CSE 332 Summer 2024
Lecture 23: P & NP

Nathan Brunelle

http://www.cs.uw.edu/332

Tractability

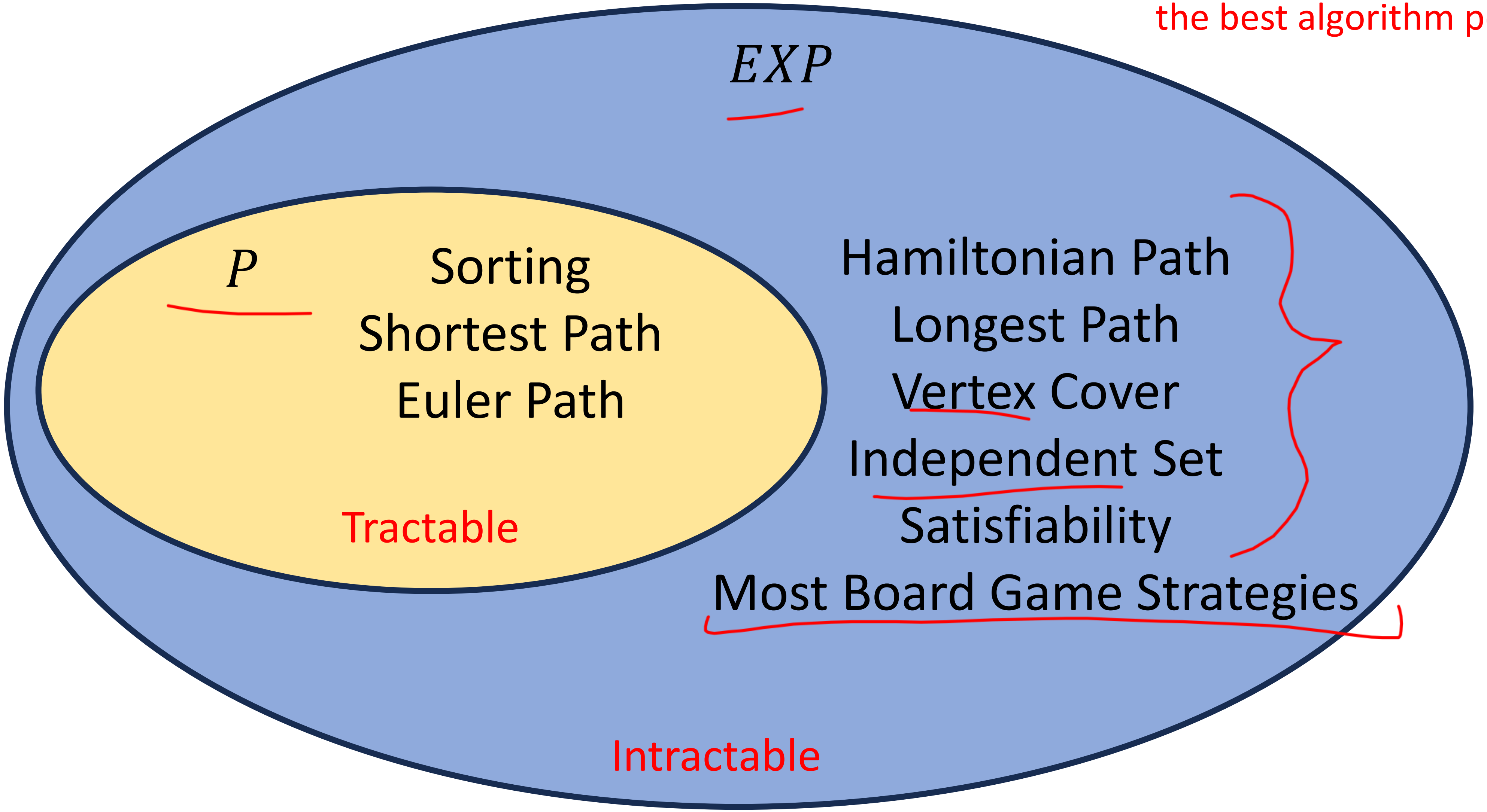
- Tractable:
  - Feasible to solve in the "real world"
- Intractable:
  - Infeasible to solve in the "real world"
- Whether a problem is considered "tractable" or "intractable" depends on the use case
  - For machine learning, big data, etc. tractable might mean $\mathrm{O}(n)$ or even $O(\log n)$
  - For most applications it's more like $O(n^3)$ or $O(n^2)$
- A strange pattern:
  - Most "natural" problems are either done in small-degree polynomial (e.g. n^2) or else exponential time (e.g. 2^n)
  - It's rare to have problems which require a running time of n^5, for example

Complexity Classes and Tractability

- To explore what problems are and are not tractable, we give some complexity classes special names:
- Complexity Class P:
  - Stands for "Polynomial"
  - The set of problems which have an algorithm whose running time is $O(n^p)$ for some choice of $p \in \mathbb{R}$.
  - We say all problems belonging to P are "Tractable"
- Complexity Class EXP:
  - Stands for "Exponential"
  - The set of problems which have an algorithm whose running time is $O\left(2^{n^p}\right)$ for some choice of $p \in \mathbb{R}$
  - We say all problems belonging to $EXP - P$ are "Intractable"
    - Disclaimer: Really it's all problems outside of P, and there are problems which do not belong to EXP, but we're not going to worry about those in this class

Members

Important!
Some of the problems listed in EXP could also be members of P
Since membership is determined by a problem's *most* efficient algorithm, knowing if a problem belongs to P requires knowing the best algorithm possible!



