



# CSE 332: Data Structures & Parallelism

## Lecture 25: P, NP, NP-Complete (part 2)

Ruth Anderson

Autumn 2019

# Today's Agenda

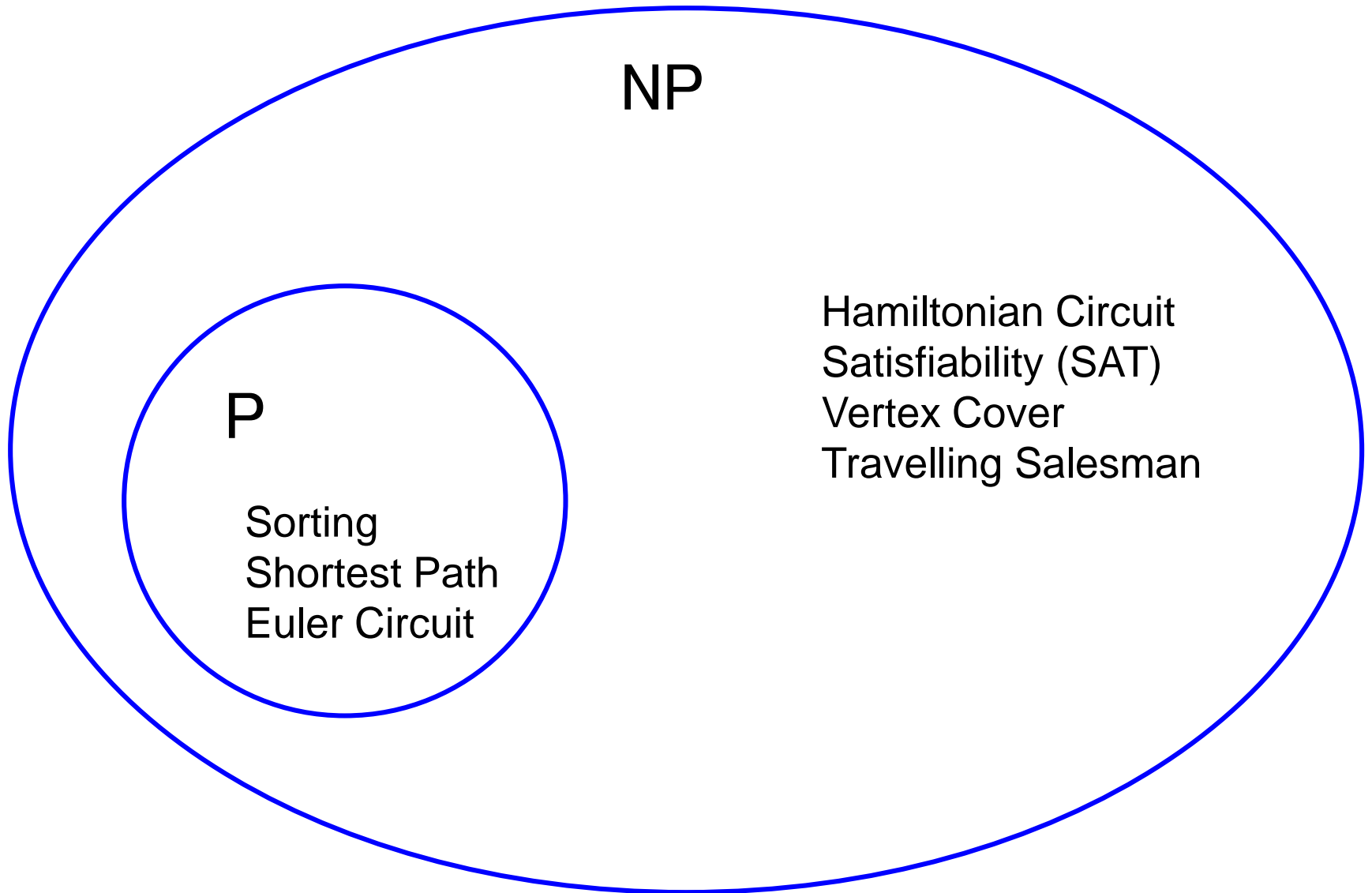
- A Few Problems:
  - Euler Circuits
  - Hamiltonian Circuits
- Intractability: P and NP
- NP-Complete
- What now?

