

MIDTERM  
CIS 132 - Winter 07  
Warmuth

NAME: \_\_\_\_\_  
Student ID: \_\_\_\_\_

**This exam is closed book and closed notes.**

Show partial solutions to get partial credit.

If your questions are not written legibly, you won't get full credit.

Clarity and succinctness will be rewarded!

question 1: \_\_\_\_\_(out of 20)

question 2: \_\_\_\_\_(out of 12)

question 3: \_\_\_\_\_(out of 12)

question 4: \_\_\_\_\_(out of 11)

question 5: \_\_\_\_\_(out of 15)

question 6: \_\_\_\_\_(out of 15)

question 7: \_\_\_\_\_(out of 15)

Total: \_\_\_\_\_(out of 100)

Extra Credit: \_\_\_\_\_(out of 15)

1. a. *Give a function that is not Turing computable.*

$$f(x) = \begin{cases} 1 & \text{if there exists a } T \text{ such that } x = e(T) \text{ and } x \in L(T) \\ 0 & \text{otherwise} \end{cases}$$

Or, more generally,  $f$  can be the characteristic function of any non-TA language.  
Or, for any non-RE language  $L$ , we can define  $f$  as

$$f(x) = \begin{cases} 1 & \text{if } x \text{ in } L \\ \text{undefined} & \text{otherwise} \end{cases}$$

- b. *Are non-deterministic Turing machines more powerful than deterministic Turing machines?*

No. A deterministic TM can simulate an NTM. See the proof of Theorem 9.2 for details.

- c. *Is there a language over a finite alphabet that is uncountably infinite? Give such a language or reason that there can't be one (in one sentence).*

No. Any language is a subset of  $\Sigma^*$  which is countable.

- d. *Give a closure property that holds for Turing decidable languages that does not hold for Turing acceptable languages.*

Complement or set difference.

- e. *What are the inputs to a universal Turing machine and what does it do?*

The input is  $e(T)e(w)$ ; it simulates  $T$  on  $w$ .

f. *State the Church-Turing thesis.*

Turing machines are as powerful as any computational device.

2. Show that the class of Turing acceptable languages is closed under concatenation.

$$L_1 \cdot L_2 = \{w_1w_2 \mid w_1 \in L_1 \text{ and } w_2 \in L_2\}$$

*Hint: From Tms  $T_1, T_2$  accepting  $L_1$  and  $L_2$ , respectively, construct a new machine  $T$  accepting  $L_1 \cdot L_2$ . You might want to use nondeterminism.*

Since  $L_1, L_2$  are T.a. there are TMs  $T_1, T_2$  which accept them. We will use these TMs to construct a TM  $T$  which accepts  $L_1 \cdot L_2$ .

$T$  will act as follows. Given an input  $y$ ,  $T$  does a pass over the input while copying the letters to tape 2 and non-deterministically switches. After switching it copies the rest of the input to tape 3. At this point tape 2 has a prefix  $x_1$  of the input and tape 3 a suffix  $x_2$  such that  $x_1, x_2 = y$ . Then  $T$  simulates  $T_1$  on tape 2; if  $T_1$  accepts then it simulates  $T_2$  on tape 3. If  $T_2$  also accepts,  $T$  accepts, otherwise it rejects.

If there exists  $x_1, x_2$  such that  $y = x_1x_2$  and  $x_1 \in L_1$  and  $x_2 \in L_2$ , then both  $T_1$  and  $T_2$  will accept these strings and hence  $T$  accepts  $y$ .

Conversely if  $T$  does not accept  $y$  then there can no acceptable  $x_1, x_2$  and so  $y \notin L_1 \dot{L}_2$ .

3. In a famous letter to Leonard Euler dated June 7th 1742, Christian Goldbach first conjectures that every even number  $\geq 4$  is the sum of two primes

$$4 = 2 + 2 \quad 8 = 5 + 3 \quad 26 = 23 + 3$$

The conjecture has been checked numerically for numbers  $\leq 6 \cdot 10^{16}$ . However no general proof is known.

Show that if **Halt** was solvable then Goldbach's conjecture could be resolved.