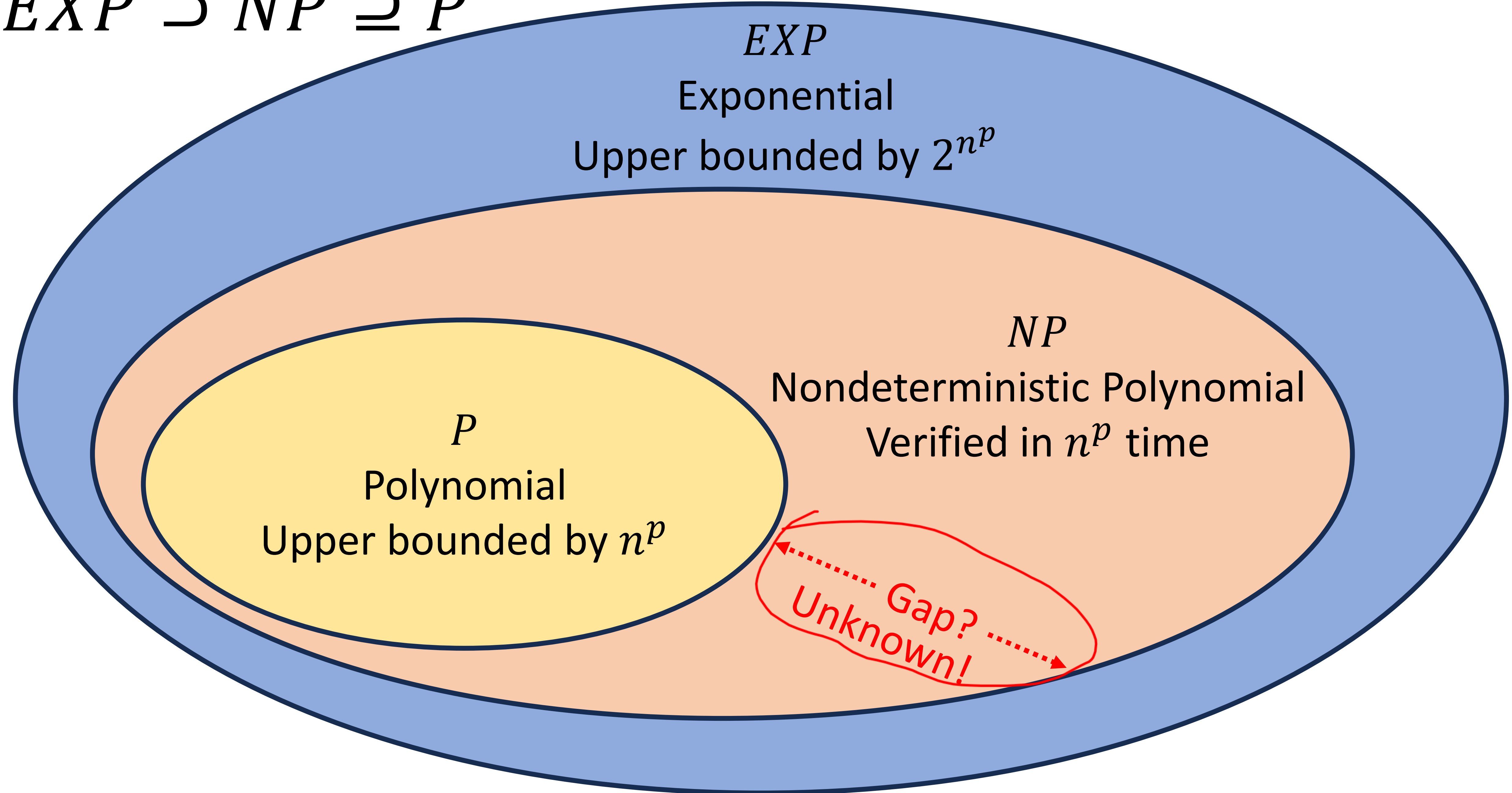
Class NP

- NP
  - The set of problems for which a candidate solution can be verified in polynomial time
  - Stands for "Non-deterministic Polynomial"
    - Corresponds to algorithms that can guess a solution (if it exists), that 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 step of an exhaustive search
- $P \subseteq NP$
  - Why?

$P = NP$?

