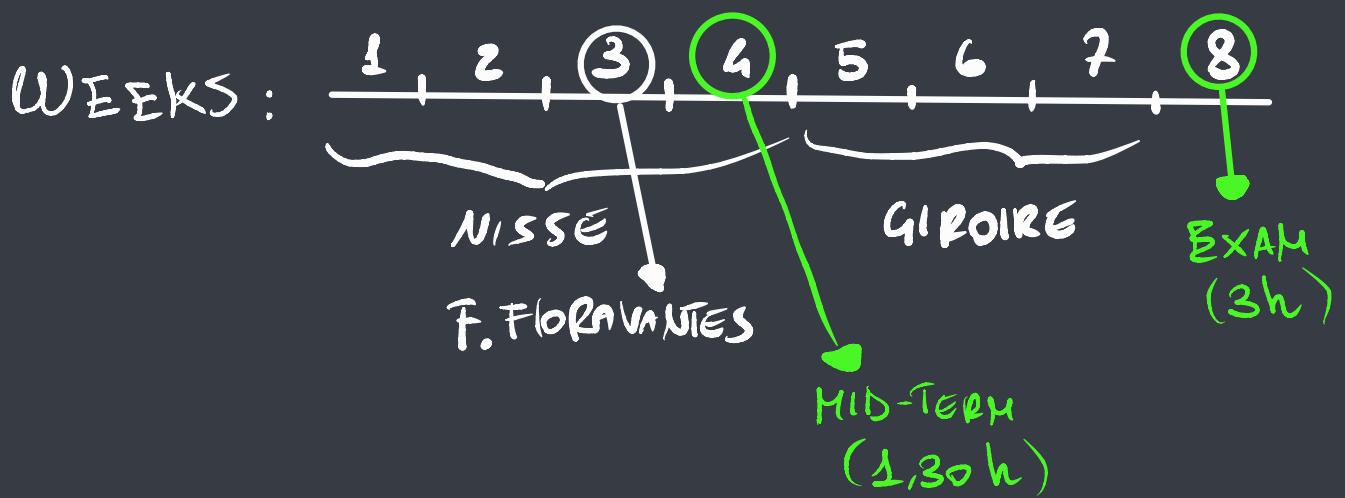


Graph Algos & Comb. Opt



2021



INTRO TO GRAPHS

TH

- $T = (V, E)$ is a tree $\Leftrightarrow T$ is CONNECTED and $|V| = |E| + 1$

PROOF (\Leftarrow)

- (RIC) T is a TREE if CONNECTED and NO CYCLES
- Suppose T not a tree $\Rightarrow \exists$ cycle (v_1, \dots, v_k) \Rightarrow we can so remove $\{v_i, v_j\} \in E$ obtain T' connected
- $|E'| = |E| - 1 = |V| - 2 = |V'| - 2 < |V| - 1$
- $\Rightarrow |E'| < |V'| - 1 \Rightarrow |V'| > |E'| + 1 \perp \Rightarrow T$ NOT connected

(\Rightarrow) . INDUCTION ON $|V|$

- PART 1
- Let $P = (v_1, \dots, v_k)$ be the longest PATH in T
 - Let v_1 be a LEAF
 - suppose that $\deg(v_1) > 1$ AND $\exists x \in N(v_1) \setminus \{v_2\}$
 - $\Rightarrow x \notin P$ otherwise would create a cycle
 - \Rightarrow so there should be $P' = (x, v_1, \dots, v_k)$ but would be longer than $P \perp$

(cause it's connected!)

