CS4102 Algorithms
Spring 2019

No time for a warm-up today!

Today's Keywords

- Reductions
- Bipartite Matching
- Vertex Cover
- Independent Set
- NP-Completeness

2

CLRS Readings

- Chapter 34

3

Homeworks, etc

- HW9 due ~~Monday 4/29~~ Tuesday 4/30 at 11pm
  - Written (use LaTeX)
  - Reductions
- Final Exam: Saturday, May 4, 2-5pm
  - Heavily from material since midterm
    - May ask for runtime of an algorithm, some knowledge of D&C
    - Won't directly ask you to solve recurrences
  - Practice final online by tomorrow
  - Review session?

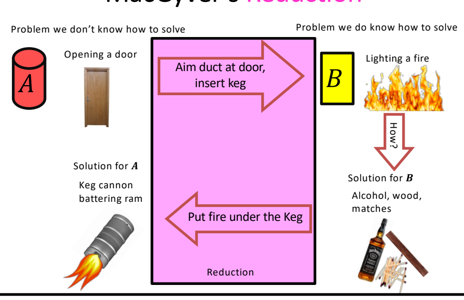
4

Reductions

- Algorithm technique of supreme ultimate power
- Convert instance of problem A to an instance of Problem B
- Convert solution of problem B back to a solution of problem A

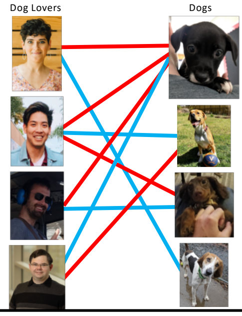
5

MacGyver's Reduction



6

Maximum Bipartite Matching

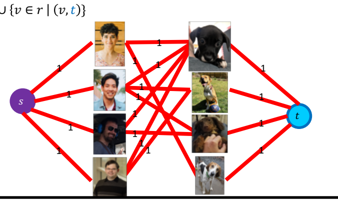


7

Maximum Bipartite Matching Using Max Flow

Make $G = (L, R, E)$ a flow network $G' = (V', E')$ by:

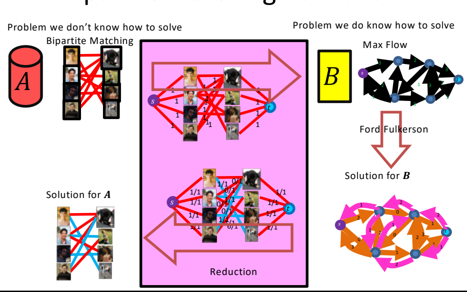
- Adding in a source and sink to the set of nodes:
  - $V' = L \cup R \cup \{s, t\}$
- Adding an edge from source to L and from R to sink:
  - $E' = E \cup \{u \in L \mid (s,u)\} \cup \{v \in r \mid (v,t)\}$
- Make each edge capacity 1:
  - $\forall e \in E', c(e) = 1$



Remember: need to show
1. How to map instance of MBM to MF (and back) - construction
2. A valid solution to MF instance is a valid solution to MBM instance

8

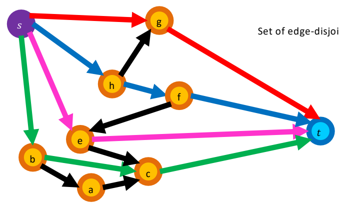
Bipartite Matching Reduction



9

Edge-Disjoint Paths

Given a graph $G = (V, E)$, a start node s and a destination node t, give the maximum number of paths from s to t which share no edges

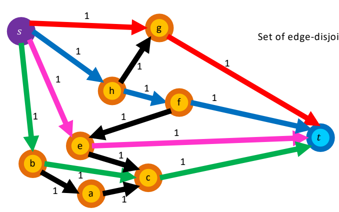


Set of edge-disjoint paths of size 4

10

Edge-Disjoint Paths Algorithm

Make s and t the source and sink, give each edge capacity 1, find the max flow.

