# Approximation Algorithms I

Definitions
Vertex cover
Set cover
Partition

} NP-complete problems or NP-hard

## Approximation Algos & Schemes

An algorithm for a problem of size $n$ has an approximation ratio $\rho(n)$ if for any input, algorithm produces a solution of cost $C$ such that

$$\max\left(\frac{C}{C_{opt}}, \frac{C_{opt}}{C}\right) \leq \rho(n)$$

Algorithm is an $\rho(n)$-approximation algorithm

An approximation scheme takes as input $\varepsilon > 0$ and for any fixed $\varepsilon$, the scheme is a $(1+\varepsilon)$-approximation algorithm.

Polynomial time approximation scheme (PTAS): polynomial in $n$

Fully PTAS: polynomial in $n$ and $\frac{1}{\varepsilon}$

$O(n^{2/\varepsilon})$ PTAS not FPTAS. $O(n/\varepsilon^2)$ FPTAS

# Vertex Cover

Undirected graph $G(V, E)$
Find a subset $V' \subseteq V$ such that if $(u,v)$ is an edge of $G$, then either
$u \in V'$ or $v \in V'$ or both.
Find a $V'$ so $|V'|$ is minimum.

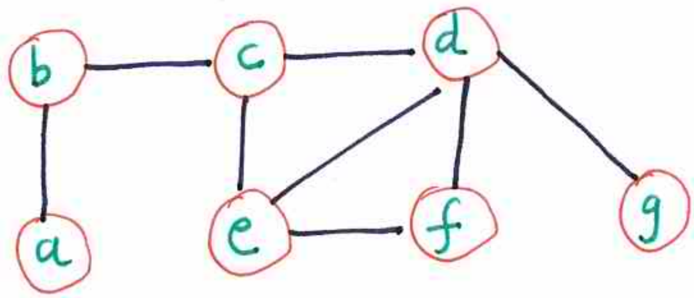
## Approx-Vertex-Cover

$C \leftarrow \phi$
$E' \leftarrow E$
while $E' \neq \phi$
  Pick $(u,v) \in E$ arbitrarily
  $C \leftarrow C \cup \{u\} \cup \{v\}$
  Delete from $E'$ all edges incident on $u$ or $v$

Return $C$

Runs in poly time. Produces a vertex cover.
How close to optimal?

## EXAMPLE




Approx-Vertex-Cover could pick $(b,c)$, $(e,f)$, $(d,g)$

$C = \{b, c, d, e, f, g\}$ $|C| = 6$

Optimal solution $C_{opt} = \{b, d, e\}$ $|C_{opt}| = 3$

## Approx-Vertex-Cover is a 2-approximation algorithm

Proof: Let $A$ denote the edges that are picked.

Optimal cover $C_{opt}$ must include at least one endpoint of each edge in $A$ (and other edges)

No two edges in $A$ share an endpoint.

$|A|$ is a lower bound for $|C_{opt}|$, $|C_{opt}| \geq |A|$

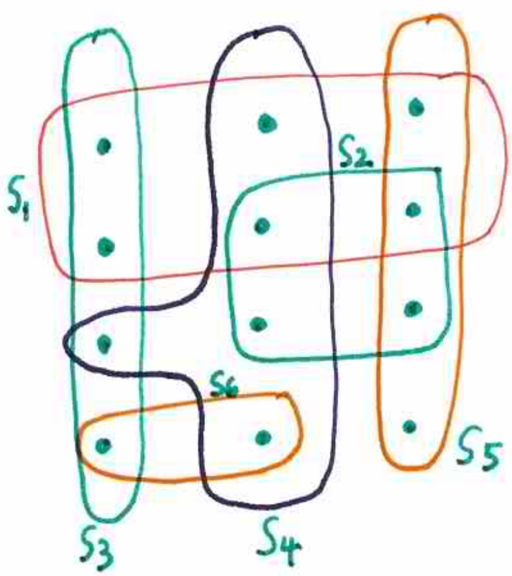
Number of vertices in $C = 2|A|$

$|C| \leq 2|C_{opt}|$

$\boxtimes$

# Set-Cover

Given a set $X$ and a family of (possibly overlapping) subsets $S_1, S_2, \ldots, S_m \subseteq X$ such that $\bigcup_{i=1}^{m} S_i = X$, find $C \subseteq \{1, 2, \ldots m\}$ such that $\bigcup_{i \in C} S_i = X$, while minimizing $|C|$.




Approx_Set_Cover (on next page) selects $S_1, S_4, S_5, S_3$ in that order

Optimal: $S_3, S_4, S_5$

## Approx_Set_Cover

$$\boxed{|X| = n}$$

$C = \emptyset$
While elements in $X$ remain
  Pick largest $S_i$; $C = C \cup \{i\}$
  Remove all elements in $S_i$ from $X$
    and other $S_j$

Return $C$

Poly time, returns a cover

## Approx_Set_Cover is a $(\ln(n)+1)$-approximation algo

Proof: Assume there is a cover $C_{opt}$ $|C_{opt}| = t$

Let $X_k$ be set of elements in iteration $k$
$(X_0 = X)$

$\forall k$, $X_k$ can be covered by $t$ sets.

