NP-Completeness!

CSE 332 Spring 2021

Instructor: Hannah C. Tang

Teaching Assistants:

Aayushi Modi	Khushi Chaudhari	Patrick Murphy
Aashna Sheth	Kris Wong	Richard Jiang
Frederick Huyan	Logan Milandin	Winston Jodjana
Hamsa Shankar	Nachiket Karmarkar	

gradescope

gradescope.com/courses/256241

- You are at the bottom of a hill; you wish to be at the top. What are some ways you could achieve your goal?
  - Feel free to give serious and/or silly answers

- How does the Hamiltonian Circuit problem statement differ from the Travelling Salesman's?

Announcements

- Please nominate your TAs for the Bob Bandes TA Award!
  - https://www.cs.washington.edu/students/ta/bandes
- And help us learn to be better instructors!
  - Lecture: https://uw.iasystem.org/survey/242637/
  - AA:
  - AB:
  - AC:
  - AD:
  - AE:

Lecture Outline

- **Review: NP**

- Reductions, NP-Hard, and NP-Complete

- Bonus: NP-Intermediate

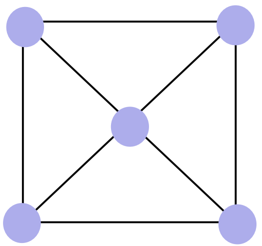
The Complexity Class NP

- ***NP*** is the set of all problems for which a given candidate solution can be verified in *polynomial worst-case time*
  - Compare against ***P***, which are the problems that can be solved in *polynomial worst-case time*
  - It is generally believed that P $\neq$ NP
    - i.e. there are problems in NP that are not in P

- Examples of problems in NP:
  - *Hamiltonian circuit*
  - *Satisfiability*
  - *Vertex Cover*
  - *All problems that are in P* ***(why???)***

- "NP" is **NOT** an abbreviation for "not polynomial"

Hamiltonian Circuit



- *Input*: A connected unweighted undirected graph G = (V, E)
- *Output:* A cycle visiting every vertex exactly once
- *Verification Algorithm*:
  - Traverse candidate path, marking visited vertices
  - Return true if all vertices are marked and $v_0 == v_n$
  - Runtime: $O(|V|)$
- *Solution Algorithm*:
  - Enumerate *all paths*, check if one of them is a circuit
    - Can use your favorite graph search algorithm to enumerate paths
  - Runtime: $O(2^{|V|})$

Travelling Salesman

- *Input*: A complete weighted undirected graph G=(V,E) and a number *m*

- *Output*: A circuit visiting each vertex exactly once and has total cost <=*m* (if such a circuit exists)

- *Verification Algorithm*:
  - Traverse candidate path, marking visited vertices
  - Return true if all vertices are marked, v_0 == v_n, and weight <= m
  - Runtime: O(|V|)

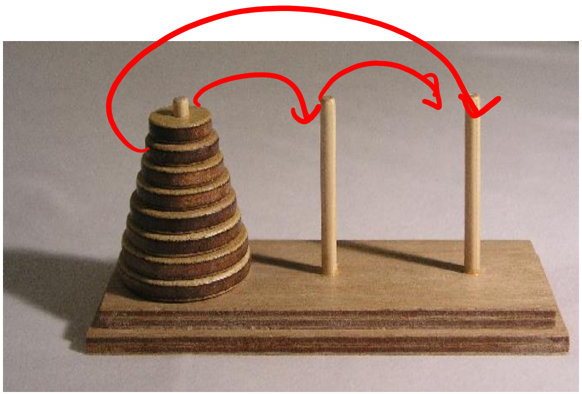
- *Solution Algorithm*:
  - Enumerate *all paths*, check if one is a circuit with appropriate weight
  - Runtime: $O(2^{|V|})$

