CS 401

Polynomial Time Reduction

Xiaorui Sun

Stuff

Homework 4 due today 11:59pm

Homework 5 will be released tomorrow

Final exam: Wednesday May 7 3:30pm – 5:30pm

- If you have exam conflicts, please let me know ASAP via email

Polynomial Time Reduction

Def A $\leq_P$ B: if there is an algorithm for problem A using a 'black box' (subroutine) that solve problem B s.t.,

- Algorithm uses only a polynomial number of steps
- Makes only a polynomial number of calls to a subroutine for **B**

Example

Algorithm for A:
Int i=0, i'=0;
Int j=0, j'=0;
…..
i=i+j;
(computation on i, j, i', j')
…..
Int x = B(i, j)
Int y = B(i', j')
…..
(compute z based on x and y)
Return z

Polynomial Time Reduction

Def A $\leq_P$ B: if there is an algorithm for problem A using a 'black box' (subroutine) that solve problem B s.t.,

- Algorithm uses only a polynomial number of steps
- Makes only a polynomial number of calls to a subroutine for **B**

Question: Is the following polynomial time reduction correct?

Interval Scheduling $\leq_P$ Max Independent Set

- Yes. Without the blackbox of max independent set, we still have a polynomial time algorithm for interval scheduling.
- If problem A can be solved in polynomial time, then A $\leq_P$ B holds for any problem B

Polynomial Time Reduction

Def $A \leq_P B$: if there is an algorithm for problem A using a 'black box' (subroutine) that solve problem B s.t.,

- Algorithm uses only a polynomial number of steps
- Makes only a polynomial number of calls to a subroutine for **B**

Question: Is the following polynomial time reduction correct?

Max Independent Set $\leq_P$ Interval Scheduling

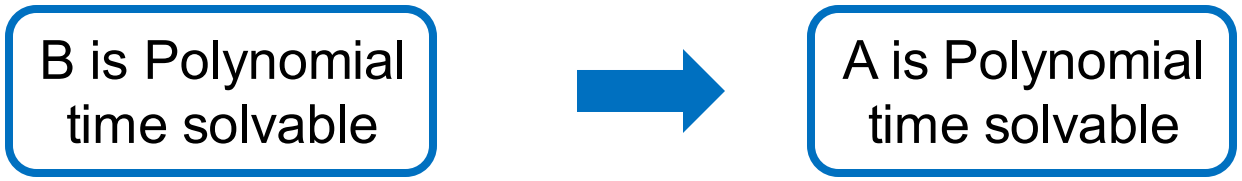
- Unlikely. If there is such a polynomial time reduction, then MIS can be solved in polynomial time.
- If $A \leq_P B$, then A is no harder than B

Polynomial Time Reduction

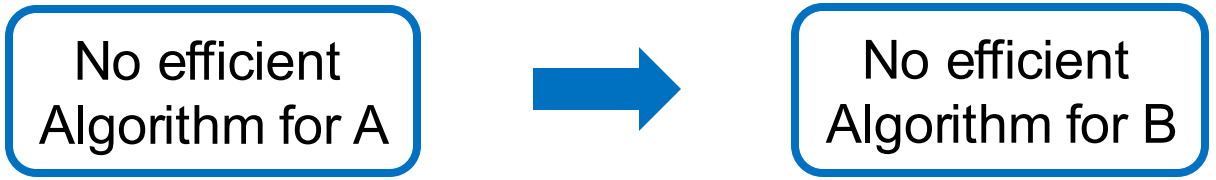
Def $A \leq_P B$: if there is an algorithm for problem A using a 'black box' (subroutine) that solve problem B s.t.,

- Algorithm uses only a polynomial number of steps
- Makes only a polynomial number of calls to a subroutine for **B**

So,



Conversely,



In words,

- Problem A is polynomial-time reducible to problem B
- B is as hard as A (it can be even harder)
- Informally, A is a special case of B

Polynomial Time Reduction

Purpose. Classify problems according to relative difficulty.

