

CSE105 (Winter 2014): Homework 1

Instructor: Daniele Micciancio

Due on Wednesday, January 15, 2014, in class.

This homework assignment consists of three sections. Section 1 is just a set of review/calibration questions to test your knowledge of the basic mathematical notation and definitions studied in previous courses. In Section 2, you are asked to put your basic math knowledge to work, to give mathematical definitions in the style used in this course. Notice: the problems hardly require any problem solving skills! They are primarily meant to test your ability to make proper use of mathematical concepts and notation. If you encounter any trouble, take it as a sign that you should invest a substantial amount of extra time at the beginning of the course to brush up your basic math knowledge. Section 3 contains some simple DFA design problems.

You should submit your solutions as follows:

1. A writeup for Sections 1 and 2. This should be submitted before the lecture, in class on the due date. You may write it by hand, or type it, but make sure that your writeup is neat and well organized.¹ Present your answers following the order in the assignment, and marking each answer with the appropriate section and part number.
2. Solutions to Section 3 should be submitted as a single (zipped) file through the turnin command on your class account. Detailed submission instructions will be posted on the course web page.

You will receive 0,1 or 2 points for each section, as described on the course webpage. Homework is primarily a way for you to get some practice, and improve your understanding, rather than evaluation and collection of fractional partial credit. You are encouraged to post questions on the discussion board, get help from other students and the course staff at office hours and discussion, etc., until you get it right. Of course, solutions must be your own, you should write them individually, and they should reflect your mastery of the material. We will try to provide feedback on the writeup portion of the assignment, which will be returned to you after grading.

1 Math Review

(a) Solve the following selection of exercises from the textbook. (As extra practice, you are encouraged to solve all exercises from Chapter 0.)

1. Exercise 0.2
2. Exercise 0.3
3. Exercise 0.9

(b) Consider the recursive program (or inductive function definition):

$$f(x) = \text{if } (x == 0) \text{ then return}(0) \text{ else return}(f(x - 1) + 2 * x - 1)$$

that takes as input a nonnegative integer $x \geq 0$, and outputs some integer $f(x)$.

Prove (by induction) that for every nonnegative integer x , $f(x) = x^2$.

¹If you submit hand written solutions, it is recommended that you follow the standard practice of first writing the solutions on scratch paper, and then copy them nicely for your final submission. Whether you type it or not, make sure you carefully read your solutions before submission to make sure they are clear and readable.

(c) (This is problem 0.12 from the textbook.) Prove that every graph with 2 or more nodes contains two nodes that have equal degree.

2 Modeling computation

(a) (This is Exercise 1.24 from the textbook.) Read the description of Finite State Transducer (FST) from Sipser Exercise 1.24, and answer the exercise questions **a.-h.**

(b) (This is Exercise 1.25 from the textbook.) Give a formal definition of this model as a mathematical object, following the pattern in Definition 1.5 from the textbook (or the lecture notes posted on the course web page.) Notice that FSTs (as described by Sipser in Exercise 1.24) may use different alphabets for input and output. For example, T_2 uses input alphabet $\Sigma = \{a, b\}$ and output alphabet $\Gamma = \{0, 1\}$. Also, FSTs do not have a set of accept states.

(c) A FST computes a function $f_T: \Sigma^* \rightarrow \Gamma^*$ mapping strings (over the input alphabet Σ) to strings (over a possibly different output alphabet Γ). Give a formal definition of the function f_T computed by a FST (as specified in parts (a) and (b) above) following the pattern used in class to define the function $f_M: \Sigma \rightarrow \{0, 1\}$ computed by a DFA. (See Section 1.1 of the Lecture Notes on DFA/NFA as a guideline.)

(d) The FSTs as described by Sipser in Exercise 1.24, at every step (i.e., upon reading any input symbol) output precisely one output symbol. As a result, on input a string of length n , an FST always outputs a string of length n . For many interesting applications of FSTs, this is too restrictive. Consider a more general kind of (deterministic) FSTs that still reads one input symbol at a time, but at every step may output 0,1 or more symbols from the output alphabet. How would you modify the definition you gave in part (b)? Provide a formal definition of this more general kind of FSTs (as usual, in the style of Definition 1.5 in the textbook, the lecture notes, and part (b) above.)

3 DFA design

For each of the following languages (over the alphabet $\Sigma = \{0, 1\}$), design a corresponding DFA. Test your DFA using jflap on a set of strings of your choice, and save it to a corresponding file `hw1a.jff`, ..., `hw1e.jff`.

(a) The set of strings that start with 0

(b) The set of strings that end with 0

(c) The set of strings of even length

(d) The set of strings of even length that start with 0 and end with 0, i.e., the intersection of the languages in parts (a), (b) and (c).

(e) All strings of the form $x_1y_1z_1x_2y_2z_2 \dots x_ny_nz_n$ such that if $x = \sum_k 2^{k-1}x_k$ (i.e., x is the integer with binary representation $x_nx_{n-1} \dots x_1$), and similarly for $y = \sum_k 2^{k-1}y_k$ and $z = \sum_k 2^{k-1}z_k$, then $z = x + y$. (Notice that all strings in this language have length $3n$ for some $n \geq 1$.)