CS 3100
Data Structures and Algorithms 2

Lecture 23: Reductions

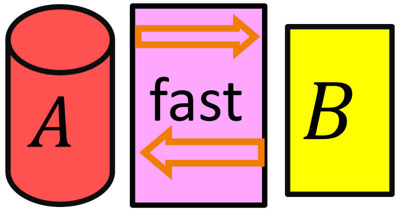
Co-instructors: Robbie Hott and Ray Pettit

Spring 2024

Readings from CLRS 4th Ed: Network flow etc. in Chapter 24
(Reductions covered in CLRS but in a context we're not studying in CS3100)

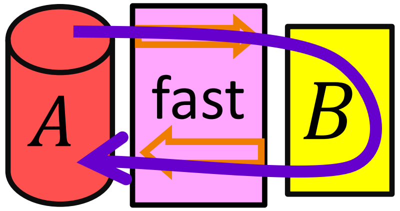
Two Ways to use Reductions

Suppose we have a "fast" reduction from **A** to **B**



1. A "fast" algorithm for **B** gives a fast algorithm for **A**

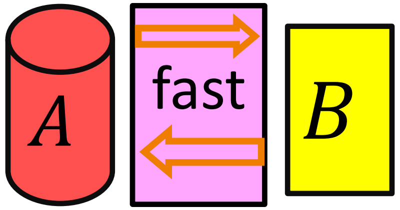
Then **A** is fast



If **B** is fast

2. If we have a worst-case lower bound for **A**, we also have one for **B**

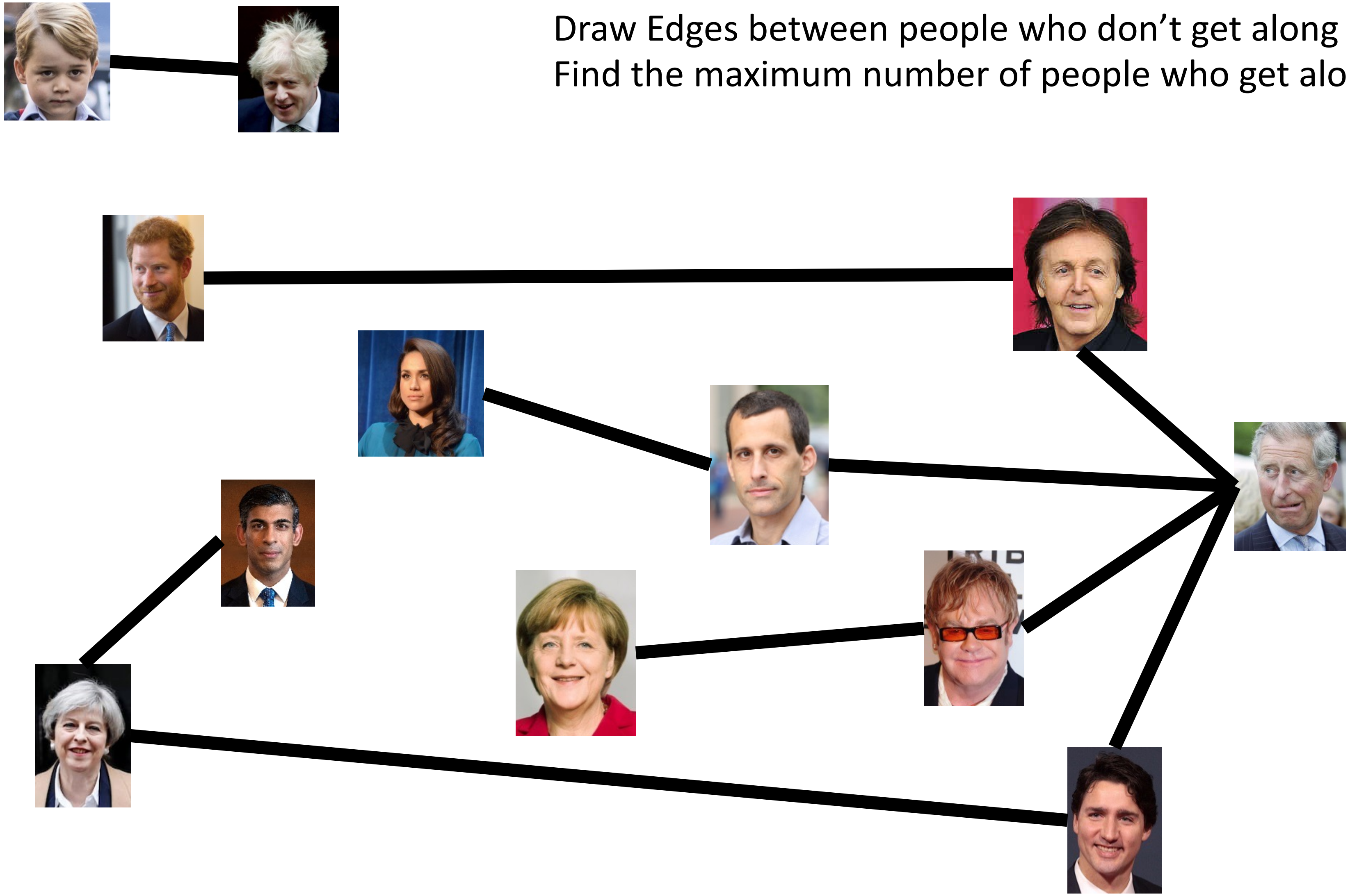
If **A** is slow



Then **B** is slow

Party Problem

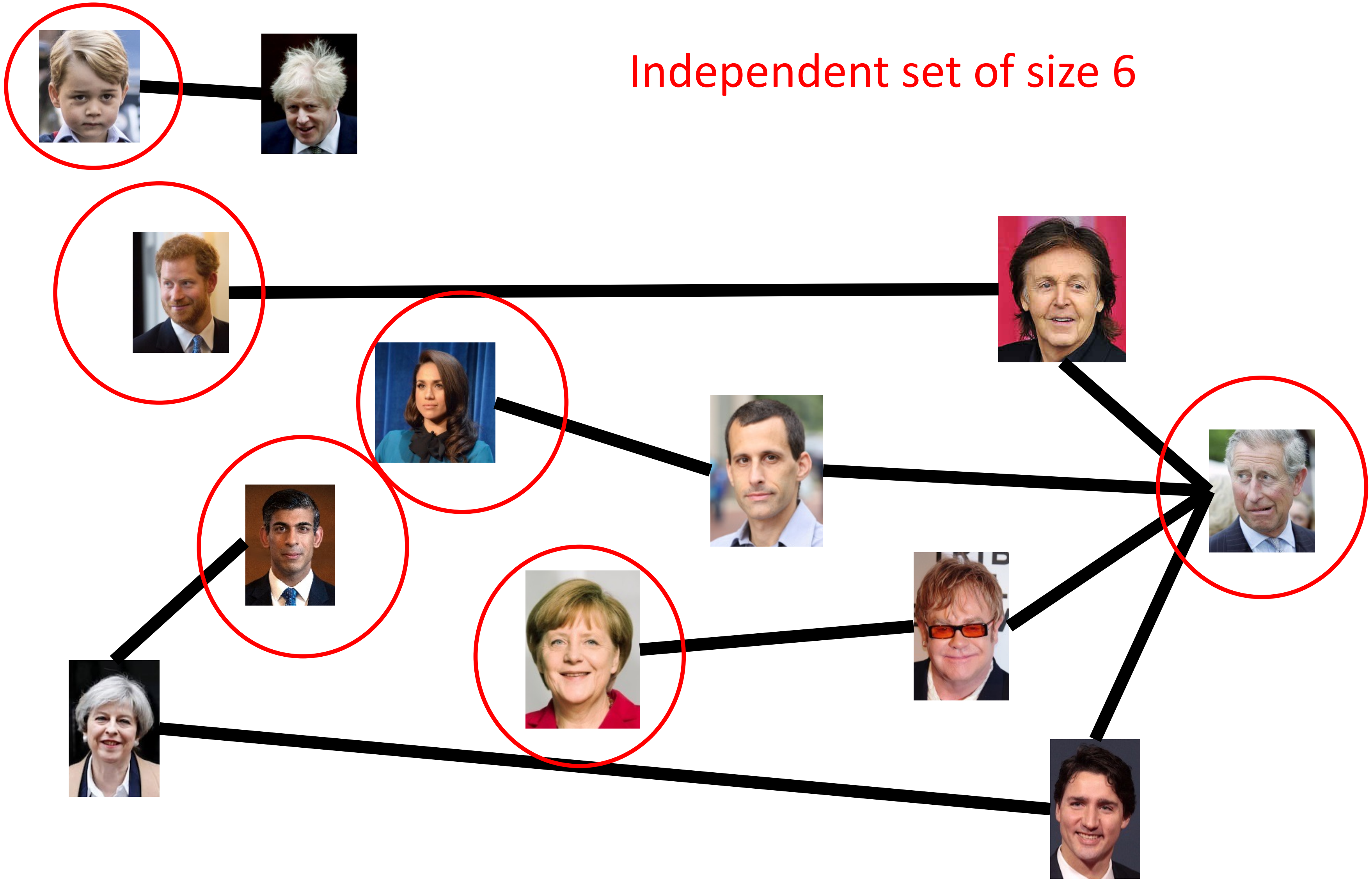
Draw Edges between people who don't get along
Find the maximum number of people who get along



