CS 154 - Introduction to Automata and Complexity Theory
Solutions to Sample Midterm

The questions below are from previous exams in CS154. They should give you some idea of what kinds of questions to expect and what we want for solutions. The actual midterm won't be this long and will have a little more emphasize on decidability.

Problem 1. [20 points] Decide if the following statements about languages over $\{0, 1\}$ are TRUE or FALSE, and circle the right answer using the boxes provided on the side. *You must also give a brief explanation of your answer to receive full credit.*

	F

(a). If a DFA M has a loop then the language $L(M)$ is infinite.

FALSE – The loop might not be reachable from the initial state, or might not be possible to reach a final state from it.

	F

(b). There is a regular language L for which there is exactly one regular expression R with $L(R) = L$.

FALSE – For every regular expression R, $R + R$ accepts the same language.

	F

(c). Let L be a language and h a homomorphism. If $h(L)$ is regular, then L must be regular.

FALSE – Consider $L = \{0^n 1^n \mid n \geq 0\}$ and the homomorphism $h(0) = \epsilon, h(1) = \epsilon$.

T	

(d). Let L be a regular language, and L^R its reverse. The language $L \cdot L^R$ is regular.

TRUE – If L is regular then so is L^R (to see this consider a DFA for L, reverse all transitions, add a new initial state with an ϵ-transition to all previous final states, and make the previous intial state final.) The concatenation of two regular languages is another regular.

Problem 2. [30 points]

a). [15 points] Consider the following language over the alphabet $\Sigma = \{a, b, c\}$.

$$L = \{a^n b^p (c+b)^{n-p} \mid 1 \leq n \quad \text{and} \quad 1 \leq p \leq n\}.$$

Here, $(c+b)^{n-p}$ means a sequence of $n-p$ symbols from the set $\{c, b\}$.

Show that L is non-regular using closure properties. Do **not** use the pumping lemma, but you can use any language proven in the book to be non-regular (or regular).

Solution:

Consider the following homomorphism: $h(a) = a, h(b) = b, h(c) = b$. The image of L is $h(L) = \{a^n b^n |\ n \geq 1\}$ which is non-regular. Therefore L has to be non-regular.

An alternative solution is to intersect L with $L(a^*b^*)$, which is regular. The intersection is $\{a^n b^n |\ n \geq 1\}$, which is non-regular (contradicting that L is regular by properties of the intersection).

b). [15 points] Consider the following language L over the alphabet $\Sigma = \{a, b, c\}$:

$$L = \{a^m b^n c^k \mid m \geq n \geq 0 \ \wedge \ k \geq 0\} \cup \{a^k b^m c^n \mid k \geq 0 \ \wedge m \geq n \geq 0\}$$

Prove that the language L is non-regular using the pumping lemma. For your convenience we provide a template. Fill the blanks to complete the proof.

Solution: *Assume L is regular, and let $n > 0$ be a pumping lemma constant. Let $w =$* $\boxed{a^n b^n c^{n+1}}$ *be a string in L. Then, by the pumping lemma, there are x, y, z such that $xyz = w$ and $|xy| \leq n$ and $|y| \geq 1$. For every $k \geq 0$, $xy^k z$ is also in L. However, choosing $k =$* $\boxed{0}$ *we obtain a contradiction because*

Since $x = a^p$, $y = a^q$ (for $q > 1$), and $z = a^{n-p-q} b^n c^{n+1}$, then

$$xy^0 z = a^p a^{n-p-q} b^n c^{n+1} = a^{n-q} b^n c^{n+1} \notin L.$$

Problem 3. [20 points] Describe algorithms for the following decision problems. You may use any of the decision algorithms given in the class or the textbook as a "subroutine." As usual, you may assume any convenient representation (regular expression, DFA, NFA, or ϵ-NFA) of the regular languages provided in the input. Make sure that your answer clearly specifies when your algorithm returns YES or NO.

a). [10 points] Given regular languages L_1 and L_2, decide whether L_1 is a proper subset of L_2 (i.e. $L_1 \subset L_2$).

Solution: if $L_1 \cap \overline{L_2} = \emptyset$ and $\overline{L_1} \cap L_2 \neq \emptyset$ then YES. Otherwise NO.

b). [10 points] Given a regular language L over $\Sigma = \{0, 1\}$, does L contain all possible strings finishing in 0101?

Solution: Let L_2 be the language of all strings finishing in 0101, which is regular (hint: build an NFA for it). If $\overline{L} \cap L_2 = \emptyset$ then YES. Otherwise NO.

Problem 4. [40 points]

Register Finite Automata (RFA) are DFAs equipped with a m-digit register that counts decimal numbers using its m digits. We assume that $m \geq 1$. Informally, a RFA is a DFA that can additionally perform register operations on a transition. The operations on registers are *increment* and *decrement*.

Increment (INC) adds one to the decimal value currently stored. That is,

$$\textsc{inc}(x) = x \oplus_{10^m} 1,$$

where $(a \oplus_{10^m} b)$ means $(a + b \bmod 10^m)$. Note that performing an increment on the register with the value $10^m - 1$ ($i.e$, $999 \cdots 9$), results in the value 0 ($000 \cdots 0$).

Similarly, the decrement operation (DEC) decrements the value of the register by one. However, decrementing a register with the value $00 \cdots 0$ results in the value $999 \cdots 9$.

