Topic 24: Approximation Algorithms

(CLRS 35.0–35.3)

CPS 230, Fall 2001

- Finding solution to NP-complete problem is difficult.
- Two possible approaches.
  - If input is small enough, use exponential algorithm.
  - Otherwise, craft poly-time **approximation algorithm**.

We'll look at approximation algorithms for

1. Vertex Cover
2. Traveling Salesman Problem
3. Set Partition Problem

Definitions

- Optimization problem on input of size n.
- C^* = cost of optimal solution.
- C = cost of approximation algorithm's solution.
- **Ratio Bound:** $\rho(n)$ such that for input size n

$$\max\left(\frac{C}{C^*}, \frac{C^*}{C}\right) \leq \rho(n) \ .$$

- **Relative Error Bound:** $\epsilon(n)$ such that

$$\frac{|C - C^*|}{C^*} \leq \epsilon(n) \ .$$

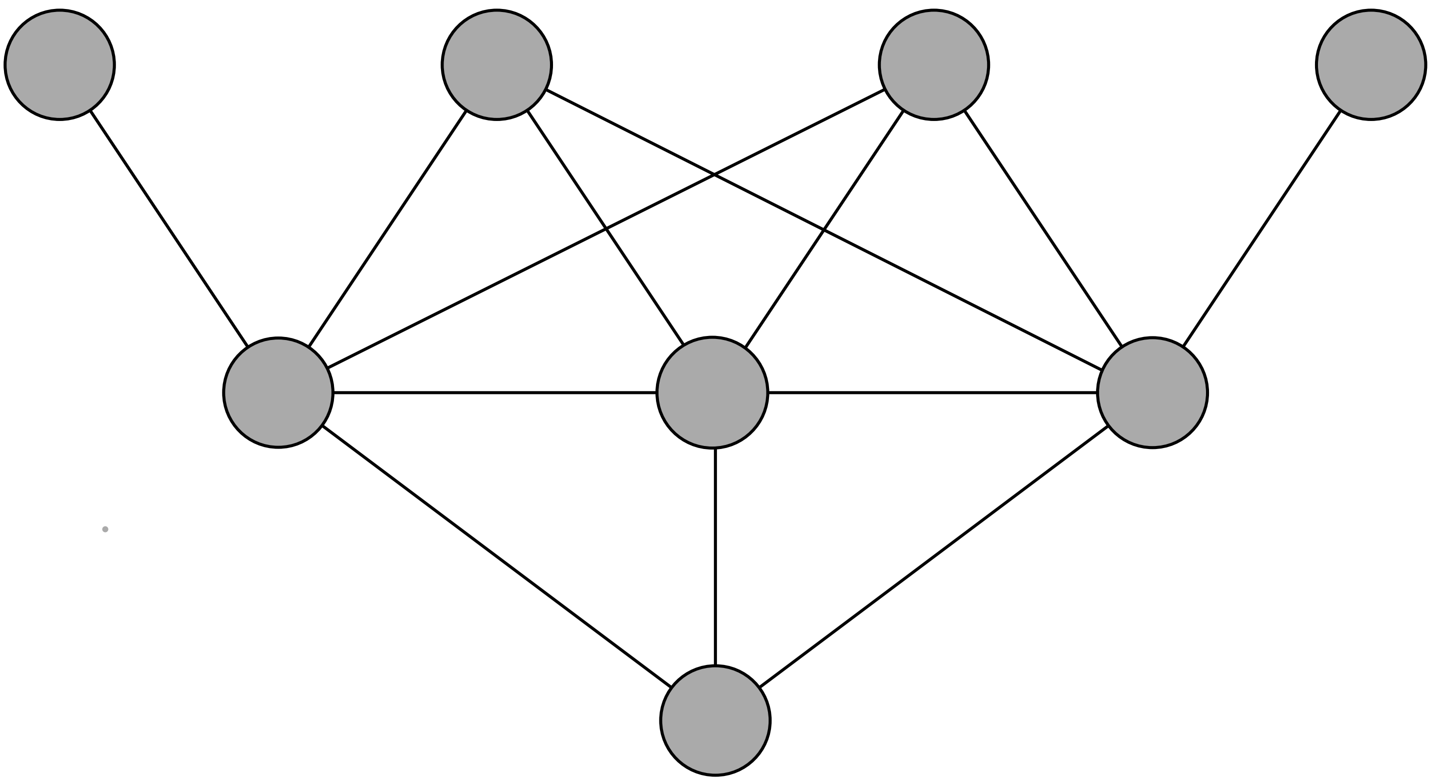
for any n.

Vertex Cover Problem

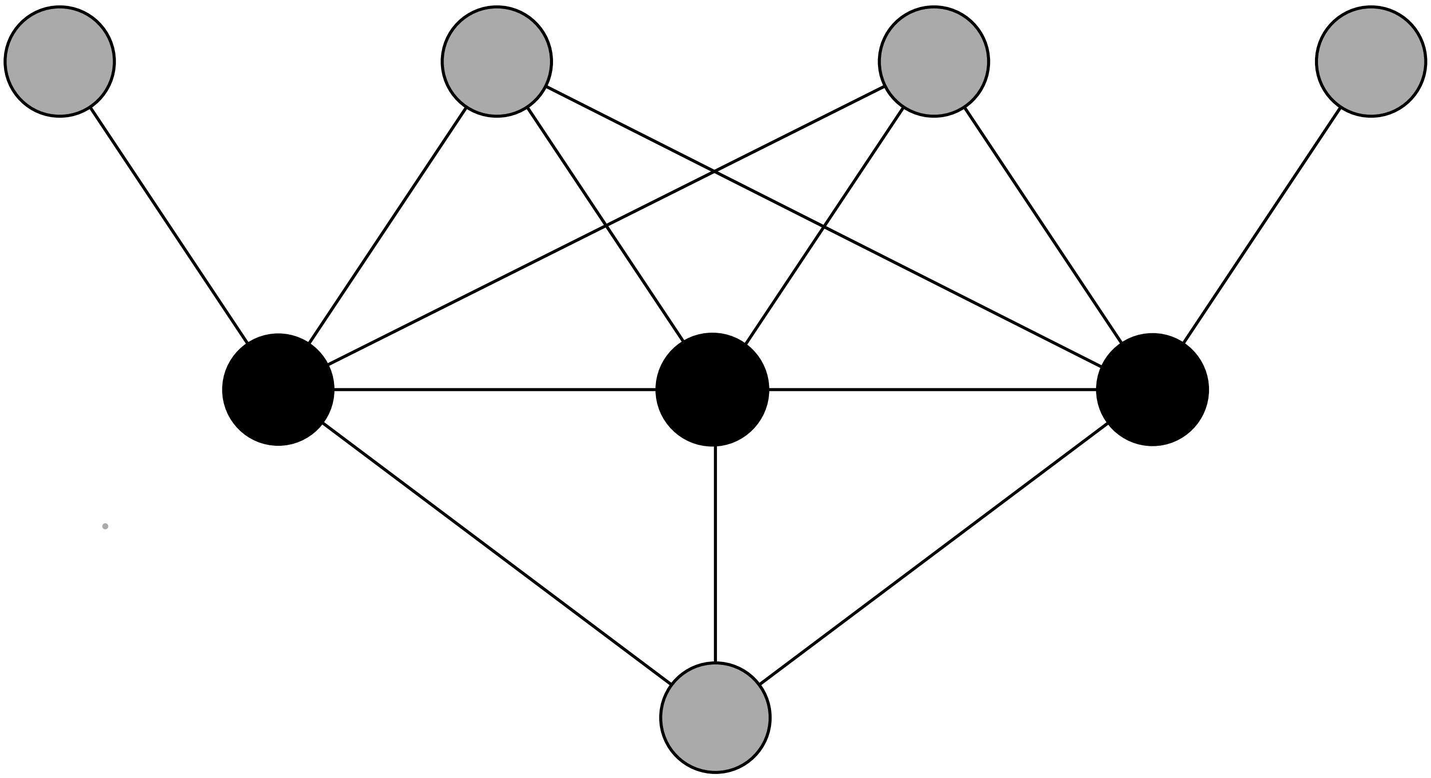
- Undirected graph $G = (V, E)$.
- **Vertex cover** of G is $V' \subseteq V$ such that for every $(u, v) \in E$, either $u \in V'$ or $v \in V'$ (or both).
- **Vertex-cover problem:** find vertex cover of minimum size (**optimal vertex cover**).
- NP-complete (reduction from CLIQUE; see CLRS).

