SAMPLE MIDTERM QUESTIONS

JOEL FRIEDMAN

The following are sample midterm problems beyond the problems in the handout, "Computability and Self-Referencing in CPSC 421" (see Section 7) and the homework problems.

Here is a brief summary of the material so far. The theme is: what can and cannot (especially cannot) by computers in various models, with various resources.

1. Generally speaking, we work in settings that have a countable number of computer programs, but an uncountable number of languages. Hence there are (a whole lot) of languages that cannot be recognized by any computer program.
2. We can name some programs that cannot be recognized by any Turing machine (or similar model), such as complements of the Acceptance problem or complements of the Halting problem. In general:
   - (a) We know that the Acceptance problem and Halting problem are both examples of languages that can be recognized but not decided; and
   - (b) if L is a language that is recognizable but not decidable, then its complement cannot be recognized.
   - (c) A number of other problems, such as identifying a line of "dead code," cannot be recognized, and no algorithm can eliminate all such lines for every program.
3. The fact that the Acceptance problem—in many models of computation—is not decidable follows from general axioms, the most tedious to verify of which is (Axiom 2) the existence of a universal program in the setting in which we are working.
4. The class, P, of polynomial time decidable languages, tries to describe languages that can be decided by a reasonable efficient algorithm; however, there are a number of tricky aspects to the definition of P and it is not clear than a $10^{1000}n^2$ time algorithm is really practically usable.
5. There are a number of variants of Turing machines that are interesting, such as "oracle Turing machines" and "non-deterministic Turing machines."
6. You will be rich and famous (in academia, relatively speaking) if you can resolve the question of whether NP is the same class as P.

Research supported in part by an NSERC grant.

Sample Midterm Problems

(1) If $f\colon \mathbb{Z} \to \mathbb{Z}$ is a function, we define a *Turing machine with oracle* f to be a Turing machine with a special "oracle" worktape and two special states $q_{\text{ask oracle}}$ and $q_{\text{oracle reply}}$; the Turing machine runs as usual except for the following exception: whenever we enter the state $q_{\text{ask oracle}}$, if the special oracle worktape is a string over $\{1,\ldots,9,0\}$ representing the integer n in base 10, then at the next step we transition to the state $q_{\text{oracle reply}}$ and (magically) the value $f(n)$ appers on the oracle tape.

Fix a function f as above.

(a) Explain why Axioms 1, 3, 4, and 5 hold for Turing machines with oracle f.

(b) Does Axiom 2 hold for Turing machines with oracle f, i.e., does there exist a universal machine for Turing machines with oracle f? Explain. [Hint: Every Turing machine has the same access to the oracle.]

(2) Define a *Geiger Turing machine* to be a Turing machine with a special "Geiger" worktape and two special states $q_{\text{ask Geiger}}$ and $q_{\text{Geiger reply}}$; the Turing machine runs as usual except for the following exception: whenever we enter the state $q_{\text{ask oracle}}$, then at the next step we transition to $q_{\text{Geiger reply}}$, and a "random" integer between 1 and 10 is (magically) written on the worktape. This "random integer" depends on the particular Turing machine, the input to the Turing machine, and number of steps taken by the Turing machine; you do not know how this "random integer" is determined, and it may be an uncomputable function of the machine description, the input, and the number of steps taken.

Which of Axioms 1–5 hold (really meaning "can be seen to hold for the standard reasons") for Geiger Turing machines? [Remember that every Turing machine has its own random number generator.]

(3) Explain why, generally speaking, if a language, L, is recognizable but not acceptable, then the complement of L is not recognizable. Explain what this has to do with Axiom 5.

(4) Consider the language HALT-PROG, of description of Turing machines, $\langle M\rangle$ that eventually halt when given an empty input. Argue that HALT-PROG is recognizable but not decidable. Argue that its complement is not even recognizable.

(5) Consider the language, OUTPUTS, of descriptions, $\langle p, i\rangle$ of a 421Simple program, p, and an input, i, such that on input i the program p reaches a line of code which writes to the OUTPUT array (via a line of the form "LET OUTPUT[..."). Is OUTPUTS recognizable? Decidable?

(6) Consider the language, NO-OUTPUT, of descriptions, $\langle p, i\rangle$ of a 421Simple program, p, and an input, i, such that on input i the program p never reaches a line of code which writes to the OUTPUT array (via a line of the form "LET OUTPUT[..."). Is NO-OUTPUTS recognizable? Decidable?

(7) An *algebraic number* is a complex number, z, that satisfies an equation

$$a_n z^n + a_{n-1} z^{n-1} + \cdots + a_1 z + a_0 = 0$$

for some integers $a_n, \dots, a_0$. Show that the algebraic numbers are a countable set. [Recall that for any $(a_0, a_1, \dots, a_n)$, there are at most n complex numbers that satisfy the above displayed equation.]

(8) In class we have shown that if S is a set, then there is no function $f\colon S \to \mathrm{Power}(S)$ that is surjective, i.e., each element of Power(S) is in the image of f. The proof is to assume such an f exists, and to consider

$$T = \{s \in S \; s \notin f(s)\}$$

and to obtain a contradition.

- (a) Explain what is the contradiction.
- (b) Give an example where S has three elements and f is the function of your choice.
- (c) Explain why this proof by contradiciton is sometimes called *diagonalizatoin*; you might use your example in the previous part.

