University of Washington

CSE 322: Introduction to Formal Models in Computer Science

Homework #7

Due: Wednesday, June 5, 2002, 10:30am

Spring 2002 May 31, 2002

Written homework is due at the *beginning* of class on the day specified. Any homework turned in after the deadline will be considered late. **Late homework policy:** You may turn in your homework after the deadline and before 5pm on the day it was due, but at a cost of a **20% penalty**. No homework will be accepted after 5pm on the due date.

Please staple all of your pages together (and order them according to the order of the problems below) and have your name on each page, just in case the pages get separated. Write legibly (or type) and organize your answers in a way that is easy to read. Neatness counts!

For each problem, make sure you have acknowledged all persons with whom you worked. Even though you are encouraged to work together on problems, the work you turn in is expected to be your own. When in doubt, invoke the *Gilligan's Island* rule (see the course organization handout) or ask the instructor.

* * *

Regular problems (to be turned in) **:**

1. Prove that the language $L = \{ 0^n \mid n \text{ is prime} \}$ is not context-free by using the pumping lemma for context-free languages.
2. Sipser, Problem 3.8a. Either a state diagram or a sufficiently detailed description of the machine (like those in the examples in the text) is good enough.

* * *

Bonus Problem (optional):

1. Describe how to construct a Turing machine that recognizes the language L from Problem #1 of the Regular problems. Use as much detail as necessary so that someone who knows how a Turing machine works could work out the details of the actual machine, but not so much detail that you end up programming every state and transition.

* * *

Suggested problems (highly recommended, but not to be turned in) **:**

1. Sipser, Exercise 3.1, 3.2, 3.5.
2. 3.7.
3. 3.8b, c.
4. Variation of Regular Problem #2. Suppose the language were $\{ w \mid w$ contains an equal number of 0s, 1s, and 2s $\}$ and construct a Turing machine that recognizes it.
5. Construct a Turing machine that, when presented with the input 01^n on the tape (for any $n \geq 0$) and head on the leftmost tape symbol, halts in the accept state with 01^{2n} on the tape and the head on the leftmost tape symbol. (In other words, it "computes" $2n$,

given *n*. Thought question: Why is it convenient to have the 0 as part of the input and output?)

6. Describe how a PDA with two stacks (not just one) could be used to simulate a Turing machine.
7. On page 125 of Sipser, it says:

   "We now turn to a much more powerful model, first proposed by Alan Turing in 1936, called the ***Turing machine***. Similar to a finite automaton but with an unlimited and unrestricted memory, a Turing machine is a much more accurate model of a general purpose computer. A Turing machine can do everything that a real computer can do."

   Let us consider these statements. Every computer ever built (and indeed any computer that will ever be built) has a finite amount of memory. Yes, it may be a large amount of memory, but it is still finite. So, in fact, for any computer that has or will ever exist, there is a finite automaton that *exactly* models its behavior. (For example, if a PC has 1,000,000 bits of memory, then it can be in one of $2^{1,000,000}$ states, which could all be states in a very large DFA.) So why does Sipser say that Turing machines are a much more accurate model of a general purpose computer? Indeed, why should we even bother to study pushdown automata or Turing machines when finite automata are powerful enough to model any real computer?