Claim: G contains an ind. set of size exactly K
$\iff$ the formula f is satisfiable

proof: 1) suppose f is satisfiable. Pick any satisfying assignment. Each clause in f has ≥ 1 TRUE literal. Thus we can choose a subset S of K vertices in G that contains exactly one vertex per group such that the corresponding K literals are all TRUE. The set S is an ind. set because it does not contain both endpoints of any edge of a group, nor of any edge that connects inconsistent literals (as it is derived from a consistent truth assignment)

2) suppose G contains an ind. set of size K. Each vertex in S must be in a different group. Assign TRUE to each literal of S. Since inconsistent literals are connected by an edge, this assignment is consistent. Since S contains 1 vertex per group, each clause in f contains (at least) one TRUE literal $\Rightarrow$ f is satisfiable

Exercises: (easy)

(Maximum) Clique : compute the largest complete subgraph in a given graph

other name for a complete graph

Show that Maximum Clique is NP-hard

Def.: a vertex cover of a graph is a set of vertices that includes at least one endpoint of every edge of the graph

(Minimum) Vertex Cover: compute the smallest vertex cover in a given graph

Show that Minimum Vertex Cover is NP-hard

Approximation Algorithms

... for NP-hard problems. Assumption: $P \neq NP$

Optimization problems:

$\Pi : I \times S$ — inputs (I), solutions (S)

$c : S \longrightarrow \mathbb{R}^+$

$\forall i \in I \quad S(i) = \{ s \in S : i \Pi s \}$

"feasible solutions"

$s^* \in S(i)$ and $c(s^*) = \substack{\min \\ \max}\ c(S(i))$

Approximation:

$s \in S(i)$

ok if $s \neq s^*$, but I want:

1) guarantee on the quality of s

2) guarantee on the complexity: polynomial-time algorithm

Definition: let Π an optimization problem, and let A_π an algorithm for Π that returns, $\forall i \in I$, $A_\pi(i) \in S(i)$. We say that A_π has an <u>approximation factor</u> of $\rho(n)$ if $\forall i \in I$ s.t. $|i| = n$ we have min.: $$\frac{c\left(A_\Pi(i)\right)}{c\left(s^*(i)\right)} \leq \rho(n)$$

max.: $$\frac{c\left(s^*(i)\right)}{c\left(A_\Pi(i)\right)} \leq \rho(n)$$

$c : S \to \mathbb{R}^+ \Rightarrow \quad \geq 1$

Goal : $\rho(n) = 1 + \varepsilon$ with ε as small as possible

We'll get $\varepsilon = 1$ for vertex cover $\rightarrow$ "2-approximation"
$\varepsilon = \log_2 n$ for set cover

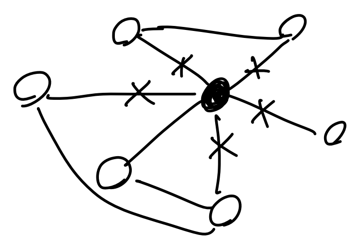
$\exists$ problems for which one can prove that $\rho(n) = \Omega(n^\varepsilon)$
$\forall\, \varepsilon < 1$ (e.g., clique)

Much stronger approximation : $\rho(n) = 1 + \varepsilon \;\; \forall \varepsilon > 0$

Definition : an <u>approximation scheme</u> for Π is an algorithm with 2 inputs $A_\Pi(i, \varepsilon)$ that $\forall \varepsilon$ is a $(1+\varepsilon)$-approximation

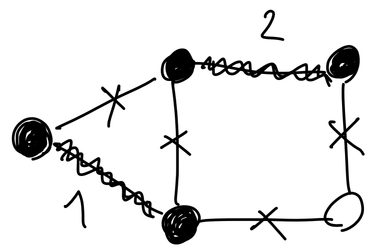
Definition: an approximation scheme is <u>polinomial</u> (PTAS) if $A_\Pi(i, \varepsilon)$ is polinomial in $|i|$ $\forall \varepsilon$ fixed.

Approximation algorithms for Vertex Cover



greedy approach:
- select the vertex with highest degree
- "remove" touched edges
- repeat

unfortunately one can show that for this algorithm
$\rho(n) = \Omega(\log n)$ (exercise!)



greedy approach:
- choose any edge
- add its endpoints to the solution
- "remove" touched edges
- repeat

Approx_VERTEX_COVER (G)

$V' = \emptyset$

$E' = E$

while $E' \neq \emptyset$ do

let (u, v) be an arbitrary edge of E'

$V' = V' \cup \{u, v\}$

$E' = E' \setminus \{(u, z), (v, w)\}$

return V'

Complexity : $O(n + m)$

Analysis : $|V'| / |V^*| \leq 2$

A = set of selected edges

A is a <u>matching</u> : $\forall e, e' \in A \Rightarrow e \cap e' = \emptyset$

i.e. no vertices in common

Appox_Vertex_Cover selects a <u>maximal</u> matching

↳ $\forall$ edge y, $A \cup y$ is <u>not</u> a matching

1) $|V^*|$ vs. $|A|$?

$\underbrace{|V^*| \geq |A|}$

A is a matching => in V^* there must be ≥ 1 vertex $\forall$ edge of A

2) $|V'|$ vs. $|A|$?

$\underbrace{|V'| = 2|A|}$ by construction

$$\Rightarrow |V'| \leq \cancel{2|A|} \leq 2|V^*|$$

$$\Rightarrow \frac{|V'|}{|V^*|} \leq 2$$

=> Approx_Vertex_Cover is a 2-approximate algorithm for Vertex Cover

Exercise: show that the approximation factor of Approx_Vertex_Cover is <u>exactly</u> 2