Lecture 8 - Tree Algorithms

Monday, February 7, 2022 6:53 AM

Quiz 3 p 2 alternative induction proof:

(weakly connected)
For digraph D where $\forall v \in V(D): d^+(v) \geq 1$
$$\Rightarrow \exists C \in D$$

We'll use induction + edge contraction on $m = |E(D)|$

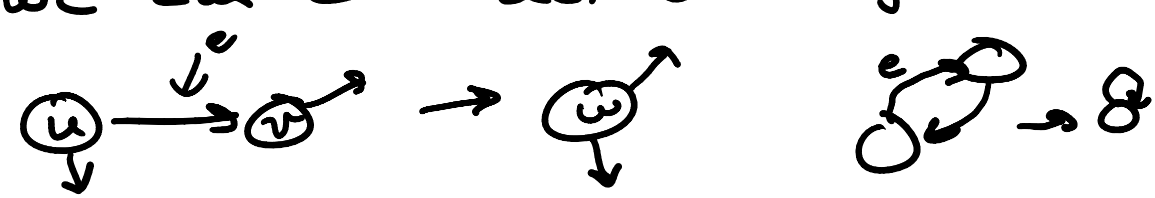
Basis: $P(1)$

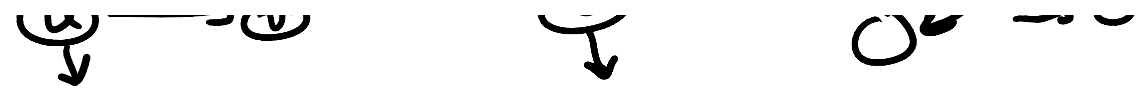
IMPORTANT: any construction we do must be able to realize ALL possible configurations that fit our assumption

weak → can't just add edge, as not all graphs have a self loop

Strong → can't just delete an edge, as we'd need to guarantee we don't break a cycle

* But we can contract an edge





P(n) we have a D fitting assumptions

we select some $e \in E(P(n))$ to contract to create P(k)

Note: won't remove a cycle

I.H. on P(k) $\rightarrow \exists C$ on P(k)

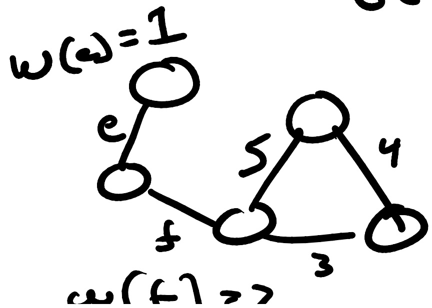
we uncontract that edge, retaining any cycles $\Rightarrow \exists C$ on P(n) $\square$

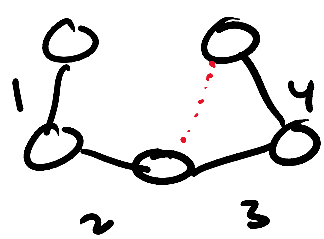
Tree Algorithms

weighted graph $G = (V, E, w_E)$

weighted edges: consider our edge set $E \rightarrow$ there is $\forall e \in E$ some associated $w \in W$ for that edge

G





MST of G

Minimum spanning tree (MST)

a spanning tree on a weighted graph that has a minimum sum of weights

To determine MST: Krushkal's Algo

output tree $\quad$ input $G = (V, E, w)$

$V(T) \leftarrow V(G)$

$E(T) = \emptyset$

sort w, E in nondecreasing order $O(m \log m)$

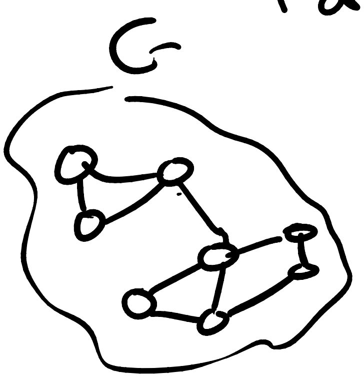
for all $w, e \in W, E$: $O(m)$

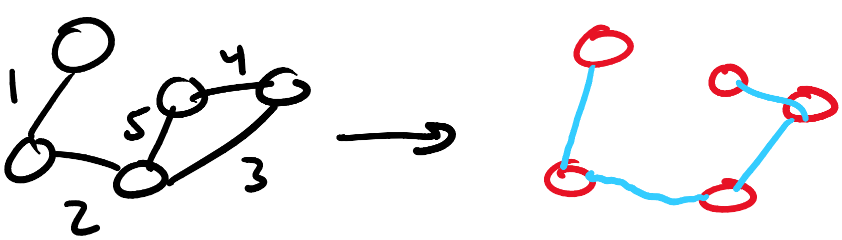
  if numComp$(T+e) <$ numComp(T): each is $O(n+m)$

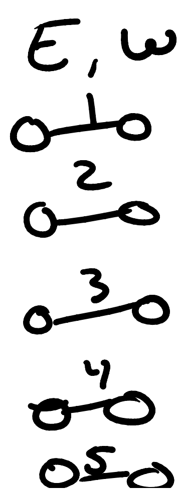
    $E(T) \leftarrow e$

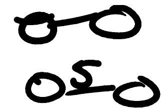
  if numComp$(T) = 1$: $O(n \log n)$

    break



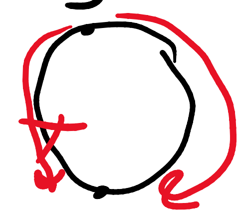






Prove Kruskal's algorithm outputs a MST

T: any edge we add will only connect components
→ always a cut edge
→ not on a cycle ✓