$\Rightarrow$ one of them covers at least $\frac{|X_k|}{t}$ elements.

$\Rightarrow$ algo picks a set of (current) size $\geq \frac{|X_k|}{t}$

$\Rightarrow$ $\forall k \quad |X_{k+1}| \leq \left(1 - \frac{1}{t}\right)|X_k|$

$\boxed{\text{More careful analysis (see CLRS, Ch 35) relates } \rho(n) \text{ to harmonic numbers. } t \text{ should shrink!}}$

## Proof (contd.)

$\Rightarrow \forall k, \ |X_{k+1}| \leq \left(1 - \frac{1}{t}\right) |X_k|$

elements in $X = X_0$

$\Rightarrow \forall k, \ |X_k| \leq \left(1 - \frac{1}{t}\right)^k \cdot n$

$\leq e^{-k/t} \cdot n$

Algorithm terminates when $|X_k| < 1$, i.e., $|X_k| = 0$ and cost $= k$.

$e^{-k/t} \cdot n < 1$

$e^{k/t} > n$

When $\frac{k}{t} > \ln(n)$ and algorithm terminates.

$\therefore \frac{k}{t} \leq \ln(n) + 1$

So we have an $(\ln(n)+1)$ - approximation algorithm. $\boxtimes$

Approximation ratio gets worse for larger problems.

# PARTITION

Set $S$ of $n$ items with weights $s_1, \dots s_n$

Assume $s_1 \geq s_2 \geq \dots \geq s_n$ WLOG

Partition into A and B to minimize

$$\max\Big(\underbrace{\sum_{i \in A} s_i}_{w(A)}, \underbrace{\sum_{i \in B} s_i}_{w(B)}\Big)$$

Define $2L = \sum_{i=1}^{n} s_i = w(S)$

Optimum solution $\geq L$.

Want a PTAS. Note: 2-approx algo trivial.
$(1+\varepsilon)$-approximation

(FPTAS also exist for this problem)

## APPROX - PARTITION

Define $m = \left\lceil \frac{1}{\varepsilon} \right\rceil - 1$ $\quad \varepsilon \approx \frac{1}{m+1}$

First phase: Find an optimal partition $A'$, $B'$ of $s_1, \ldots, s_m$

takes $O(2^m)$ time!

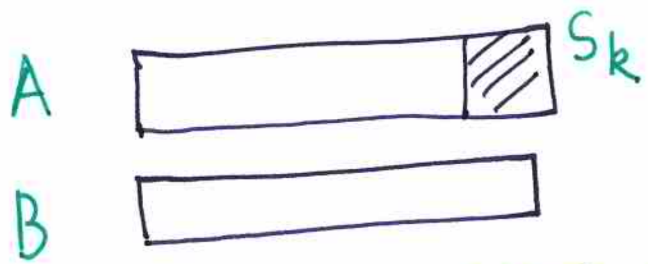
Second phase: $A \leftarrow A'$ $\quad B \leftarrow B'$

for $i = m+1$ to $n$
  if $w(A) \leq w(B)$
    $A = A \cup \{i\}$
  else $B = B \cup \{i\}$

## APPROX-PARTITION IS PTAS.

WLOG, assume $w(A) \geq w(B)$

approximation ratio $= \dfrac{w(A)}{L}$




$k$ is the LAST item added to $A$.

Could have been added in first or second phase.

## PROOF (contd.)

1) $k$ is added to $A$ in first phase.
This means $A = A'$. We have an optimal partition since we can't do better than $w(A')$ when we have $n \geq m$ items, and we know $w(A')$ is optimal for the $m$ items.

2) $k$ is added to $A$ in second phase.
We know $w(A) - S_k \leq w(B)$
This is why $k$ was added to $A$. (Note $w(B)$ may have increased after this addition to $A$).

$w(A) + w(B) = 2L$

$$\Rightarrow w(A) - S_k \leq 2L - w(A)$$
$$\Rightarrow w(A) \leq L + \frac{S_k}{2}$$

Since $S_1 \geq S_2 \ldots \geq S_n$ we can say that
$S_1, S_2, \ldots S_m$ all $\geq S_k$

$2L \geq (m+1) S_k$ since $k > m$.

$$\frac{w(A)}{L} \leq \frac{L + S_k/2}{L} = 1 + \frac{S_k}{2L} \leq 1 + \frac{S_k}{(m+1)S_k}$$
$$= 1 + \frac{1}{m+1}$$
$$= 1 + \varepsilon.$$