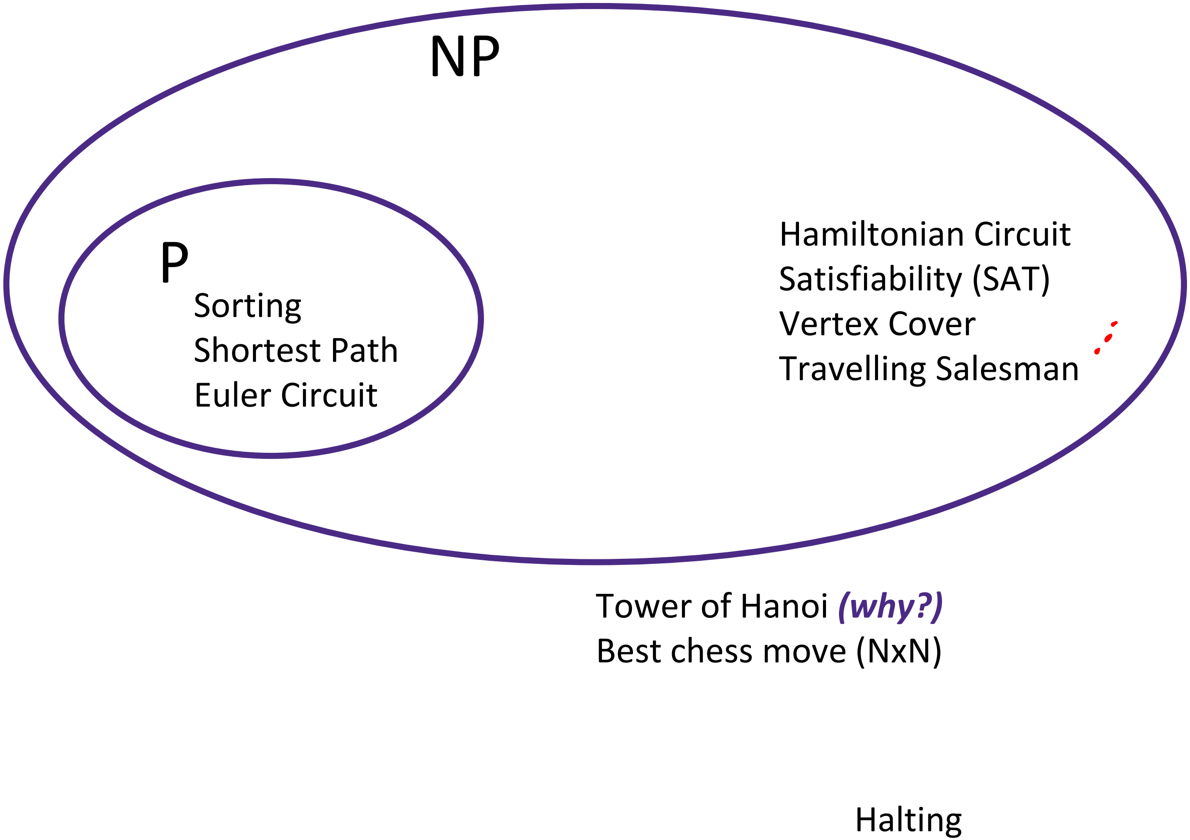
Tower of Hanoi

- *Input*: *n* disks of increasing size and 3 pegs
- *Output*: A series of moves transferring *n* disks to any other peg without placing a larger disk over a smaller one
- Algorithm:

```
while (!done):
  transferDisk(peg A, peg B)
  transferDisk(peg A, peg C)
  transferDisk(peg B, peg C)
```

  - Runtime: $O(2^n)$
  - Length of solution: $O(2^n)$




NP
P
Sorting
Shortest Path
Euler Circuit
Hamiltonian Circuit
Satisfiability (SAT)
Vertex Cover
Travelling Salesman
Tower of Hanoi *(why?)*
Best chess move (NxN)
Halting

Lecture Outline

- Review: NP
- **Reductions, NP-Hard, and NP-Complete**
- Bonus: NP-Intermediate

Reductions

- A ***reduction*** is using a solution for Problem Q to solve a different Problem P
  - "Problem P reduces to Q"

- Examples: P
  - "Climbing a hill" reduces to:
    - Flying
    - Helicopter
    - Piggyback ride from Hannah

    Q
  - Hamiltonian Circuit reduces to Travelling Salesman!