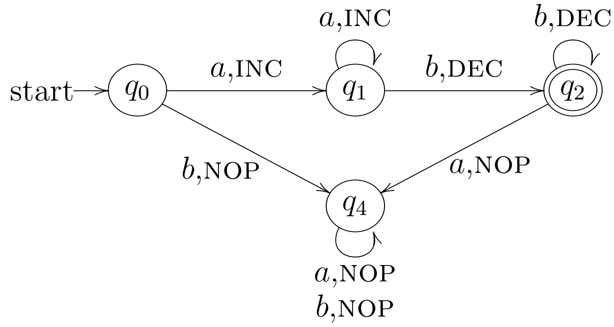
A RFA is tuple $\langle Q, \Sigma, \delta, r, q_0, F \rangle$, where

- Q is a set of states,
- Σ is the input alphabet,
- $\delta : Q \times \Sigma \mapsto Q$ is the transition function,
- $r : Q \times \Sigma \ \mapsto \ \{\textsc{inc},\ \textsc{dec},\ \textsc{nop}\}$ is a register operation,
- $q_0 \in Q$ is a start state,
- $F \subseteq Q$ is a set of final states.

The function r tells, for each state q and input letter a, whether the register is incremented $r(q, a) = \textsc{inc}$, decremented $r(q, a) = \textsc{dec}$, or alternatively no operation is performed, i.e., $r(q, a) = \textsc{nop}$. If the machine is in state q and input a is read then the state and register value changes according to δ and r, respectively.

Initially, the machine starts executing from q_0, with value 0 in the register. A machine accepts a word w if it reaches a final state $q_f \in F$, **and** has register value 0, after reading w.

a). [10 points] Consider the following RFA:



For your convenience, assume the alphabet is $\{a, b\}$ and the register has 1 digit, i.e, $m = 1$. Trace the "execution" of the string $aaabbbb$ through the RFA. Show, after each input, the state and the value of the register, by filling the template provided.

Solution:

$$(q_0, 0) \xrightarrow{a} \boxed{(q_1, 1)} \xrightarrow{a} \boxed{(q_1, 2)} \xrightarrow{a} \boxed{(q_1, 3)} \xrightarrow{b} \boxed{(q_2, 2)} \xrightarrow{b} \boxed{(q_2, 1)} \xrightarrow{b} \boxed{(q_2, 0)} \xrightarrow{b} \boxed{(q_2, 9)}$$

Is the string accepted by the RFA?

Solution:

No. In order to be accepted the reached configuration should be $(q_2, 0)$.

b). [10 points] Again assuming a 1 digit register, describe the language accepted by the RFA in part **(a)**. (Caution: Is $a^1 b^{11}$ accepted?)

Solution:

$$L = \{0^m 1^n \mid (m - n) \textit{ is divisible by } 10\} = \{0^m 1^n \mid m = n \bmod 10\}.$$

Assuming m digit register, we inductively define functions $\widehat{\delta}$ and $\widehat{r}$, such that $\widehat{\delta}(q, w)$ denotes the state reached by input w starting from state q, and $\widehat{r}(q, w)$ is an integer value that is added to the register starting from the state q on an input w. Formally,

$$\begin{aligned}
\widehat{\delta}(q, \epsilon) &= q \\
\widehat{\delta}(q, wa) &= \delta(\widehat{\delta}(q, w), a) \\
\widehat{r}(q, \epsilon) &= 0 \\
\widehat{r}(q, wa) &= \begin{cases} \widehat{r}(q, w) \oplus_{10^m} 1 & \text{if } r(\widehat{\delta}(q, w), a) = \text{INC} \\ \widehat{r}(q, w) \ominus_{10^m} 1 & \text{if } r(\widehat{\delta}(q, w), a) = \text{DEC} \\ \widehat{r}(q, w) & \text{if } r(\widehat{\delta}(q, w), a) = \text{NOP} \end{cases}
\end{aligned}$$

d). [10 points] **Prove** the following assertion:

"Every Regular language is accepted by a RFA."

Start from a DFA D and provide a construction of a RFA R such that $L(D) = L(R)$. No justification is required.

Solution: Given a DFA for a regular language, trivially convert it into a RFA by adding $r(q,a) = \textsc{nop}$, for all $q \in Q$ and $a \in \Sigma$.

e). [10 points] **Prove** the following assertion:

"Every RFA language is regular."

Start with an RFA $R = \langle Q_R, \Sigma_R, \delta_R, r, q_0^R, F_R \rangle$ with a register of m digits and construct a DFA $D = \langle Q_D, \Sigma_D, \delta_D, q_0^D, F_D \rangle$ that accepts the same language.

For your convenience you just have to fill the following table. No justification is required.

Solution:

Q_D	$Q_R \times \{0, \ldots, 10^m - 1\}$
Σ_D	Σ_R
δ_D	$\delta_D((q,x),a) = (\delta_R(q,a), x \oplus_{10^m} \widehat{r}(q,a))$
q_0^D	$(q_0^R, 0)$
F_D	$(q_f, 0)$ for $q_f \in F_R$

Problem 5. [15 points] Prove that the following language L_{comp} is non recursive:

$$L_{comp} = \{(\langle M_1 \rangle, \langle M_2 \rangle \mid L(M_1) = \overline{L(M_2)}\}.$$

The strings in L_{comp} encode two Turing Machines M_1 and M_2 such that the language of M_1 is the complement of the language of M_2.

Hint: First prove that the language L_{all} is non-recursive, where

$$L_{all} = \{\langle M \rangle \mid L(M) = \Sigma^*\}.$$

Solution:

First, note that L_{all} is not recursive by Rice's theorem, since $L = \Sigma^*$ is a non-trivial property of the recursively enumerable languages.

Now, we prove by reduction from L_{all} that L_{comp} is non-recursive. We can check whether M with alphabet Σ has $L(M) = \Sigma^*$ by the following:

1. Construct the TM M_e with alphabet Σ consisting of a single non-final state and no transitions. Clearly $L(M_e) = \emptyset$.
2. Check $\langle M_e, M \rangle \in L_{comp}$, since $\overline{L(M_e)} = \Sigma^*$

Hence, L_{comp} must not be decidable.