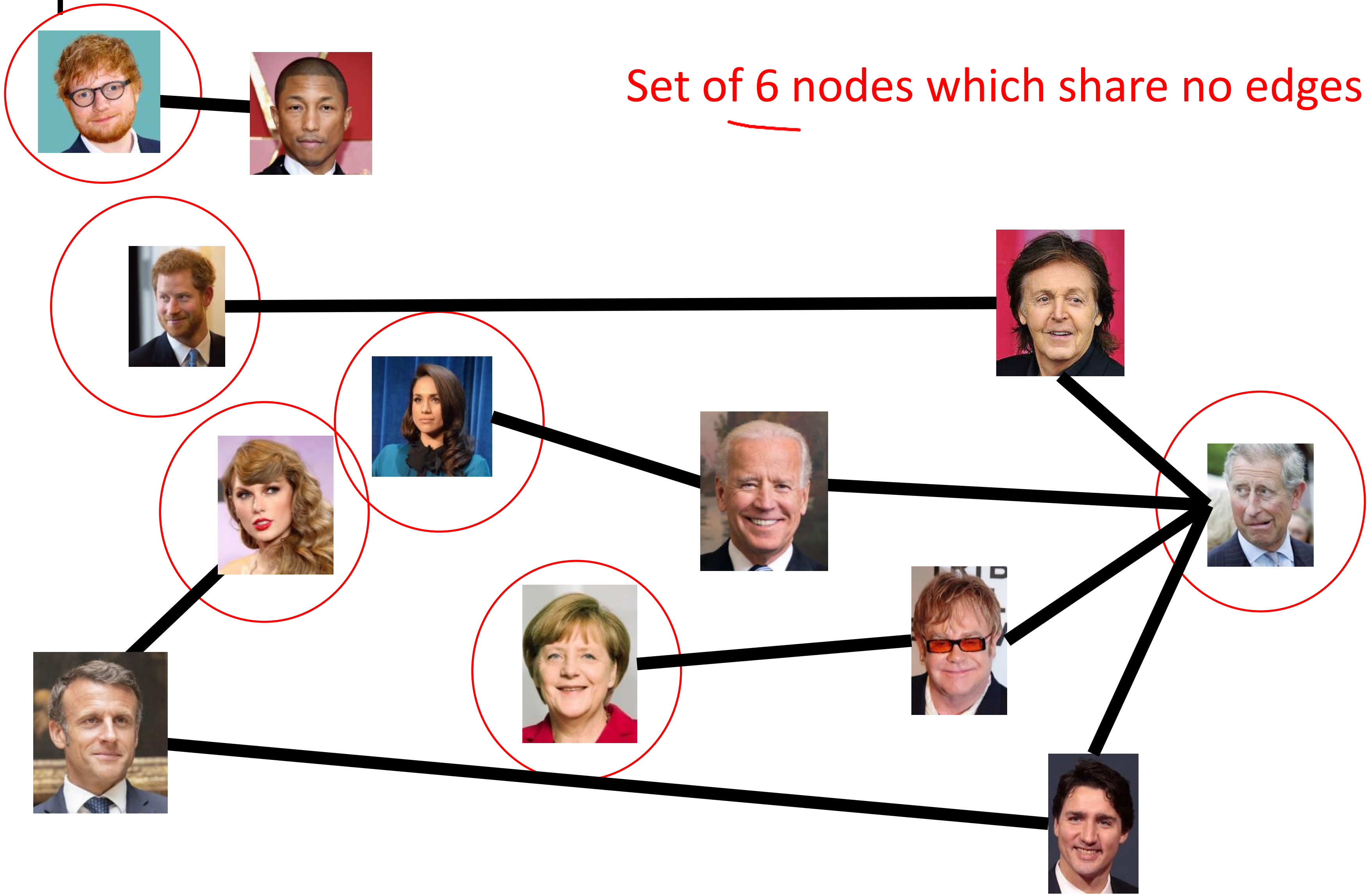
$P \subset NP$

$$EXP \supset NP \supseteq P$$

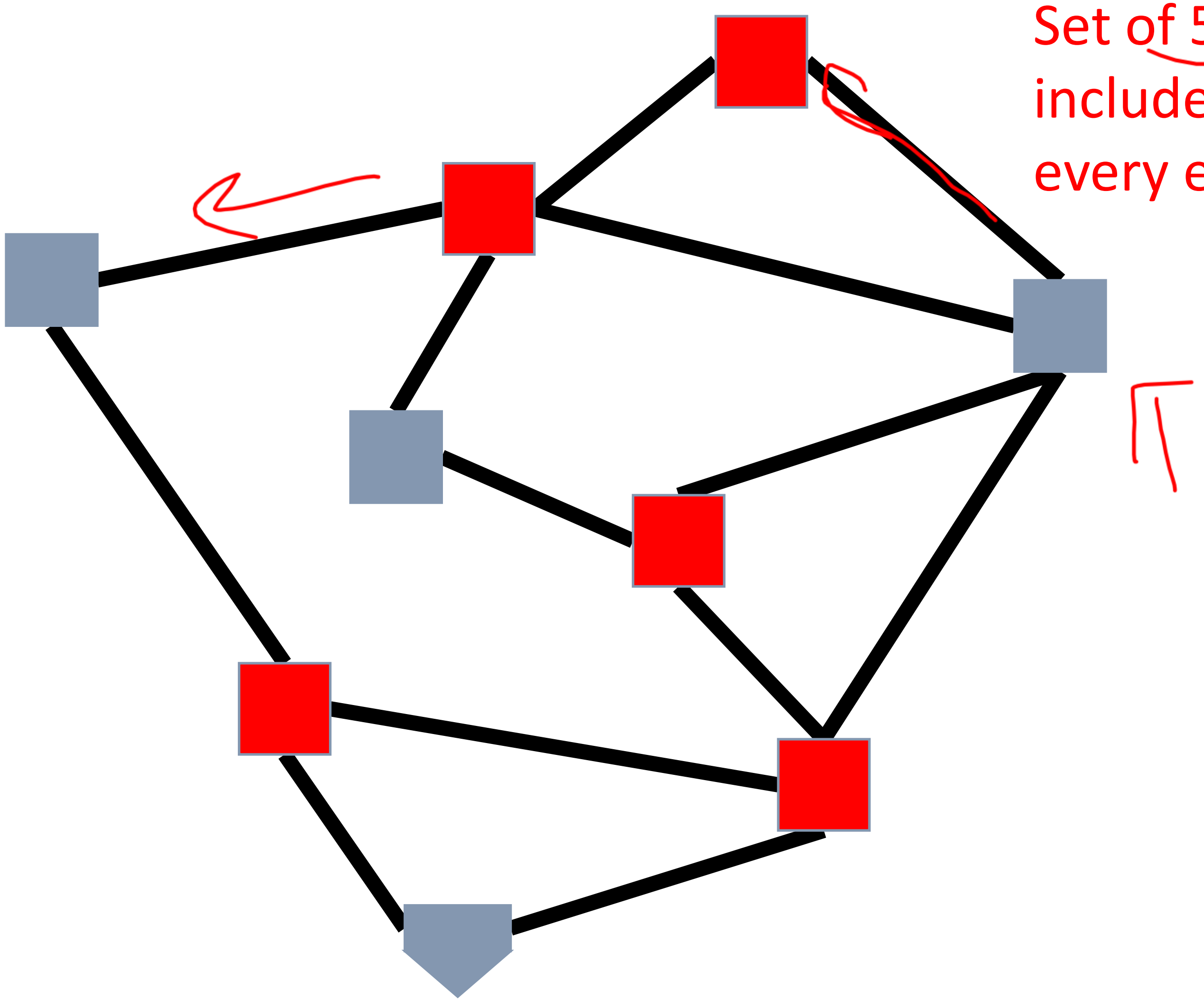


Independent Set

Set of 6 nodes which share no edges



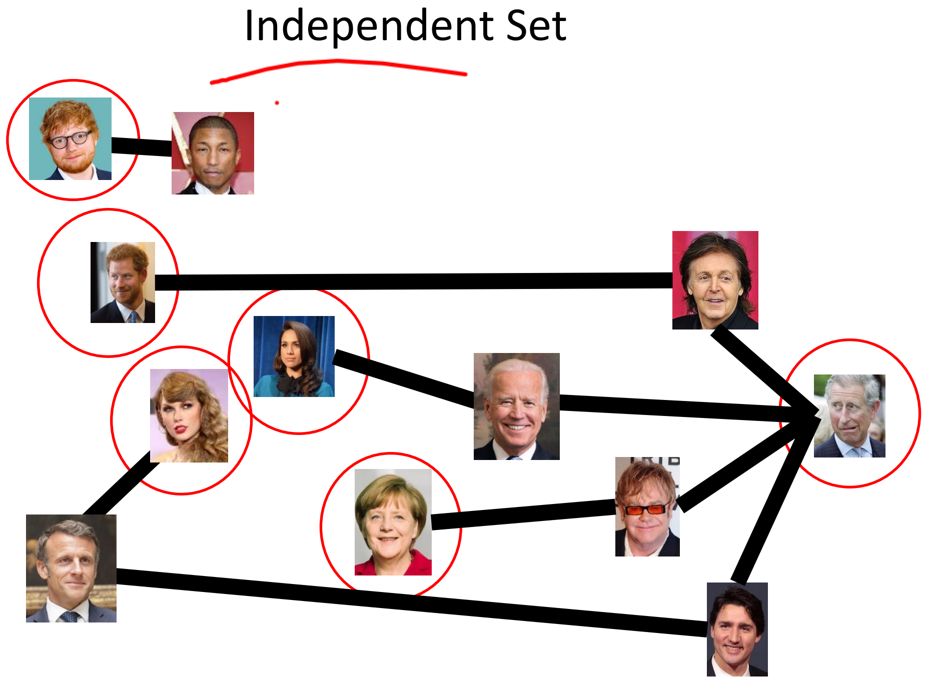
Vertex Cover

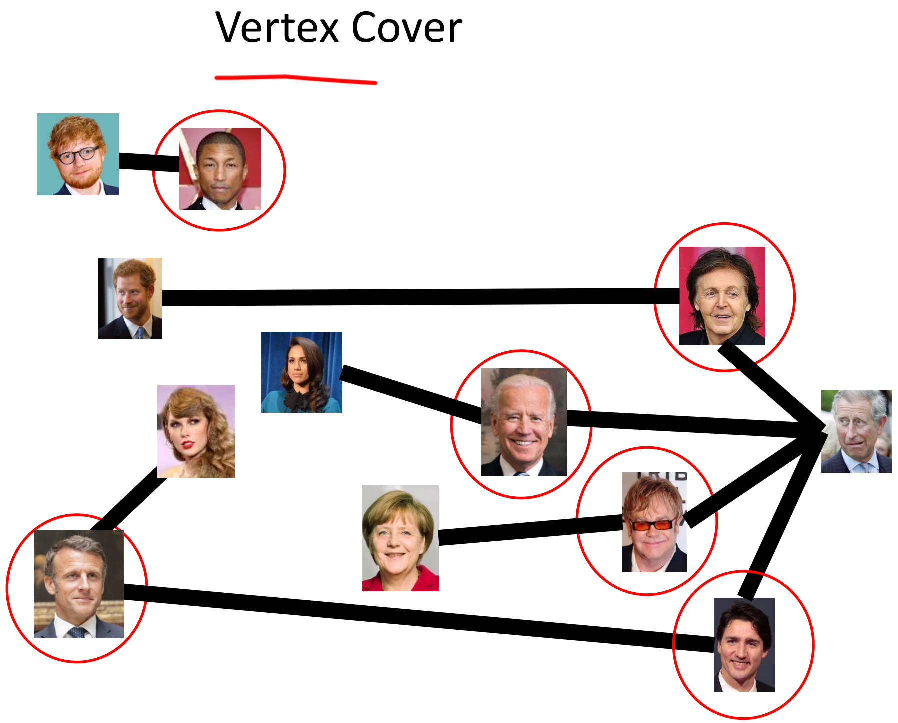


Set of 5 nodes which includes an endpoint of every edge

Way Cool!

