$ that is the same as M, except for the transition function δ'. Whenever we have a rule in M of the form $\delta(q_1, a, B) = (q_2, C)$ in δ, we include the rule $\delta(q_2, a, C^R) = (q_1, B^R)$ in δ'. In short, we replace each edge

$$\begin{tikzcd}
q_1 \arrow[r, bend left, "{(a, B, C)}"] & q_2
\end{tikzcd}$$

with its reversed edge

$$\begin{tikzcd}
q_1 & q_2 \arrow[l, bend right, "{(a, C^R, B^R)}"']
\end{tikzcd}$$

□

This proof idea almost work, but not quite. We need to make some additional adjustments to the proof. Explain what these additional adjustments are. Make your arguments succinct.

We make the following adjustments.

1. Create a new initial state q_0' in M' with λ-transitions to all of the old final states F in M.
2. Create a new final state q_f' in M' to which there is a λ-transition from the old initial state in M.
3. The old initial state in M is no longer an initial state in M'. The old final states in M are no longer final states in M'.
4. We are not permitted to pop more than one symbol at a time in a PDA. The string C^R might be longer than a single symbol. To pop a string from the stack, make a chain of edges, each of which pop exactly one symbol in C^R, linked together by intermediate dummy states. Introduce such chains for each rule in M, where needed.
5. The PDA M may accept with a non-empty stack. To simulate this, make a self-loop in $\delta(q_0', \lambda, \lambda) = (q_0', A)$ for each symbol $A \in \Sigma$ in the alphabet. This permits us to fill the stack with whatever symbols we want before leaving the new initial state.
6. The PDA M always starts with an empty stack. We need to verify that the stack in M' is empty after simulating M in reverse. Thus, modify the λ-transition in step 2 above to instead be $\delta'(q_0, \lambda, z) = (q_f', \lambda)$.

We now have a proper PDA M' that accepts L^R.

7. Closure property (Bonus)

Prove that if $L_1 \in \mathsf{REC}$ and $L_2 \in \mathsf{RE}$, then $L_1 \setminus L_2 \in \text{co-}\mathsf{RE}$. Make your proof succinct.

Since $L_2 \in \mathsf{RE}$, then $\overline{L_2} \in \text{co-}\mathsf{RE}$. Since $L_1 \in \mathsf{REC}$ and $\mathsf{REC} \subset \text{co-}\mathsf{RE}$, then $L_1 \in \text{co-}\mathsf{RE}$. Now, $L_1 \setminus L_2 = L_1 \cap \overline{L_2}$, both L_1 and $\overline{L_2}$ are in co-RE, and co-RE is closed under intersection.