W, 11/06/19

Fall'19 CSCE 629

Analysis of Algorithms

Fang Song

Texas A&M U

Lecture 26

- Reductions

Credit: based on slides by A. smith & K. Wayne

Recall: polynomial-time reduction

- **Def.** Problem X polynomial reduces to Problem Y if arbitrary instance of X can be solved using:
  - Polynomial number of standard computation steps
  - & polynomial number of calls to oracle that solves A

Notation. $X \leq_{P,Cook} Y$ (or $X \leq_P Y$)

! Mind your direction, don't confuse $X \leq_{\mathrm{P}} Y$ with $Y \leq_{\mathrm{P}} \mathrm{X}$

Quiz

- Which of the following poly-time reductions are known?
  A. $\mathrm{FIND{-}MAX{-}FLOW} \leq_P \mathrm{FIND{-}MIN{-}CUT}$
  B. $\mathrm{FIND{-}MIN{-}CUT} \leq_P \mathrm{FIND{-}MAX{-}FLOW}$
  C. Both A and B
  D. Neither A nor B

VALUES VS. ACTUAL FLOW/CUT

Simplification: decision problems

- Search problem. Find some structure.
  - Example. Find a minimum cut.
- Decision problem.
  - Problem X is a set of strings [e.g., strings that encode graphs containing a triangle]
  - Instance: string s [e.g., encoding of a graph]
  - YES instance: $s \in X$; NO instance: $s \notin X$
  - Algorithm A solves problem X: $A(s) = yes$ iff. $s \in X$
  - Ex. Does there exist a cut of size $\leq k$?
- Self-reducibility. Search problem $\leq_P$ Decision version
  - Applies to all NP-complete problems in this chapter [Recall HW1]
  - Justifies our focus on decision problems

Polynomial-time transformation

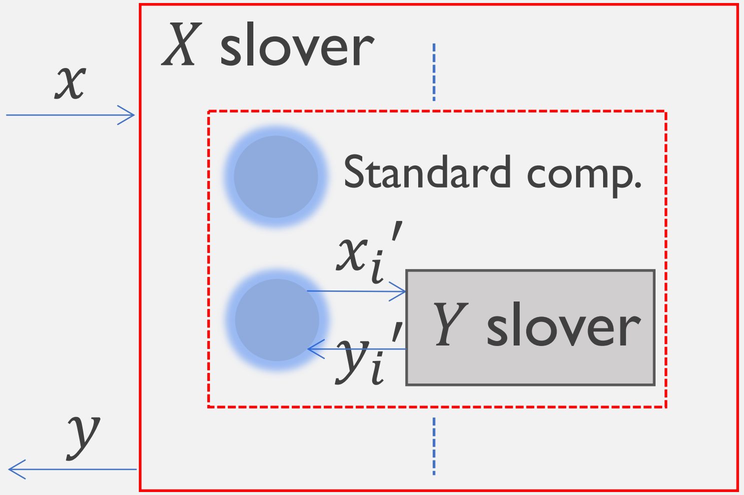
- Karp reduction. (Decision) problem X polynomial transforms to Problem Y if given any x, we can construct y such that
  - size $|y| = poly(|x|)$
  - $x \in X$ iff. $y \in Y$.

$$X \leq_{P,Karp} Y$$

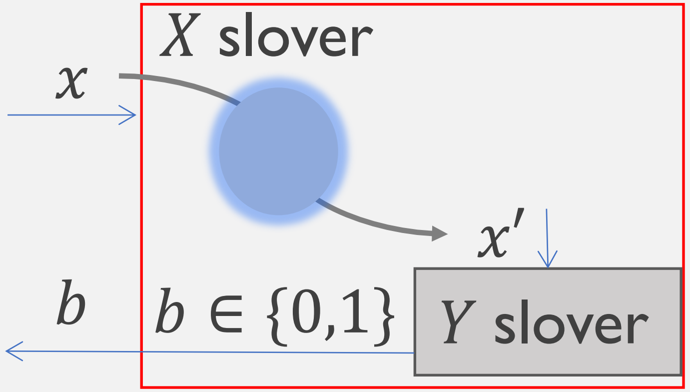
Polynomial-time reduction vs. transformation

Cook (Turing) reduction vs. Karp reduction

$X \leq_{P,Cook} Y$



$X \leq_{P,Karp} Y$ (Decision problems)



N.B. Polynomial transformation is polynomial reduction with just one call to oracle for Y, exactly at the end of the algorithm for X.

