Computational Complexity Handout

Adam Elga

February 18, 2003

Most of this is adapted from Papadimitriou, C. H. *Computational Complexity*, Chapter 2, which is on course reserve at Firestone. Some of it is reproduced directly, the rest is paraphrase.

1 Big-O notation

Let $f, g : N \to N$. Def of "f is of the order of g":

$f(n) = O(g(n))$ iff there are positive integers c and n_0 such that for all $n \geq n_0$, $f(n) \leq cg(n)$.

Informal gloss: eventually f is bounded by some constant factor of g.

Write $f(n) = \Theta(g(n))$ ("f and g are of the same order") iff $f(n) = O(g(n))$ and $g(n) = O(f(n))$.

Important facts (examples): $10000n$ grows slower than n^2 grows slower than n^3 grows slower than $n^{10000000}$ grows slower than 2^n.

2 Languages: sets of yes/no questions

Complexity theorists say "language" to mean "set of strings". Machine M decides L iff for any string s, $M(s) = 1$ if s is in L, and $M(s) = 0$ if s is not in L.

Intuitively: a language is a family of yes/no questions.

M decides on s in time t if M halts in t or fewer steps on s.

M decides L in time f(n) if for any string s of length n, M decides s in time less than or equal to f(n).

M decides L in space f(n) if for any string s of length n, M decides s by writing to a zone of squares no bigger than f(n).[1]

Example: Primality testing.

[1] Here we assume that M has a special input tape that it can only read consecutive elements from, and a special output tape that it can only write consecutive elements to. We don't count the space used by what's written on the input and output tapes.

3 Important complexity classes

Complexity class Time(f(n)): set of languages decidable in time f(n) by some multitape Turing machine.

Complexity class P: set of languages decidable in time n^k (for some k) by some Turing machine.

Nondeterministic machine: same as regular TM, except that the current state of the machine doesn't uniquely determine what to do next. When n alternatives are given, picture the whole machine and tape splitting into n copies, each of which takes one alternative.

We say that a nondeterministic machine N accepts s if at least one of its descendents eventually outputs 1. Otherwise we say that it rejects s.

N decides L if for every s, s is in L iff N accepts s (i.e., one of its descendants outputs 1).

N decides L in time f(n) if for every s of length n, s is in L iff N accepts s (i.e., one of its descendants outputs 1), and if every branch of N on s has length less than or equal to f(n).

Complexity class NP: set of languages decidable in time n^k (for some k) by some nondeterministic Turing machine.

Example of a problem. Traveling salesman problem: given (integer-valued) distances between cities and some bound B, is there a nonrepeating route with total length less than or equal to B?

Is it in NP? Why?

Is it in P? Big question.

4 Reductions

"L1 is reducible to L2 iff there is a function R from strings to strings computable by a TM in space O(log n) such that for all x: x is in L1 iff R(x) is in L2."