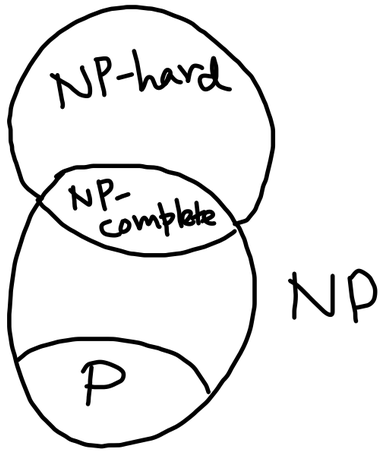
P, NP, NP-complete, NP-hard



∈ NP-hard but ∉ NP-complete

↳ Halting Problem

NP: Non-deterministic Polytime Algo (verification is possible in polytime)

NP-complete:

- NP ①
- Polynomial time Reducible to any prob in NP ②

NP-hard

↳ ②

Possible ways to deal with NP- complete Problems

1) Input size is very small.

2) Special sub-cases are of interest & these can be Solved in poly time.

3) Near- optimal Solution.

Approximation Algorithms

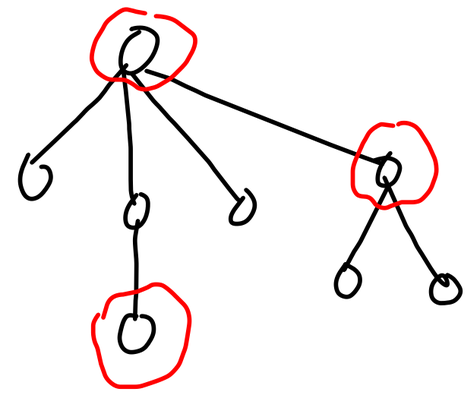
– Near-optimal solutions in poly time.

An algo for a problem has an approximation ratio $p(n)$, if for any i/p size n, the cost of the solution produced by the algo, say C, is within a factor of $p(n)$ of the cost C^* corresponding to the optimal solution.