Design algorithms. If $A \leq_P B$ and B can be solved in polynomial-time, then A can also be solved in polynomial time.

Establish intractability. If $A \leq_P B$ and A cannot be solved in polynomial-time, then B cannot be solved in polynomial time.

Establish equivalence. If $A \leq_P B$ and $B \leq_P A$, we use notation $A \equiv_P B$.

up to cost of reduction

Polynomial Time Reduction

Basic reduction strategies

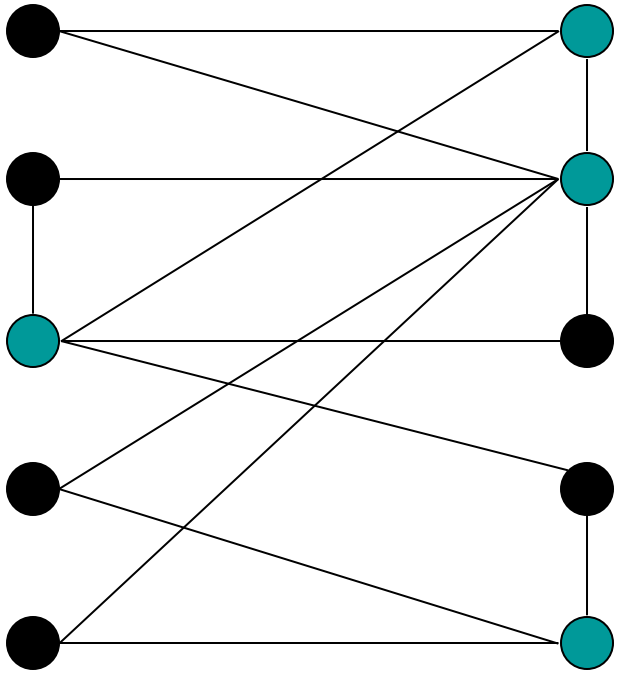
- Reduction by simple equivalence.
- Reduction from special case to general case.
- Reduction by encoding with gadgets.

Example 1: Vertex Cover $\equiv_p$ Indep Set

INDEPENDENT SET: Given a graph G = (V, E) and an integer k, is there a subset of vertices $S \subseteq V$ such that $|S| \geq k$, and for each edge at most one of its endpoints is in S?

Ex. Is there an independent set of size ≥ 6? Yes.

Ex. Is there an independent set of size ≥ 7? No.



independent set

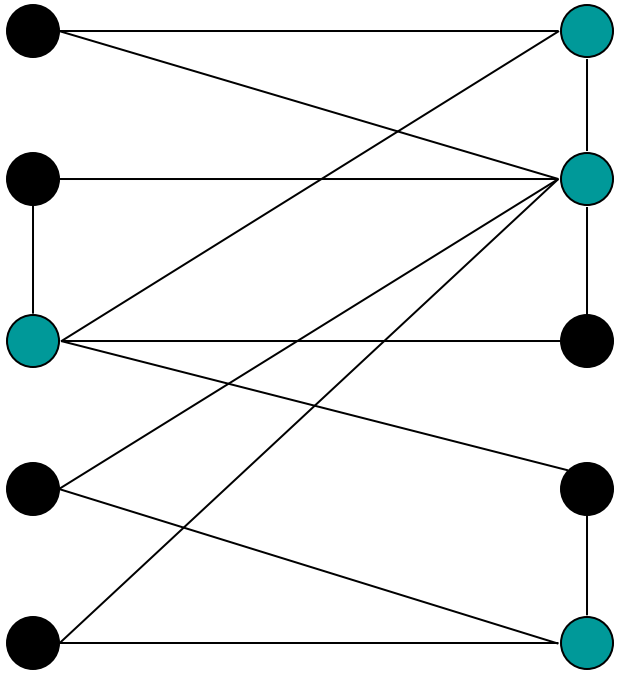
9

Example 1: Vertex Cover $\equiv_p$ Indep Set

VERTEX COVER: Given a graph G = (V, E) and an integer k, is there a subset of vertices $S \subseteq V$ such that $|S| \leq k$, and for each edge, at least one of its endpoints is in S?

