

Theory of Computation
CS 378
Fall 2022

Problem Set 0

Due **AT THE BEGINNING OF CLASS** Thursday, September 22

- Please read Chapter 0 and Chapter 1, pages 31–47, of Sipser.
- Standard file naming and HW submission conventions apply: Please submit your type-written answers in a PDF file named

CS378_PS0_<userid>.pdf

where <userid> is replaced by your Colby userid—for example, my file would be called CS378_PS0_eaaron.pdf. Please use your full userid, including your class year. Place the properly named file in your **Private** directory in your CS378 filespace on **filer**.

- *A general note:* When writing up your homework, please explain your arguments clearly and write neatly. Graders may not award credit to incomplete or illegible solutions. Clear communication *is* the point, on every assignment.
 - *A general note:* Neatly written (or typeset) solutions, with enough blank space on the page to allow graders to write comments before returning the papers, are greatly appreciated!
 - **NOTE:** You may not use any material other than your Sipser textbook (and lecture notes) for exercise 1 below.
1. Sipser, Exercise 0.7. For simplicity, your relations should use the domain $A \times A$ where $A = \{1, 2, 3\}$. (*Note:* You are being asked to give an actual relation for each part. Write your relations as sets of ordered pairs as on page 9 of Sipser.)
 2. One formal technique for proving that two sets X and Y are equal is to prove that $X \subseteq Y$ and that $Y \subseteq X$. To prove that $X \subseteq Y$, we must prove that for any x , if $x \in X$, then $x \in Y$. Similarly, to prove that $Y \subseteq X$, we must prove that for any y , on the assumption that $y \in Y$, it is also true that $y \in X$. For example, in Theorem 0.20, Sipser formally proves that $\overline{A \cup B} = \overline{A} \cap \overline{B}$ (although it could perhaps be argued that a few small details are missing from that proof!).

For this exercise, your task is to give a detailed, formal proof that $\overline{A \cap B} = \overline{A} \cup \overline{B}$. (This and the result of Theorem 0.20 are collectively known as *DeMorgan's Laws*.)
 3. Often in computer science and mathematics, we are given two different definitions of a structure and we need to determine if the two definitions are equivalent. In other words, we need to determine if the two definitions define the same thing. For example, consider the following two definitions of the set of strings of balanced parentheses:

Definition 1 *The strings of balanced parentheses is the set of strings w over alphabet $\{(,)\}$ such that*

- (a) *String w has an equal number of '('s and ')'s, and*
- (b) *Any prefix of w has at least as many '('s as ')'s.*

(Note: A prefix of string w is a substring of w beginning at the first symbol in the string. Thus, if $w = (())$ then $(, (, ((),$ and $((())$ are among the prefixes of w .)

Definition 2

- (a) *The empty string ϵ is balanced.*
- (b) *If x is a balanced string then (x) is also balanced.*
- (c) *If x and y are balanced strings then xy is also balanced.*
- (d) *Nothing else is a balanced string.*

Your task is to prove that these two definitions are equivalent.

- *Hint:* To do this, let A denote the set of strings defined by Definition 1 and let B denote the set of strings defined by Definition 2. We would like to show that $A = B$. To do this, we can show that $A \subseteq B$ and that $B \subseteq A$. Each of these will require a separate proof by induction and each inductive proof takes some work. Be very precise and rigorous in your proof. It will take some work and careful reasoning. In particular, an easy pitfall is to accidentally use some intuitive understanding of balanced parentheses instead of only appealing to the definitions and using rigorous proof.

As always for inductive proofs, please remember to clearly state what is being proved, prove the relevant base case, state the relevant inductive hypothesis, and prove the relevant inductive case.

- *Hint:* If standard mathematical induction doesn't work, try strong induction for inductive proofs! (See pg. 23 your Sipser textbook.) Also, recall that structural induction can be useful for proving that a property holds for all elements of a recursively defined set.
- *Hint:* When proving $A \subseteq B$, consider non-empty string w in A . What if w had some proper, non-empty prefix x that also had equal numbers of '('s and ')'s?

Note: This exercise may be challenging. Please get an early start on it, and as always, please feel free to see me with questions about it!

4. Give state diagrams for DFA's for each of the following languages over the alphabet $\{0,1\}$. For these exercises—and in general, when presenting machine diagrams in CS378—please give both a state-transition machine diagram and an English explanation for each DFA that explains the design and argues its correctness. Please be sure your answers contain enough information to unambiguously determine the full 5-tuple for each DFA! (One good way to do that is to explicitly give the 5-tuple, with δ defined by the diagram.)

- (a) $\{w \mid w \text{ has length at least 3 and an even number of 1's}\}.$
- (b) $\{w \mid w \text{ does not contain the pattern } 00 \text{ but does contain at least one occurrence of the pattern } 11\}.$
- (c) $\{w \mid \text{Every 3 consecutive symbols in } w \text{ contains at least one } 1 \}.$