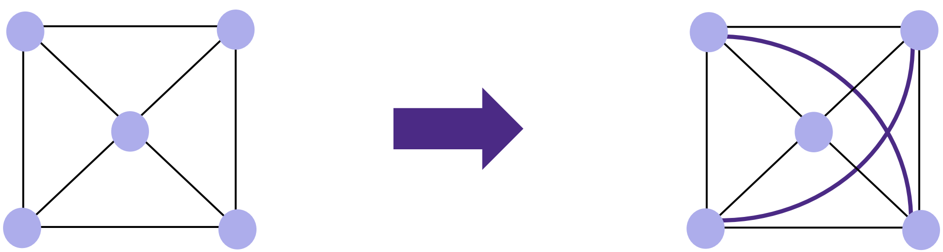
Reduction vs Decomposition

- A ***reduction*** is using a solution for Problem Q to solve a different Problem P
  - "Problem P reduces to Q"
  - We don't modify Problem Q's algorithm; we only modify Problem P's inputs/outputs to match Problem Q's expected inputs/outputs
  - "P reduces to Q" therefore means "P can be solved by Q"

- In contrast, ***decomposition*** is taking a task and breaking it into smaller parts
  - We might use "canned solutions" for solving these smaller parts

Hamiltonian Circuit to Travelling Salesman (1 of 3)

- Hamiltonian Circuit:
  - *Input*: A connected unweighted undirected graph
  - *Output:* A cycle visiting every vertex exactly once
- Travelling Salesman:
  - *Input*: A ① complete ③ weighted undirected graph and a number *m* ②
  - *Output*: A circuit visiting each vertex exactly once and has total cost <=*m* (if such a circuit exists)

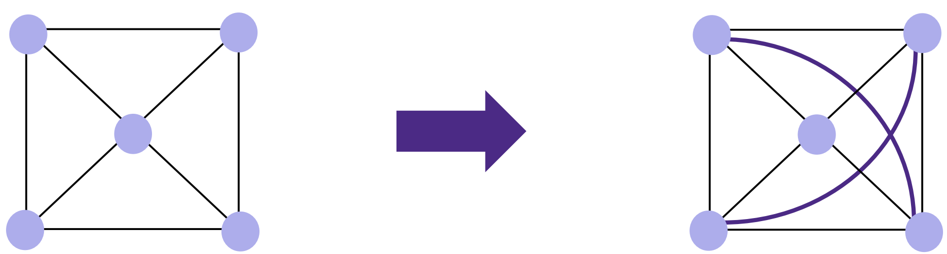


gradescope

gradescope.com/courses/256241

2 3

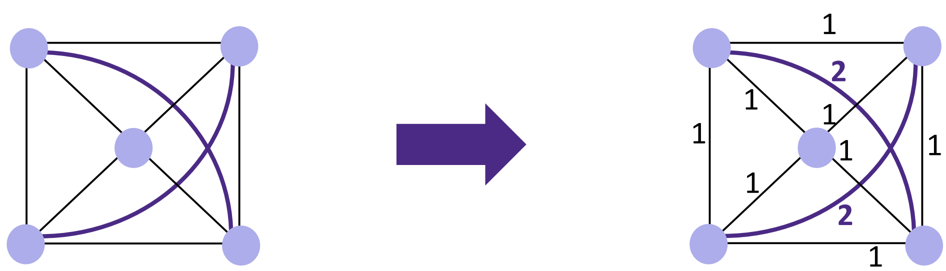
- What *m* and edge weights should we chose?
  - The existing black edges need to "look unweighted"
  - We don't want to use any of the added purple edges



Hamiltonian Circuit to Travelling Salesman (2 of 3)

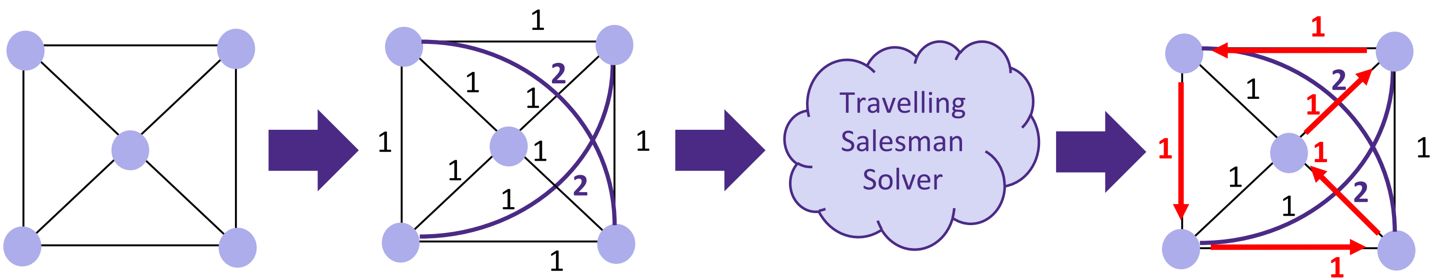
- What *m* and edge weights should we chose?
  - The existing black edges need to "look unweighted"
  - We don't want to use any of the added purple edges