- PART 2
- Let $S = T \setminus \{v_1\}$ be a subtree
 - we know $|V_S| = |E_S| + 1$
 - $|V| = |V_S| + 1$ AND $|E| = |E_S| + 1$

$$\Rightarrow \begin{cases} |V| = |V_S| + 1 \\ |E| = |E_S| + 1 \\ |V_S| = |E_S| + 1 \end{cases} \quad \left\{ \begin{array}{l} |V_S| = |V| - 1 \\ |E_S| = |E| - 1 \\ |V| - 1 = |E| - x + 1 \end{array} \right. \Rightarrow |V| = |E| + 1$$



CONNECTIVITY AND MISTANCES

LENGTH of PATHS

- sum of weights of arcs in PATH

DISTANCE

- minimum length from among (s,t) -PATH

BFS (UNWEIGHTED GRAPHS)

- INPUT $G = (V, E)$, $v \in V$
- OUTPUT $d(v) = d = (v, v) \forall v \in V$

ALGO

Horizon →
is a queue

$$d(v) = \emptyset, d(v) = \infty \forall v \in V \setminus \{v\}$$

$$\text{Horizon} = \{v\}$$

$$\text{Explored} = \{\emptyset\}$$

$$T = (V_T, E_T) = (\{v\}, \{\emptyset\})$$

WHILE ($\text{Horizon} \neq \{\emptyset\}$)

$$v = \text{head}(\text{Horizon})$$

FOR ($u \in N(v) \setminus \{\text{Horizon} \cup \text{Explored}\}$)

$$d(u) = d(v) + 1$$

$$V_T = V_T \cup u$$

$$E_T = E_T \cup (u, v)$$

add $l(v, u)$ to $(\text{Horizon}, u)$

$$\text{Horizon} = \text{Horizon} \setminus \{v\}$$

$$\text{Explored} = \text{Explored} \cup \{v\}$$

COMPLEXITY

$O(|E|)$

NOTE 1
 T is the
shortest path
tree rooted
in v

NOTE 2
We can check connectivity
 G connected \Leftrightarrow
 $d(v, v) < \infty \forall v \in V$

DIAMETER OF A GRAPH

INPUT } UNWEIGHTED TREE $T = (V, E)$
 $v \in V$

OUTPUT $\text{diam}(G) = \max_{\forall u, v \in V} d(v, u)$

ALGO

- $\text{BFS}(v)$

- Let u be such that $d(v, u) = \max_{v \in V} d(v, u)$

- $\text{BFS}(u)$

- Let w be such that $d(u, w) = \max_{v \in V} d(u, v)$

RETURN $\text{diam}(G) = d(u, w)$

COMPLEXITY
2 BFS

DIKSTRA (WEIGHTED GRAPHS)

NOTE

BFS doesn't work on WEIGHTED GRAPHS

INPUT $G = (V, E)$ **OUTPUT** $d(v) = d(v, v) \forall v \in V$
 w positive weights
 $v \in V$

ALGO

$d(v) = 0, d(v) = \infty \forall v \in V \setminus \{v\}$
 $T = (V_T, E_T) = (\{\emptyset\}, \{\emptyset\})$

$E_{\text{EXPLORED}} = \{\emptyset\}$

$\text{PARENT}(v) = \{\emptyset\} \forall v \in V$

WHILE ($E_{\text{EXPLORED}} \neq V$)

COMPLEXITY WITH MIN HEAP
 $O(|E| + |V| \log |V|)$

Let $v \in V \setminus E_{\text{EXPLORED}}$ be the one with min. $d(v)$

$V_T = V_T \cup \{v\}$

$E_T = E_T \cup \{(v, \text{parent}(v))\}$

FOR ($u \in N(v) \setminus E_{\text{EXPLORED}}$)

IF ($d(u) > d(v) + w_{vu}$)

$d(u) = d(v) + w_{vu}$

$\text{parent}(u) = v$

NOTE 1
 T is the shortest path tree rooted in v

NOTE 2
 We can check connectivity
 G connected $\Leftrightarrow d(v, v) < \infty \forall v \in V$

TERMINATION

• After i^{th} iteration $|\text{EXPLORER}| = i$ so it terminates in $|V|$ iterations

CORRECTNESS

• $d(v)$ is optimal when v is added to EXPLORER

• PROOF (by induction on steps.)

