Day 3: Intro to Comp. Complexity, P & NP

Zach Neveu

May 14, 2019

1 Review

- Huge difference between $O(n^k)$ and $O(k^n)$ (Polynomial vs. Exponential)
- Intractable Problems can only be solved (exactly) with an Exponential time algorithm
- Intractable $\neq$ Unsolvable!!

2 New Stuff: Intractable Problems

How to prove a problem is intractable?

1. If solution has size that is exponential, then any algorithm to find that solution can't be polynomial.
2. If problem can't be solved by any algorithm at all (undecidable) - Halting Problem
3. Certain niche problems with small output that are solvable but intractable

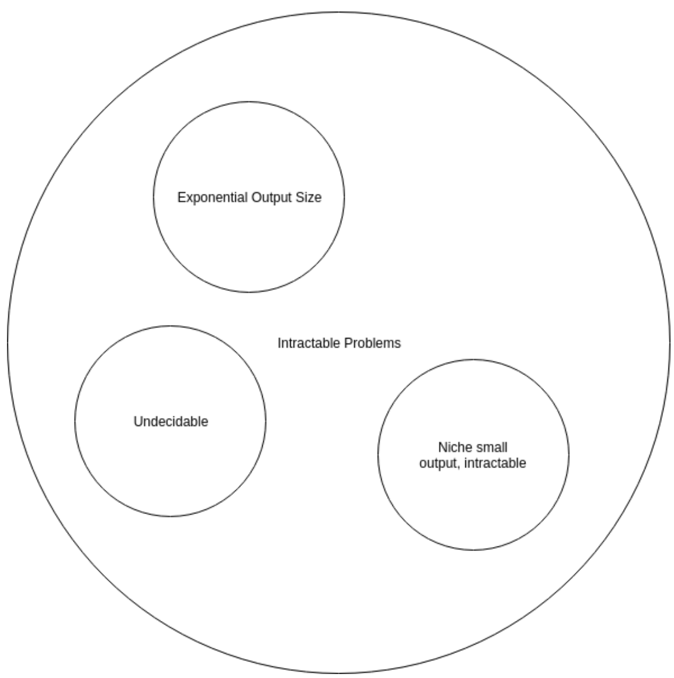

Figure 1: Intractable Problem Categories

3 NP-Completeness

Properties

- No one has ever found a polynomial-time algorithm to solve
- If someone found one algorithm to solve a single NP complete problem in polynomial-time, it would solve all other NP-Complete problems too.
- NP complete problems are either all tractable, or all intractable.
- All in same boat, we just don't know what boat.
- Tons of practical NP complete problems
- Seems unlikely that NP-Complete problems are tractable.
- It is widely believed that the NP-Complete problems are intractable.

What is NP-Complete?

If you can't find an efficient algorithm what do you say?

- "I can't find a solution" - get fired
- "A solution provably doesn't exist" - exceedingly unlikely
- "I can't solve it, but neither can anyone" - show that NP-Complete

Decision vs. Optimization Problems

- Computational Complexity initially developed for decision problems
- Must prove that it can be applied to optimization problems as well
- TSP-opt - Given weighted graph, find shortest Hamiltonian cycle - optimization problem
- TSP-dec - Given weighted graph, is there a Hamiltonian Cycle with weight $\leq k$
- Knapsack-opt - Given objects w/ values and sizes and size bound, find subset to maximize values within size bound.
- Knapsack-dec - Given objects w/ values and sizes and size bound, is there a subset of the objects that are within the size bound and have a value $\geq k$.
- If optimization is tractable, then decision is tractable - just plug in answer from optimization
- If decision is tractable, then optimization is also tractable - just run binary search over k values (adds log of constant)
- Decision and Optimization always have same complexity

4 Complexity Classes

P

- The set of all decision problems that can be solved by a polynomial-time algorithm.
- HC Problem: Given a graph, is it Hamiltonian (does it contain at least one Hamiltonian cycle)?

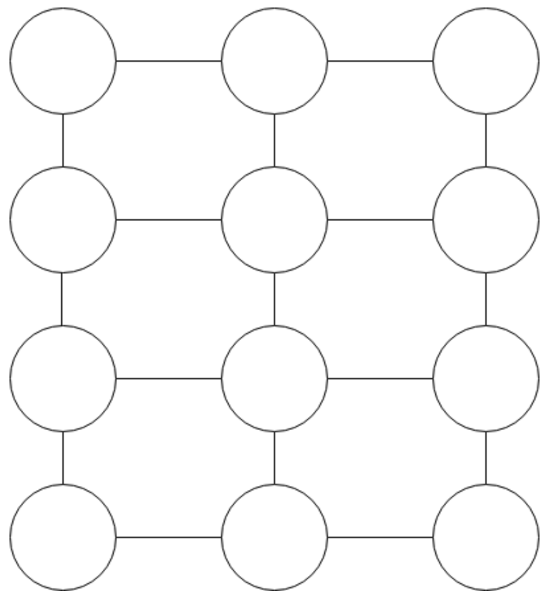

Figure 2: Does this graph have a Hamiltonian Cycle?

- A Verification Algorithm takes some information and checks it to make sure it satisfies the problem requirements.
- Example: given a cycle, verify that it is in fact Hamiltonian
- A **Yes Instance** of a problem will have some certificate that a verification algorithm can verify
- a **No Instance** will not contain a valid certificate to be verified
- Example: TSP-dec
  - certificate: sequence of nodes in the found HC
  - Verification algorithm: Check that each node appears once, edges are valid, and $\sum weights \leq k$
- Example: Matching - given graph and k, is there a matching of size k?
  - Certificate: list of edges in found matching
  - Verification algorithm: make sure no two edges have same end point, edges exist, and number of edges is k
- Shortest Path - given graph, source and dest nodes, and k is there a path from start to end cheaper or equal to k?

- certificate: ordered nodes in the path found
- verification: check that edges exist, $\sum weight \leq k$

NP

- Set of all decision problems such that yes instances have a certificate that can be verified in polynomial-time, and no instances do not have a certificate that can be verified in polynomial-time.
- Basically - a verification algorithm exists that works, and it runs in polynomial-time.
- Verification alg. for HC problem runs in polynomial time for yes instances, so HC $\in$ NP
- TSP: also in NP
- Matching: also in NP