Example

- Find optimal vertex cover:



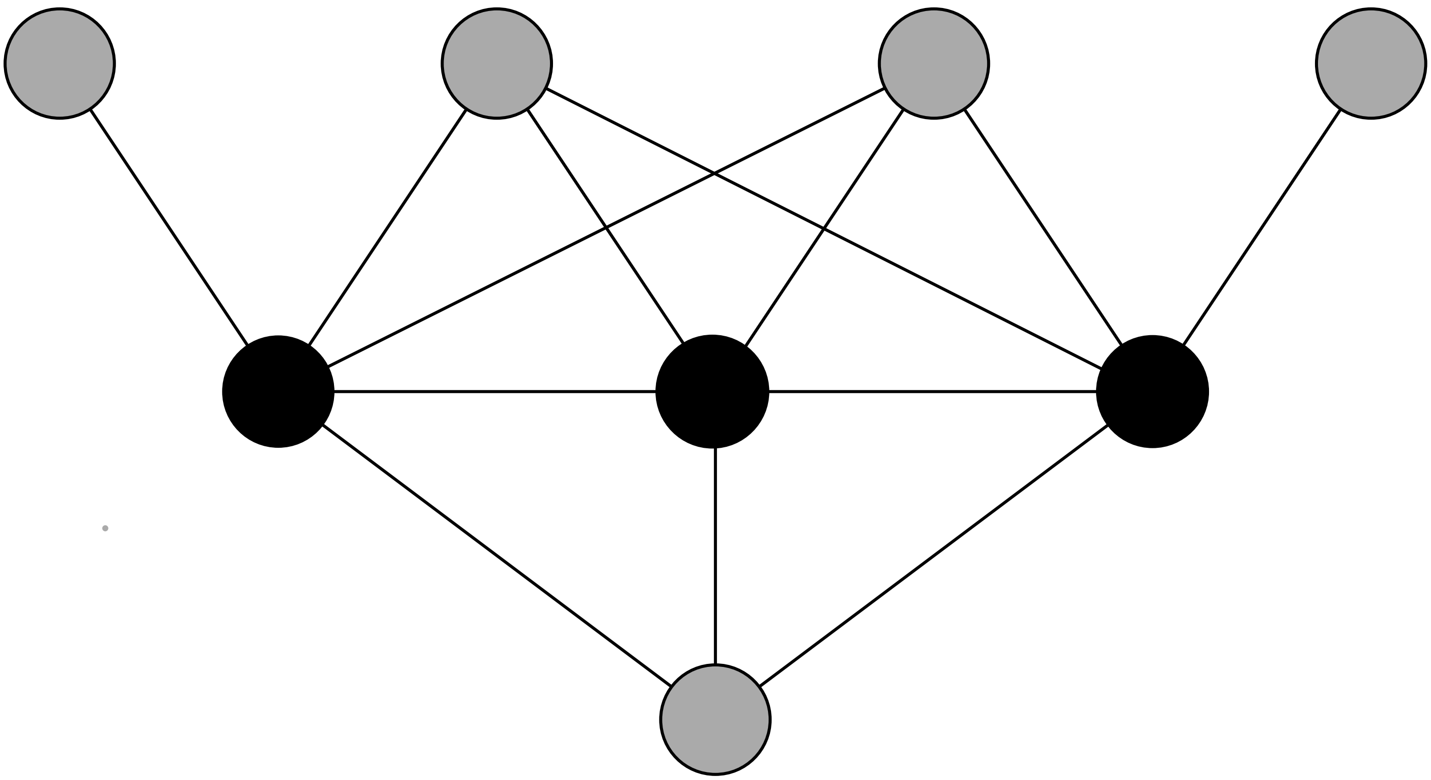
A possible solution



- Only solution for this graph.
- How might we approximate a solution to vertex cover problem?

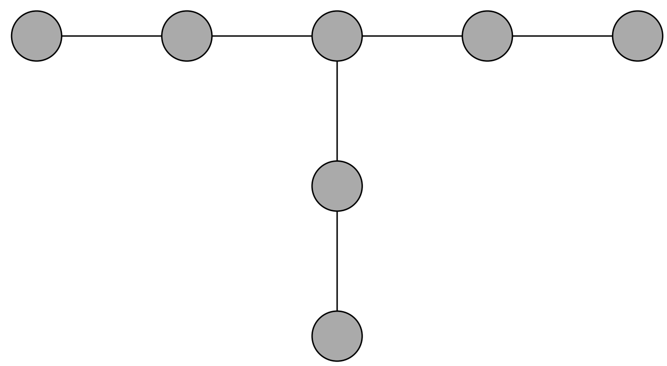
Idea

- Choose vertices of max degree.
- Works for previous example.

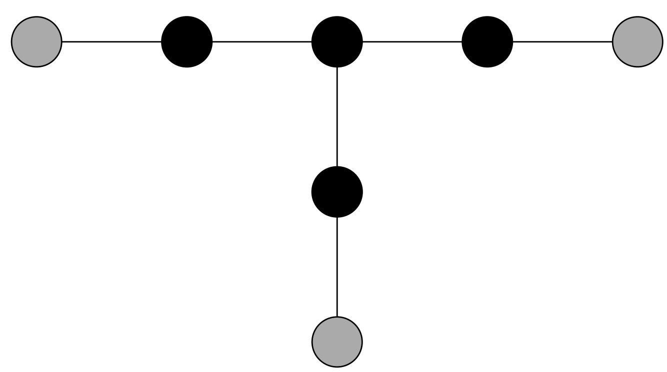


Problem

- What about the following graph?

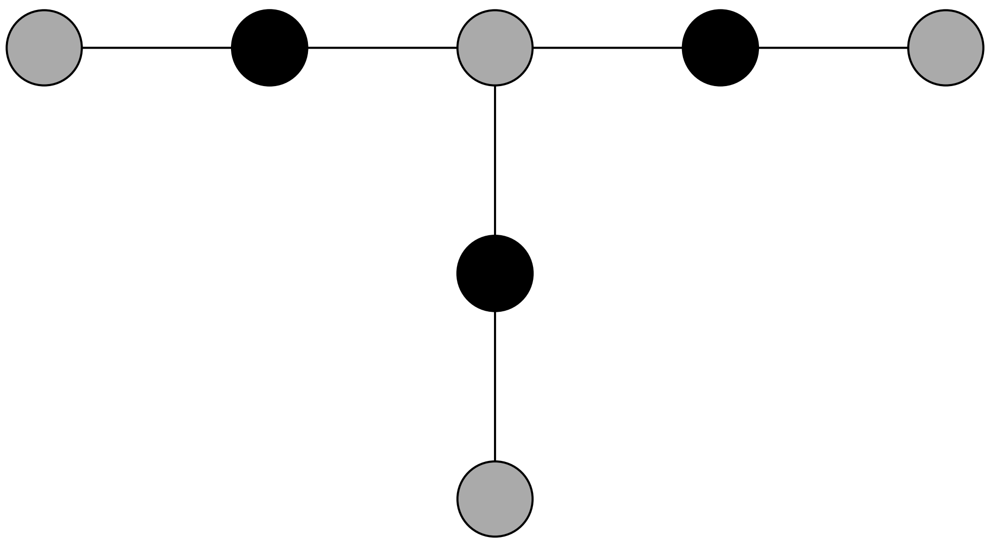


- max-degree strategy gives:



Problem

- Actual optimal solution is:

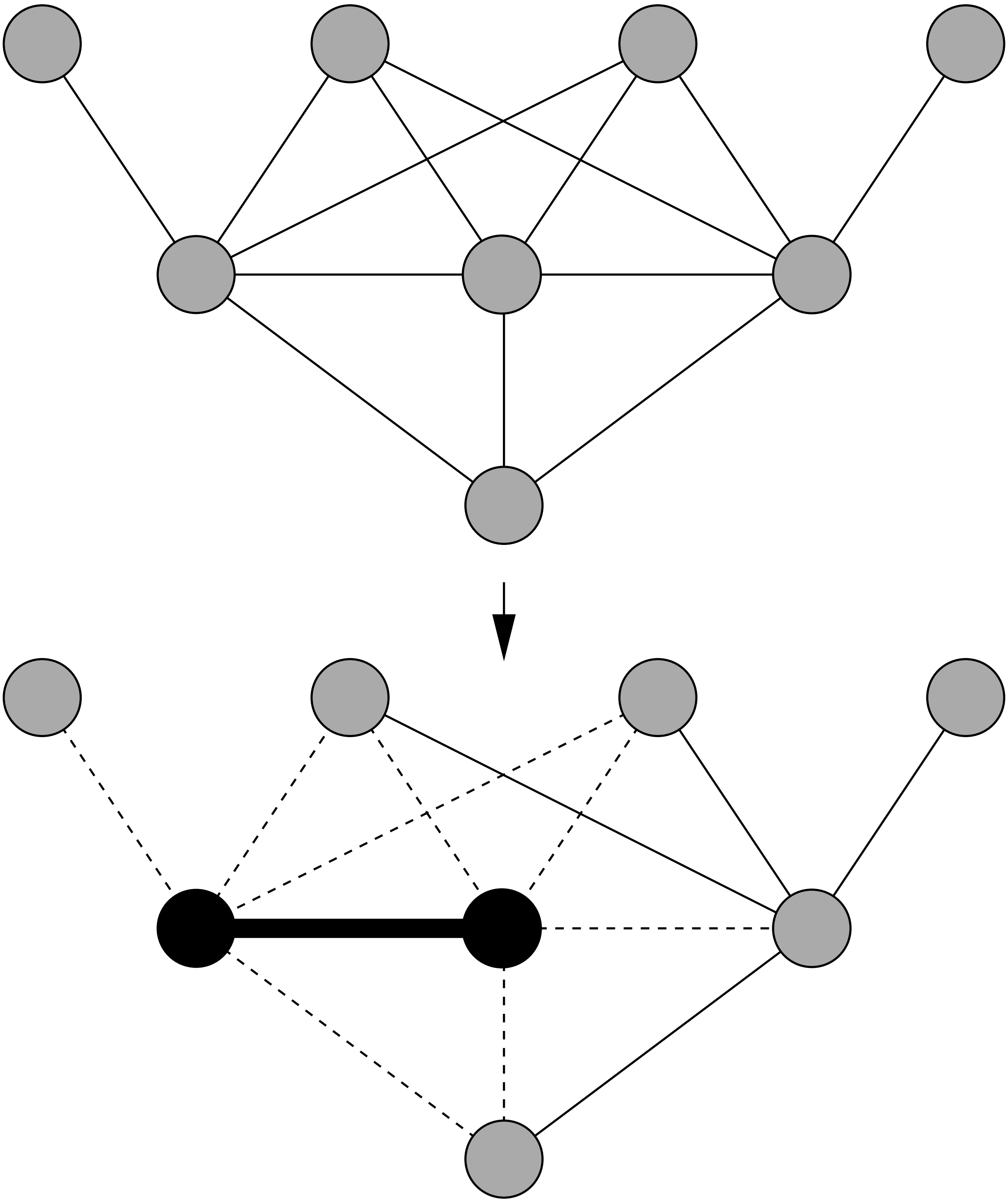


- Is there a better approximation alg.?

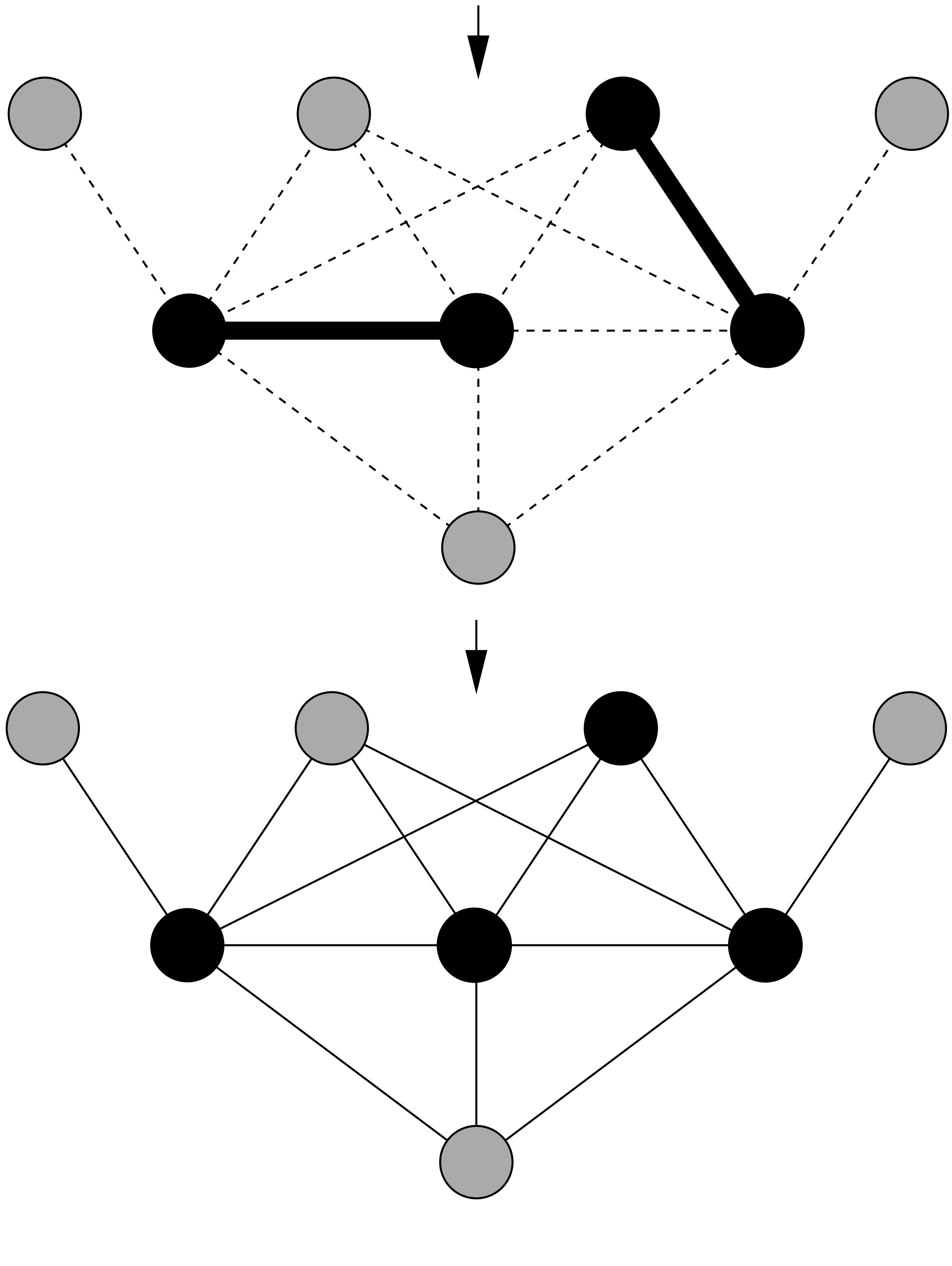
Approximation Algorithm

```
APPROX-VERTEX-COVER(G)
C ← ∅          ▷ C to be cover
E' ← E[G]
while E' ≠ ∅
      do let (u, v) be an arbitrary edge of E'
         C ← C ∪ {u, v}
         remove from E' every edge incident on
            either u or v
return C
```

Vertex Cover Approximation Example



Vertex Cover Approximation Example Cont.