• BASE : For $i=1 \Rightarrow d(v) = \text{BPD}(v) = \emptyset$ ✓

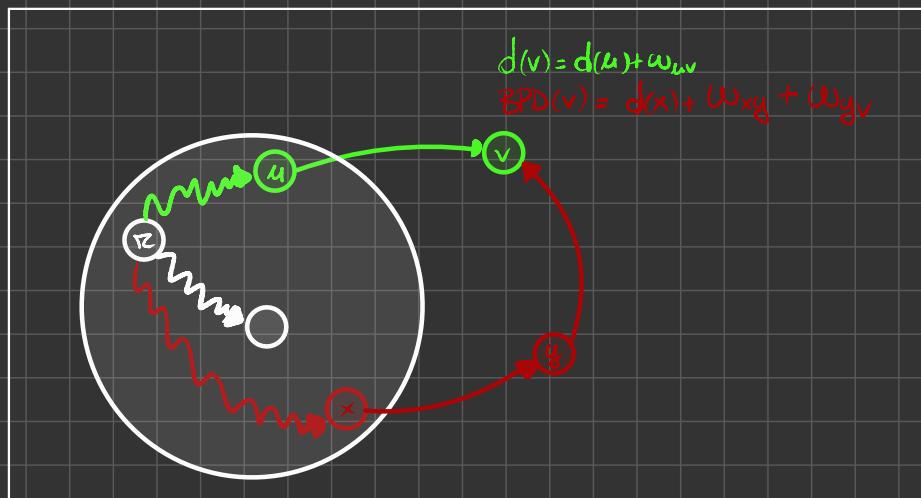
• Assume , at the start of i^{th} STEP, $d(u) = \text{BPD}(u)$ True EXPLORER

• Let $d(v) = d(u) + w_{uv}$ the best choice of i^{th} step

* • it's TENACIOUS that, at step i , $d(v) < d(z)$ if z not yet explored
by the GREEDY choice of the algo

• Assume now that there's a shortest path $P = (r, \dots, x, g, v)$

of length $\text{BPD}(v)$ | $\text{BPD}(v) < d(v)$



• So

$$\left\{ \begin{array}{l} ① d(v) > \text{BPD}(v) \\ ② d(x) + w_{xy} \leq \text{BPD}(v) \\ ③ d(y) \leq d(x) + w_{xy} \\ ④ d(y) \geq d(v) \end{array} \right. \quad \left\{ \begin{array}{l} ① d(v) > \text{BPD}(v) \\ ② d(y) \leq \text{BPD}(v) \\ ③ d(y) \geq d(v) \end{array} \right. \quad \left\{ \begin{array}{l} ①, (2,3) \quad d(y) \leq \text{BPD}(v) < d(v) \\ ④ \quad d(y) \geq d(v) \end{array} \right.$$

$$\Rightarrow \left\{ \begin{array}{l} d(y) < d(v) \\ d(y) \geq d(v) \end{array} \right. \Rightarrow d(v) \leq d(y) < d(v) \Rightarrow d(v) < d(v)$$



KRUSKAL (MST)

INPUT $G = (V, E)$ CONNECTED
 $\{w\}$ positive weight

OUTPUT MST $T = (V_T, E_T)$

ALGO

$$T = (V, \{\emptyset\})$$

Let (e_1, \dots, e_m) NON-DECRL. order of edges

FOR ($i=1, \dots, m$)

IF ($E_T \cup e_i$ NOT create CYCLE)

$$E_T = E_T \cup e_i$$

COMPLEXITY
 $O(|E| \log |E|)$

TERMINATE

• OBV.

CORRECT

• TH The spanning tree T returned is a MST

• Proof

• T is obr. a SPANNING TREE, because G is connected, and T is acyclic by def.

