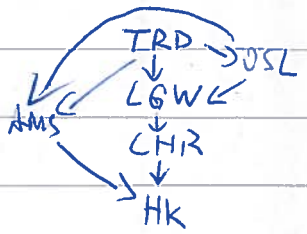
§13 Optimization

- Weighted graphs (directed/undi/multi)
- e.g. distance (Real dist)
  travel cost
  travel time
  capacity
  ....
  vertex = city
  edge = road/airline/...



- Ques: shortest distance (on the graph)
  spanning tree
  (Hamilton Paths/Cycles) postman
  ....

§13.1 Shortest-Path Algorithm

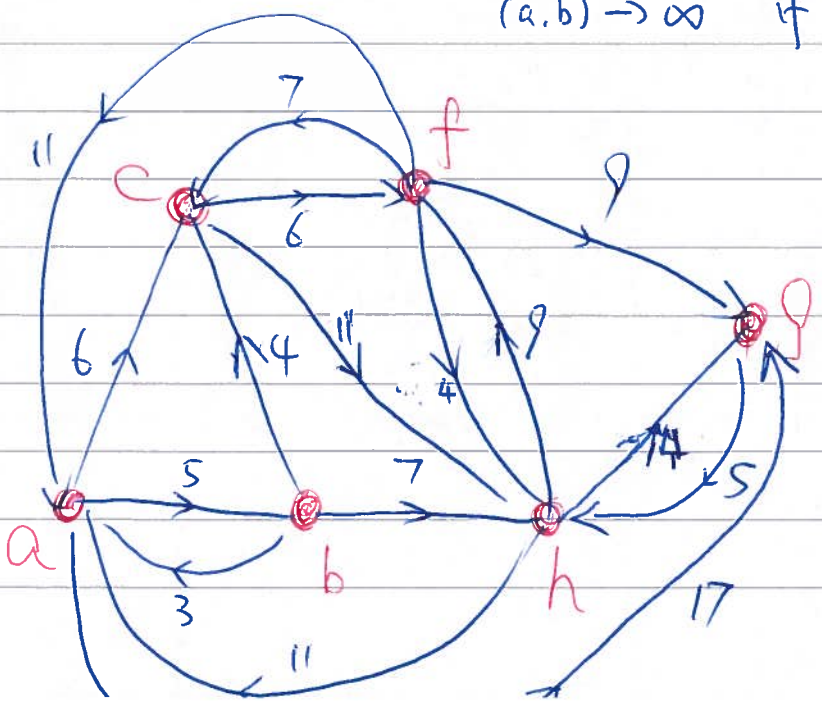
- $G=(V,E)$ loop-free directed (non-multi)

weight is a function: $wt: E \to \mathbb{R}_{\geq 0}$

$\rightsquigarrow$ extension to $wt: V \times V \to \mathbb{R}_{\geq 0} \cup \{\infty\} = \overline{\mathbb{R}}_{\geq 0}$

$(a,b) \to wt(e)$ if $\exists e: a \to b$

$(a,b) \to \infty$ if $\nexists e: a \to b$



(2)

Recall A path $\wp: a \rightsquigarrow b$ is

$a = v_0 \xrightarrow{e_1} v_1 \xrightarrow{e_2} v_2 \to \cdots \xrightarrow{e_\ell} v_\ell = b$.

(weighted)

- Length of $\wp = wt(e_1) + \cdots + wt(e_\ell)$
- distance function: $d: V \times V \to \overline{\mathbb{R}}_{\geq 0}$

$$d(a,b) = \min \text{ length of } \wp \begin{cases} \wp: a \rightsquigarrow b \\ \infty \text{ if } \nexists \text{ path } a \rightsquigarrow b. \end{cases}$$

Aim: Fix $v_0 \in V$ & determine $d(v_0, v)$ $\forall v \in V$

Alg: Greed Algorithm (general algo.)
principle is to achieve "min" locally & hence globally

|| Find the closest vertex from the rest vertices

- Dijkstra's:

Step 1) Set $i=0$, $S_0 = \{v_0\}$. Label v_0 with $(0, -)$ & $v \neq v_0$ with $(\infty, -)$.

If $n=1$ then ▨ else ↘

Step 2) For each $v \in \overline{S_i}$, replace the label (if possible) on v by $(L(v), y)$ where

$$L(v) = \min_{u \in S_i} \{L(v), L(u) + wt(u,v)\}$$

& y is the vertex in S_i that produces min $L(v)$ above.

