CSE 101 3-10-22

- SET 5 : Sunday 11:59 PM
- Pa8 : due tonight
- quizs : soln Posted.

Weighted Graphs

Defn $G = (V, E)$ is a weighted graph if we have a fcn.

$$w : E \to \mathbb{R}^+,$$

weight of a Path: $(x \rightsquigarrow y)$

Let $P: x = v_0, v_1, v_2, \ldots, v_{k-1} = y$

$$w(P) = \sum_{i=0}^{k-1} w(v_i, v_{i+1}) = \sum_{e \in E(P)} w(e)$$

shortest Path weight:

$$\delta(x,y) = \begin{cases} \min w(P) & P \text{ is an } x \rightsquigarrow y \text{ Path} \\ \infty & y \text{ not reachable from } x \end{cases}$$

shortest Path

a Path P from x to y

such that $w(P) = \delta(x,y)$.

SSSP Problem:

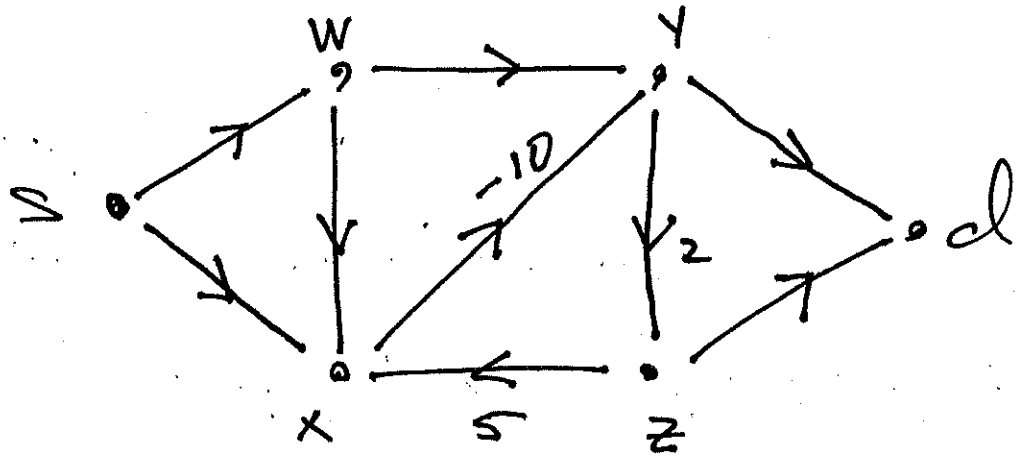
Given $s \in V(G)$ (source),
determine $\delta(s,x)$ for all $x \in V(G)$.
For those x with $\delta(s,x) < \infty$,
determine a shortest s-x path.

Rmk

- Two algorithms:
  - Dijkstra: requires pos. edge weights
  - Bellman-Ford: allows neg. weights
- Both algorithms actually find a shortest (lightest) walk in G.
  (a min-weight walk is necessarily a path.)

- the last point requires that there be no negative weight cycles, reachable from the source s.

EX.



C: x, y, z, x has weight = -3

- Dijkstra: outlaws neg. weight edges.
- Bellman-Ford: Detects existence of neg. weight cycles reachable from source, returns false in that case. Otherwise returns true.

Two vertex attributes:

$\pi[x]$: parent (or predecessor) of x.

$d[x]$: estimate of $\delta(s, x)$.

Parents define predecessor subgraph

$$V_\pi = \{ x \in V \mid \pi[x] \neq nil \} \cup \{ s \}$$

$$E_\pi = \{ (\pi[x], x) \mid \pi[x] \neq nil \}$$

To extract shortest paths use

PrintPath(G, s, x)

in sec. 22.2 p. 601 of CLRS.

