CSE 431
Introduction to Theory of Computation
Homework #1
Due: Friday, April 9, 2010

W. L. Ruzzo 2 April 2010

Homework:

1. 3.2e
2. 3.6
3. 3.9 Recall from 322 that we have various examples of languages that are *not* context-free languages, i.e., are not accepted by any 1-PDA. $\{a^n b^n c^n | n \geq 0\}$ and $\{w\#w | w \in \{0,1\}^*\}$ are two standard examples, either of which could be used in this problem.
4. I've claimed that Turing Machines are equivalent to ordinary computers, but haven't proved it, so I'll let you do it.
   (a) Sketch briefly how you would program a TM simulator on an ordinary computer. In particular, how would you keep track of the TM's tape contents? State? How would they be updated for each step? [I don't need a lot of detail here; probably 1/2 page is adequate. You may assume your computer's memory is big enough to hold the TM's tape, or at least the finite portion of it that the TM will visit during any finite computation.]
   (b) My idea of an "ordinary" computer is very simple: It has memory words $0, 1, 2, \ldots$ holding integers. I don't want to fuss about how many bits a word has (this is a theory class, after all), so let's assume a "word" can hold any integer, no matter how big (but finite, of course). It has only three kinds of instructions — ADD, SUBTRACT, BRANCH-IF-GREATER-THAN-ZERO. ADD/SUBTRACT are 3-operand instructions, e.g. ADD i, j, k will add word i to word j, storing the result in word k. There are also literal and indirect addressing modes. E.g. ADD $i, =j, \uparrow k$ will add the contents of word i to the literal value j, storing the result in word whose address is stored in word k. An n-bit input is presented by setting word 0 to n, and putting the input bits in words 1 through n. All other words are initially zero. Your program is *not* stored in memory; assume it's in a separate execute-only memory. Is this model sufficiently powerful to do the simulation of a TM you sketched in part (a)? Sketch some of the tricks you'd need to do this. [Again, just a paragraph.]
   (c) Now sketch how a "computer" as defined in part (b) could be simulated by a Turing Machine. You will probably find it convenient to assume your TM is a multi-tape TM. Give both high-level and implementation-level descriptions, as defined in section 3.3 of the text. I estimate this will take 1-2 pages. If you're answer differs greatly from this, please contact us.
5. 3.18
6. 3.19