Complexity

Agenda

- What is a complexity class?
- What are the important complexity classes?
- How do you prove an algorithm is in a certain class

Complexity class

- A complexity class is a set
- All problems within that set can be solved within certain resource bounds
- There are a lot of them (https://complexityzoo.uwaterloo.ca/Complexity_Zoo)
- However a few are more important
  - P
  - NP
  - PSPACE

P

- The class of problems solvable in $O(n^c)$ time for some real c.
- Matrix multiplication
- Dijkstra, Prim, Kruskal
- Almost all major algorithms we have talked about so far

NP

- Problems where checking if a possible answer is correct can be done in Polynomial time

- Note: If you can solve a problem correctly in poly time, you can check if it is correct in poly time

- Thus $P \subseteq NP$

Other definitions

- Deterministic - Programs that must always behave the same way on the same input.
- Nondeterministic - Program has the ability to simulate multiple “branches” at the same time
- NP - Nondeterministic Polynomial time

How to prove algs are in certain classes

- P
  - Give an algorithm that solves the problem
  - Prove that the algorithm runs in polynomial time
- NP
  - Idea of a “certificate”, some sort of proof of a solution
  - State what the certificate is
  - Show that verifying the certificate can be done in polynomial time

Examples - Which class?

- Hamiltonian path
- Composite number
- Cycle detection
- Longest path in a graph

P vs NP

- Most famous unsolved problem in CS

- $1,000,000 prize from Clay Institute
  - Fame
  - Fortune
  - Bragging rights

- “Does P=NP? If it is easy to check if a solution is correct is it also easy to solve the problem?”

Reductions

- Some problems can be solved using other problems
- Example: Assume that we know how to solve the s-t cut problem
  Note that we can use that to solve the global max cut problem by making n^2 calls
- If you can take a problem A, convert it to another problem B in polynomial time, and solve the original problem as the new problem, we say that $A \leq_p B$
- A polynomially reduces to B
  “A can be solved using B”

Remarks

- Note that poly(x) * poly(x) is a polynomial

- If we can take problem A, convert in poly time, and then solve in poly time as B, then A can be solved in poly time

NP-Completeness

- Is there a problem that can solve all problems in NP?
  I.e. for any arbitrary problem A, $A \leq_p B$

- Yes!

Cook-Levin Theorem

- All problems in NP can be polynomially reduced to SAT
- SAT: Boolean Satisfiability Problem
- All problems in NP reduce to SAT

NP-Complete

- We say that SAT is NP-Complete
- Meaning: All problems in NP can be solved with SAT with a polynomial time reduction
- Are there other NP-Complete problems?
- Yes! Lots!

NP-Complete

- In the same way that problems can be solved using SAT, SAT can be solved using many other problems

- If you can use a problem in NP to solve sat with a poly reduction, that problem is also NP-Complete

NP-Complete reductions

- Proofs follow the following form

- To prove something is NP-Complete

1. Prove that the problem is in NP

2. Prove that if you can solve this new problem, you can solve another NP-Complete problem

Examples

- Independent Set reduces to Vertex Cover
- Hamiltonian Cycle Pebbling Game

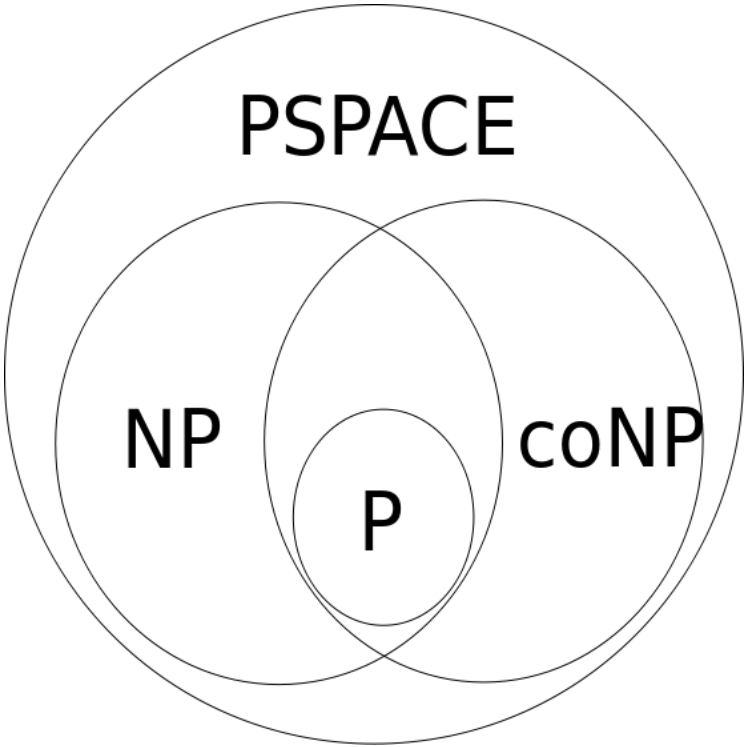
Co-NP

- Problems where verifying that a claim is false takes polynomial time
- Primality Testing

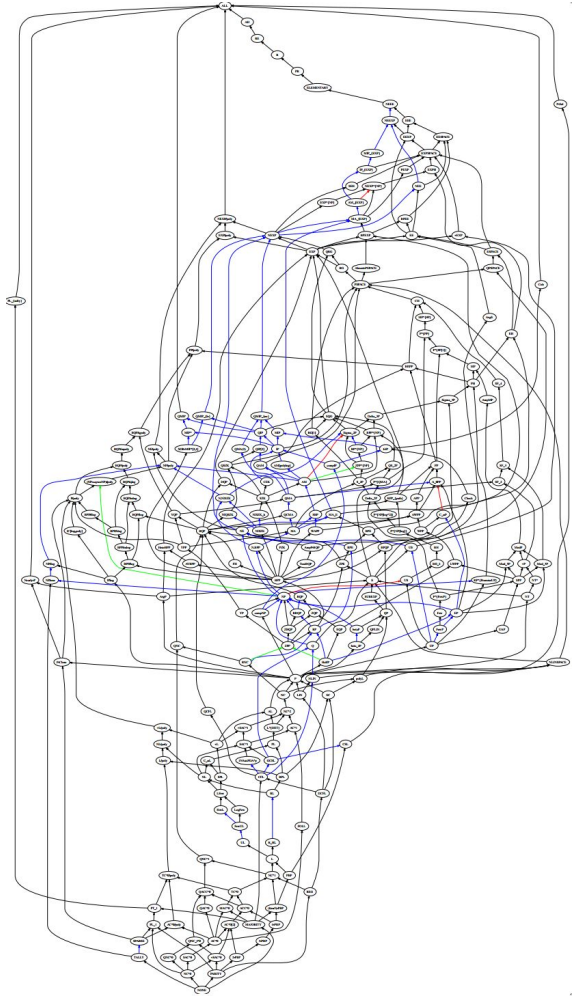
PSPACE

- Polynomial Space instead of polynomial time

The big picture



And a much scarier version



Pseudopolynomial Time

- How do you check if a number is prime (easy way)

Pseudopolynomial time

- Some problems are polynomial in the **value** of the input
- But we want it to be polynomial in the **representation** of the input
- The representation of a number n has value n but representation length $\log n$
- Problems solvable in polynomial time of the **value** but not in the **representation** are called pseudopolynomial time algorithms
- So don’t fall for that trap!

List of NP-Complete Problems

- Knapsack
- Hamiltonian path/cycle
- SAT
- 3SAT
- Independent set
- Clique
- Vertex Cover
- Max Cut

A bit more practice

Prove that Clique is NP-Complete