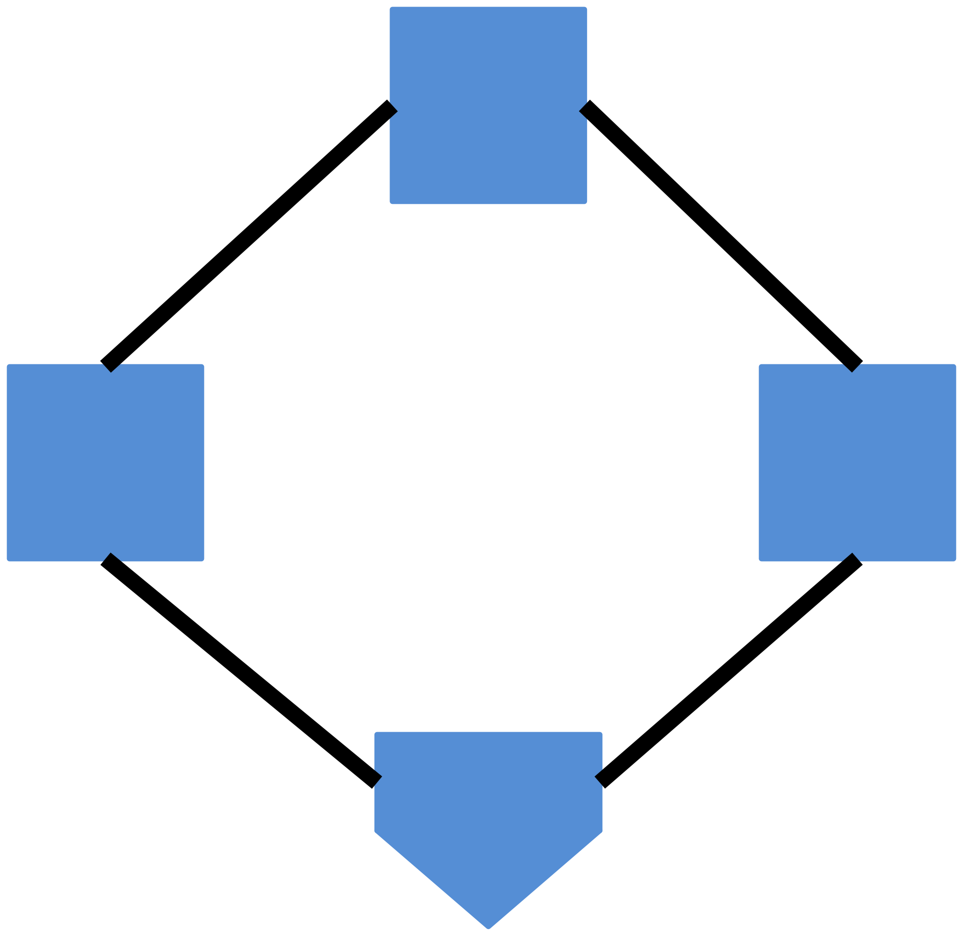
Maximum Independent Set

- Independent set: $S \subseteq V$ is an independent set if no two nodes in S share an edge
- Maximum Independent Set Problem: Given a graph $G = (V, E)$ find the maximum independent set S