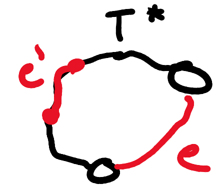


S: Assuming G is connected, we go until T is fully connected, as T contains all $v \in V(G)$
$\Rightarrow$ T is spanning ✓

M: psuedo-algorithmic argument
Consider Kruskal outputs a S.T. that is not minimum where $T^* = MST$

- Consider some $e \in E(T)$ s.t. $e \notin E(T^*)$ where e is the first such edge chosen
- Adding e to T^* creates a cycle

- Adding e to T^* creates a cycle
- Consider $e' \in C$, $e' \notin E(T)$



Note: T^* has all edges in T that were selected before e

$\rightarrow$ so e and e' were both available for selection by $T \Rightarrow w(e) \leq w(e')$

define $T' = T^* + e - e'$

Note: $W(T') \leq W(T^*)$

sum of weights

$\Rightarrow$ we have T' with more edges in common with T than T^*

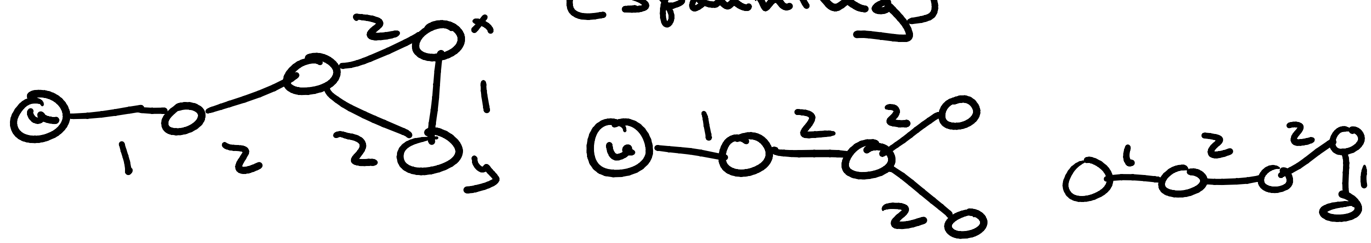
$\Rightarrow$ If we repeat this argument, $T \rightarrow T' \rightarrow T^*$, we convert T to T^* $\square$

Shortest paths

specifically: single-source shortest paths (SSSP) shortest paths (SSSP)

→ from vertex u, identifying all shortest path distances $d(u,v)$ to all other $v \in V(G)$

Also: consider all u,v-paths
→ shortest paths tree (spanning)



G   SSSP tree   MST

⇒ Note <u>not</u> equal to MST

Dijkstra's algorithm

$\forall v \in V(G)$: $D(v) = \infty$ ← distance to v

$D(u) = 0$

$S = V(G)$ ← unvisited set

while $S \neq \emptyset$:

#edges $w = \min(D, S)$ ← vertex in S with min. value of D

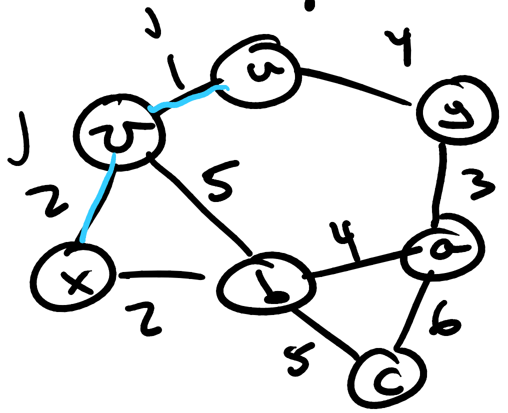
$\forall x \in N(w)$ s.t. $x \in S$:

#verts ↓ $O(n+m)$ ↓ #edges
work complexity

$\forall x \in N(w)$ s.t. $x \in S$:
  $t = W(x,w)$ ← weight of edge
  if $D(w) + t < D(x)$
    $D(x) = D(w) + t$

$S = S - w$

Example of SSSP with Dijkstra



	D_0	D_1	D_2			
$\boxed{u}$	0	0	0	0	0	0
$\boxed{v}$	∞	1	1	1		
$\boxed{x}$	∞	∞	3	3		
y	∞	4	4	4		
a	∞	∞	∞	∞		
b	∞	∞	6	5		
c	∞	∞	∞	∞		

exercise 4 reader

Prove correctness of Dijkstra

↳ Prove at every iteration:

X: visited vertices

1 - $\forall v \in X$ $D(v) = d(u,v)$ ← actual shortest u,v-path length

2 - $\forall v \notin X$ $D(v)$ is shortest u,v-path from X weak

We'll do induction on $|X|$

We'll do weak induction on $|X|$

$P(1) \Rightarrow X = \{u\}$ $\quad D(u) = d(u,u) = 0$

all $v \in N(u)$ they take on distance of (u,v) edge weight

$P(k) \Rightarrow |X| = k$

assume via I.H. that the above two cases hold

$P(k+1) = X' = X + v$

v is selected s.t. $D(v)$ is least of all $v \notin X$

First show: $D(v) = d(u,v)$

By I.H. $\rightarrow$ shortest path directly from X to v is $D(v)$, so any other possible path that exits X and reach v is bounded below by $D(v)$

✓

Secondly show: $D(w)$ is correct for $X' = X + v$, $w \notin X'$

via I.H $\rightarrow D(w)$ is shortest u,w-path via I.H $\rightarrow D(w)$ is shortest u,w-path distance directly from X

- We update $D(w) = \min\Big(\underbrace{D(w)}_{\text{distance from } X},\ \underbrace{D(v) + w(v,w)}_{\text{distance from } X'}\Big)$

$\longrightarrow$ shortest possible path to w through X', as v is the only way to get to w through a vertex not originally in X $\square$