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

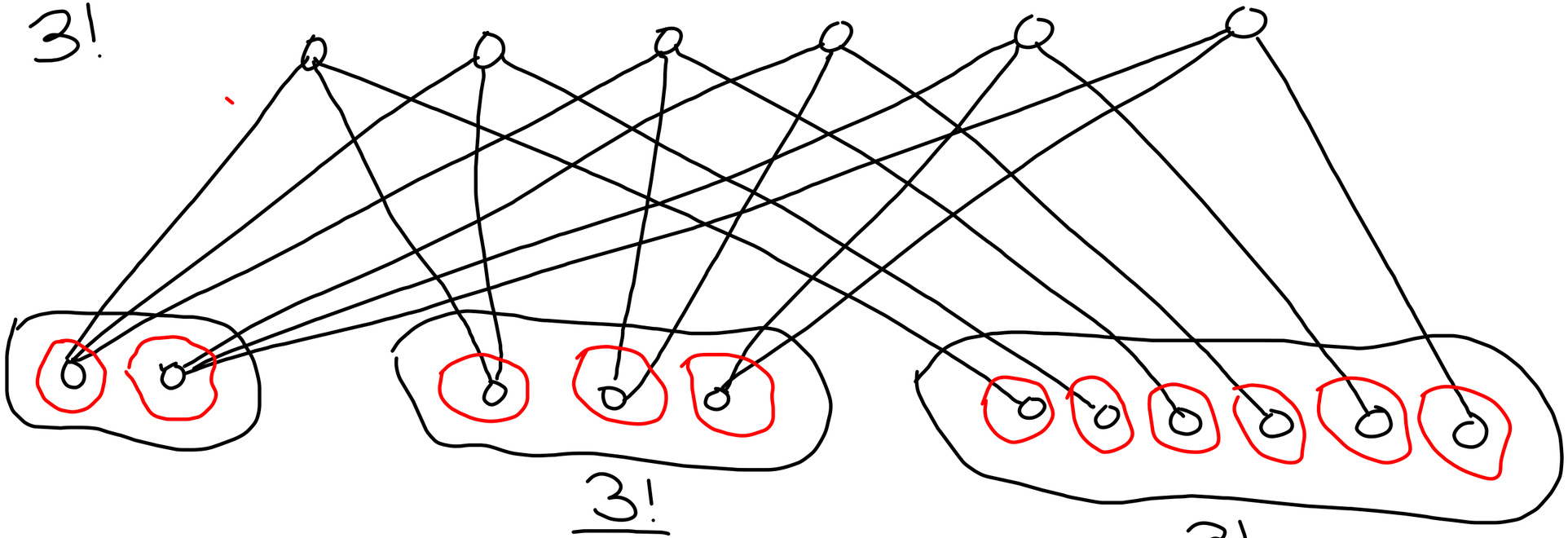
Vertex Cover

Given $G=(V,E)$. Return V' s.t. $V' \subseteq V$, $\forall (u,v) \in E$ either $u \in V'$ or $v \in V'$ or both & $|V'|$ is minimum among all.



Algorithm

- Obtain the max^{m} degree vertex $\hat{v}$. Put it in V'.
- Remove all the edges incident on v.
- Iteratively apply the same algo.



Vertex Cover → 6

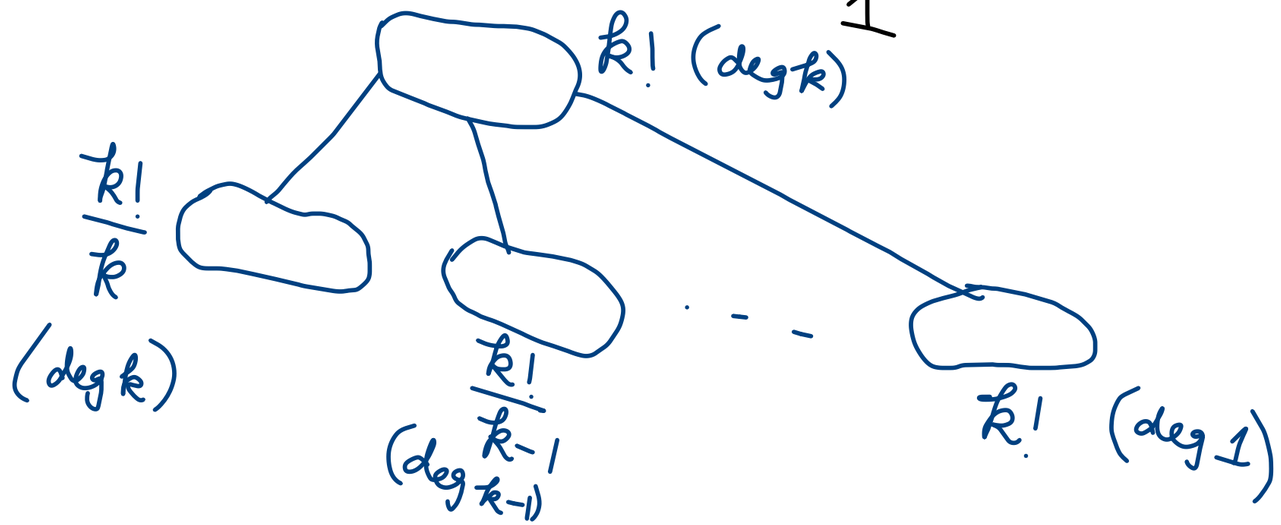
(11)

depends on n

$$= \frac{k!\left(1+\frac{1}{2}+\cdots+\frac{1}{k}\right)}{k!}$$

$$= \log k$$

$$\approx \log \log n$$



Vertex Cover . 2-Approx Algo

- $V' \leftarrow \phi$
- Choose $e=(u,v) \in E$ (*)
- $V' = V' \cup \{u, v\}$
- Remove all edges incident on u & v from E. Let the update Graph is $G^* = (V^*, E^*)$
- Repeat the same algo for G^*. until $V^* = \phi$

$C \rightarrow$ # vertex returned by this algo.