Analysis of Vertex Cover Approximation

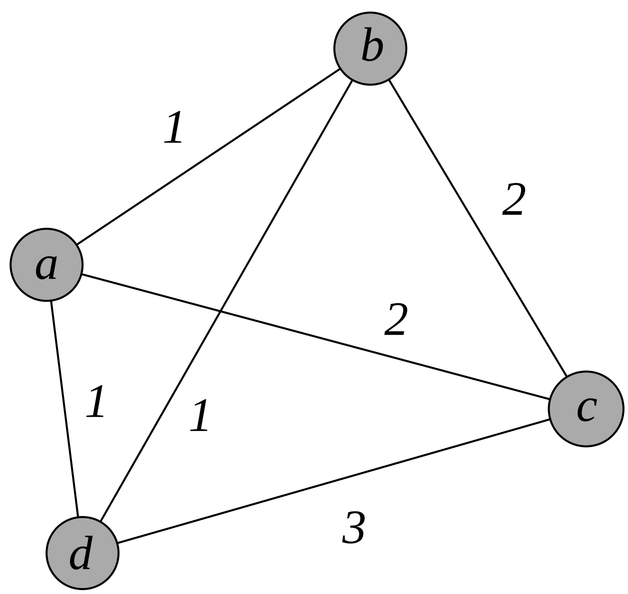
- **Correctness**
  - Only remove "covered" edges from E'.
  - APPROX-VERTEX-COVER returns a vertex cover.
- **Running Time** is $O(|V| + |E|)$.

Further Analysis

- **Theorem** APPROX-VERTEX-COVER has ratio bound 2.
- **Proof**
  - $A = \{$edges picked in line 4$\}$ (A is a set).
  - No two edges in A share an endpoint.
  - $|C| = 2\,|A|$.
  - Optimal cover, C^* must include at least one endpoint for each edge in A.
  - $|A| \leq |C^*|$.
  - Conclude that $|C| \leq 2\,|C^*|$;
    that is, size of approximate cover is
    at worst twice size of optimal cover!

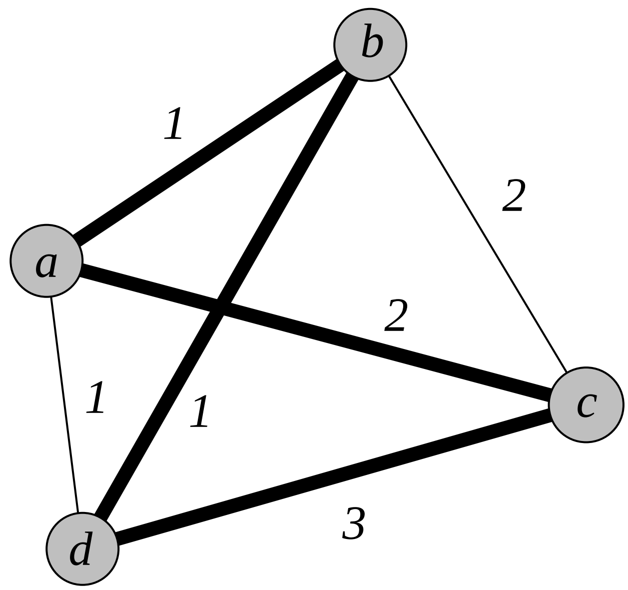
Traveling-Salesman Problem

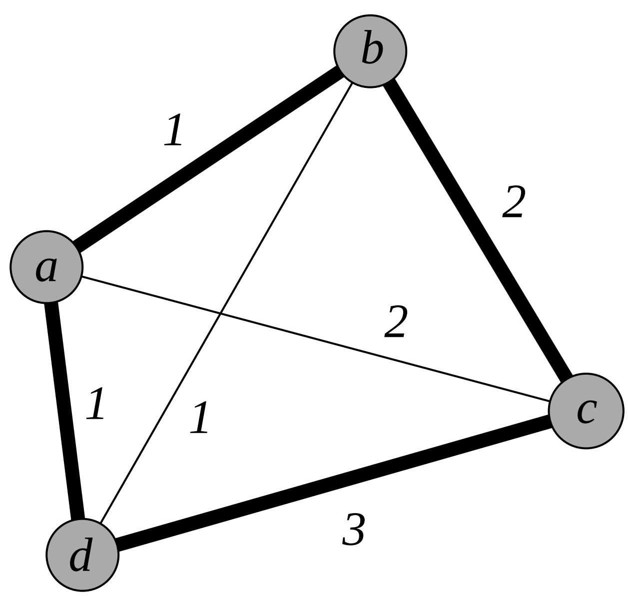
- Given: **complete** undirected graph $G = (V, E)$.
- Each edge $(u, v) \in E$ has integer cost $c(u, v)$.
- Each path has an associated cost.



Traveling-Salesman Problem (TSP):
Optimization: find min-cost hamilt. cycle of G; i.e., a min-cost cycle visiting each vertex exactly once.
Decision: NP-complete (reduction from HAM-CYCLE, see CLRS).

Examples



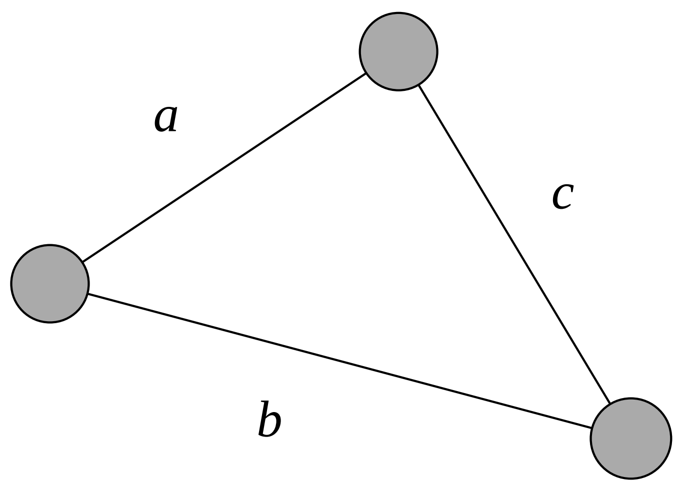


TSP Approximation Algorithm

Suppose weights satisfy **triangle inequality:**

$$c(u, w) \leq c(u, v) + c(v, w)$$

for all $u, v, w \in V$



$a + b \leq c$

TSP is still NP-complete! However...

