Introduction to Complexity Theory – CS-28100
Homework 2 – April 5, 2006
Instructor: Ketan Mulmuley Ry-165B

HOMEWORK. Please **print your name on each sheet.** Please try to make your solutions readable.

This homework is due on **Wednesday, April 12** at the **beginning of the class.**

2.1 In this question, we would like you to write out all the rules for the Turing machines in full detail. You should also provide a brief high-level explanation of the meaning of the various states and how your program works.

  a Write a Turing machine that recognizes the language of palindromes over $\{0, 1\}$. (A palindrome is a string equal to its reversal.)

  b Write a Turing machine that adds two $(0, 1)$ strings in binary. (This is probably easier on a multitape machine.)

  c Write a Turing machine that multiplies two $(0, 1)$ strings in binary. (You will probably want to use the answer to the previous question as a subroutine.)

2.2 We have a list $a_1, a_2, \ldots, a_N$ of integers. We would like to be able to perform the "indexing" operation, i.e., given n and the list, find a_n.

For this quesion, your input alphabet will be {'(','0','1',',',')'}. (0,1,parentheses, and a comma). You will be given input L which consists of pairs of binary strings written in the following format:

$$(0100011, 0101010), (101000, 10101000), \ldots$$

Write a multitape Turing machine that, given this input L on one tape and a $\{0, 1\}$ string x on another tape, searches for a pair in which the first element is x and writes the second element of that pair on a third tape. If it cannot find the string x as a first element, it should go into an error state.

2.3 Prove that any Turing machine can be simulated by a Turing machine with only one state. (Hint: consider the instantaneous description.)

2.4 Prove that any Turing machine can be simulated by a Turing machine with only two (non-blank) letters in its alphabet.

2.5 In class, we showed how we can emulate a multitape Turing machine with a single-tape turing machine. Prove that this simulation is efficient in the following sense: if a multitape Turing machine performs a computation in T steps, our simulation will perform at most $O(T^2)$ steps.