CS601:	**Homework 4**	Due in main office before 4:30 pm, March, 3, 2009

Note: Because the midterm is March 9, I will hand out solutions to Hw4 at the discussion section on March 6, so that will be the latest I will accept a late Hw4.

Problems:

1. [20 pts.] Describe an efficient algorithm for the following decision problem. Analyze the time complexity of your algorithm and give a clear explanation why it works.

$$\text{EQUAL-DFAs} = \{\langle D_1, D_2\rangle \mid D_1, D_2 \text{ DFAs}, \mathcal{L}(D_1) = \mathcal{L}(D_2)\}$$

   [Hint: there is a perfectly acceptable natural algorithm that runs in $O(n^2)$ time on a RAM. A small amount of extra credit if you figure out – and that does not mean look up – a significantly faster algorithm.]

2. [80 pts.] For each of the following five sets, state whether it is recursive, r.e., but not recursive, or co-r.e., but not recursive, or none of the above. If any of them is r.e. complete, or co-r.e. complete please prove that as well.

   Prove your answers. I suggest that to prove that a set S is not r.e. you show that $\overline{K} \leq S$ and to prove that it is not co-r.e. you show that $K \leq S$.

   Since this is the first problem set where you must create reductions and show that they are correct, please put some care into showing that all of your reductions are polynomial-time computable and that the chain of "⇔"s really do hold in both directions. That is, for each reduction, please start at the left and prove the implications to the right and then start at the right and prove the implications to the left.

   (a) $S_a = \{n \mid W_n \neq \emptyset\}$

   (b) $S_b = \{n \mid M_n(\mathbf{N}) \subseteq \{17\}\}$

   (c) $S_c = \{n \mid M_n$ is total, i.e., converges on all inputs$\}$

   (d) $S_d = \{(P, D) \mid \mathcal{L}(D) \subseteq \mathcal{L}(P)$ for PDA P; DFA $D\}$ [You may assume the fact that there are polynomial-time computable functions taking an input PDA to an equivalent CFG, and vice-versa.]

   (e) $S_e = \{\langle n, m, s\rangle \mid$ for some input w of length n, $M_m(w)$ uses more than s work tape cells$\}$. (Recall that we say that a TM uses each work-tape cell that it ever visits during its computation.)

   Here $\langle n, m\rangle$ is the same thing as $P(n, m)$, and $\langle n, m, p\rangle$ is the same thing as $\langle n, \langle m, p\rangle\rangle$.