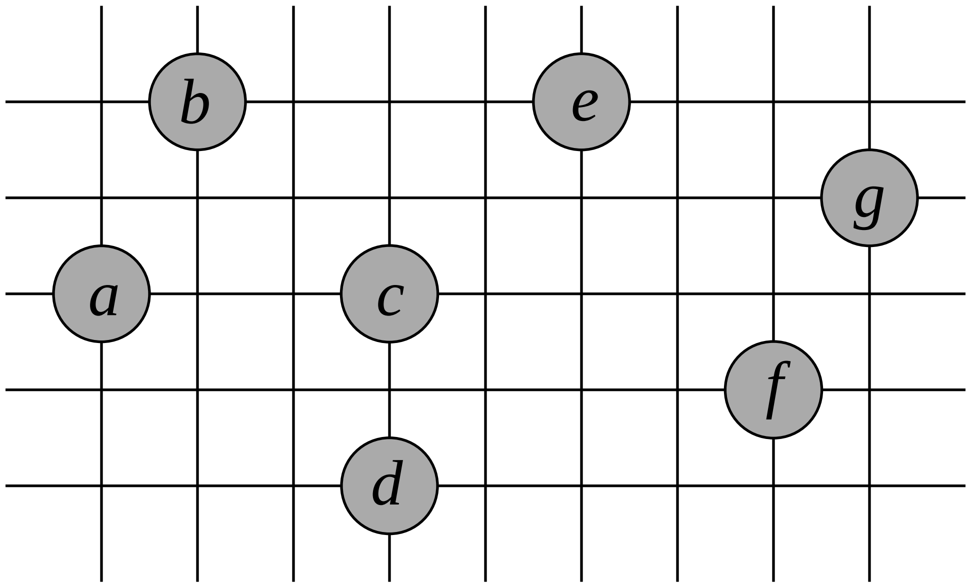
TSP Approximation Algorithm

APPROX-TSP-TOUR(G, c)

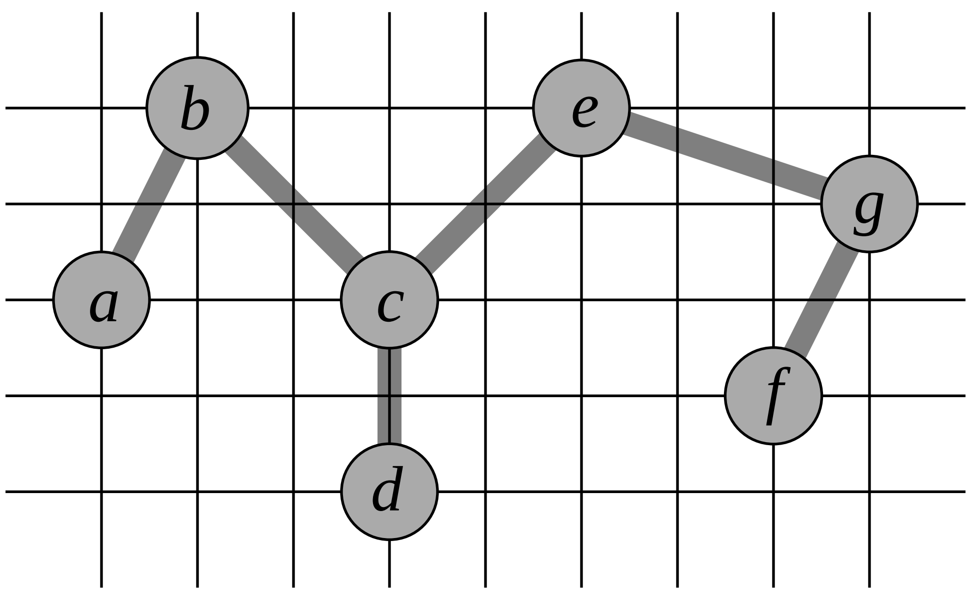
select a vertex $r \in V[G]$ to be a "root" vertex
grow minimum spanning tree T for G from
        root r using MST-PRIM(G, c, r)
let L be the list of vertices visited in
        preorder tree walk of T
return hamiltonian cycle H that visits
        vertices in the order L

Example

- Find shortest tour for:

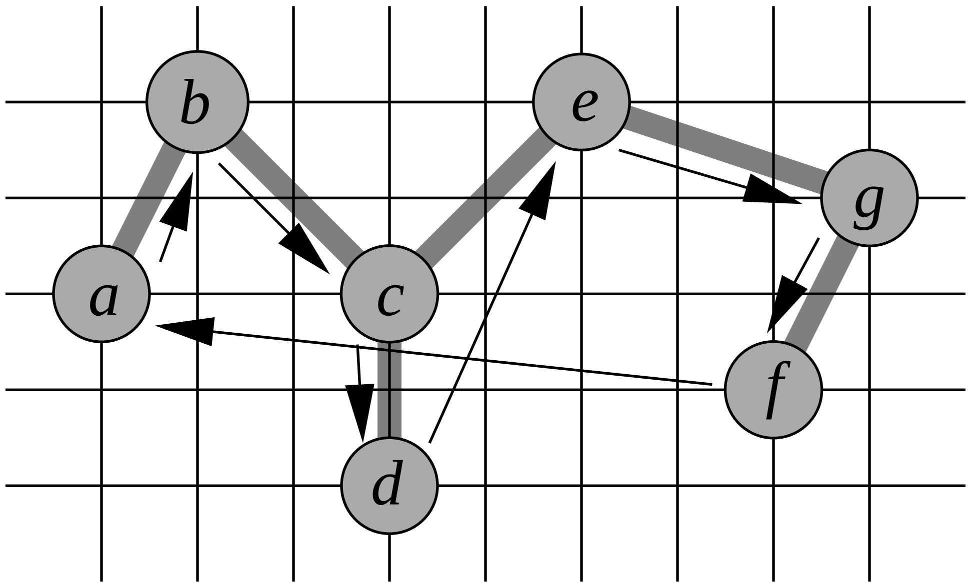


- Find MST (with root a).

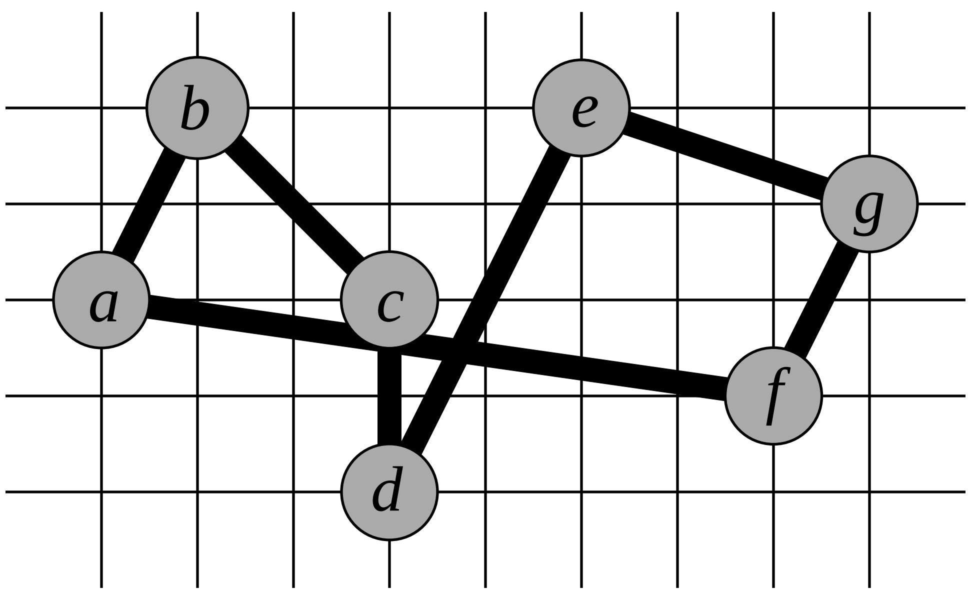


Example

- Pre-order walk of MST (node first, then children)

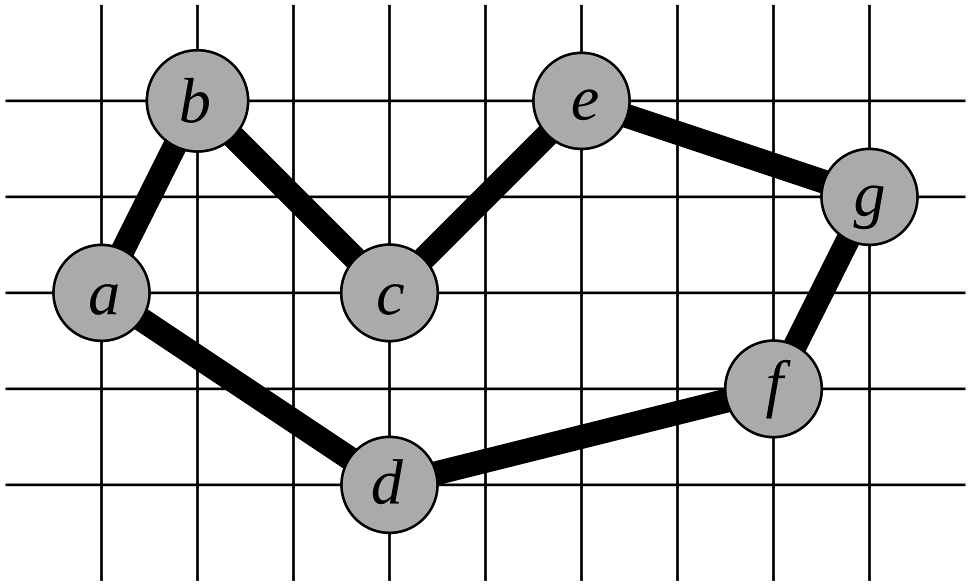


- Yielding tour:



- Total distance $\approx$ 24.00 units.