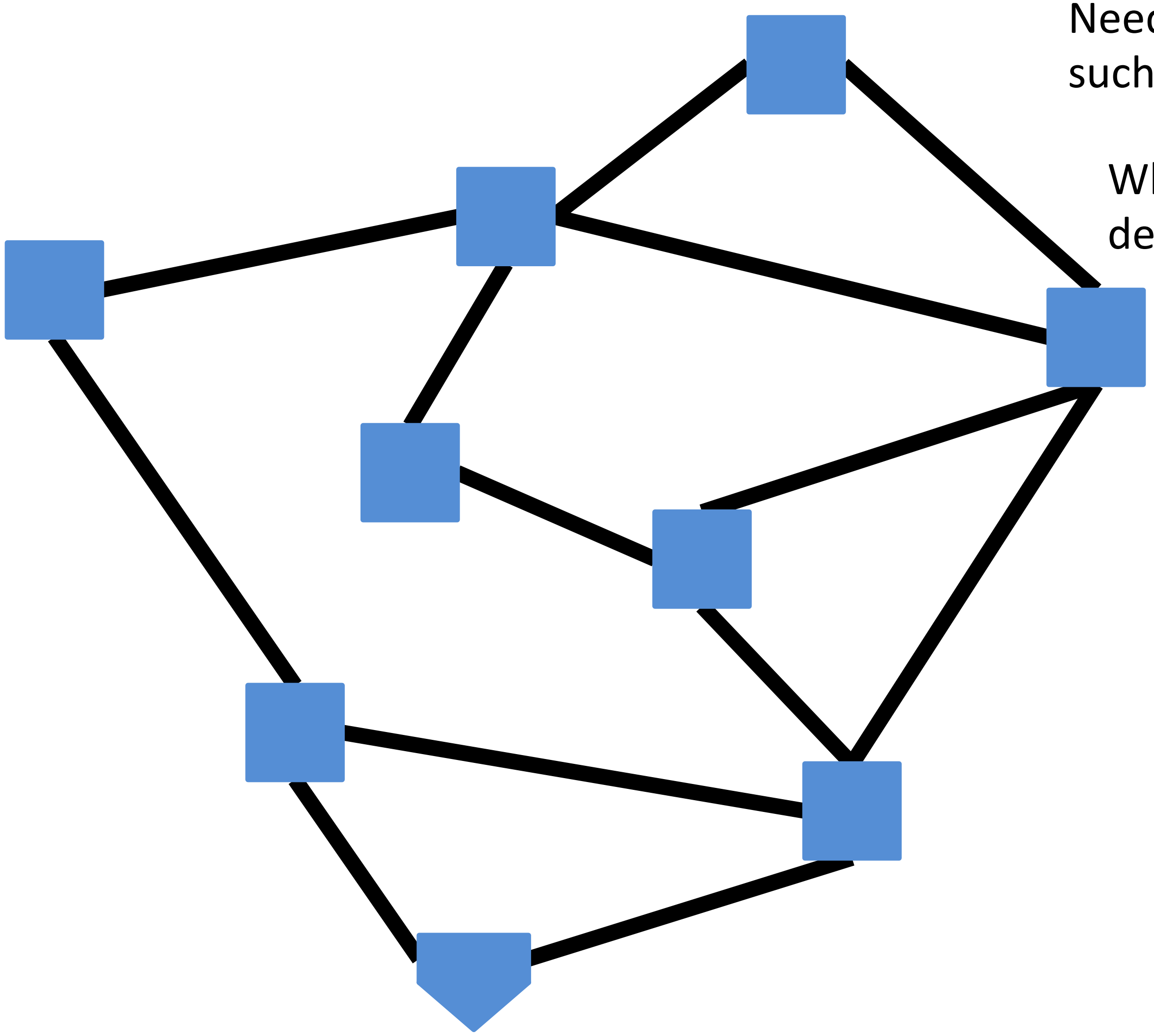
Example



Generalized Baseball



Generalized Baseball



Need to place defenders on bases such that every edge is defended

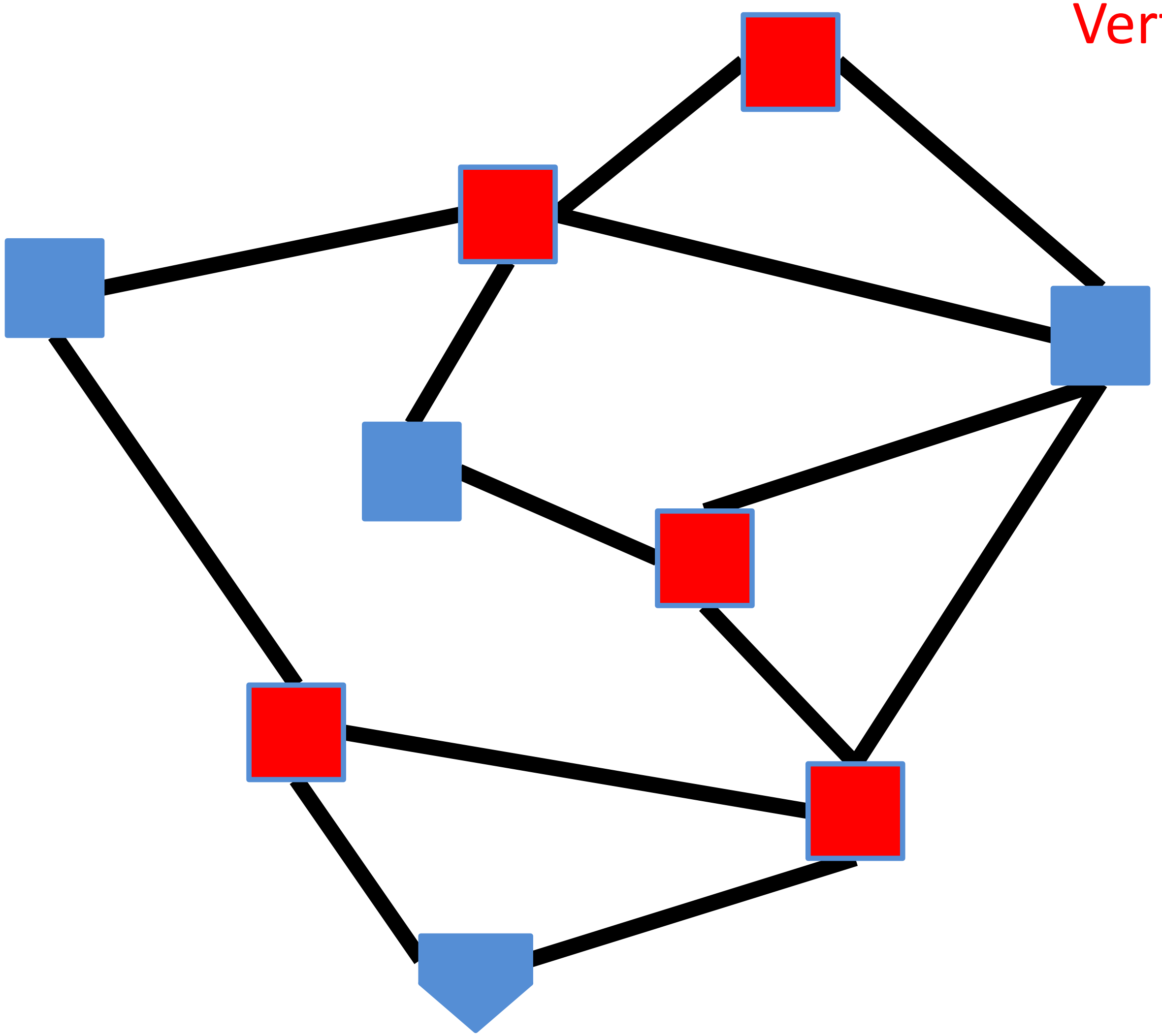
What's the fewest number of defenders needed?

Minimum Vertex Cover

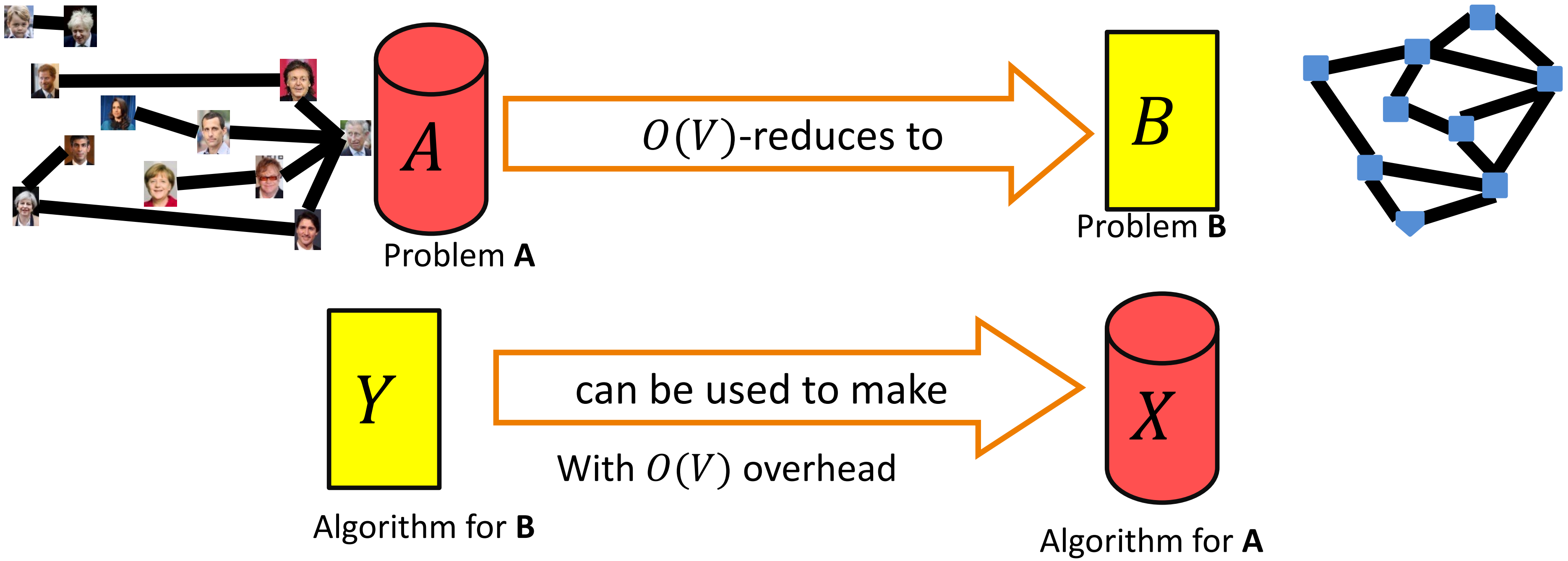
- Vertex Cover: $C \subseteq V$ is a vertex cover if every edge in E has one of its endpoints in C
- Minimum Vertex Cover: Given a graph $G = (V, E)$ find the minimum vertex cover C