Step 3) If $\forall v \in \overline{S_i}$ is labeled with $(\infty, -)$. ▨

(3)

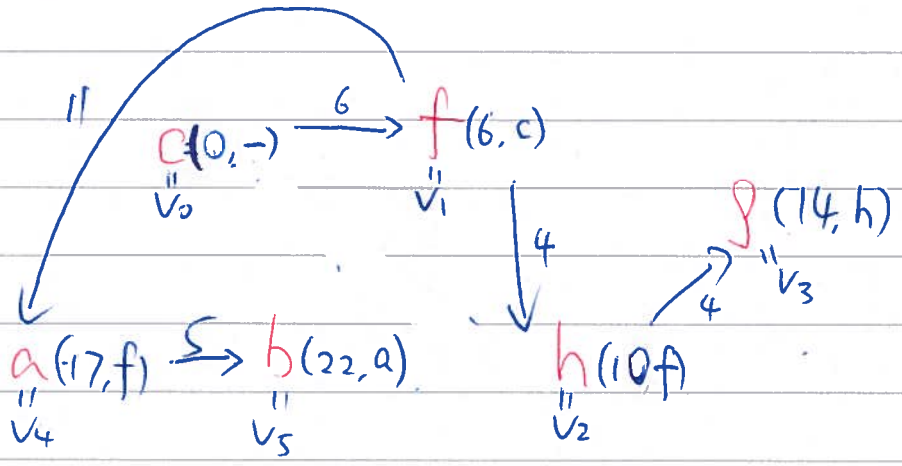
Otherwise:

3.1) Choose $v_{i+1} \in \overline{S_i}$ s.t. $L(v_{i+1})$ is min among $\overline{S_i}$.
(so v_{i+1} is the vertex in $\overline{S_i}$ that is closest to v_0)

3.2) $S_{i+1} = S_i \cup \{v_{i+1}\}$

3.3) $i = i+1$. If $i = n-1$ ⏹ otherwise back to 2)

Ex: $c = v_0$



- Complexity (worst-case).

$i = 1 \sim (n-1)$

$u \in S_i$ $L(u) + wt(u,v)$ Compare $L(v)$ — $i(n-i)$

$v \in \overline{S_i}$ — $m(n-i)$

$$\sum_{i=1}^{n-1} i(n-i) = \sum_{i=1}^{n-1} in - \sum_{i=1}^{n-1} i^2 \sim \Theta(n^3).$$

$$\boxed{\Theta(m \log_2 n)} \leftarrow \Theta(mn) \leftrightarrow \Theta(n^3)$$

(4)

• $L(v)$

v_i = ith-closest vertex

$S_i = \{v_0 \cdots v_i\}$.

• $L(v_0)=0$, $L(v)=\infty$ $v \neq v_0$. $S_0 = \{v_0\}$.

$i \geq 0$. $v \in S_i$

$$L_{i+1}(v) = \min\{L_i(v), L_i(v_i) + wt(v_i, v)\}$$

$$= \min_{1 \leq j \leq i} \{d(v_0, v_j) + wt(v_j, v)\}$$

$v_{i+1} = v \in S_i$ s.t. $L_{i+1}(v)$ minimalizes.

• Complexity $\sum_{i=1}^{n-1} 3(n-i) \sim \Theta(n^2)$

4.9.

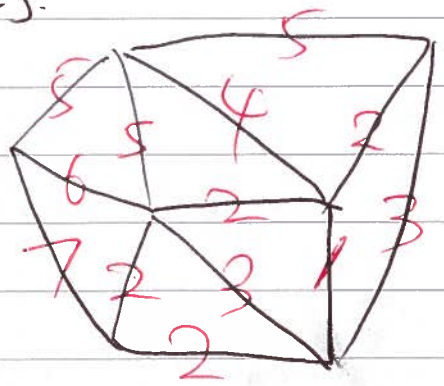
§13.2 Minimal Spanning Tree.

- Senario: To set up a network of computers

$G=(V,E)$ $V=\{\text{computers}\}$ $wt(E)=$ cost to build a connection

Aim: spent minimal cost to get a connected network.

e.g.



(no edge $\Leftrightarrow$ cost $=\infty$)

$\Downarrow$

The resulting subgraph is a tree

(? otherwise delete 1 edge in 1 cycles)

Def. MST is a tree $T \subset G$ s.t. it is spanning, connected & $wt(T)=\sum_{e\in T} wt(e)$ is minimal.

Idea 1. Use shortest edges first!

$\Rightarrow$

Kruskal's algorithm:

Step 1 Set $i=1$ & $e_1 \in E$ s.t. $wt(e_1)$ is minimal.