$\boxtimes$

# Approx_Vertex_Cover_Natural

$C \leftarrow \emptyset$

$E' \leftarrow E$

while $E' \neq \emptyset$

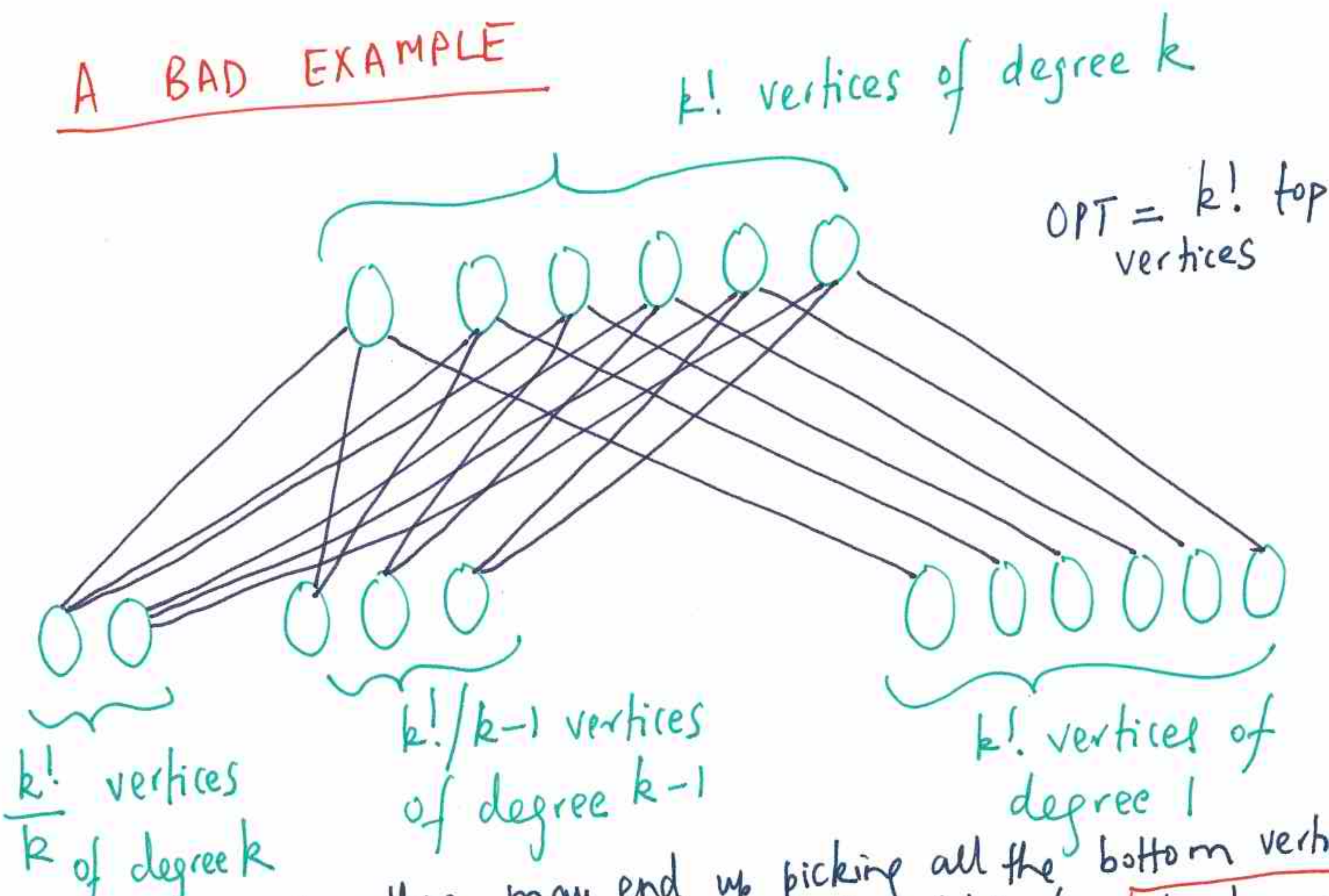
  pick $v$ with maximum degree

  $C = C \cup \{v\}$

  Remove $v$ and all incident edges from $E'$

return $C$

## A BAD EXAMPLE




Algorithm may end up picking all the bottom vertices

$SOL = k!\left(\frac{1}{k} + \frac{1}{k-1} + \cdots 1\right) \approx k! \log k$. $\boxed{\log k \text{ worse}}$

# APPROX-VERTEX-COVER-NATURAL IS LOG(n)-APPROX

$|G| = n$ (# edges) $\quad G \equiv G_0$

$G_0 \to G_1 \to G_2 \; \ldots \; G_m$ with vertex selection & edge deletion

$m = |C^*|$ # vertices in optimal vertex cover

Picking maximum degree vertex of $G_{i-1}$
→ call the degree $d_i$
→ delete edges incident on picked vertex to get $G_i$

$$|G_m| = |G_0| - \sum_{i=1}^{m} d_i$$

# edges

Also, $\displaystyle\sum_{i=1}^{m} d_i \geq \sum_{i=1}^{m} \frac{|G_{i-1}|}{m}$

(because given $|G_{i-1}|$ edges can be covered by $m$ vertices we know there is a vertex with degree at least $\frac{|G_{i-1}|}{m}$)

$$\geq \sum_{i=1}^{m} \frac{|G_m|}{m} \quad \text{since } |G_i| \leq |G_{i-1}|, \forall i$$

$$= |G_m|$$

$\Rightarrow |G_0| - |G_m| \geq |G_m|$ ⇒ After $m$ iterations we have deleted half or more edges from $G/G_0$.

smaller than $\sum_{i=1}^{m} d_i$ ⇒ $m \cdot \log|G|$ vertex cover. ⊠