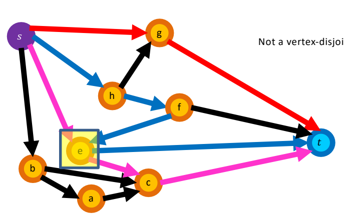


Set of edge-disjoint paths of size 4

11

Vertex-Disjoint Paths

Given a graph $G = (V, E)$, a start node s and a destination node t, give the maximum number of paths from s to t which share no vertices



Not a vertex-disjoint path!

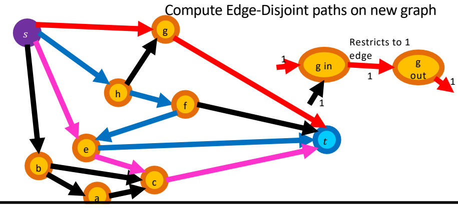
12

Vertex-Disjoint Paths Algorithm

Idea: Convert an instance of the vertex-disjoint paths problem into an instance of edge-disjoint paths

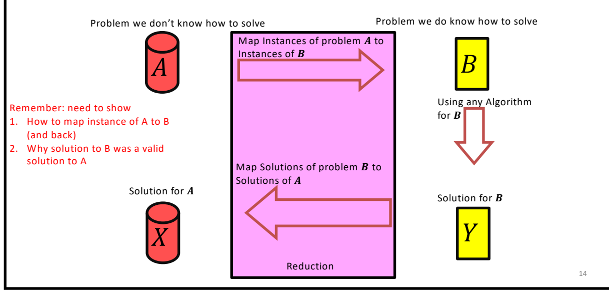
Make two copies of each node, one connected to incoming edges, the other to outgoing edges

