Number Six: What do you want?
Number Two: Information!
Number Six: Whose side are you on?
Number Two: That would be telling. We want information!
Number Six: You won't get it!
Number Two: By hook or by crook, we will!
— Opening sequence of 'The Prisoner' (1967–68)

19 Lower Bounds (April 12)

19.1 What *Are* Lower Bounds?

So far in this class we've been developing algorithms and data structures for solving certain problems and analyzing their time and space complexity.

Let $T_A(X)$ denote the running of algorithm A given input X. Then the worst-case running time of A for inputs of size n is defined as follows:

$$T_A(n) = \max_{|X|=n} \big(T_A(X)\big).$$

The worst-case complexity of a *problem* Π is the worst-case running time of the *fastest* algorithm for solving it:

$$T_\Pi(n) = \min_{A \text{ solves } \Pi} \big(T_A(n)\big) = \min_{A \text{ solves } \Pi} \left(\max_{|X|=n} \big(T_A(X)\big)\right).$$

Now suppose we've shown that the worst-case running time of an algorithm A is $O(f(n))$. Then we immediately have an *upper bound* for the complexity of Π:

$$T_\Pi(n) \le T_A(n) = O(f(n)).$$

The faster our algorithm, the better our upper bound. In other words, when we give a running time for an algorithm, what we're really doing — and what most theoretical computer scientists devote their entire careers doing[1] — is bragging about how *easy* some problem is.

Starting with this lecture, we've turned the tables. Instead of bragging about how easy problems are, now we're arguing that certain problems are *hard* by proving *lower bounds* on their complexity. This is a little harder, because it's no longer enough to examine a single algorithm. To show that $T_\Pi(n) = \Omega(f(n))$, we have to prove that *every* algorithm that solves Π has a worst-case running time $\Omega(f(n))$, or equivalently, that *no* algorithm runs in $o(f(n))$ time.

[1] This sometimes leads to long sequences of results that sound like an obscure version of "Name that Tune":

Lennes: "I can triangulate that polygon in $O(n^2)$ time."
Shamos: "I can triangulate that polygon in $O(n \log n)$ time."
Tarjan: "I can triangulate that polygon in $O(n \log\log n)$ time."
Seidel: "I can triangulate that polygon in $O(n \log^* n)$ time."
[Audience gasps.]
Chazelle: "I can triangulate that polygon in $O(n)$ time."
[Audience gasps and applauds.]
"Triangulate that polygon!"