Optimal Solution



- Total distance $\approx$ 20.44 units.

- **Theorem:** Approx-TSP-Tour with triangle inequality has ratio bound 2.
- **Proof:**
  - $H^* =$ optimal tour for G.
  - T is a MST for $G \rightarrow c(T) \leq c(H^*)$.
  - $W =$ full walk of T. $c(W) = 2c(T)$.
  - $c(W) \leq 2c(H^*)$.
  - H is preorder walk of T. By triangle inequality, $c(H) \leq c(W)$ [why?]
  - $c(H) \leq 2c(H^*)$

Best ratio was $\frac{3}{2}$ for long time; now ϵ.

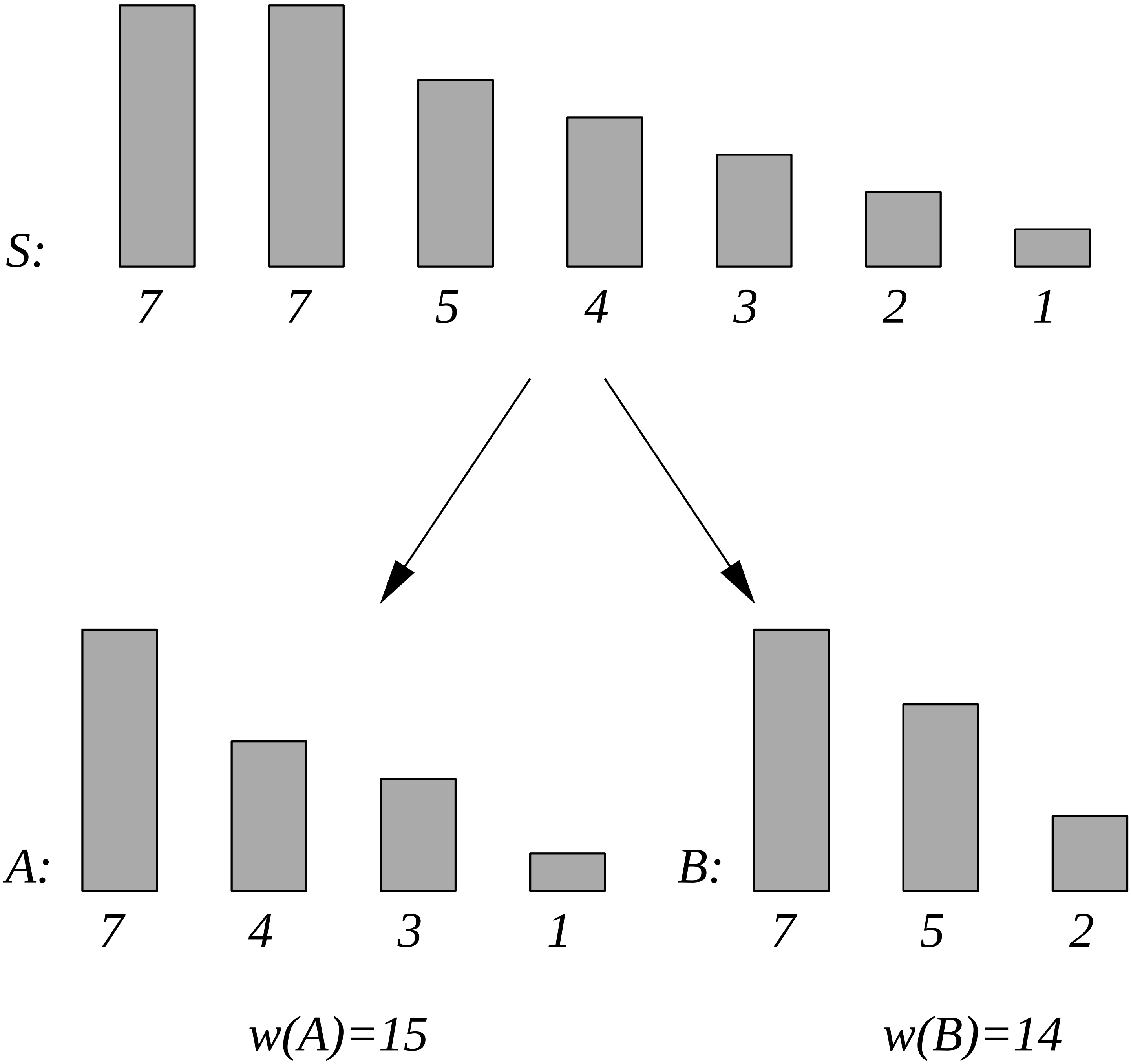
ϵ-Approximation Schemes

- Input of size n and relative error bound $\epsilon > 0$.
- Returns solution with $\frac{|C-C^*|}{C^*} < \epsilon$.
- **Polynomial-time Approximation Scheme**
  - $O(n^{O(1)})$ time for any constant ϵ.
- Fully Polynomial Time Approximation Scheme
  - Polynomial in both n and $1/\epsilon$ (see CLRS)

Partition Problem

- Given:
  - $S = \{a_1, a_2, \ldots, a_n\}$
  - $a_1 \geq a_2 \geq \ldots \geq a_n$
- **Problem:** Partition S into $A \cup B$ such that $\max(w(A), w(B))$ is minimized.
- NP-Complete (reduction from 3D matching).
- Can we find a polynomial-time approximation scheme?