$C^* \rightarrow$ # vertex returned by optimal algo

$A \rightarrow$ # edges chosen in (*)

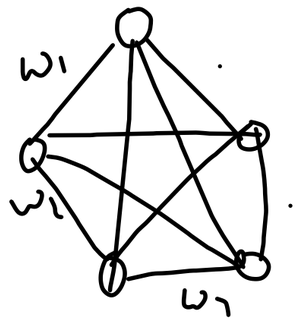
$C = 2A$ | $C^* \geq A$

$$C \leq 2C^*$$

The traveling Salesman Problem

(Following triangle inequality)

(All weights are (+)ve)



- Visit each node exactly once.
- Return to your starting point.
- Minimize the total cost & return the cost.

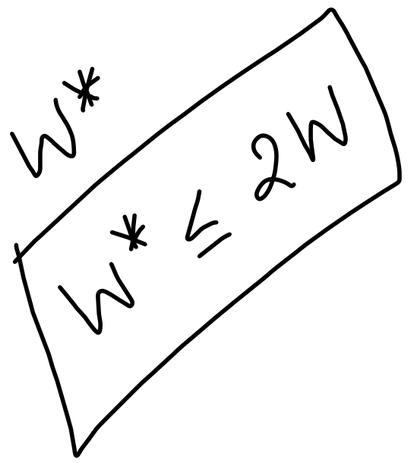
(Return a Walk for which the cost is minimum)

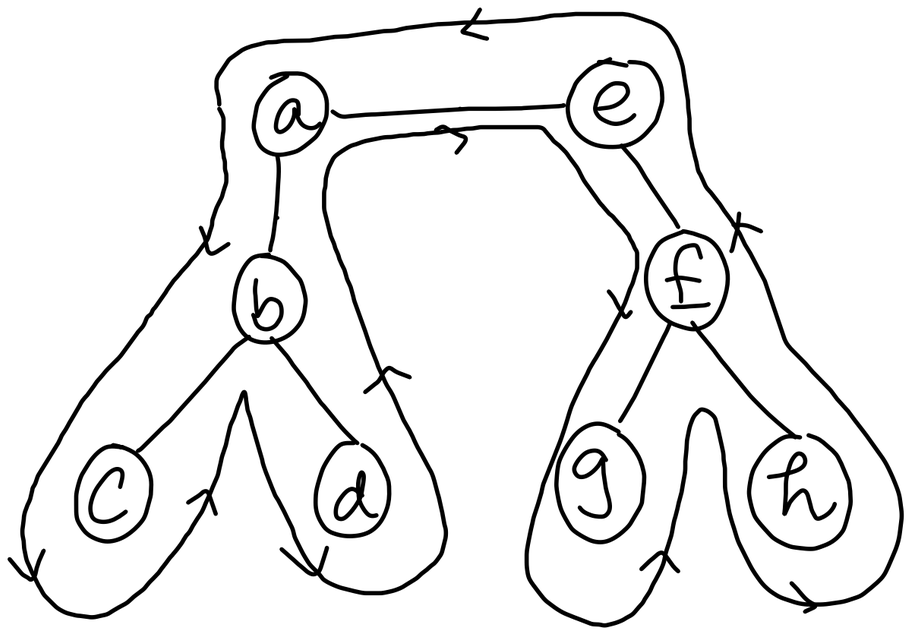
(W)

$$c(v_i, v_j) \leq c(v_i, v_k) + c(v_k, v_j)$$

Algorithm

└ Find Minimal Spanning tree





Return the cost of this Walk. (W)