S is an independent set of G iff $V - S$ is a vertex cover of G

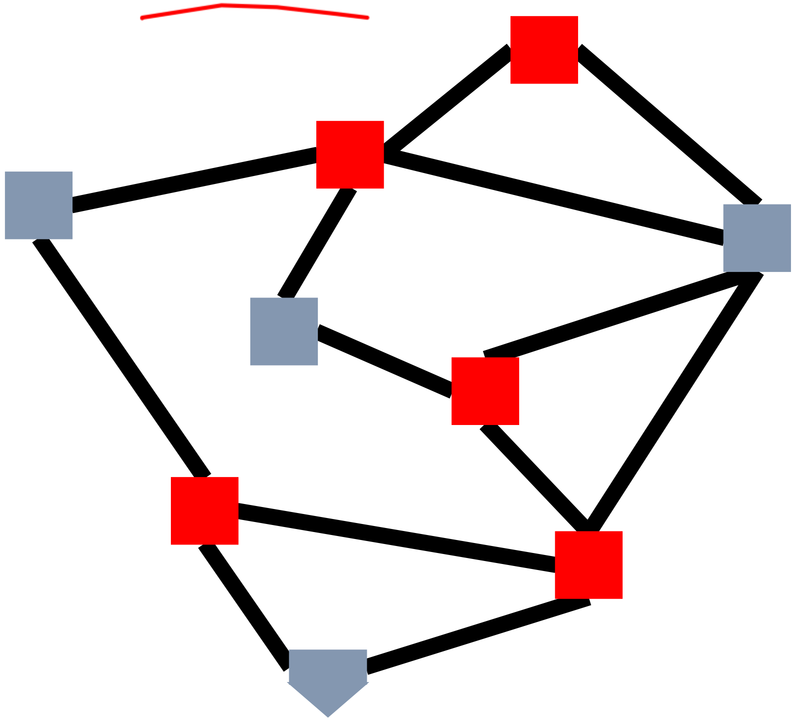




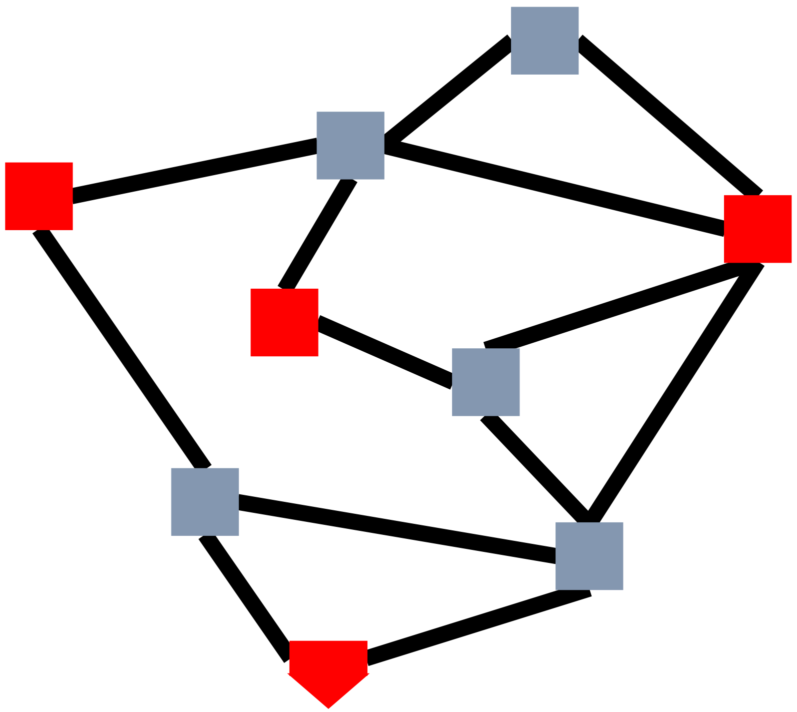
Way Cool!

S is an independent set of G iff $V - S$ is a vertex cover of G

Vertex Cover



Independent Set



Solving Vertex Cover and Independent Set

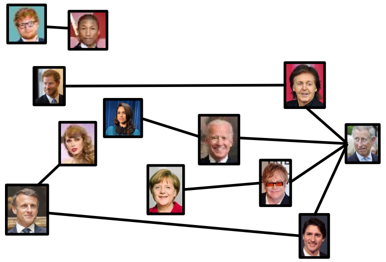
- Algorithm to solve vertex cover
  - Input: $G = (V, E)$ and a number k
  - Output: True if G has a vertex cover of size k
    - Check if there is an Independent Set of G of size $|V| - k$
- Algorithm to solve independent set
  - Input: $G = (V, E)$ and a number k
  - Output: True if G has an independent set of size k
    - Check if there is a Vertex Cover of G of size $|V| - k$

Either both problems belong to P, or else neither does!

We need to build this Reduction

Independent Set Input

k



O(V) Time

Relate Independent Set input to Vertex Cover output

Use same graph

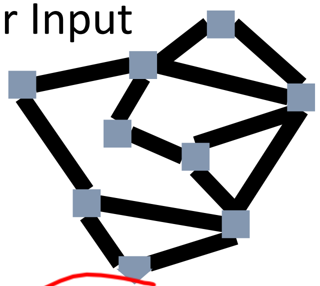
Set $k = |V| - k$

Relate Independent Set output to Vertex Cover output

Give same output

Reduction

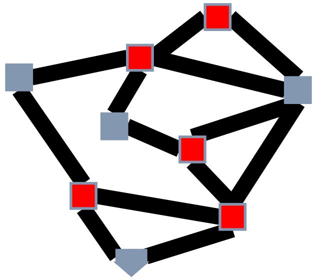
Vertex Cover Input



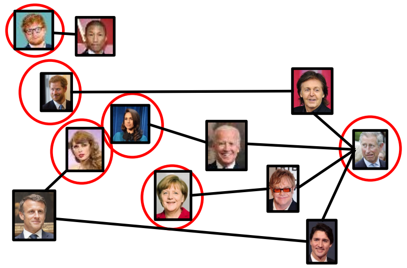
k

Any Algorithm for Vertex Cover

Vertex Cover Ouput



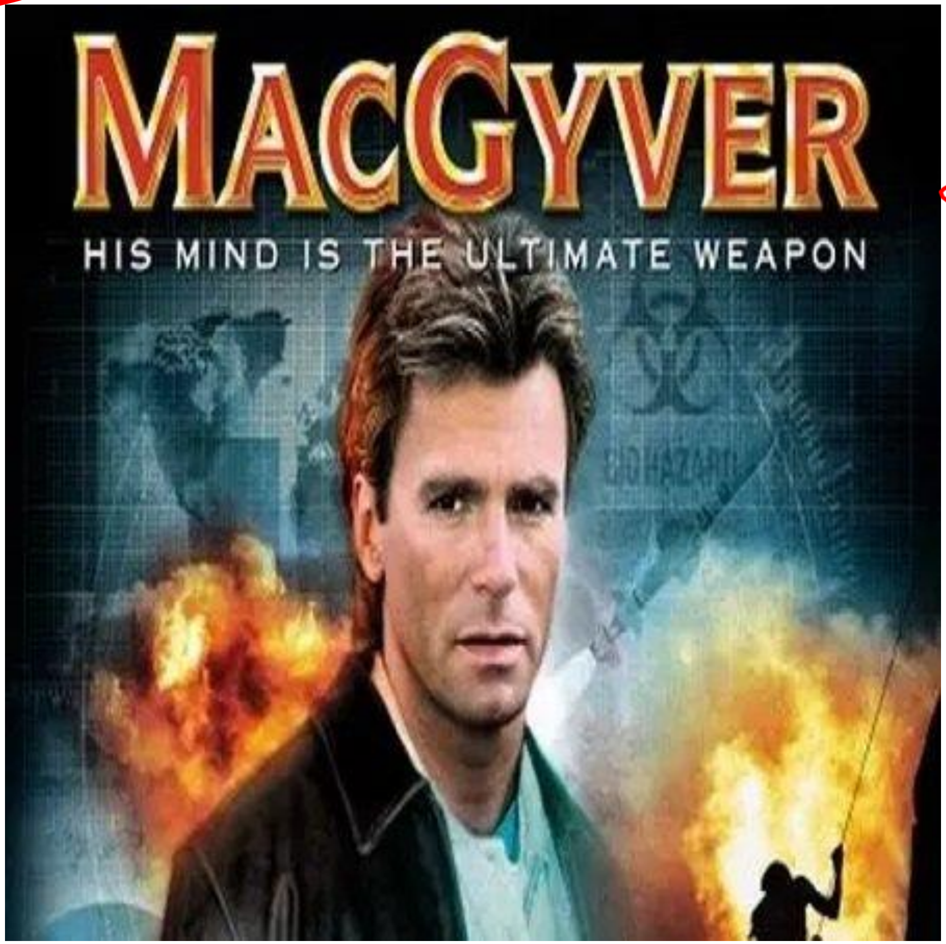
Independent Set Output



Reductions

Shows how two different problems relate to each other

MOVIE TIME!