# A Glimmer of Hope

- If given a candidate solution to a problem, we can **check if that solution is correct in polynomial-time**, then **maybe** a polynomial-time solution exists?
- Can we do this with Hamiltonian Circuit?
  - Given a candidate path, is it a Hamiltonian Circuit?

# The Complexity Class NP

- *Definition*: NP is the set of all problems for which a given *candidate solution* can be *tested* in polynomial time
- Examples of problems in NP:
  - *Hamiltonian circuit*: Given a candidate path, can test in linear time if it is a Hamiltonian circuit
  - *Vertex Cover*: Given a subset of vertices, do they cover all edges?
  - *All problems that are in P* (why?)



# Why do we call it “NP”?

- NP stands for *Nondeterministic Polynomial time*
  - Why “nondeterministic”? Corresponds to algorithms that can guess a solution (if it exists), the solution is then verified to be correct in polynomial time
  - Can also think of as allowing a special operation that allows the algorithm to magically guess the right choice at each branch point.
  - Nondeterministic algorithms don't exist – purely theoretical idea invented to understand how hard a problem could be

# Your Chance to Win a Turing Award!

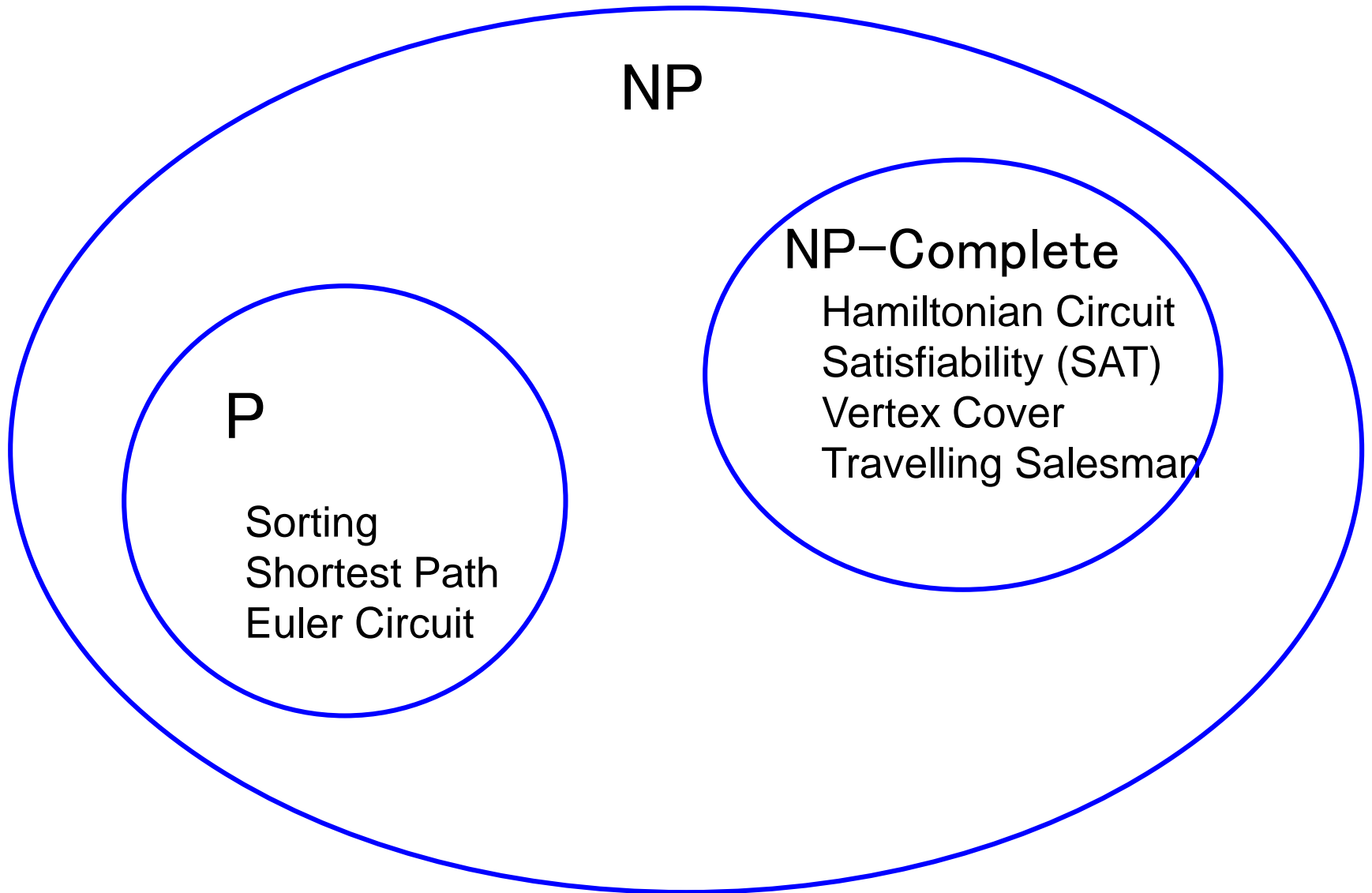
It is generally believed that  $P \neq NP$ ,  
*i.e.* there are problems in NP that are **not** in P