- Let the weight of all the *existing* edges be 1
- Let the weight of the *added* edges be a large number
  - Infinity? |V|? |E|? 2?
- Let *m* = |V|

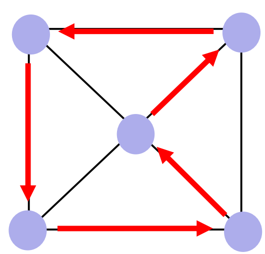


Hamiltonian Circuit to Travelling Salesman (3 of 3)

- Assume there exists an algorithm to solve Travelling Salesman. How do we adapt its output to fit Hamiltonian Circuit's output?



  - Remove the added weights and edges!

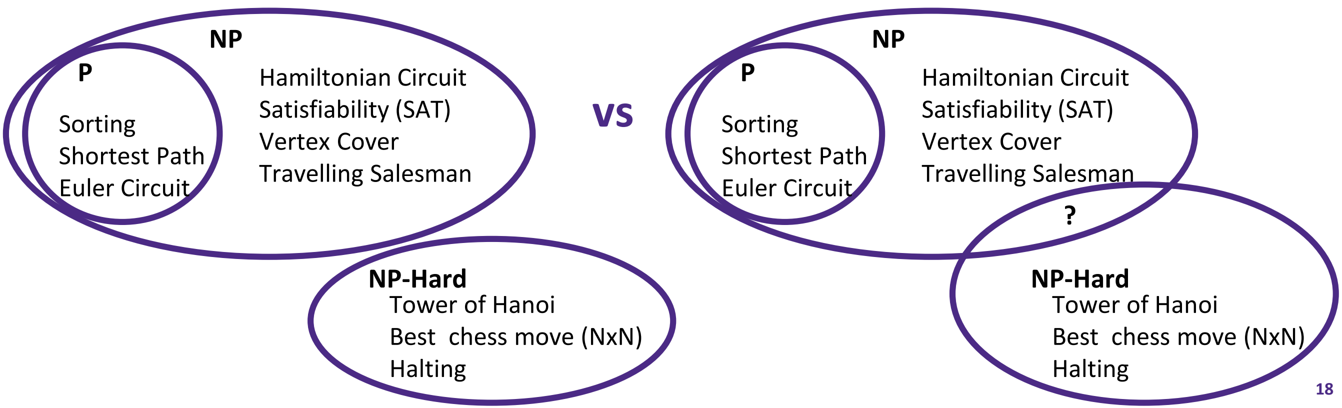


Timing Our Reduction

- Runtime of our reduction: $O(V^2)$
  - Don't include the runtime of our TSP solver
  - Adapting the Input: $O(V^2)$
    - Adding enough edges to complete the graph: $O(V^2)$
    - Adding weights to all the (complete) edges: $O(V^2)$
  - Adapting the Output:
    - Removing added edges: $O(V^2)$
    - Removing weights: $O(V^2)$

The Complexity Class NP-Hard

- ***NP-hard*** is the set of all problems to which *every problem in NP* can be reduced in *polynomial time*
  - Example problem: *Tower of Hanoi*
- There are problems in NP-hard that aren't in NP. Are there problems in NP that are NP-hard?
  - i.e.: are there problems *in NP* that to which *everything else in NP* can be reduced?