MacGyver's Reduction

Problem we don't know how to solve

A

Opening a door

Solution for $\boldsymbol{A}$

Keg cannon battering ram

Aim duct at door, insert keg

Put fire under the Keg

Reduction

Problem we do know how to solve

B

Lighting a fire

How?

Solution for $\boldsymbol{B}$

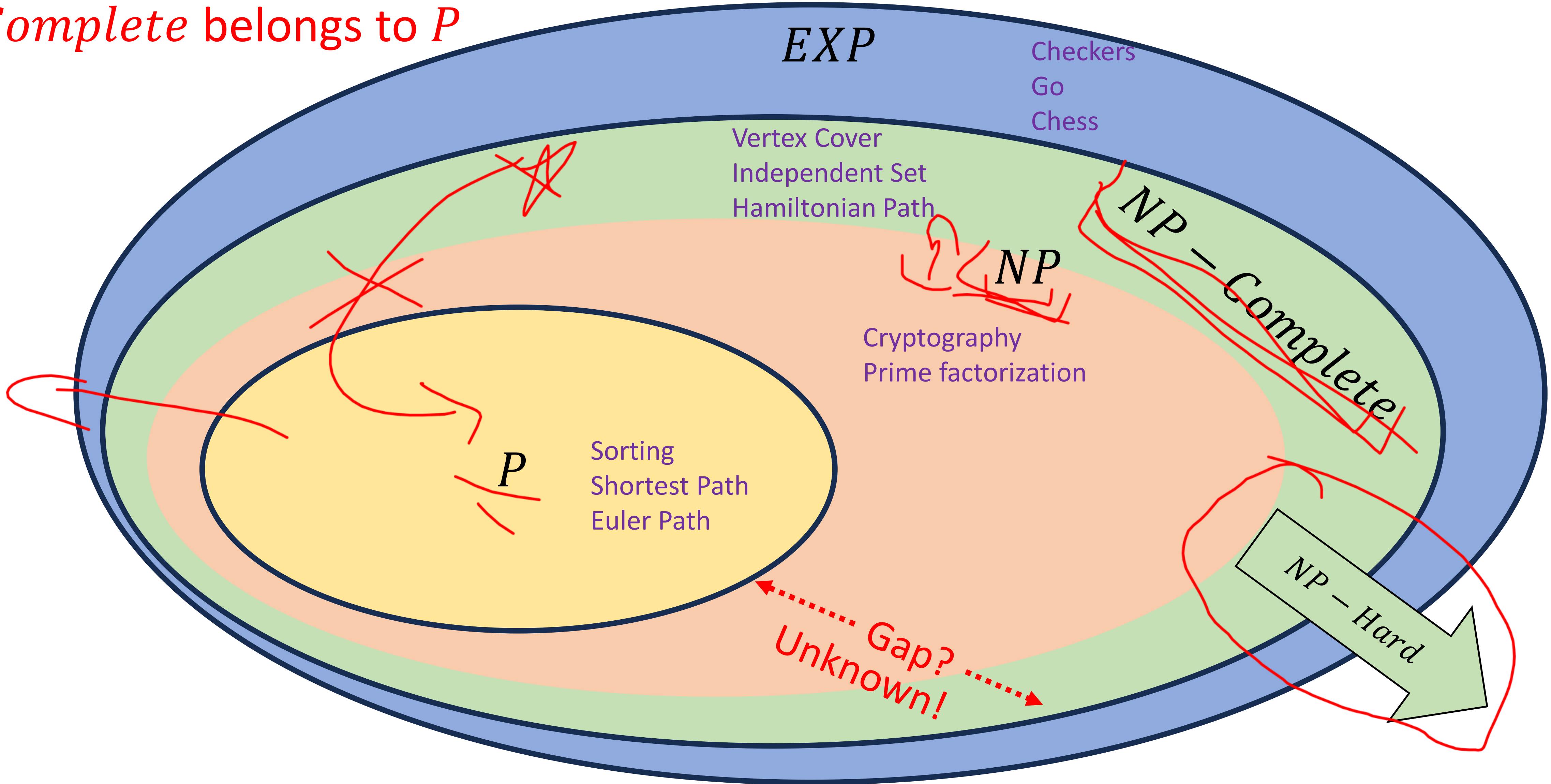
Alcohol, wood, matches

NP-Complete

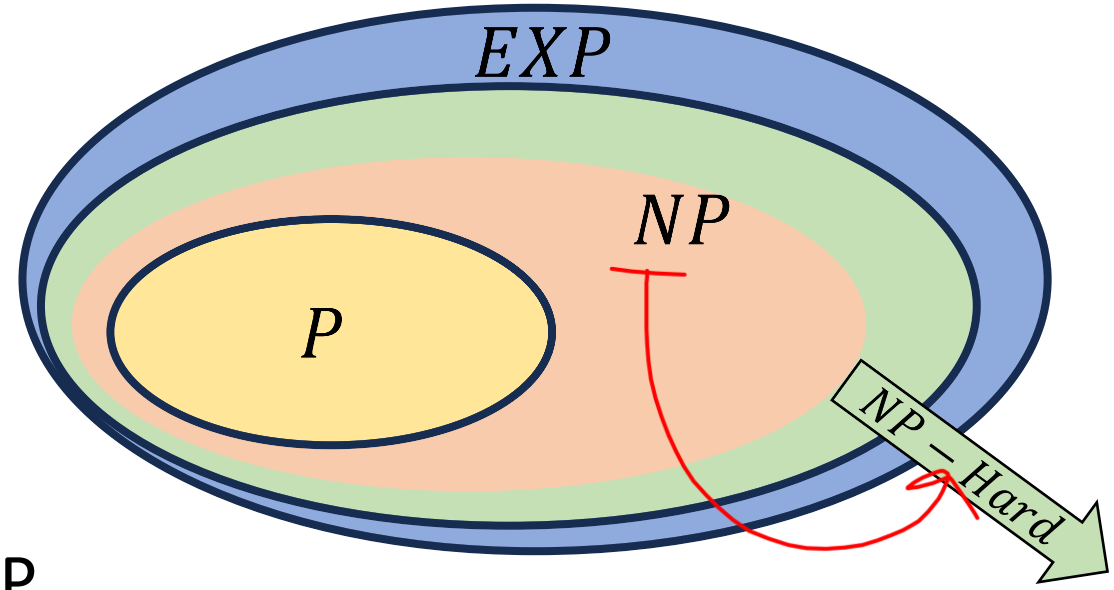
- A set of "together they stand, together they fall" problems
- The problems in this set either all belong to P, or none of them do
- Intuitively, the "hardest" problems in NP
- Collection of problems from NP that can all be "transformed" into each other in polynomial time
  - Like we could transform independent set to vertex cover, and vice-versa
  - We can also transform vertex cover into Hamiltonian path, and Hamiltonian path into independent set, and …

