CS 341 Automata Theory
STUDENT NAME - EID
Homework 11
Due: Tuesday, April 3

This assignment reviews Turing machine construction and covers Sections 17.3 - 17.6 and Chapter 18 and 19.

1) Define a Turing Machine M that computes the function $f:\ \{a,b\}^* \to N$, where:

$$f(x) = \text{the unary encoding of } max(\#a(x), \#b(x)).$$

For example, on input `aaaabb`, M should output `1111`. M may use more than one tape. It is not necessary to write the exact transition function for M. Describe it in clear English.

Solution. □

2) Construct a Turing machine M that converts binary numbers to their unary representations. So, specifically, on input $< w >$, where w is the binary encoding of a natural number n, M will output 1^n. (Hint: use more than one tape.)

Solution. □

3) In Example 17.9, we showed a Turing machine that decides the language WcW. If we remove the middle marker c, we get the language WW. Construct a Turing machine M that decides WW. You may exploit nondeterminism and/or multiple tapes. It is not necessary to write the exact transition function for M. Describe it in clear English.

Solution. □

4) In Example 4.9, we described the Boolean satisfiability problem and we sketched a nondeterministic program that solves it using the function choose. Now define the language SAT $= \{< w >: w \text{ is a wff in Boolean logic and } w \text{ is satisfiable}\}$. Describe in clear English the operation of a nondeterministic (and possibly n-tape) Turing machine that decides SAT.

Solution. □

5) What is the minimum number of tapes required to implement a universal Turing machine?

Solution. □

6) Encode the following Turing Machine as an input to the universal Turing machine:

$$M = (K, \Sigma, \Gamma, \delta, q_0, \{h\}), \text{ where: } K = \{q_0, q_1, h\},\ \Sigma = \{a, b\},\ \Gamma = \{a, b, c, \square\}, \text{ and } \delta =$$

q	σ	$\delta(q,\sigma)$
q_0	a	$(q_1, b, \rightarrow)$
q_0	b	$(q_1, a, \rightarrow)$
q_0	$\square$	$(h, \square, \rightarrow)$
q_0	c	$(q_0, c, \rightarrow)$
q_1	a	$(q_0, c, \rightarrow)$
q_1	b	$(q_0, b, \leftarrow)$
q_1	$\square$	$(q_0, c, \rightarrow)$
q_1	c	$(q_1, c, \rightarrow)$

Solution. □

7) Churchs Thesis makes the claim that all reasonable formal models of computation are equivalent. And we showed in, Section 17.4, a construction that proved that a simple accumulator/register machine can be implemented as a Turing machine. By extending that construction, we can show that any computer can be implemented as a Turing machine. So the existence of a decision procedure (stated in any notation that makes the algorithm clear) to answer a question means that the question is decidable by a Turing machine. Now suppose that we take an arbitrary question for which a decision procedure exists. If the question can be reformulated as a language, then the language will be in D iff there exists a decision procedure to answer the question. For each of the following problems, your answers should be a precise description of an algorithm. It need not be the description of a Turing Machine:

   (a) * Let $L = \{< M >: M$ is a DFSM that doesnt accept any string containing an odd number of 1s$\}$. Show that L is in D.

   Solution. □

   (b) Consider the problem of testing whether a DFSM and a regular expression are equivalent. Express this problem as a language and show that it is in D.

   Solution. □

8) Consider the language $L = \{w = xy : x, y \in \{a, b\}^*$ and y is identical to x except that each character is duplicated$\}$. For example `ababaabbaabb` $\in L$.

   (a) * Show that L is not context-free.

   Solution. □

   (b) Show a Post system that generates L.

   Solution. □

9) Consider the language $L = \{< M >: M$ accepts at least two strings$\}$.

   (a) Describe in clear English a Turing machine M that semidecides L.

   Solution. □

   (b) Suppose we changed the definition of L just a bit. We now consider:

   $$L' = \{< M >: M \text{ accepts exactly 2 strings}\}.$$

   Can you tweak the Turing machine you described in part a to semidecide L'?

   Solution. □