Example

Vertex cover of size 5



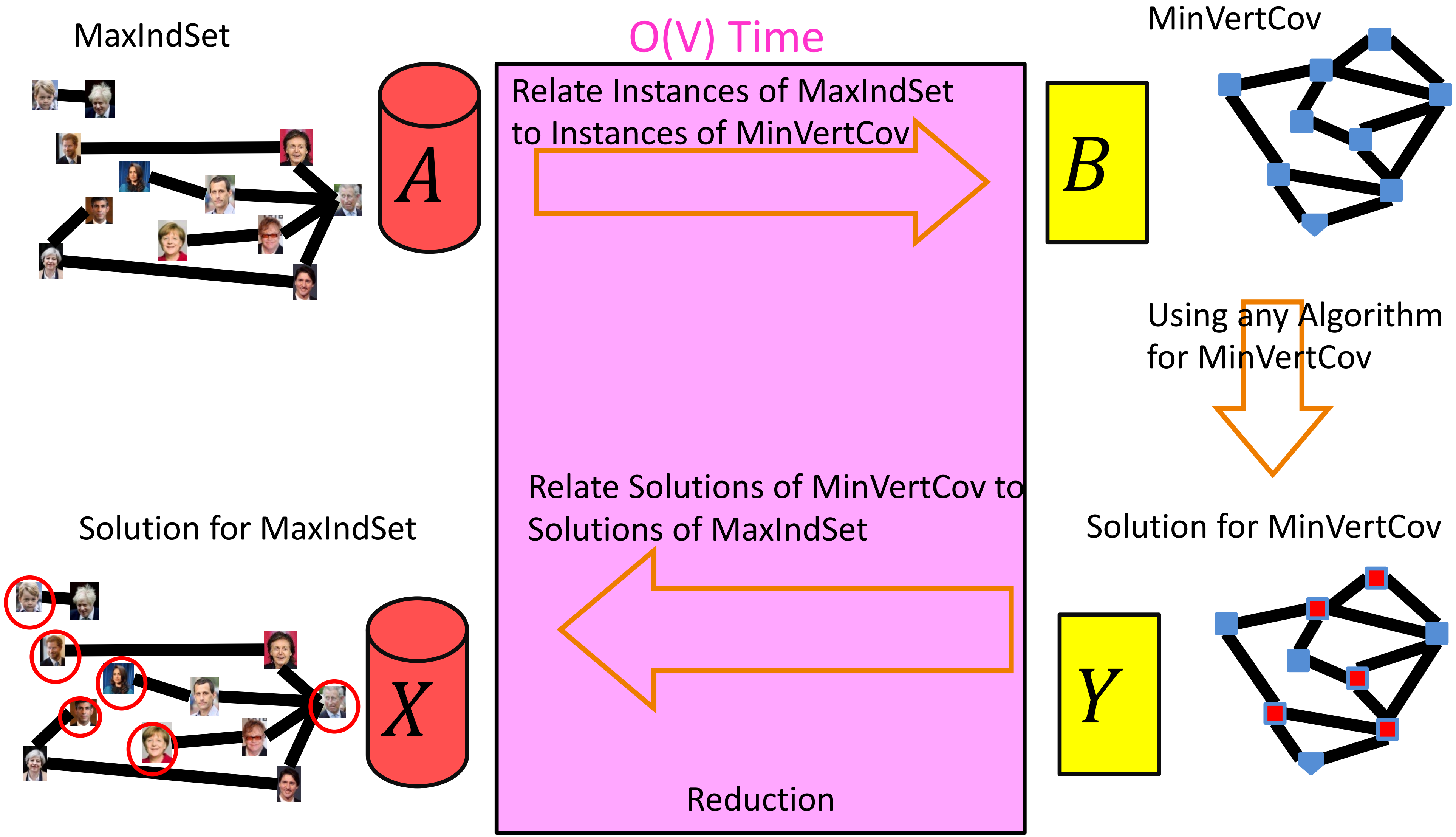
MaxIndSet$\leq_V$MinVertCov



If A requires time $\Omega(f(n))$ time then B also requires $\Omega(f(n))$ time

$$A \leq_V B$$

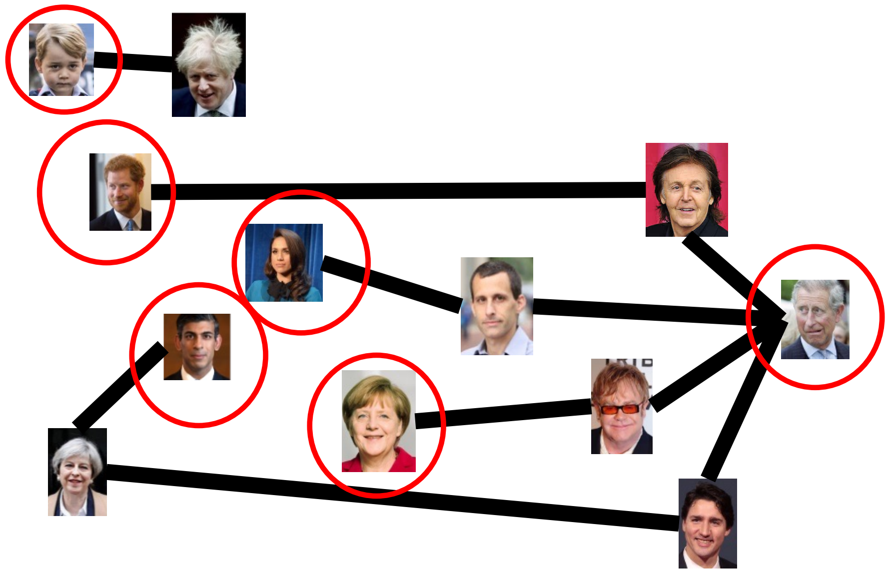
We need to build this Reduction



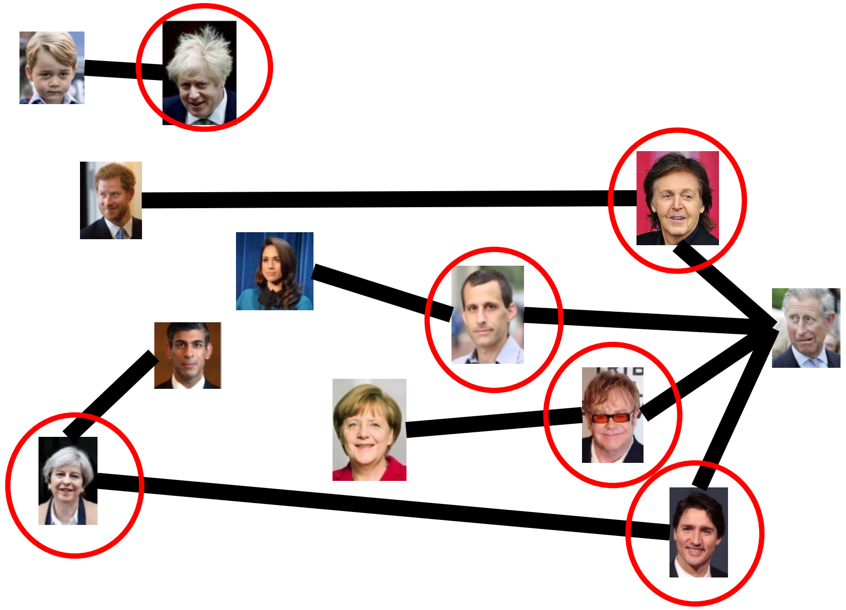
Reduction Idea

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

