Homework 7: Turing machines

CSE 30151 Spring 2016

Due 2016/03/29

Instructions

Please note that you will **lose one point** if you don't follow these instructions.

- You can prepare your solutions however you like, but you must submit them as a single PDF file.
- Please name your PDF `netid-hw7.pdf`, where `netid` is replaced with your NetID, or `netid-hw7-1234.pdf`, where `1234` is replaced with the problems you are submitting.
- If you use the same name twice, only the most recent version will be graded!
- Submit your PDF file in Sakai. Don't forget to click the Submit (or Resubmit) button!

Problems

1. Write a **formal description** of a Turing machine that decides the language

   $$\{\mathtt{a}^n\mathtt{b}^n\mathtt{c}^n \mid n \geq 0\}.$$

   You may use the convention that if a state doesn't have a transition for some symbol, an implicit transition to the reject state is understood.

2. Write an **implementation-level description** as defined in Section 3.3 of a Turing machine that decides the language

   $$\{\mathtt{a}^n \mid n \text{ is a Fibonacci number}\}.$$

3. **Doubly infinite tapes.** [Problem 3.11] A Turing machine with a doubly infinite tape is like a TM as defined in the book, but with a tape that extends infinitely in both directions (not just to the right). Initially, the head is at the first symbol of the input string, as usual, but there are infinitely many blanks to the left. Show how, given a TM with doubly infinite tape, to construct an equivalent standard TM. An **implementation-level description** in the style of Proof 3.13 is fine, and it's also fine to use any results proved in the book or in class.

4. **Brain fun.** This problem is about a programming language known as $\mathcal{P}''$ in polite company.[1] A $\mathcal{P}''$ program has a half-infinite tape like a Turing machine, with each cell containing a symbol drawn from $\{a_0 \dots, a_n\}$. The tape is initialized to the input string followed by infinitely many a_0's. The language has four one-character commands and a looping construct:

$<$	Move the head to the left (if possible).
$>$	Move the head to the right.
$+$	Increment the symbol under the head. (a_n becomes a_0.)
$-$	Decrement the symbol under the head. (a_0 becomes a_n.)
[*cmds*]	While the symbol under the head is not a_0, execute *cmds* and repeat.

Let's say that when the program finishes, if the current square has a symbol other than a_0, the program accepts the input string; otherwise it rejects.

Show how a $\mathcal{P}''$ program can be compiled into a TM.

(a) Note that $\mathcal{P}''$ has sequential execution and while loops like C does. In a few sentences, describe intuitively how to simulate this in a TM.

(b) Show how, given a $\mathcal{P}''$ program, to construct the **formal description** of an equivalent TM. You don't need to prove the correctness of your construction. Recall that we added an N ("no move") action to the definition of TM.

(c) Optional, not for credit: How might a TM be converted into an equivalent $\mathcal{P}''$ program?

[1] `http://bit.ly/pprimeprime`