Ex. Is there a vertex cover of size ≤ 4? Yes.

Ex. Is there a vertex cover of size ≤ 3? No.



Example 1: Vertex Cover $\equiv_p$ Indep Set

Claim: For any graph $G = (V, E)$, S is an independent set iff $V - S$ is a vertex cover

Pf: =>

Let S be a independent set of G

Then, S has at most one endpoint of every edge of G

So, $V - S$ has at least one endpoint of every edge of G

So, $V - S$ is a vertex cover.

<= Suppose $V - S$ is a vertex cover

Then, there is no edge between vertices of S (otherwise, $V - S$ is not a vertex cover)

So, S is an independent set.

Polynomial Time Reduction

Basic reduction strategies

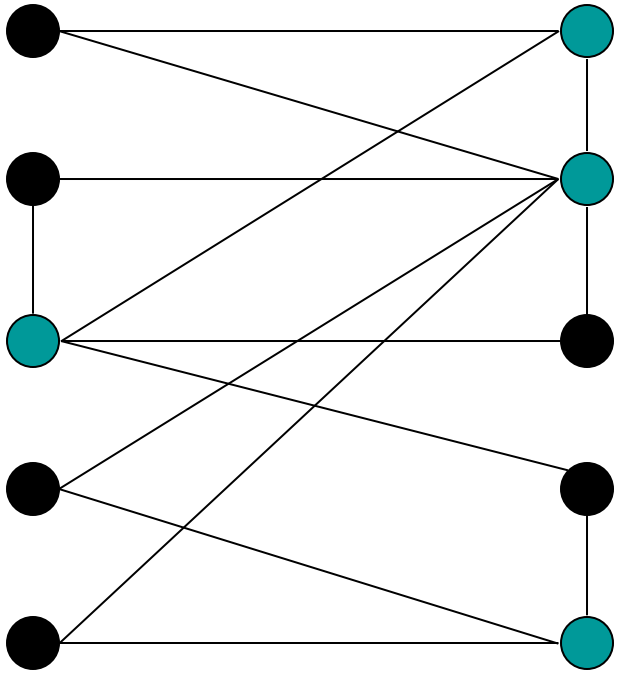
- Reduction by simple equivalence.
- Reduction from special case to general case.
- Reduction by encoding with gadgets.

Example 2: Vertex Cover $\leq_p$ Set Cover

VERTEX COVER: Given a graph G = (V, E) and an integer k, is there a subset of vertices $S \subseteq V$ such that $|S| \leq k$, and for each edge, at least one of its endpoints is in S?

Ex. Is there a vertex cover of size ≤ 4? Yes.

Ex. Is there a vertex cover of size ≤ 3? No.



Example 2: Vertex Cover $\leq_p$ Set Cover

SET COVER: Given a set U of elements, a collection $S_1, S_2, \ldots, S_m$ of subsets of U, and an integer k, does there exist a collection of $\leq$ k of these sets whose union is equal to U?

Ex:

U = { 1, 2, 3, 4, 5, 6, 7 }
k = 2

S_1 = {3, 7}	S_4 = {2, 4}
S_2 = {3, 4, 5, 6}	S_5 = {5}
S_3 = {1}	S_6 = {1, 2, 6, 7}

Example 2: Vertex Cover $\leq_p$ Set Cover

Claim: VERTEX-COVER $\leq_P$ SET-COVER.