*Hint: Construct a Turing machine and feed it to **Halt**. It suffices to give a high-level outline of the Turing machine construction in pseudo code.*

Our strategy will be the following:

- (a) Show that there is a machine  $D$  which computes

$$d(n) = \begin{cases} 1 & \text{if } n \text{ is the sum of two primes} \\ 0 & \text{otherwise} \end{cases}$$

- (b) Create a machine  $G$  which takes no input, and will successively compute  $d(n)$  for  $n = 4, 6, 8, \dots$  and halt immediately if a value of  $n$  is found such that  $d(n) = 0$ .
- (c) Use the machine which solves the halting problem to determine if  $G$  halts when run on the empty string.

The function  $d(n)$  is clearly computable: we can just try all prime numbers less than  $n$  and check if any two sum to  $n$ .

Clearly if we find that  $G$  halts then there is some even number  $n$  which is not the sum of two primes, and thus Goldbach was wrong; otherwise Goldbach's conjecture holds for all integers.

4. *Prove that the set  $N \times N$  is countable infinite by giving a bijection to the natural numbers.  
Hint: One way is to give a list of the set.*
- See proof in book (page 392).

5. Prove both directions of the following theorem:

*Theorem: A language  $L$  is Turing-acceptable  $\iff L$  is “recursively enumerable”.*

*Definitions:*

*A language  $L$  is recursively enumerable if there exists a two-tape Turing machine  $M$  that achieves the following when started with both tapes empty:*

- *It writes (outputs) a sequence of words on the first tape that are separated by single blanks.*
- *The tape head of the first tape never moves left.*
- *The possibly infinite sequence of words output on the first tape contains all words of  $L$  at least once and contains no word that is not in  $L$ .*
- *If  $L$  is finite then  $M$  may halt when it has printed all the elements of  $L$  on the first tape or it may continue to make moves without printing any other strings on tape one.*

*Hint: From a Tm that accepts  $L$  construct one that enumerates  $L$  (and vice versa).*

See proof in book on page 369.

Prove two of the remaining three problems. The third problem counts as extra credit.

6. Show that  $2^{1^*}$  (i.e. the set of subsets of  $\{\Lambda, 1, 11, 111, 1111, \dots\}$ ) is uncountably infinite using a diagonalization argument.

To get a contradiction, assume  $2^{1^*}$  is countable. Then the elements of  $2^{1^*}$  can be listed in some order  $s_1, s_2, \dots$

Now consider the following set

$$s = \{1^i \mid 1^i \notin s_i\}$$

$s$  is a set of unary strings, and so  $s \in 2^{1^*}$ . Thus there exists a  $j$  such that  $s = s_j$ . However for any  $j$ , if  $1^j \in s_j$  then  $1^j \notin s$  and if  $1^j \notin s_j$  then  $1^j \in s$ , so  $s \neq s_j$ .

This is a contradiction and we can conclude that  $2^{1^*}$  is uncountable.

7. Assume you have shown already that **Halt** is unsolvable. Now show that **Accepts** is unsolvable by reducing **Halt** to **Accepts**.  
Show all three parts of the proof.

### **Halt**

Input: A Turing machine  $T$  and input  $w$ .

Question: Does  $T$  halt on  $w$ .

### **Accepts**

Input: A Turing machine  $T$  and input  $w$ .

Question: Does  $T$  accept  $w$ .

*Hint: Use the identity function as your mapping  $F$  and see what part of the proof does not work. Then modify  $F$  slightly.*

Again show all three parts of the proof. **Halt**  $\leq$  **Accept**

1.  $F(T, w) = (T', w)$  where  $T'$  is just  $T$  where any transitions to  $h_r$  have been made into transitions to  $h_a$ .
2. This is a simple search and replace operation, and thus it is computable.
3. If  $T$  halts on  $w$  then it finishes in either  $h_a$  or  $h_r$  and so  $T'$  must finish in  $h_a$ , so  $T'$  accepts  $w$ .

If  $T'$  accepts  $w$  then it finishes in  $h_a$ . Either it got to  $h_a$  via a transition to  $h_a$  in  $T$  or via a transition to  $h_r$  in  $T$ . Since both of these are halting states, it follows that  $T$  halts on  $w$ .

8. *Is any countably infinite union of Turing acceptable languages Turing acceptable. Prove this or give a counter example.*

No. Consider  $NSA = \{x_1, x_2, \dots\}$ . For any  $i$ ,  $\{x_i\}$  is a finite and hence Turing acceptable language. And yet we have proven elsewhere that

$$NSA = \bigcup_{i=1}^{\infty} \{x_i\}$$

is not Turing acceptable.