Math class is tough!

— Teen Talk Barbie (1992)

That's why I like it!

— What she should have said next

The Manhattan-based Barbie Liberation Organization claims to have performed corrective surgery on 300 Teen Talk Barbies and Talking Duke G.I. Joes—switching their sound chips, repackaging the toys, and returning them to store shelves. Consumers reported their amazement at hearing Barbie bellow, 'Eat lead, Cobra!' or 'Vengeance is mine!,' while Joe chirped, 'Will we ever have enough clothes?' and 'Let's plan our dream wedding!'

— Mark Dery, "Hacking Barbie's Voice Box: Vengance is Mine!", *New Media* (May 1994)

21 NP-Hard Problems (April 24)

21.1 'Efficient' Problems

A long time ago[1], theoretical computer scientists like Steve Cook and Dick Karp decided that a minimum requirement of any efficient algorithm is that it runs in polynomial time: $O(n^c)$ for some constant c. People recognized early on that not all problems can be solved this quickly, but we had a hard time figuring out exactly which ones could and which ones couldn't. So Cook, Karp, and others, defined the class of *NP-hard* problems, which most people believe *cannot* be solved in polynomial time, even though nobody can prove a super-polynomial lower bound.

Circuit satisfiability is a good example of a problem that we don't know how to solve in polynomial time. In this problem, the input is a *boolean circuit*: a collection of and, or, and not gates connected by wires. We will assume that there are no loops in the circuit (so no delay lines or flip-flops). The input to the *circuit* is a set of m boolean (true/false) values $x_1, \ldots, x_m$. The output is a single boolean value. Given specific input values, we can calculate the output in polynomial (actually, *linear*) time using depth-first-search and evaluating the output of each gate in constant time.

The circuit satisfiability problem asks, given a circuit, whether there is an input that makes the circuit output TRUE, or conversely, whether the circuit *always* outputs FALSE. Nobody knows how to solve this problem faster than just trying all 2^m possible inputs to the circuit, but this requires exponential time. On the other hand, nobody has ever proven that this is the best we can do; maybe there's a clever algorithm that nobody has discovered yet!