Weights are given on the edges

Walk → W

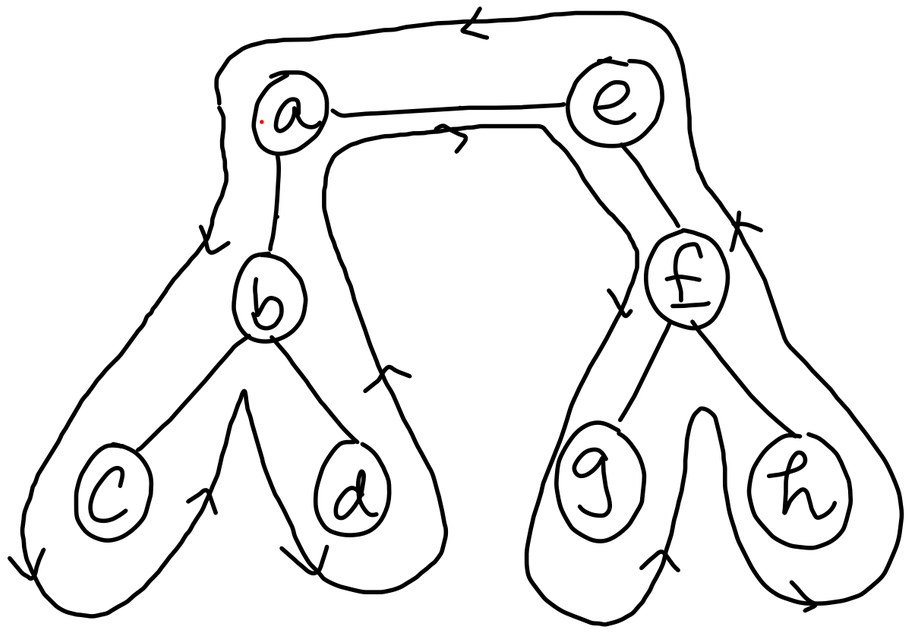
a → b → c → b → d → b → a → e → f → g → f → h → f → e → a

W^* → optimal tour
T → MST
W → Walk through T

$c(W) = 2\,c(T)$

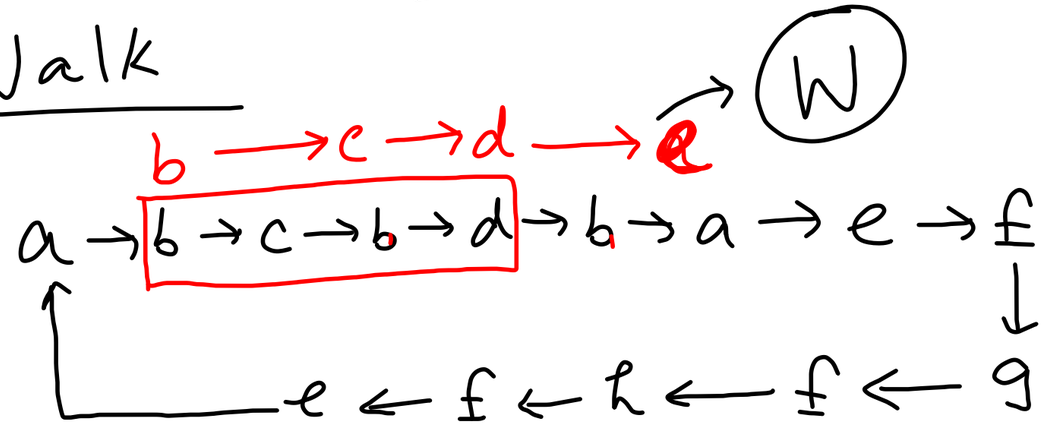
$c(T) < c(W^*)$

$$c(W) < 2\,c(W^*)$$



Weights are given on the edges

Walk



W* → optimal tour
T → MST
W → Walk through T

$c(W) = 2\,c(T)$

$c(T) < c(W^*)$

$c(W) < 2\,c(W^*)$

{ Return the cost of this Walk. (W)

$a \to b \to c \to d \to e \to f \to g \to h \to a$

New walk using triangle inequality.

ϵ - parameter

↓

Approximation factor = $(1+\epsilon)$

Algo → $O(n^{\frac{1}{\epsilon}})$

PTAS

FPTAS

Is every AVL tree is Red Black?