$$EXP \supset NP - Complete \supseteq NP \supseteq P$$

$P = NP$ iff some problem from $NP - Complete$ belongs to P

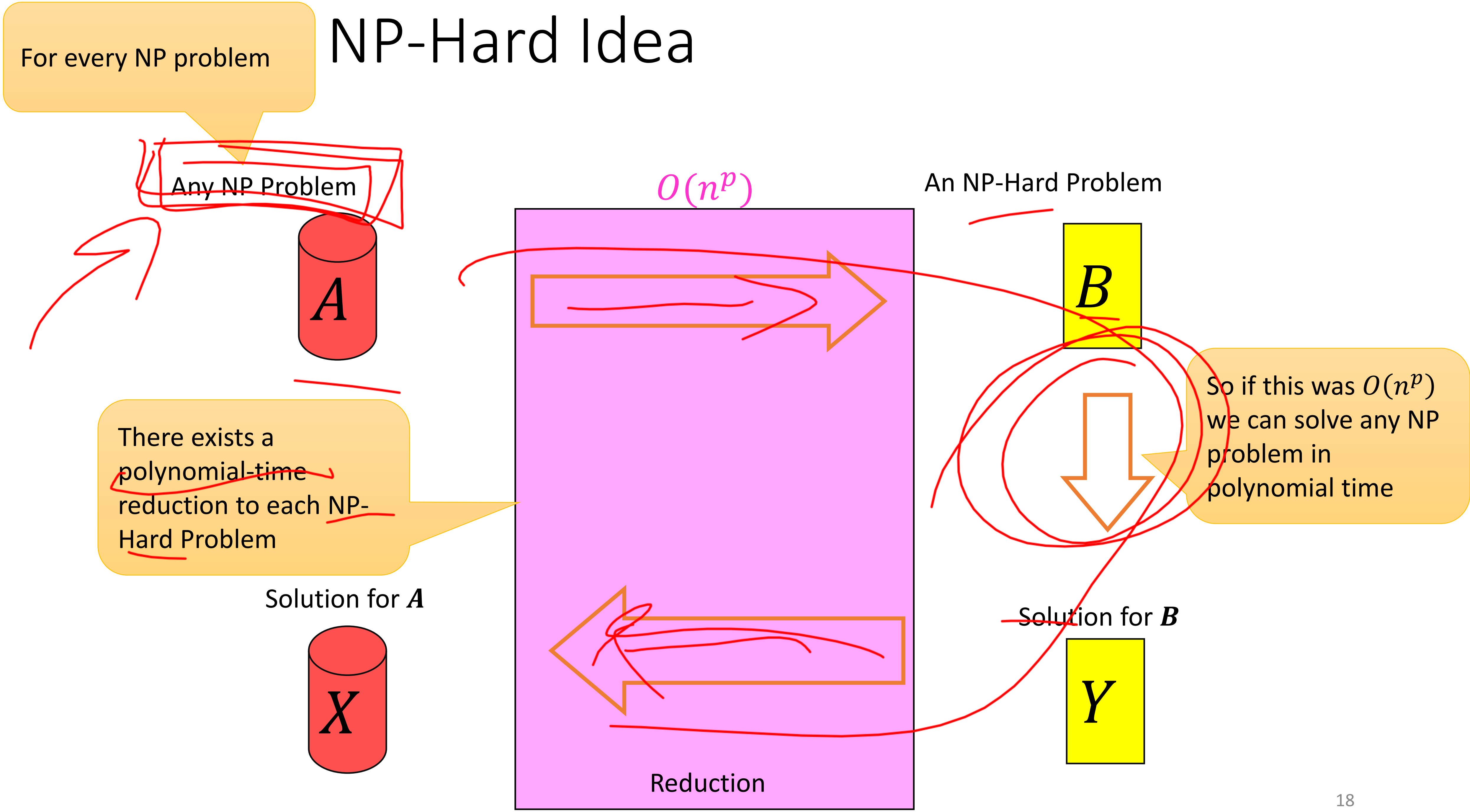


NP-Hard



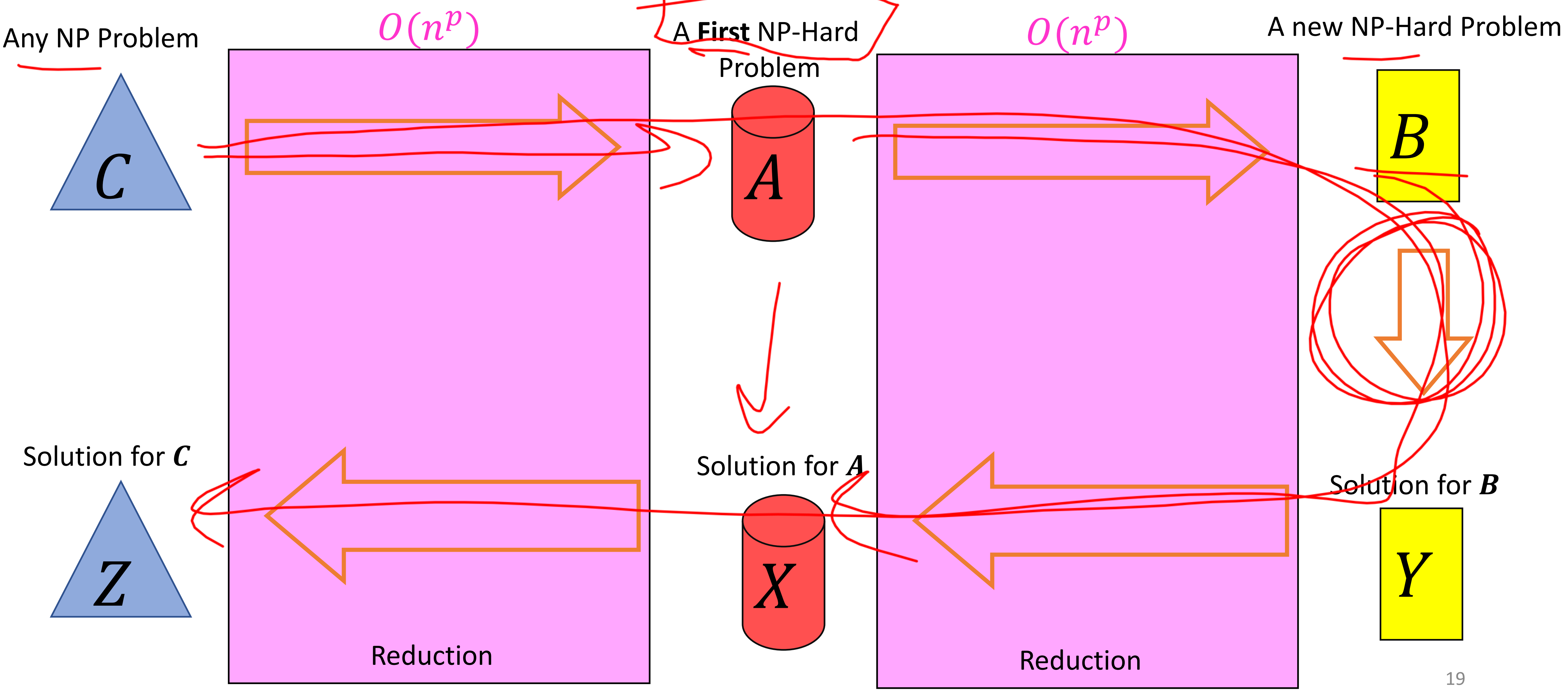
- How can we try to figure out if P=NP?
- Identify problems at least as "hard" as NP
  - If any of these "hard" problems can be solved in polynomial time, then all NP problems can be solved in polynomial time.
- Definition: NP-Hard:
  - B is NP-Hard provided EVERY problem within NP reduces to B in polynomial time