Open question. Are these two concepts the same?

Basic reduction strategies

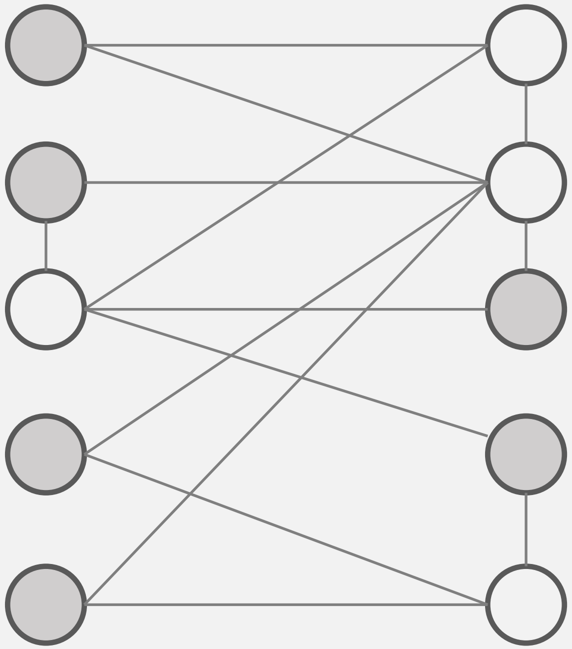
- Reduction by simple equivalence
- Reduction from special case to general case
- Reduction by encoding with gadgets

Independent set

Input. Graph $G = (V, E)$ and an integer k

- Independent set $S \subseteq V$: subset of vertices such that for each edge at most one of its endpoints is in S

Goal. Decide if there is an independent set S with $|S| \geq k$



 independent set

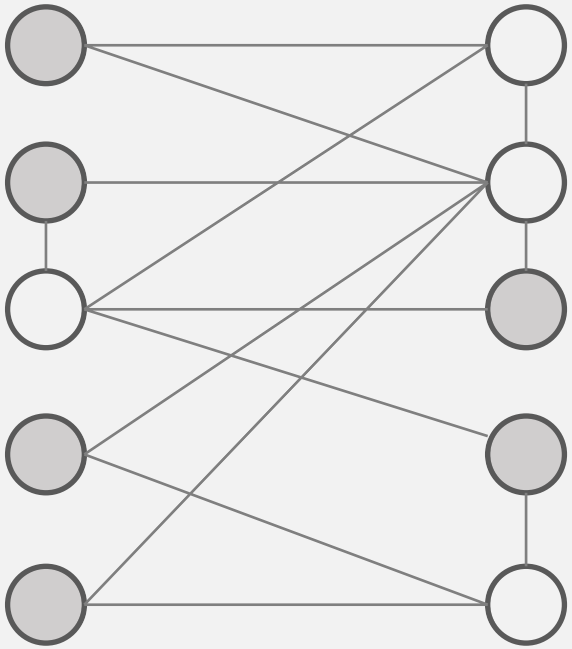
- Is there an independent set of size ≥ 6? ☺
- Is there an independent set of size ≥ 7? ☹

Vertex cover

Input. Graph $G = (V, E)$ and an integer k

- Vertex cover $S \subseteq V$: subset of vertices such that for each edge at least one of its endpoints is in S

