CSC364 Summer 2004 — Homework 1

The following questions are assigned each week. You need only hand in those questions marked with a star * as part of your assignment, and only those questions will be marked. Make sure you include a completed and signed cover page for each problem.

Week 1

1. To illustrate how asymptotic notation can be used to rank the efficiency of algorithms, use the relations $\subset$ (strict subset) and $=$ (set equality) to put the *orders* of the following functions into a sequence:

$$2n^5 \qquad 8^{\log_2 n} \qquad 12n^{2n} \qquad 3^{(n^2)} \qquad 2^{100^{50}} \qquad n^2 \log_2 n \qquad 3\sqrt{2n}$$

Do not use the relation $\subseteq$. Prove your answers. You need not prove relationships implied by transitivity.

*2. (hand in) Prove or disprove each of the following statements.

(a) $15n^2 - 12n \in \Theta\left(n^2\right)$
(b) $n! \in \mathrm{O}\left(n^n\right)$
(c) $6n^4 + 14n^3 \in \Omega\left(n^3\right)$
(d) $6n^4 + 14n^3 \in \Omega\left(n^4\right)$
(e) $n^2 \log n \in \Theta\left(n^2\right)$
(f) $\frac{n}{\log n} \in \mathrm{o}\left(n\right)$
(g) $n^{1.001} + n \log n \in \Theta\left(n^{1.001}\right)$
(h) $n^3 2^n + 6n^2 3^n \in \mathrm{O}\left(n^3 2^n\right)$
(i) $n\sqrt{n} + n \log n \in \mathrm{O}\left(1.001^n\right)$

3. Let a^n denote a string of length n characters, each character being an a. Write a Turing Machine to recognize strings of the form $a^{2n}b^n$, where a and b are elements of the input alphabet Σ and n is some nonnegative integer. Explicitly give your machine's alphabet, set of states, and transition function. Prove that your TM is correct.

Week 2

*4. (hand in) Write a Turning Machine which takes a positive integer p as input and determines whether p is a power of 2. In other words, the machine is to accept if and only if $p = 2^k$ for some integer k. Explicitly give the structure of your TM, your input and tape alphabets, input encoding, set of states (commenting on the meaning of each state as necessary), and transition function. Argue the correctness of your TM (you should convince the grader you are correct though a full formal proof is not required).

5. For a language $\mathcal{L}$, we define an enumerator to be a Turning Machine which writes each string in $\mathcal{L}$ to its tape exactly once, strings separated by some symbol #. For finite languages, the TM will halt, but for infinite languages the TM will continue generating new strings forever. Note that an enumerator does not take input, but may use part of the tape as work space.

   Consider $\mathcal{L}$ being the infinite set of all strings consisting of 0's and 1's. Order these strings lexicographically: order by length (all strings of length 2 come before any strings of length 3), then alphabetically (011 before 110). Write an enumerator TM to output (write to tape) the sequence of strings formed by this ordering. Separate outputted strings by the # symbol. Your machine should output:

$$\#0\#1\#00\#01\#10\#11\#000\#001\ldots\#111\#0000\#0001\ldots$$

6. Consider a Turing Machine model in which the tape is *one-way infinite*. That is, the tape has a finite end on the left and is infinite to the right. In this model, we assume the input string is placed in the leftmost cells of the tape, and the tape head can never fall off the left end. If the head is at the leftmost cell and the transition function calls for it to move left, the tape head instead stays in that leftmost cell. Show the one-way infinite TM model is equivalent to the ordinary two-way infinite TM model used in our class.

Week 3

7. Show that if $\mathcal{L}$ is the language accepted by a k-tape, l-dimensional, nondeterministic Turing Machine with m heads per tape, then $\mathcal{L}$ is accepted by some ordinary deterministic Turing Machine with one single-dimensional tape and one head.

8. Suppose the tape alphabets of all Turing Machines are selected from some infinite set of symbols $a_1, a_2, \ldots$. Show how each Turing Machine may be encoded as a binary string.

*9. (hand in) Let $L_1, L_2, \ldots, L_k$ be a sequence of languages over alphabet Σ such that:

   (a) For all $i \neq j$, $L_i \cap L_j = \emptyset$; i.e., no string is in two of the languages.

   (b) $L_1 \cup L_2 \cup \cdots \cup L_k = \Sigma^*$; i.e., every string is in one of the languages.

   (c) Each of the languages L_i, for $i = 1, 2, \ldots, k$, is semi-decidable.

   Prove that each of the languages is therefore decidable.

Week 4

10. Given a string $w \in \{0,1\}^*$, let $\overline{w}$ be the one's complement of w, i.e., $\overline{w}$ is the result of flipping each 0 to 1 and each 1 to 0 in w. Let

$$S = \{\langle M \rangle \mid M \text{ is a TM that accepts } \overline{w} \text{ whenever } M \text{ accepts } w\}.$$

Prove that S is undecidable.

11. Prove each of the following problems either decidable or undecidable.

    (a) Given a TM M, an input string w, and a number $k \geq 0$, does M use at most k tape squares on input w?

    (b) Given a TM M and an input string w, does there exist a $k \geq 0$ such that M uses at most k tape squares on input w? (That is, does M use a finite amount of tape on input w?)

*12. (hand in) The problem of obsolete code is well known in software engineering. When code is maintained and updated, some lines of code may become obsolete in that they will never be executed. It would be nice to be able to identify and remove these lines of code. You will show that such a task is impossible on Turing Machines.

A *useless state* in a Turing Machine is one that is never entered on any input string. Consider the problem of identifying whether a Turing machine has any useless states. Formulate this problem as a language and show that it is undecidable.