Independent Set



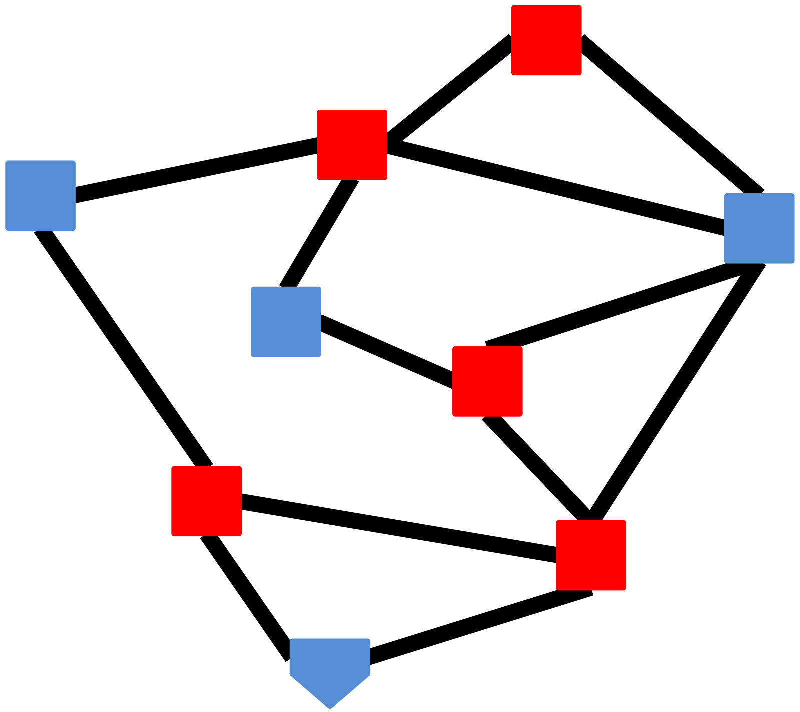
Vertex Cover



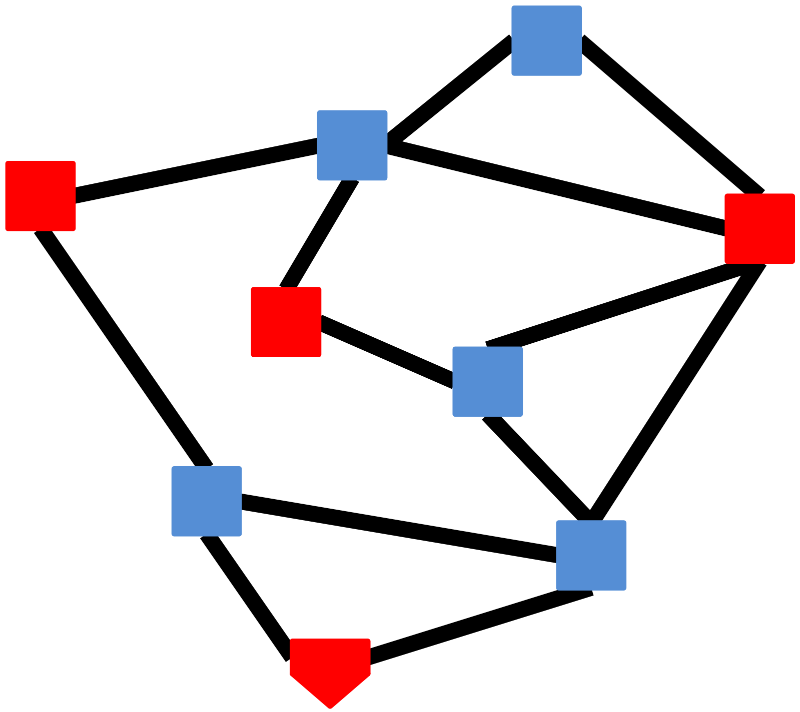
Reduction Idea

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

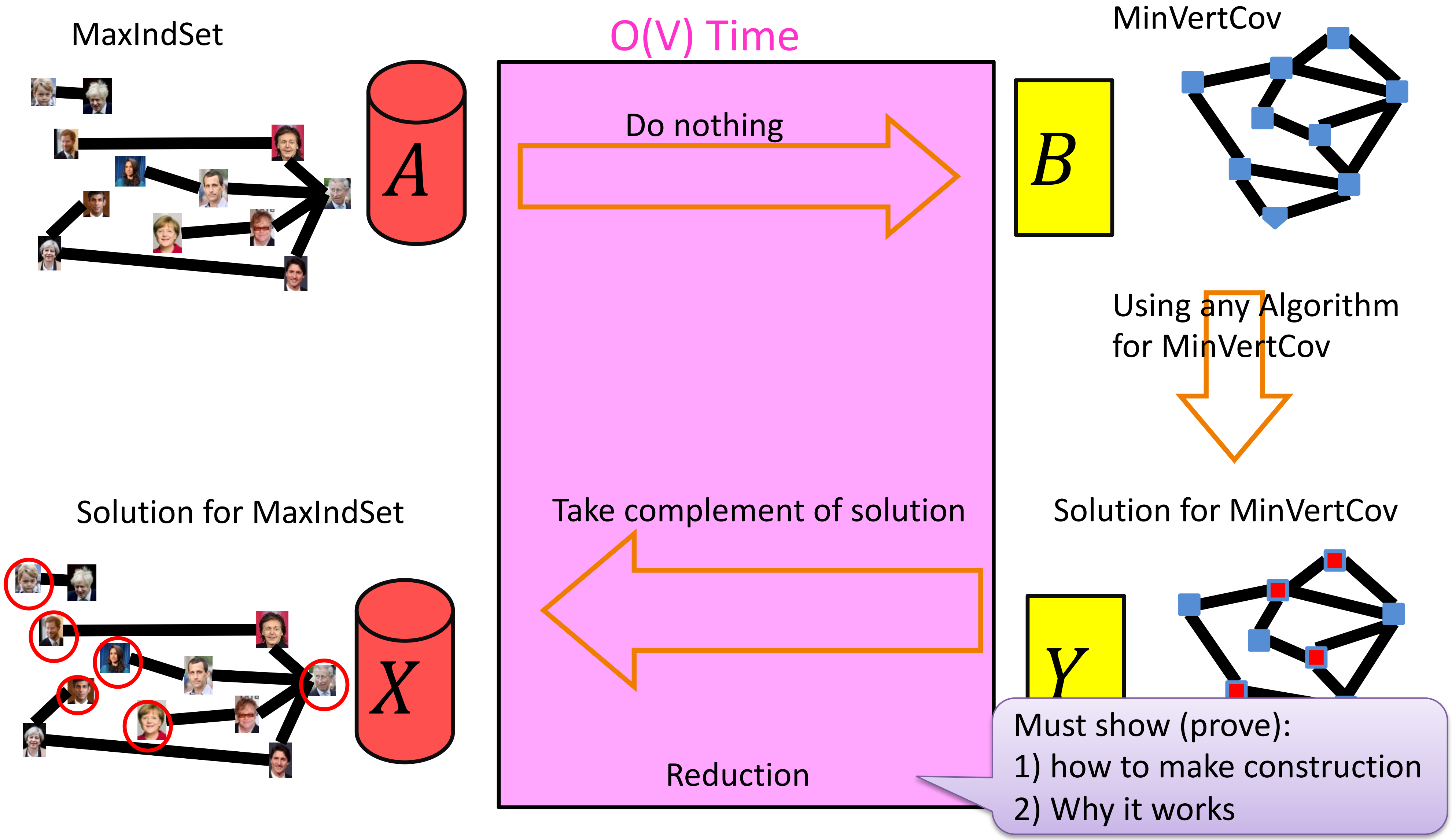
Vertex Cover



Independent Set



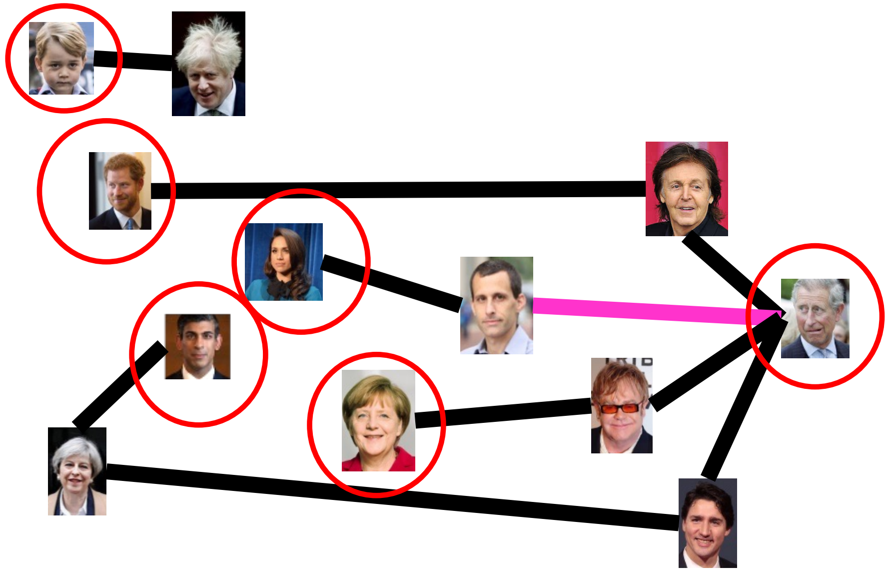
MaxVertCov V-Time Reducible to MinIndSet