Both algorithms use:

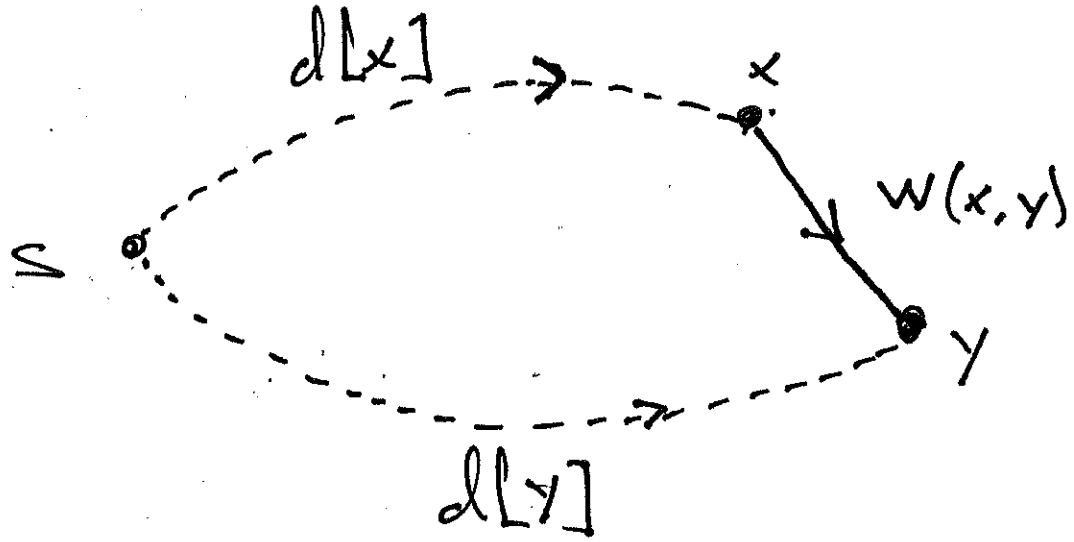
Initialize (G, s)

1. for all $x \in V$
2. $\quad d[x] = \infty$
3. $\quad p[x] = nil$
4. $d[s] = 0$

Relax (x, y) pre: $y \in adj[x]$.

1. if $d[y] > d[x] + w(x, y)$
2. $\quad d[y] = d[x] + w(x, y)$
3. $\quad p[y] = x$

Picture



Rmks

- Relax(x,y) changes only fields of y
- after Relax(x,y) we have

$$d[y] \leq d[x] + w(x,y)$$

- Both algorithms perform a sequence of calls to Relax(·,·) on various edges in G.

Lemma 1

Let $x \in V(G)$. Suppose that after Initialize(G, s), some sequence of calls to Relax$(\cdot,\cdot)$ causes $d[x]$ to be finite. Then G contains an s-x path of weight $d[x]$.

Lemma 2

After Initialize(G, s), the inequality

$$\delta(s, x) \leq d[x] \quad (\forall x \in V)$$

is maintained over any seq. of relaxations.

Lemma 3

If $P: s = x_0, x_1, x_2, \ldots, x_k$ is a shortest $s - x_k$ path, and the edges of P are relaxed in order:

$$(x_0, x_1), (x_1, x_2), (x_2, x_3), \ldots, (x_{k-1}, x_k)$$

then $d[x_k] = \delta(s, x_k)$. This is true regardless of any other relaxation steps that occur, even if interspersed with the above relaxations.

Bellman-Ford:

see pseudo-code

Runtime: $n = |V|$, $m = |E|$

- initialize: $\Theta(n)$
- Relax costs $\Theta(1)$.
  each edge relaxed $n-1$ times
  so cost of 1st for loop:
  $(n-1)\Theta(m) = \Theta(m \cdot n)$.
- 2nd for loop: $\Theta(m)$
- total $= \Theta(mn) + \Theta(m) = \Theta(m \cdot n)$.

