

INHN0013 Information Theory & Theory of Computation

Exercise Sheet 3

Assigned: Wednesday October 29 2025

Due: Wednesday November 5 2025

You should know how to solve these problems and ideally you will solve them in the allotted one week. Some of the specific problems below might appear on the optional or actual final exam. You will see the solutions for instances that appear on the optional exams. Many of the problems (although not necessarily every single subproblem of every problem) will be discussed in the tutorials.

1. Prove or disprove that the languages below are regular:

- (a) $L_1 = \{0^m 1^n \mid m \neq n\}$
- (b) $L_2 = \{w \mid w \in \{0, 1\}^*\text{ and }w\text{ has twice as many 1's as 0's}\}$
- (c) $L_3 = \{w \mid w \in \{0, 1\}^*\text{ and the number of 0s in }w\text{ is a power of 2}\}$
- (d) $L_4 = \{1^k y \mid y \in \{0, 1\}^*\text{ and }y\text{ contains at least }k\text{ 1s, for }k \geq 1\}$

2. Prove or disprove the following claims:

- (a) Let L_1 be a regular language and L_2 a non-regular language. Then $L_1 \cap L_2$ is a non-regular language.
- (b) Let L_1 be a regular language and L_2 a non-regular language. Then $L_1 \cup L_2$ is a regular language.
- (c) Let L_1 be a regular language and L_2 a non-regular language. Then $L_1 \cup L_2$ is a non-regular language.
- (d) Let L_1 be a regular language and L_2 a non-regular language. Then $L_1 \cup L_2$ is a regular language.

3. For any string $w = w_1 w_2 \dots w_n$, the reverse of w , written w^R , is the string w in reverse order, $w_n \dots w_2 w_1$. For any language A , let $A^R = \{w^R \mid w \in A\}$. Prove or disprove the claim that if A is regular, then A^R is also regular.

4. Closure of context-free languages:

- (a) Use the languages $A = \{a^m b^n c^n \mid m, n \geq 0\}$ and $B = \{a^n b^n c^m \mid m, n \geq 0\}$ to show that the class of context-free languages is not closed under intersection.
- (b) Use part (a) and DeMorgan's law to show that the class of context-free languages is not closed under complementation.

5. Design CFGs for the following languages over alphabet $\{0, 1\}$:

- (a) $\{w \mid w \text{ contains at least 4 0s}\}$
- (b) $\{w \mid w \text{ starts and ends with the same symbol}\}$
- (c) $\{w \mid w \text{ starts and ends with a different symbol}\}$
- (d) $\{w \mid \text{the length of }w \text{ is odd}\}$
- (e) $\{w \mid w = w^R\}$
- (f) The empty set
- (g) $\{w \mid w \text{ has a 1 in every odd position}\}$
- (h) $\{w \mid w \text{ contains more 0's than 1's}\}$

6. Design a CFG for the language $L = \{a^i b^j c^k \mid i, j, k \geq 0 \text{ and } i = j \text{ or } j = k\}$. Is your grammar ambiguous? Why or why not?

7. Let G be a CFG in CNF. Prove that for every $w \in L(G)$ s.t., $n = |w| \geq 1$, we need exactly $2n - 1$ derivations to generate w .

8. Consider the CFG G below:

$$E \rightarrow aAbE | bBaE | \epsilon$$

$$A \rightarrow aAbA | \epsilon$$

$$B \rightarrow bBaB | \epsilon$$

- (a) Describe $L(G)$ using set notation (e.g., $L = \{w \in \{a,b\}^* \mid \dots\}$)
- (b) Convert G into CNF

9. Consider the grammar $G = (\{S\}, \{a,b\}, P, S)$ where P is given as follows:

$$S \rightarrow aSb$$

$$S \rightarrow aaSb$$

$$S \rightarrow \epsilon$$

- (a) Describe $L(G)$ using set notation (e.g., $L = \{w \in \{a,b\}^* \mid \dots\}$)
- (b) Is G ambiguous? If so give an example of a string with two different leftmost derivations (at every step, the leftmost variable is replaced).
- (c) If your answer in part (b) was “yes”, is it possible to find an unambiguous grammar for the same language? If so, do it; otherwise argue why not.
- (d) Convert G into CNF

10. For each of the languages over alphabet $\{a,b\}$ below, give a pushdown automaton accepting the language. You do not have to write formal proofs that your PDAs actually recognize the languages they are designed for. However, for each such automaton, you should give an informal description of how it works, as well as a transition diagram. To make your solutions easier to understand, your informal descriptions should refer to the specific states and transitions in your diagram when explaining how the PDAs work. That is, we need descriptions of the form

“... the PDA stays in state q_1 and pushes 0’s on the stack as long as the input contains 0’s, then nondeterministically jumps to state q_2 where it loops, popping a 0 off the stack whenever it sees a 1 in the input...”

rather than

“... the PDA pushes 0s on the stack as long as the input contains 0s, then starts popping a 0 off the stack whenever it sees a 1 in the input...”

- (a) $L_1 = \{w \mid w \text{ contains more } a\text{'s than } b\text{'s}\}$
- (b) $L_2 = \{w \mid w \text{ has odd length and its middle symbol is an } a\}$
- (c) $L_3 = \{w \mid w \text{ contains twice as many } a\text{'s as } b\text{'s}\}$
- (d) $L_4 = \{ww^R \mid w \in \{a,b\}^* \text{ and } w \text{ contains } aba \text{ as a substring}\}$