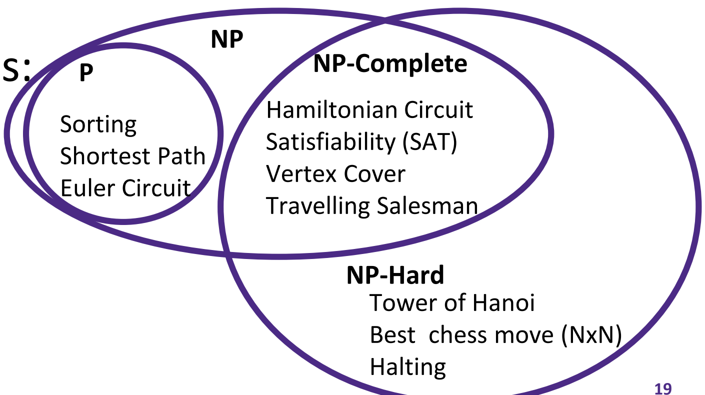
The Complexity Class NP-Complete

- ***NP-complete*** is the (non-empty!) set of NP-hard problems that are *also in NP*
  - i.e. the set of all problems for which a given candidate solution can be verified in poly~~no~~mial worst-case time and to which every problem in NP can be reduced in polynomial worst-case time
  - As with NP, we're pretty sure they can't be solved in polynomial time
  - These are thought of as the *hardest* problems in the class NP

NP

NP Hard

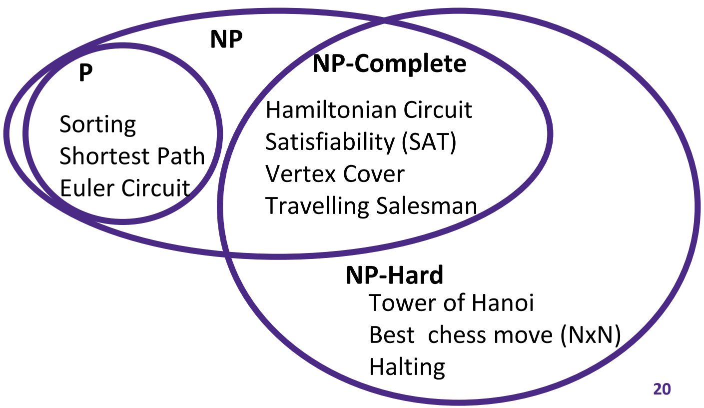
- Example NP-complete problems:
  - Hamiltonian Circuit
  - Satisfiability
  - Vertex Cover
  - Travelling Salesman



Cool Corollaries of NP-Complete

- If any NP-complete problem can be solved in polynomial time, then *all NP-complete problems can be solved in polynomial time*
  - NP-complete problems are also in NP
  - Reduce all other NP-complete problems to the polynomial-solvable one (using a polynomial time reduction), then solve

- If any NP-complete problem is in P, then *all of NP is in P*
  - All problems in NP are reducible to an NP-complete problem
  - Reduce all other NP problems to the polynomial-solvable one, then solve



- Also …

Reductions, Redux

- Thus far, we've thought of "A reduces to B" as "unknown-problem A can be solved with known-problem B's solution"
  unknown known
  - "A reduces to B" therefore means "A can be solved by B"
  - You can use reductions in the opposite sense!

- "A reduces to B" also means "known-problem A is 'no harder' than unknown-problem B"
  known unknown
  " $A_{\text{difficulty}} \leq B_{\text{difficulty}}$ "
  - Proof by contradiction:
    - Suppose B is "easy" and A is "hard"
    - Because A reduces to B, every instance of A is solvable by transforming it to an instance of B
    - Therefore A is "easy" – **CONTRADICTION!**

- This is why we say NP-complete are "the hardest NP problems"
  - NP-complete = NP-hard problems that are also in NP
  - NP-hard = problems to which every problem in NP can be reduced

A Hard Problem

- 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*

- *(Let's also pretend you were playing Animal Crossing during our revelation that Travelling Salesman Problem is NP-complete)*

What to do with a Hard Problem

- Your problem (i.e., TSP) seems really hard

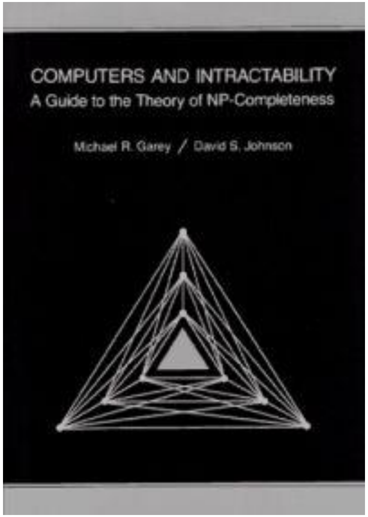
- If you can transform a known NP-complete problem into the one you're trying to solve using a polynomial time reduction, then you know your problem is *at least* NP-complete
  - Hamiltonian Circuit (which you *do* recall as NP-complete) can be reduced to TSP in polynomial time
  - Therefore your boss's task (TSP) is at least NP-complete