Compute Edge-Disjoint paths on new graph



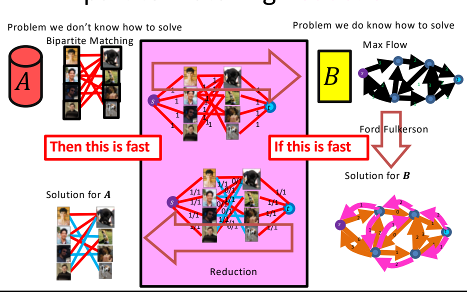
13

In General: Reduction

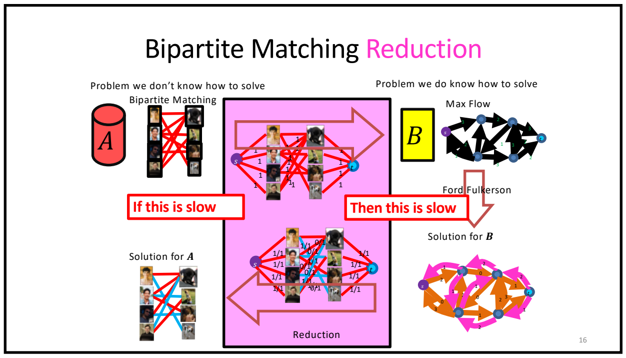


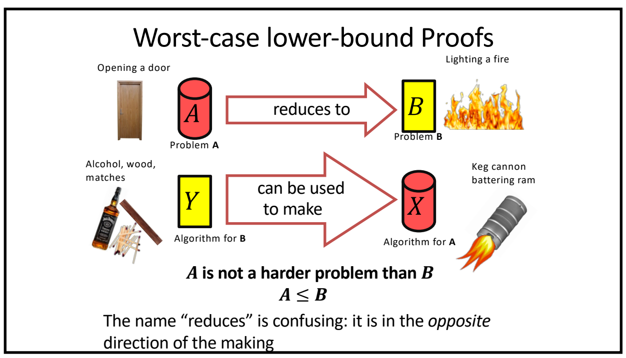
14

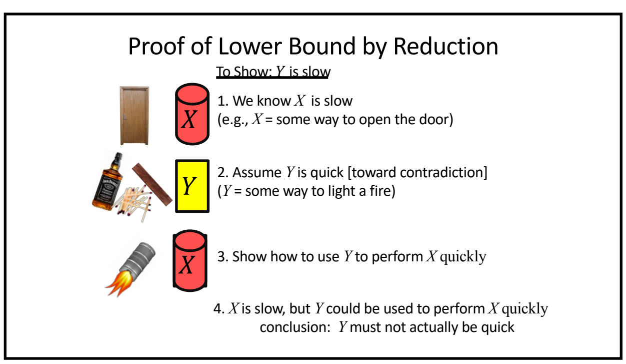
Bipartite Matching Reduction



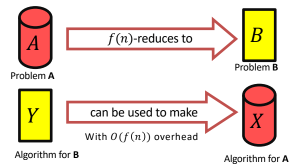
15


Bipartite Matching Reduction
Problem we don't know how to solve
Bipartite Matching
Problem we do know how to solve
Max Flow
A
B
If this is slow
Then this is slow
Ford Fulkerson
Solution for B
Solution for A
Reduction
16


Worst-case lower-bound Proofs
Opening a door
Lighting a fire
A
reduces to
B
Problem A
Problem B
Alcohol, wood, matches
Keg cannon battering ram
Y
can be used to make
X
Algorithm for B
Algorithm for A
A is not a harder problem than B
A ≤ B
The name "reduces" is confusing: it is in the opposite direction of the making


Proof of Lower Bound by Reduction
To Show: Y is slow
X
1. We know X is slow
(e.g., X = some way to open the door)
Y
2. Assume Y is quick [toward contradiction]
(Y = some way to light a fire)
X
3. Show how to use Y to perform X quickly
4. X is slow, but Y could be used to perform X quickly
conclusion: Y must not actually be quick

Reduction Proof Notation



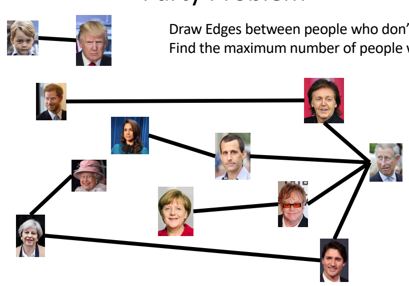
A is not a harder problem than B
$A \leq B$