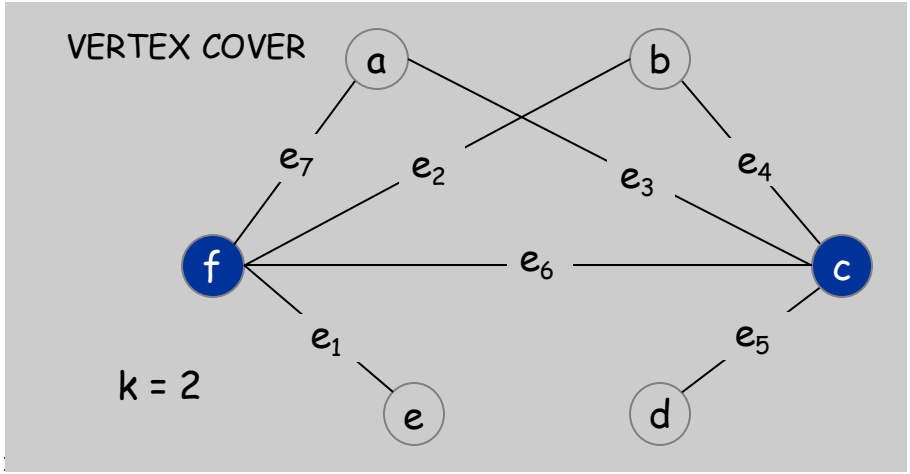
Pf: Given a VERTEX-COVER instance G = (V, E), k, we construct a set cover instance whose size equals the size of the vertex cover instance.

Construction:

Create SET-COVER instance:

- k = k, U = E, S_v = {e $\in$ E : e incident to v }

Set-cover of size $\leq$ k iff vertex cover of size $\leq$ k. ▪



SET COVER

U = { 1, 2, 3, 4, 5, 6, 7 }
k = 2
S_a = {3, 7}
S_b = {2, 4}
S_c = {3, 4, 5, 6}
S_d = {5}
S_e = {1}
S_f = {1, 2, 6, 7}

Polynomial Time Reduction

Basic reduction strategies

- Reduction by simple equivalence.
- Reduction from special case to general case.
- Reduction by encoding with gadgets.

Satisfiability

Literal: A Boolean variable or its negation. x_i or $\overline{x_i}$

Clause: A disjunction of literals. $C_j = x_1 \vee \overline{x_2} \vee x_3$

Conjunctive normal form: A propositional formula Φ that is the conjunction of clauses. $\Phi = C_1 \wedge C_2 \wedge C_3 \wedge C_4$

SAT: Given CNF formula Φ, does it have a satisfying truth assignment?

3-SAT: SAT where each clause contains exactly 3 literals.

Ex: $(\overline{x_1} \vee x_2 \vee x_3) \wedge (x_1 \vee \overline{x_2} \vee x_3) \wedge (x_2 \vee x_3) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_3})$

Yes: x_1 = true, x_2 = true x_3 = false.

3 Satisfiability Reduces to Independent Set

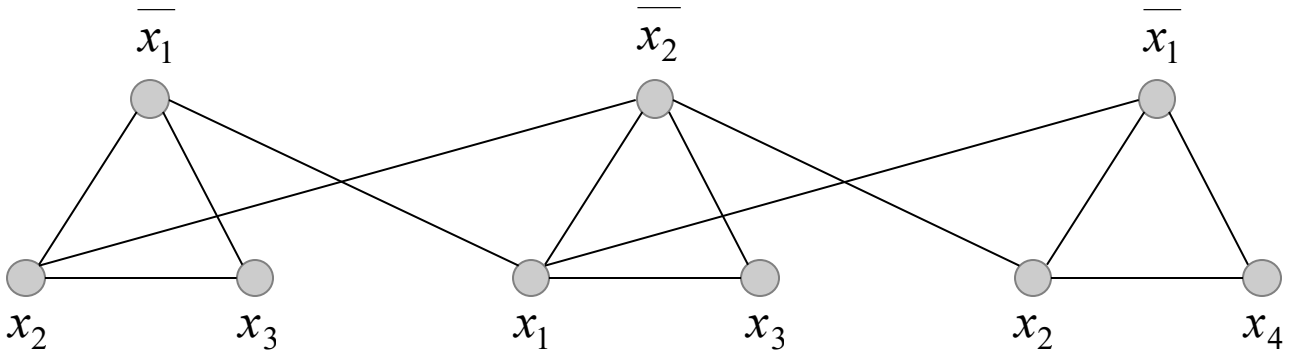
Claim: 3-SAT $\leq_P$ INDEPENDENT-SET.

