

Homework 08

Due: Tuesday, November 28, 2006

Instructions

1. Please review the homework grading policy outlined in the course information page.
2. On the *first page* of your solution write-up, you *must* make explicit which problems are to be graded for regular credit, which problems are to be graded for extra credit, and which problems you did not attempt. Use a table that looks like this:

Problem	1	2	3	4	5	6	7	8	9	...
Credit	RC	RC	RC	EC	RC	EC	NA	NA	EC	...

where “RC” denotes “regular credit”, “EC” denotes “extra credit”, and “NA” denotes “not attempted”. Failure to include such a table will result in an arbitrary set of problems being graded for regular credit, no problems being graded for extra credit, and a 5% penalty assessment.

3. You must also write down with whom you worked on the assignment. If this varies from problem to problem, write down this information separately with each problem.

Problems

Required: 4 of the following 5 problems

Points: 25 points per problem

1. Consider the language

$$TRIVIAL_{TM} = \{\langle M \rangle \mid M \text{ is a TM and } L(M) \text{ is either } \Phi \text{ or } \Sigma^*\}$$

That is, a TM is considered *trivial* if it always accepts its input or never accepts its input.

- a. Construct a mapping reduction $A_{TM} \leq_m TRIVIAL_{TM}$ that works as follows: It should take as input any $\langle M, w \rangle$, where M is a TM and w is a string, and produce as output an encoding $\langle M' \rangle$ of a TM M' that behaves in one of two ways, depending on the input $\langle M, w \rangle$: If M accepts w , then M' should accept every input given to it; if M does not accept w (either by rejecting or looping), then M' should accept just the single string 001 and no others. Explain why this is a valid mapping reduction.
- b. Create a Scheme implementation like those in the directory <http://www.ccs.neu.edu/home/rjw/csu390-f06/LectureMaterials/SchemeDemos> that models this mapping reduction, and run some experiments demonstrating its behavior. Use DrScheme (with language set to either Intermediate Student with lambda or Advanced Student), and turn in printout of both the Definitions window and the Interactions window. You should provide test results illustrating the behavior of the function produced as output when: (i) M accepts w ; (ii) M rejects w ; and (iii) M loops on w . Add appropriate annotation (handwritten is fine) clearly describing exactly what the individual interactions in the Interactions window illustrate. Create tests just like those in the files in the SchemeDemos directory. The best way to do this is to just copy one of those files and make as few changes as necessary to implement and test your function. Much of what's already in those files, including the data definitions and functions to use as M for testing, can be left as is.

2. Do Exercise 5.1 using a mapping reduction. *Hint:* Use the result stated in Theorem 5.13, even though we didn't (and won't) cover this result or its proof in class.
3. Do Problem 5.9 using a mapping reduction.
4. Use a mapping reduction to prove that

$$DECIDER_{TM} = \{\langle M \rangle \mid M \text{ is a TM and } M \text{ is a decider}\}$$

is undecidable.

5. Prove that $REGULAR_{TM}$ (defined in class and on p. 191 of Sipser) is neither Turing-recognizable nor co-Turing-recognizable. *Hint:* For part of this you should create one new mapping reduction.

For any of these problems, where appropriate you should cite and build on (and not re-derive) any results already provided in the assigned readings or in class.