(9) Consider the phrase, "In Smalltown, there is one barber and she cuts the hair of anyone who does not cut his/her own hair." Explain the inherent contradiction that arises from this statement.

(10) Consider the phrase, "n is the smallest positive integer not described by a sentence of fewer than 100 words." Explain the paradox in this phrase, and explain one way to resolve this paradox, in a way that sentences that give an explicit calculation of an integer still refer to this integer.

(11) Russell's paradox arises from the phrase, "Let S be the set of all sets that don't contain themselves." Explain the paradox that arises, and indicate in a sentence or two how set theorists resolve this paradox."

(12) Give an implementation-level description of a universal Turing machine.

(13) Give a formal description of a Turing machine to recognize the language of words over $\{0, 1\}$ whose first and last letter are the same.

(14) Give a formal description of a 421Simple program to recognize the language of words over $\{0, 1\}$ whose first and last letter are the same.

(15) Give an implementation-level description of a Turing machine that recognizes a language recognized with a k-tape Turing machine using a 1-tape Turing machine.

(16) Give an implementation-level description of a Turing machine to decide the language PRIMES of strings over the alphabet $\{0, 1, \dots, 9\}$ that represent prime numbers written in base 10.

(17) Explain how a one-tape Turing machine can simulate an $f(n)$-time two-tape Turing machine in time order $f^2(n)$. Explain why we cannot improve the $f^2(n)$ to any $f^\gamma(n)$ with $\gamma < 2$. [Hint: In class we have stated that a one-tape Turing machine requires at least cn^2 time to recognize PALINDROMES (over any finite, non-empty alphabet), but can be performed in $O(n)$ time on a two-tape Turing machine.]

(18) Problems from Sipser, Chapter 3, regarding Turing machines: 3.5, 3.7, 3.8, 3.11, 3.12, 3.15(a,b,e), 3.16(a,d), 3.22. 3.5, 3.7, and 3.22 regard the definition of a Turing machine; 3.8 ask for implementation-level descriptions of certain languages; 3.11 and 3.12 regard variations on the definition of Turing machines; 3.15 and 3.16 regard combining two Turing machines.

(19) Consider the set of all Turing machines, M, such that (1) M always moves to the right on each step, (2) M has only three states: $q_0, q_{\mathrm{accept}}, q_{\mathrm{reject}}$ (and all of these states are distinct), and (3) the input alphabet of M is

$\{0,1\}$. As usual, define the result of M on input i to be (1) `yes` if M halts on input i in the state q_{accept}, (2) `no` if M halts on input i in the state q_{reject}, and (3) `NoHalt` if M on input i doesn't halt.

(a) Describe which languages such (very limited) Turing machines can recognize.
(b) Which of Axioms 2–5 are satisfied by this set of Turing machines?

(20) Give an implementation level description of a Turing machine whose input alphabet is $\{a,b,\dots,z,\#\}$ such that on input consisting of some number of words over $\{a,b,\dots,z\}$, separated by $\#$'s, the Turing machine accepts this input if the words are sorted in increasing lexicographical order. [Lexicographical order is "dictionary order," i.e., you compare two words by looking at their first letter, and in a tie you look at the second letter; if one word ends and another does not, then the ending word is first in order.] For example,

$$ab\#aba\#bzza\#bzzb$$

would be accepted.

(21) Give an implementation-level description of a Turing machine that accepts the language whose words are of the form $w_1\#w_2$, where w_1,w_2 are integers described in base 10 such that $3w_1+5=w_2$ (where the strings w_1,w_2 are interpreted as integers).

(22) Let L be the language consisting of 1 if the Twin Prime Conjecture (a currently unresolved conjecture) is true, and 0 otherwise. Is L a recognizable language? Explain.

(23) Give an implementation-level description and a fromal description (i.e., write out the values of the δ-function) of a Turing machine that recognizes the language of words over $\{1,2\}$ that have an even number of 1's.

(24) Give an implementation-level description and a fromal description (i.e., write out the values of the δ-function) of a Turing machine that recognizes the language of words over $\{1,2\}$ that have an even number of 1's and an odd number of 2's.

(25) Give an implementation-level description and a fromal description (i.e., write out the values of the δ-function) of a Turing machine that recognizes the language of words over $\{1,2\}$ that begin and end with the same letter.

(26) Which of the following sets are countable and why?
(a) the integers;
(b) the rational numbers;
(c) the set of subsets of the integers;
(d) the set of strings over a finite alphabet;
(e) the set of languages over a finite alphabet (recall that a language is a collection of strings);
(f) the set of sequences over $\{1,2,3,4\}$, i.e,, the set $\{1,2,3,4\}^{\mathbb{N}}$.

(27) Outline how a universal Turing machine can be designed. Outline how a universal C program can be designed. What are the relative advantages and disadvantages of the two different settings?

Department of Computer Science, University of British Columbia, Vancouver, BC V6T 1Z4, CANADA, and Department of Mathematics, University of British Columbia, Vancouver, BC V6T 1Z2, CANADA.

E-mail address: `jf@cs.ubc.ca or jf@math.ubc.ca`

URL: `http://www.math.ubc.ca/~jf`