• Suppose by contradd. T NOT MST

• So you have $e_j \in E$ such that by adding it to E_T , and removing another edge you get $T' \mid w(T') < w(T)$

• By the ordering e_j , that has already been added to E_T , should be such that $w(e_j) < w(e_i)$ so you get a CONTRADICTION \perp



20/09/20

APPROXIMATION ALGO. FOR NP-HARD PROBLEM

MAX MATCHING (P)

MATCHING

- Given $G = (V, E)$ a matching in G is a partition of disjoint edges

$$M \subseteq E \mid e \cap f = \{\phi\}, \forall e, f \in M$$

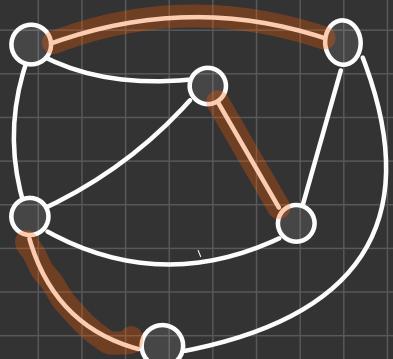
MAX MATCHING PRB.

- \exists Matching of size $\geq k \in \mathbb{N}$?

LEMMA

- $\forall G = (V, E)$ and any matching $M \Rightarrow |M| \leq \frac{|V|}{2}$

Proof



M-ALTERNATING PATH

- A path in G is M-ALTERNATING if for any 2 consecutive edges exactly one is in the matching M

AUGMENTING PATH

- $P = (e_1, \dots, e_k)$ AUGMENTING if P is M-ALTERNATING $\wedge e_1 \notin M \wedge e_k \notin M$



- If \exists AUGMENTING PATH $P \Rightarrow M$ is not MAX

Proof

- suppose M max

- Build $M' := \{e \in E \mid e \in P \wedge e \notin M\}$

- $|M'| = |M| + 1 \Rightarrow |M'| > |M| \Rightarrow M$ is not Max ↯

TM[BERGLO]

- A matching M in G is maximum $\Leftrightarrow \nexists$ an M-Augmenting Path P

Proof

(\Rightarrow) Look at TM done before.

(\Leftarrow) Let M be a matching without augmenting Path

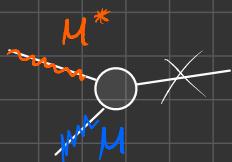
- Let M^* be a maximum matching

- Consider $A = M \setminus M^*$ and $B = M^* \setminus M$

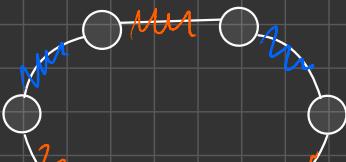
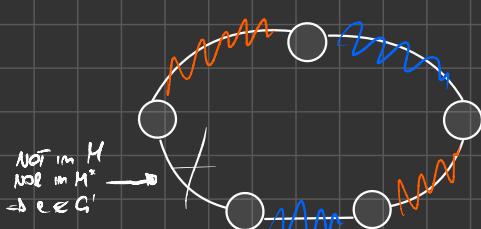
- Let $G' = (V, A \cup B)$ subgraph of G

- CLAIM every edge in G' has degree ≤ 2

Proof



- So in Graphs with $d \leq 2$ we have 2 cases:



$$\Rightarrow |M'| = |M|$$

□

ALGO

Find a matching M

WHILE (\exists M-augmenting path) // $m/2$ TCS

| AUGMENT M // $O(m^3)$ TO FIND AUG.?

[EDMONDS QS]

- We can find an M -AUGMENTING Path in Poly. time $O(m^3)$

\Rightarrow We can so COMPUTE MAX-MATCHING IN Poly Time

$$O(m^3) \cdot \frac{m}{2} = O(m^4)$$

COMPLEXITY

\sim BY LEMMA $|M| \leq \frac{m}{2}$

BIPARTITE
GRAPH

- $G = (V, E)$ is BIPARTITE if \exists a partition $V = V_1 + V_2$ | V_1, V_2 are STABLE (STABLE if no edges in same partition)

NOTE

- in BIP. GRAPH the maximum MATCHING can be solved efficiently

VERTEX COVER (NP COMPLETE)

VERTEX COVER

MIN VERTEX PROB

SOL

- Given $G = (V, E)$ a vert. cov. is $Q \subseteq V \mid i \in C \vee \exists j \in Q, (i, j) \in E$
- \exists a vert. cover of size $\leq k$?