- But no one has been able to show even one such problem!
- This is the fundamental open problem in theoretical computer science
- Nearly everyone has given up trying to prove it. Instead, theoreticians prove theorems about what follows once we assume  $P \neq NP$  !

# NP-completeness

- Set of problems in NP that (we are pretty sure) **cannot** be solved in polynomial time.
- These are thought of as the **hardest** problems in the class NP.
- **Interesting fact:** If any one NP-complete problem could be solved in polynomial time, then **all** NP-complete problems could be solved in polynomial time.
- **Also:** If any NP-complete problem is in P, then all of NP is in P





# Saving Your Job

- Try as you might, every solution you come up with for the Hamiltonian Circuit problem runs in exponential time.....
- You have to report back to your boss.
- Your options:
  - Keep working
  - Come up with an alternative plan...

# In general, what to do with a Hard Problem

- Your problem seems really hard.
- If you can **transform a known NP-complete problem into the one you're trying to solve**, then you can stop working on your problem!

# Your Third Task

- Your boss buys your story that others couldn't solve the last problem.
- Again, your company has to send someone by car to a set of cities. There is a road between every pair of cities.
- The primary cost is distance traveled (which translates to fuel costs).
- Your boss wants you to figure out how to drive to each city exactly once, then return to the first city, while staying within a fixed mileage budget  $k$ .

# Travelling Salesman Problem (TSP)

- Your third task is basically TSP:
  - Given complete weighted graph  $G$ , integer  $k$ .
  - Is there a cycle that visits all vertices with cost  $\leq k$ ?
- One of the canonical problems.
- Note difference from Hamiltonian cycle:
  - graph is complete
  - we care about weight.

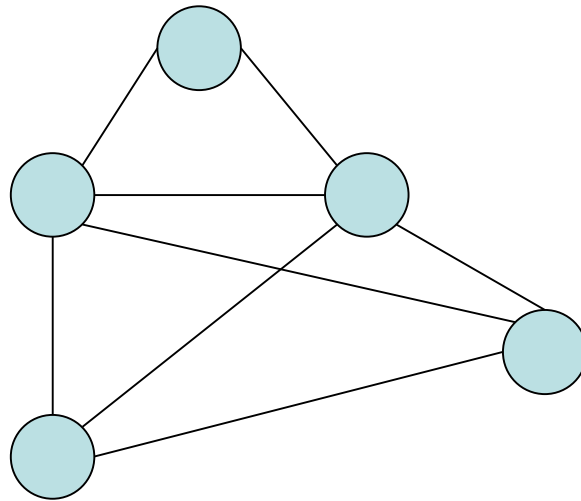
# Transforming Hamiltonian Cycle to TSP

- We can “reduce” Hamiltonian Cycle to TSP.
- Given graph  $G=(V, E)$ :
  - Assign weight of 1 to each edge
  - Augment the graph with edges until it is a complete graph  $G'=(V, E')$
  - Assign weights of 2 to the new edges
  - Let  $k = |V|$ .