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

19

Party Problem

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

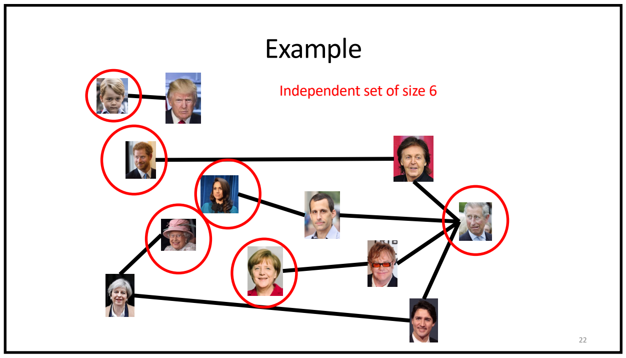


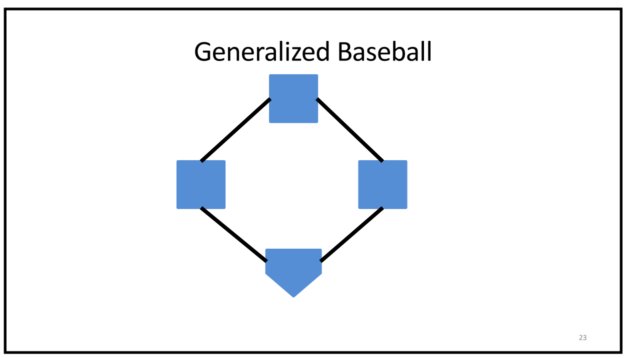
20

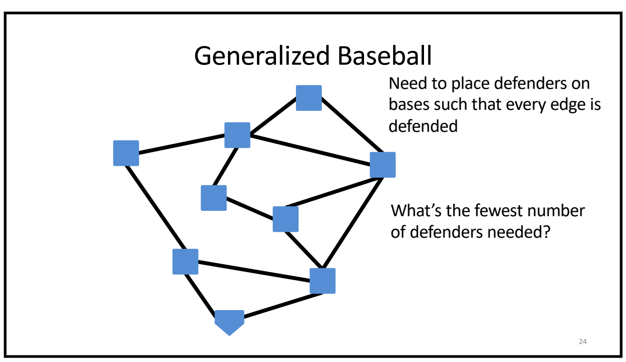
Maximum Independent Set

- 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

21


Example
Independent set of size 6
22

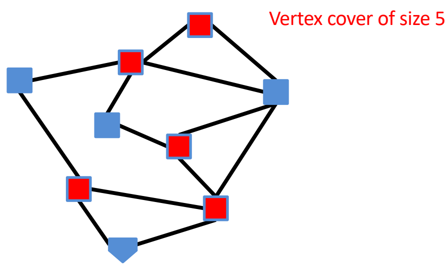

Generalized Baseball
23


Generalized Baseball
Need to place defenders on bases such that every edge is defended
What's the fewest number of defenders needed?
24