Proof: ⇒

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

Let S be an independent set



Consider any edge $(x, y) \in E$

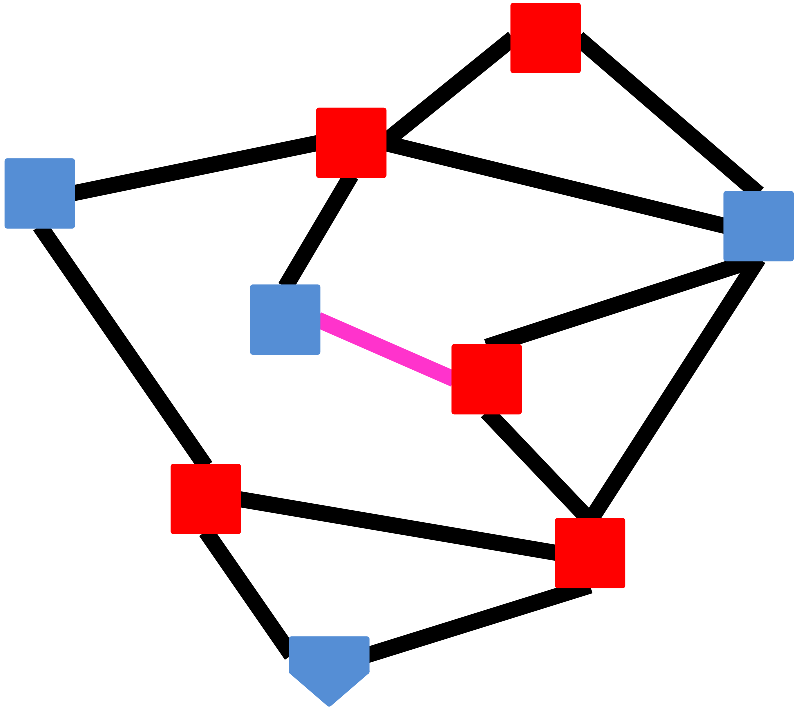
If $x \in S$ then $y \notin S$, because other wise S would not be an independent set

Therefore $y \in V - S$, so edge (x, y) is covered by $V - S$

Proof: $\Leftarrow$

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

Let $\mathrm{V} - S$ be a vertex cover



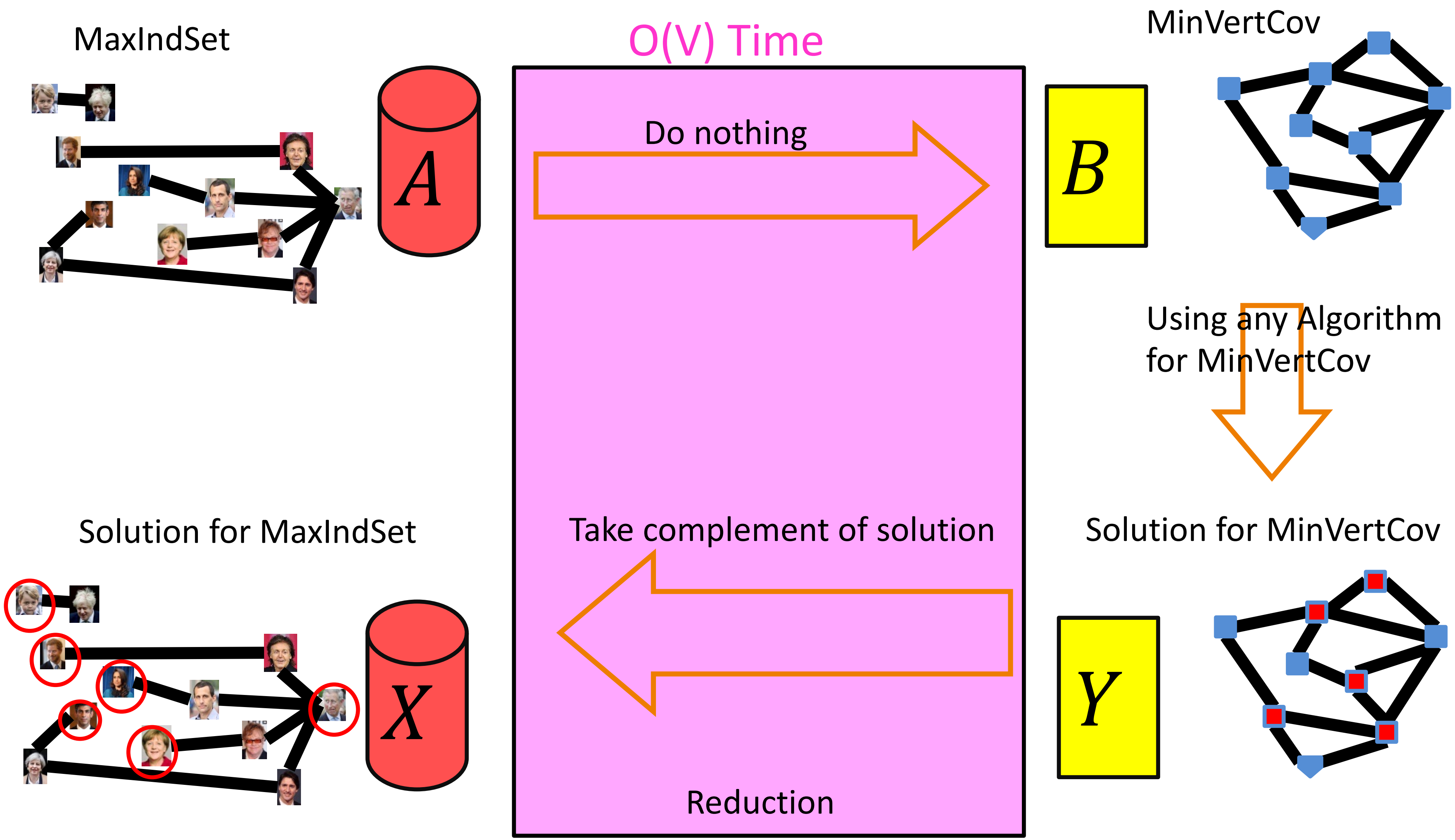
Consider any edge $(x, y) \in E$

At least one of x and y belong to $V - S$, because $\mathrm{V} - S$ is a vertex cover

