CSc 30400 Introduction to Theory of Computer Science

4th Homework Set

Due Date: 4/22

Instructions:

- This homework set covers Chapters 3 and 4 (Computability).
- Submit your solutions by email or hand them in class **before the beginning of the class!**
- In your solutions you might use JFLAP if you find it convenient (for example for constructing transition diagrams) but you don't receive extra credit for using it. P1 examines if you understand how Turing Machines work. If you still have problems with that you can find it helpful to run a Turing Machine on JFLAP. However you won't be tested on whether you know how to use JFLAP but on your ability to understand how Turing Machines work, so make sure that you spend some time testing the machines by hand.
- Exercises are divided into three sections: Practice, Easy and Hard. Practice questions just help you understand the material and are a good way to get easy points. "Easy questions" should be relatively easy but require more effort or understanding than the practice ones! Hard questions require even more understanding than the easy ones.
- Grading details appears in the beginning of each question. The maximum amount of points you can get is 80.

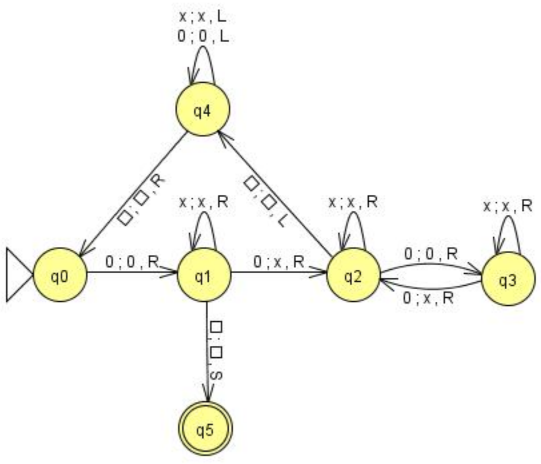
1 Practice questions

P 1. (3 points) Run the TM of the figure by hand to check whether it accepts for the following input strings or not.

(a) 000000

(b) 0000000

(c) 00000000



2 Easy questions

E 1. (15 points) As you know, we generally avoid giving full low-level descriptions of Turing machines, because this usually requires a lot of effort. Instead, we give a high-level algorithmic description. This description usually contains instructions which we are confident enough that a Turing Machine can implement. Then we combine smaller machines together to form larger ones capable of performing more complicated things. Your task for this problem will be to design partial Turing machines to show that such commands can indeed be implemented easily. For the remainder, the input alphabet is $\Sigma = \{a, b\}$ and the tape alphabet Γ contains $a, b, \Box$ and any other symbols that you want.

Design Turing machines that perform the tasks given below. When the task is done your machine should move to an accepting state (in this question the actual transition diagrams are required).

(a) Move the head to the last letter of the input.

(b) Replace all as with bs.

(c) Move the whole input one block to the right.

E 2. (5 points) Consider the language $L_w = \{w\#w : \ w \in \{0,1\}^*\}$. Show that L_w is a decidable language. Give high-level description of a Turing Machine deciding A (make sure that the description is directly implementable).

E 3. (5 points) Prove that Recursive languages are closed under complement (you might want to use the book's definition of a Turing Machine for this question).

E 4. (12 points) True or false? (Justify your answer)

(a) If for two sets A, B we have that $A \subsetneq B$ then $|A| < |B|$.

(b) If B is a countable set, A is an infinite set and $A \subseteq B$ then A is countable.

(c) If A, B are countable sets then $A \cup B$ is also countable.

(d) If A is a countable set and B is a finite set then B^A (that is, the set of functions from A to B) is countable.

E 5. (5 points) Consider the following situation. You are a professor of a programming course and you assign homework to your students to write a program performing some required task (for simplicity assume that the task is to address a specific yes-no problem - the program should output "yes" if the answer for the problem is "yes" for the given input and "no" otherwise). You would ideally like to automate grading by letting a computer program check whether a student's solution is correct: you would first write a correct program addressing the problem of the assignment. Then you would program a comparer, a program that compares the student's program with yours and decides whether the two programs agree on every input.

Is this doable? Why?

3 Hard questions

H 1. (10 points) The LR-DTM (left-right Turing machine) is exactly similar to the Turing machine we have described with the only difference that in each transition the head must move either to the right or to the left (there is no “Stay” move). Prove that LR-DTM are equivalent with DTM.

Remember that equivalence means that you have to prove two directions!!!

H 2. (10 points)

(a) Prove that $\mathbb{N} \times \mathbb{N} \times \mathbb{N}$ is countable.

(b) Show that $2^{\mathbb{N}}$ (that is the set of all subsets of $\mathbb{N}$) is uncountable.

H 3. (10 points) The following construction attempts a proof for showing closure of Recursive and Recursively Enumerable Languages under union.

Suppose that there are two Turing Machines M_1 and M_2 (we assume that both TM have unique accept and reject states that when reached the machine halts immediately and either accepts or rejects respectively). We create a Turing Machine M which simulates the following:

For a given input string x:

Run M_1 on x.

- If M_1 accepts M should accept.
- If M_1 rejects:
  Run M_2 on x.
  - If M_2 accepts M should accept.
  - If M_2 rejects M should reject.

Answer the following questions (justify your answer)

(a) If M_1 decides L_1 and M_2 decides L_2, does M decide $L_1 \cup L_2$?

(b) If M_1 recognizes L_1 and M_2 recognizes L_2, does M recognize $L_1 \cup L_2$?

H 4. (5 points) Consider the following predicate

$$Halt_{1000}(< M >, x) = \begin{cases} 1, & M(x) \text{ halts after performing 1000 transitions;} \\ 0, & \text{else.} \end{cases}$$

Is $Halt_{1000}$ a computable predicate? Why?