Goal. Decide if there is an independent set S with $|S| \leq k$



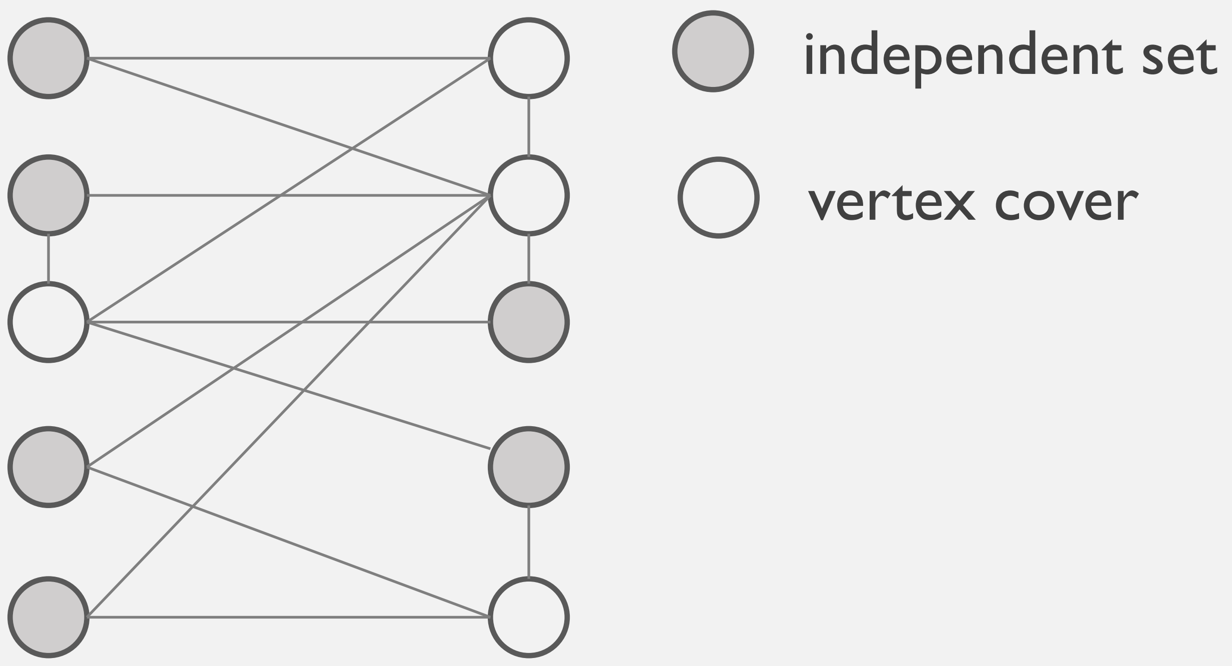
Vertex cover

- Is there an vertex cover of size ≤ 4? ☺
- Is there an independent set of size ≤ 3? ☹

Independent set and Vertex cover

Claim. $\text{VERTEX−COVER} \equiv_P \text{INDEPENDENT−SET}$

Pf. We show S is an independent set iff. $V \backslash S$ is a vertex cover



Independent set and Vertex cover

Claim. $\text{VERTEX-COVER} \equiv_P \text{INDEPENDENT-SET}$

Pf. We show S is an independent set iff. $V \setminus S$ is a vertex cover

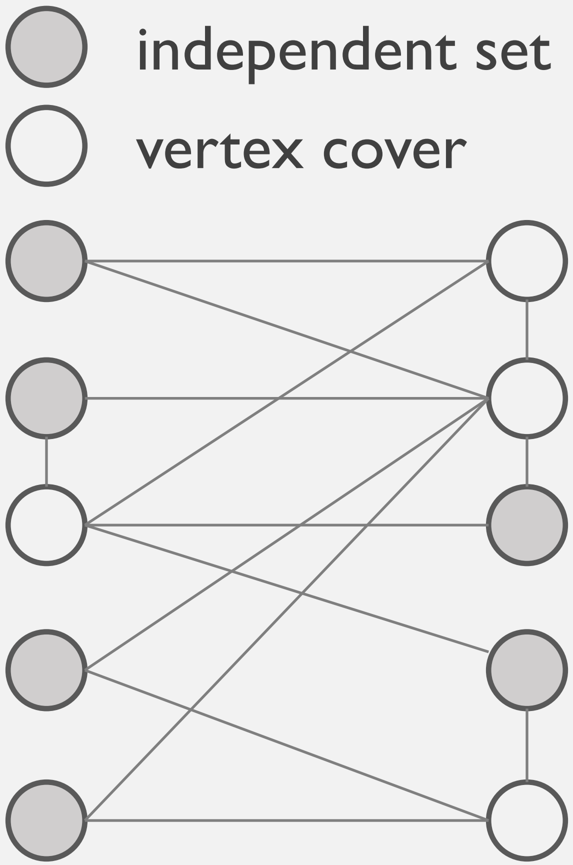
$\leq$ ($\Leftarrow$) Let S be any independent set