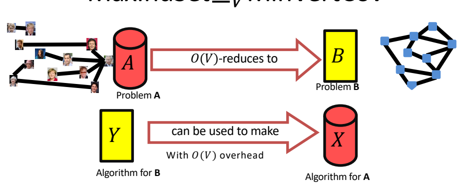
Minimum Vertex Cover

- 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

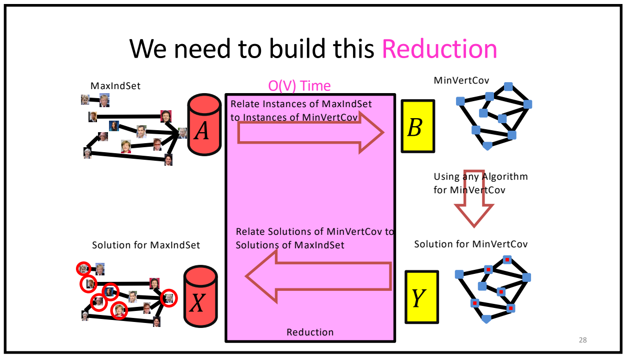


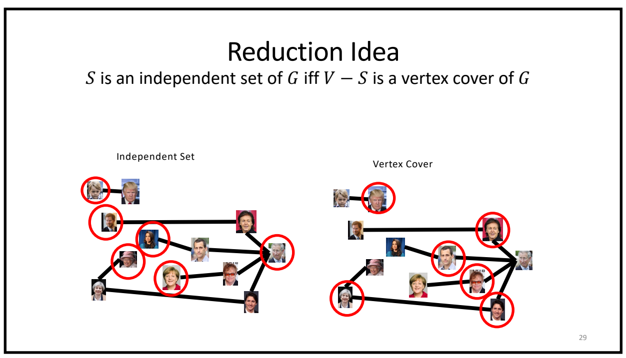
MaxIndSet$\le_V$MinVertCov

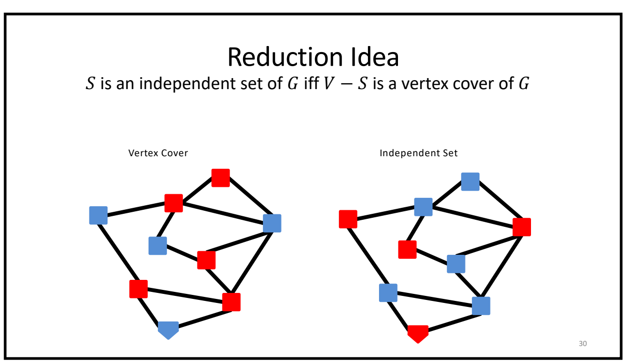


If A requires time $\Omega(f(n))$ time then B also requires $\Omega(f(n))$ time

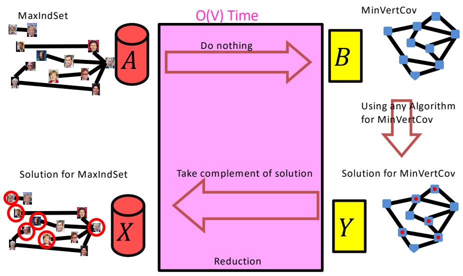
$A \le_V B$


We need to build this Reduction
MaxIndSet
O(V) Time
Relate Instances of MaxIndSet to Instances of MinVertCov
MinVertCov
A
B
Using any Algorithm for MinVertCov
Solution for MaxIndSet
Relate Solutions of MinVertCov to Solutions of MaxIndSet
Solution for MinVertCov
X
Y
Reduction
28


Reduction Idea
S is an independent set of G iff $V - S$ is a vertex cover of G
Independent Set
Vertex Cover
29


Reduction Idea
S is an independent set of G iff $V - S$ is a vertex cover of G
Vertex Cover
Independent Set
30

MaxVertCov V-Time Reducible to MinIndSet

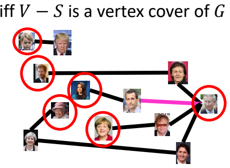


31

Proof: $\Rightarrow$

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 otherwise S would not be an independent set

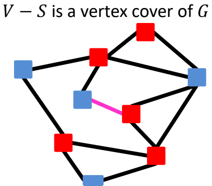
Therefore $y \in V - S$, so edge (x, y) is covered by $V - S$

32

Proof: $\Leftarrow$

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

Let $V - S$ be a vertex cover

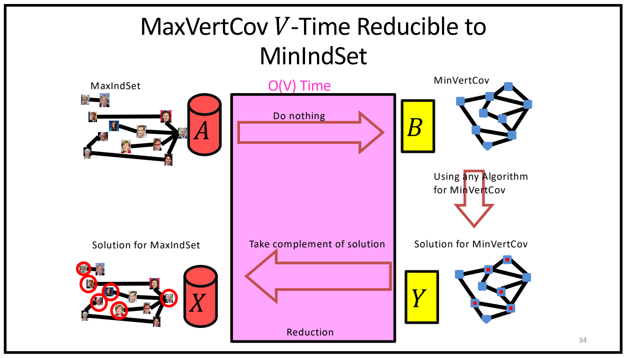


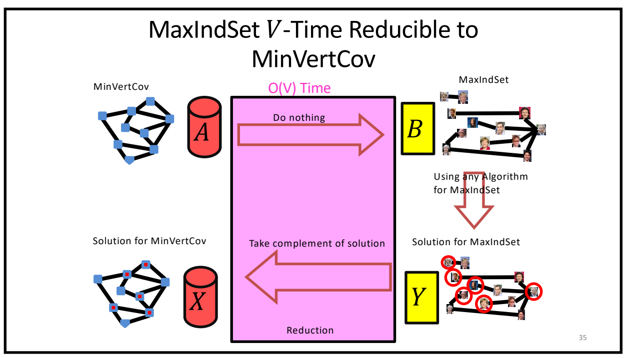
Consider any edge $(x, y) \in E$

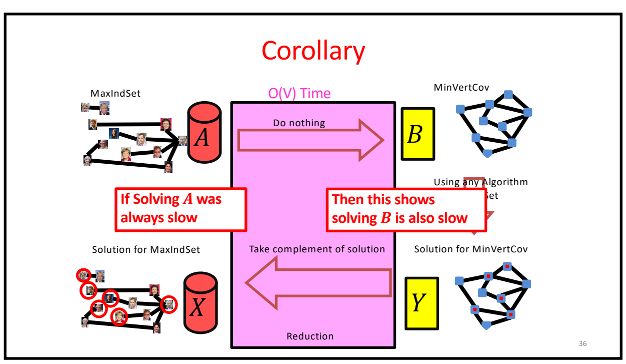
At least one of x and y belong to $V - S$, because $V - S$ is a vertex cover