Pf: Given an instance Φ of 3-SAT, we construct an instance (G, k) of INDEPENDENT-SET that has an independent set of size k if and only if Φ is satisfiable.

Construction

- G contains 3 vertices for each clause, one for each literal.
- Connect 3 literals in a clause in a triangle.
- Connect literal to each of its negations.

G



k = 3

$$\Phi = \left(\overline{x_1} \vee x_2 \vee x_3 \right) \wedge \left(x_1 \vee \overline{x_2} \vee x_3 \right) \wedge \left(\overline{x_1} \vee x_2 \vee x_4 \right)$$

3 Satisfiability Reduces to Independent Set

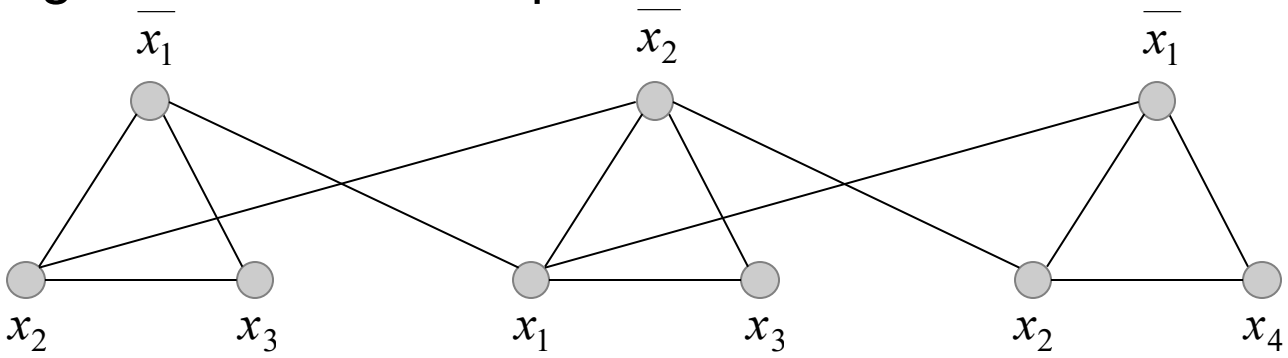
Claim: G contains independent set of size k = $|\Phi|$ iff Φ is satisfiable.

Pf $\Rightarrow$ Let S be independent set of size k.

- S must contain exactly one vertex in each triangle.
- Set these literals to true. ← and any other variables in a consistent way
- Truth assignment is consistent and all clauses are satisfied.

Pf $\Leftarrow$ Given satisfying assignment, select one true literal from each triangle. This is an independent set of size k. ▪

G



k = 3

$$\Phi = \left(\overline{x_1} \vee x_2 \vee x_3\right) \wedge \left(x_1 \vee \overline{x_2} \vee x_3\right) \wedge \left(\overline{x_1} \vee x_2 \vee x_4\right)$$

Review

Basic reduction strategies:

- Simple equivalence: INDEPENDENT-SET $\equiv_P$ VERTEX-COVER.
- Special case to general case: VERTEX-COVER $\leq_P$ SET-COVER.
- Encoding with gadgets: 3-SAT $\leq_P$ INDEPENDENT-SET.

Transitivity. If $X \leq_P Y$ and $Y \leq_P Z$, then $X \leq_P Z$.

Pf idea. Compose the two algorithms.

Ex: 3-SAT $\leq_P$ INDEPENDENT-SET $\leq_P$ VERTEX-COVER $\leq_P$ SET-COVER.