- Consider an arbitrary edge (u, v)
- S independent $\Rightarrow u \notin S$ or $v \notin S \Rightarrow u \in V \setminus S$ or $v \in V \setminus S$
- Thus $V \setminus$S covers (u, v)

$\geq$ ($\Rightarrow$) Let $V \setminus S$ be any vertex cover

- Consider two nodes $u \in S$ and $v \in S$
- Observe that $(u, v) \notin E$ since $V \setminus S$ is a vertex cover
- Thus no two nodes in S are joined by an edge

$\Rightarrow$ S is an independent set



Basic reduction strategies

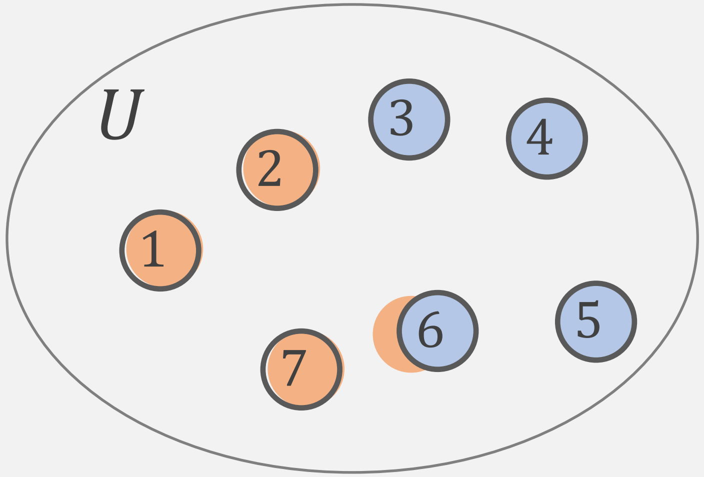
- Reduction by simple equivalence
- Reduction from special case to general case
- Reduction by encoding with gadgets

Set cover

Input. Set U of n elements, $S_1, \ldots, S_m$ of subsets of U, integer k

Goal. Decide if there is an collection of $\leq k$ of these sets whose union is equal to U

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



Sample application.

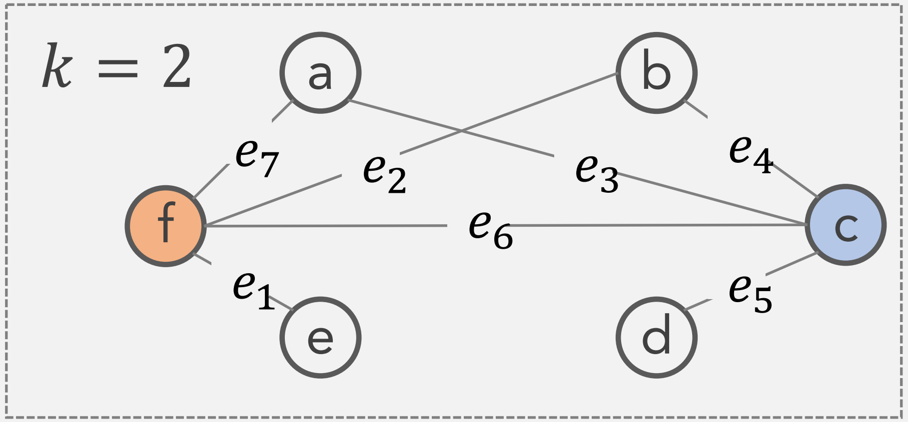
- Set U of n capabilities that our computer system needs to have
- m available pieces of software, ith software provides the set $S_i \subseteq U$ capabilities
- Goal: achieve all n capabilities using fewest pieces of software

Vertex cover reduces to set cover

Claim. $\text{VERTEX}-\text{COVER} \leq_P \text{SET}-\text{COVER}$

Pf. Given a $\text{VERTEX}-\text{COVER}$ instance $G = \langle (V, E), k \rangle$, we construct a $\text{SET}-\text{COVER}$ instance whose solution size equals the size of the vertex cover instance

Reduction: on input $\langle G = (V, E), k \rangle$
Output: // a SET-COVER instance
$k = k, U = E, S_v = \{e \in E : e \text{ incident to } v\}$ for every $v \in V$



$\Rightarrow$

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