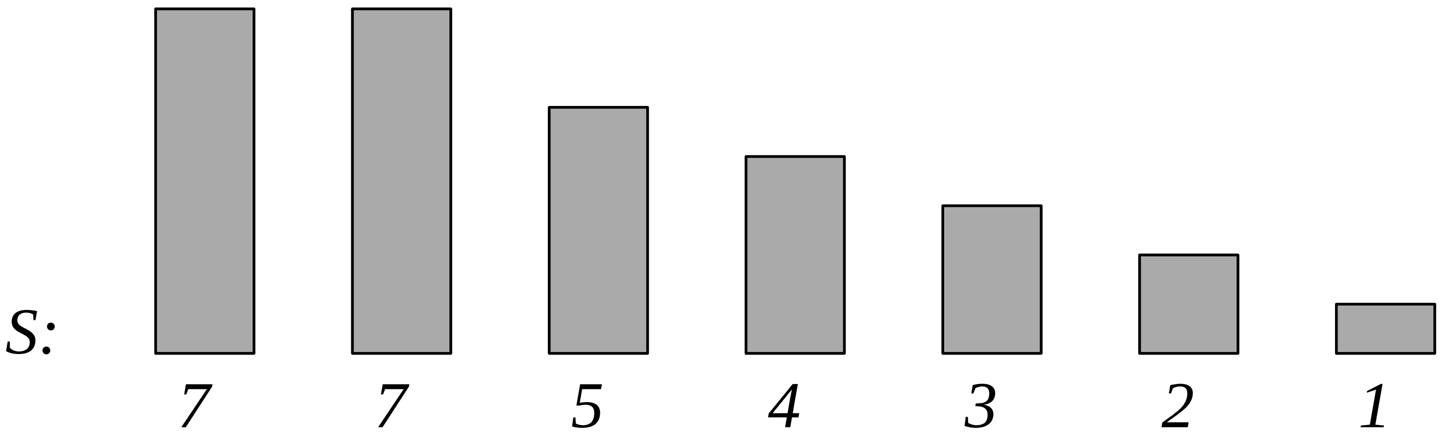
Example



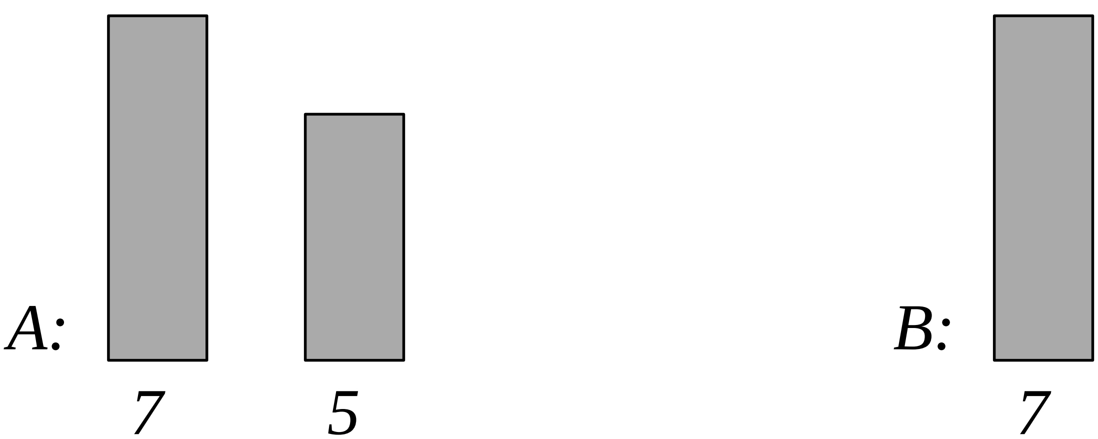
Approximation Scheme

- Let $m = \lfloor 1/\epsilon \rfloor$
- Find optimal partition of $S' = \{a_1, a_2, \ldots, a_m\}$ by exhaustive enumeration.
- Consider $a_{m+1}, a_{m+2}, \ldots, a_n$ in turn and add to currently lighter set.

Example



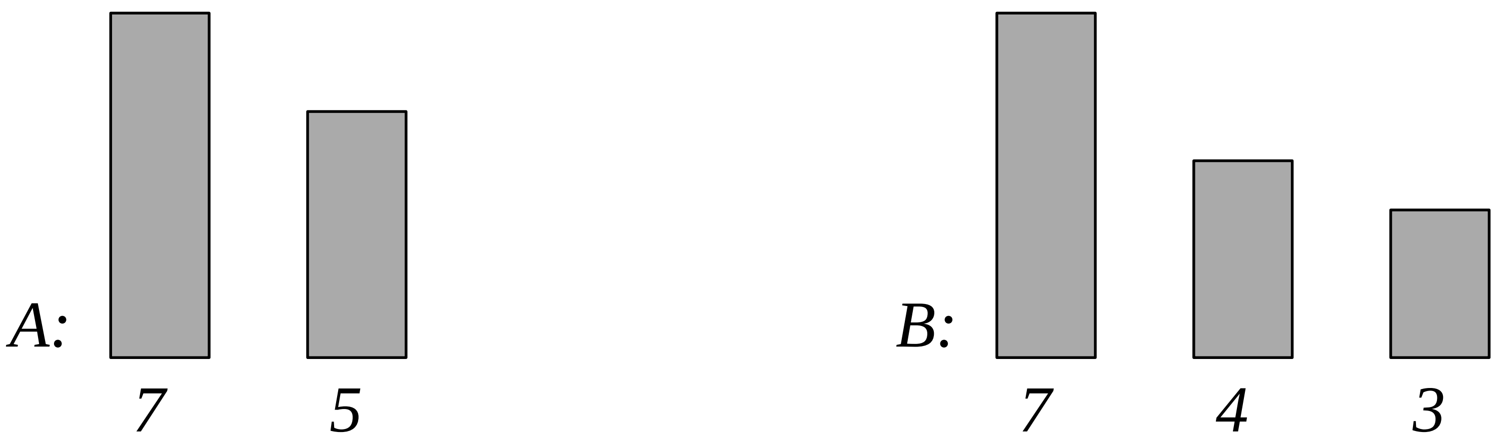
- $\epsilon = 1/3$
- $m = 3$
- Partition $\{7, 7, 5\}$



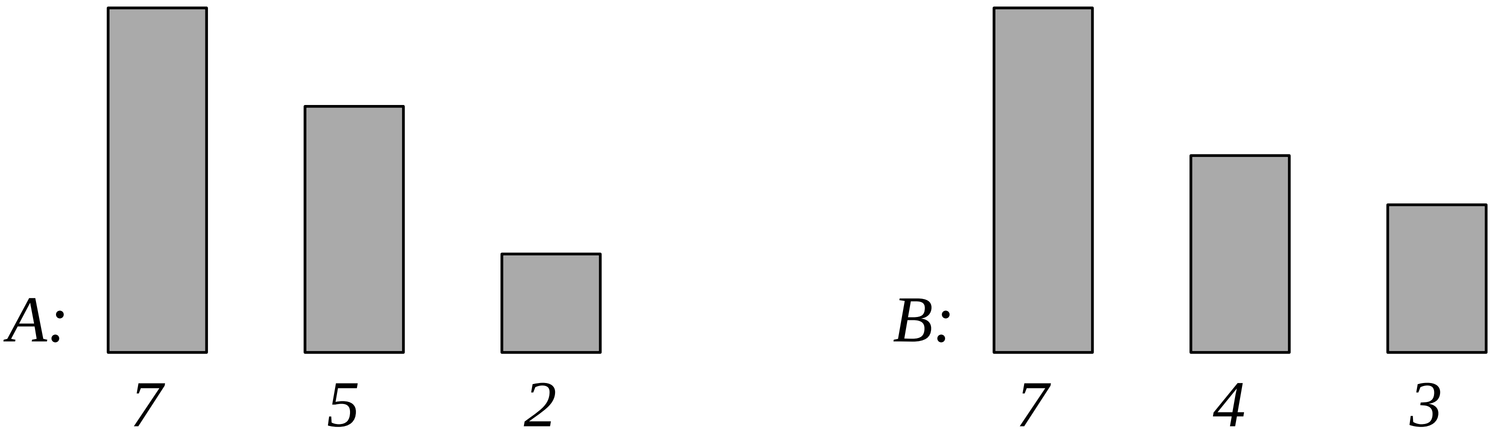
Example Cont.

- Insert 4.