For $1 \le i \le n-2$, suppose edges $e_1, \dots, e_i$ have been selected.

Choose e_{i+1} in the remaining edges s.t.

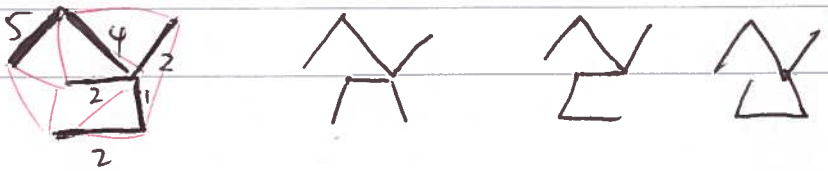
✓ $\nexists$ cycles in the subgraph of G consisting of edges $\{e_1, \dots, e_{i+1}\}$

2/ $wt(e_i)$ is minimal among edges satisfying ✓.

Step 3 $i=i+1$ If $i=n-1$ ⊡

$i<n-1$, to step 2.

e.g.



4 possibilities

Thm $G=(V,E)$ loop-free, wt, conn., undirected.
Then ∀ spanning tree produced by K's alg. is optimal.

- NOT OBVIOUS! since even at the beginning, it is possible ∃ many choice of e_1 & as we see before, ∃ many MST.

Proof By contradiction.
Suppose T is a spanning tree of G produced by K's alg.
Let $E(T)=\{e_1,\cdots,e_{n-1}\}$.
Then ∃ T', a MST of G s.t. $wt(T')<wt(T)$.
So ∃ $1\le r\le n-1$ s.t. $\{e_1,\cdots,e_{r-1}\}\subset E(T')$
but $e_r\notin E(T')$.

Among these MST T' → Choose T' s.t. r reaches maximal.
Let $E(T')=\{e_1,\cdots,e_{r-1},e'_r,\cdots,e'_{n-1}\}$.
By K's algorithm $wt(e'_j)\ge wt(e_r)$ $\forall j\ge r$.

In T', ∃! path connecting the endpoints of $e_r: x-y$,
denote it by $p: x\overset{a_1}{-}x_1-\cdots-x_{m-1}\overset{a_m}{-}y$ $(a_i\in E(T'))$.
Since T is tree, then
$\{a_1,\cdots,a_m\}\not\subseteq E(T)$.
Say $a_k\notin E(T)$ but $a_k\in E(T')$, so $a_k\in\{e'_r,\cdots,e'_{n-1}\}$
& $wt(a_k)\ge wt(e_r)$

Let $T''=T\setminus\{a_k\}\cup\{e_r\}$.
which is also a tree (why?)
& $wt(T'')\le wt(T')<wt(T)$
& $E(T'')\cap E(T)=\{e_1,\cdots,e_r\}$. contradiction to the choice of T'.

(3)

Idea2 Build a tree & spanning the tree.

Prim's algo.

Step1. $i=1$. $\forall v_i \in V$ & $T_1 = \{v_1\}$.

Step2 $1 \le i \le n-1$ & we have a tree T_i with i vertices.

(V_i, E_i)

Let e_{i+1} be an edge in the remaining edges s.t.

✓ $T_i \cup \{e_{i+1}\}$ is still a tree

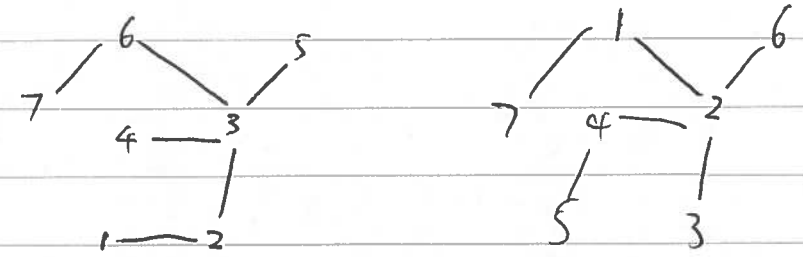
? $w(e_{i+1})$ is minimal

$T_{i+1} = T_i \cup \{e_{i+1}\}$

Step3 $i = i+1$ If $i = n$ ▨

If $i < n$ back to Step2.

e.g. (Note that we can start with ∀ vertex)



How to check if e_{i+1} satisfies ✓ ?

easy: one (exactly) of its endpoints is in V_i.

In K's algo. How to check $e_{i+1} \cup \{e_1, \dots, e_i\}$ contains no cycles? Harder.

(4)

Thm Prim's algor. also produces MST.

~~Proof~~ "Similarly"