Algorithms - Day 5

Instructor: Pat Devlin — prd41@math.rutgers.edu

Summer, 2016

Reduction and NP-completeness

One of the most useful (and surprising) techniques in complexity theory is the idea of problem reduction.

Definition 1 We say problem A *reduces* to problem B if and only if we can efficiently use an algorithm that solves B to construct an algorithm to solve A.

Example 2 The problem "find the max element" reduces to the sorting problem.

Example 3 The problem "can this partially filled in Sudoku puzzle be completed?" reduces to the graph coloring problem.

A reduces to B $\implies$ B is at least as hard as A

Question 4 If we can solve the graph coloring problem in polynomial time, then we can solve Sudoku puzzles in polynomial time. Why?

Definition 5 We say a problem, A, is NP-complete if and only if it is in NP, and every other problem in NP can be reduced to A.

Remark: A problem is NP-complete essentially means that it's the hardest that any problem in NP could possibly be.

Proposition 6 *If A is NP-complete, then $P = NP$ if and only if A is in P.*

Proof:

Proposition 7 *If A is NP-complete, and A reduces to a problem B in NP, then B is NP-complete.*

Proof:

The remarkable thing is that NP-complete problems exist. And in fact, they're *everywhere*!

All of the following problems are NP-complete[1]

- Subset sum problem [homework 4, problem 2] (Given a set of integers, is there a subset summing to 0?)
- Clique problem [homework 4, problem 5] (What's the largest t such that K_t is a subgraph of G? [i.e., what's the size of the largest 'clique' of G?])
- Independent set problem [homework 4, problem 5] (What's the size of the largest independent set of G?)
- Graph coloring problem [homework 4, problem 7]
- Is G Hamiltonian?
- Is H a subgraph of G?
- Vertex cover problem
- Dominating set problem
- Travelling salesman problem
- Knapsack problem
- Boolean satisfiability problem (SAT)
- Longest common subsequence problem
- Maximum bipartite subgraph
- Art gallery problem
- Generalized assignment problem
- Pancake sorting problem
- Set packing problem

[1] List of course taken from wikipedia article on 'list of NP-complete problems.

More NP-complete problems coming from puzzles [the problems listed below are generalizations of problems encountered in each of the following]

- Cubic
- Edge-matching puzzles
- Fillomino
- FreeCell
- Hashiwokakero
- Heyawake
- Instant insanity
- Kakuro
- KPlumber
- Kuromasu
- Light up
- Masyu
- Mastermind
- Minesweeper
- Nonograms [Pat likes these puzzles]
- Nurikabe
- Pearl puzzles
- SameGame
- Shanghai
- Slitherlink
- Sudoku
- Verbal arithmetic

Yet more NP-complete problems coming from classic games and video games

- Battleship
- Bejeweled
- Candy Crush Saga
- Combinatorial games (think like tic-tac-toe)
- Donkey Kong
- Legend of Zelda (entire series)
- Lemmings
- Mario
- Metroid

- Othello
- Phutball
- Pokémon
- Twixt

And some more NP-complete problems from various areas of combinatorics

- 1-planarity
- 3-dimensional matching
- Assembling an optimal Bitcoin block
- Bandwidth problem
- Bipartite dimension
- Capacitated minimum spanning tree
- Cycle rank
- Degree-constrained spanning tree
- Domatic number
- Exact cover
- Feedback vertex set (and feedback arc set)
- Flow shop scheduling problem
- Graph intersection number
- Graph partition
- Longest path problem
- Maximum induced path
- Metric dimension of a graph
- Minimum k-cut
- Pathwidth
- Route inspection problem
- Set splitting
- Vehicle routing problem
- (et cetera, et cetera, et cetera, et cetera, . . .)

> Since these are all NP-complete problems, if you solve *any* of these in polynomial time, you can use that solution to solve *every* NP problem in polynomial time!

Weird fact: In computer science, problems seem to either obviously in P or obviously NP-complete. The most striking possible exception to this is the graph isomorphism problem, which seems to be neither NP-complete nor in P.

Example 8 Pat's daily math email to his wife.

Example 9 Hilbert's tenth problem and the halting problem.

Example 10 Quantum computing.

Example 11 Automated proofs.

Example 12 Combinatorial games.