• You can do hard things! Grades do matter, but not as much as you may think, but then life is uncertain anyway, so what.

• Don’t cheat. You know the student code but more importantly you won’t be proud later.

• Please read the entire exam before writing anything. There are 7 problems and they are of equal value.

• This is a closed-book exam but you are allowed a 1 page (2 sides) cheat sheet that you have to upload along with your exam.

• You should write your answers on clean and clear white pages. After the exam is over you should upload a scan or a photo to Gradescope. If you have technical difficulties you can send the files via email to the course staff. Keep a copy of your written exam in case there are technical difficulties. Try to use different sheets for different problems to make it easier for processing.

• You have 150 minutes (2.5 hours) for the exam. This does not include the time for scanning the exam after finishing the writing. You have up to an additional half hour for scanning.

• Proofs are required only if we specifically ask for them.

• You may state and use (without proof or justification) any results proved in class or in the problem sets unless we explicitly ask you for one.

• Please contact the course staff if you run into difficulties with internet, Zoom, etc. You can send email or call/text the staff.
1 Regular Expression

Give regular expressions for the following two languages.

- $L_1 = \{ w \in \{0,1\}^* \mid w \text{ does not have the substring 111} \}$. The string 1101 is in $L_1$ but not 111101.
- $L_2 = \{ w \in \{0,1\}^* \mid \text{all 0 blocks of } w \text{ are of even length} \}$. Recall a 0 block is a maximal non-empty substring of 0's in the string. The string 0000100 is in $L_2$ but not 0011100.

Briefly explain your regular expressions.

2 DFA Construction

Draw or describe a DFA for the language defined below.

$$L = \{ w \in \{0,1\}^* \mid w \text{ ends in } 10 \text{ and } |w| \text{ is odd} \}.$$ 

Your DFA must have at most 6 states. Label the states and explain them.

3 Regular Expressions

Given a language $L$ over alphabet $\Sigma$ and string $x \in \Sigma^*$ we define the language $\text{insert}(L, x)$ as the set of all strings $w$ where $w$ is obtained by taking a string $w' \in L$ and inserting $x$ in $w'$ at some position. More formally $\text{insert}(L, x) = \{uxv \mid uv \in L \}$. For example if $L = \{CS374\}$ and $x = \text{not}$ then $\text{insert}(L, x) = \{\text{notCS374, CnotS374, CSnot374, CS3not74, CS37not4, CS374not}\}$. In this problem, assuming that $L$ is regular, you will derive an algorithm that generates a regular expression $r'$ for $\text{insert}(L, x)$ from regular expression $r$ for $L$.

- For each of the base case of $r = \emptyset, \epsilon$ and $r = a, a \in \Sigma$, write down the regular expression for $\text{insert}(L, x)$.
- Assume $r = r_1 + r_2$ and that $r'_1$ and $r'_2$ are regular expressions for $\text{insert}(L(r_1), x)$ and $\text{insert}(L(r_2), x)$ respectively. Write a regular expression for $\text{insert}(L(r), x)$ terms of $r, r_1, r_2, r'_1, r'_2, x$ (you do not have to use all of these).
- Assume $r = r_1 r_2$ and that $r'_1$ and $r'_2$ are regular expressions for $\text{insert}(L(r_1), x)$ and $\text{insert}(L(r_2), x)$ respectively. Write a regular expression for $\text{insert}(L(r), x)$ terms of $r, r_1, r_2, r'_1, r'_2, x$ (you do not have to use all of these).
- Assume $r = r^*$ and that $r'_1$ is a regular expression for $\text{insert}(L(r_1), x)$. Write a regular expression for $\text{insert}(L(r), x)$ in terms of $r, r'_1, x$ (you do not have to use all of these).

4 NFAs

Let $N_1 = (Q_1, \Sigma, \delta_1, s_1, A_1)$ and $N_2 = (Q_2, \Sigma, \delta_2, s_2, A_2)$ and $N_3 = (Q_3, \Sigma, \delta_3, s_3, A_3)$ be three NFAs. Let $n_1, n_2, n_3$ be the number of states in $N_1, N_2, N_3$ respectively. Describe an NFA $N = (Q, \Sigma, \delta, s, A)$ that accepts the language $(L(N_1) \cap L(N_2)) \cup L(N_3)$. For full credit your NFA $N$ should have $O(n_1 n_2 + n_3)$ states.

- Give a high-level description of how you can construct $N$ using procedures you have learnt in lecture or homework.
- Give a formal tuple notation for $N$. That is, you need to explain $Q, \delta, s, A$ in terms of the parameters of $N_1, N_2, N_3$. 


5  NFAs and Subset Construction

Consider the NFA $N$ shown below.

- Show that 1001 is accepted by $N$ by describing an accepting walk in the NFA.
- What is $\delta^*(s, \varepsilon)$?
- What is the $\varepsilon$-closure of $\{q_3, q_5\}$?
- Consider the subset construction to create a DFA $M = (Q', \Sigma, \delta', s', A')$ from $N$. What is $\delta'(X, 0)$ where $X = \{q_1, q_2\}$?
- Argue that 100 is not accepted by $N$ by computing $\delta^*(s, 100)$.

6  Non-regularity

Prove that the language $L = \{0^n | n \geq 0\}$ is not regular. You can use any proof technique you want. If you describe a fooling set for the language you need to justify the validity of the fooling set.

7  Regularity

Given a string $w$ a prefix is any string $x$ such that there is a string $y$ such that $x y = w$. A proper prefix of $w$ is a string $x$ such that there is $y$ with $|y| \geq 1$ such that $x y = w$. We will call a string $x$ a proper-proper prefix of $w$ if there is a string $y$ such that $|y| \geq 2$ and $x y = w$. If $w = abcde$ then $abc, ab, a,$ and $\varepsilon$ are proper-proper prefixes of $w$. For a language $L$ we define the language

$$\text{PPPREFIX}(L) = \{x \mid x \text{ is proper-proper prefix of some string } w \in L\}$$

Alternatively, $$\text{PPPREFIX}(L) = \{x \mid \exists y, |y| \geq 2, x y \in L\}.$$Suppose $L$ is regular. Prove that $\text{PPPREFIX}(L)$ is regular. In particular, given a DFA $M = (Q, \Sigma, \delta, s, A)$ for $L$ describe an NFA $N$ that accepts the language $\text{PPPREFIX}(L(M))$. 