Problem Set 11

Submission instructions: Submit each problem on a **separate sheet of paper**, put your name on each sheet, and write your discussion section time and day (e.g. Tuesday 10am) in the upper righthand corner. These details may sound picky, but they make the huge pile of homeworks much easier to grade quickly and more importantly, since we return them in the discussion sections, easier for you to get them back.

Also, write on each exercise the name/netid of your group members.

Due: Monday, May 4, 2009 at 12:30 in Elaine Wilson office (SC 3229). If the door is locked, slide your solutions under the door.

1. Language classification.
   [**Category**: Understanding, **Points**: 10]

   Suppose that we have a set of Turing machine encodings defined by each of the following properties. That is, we have a set

   $$L = \left\{ \langle M \rangle \;\middle|\; M \text{ is a } \mathsf{TM} \text{ and } M \text{ has property } P \right\},$$

   and we are considering different ways to fill in P. Assume that the Turing machines M have only a single tape.

   (A) P is "M accepts an input string of length 3."

   (B) P is "on blank input, M halts in at most 300 transitions, leaving the entire tape blank."

   (C) P is "M's code has exactly 3 states to which there are no transitions ."

   (D) P is "M accepts no string of length 3."

   For each of these languages, determine whether it is Turing decidable, Turing recognizable, or not Turing recognizable. Briefly justify your answers.

2. Reduction I.
   [**Category**: Construction, **Points**: 10]

   Define the language L to be

   $$L = \left\{ M \;\middle|\; M \text{ is a } \mathsf{TM} \text{ and } L(M) \text{ is decidable but not context free} \right\}.$$

   Show that L is undecidable by reducing A_{TM} to L. (Do the reduction directly. Do not use Rice's Theorem.)

3. Reduction II
[**Category**: Construction, **Points**: 10]

Define the language L to be

$$L = \left\{ M \;\middle|\; M \text{ is a } \mathsf{TM} \text{ and } \forall n\ \exists x \in L(M) \text{ where } |x| = n \right\}.$$

Show that L is undecidable by reducing A_{TM} to L. (Do the reduction directly. Do not use Rice's Theorem.)

4. Enumerate this.
[**Category**: Construction, **Points**: 10]

Construct an enumerator for the following set:

$$L = \left\{ \langle \mathtt{T} \rangle \;\middle|\; \mathtt{T} \text{ is a Turing Machine and } |\mathsf{L}(\mathtt{T})| \geq 3 \right\}.$$

5. DFAs are from Mars, TMs are from Venus.
[**Category**: Understanding / Proof., **Points**: 10]

Consider the language

$$L = \left\{ \langle \mathtt{M}, w \rangle \;\middle|\; \mathtt{M} \text{ is a } \mathsf{DFA} \text{ and it accepts } w \right\}.$$

We will prove that L is undecidable by reducing A_{TM} to it:

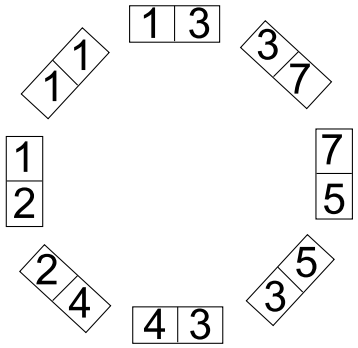
> *Proof:* For every $\langle \mathtt{M}, w \rangle$, let $\mathtt{T}_{\mathtt{M}}$ be a TM that simulates DFA $\mathtt{M}$. So $\mathtt{M}$ will accept w iff $\mathtt{T}_{\mathtt{M}}(w)$ halts and accepts w, which is exactly equivalent to $\langle \mathtt{T}_{\mathtt{M}}, w \rangle \in \mathrm{A}_{\mathsf{TM}}$; that is,
>
> $$\langle \mathtt{M}, w \rangle \in L \quad \Longleftrightarrow \quad \langle \mathtt{T}_{\mathtt{M}}, w \rangle \in \mathrm{A}_{\mathsf{TM}}.$$
>
> This completes the reduction. But since A_{TM} is undecidable, L should be undecidable too. ■

Why this result is strange? Is it because we did something wrong in the above proof? Is this proof correct? Explain briefly.

6. Using reductions when building algorithms.
[**Category**: Construction., **Points**: 10]

Reductions are not only a technique to prove hardness of problems, but more "importantly" it is used to solve problems by transforming them into other problems that we know how to solve. This is an extremely useful technique, and the following is an example of such a reduction in the algorithmic context.

A set of domino pieces has a solution iff one can put them around a circle (face up), such that every two adjacent numbers from different pieces match. The figure on the right shows a set of pieces which has a solution.



Assume we have an algorithm (i.e., think of it as a black box) **isDomino** which given a set of dominos, can tell use whether they have a solution or they do not have a solution (it returns "yes" or "no" answer). Using **isDomino**, describe an algorithm which given a set of dominos outputs "no" if there is no solution and if there is a solution, prints out one possible solution (that is, prints out an ordered list of input dominos, such that one can put them in the same order around a circle properly).

For example if the input to your algorithm is $(1,2)\#(3,5)\#(4,3)\#(1,3)\#(1,1)\#(2,4)\#(3,7)\ \#(7,5)$, it may output $(1,3)\#(3,7)\#(7,5)\#(3,5)\#(4,3)\#(2,4)\#(1,2)\#(1,1)$ (which is the solution depicted in the above figure).

(We emphasize that your solution must use **isDomino** to construct the solution. These is a direct algorithm to solve this problem, but we are more interested in the reduction here, than in an efficient solution.)