Hard Problems

Familiar Problems

Sorting: $n\log(n)$

Shortest Path: $|E|\log(|V|)$

All pairs shortest path: $|V|^3$

Topological Sort: $|E|+|V|$

Minimum Spanning Tree: $|E|\log(|V|)$

Dictionary Operations: $\log(n)$

...

Familiar Problems

All of these are $O(n^k)$

Question: Are all problems solvable in polynomial time?

Computability

An simpler question:

Are all problems solvable?

Equivalently:

Can a computer compute the value of f(x) for any f and x?

Computability

An simpler question:

Are all problems solvable?

Equivalently:

Can a computer compute the value of f(x) for any f and x?

Answer: No!

The Halting Problem

Given a computer program F, write a program H, where,

$$h(f,x) = \begin{cases} 1 & \text{if } f(x) \text{ halts} \\ 0 & \text{if } f(x) \text{ runs forever} \end{cases}$$

This is impossible!

The Halting Problem

Proof (by contradiction):

Suppose h(f,x) is computable.

First, enumerate every computer program:

$$f_1$$
$$f_2$$
$$f_3$$
$$\vdots$$

The Halting Problem

Proof, continued:

Now define a function,

$$g(x) = \begin{cases} 0 & \text{if } f_x(x) \text{ loops forever} \\ \text{loop forever} & \text{if } f_x(x) \text{ halts} \end{cases}$$

Since we listed all computable function,
for some k, $g = f_k$

The Halting Problem

Proof, continued:

What happens when we call g on its own number?

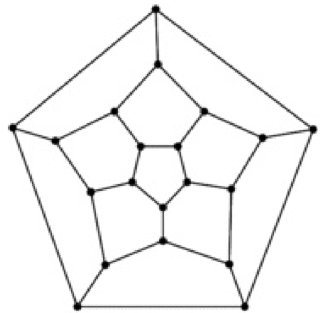
$$g(k) = \begin{cases} 0 & \text{if } g(k) \text{ loops forever} \\ \text{loop forever} & \text{if } g(k) \text{ halts} \end{cases}$$

A Contradiction!

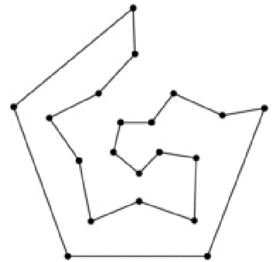
Two Problems

Hamiltonian Circuit

Given a graph G:



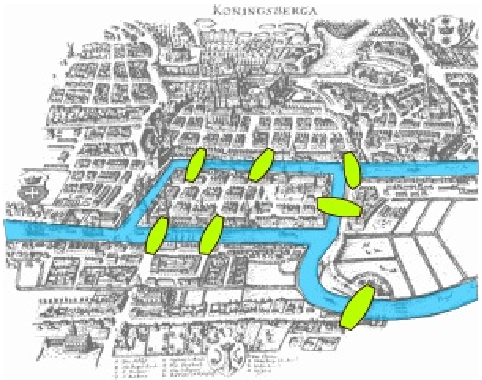
Find a path that that goes through each vertex exactly once, and returns:



Two Problems

Eulerian Circuit

Given a graph G:



Find a path that that goes through each edge exactly once, and returns:

Two Complexity Classes

P = The set of all problems solvable in polynomial time.

NP = The set of all problems verifiable in polynomial time.

Note that P is a subset of NP.

Two Complexity Classes

Eulerian Cycle in in NP.

Hamiltonian Cycle also is in NP.

Eulerian Cycle in in P.

Is Hamiltonian Cycle in P?

Two Complexity Classes

Eulerian Cycle in in NP.

Hamiltonian Cycle also is in NP.

Eulerian Cycle in in P.

Is Hamiltonian Cycle in P?

No one knows!

$P \stackrel{?}{=} NP$

Is there any problem in NP that's not in P?

Not currently known!

Huge practical consequences:
  circuit board layout,
  protein folding,
  flight schedules,
  etc.

(Also, a $1 million prize.)

P != NP

Suppose we want to show P!=NP.

Find any problem in NP that's definately not in P.

