AC32008 Theory of Computation
Class Test 1 - Friday, 28 February 2025, 10:00-12:00
Answer ALL 5 Questions

Total marks: 30

1. (i) Give a formal definition of the concatenation between languages L_1 and L_2. **[3 marks]**

> **Solution:** Concatenation between languages L_1 and L_2 is a language that comprises all strings that are themselves a concatenation of some string from L_1 and some string from L_2. Formally,
>
> $$L_1L_2 = \{xy | x \in L_1, y \in L_2\}.$$

(ii) (True or False): Is it true that for any two languages L_1 and L_2, if $(L_1L_2)^* = L_1^*$, then $L_1 = L_2$? Explain your answer briefly. **[3 marks]**

> **Solution:** False. It easy to find a counterexample for this. For example, for $L_2 = \{\epsilon\}$, whatever language L_1 is, $L_1L_2 = L_1$ and, hence, $(L_1L_2)^* = L_1^*$. But, obviously, if L_1 is any language other than $\{\epsilon\}$, L_1 is not going to be the same as L_2. So, one counterexample are languages L_1 and L_2 over alphabet $\{0, 1\}$ where $L_1 = \{0, 00\}$ and $L_2 = \{\epsilon\}$.

2. (**Harder**) Prove that the language

$$L = \{0^n1^m | n, m \geq 0, n \neq m\}$$

is not regular. **[6 marks]**

[**Hint:** Do not try to use the Pumping Lemma on L directly. One way to prove this is to use properties of regular languages, e.g. if L_1 and L_2 are regular, then $L_1 \cap L_2, L_1 \cup L_2$ and L_1^C are also regular.]

> **Solution:** There are many ways to prove this. One way is to prove that L^C, the complement of the language L, is irregular. Recall that the complement of the language is the set of all strings over the same alphabet that are not in that language. *Note, however, that* L^C *is not* $\{0^n1^n | n \geq 0\}$*, because this leaves out many of the strings that are not in* L*. As an example,* 0101 *is not in* L*, but it is also not in the set* $\{0^n1^n | n \geq 0\}$*. But since* 0101 *is not in* L*, it has to be in* L^C*. Therefore, we also need to add all the strings that are not a sequence of* 0*'s, followed by a sequence of* 1*'s.* Therefore,
>
> $$L^C = \{0^n1^n | n \geq 0\} \cup L(\mathbf{0}^*\mathbf{1}^*)^C,$$
>
> where $L(\mathbf{0}^*\mathbf{1}^*)$ is the language described by the regular expression $\mathbf{0}^*\mathbf{1}^*$ (or, in other words, the language of all strings comprising a possibly empty sequence of

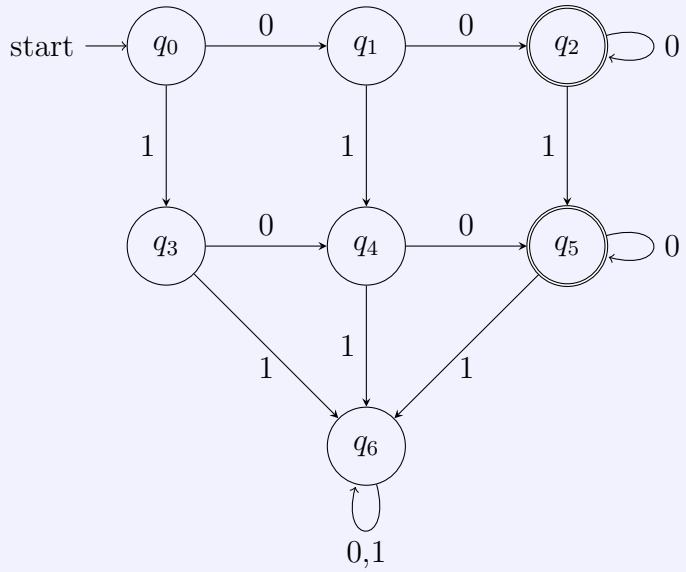
0's followed by a possibly empty sequence of 1's) We can now use the Pumping Lemma to prove that L^C is not regular - the proof is exactly the same as the proof that $\{0^n1^n|n \geq 0\}$ is not regular.

A very similar method is to note that $L^C \cap L(\mathbf{0}^*\mathbf{1}^*) = \{0^n1^n|n \geq 0\}$. Now, since we know that $\{0^n1^n|n \geq 0\}$ is not regular, and that $L(\mathbf{0}^*\mathbf{1}^*)$ is regular, we also know that L^C cannot be regular. If it was regular, then $L^C \cap L(\mathbf{0}^*\mathbf{1}^*)$, as an intersection of two regular languages, would also be regular. But this is impossible, since this language is the same as $\{0^n1^n|n \geq 0\}$, and we know this language is not regular.

3. Find a DFA that accepts the language over the alphabet $\{0, 1\}$ of all strings that have *at least* two 0's and *at most* one 1. Some strings that belong to this language are $10000000, 010, 0000010000$, whereas $00101, \epsilon$ and 10 do not belong to the language.

[5 marks]

Solution.



4. (i) Find a regular expression that describes the language of all strings over the alphabet $\{0, 1\}$ which have 1 at every third position, if it exists in the string (positions 3, 6, 9, 12 etc. from left to right). For example, strings 1, 101, 11111, 11101110100 belong to this language, while $000, 111100, 1101$ do not.

[3 marks]

Solution. Directly follows the construction of a finite automaton (with ϵ transitions) that accepts the language described by a given regular expression. This construction was covered in detail in lecture videos and notes, and on the tutorials.

5. (i) What are the differences in operation between an NFA and an NFA with epsilon

transitions? **[2 marks]**

Solution. NFA makes transitions only on reading a symbol of an input string, whereas an NFA with epsilon transitions can also make spontaneous (epsilon) transitions without reading any symbol of the string.

(ii) When do we say that a string w is accepted by an NFA? Give either informal answser or answer using transition function on strings of the NFA. **[3 marks]**

Solution. Informally, string w is accepted by an NFA if, starting from its starting state q_0, after reading the complete string w the NFA ends in at least one accepting state. Formally, w is accepted by NFA M if

$$\hat{\delta}(q_0, w) \cap F \neq \emptyset,$$

where q_0 is the starting state of M, F is the set of accepting states of M and $\hat{\delta}$ is the transition function of M on strings.