What to do with an NP-Complete Problem

- Approximation:
  - Can we get an efficient algorithm that guarantees "close to optimal"?
    - e.g.: answer guaranteed to be ≤1.5x of optimal, but algorithm runs in polynomial time
- Restrictions:
  - Many hard problems are easy with 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" most of the time?
    - e.g.: good approximation/fast enough if n is smallish

Great Quick Reference

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

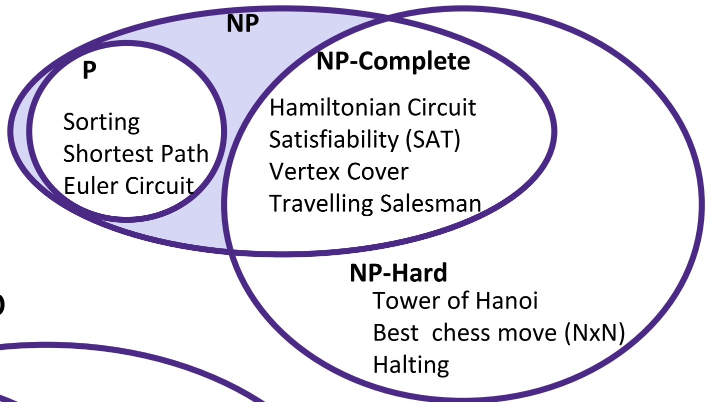


Lecture Outline

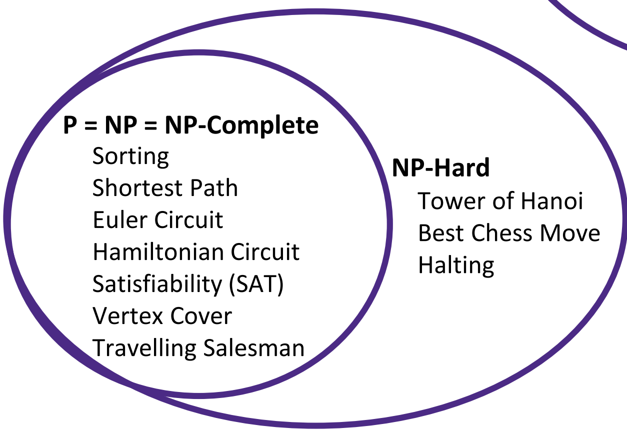
- Review: NP
- Reductions, NP-Hard, and NP-complete
- **Bonus: NP-Intermediate**

NP-Intermediate (1 of 2)

- Is there anything in NP that's not NP-complete?

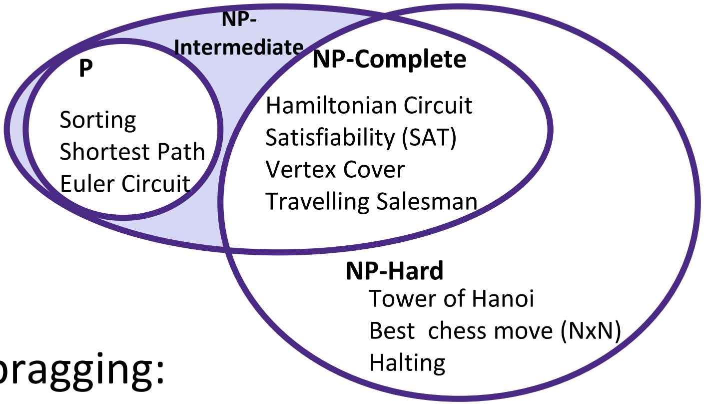


- If P = NP: no



NP-Intermediate (2 of 2)