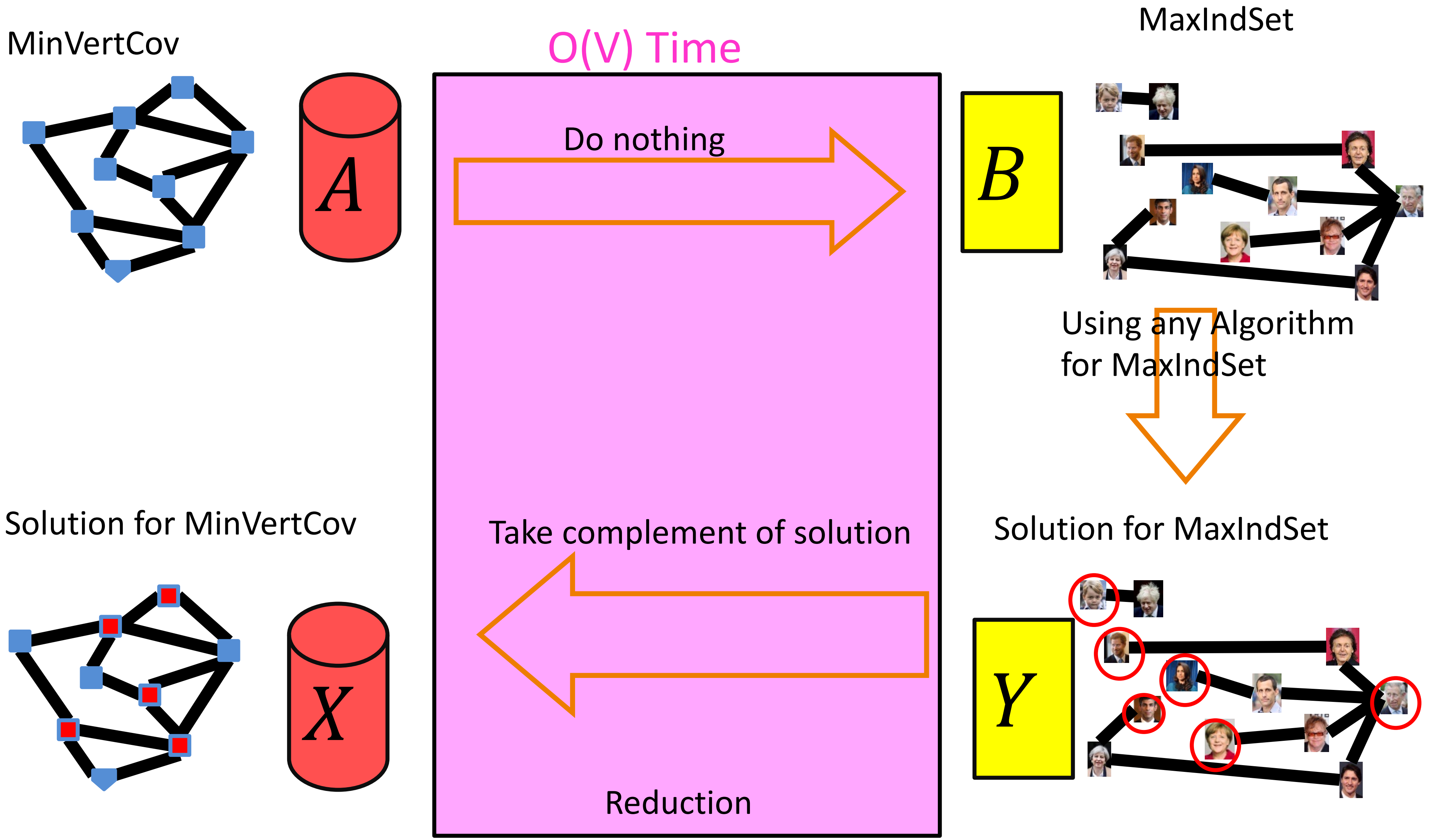
Therefore x and y are not both in S,

No edge has both end-nodes in S, thus S is an independent set

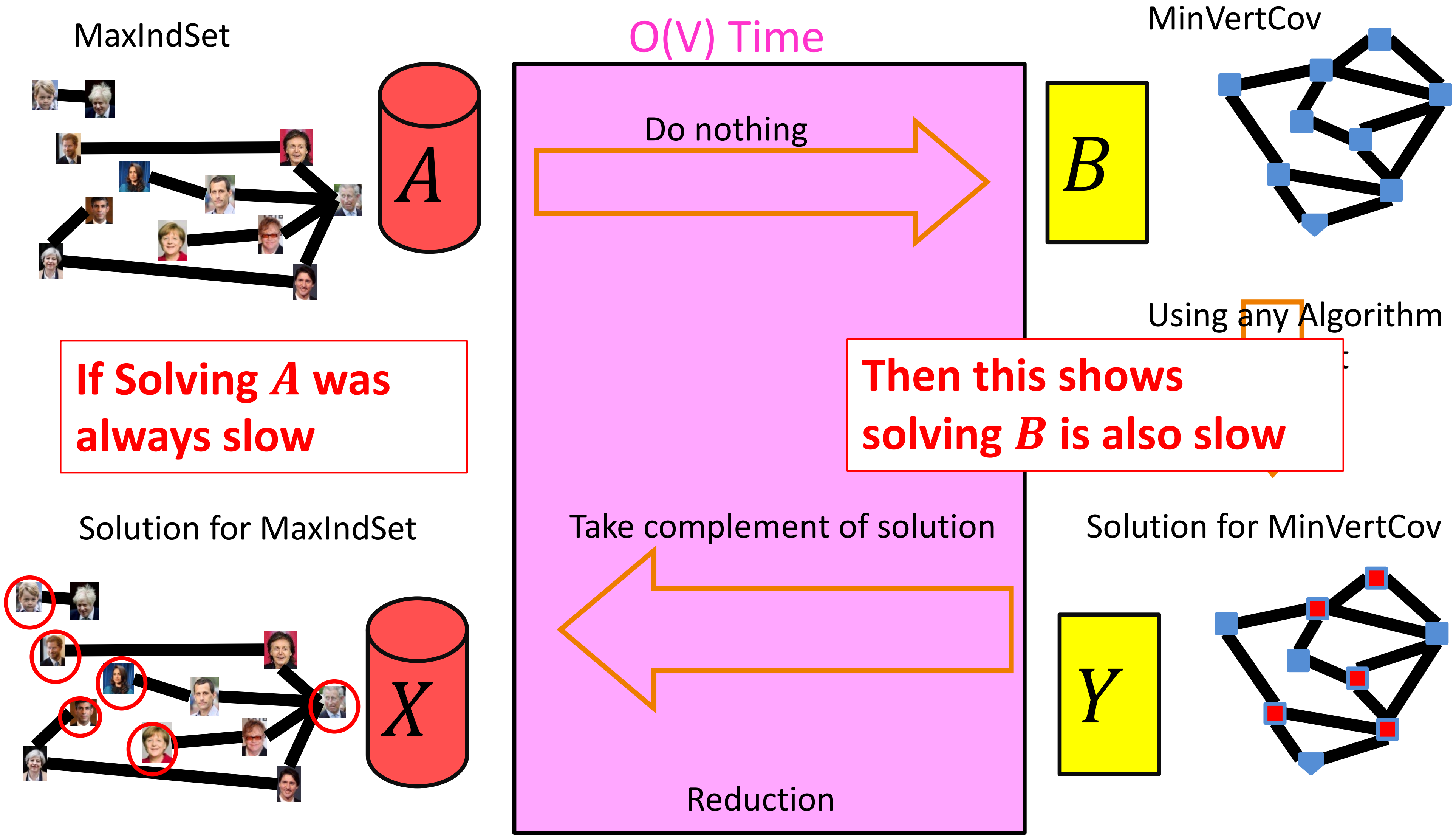
MaxIndSet V-Time Reducible to MinVertCov