Therefore x and y are not both in S,
No edge has both end-nodes in S, thus S is an independent set

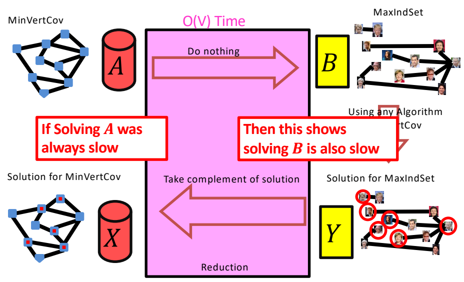
33


MaxVertCov V-Time Reducible to
MinIndSet
MaxIndSet
O(V) Time
MinVertCov
Do nothing
A
B
Using any Algorithm
for MinVertCov
Solution for MaxIndSet
Take complement of solution
Solution for MinVertCov
X
Y
Reduction
34


MaxIndSet V-Time Reducible to
MinVertCov
MinVertCov
O(V) Time
MaxIndSet
Do nothing
A
B
Using any Algorithm
for MaxIndSet
Solution for MinVertCov
Take complement of solution
Solution for MaxIndSet
X
Y
Reduction
35


Corollary
MaxIndSet
O(V) Time
MinVertCov
Do nothing
A
B
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
36

Corollary



37

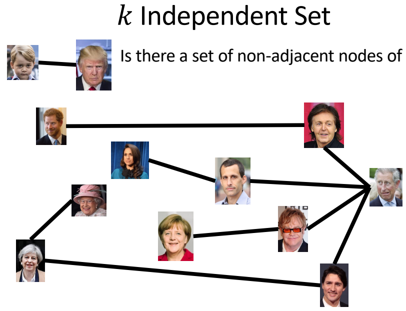
Conclusion

- 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

38

k Independent Set

Is there a set of non-adjacent nodes of size k?



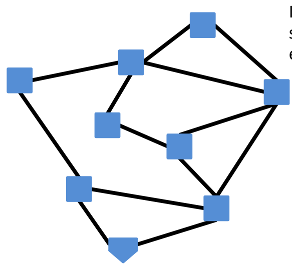
41

k Independent Set

- Independent set: $S \subseteq V$ is an independent set if no two nodes in S share an edge
- k Independent Set Problem: Given a graph $G = (V, E)$ and a number k, **determine whether there is an independent set S of size k**

43

k Vertex Cover



Is there a set of nodes of size k which covers every edge?

45

k Vertex Cover

- Vertex Cover: $C \subseteq V$ is a vertex cover if every edge in E has one of its endpoints in C
- k Vertex Cover: Given a graph $G = (V, E)$ and a number k, **determine whether there is a vertex cover C of size k**

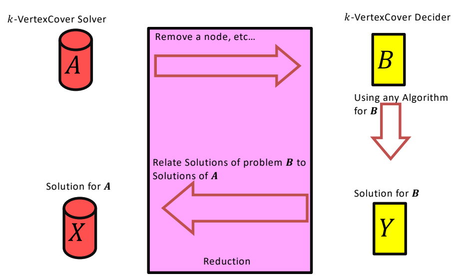
47

Problem Types

- Decision Problems: If we can solve this
  - Is there a solution?
    - Output is True/False
  - Is there a vertex cover of size k?
- Search Problems: Then we can solve this
  - Find a solution
    - Output is complex
  - Give a vertex cover of size k
- Verification Problems:
  - Given a potential solution, is it valid?
    - Output is True/False
  - Is **this** a vertex cover of size k**?**

48

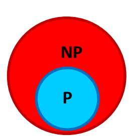
Reduction



54

P vs NP

- P
  - Deterministic Polynomial Time
  - Problems solvable in polynomial time
    - $O(n^p)$ for some number p
- NP
  - Non-Deterministic Polynomial Time
  - Problems verifiable in polynomial time
    - $O(n^p)$ for some number p
- Open Problem: Does P=NP?
  - Certainly $P \subseteq NP$



55

k-Independent Set is NP

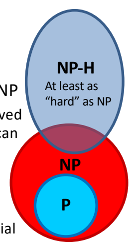
- To show: Given a potential solution, can we **verify** it in $O(n^p)$? $[n = V + E]$

How can we verify it?