- Try all subset $S \subseteq V$, check if it's a vertex cover.
Keep the one of minimum size.

$$O(\underbrace{2^{|V|}}_{\# \text{subset of } V} \cdot \underbrace{|E|}_{\# \text{check if is a vertex cover}})$$

How can we solve NP-HARD efficiently? For example, reducing the set of instances

TH [König]

Vertex cover is P
Bipartite graphs

In BIPARTITE GRAPHS, the size of a max matching is equal to the size of maximum vertex cover

LEMMA

- Let $m(G)$ the max size of a matching in G
- Let $vc(G)$ the min size of a vert. cover in G
- $\forall G \quad m(G) \leq vc(G) \leq 2m(G)$

ALGO

$$M = \emptyset$$

WHILE ($\exists e \in E / M \cup \{e\}$ is a matching) } COMPUTE A MAXIMAL MATCHING
| add e to M

BEST $V(M)$ = vertices touched by $|M|$

- It's a GREEDY ALGO that calculate in $O(|E|)$ a vert. cover of size twice of the minimum vertex cover

$$|V(M)| - 2|M| \leq 2m(G) \leq 2vc(G)$$

COMPLEXITY
 $O(|E|)$

(HANDLE NP-HARD P) → C-APPROXIMATION ALGOS

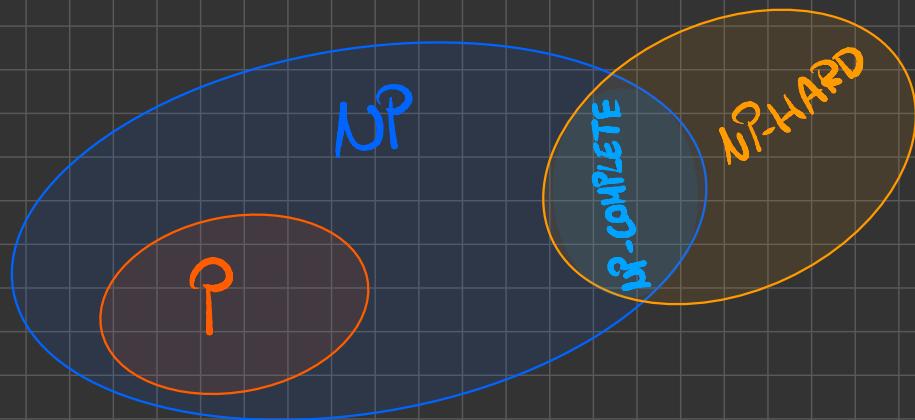
P

- set of Probs. solvable in POLYNOMIAL TIME on a DETERMINISTIC TM.
- set of Probs. solvable in POLYNOMIAL TIME on a NO-N-DETERMINISTIC TM, and their SOLs can be verified in POLYNOMIAL TIME on a DETERMINISTIC TM.
- set of Probs. | $\text{Time UP} \nexists$ } POLYNOMIAL TIME REDUCTION to any NP-HARD, an NP-HARD prob can or cannot be in NP
- set of Probs. in $NP \wedge NP\text{-HARD}$ | $\text{Time NP} \nexists$ } Poly.T. reduts to any $\tilde{P} \in NP\text{-COMPLETE}$

NP

NP-HARD

NP-COMPLETE



C-APPROX
ALGOS

- WORKS in POLY TIME
- Finds a valid SOLUTION

Has a guarantee of the "SOLUTION QUALITY"

$$\rightarrow \text{MIN} \quad \text{SOL} \leq C \cdot \text{OPT}, \quad C \geq 1$$

$$\rightarrow \text{MAX} \quad \text{SOL} \geq C \cdot \text{OPT}, \quad C \leq 1$$