NP-Hard Idea

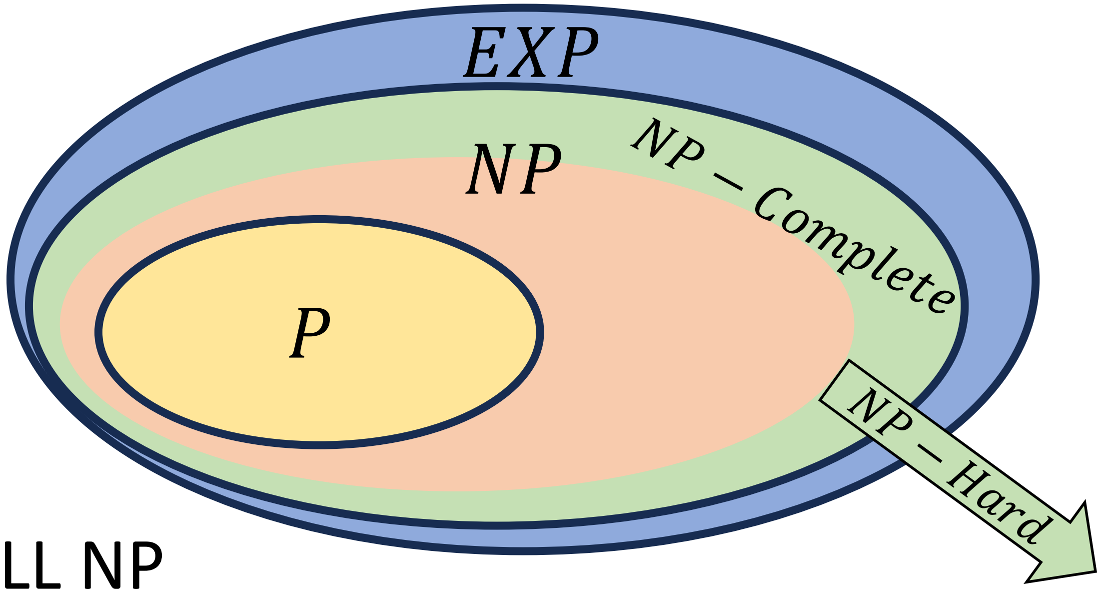


For every NP problem
Any NP Problem
A
$O(n^p)$
An NP-Hard Problem
B
So if this was $O(n^p)$ we can solve any NP problem in polynomial time
There exists a polynomial-time reduction to each NP-Hard Problem
Solution for $\boldsymbol{A}$
X
Solution for $\boldsymbol{B}$
Y
Reduction