Dijkstra(G, s)

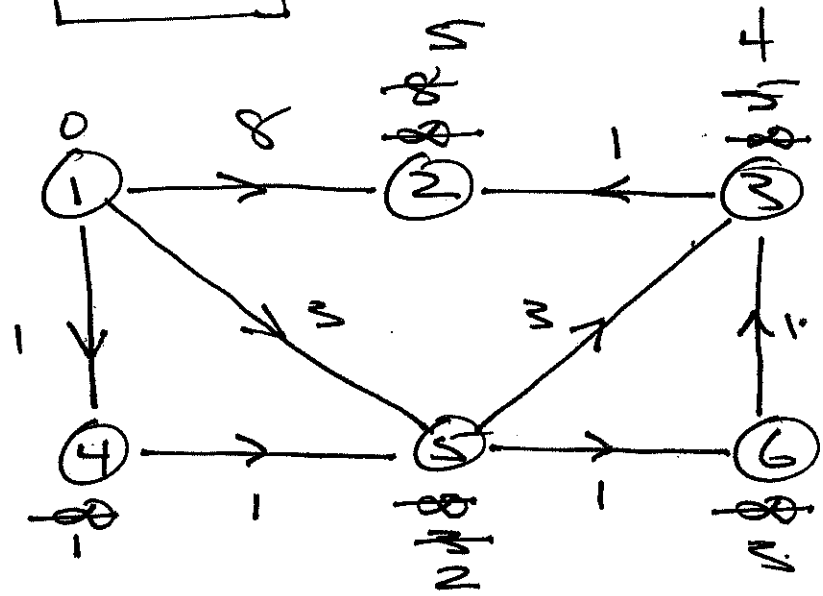
see Pseudo-code.

Note: Relax(x, y) contains an implicit call to DecreaseKey()

Relax(x, y)

1. if $d[y] > d[x] + w(x, y)$
2. DecreaseKey(y, $d[x] + w(x, y)$)
3. $p[y] = x$

Ex. $s = 1$

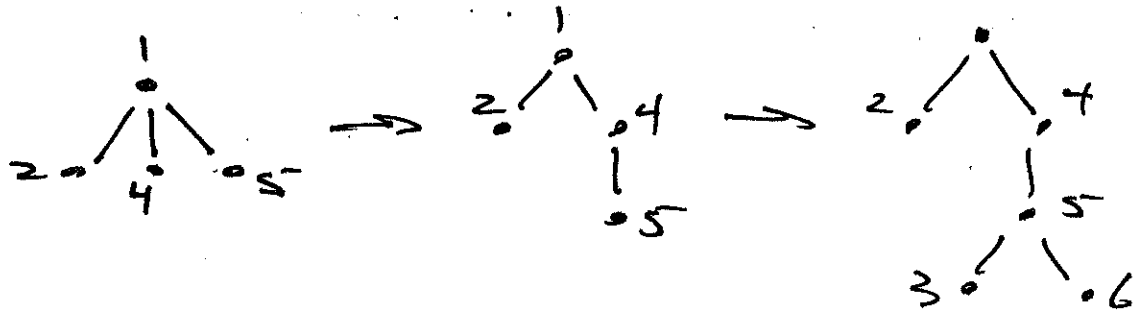


P.Q. : ~~1~~ ~~2~~ ~~3~~ ~~4~~ ~~5~~ ~~6~~

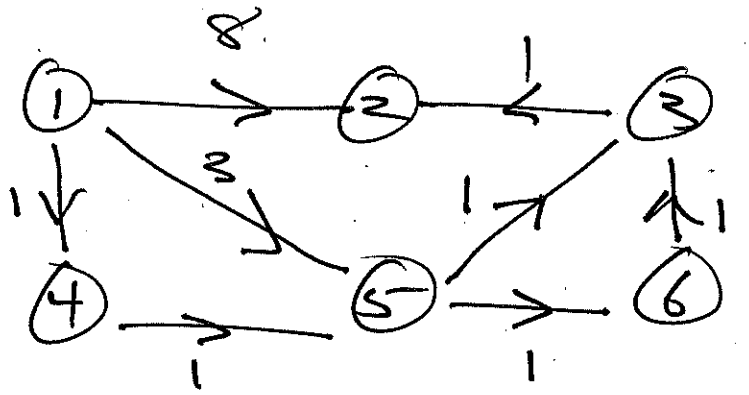
d : 0 ~~5~~ ~~4~~ 1 ~~2~~ 3

p : nil ~~3~~ ~~6~~ 1 ~~4~~ 5

tree:



Ex.



Rule: if two vertices have same min d-value, extract the one with smaller label.

Runtime:

$O((n+m)\log n)$