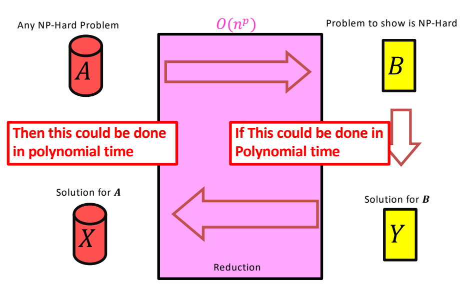
1. Check that it's of size k $O(V)$
2. Check that it's an independent set $O(V^2)$

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 if $\forall A \in NP, A \leq_p B$
  - $A \leq_p B$ means A reduces to B in polynomial time

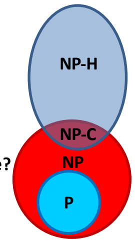


NP-Hardness Reduction



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



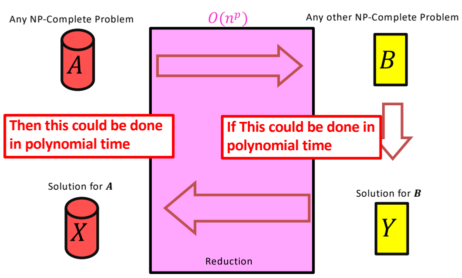
We now just need a FIRST NP-Hard problem

$A \le_p H \le_p B$

$\forall A \in NP$

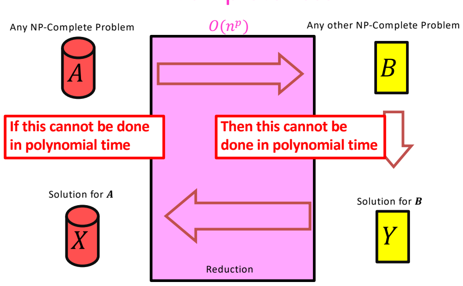
59

NP-Completeness



60

NP-Completeness



61

3-SAT

- Shown to be NP-Hard by Cook and Levin (independently)
- Given a 3-CNF formula (logical AND of clauses, each an OR of 3 variables), Is there an assignment of true/false to each variable to make the formula true?

$$\underbrace{(x \vee y \vee z)}_{\text{Clause}} \wedge (x \vee y \vee y) \wedge (u \vee y \vee z) \wedge (z \vee x \vee u) \wedge (x \vee y \vee z)$$

Variables

$x = true$
$y = false$
$z = false$
$u = true$

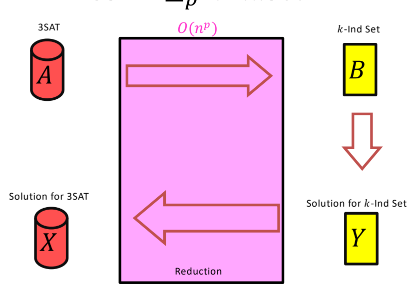
62

k-Independent Set is NP-Complete

1. Show that it belongs to NP
   - Give a polynomial time verifier (slide 56)
2. Show it is NP-Hard
   - Give a reduction from a known NP-Hard problem
   - Show $3SAT \leq_p kIndSet$