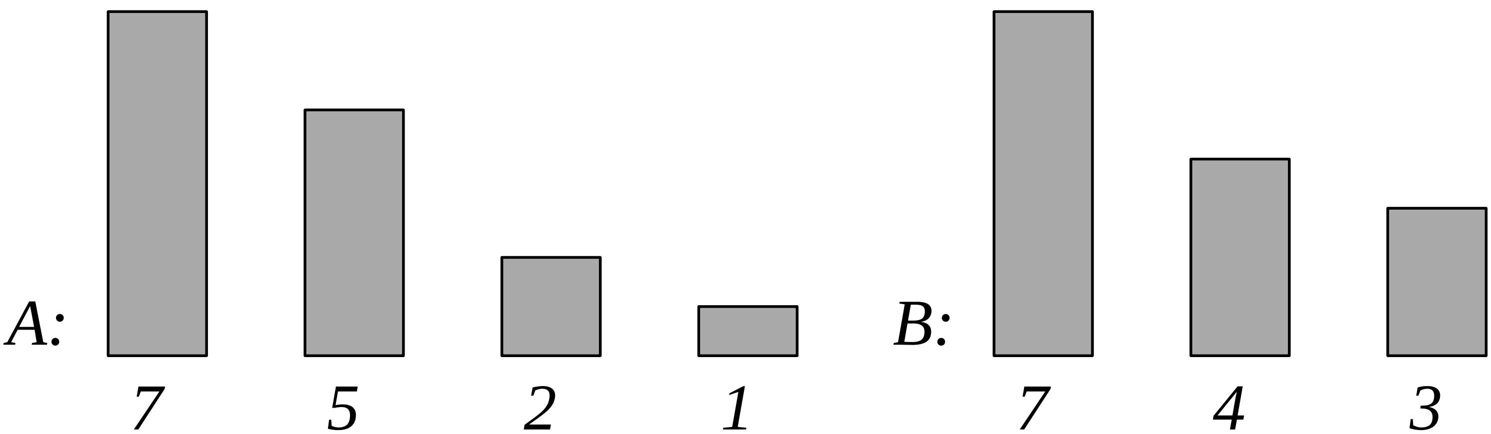
- Insert 3.



- Insert 2.



- Insert 1.



- $w(A) = 15$, $w(B) = 14$.
- What is the running time for this algorithm?

Running Time

- Finding optimal partition of S' takes $O(2^m)$ time.
- Considering each of the remaining elements of S takes $O(n)$ time.
- Total running time is

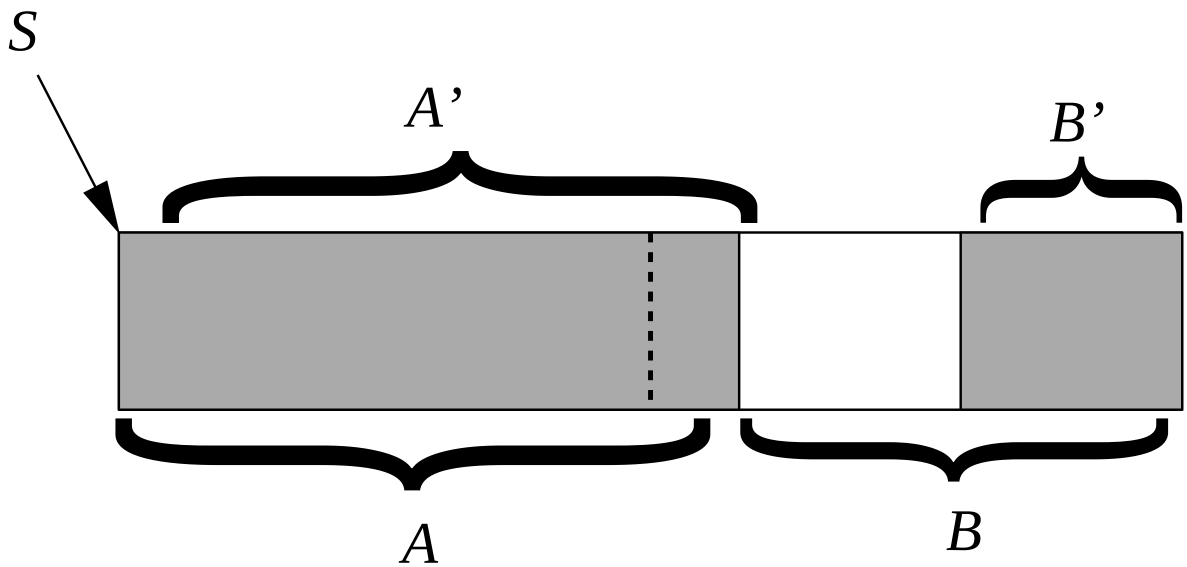
$$\begin{aligned} O(2^m + n) &= O(2^{1/\epsilon} + n) \\ &= O(n) \text{ for constant } \epsilon \end{aligned}$$

Theorem: Partition produced by approximation scheme has relative error $< \epsilon$.

Proof:

- Let $A' \cup B'$ be an optimal partition of S'.
- Assume $w(A') \geq w(B')$.

- **Case 1**
  - $w(A') \geq \frac{1}{2}w(S)$



  - Then $A = A'$, $B = B' \cup \{m+1, m+2, \ldots, n\}$.

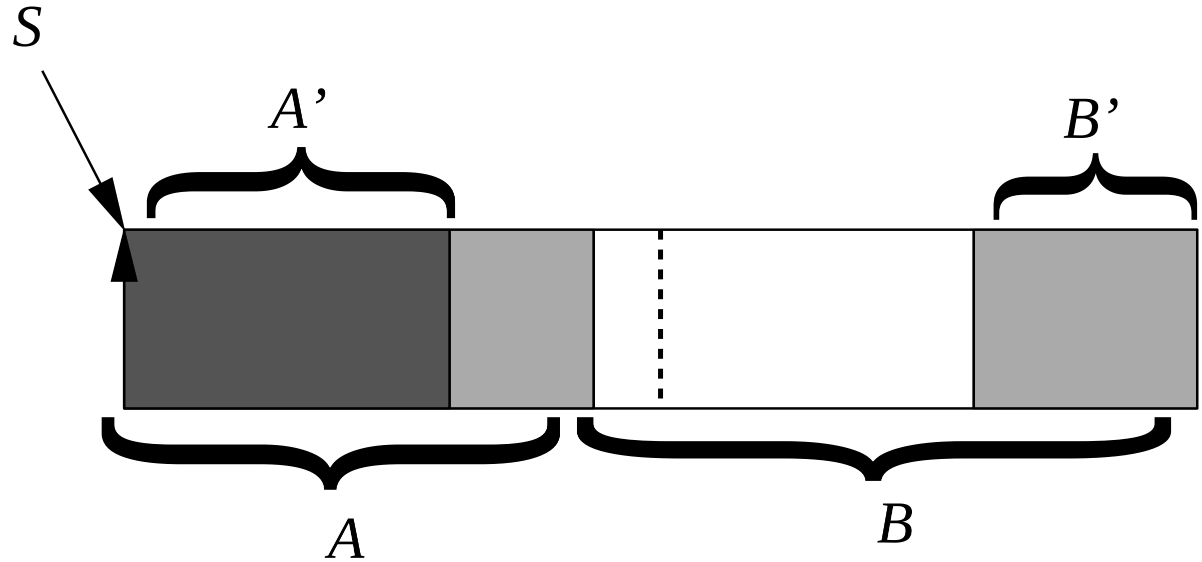
– Claim: $A \cup B$ is optimal (Relative error = 0)

* Consider optimal solution $A^* \cup B^* = S$.
* $w(A^*) \geq w(A^* \cap \{a_1, a_2, \dots, a_m\})$ [why?]
* $w(B^*) \geq w(B^* \cap \{a_1, a_2, \dots, a_m\})$
* Therefore,

$$\begin{aligned} \max\left(w(A^*), w(B^*)\right) &\geq \max\left(w(A'), w(B')\right) \\ &= w(A') \\ &= w(A). \end{aligned}$$

* Hence, $A \cup B$ is optimal.

- Case 2
  - $w(A') \leq \frac{1}{2}w(S)$



  - $|w(A) - w(B)| \leq w_{m+1}$
  - $w(A) + w(B) = w(S)$
  - $2\max\left(w(A), w(B)\right) \leq w(S) + w_{m+1}$

$$
\begin{aligned}
\text{Relative Error} &= \frac{C - C^*}{C^*} \\
&= \frac{\frac{w(S)+w(m+1)}{2} - \frac{w(S)}{2}}{\frac{w(S)}{2}} \\
&= \frac{w_{m+1}}{w(S)} \\
&\leq \frac{w_{m+1}}{(m+1)w_{m+1}} \\
&= \frac{1}{m+1} \\
&< \epsilon
\end{aligned}
$$

Bottom line: approx. alg. for partition which is poly-time and has relative error $< \epsilon$!