HONORS ALGORITHMS

2025.02.14

MIN-WEIGHT SPANNING TREE

INPUT: (V, E, w) $\quad w : E \to \mathbb{R}$

(V, E): connected graph; w: weight fctn

GREEDY ALGORITHM

$e_i = \{x_i, y_i\}$

sort edges by weight

$w(e_1) \le \cdots \le w(e_m)$

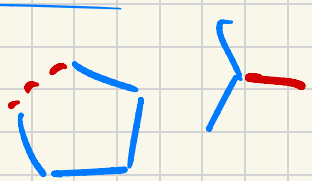
init $F = \emptyset$ (: collect edges of sp. tree)

for $i = 1$ to m

if x_i, y_i not in same component of F

then add e_i to F

return F



loop invar:

(V, F) is a forest

$w(e_1) \le w(e_2) \le \cdots \le w(e_m)$

Proof of optimality

① assume all weights distinct

Claim min.-weight sp. tree is unique

let (V, F) : greedy tree

(V, R) : ~~another~~ sp tree min-weight

Claim: if $w(R) \le w(F)$ then $F = R$

$E_i = \{e_1, \ldots, e_i\}$

$F_i := E_i \cap F$ $\quad$ $R_i := E_i \cap R$

Prove $F_i = R_i$ by ind. on i

$E_m = E$ Note: $F_m = E \cap F = F$

$R_m = R$

$E_0 = \emptyset = F_0 = R_0$ ✓

let $i \ge 1$ I.H. $F_{i-1} = R_{i-1}$

DC $F_i = R_i$

i.e. $e_i \in F_i \leftrightarrow e_i \in R_i$

Let $i \geq 1$ I.H. $F_{i-1} = R_{i-1}$

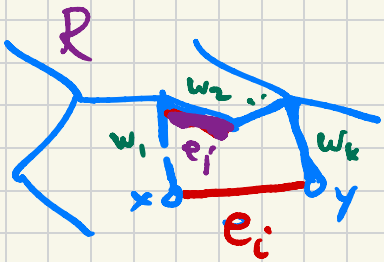
DC $F_i = R_i$

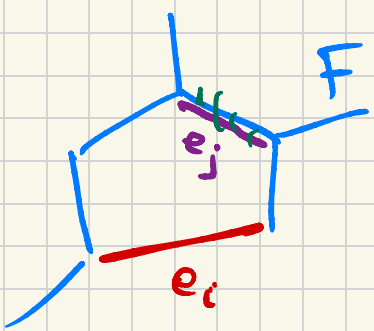
i.e. $e_i \in F_i \leftrightarrow e_i \in R_i$

Pf by contradiction:

Case 1 $e_i \in F_i$, $e_i \notin R_i$

Case 2 $e_i \notin F_i$, $e_i \in R_i$





Can $j > i$?

Case 2a $w(e_j) > w(e_i)$

in i^{th} round e_i was available

b/c $e_j \notin F_i$

So Greedy should have selected e_i

Case 2b: $\forall j$ along path $j < i$

Case 1a: $(\exists j)(w_j > w(e_i))$ (R)

$R' := R \cup \{e_i\} \setminus \{e_j\}$

$w(R') < w(R)$ $\rightarrow\leftarrow$

Case 1b: $(\forall j)(w_j < w(e_i))$

but then all e_j along path

I.H. $\rightarrow \in F$

$e_i \in F$

cycle in F $\rightarrow\leftarrow$

14 $\Rightarrow$ path $\subseteq R \Rightarrow$ cycle in R

$\rightarrow\leftarrow$

another greedy algorithm for

MIN-WEIGHT SPANNING TREE

GROW TREE FROM A SOURCE VERTEX

$W := \{source\}$ // current tree: (W, F)

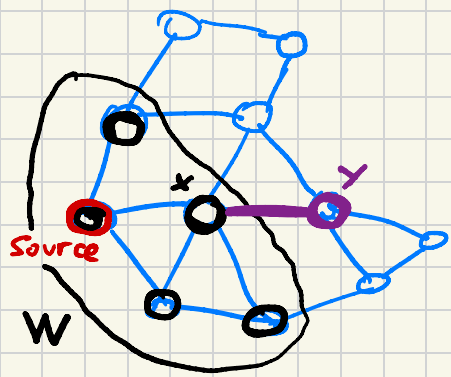
$F := \emptyset$

---

let $e = \{x, y\}$ be min-weight edge connecting W to $V \setminus W$ ($x \in W$, $y \in V \setminus W$)

add y to W

add e to F



JARNIK's algorithm 1930

PRIM's algorithm 1957

efficient implementation: like DIJKSTRA

modify:

RELAX