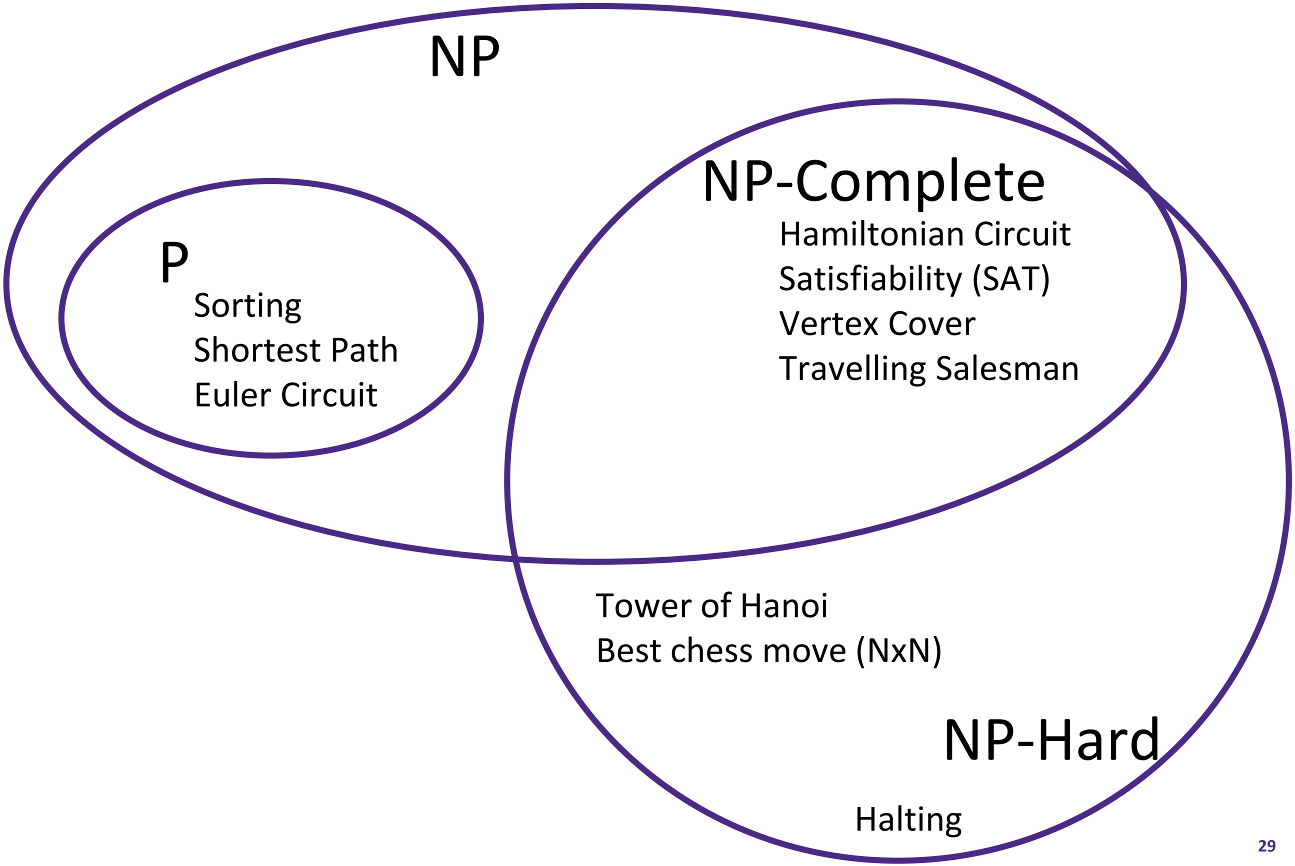
- If P $\neq$ NP, then this set is called NP-intermediate



- More local bragging:
  - In 1975, R. Ladner proved NP-intermediate is not empty if P $\neq$ NP
  - We suspect the following problems are in NP-intermediate:
    - Factoring
    - Graph isomorphism
    - Discrete logarithm


1968? 1973?


Recent


NP
P
Sorting
Shortest Path
Euler Circuit
NP-Complete
Hamiltonian Circuit
Satisfiability (SAT)
Vertex Cover
Travelling Salesman
Tower of Hanoi
Best chess move (NxN)
NP-Hard
Halting

Summary

- P and NP are defined in terms of different attributes (i.e., solvability vs verifiability), and we suspect that this means P is a proper subset of NP

- Thanks to reductions, NP-Complete problems are the "hardest" in NP. They are also "characteristic" of NP problems

- There are multiple complexity classes outside of NP – take 431!