# CS 154 - Introduction to Automata and Complexity Theory

Spring Quarter, 2000

Assignment #6 - Due date: Wednesday, 5/24/00

**ALERT:** This is your last chance to use the late credit for homeworks. If you have been wise and have saved up your credit, now is the time to use it.

**READING ASSIGNMENT:** We are almost done with Chapter 9 in the reader. Although Sections 9.4 and 9.5 were not covered in class, we strongly recommend that you skim through this portion. We will be covering Chapter 10 in the next week or two. (In the Hopcroft-Ullman textbook, these correspond to Chapters 8 and 13.)

## Problem Set

**Remarks on Problem Set:** In this part of the course, all "proofs" are to be presented at a higher level of abstraction than before. Take a look at the solution for Problem 8.3 on page 213 of the text-book to see the kind of proof we expect to see in your homeworks and exams. Pay particular attention to the following kinds of statements in that solution: "First, it shifts its input one position right, placing a left end marker ..." It should be clear that such constructions can be implemented on Turing machines and there is no need to specify the transitions of the machine in detail.

**Problem 1.** [20 points] Prove that the following language is recursive.

$$L = \{ < M, w > \mid \text{TM } M \text{ on input } w \text{ never moves its head leftwards}\}$$

You may assume that the input Turing machine is always deterministic. (*Hint:* Consider a machine that never moves its head leftwards. When can you be certain that the machine will not halt while executing on a particular input $w$?)

**Problem 2.** [20 points] Consider the language $L_{10}$ defined below:

$$L_{10} = \{ < M > \mid \text{there are at least 10 strings in the language of the Turing machine } M\}$$

**(a).** Show that $L_{10}$ is *recursively enumerable.*
**(b).** Is $L_{10}$ is *recursive*? Justify your answer.
**(c).** Prove that the complement of $L_{10}$ is *not* recursively enumerable.

**Problem 3.** [20 points] Consider the following language:

$$L = \{ <M_1, M_2, M_3> \mid L(M_1) = L(M_2).L(M_3) \}$$

The strings in $L$ encode three Turing machines with languages $L_1$, $L_2$ and $L_3$, such that $L_1 = L_2.L_3$.

You are told that following language $L_\emptyset$ is not recursively enumerable:

$$L_\emptyset = \{<M> \mid L(M) = \emptyset\}.$$

Give a *reduction* from $L_\emptyset$ to $L$ and prove that $L$ is not recursively enumerable.
(*Remark:* Example 8.2 on page 185 of the text-book proves that $L_\emptyset$ is not recursively enumerable. This is a rather instructive example in general, and we suggest that you look at it carefully.)

**Problem 4.** [20 points] The Big Computer Corporation has decided to bolster its sagging market share by manufacturing a high-tech version of the Turing machine called BWTM that is equipped with *bells-and-whistles.* This is basically the same as your ordinary Turing machine, except that each state in the machine is labeled as either a bell-state or a whistle-state. Whenever the BWTM enters a new state, it rings the bell or blows the whistle depending upon the label of the new state.

The *whistle-blowing problem* is defined as follows: given a BWTM $M$ and an input string $w$, does the execution of $M$ on input $w$ ever cause the machine to blow the whistle?

Using a reduction from the universal language $L_U$, prove that the whistle-blowing problem is undecidable.

**Problem 5.** [20 points] Consider the following two languages over the alphabet $\Sigma = \{0, 1\}$.

$$L_H = \{<M, w> \mid \text{TM } M \text{ halts on input } w\}$$

$$L_{NC} = \{<\widehat{M}> \mid L(\widehat{M}) \text{ is } \textit{non-context-free}\}$$

The language $L_H$ corresponds to the following decision problem: given a Turing machine $M$ and an input string $w$, does $M$ halt on input $w$? Similarly, the language $L_{NC}$ corresponds to the following decision problem: given a Turing machine $\widehat{M}$, is $L(\widehat{M})$ a non-context-free language?

Your goal is to show that $L_{NC}$ is undecidable, assuming that $L_H$ is undecidable. We propose the following reduction from $L_H$ to $L_{NC}$.

> Given a Turing machine $M$ and input $w$, construct a Turing machine $\widehat{M}$ which behaves as follows on being given input $\widehat{w}$.
>
> 1. $\widehat{M}$ simulates the behavior of $M$ on input $w$;
> 2. if $M$ halts on $w$, then $\widehat{M}$ checks to see if its input $\widehat{w}$ is of the form $xx$ for some string $x$, halting in a final state if $\widehat{w}$ is indeed $xx$ and halting in a non-final state otherwise.

**(a).** In the case where $M$ does not halt on $w$, what is $L(\widehat{M})$?
**(b).** In the case where $M$ does halt on $w$, what is $L(\widehat{M})$?
**(c).** Prove the correctness of the reduction described above and show that $L_{NC}$ is undecidable.