Problem Set 8 — Due Friday, March 14, 1997

A student asked me to identify the more "basic" problems to work on. For this problem set, all of problems 2–4 are pretty easy NP-completeness reductions (that is, easy for someone who already knows how to do this type of thing!). I think they're all of about the same difficulty. The last problem isn't so easy (and you have to figure out what NP-Complete problem to start from), and so I made it extra credit. You'll certain to be asked to prove some language is NP-Complete on the final.

The first problem gets at a general question of "decision vs. search." It turns out that you can usually solve an optimization problem given an oracle for the corresponding decision problem. That's what you're trying to do here: imagine that you have a black box which ***decides*** *3SAT, and use it to construct an algorithm which* ***solves*** *3SAT.*

Problem 1. Page 274, Problem 7.29.

Problem 2. Page 272, Problem 7.19.

Problem 3. A graph $G = (V, E)$ is said to be *k-colorable* if there is a way to paint its vertices using colors in $\{1, 2, \ldots, k\}$ such that no adjacent vertices are painted the same color. When k is a number, by k*COLOR* we denote the language of (encodings of) k-colorable graphs. The language *3COLOR* is NP-Complete. (You can assume this.) Use this to prove that the language *4COLOR* is NP-Complete, too. That is, you have to show that *3COLOR* is in NP, and *3COLOR*$\leq_{\mathrm{P}}$*4COLOR*.

Problem 4. Page 273, Problem 7.24.

Problem EC. Page 274, Problem 7.26.