**6.045J/18.400J: Automata, Computability and Complexity** Nancy Lynch

# Homework 5.5 (FAKE)

*Due: Never* *Vinod Vaikuntanathan*

This "fake" homework is intended as a study guide covering the material in lectures 12 (Stack and counter machines), and 13, (Mapping reducibility and Rice's Theorem). It also includes two problems to review part of the material in lecture 14: on uses of the Recursion Theorem.

**Readings:** Sipser, Sections 5.3 and 6.1; Hopcroft et al., Section 8.5

**Problem 1**: (Counter Machine Examples)
Informally but clearly describe counter machines that recognize the following languages. Try to use as few counters as you can. (You may assume that a special endmarker appears at the end of the input string.)

1. $\{a^i b^j | i \leq j\}$.
2. $\{a^i b^j c^k | i \leq j \leq k\}$.
3. $\{a^i b^j c^k | k = ij\}$.

**Problem 2**: (Counter Machine Simulation of Stack Machine)
Suppose we want to design a counter machine to simulate a $k$-stack machine, using the same strategy discussed in lecture for simulating 2 stacks with 3 counters.

1. How many counters does this require? Explain why.
2. Simulating a single step of the $k$-stack machine requires that the CM (counter machine) emulate the reading of the top symbol of each stack, without removing that symbol from the stack. Describe clearly how the CM can read the top symbol of a stack.
3. Before the CM can determine the stack machine transition to be emulated, it must know the top symbol of every stack. Describe how the CM can remember these symbols in a way that allows it to determine the next stack machine transition.
4. After the CM determines the stack machine transition to be simulated, it must modify the simulated stacks as specified by that transition. Describe clearly how the CM can do this. How can it keep track of its progress?
5. How does the CM simulate the state change of the simulated stack machine?

**Problem 3**: (Mapping Reducibility)
Define *Total* to be the language $\{\langle M \rangle | \ M$ is a Turing machine and $L(M) = \{0,1\}^*\}$, that is, the set of representations of (basic) Turing machines that accept all strings of 0s and 1s. Use mapping reducibility to prove the following statements. Cite explicitly any theorems about mapping reducibility that you use.

1. *Total* is not Turing-recognizable.
2. The complement of *Total* is not Turing-recognizable.

**Problem 4**: (Rice's Theorem Applications)
Which of the following are decidable and which are undecidable? Give proofs. Use Rice's Theorem wherever possible, in showing undecidability.

1. $\{\langle M\rangle|\ M$ is a Turing machine and $M$ has more than 30 states$\}$.
2. $\{\langle M\rangle|\ M$ is a Turing machine and $L(M)$ is recognized by some Turing machine $M'$ with more than 30 states$\}$.
3. $\{\langle M\rangle|\ M$ is a Turing machine that accepts every word of the form $0^n1^n, n \geq 0$(and possibly other words)$\}$.
4. $\{\langle M, N\rangle|M$ and $N$ are Turing machines and $L(M) \subseteq L(N) \cup A\}$, where $A = 0^*$, the set of all finite strings consisting of just zeros.

**Problem 5**: (Generalization of Rice' Theorem)
Consider the following generalization of Rice's Theorem:
If $P2$ is a non-trivial property of pairs of recognizable languages, then

$$A_{P2} = \{\langle M, N\rangle|\ M \text{ and } N \text{ are Turing Machines and } P2(L(M), L(N)) = TRUE\}$$

is undecidable.

(Note: $P2$ is defined to be *non-trivial* if there exist Turing Machines $M_1, M_2, N_1$ and $N_2$ such that $P2(L(M_1), L(N_1)) = FALSE$ and $P2(L(M_2), L(N_2)) = TRUE$).

1. Prove this theorem.
2. Apply this theorem to show that it is undecidable whether, for given Turing machines $M$ and $N$, the language recognized by $M$ is a proper subset of the language recognized by $N$. Indicate carefully why all required hypotheses are satisfied.

**Problem 6**: (Using the Recursion Theorem to prove undecidability)
Let

$$A01 = \{\langle M\rangle|\ M \text{ is a Turing machine and } M \text{ accepts the string } 01\}$$

be the language of descriptions of machines accepting the string 01. We already know how to prove that $A01$ is undecidable using mapping reducibility or using Rice's Theorem. Now give yet another proof that $A01$ is undecidable, by using the Recursion Theorem.

**Problem 7**: (Using the Recursion Theorem to prove non-enumerability) Let's define a Turing machine $M$ to be ***pretty good*** if the length of the representation of every Turing machine $M'$ that recognizes the same language as $M$ is at least the square root of the length of the representation of $M$. (That is, for every $M'$ such that $L(M') = L(M)$, $|\langle M'\rangle| \geq \sqrt{|\langle M\rangle|}$. Prove that there is no enumerator that outputs the set of representations of pretty-good Turing machines.

**Problem 8**: Prove that every language that is Turing-recognizable can be mapping-reduced to the canonical Turing machine acceptance problem, $A_{TM}$.