Showing NP-Hardness

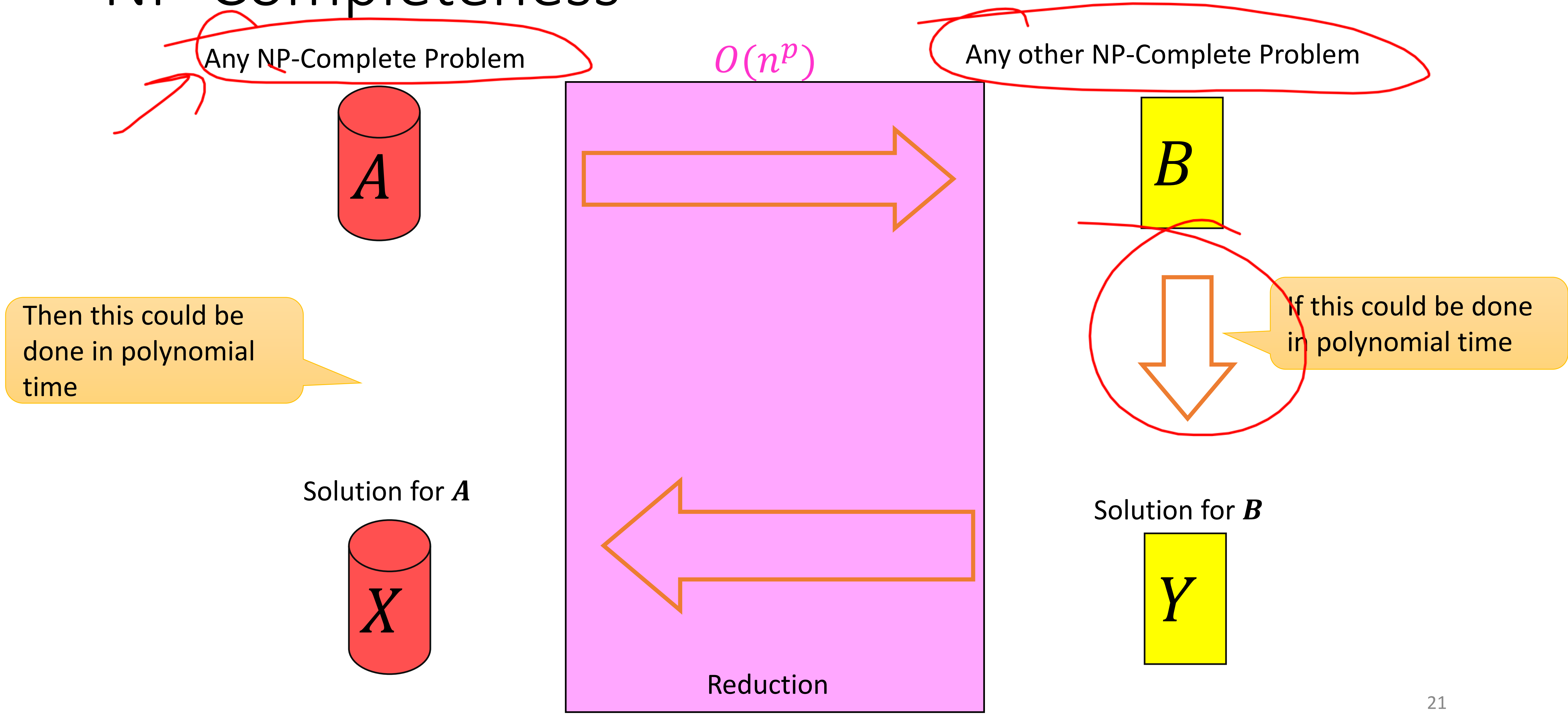


NP-Complete

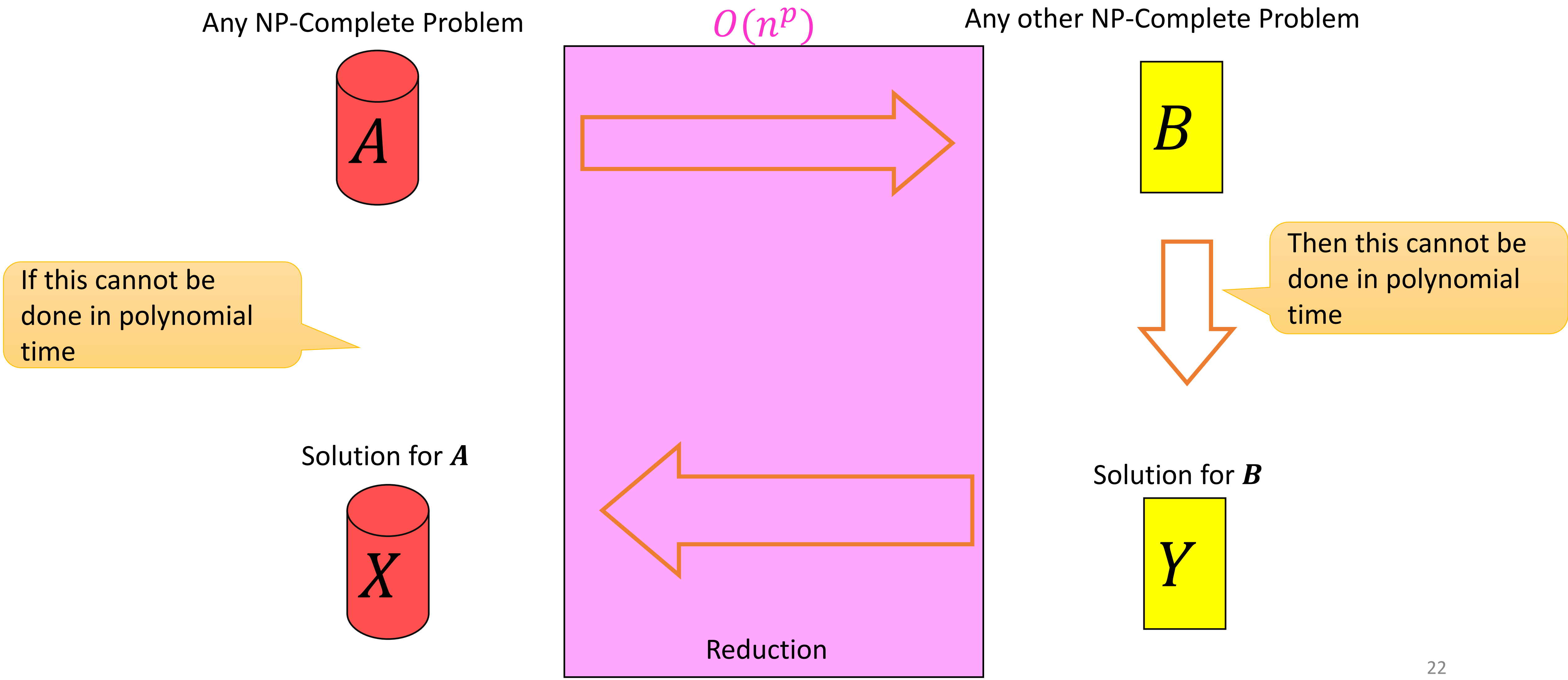


- "Together they stand, together they fall"
- Problems solvable in polynomial time iff ALL NP problems are
- NP-Complete = NP ∩ NP-Hard
- **How to show a problem is NP-Complete?**
  - Show it belongs to NP
    - Give a polynomial time verifier
  - Show it is NP-Hard
    - Give a reduction from another NP-H problem

NP-Completeness



NP-Completeness



Overview

- Problems not belonging to P are considered intractable
- The problems within NP have some properties that make them seem like they might be tractable, but we've been unsuccessful with finding polynomial time algorithms for many
- The class $NP - Complete$ contains problems with the properties:
  - All members are also members of NP
  - All members of NP can be transformed into every member of $NP - Complete$
    - Because they are both NP and $NP - Hard$
  - If any one member of $NP - Complete$ belongs to P, then $P = NP$
  - If any one member of $NP - Complete$ is outside of P, then $P \neq NP$

Why should YOU care?

- If you can find a polynomial time algorithm for any $NP - Complete$ problem then:
  - You will win $1million
  - You will win a Turing Award
  - You will be world famous
  - You will have done something that no one else on Earth has been able to do in spite of the above!
- If you are told to write an algorithm a problem that is $NP - Complete$
  - You can tell that person everything above to set expectations
  - Change the requirements!
  - **Approximate the solution**: Instead of finding a path that visits every node, find a path that visits at least 75% of the nodes
  - **Add Assumptions**: problem might be tractable if we can assume the graph is acyclic, a tree
  - **Use Heuristics**: Write an algorithm that's "good enough" for small inputs, ignore edge cases

Why should YOU care? $P \neq NP$

- The entire field of cryptography relies on it (nearly at least)
  - Requires decrypting with a key is easier than decrypting without a key
    - This is strongly related to requiring a difference in difficulty between verifying a candidate solution and finding a solution in the first place
- If $P \neq NP$
  - Some problems remain intractable
  - Cryptography persists
- If $P = NP$
  - We may get efficient solutions for important problems
  - Cryptography is potentially doomed.

Does P=NP?

	P≠NP	P=NP	Ind	DC	DK	DK and DC	other
2002	61 (61%)	9 (9%)	4 (4%)	1 (1%)	22 (22%)	0 (0%)	3 (3%)
2012	126 (83%)	12 (9%)	5 (3%)	5 (3%)	1 (0.66%)	1 (0.66%)	1 (0.66%)
2019	109 (88%)	15 (12%)	0	0	0	0	0

https://www.cs.umd.edu/users/gasarch/BLOGPAPERS/pollpaper3.pdf

When Will P=NP be resolved?

	02–09	10–19	20–29	30–39	40–49	50–59	60–69	70–79
2002	5 (5%)	12 (12%)	13 (13%)	10 (10%)	5 (5%)	12 (12%)	4 (4%)	0 (0%)
2012	0 (0%)	2 (1%)	17 (11%)	18 (12%)	5 (3%)	10 (6.5%)	10 (6.5%)	9 (6%)
2019	0 (0%)	0 (0%)	26 (22%)	20 (17%)	14 (12%)	9 (7%)	7 (6%)	5 (4%)

	80–89	90–99	100–109	110–119	150–159	2200–3000	4000–4100
2002	1 (1%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	5 (5%)	0 (0%)
2012	4 (3%)	5 (3%)	2 (1.2%)	5 (3%)	2 (1.2%)	3 (2%)	3 (2%)
2019	0 (0%)	0 (0%)	1 (0.8%)	10 (12%)	10 (12%)	1 (0.8%)	11 (9%)

	Long Time	Never	Don't Know	Sooner than 2100	Later than 2100
2002	0 (0%)	5 (5%)	21 (21%)	62 (62%)	17 (17%)
2012	22 (14%)	5 (3%)	8 (5%)	81 (53%)	63 (41%)
2019	7 (6%)	11 (9%)	0 (0%)	84 (66%)	40 (34%)

Notable Statements on P vs NP

Scott Aaronson I believe $\mathrm{P} \neq \mathrm{NP}$ on basically the same grounds that I think I won't be devoured tomorrow by a 500-foot-tall robotic marmoset from Venus, despite my lack of proof in both cases.

Suggested rephrased question:

will humans manage to prove $\mathrm{P} \neq \mathrm{NP}$ *before they either kill themselves out or are transcended by superintelligent cyborgs? And if the latter, will the cyborgs be able to prove* $\mathrm{P} \neq \mathrm{NP}$*?*

Neil Immerman $\mathrm{P} \neq \mathrm{NP}$ will be resolved somewhere between 2017 and 2034, using some combination of logic, algebra, and combinatorics.

Donald Knuth: (Retired from Stanford) It will be solved by either 2048 or 4096. I am currently somewhat pessimistic. The outcome will be the truly worst case scenario: namely that someone will prove "P=NP because there are only finitely many obstructions to the opposite hypothesis"; hence there will exists a polynomial time solution to SAT but we will never know its complexity!