63

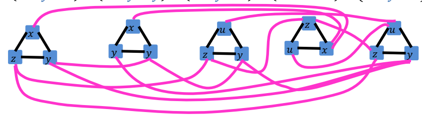
$3SAT \leq_p kIndSet$



64

Instance of 3SAT to Instance of kIndSet

$(x \vee y \vee z) \wedge (x \vee y \vee y) \wedge (u \vee y \vee z) \wedge (z \vee x \vee u) \wedge (x \vee y \vee z)$



For each clause, produce a triangle graph with its three variables as nodes

Connect each node to all of its opposites

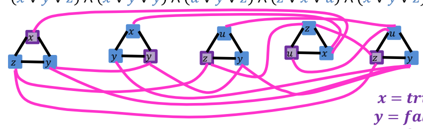
Let $k =$ number of clauses

There is a k-IndSet in this graph, iff there is a satisfying assignment

65

kIndSet ⇒ Satisfying Assignment

$(x \vee y \vee z) \wedge (x \vee y \vee y) \wedge (u \vee y \vee z) \wedge (z \vee x \vee u) \wedge (x \vee y \vee z)$



$x = true$
$y = false$
$z = false$
$u = true$

One node per triangle is in the Independent set: because we can have exactly k total in the set, and 2 in a triangle would be adjacent

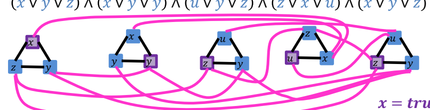
If x is selected in some triangle, x is not selected in any triangle: Because every x is adjacent to every x

Set the variable which each included node represents to "true"

66

Satisfying Assignment ⇒ kIndSet

$(x \vee y \vee z) \wedge (x \vee y \vee y) \wedge (u \vee y \vee z) \wedge (z \vee x \vee u) \wedge (x \vee y \vee z)$



$x = true$
$y = false$
$z = false$
$u = true$

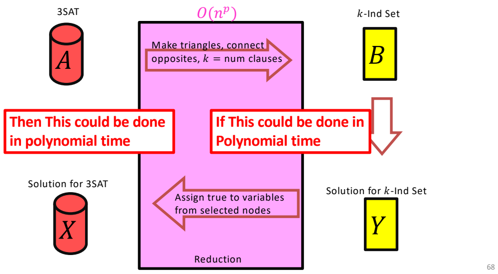
Use one true variable from the assignment for each triangle

The independent set has k nodes, because there are k clauses

If any variable x is true then x cannot be true

67

$3SAT \leq_p kIndSet$



68

k-Vertex Cover is NP

- To show: Given a potential solution, can we verify it in $O(n^p)$? $[n = V + E]$

How can we verify it?

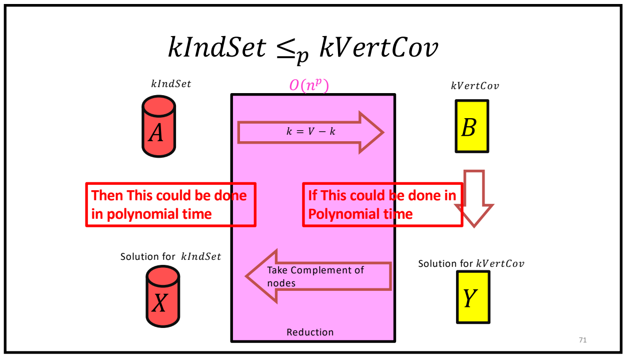
1. Check that it's of size k $O(V)$
2. Check that it's a Vertex Cover $O(E)$

69

k-Vertex Cover is NP-Complete

1. Show that it belongs to NP
   – Give a polynomial time verifier (slide 69)
2. Show it is NP-Hard
   – Give a reduction from a known NP-Hard problem
   – We showed $kIndSet \leq_p kVertCov$

70



$kIndSet \leq_p kVertCov$
$kIndSet$
$O(n^p)$
$kVertCov$
A
$k = V - k$
B
Then This could be done in polynomial time
If This could be done in Polynomial time
Solution for $kIndSet$
Take Complement of nodes
Solution for $kVertCov$
X
Y
Reduction
71