CS 3100 – Models of Computation – Fall 2011

PRACTICE FINAL EXAM – CLOSED BOOK – 100 points

Your Name: **Your UNID:**

1. **(10 pts)** State the Schröder-Bernstein theorem, and apply it to prove that the set of all *Prime* numbers has the same cardinality as the set of all *Even* numbers. Present the 1-1 total and into mappings going both ways.

   Primes to Even: for the ith prime, generate the product of the primes from 1 through i (will be even because 2 is an even prime).

   Even to Primes: List the $i/2$nd prime for everh i being an Even.

   That is

   ```
   Even to Primes

   0 -> 2
   2 -> 3
   4 -> 5
   6 -> 7
   8 -> 11
   ...

   Primes to Even

   2 -> 2
   3 -> 2*3
   5 -> 2*3*5
   ...
   ```

2. **(10 pts)** Prove that the following language is recursively enumerable. Your proof must be a precise English argument presented as bulletted steps that shows how to enumerate the contents of this language.

   $$L_{PCP} = \{\langle p \rangle \mid p \text{ is a } PCP \text{ instance with a solution}\}$$

   Can enumerate successful solutions. Generate each and every arrangement with repetitions systematically (this can be done; detail a few steps; example: if we have tiles 1,2,3,4, one can enumerate all strings of length 1 over 1,2,3,4, strings of length 2, etc.).

   Toss out those that are not solutions. List those that are solutions.

3. **(5 pts)** Prove by contradiction that $\overline{A_{TM}}$ is not recursively enumerable. You can assume standard results about A_{TM} after stating them.

   If enumerable, then it becomes decidable because its complement is enumerable. This will make A_{TM} deciable which is a contradiction.

4. **(10 pts)** Based on cardinality arguments, show that non-RE languages (RE = recursively enumerable) must exist. You must show to some degree of detail any intermediate cardinality arguments you wish to make about various sets.

   Since there are $\aleph_1$ languages (a proof one can arrive at by the diagonal construction) but only $\aleph_0$ RE languages, there must be those that are non-RE.

5. **(15 pts)** Someone claims that the following is a mapping reduction from A_{TM} to $Regular_{TM}$, where the latter is the language of all TMs (TM codes, actually) that have a regular language. They explain that given M and w, the mapping reduction produces $M^{'}$ described below.

```
M'(w) {
  if w is of the form 0^n 1^n then goto accept_M' ;
  Run M on w ;
  If M accepts w, goto accept_M' ;
  If M rejects w, goto reject_M' ; }
```

   Their proof goes as follows:

$$Decider_{Regular_{TM}}(M^{'}) = \begin{cases} accepts & \Rightarrow\ L(M^{'})\ is\ regular \\ \quad \Rightarrow\ Language\ is\ \Sigma^* & \Rightarrow\ M\ accepts\ w \\ rejects & \Rightarrow\ L(M^{'})\ is\ not\ regular \\ \quad \Rightarrow\ Language\ is\ 0^n1^n & \Rightarrow\ M\ does\ not\ accept\ w \end{cases}$$

   Explain their proof.

   The proof is explained above! The above clearly shows how the languages are connected. Now, if we can decide regularity, we can obtain a decider for A_{TM} based on the above mapping reduction.

6. **(10 pts)** Draw a BDD for the Boolean function $oddParity(x)$ over a nibble x (nibble = 4 bits, calling the bits x_0 through x_3). This function is true exactly when there are an odd number of 1s in x. Does the variable order matter for this function? Why or why not?

   It does not matter; XOR based parity is commutative and associative (order-independent). Thus no matter how you draw the BDD, the "rest of the variables" down in the BDD need to decode in the same manner.

7. **(10 pts)** Draw a BDD for the Boolean function $lessThan(x, y)$ over inputs $x = x_2x_1x_0$ and $y = y_2y_1y_0$. This function is true exactly when the binary magnitude of x is strictly less than that of y. Does the variable order matter for this function? Why or why not?

   It does matter, as with the equality comparison. The "rest of the bits" need to decode differently. I.e. if $<$ is violated early in the variable order, we can skip the rest of the comparison. To know this early violation property, we need to interleave the bits. That BDD will be smaller.

8. **(5 pts)** What is the Rice's Theorem, and how does it help prove the undecidability of $Regular_{TM}$? Explain in some detail.

   Every non-trivial partitioning of TM codes based on the languages of the TMs is undecidable. The regularity-check is such an attempt at a non-trivial partition; tries to separate Sigma-star into those that encode TMs with regular languages and those that do not.

9. **(3 pts)** How does computability theory and complexity theory relate in terms of the use of mapping reductions?

   They both use it to relate the hardness of problems. The mapping reduction idea is common to them.

10. **(3 pts)** Explain in a few sentences what NP-languages are, and what NP-complete is. Why are Computer Scientists interested in studying NP-complete languages?

    Languages with an NP decider; NPC means within NP and NPhard where NPhard means every NP problem has a mapping-reduction to it. NPC are likely exponential. More on Wikipedia and other places.