Similary to comparison sort taking at least nlog(n) comparisons.

Find a problem in NP that takes at least,

$$2^n,\ n!,\ \binom{n}{k},\ \ldots$$

P = NP

Suppose we want to show P=NP.

Find the hardest problem in NP, and show that it's in P.

Ok, so what's the hardest problem in NP?

Two More Complexity Classes

NP-Hard =
problems at least as hard as anything in NP.

NP-Complete =
problems in both NP-Hard and NP

We just need to show one problem in NP-Complete is also in P!

NP-Complete

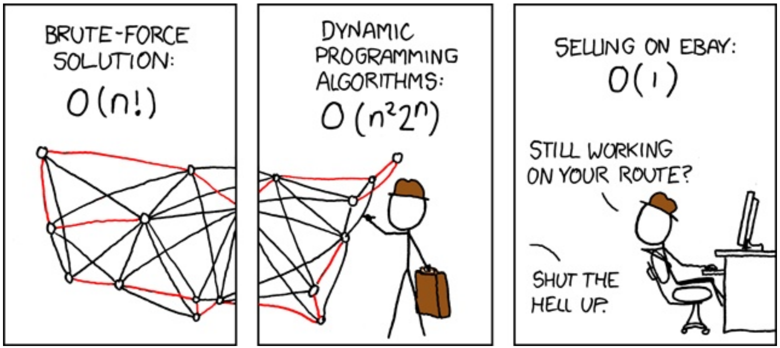
Hamiltonian Cycle in NP-Complete.

Thousands of others are too:

- Longest Path
- Boolean Satisfiability
- Traveling Salesman
- Set Covering
- Graph Coloring
  - ...

Traveling Salesman

Given set of n cities, and distances between the cities, find the shortest cycle visiting each city exactly once.



This is just Hamiltonian Cycle, but with edge weights.

Longest Path

Problem:
Given a graph, find the longest path between any two vertices.

Longest Path is NP-Complete.

But Shortest Path is in P!

15-Puzzle

Find the fewest number of moves needed to solve the k-Puzzle.

Pretty good solutions using Best First Search with a Manhattan Distance heuristic.

Minesweeper

Is a certain assignment of flags constistant with the adjacent numbers.

Easy in practice, but not easy in general.

SAT

Given a set of Boolean variable,s can we assign true/false values to them to make an arbitrary formula true?

For example, give the formula:

$$(\neg x_1 \vee x_2) \wedge (x_1 \vee x_3) \vee \neg(x_2 \vee \neg x_4) \wedge \neg x_3$$

Can we assign values to the x's to make this true?

3SAT

Let's make things easier:

Every formula takes the same form:

$$(\neg x_1 \vee x_2 \vee x_4) \wedge (x_1 \vee \neg x_3 \vee x_4) \wedge (x_2 \vee \neg x_4 \vee \neg x_5)$$

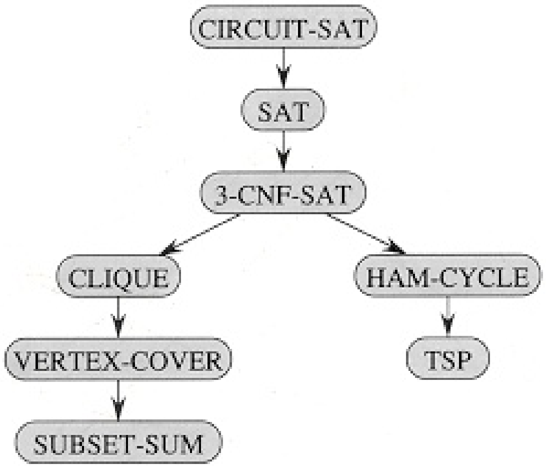
Formula:
and number of clauses seperated by OR
Clause:
three terms seperated by AND
Term:
either x_i or

3SAT is NP-Complete, 2SAT is P.

Proving NP-Completeness

Show a reduction from some known NP-Complete problem.

"If we can solve Hamiltonian Cycle we can solve 3SAT. "



Practical Concerns

Moral of the story:

Don't waste time trying to write a clever program to solve a NP-complete problem.

Look for a good approximation or heuristic.