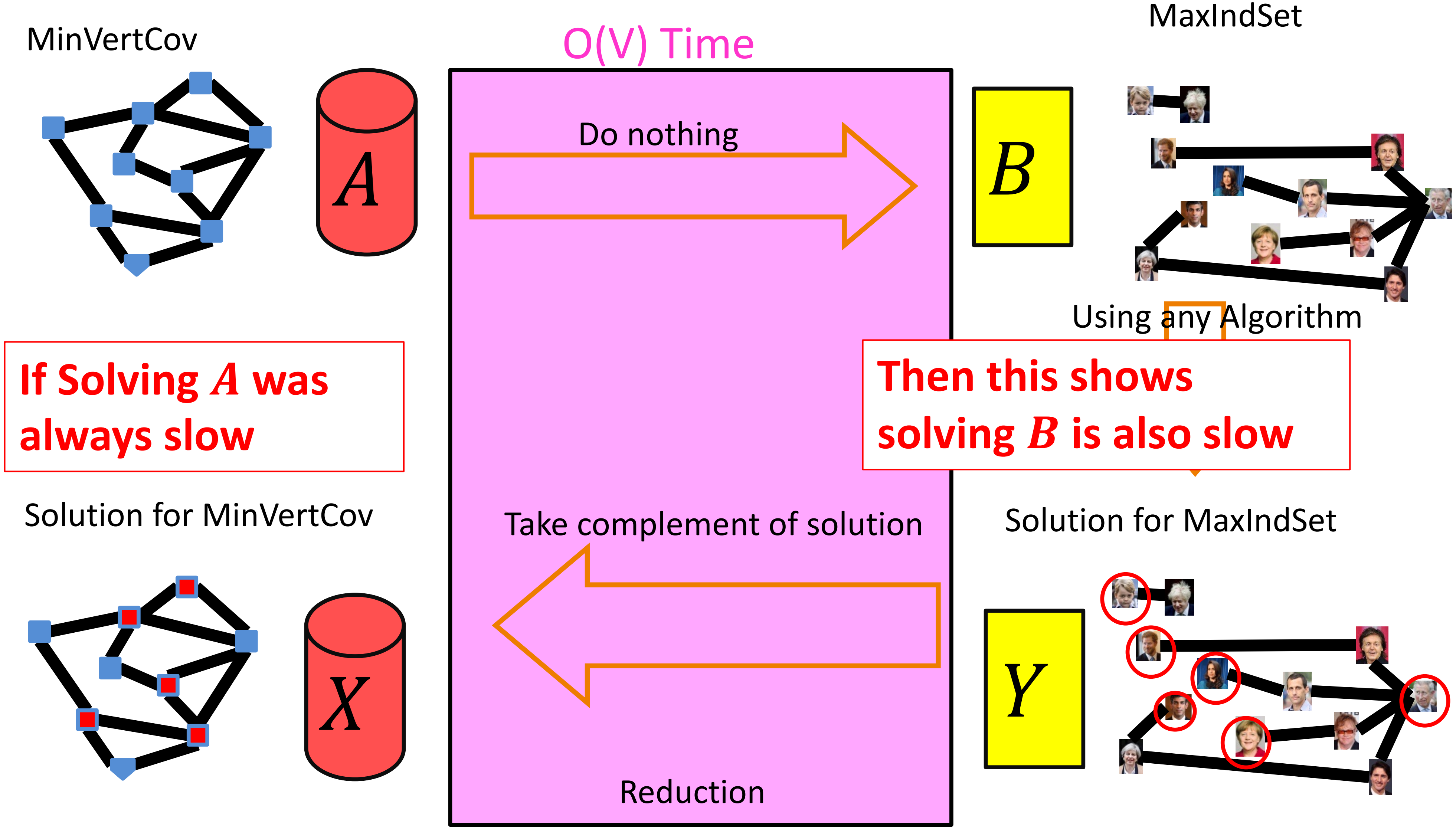
MinVertCov V-Time Reducible to MaxIndSet



Corollary


MaxIndSet
O(V) Time
MinVertCov
Do nothing
A
B
Using any Algorithm
If Solving A was always slow
Then this shows solving B is also slow
Solution for MaxIndSet
Take complement of solution
Solution for MinVertCov
X
Y
Reduction

Corollary


MinVertCov
O(V) Time
MaxIndSet
Do nothing
A
B
Using any Algorithm
If Solving A was always slow
Then this shows solving B is also slow
Solution for MinVertCov
Take complement of solution
Solution for MaxIndSet
X
Y
Reduction

Conclusion

- MaxIndSet and MinVertCov are either both fast, or both slow
  - Spoiler alert: We don't know which!
    - (But we think they're both slow)
  - Both problems are NP-Complete

NP Complete Problems: A Short Overview

Problems and Exponential Solutions

- We've **not** been able to find polynomial-time algorithms for some problems
- Example graph problems
  - *Travelling Salesperson Problem.* (Similar problem: does a graph have a *Hamilton Cycle*?)
  - Can a *graph be colored* with k colors (for k>2)?
  - *Max Independent Set* and *Min Vertex Cover*
- Other examples
  - *Satisfiability:* Given a Boolean logic expression like:
    $(x_1 \lor \neg x_2) \land (\neg x_1 \lor x_3 \lor x_4) \land (\neg x_5)$
    can we assign True or False to the Boolean variables so this is true?

https://en.wikipedia.org/wiki/List_of_NP-complete_problems

Exponential? It's an Open Question

- Perhaps we just haven't found a polynomial algorithm yet
  - Option 1: Keep trying!
  - Option 2: Prove an exponential lower bound for a problem, thus showing it's impossible for the problem to have a polynomial solution
- For the problems just listed, no one's made such a lower bound proof either ☹

- **Bottom-line:** We don't know if these problems are exponential or not!

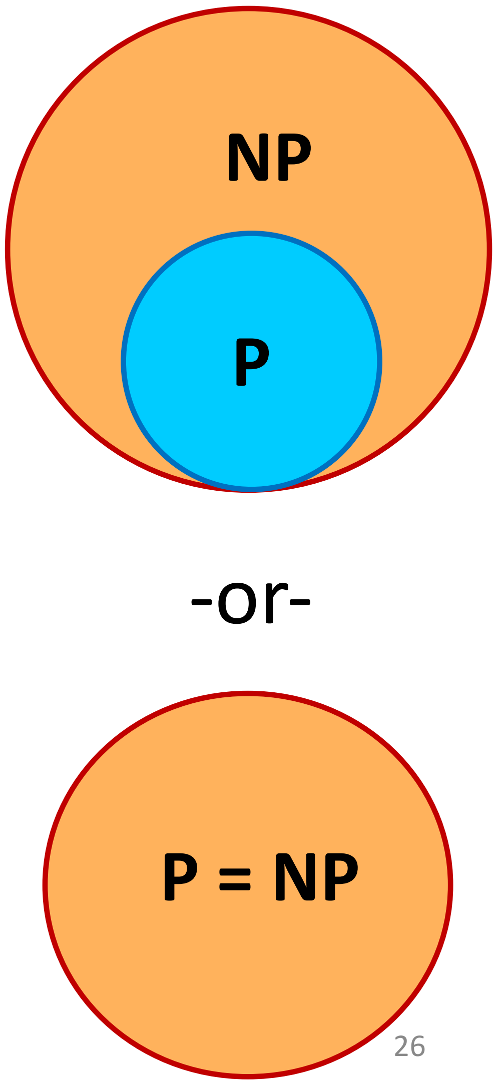
Before we go further on this topic….

- This is a complex (and interesting!) topic in CS theory
- In these slides, we will approach things from a simpler viewpoint than you'd get in a CS theory course (like CS 3120)

- For example, the math and theory related to this starts with ***decision problems***
  - But we're not going to say much about that, and instead we'll be less formal than perhaps we should be

Classes of Problems: P vs NP

- P
  - P is the set of problems solvable in polynomial time
    - Find a solution in $O(n^c)$ for some number c
- NP
  - Non-Deterministic Polynomial Time
  - NP is the set of problems ***verifiable*** in polynomial time
    - Verify a proposed solution (not find one) in $O(n^c)$ for some number c
- Open Problem: Does P=NP?
  - Certainly $P \subseteq NP$



NP-Complete Problems

- Computer scientists have identified a group of problems we call **NP-Complete**
- A problem A $\in$ NP-Complete if:
  - A $\in$ NP (a solution can be verified in polynomial time)
  - Any other problem in NP can be reduced to A in polynomial time, i.e. for any B $\in$ NP, B $\leq_p$ A
- Some consequences
  - There must be a reduction between any two NP-Complete problems
  - If one NP-Complete problem has a polynomial solution, all problems in NP-Complete and NP are polynomial, and thus P = NP
  - If one NP-Complete problem has an exponential lower bound, they all do and $P \neq NP$

Take-Aways for CS 3100

- What an NP-Complete Problem is
  - Informally: a group of problems that are “equivalent” in that they’re either all polynomial or all exponential, and we don’t know which
- The big open question in CS: Does P=NP?
  - Given a problem, if we can verify a solution is polynomial time, does this mean we can always solve it directly in polynomial time?
- Some problems we’ve studied are NP-Complete problems
  - Max Independent Set, Min Vertex Cover, dynamic programming problems that are pseudo-polynomial

More about this topic in CS3120!
https://en.wikipedia.org/wiki/NP-completeness