## Notes:

- The transformation must take polynomial time
- You reduce the known NP-complete problem into your problem (not the other way around)
- In this case we are assuming Hamiltonian Cycle is our known NP-complete problem (in reality, both are known NP-complete)

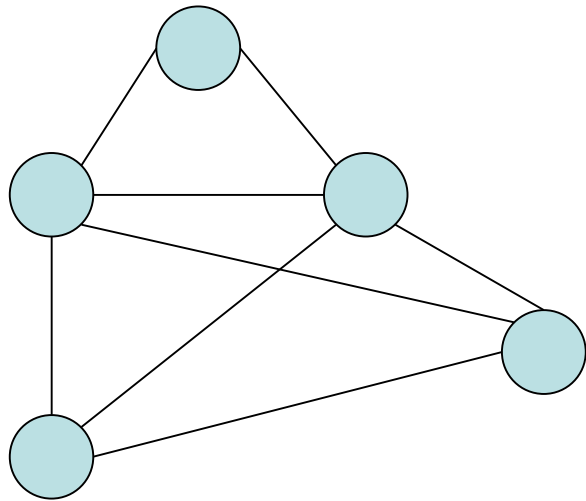
# Example



G

Input to Hamiltonian  
Circuit Problem

# Example

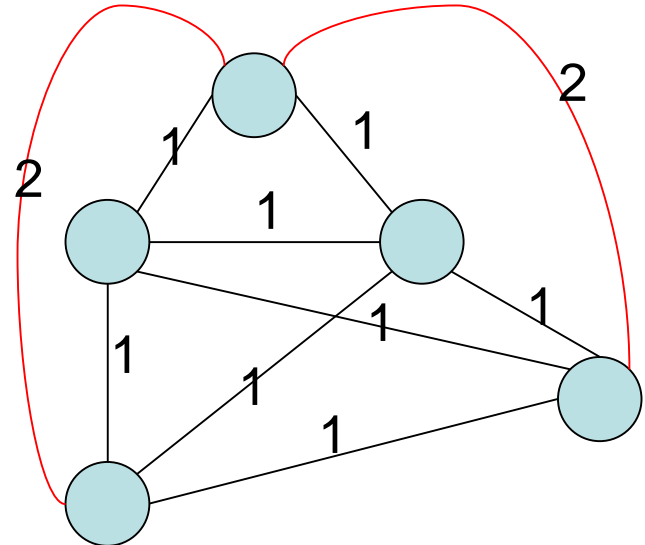


**G**

Input to Hamiltonian  
Circuit Problem



Polynomial time  
transformation



**G'**

Input to Traveling  
Salesman Problem



# Polynomial-time transformation

- $G'$  has a TSP tour of weight  $|V|$  iff  $G$  has a Hamiltonian Cycle.
- What was the cost of transforming HC into TSP?
- In the end, because there is a polynomial time transformation from HC to TSP, we say *TSP* is “at least as hard as” *Hamiltonian cycle*.

# What do we do about it?

- Approximation Algorithm:
  - Can we get an efficient algorithm that guarantees something *close* to optimal? (e.g. Answer is guaranteed to be within 1.5x of Optimal, but solved in polynomial time).
- Restrictions:
  - Many hard problems are easy for restricted inputs (e.g. graph is always a tree, degree of vertices is always 3 or less).
- Heuristics:
  - Can we get something that seems to work well (good approximation/fast enough) *most* of the time? (e.g. In practice,  $n$  is small-ish)

# Great Quick Reference

- *Computers and Intractability: A Guide to the Theory of NP-Completeness*, by Michael S. Garey and David S. Johnson

