CS 3100 – Models of Computation – Fall 2011
Assignment 11 Solutions

1. (**20%**) Write a proof, from first principles (by building the D function) for $Halt_{TM}$ being undecidable. $Halt_{TM} = \{\langle M, w\rangle \mid M\ is\ a\ TM\ that\ halts\ on\ w\}$. Follow the structure of arguments given on Page 1 of `http://www.eng.utah.edu/~cs3100/lectures/l27/notes27.pdf`.

```
* Suppose there is a decider "H" for Halt_TM.

* H(M,w) accepts if  M will halt on w, and rejects otherwise.

* Derive a contradiction as follows. Define a machine D

* D(M) {
  if H(M,M) ACCEPTS,  -- i.e. M halts on M
   then LOOP        -- D rejects M
   else ACCEPT        -- D accepts M when M does not halt on M
  }

* Now does D halt on D? I.e. does D(D) halt or loop?

* D(D) = H(D,D)

  - if H(D,D) accepts, i.e. D halts on D, then D loops on D

  - if H(D,D) rejects, i.e. D loops on D, then D halts on D

* The contradiction is complete. Hence H cannot exist.
```

2. (**20%**) Write a detailed mapping reduction proof from $Halt_{TM}$ to A_{TM}, showing details similar to those in Figure (a), Page 2, `http://www.eng.utah.edu/~cs3100/lectures/l27/notes27.pdf`.

```
* Given an M and w, make a copy of M (calling it M'), and then change every accept state to a non-a
  state, and vice versa.

* Now, given A_TM, build a decider for Halt_TM as follows:

  - Feed M,w to one copy of A_TM, calling this A_TM's outputs a1 and r1

  - Feed M',w to another copy of A_TM, calling this A_TM's output a2 and r2

  - Now what truth-tables are possible?
```

```
a1 r1 a2 r2  Halt_TM_a   Halt_TM_r
-----------------------------------
0  1  0  1   0           1
1  0  0  1   1           0
0  1  1  0   1           0
-----------------------------------
```

3. Explain what the sets A and B of Figure (b) are for these proofs. Write out the "if and only if" style proof "punchline" (e.g. $x \in A$ if and only if $f(x) \in B$; hence a solver for B would solve A) to make sure you understand what is going on. You can get ideas on how to write from Page 2.

   - (**5%**) $A_{TM} \leq_m PCP$.

     ```
     We build the mapping reduction in such a way that

     M,w in A_TM    if and only if    Puzzle_{M,w}  has a solution.
     ```

   - (**5%**) $PCP \leq_m CFG_{amb}$. Here, CFG_{amb} is the language of CFG encodings that are ambiguous.

     ```
     Here, we build a mapping reduction in such a way that

     P in Solvable_PCPs    if and only if    CFG_{P} is unambiguous.
     ```

4. (**40%**) Encode the following Lewis Carroll puzzle using the DDCal tool and find a proof. You may have to strengthen the given conditions. Thoroughly explain how BDDs helped you solve this puzzle (one-page description).

   I'll help you by giving a template, below.

   ```
   # A puzzle by Lewis Carroll :
   #
   #  From the premises
   #
   #(a)   None of the unnoticed things, met with at sea, are mermaids.
   #
   #(b)   Things entered in the log, as met with at sea, are sure to be worth remembering.
   #
   #(c)   I have never met with anything worth remembering, when on a voyage.
   #
   #(d)   Things met with at sea, that are noticed, are sure to be recorded in the log.
   #
   # Prove that I have never met with a mermaid at sea

   # N = it is noticed, M = it is a mermaid, L = entered in log,
   # R = worth remembering, I = I have met with it at sea, T = met at sea

   #  First specify the desired variable ordering. DDcal can later reorder
   ```

```
var = T*N*M*L*R*I


#(a)   None of the unnoticed things, met with at sea, are mermaids.
A1 = T*N' => M'

#(b)   Things entered in the log, as met with at sea, are sure to be worth remembering.
A2 = T*L => R

#(c)   I have never met with anything worth remembering, when on a voyage.
A3 = I => R'

#(d)   Things met with at sea, that are noticed, are sure to be recorded in the log.
A4 = T*N => L

# Prove that I have never met with a mermaid at sea
proofGoal = M => I'

# Negate proof-goal and add it in
contra1 = A1 * A2 * A3 * A4 * proofGoal'

#Oops, need frame axiom: not met at sea => I have not met with it at sea
frame = T' => I'

contra = contra1 * frame

[contra1 contra]
```

5. (**10%**) Write a one-page writeup on NP-completeness, Read about NP-complete problems in `http://en.wikipedia.org/wiki/NP-complete`. Mention some of the common NP-complete problems. Elaborate on some of the common misunderstandings about NP-complete problems that are listed there. You may survey other sources also - but **please cite every source you survey!**

   `Summarize well.`