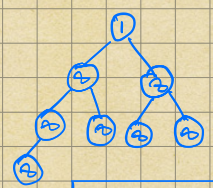
- In kruskal's algorithm;

(1). Every vertex has a key attached to it.

	1	2	3	4	5	6	7	8	9
in-heap:	T	T	T	T	T	T	T	T	T
key:	0	∞	∞	∞	∞	∞	∞	∞	∞
Pred:	–	–	–	–	–	–	–	–	–
index:	1	2	3	4	5	6	7	8	9

↳ index or location of each matrix in the heap



keys:	1	∞	∞	∞	∞	∞	∞	∞	∞

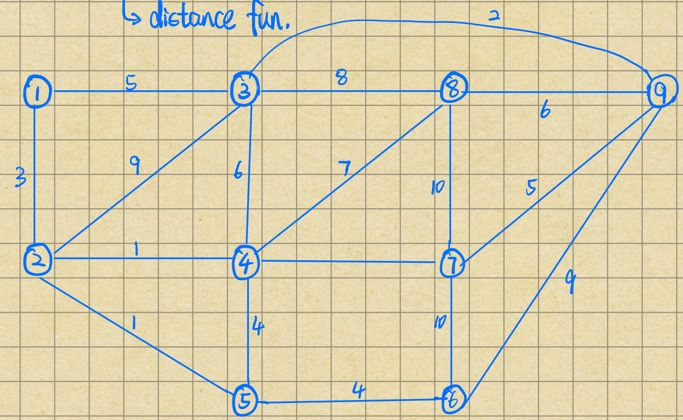
- $n=|V|$; $m=|V|$

$n\cdot \underbrace{O(\lg n)}_{\text{delete min}} + m\underbrace{(\lg n)}_{\text{decrease key}}$

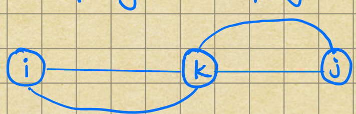
- The Shortest Path Problem:

$G=(V,E,d)$ $\quad d: E \to R$

↳ distance fun.



- Principle of dynamic programming (Bellman)



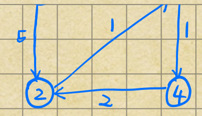
memory-less Process

Ex. Matrices: $\overset{n_1\times n_2}{A_1}\ \overset{n_2\times n_3}{A_2}\ \overset{n_3\times n_4}{A_3} \cdots A_n$

$\overset{2\times 3}{A_1}\cdot \overset{3\times 5}{A_2}\cdot \overset{5\times 7}{A_3}$

$$\begin{cases} (A_1 * A_2)\cdot A_3 = 30 + 70 = 100 \\ A_1(A_2 * A_3) = 40 + 105 = 145 \end{cases}$$

Ex. $1 \xrightarrow{1} 3$ $\quad \begin{array}{c|cccc|} & 1 & 2 & 3 & 4 \\ 1 & 0 & 5 & 1 & \infty \end{array}$ $\quad$ (i) ⟶ the length of the shortest path from i to j

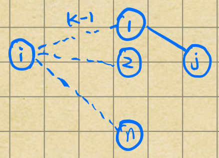


$$\begin{array}{c|cccc} 2 & \infty & 0 & 1 & \infty \\ 3 & \infty & \infty & 0 & 1 \\ 4 & \infty & 2 & \infty & 0 \end{array}$$

- Define : $d_k(i,j)$ = the length of shortest path from i to j with the # of edges at most k.

$$d_0(i,j): \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{bmatrix} 0 & \infty & \infty & \infty \\ \infty & 0 & \infty & \infty \\ \infty & \infty & 0 & \infty \\ \infty & \infty & \infty & 0 \end{bmatrix} \end{array}$$

- Assume that $d_{k-1}(i,j)$, comput $d_k(i,j)$

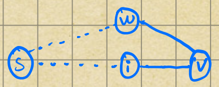


$$d_k(i,j) = \min_{l=1 \cdots n}\left(d_{k-1}(i,l) + d_1(l,j)\right) \rightsquigarrow O(n^4)$$

$$\begin{cases} + \to \overset{min}{\wedge} \\ * \to + \end{cases}$$ min-plus multiplication

1. Bellman Problem: Shortest Path algorithm

$G=(V,E)$; $|V|=n$; $|E|=m$; $w: E \to R$



- A shortest path in a graph w/o negative cycles can only have at most $n-1$ edges
- each vertex V: $l(v) \longrightarrow$ the length of shortest path from s to v.

  $\pi(v) \longrightarrow$ predecessor of V in the estimated shortest path from s to v

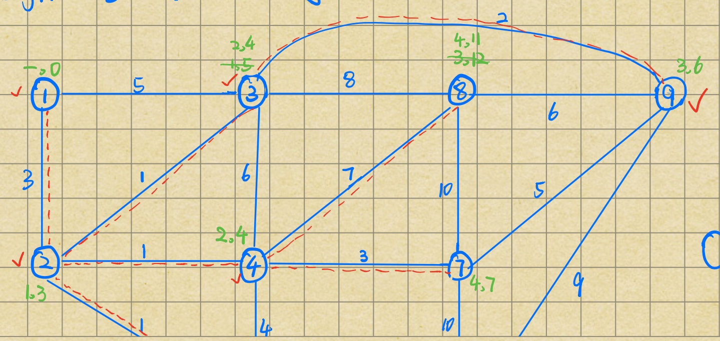
- $l(u) = \min(\underbrace{l_{k-1}(u)}, \underbrace{l_k(w)+d(w,u)})$

  $\begin{cases} \pi_k(u) = \pi_{k-1}(u) \\ l_k(u) = l_{k-1}(u) \end{cases}$

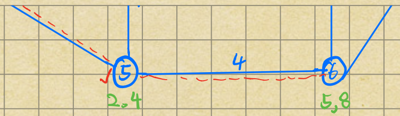
  $\rightarrow$ don't change labeled level of u.

Ex. See note

2. Dijkstra's Shortest Path Algorithm



$O(n+m)\log n$


5
4
6
2,4
5,8