CSE 101-02 5-31-22

- SETs : closes Sunday 11:59 PM

  incentive: response rate ≥ 85%

  ⇒ will boost all grades by 0.5%

- final exam (101-02) Tue 6/7, 4-6 PM

- Pas: ext 1 day.

Priority Queues:

Heap implementation.

HeapInsert (A, K) Pre: heapSize[A] < length[A]

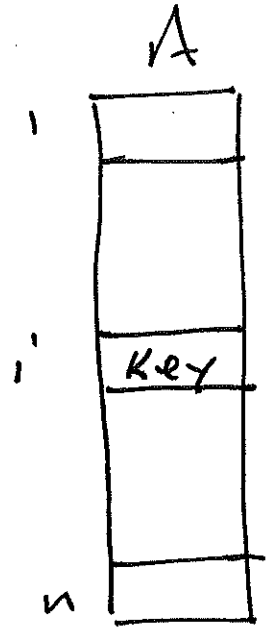
cost: $\Theta(\log n)$.

Exercise:

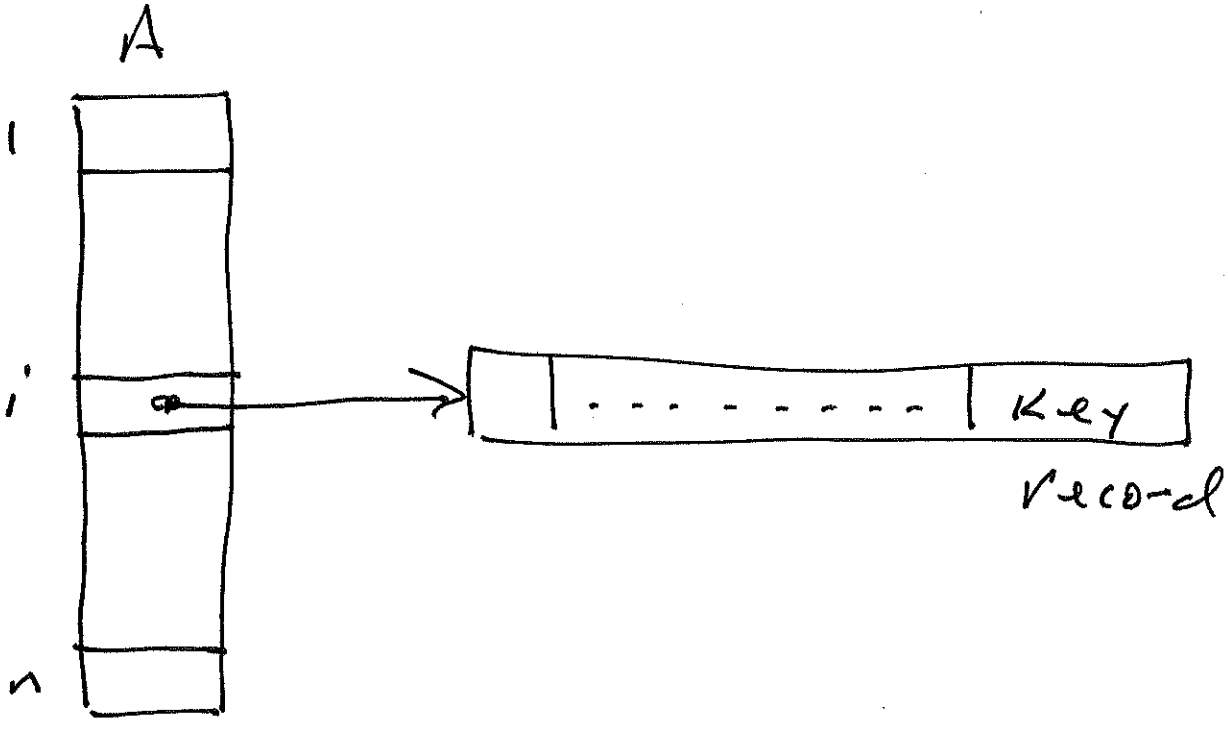
re-write all algorithms using a min heap, min Priority Queue

Notice: our heap implementation assumes no satellite data.

Our Picture of heap / P.Q.



General Picture



Exercise

re-write all BFS/P.Q. algorithms in terms of general Picture.

why have a Priority queue?

SSSP in a weighted graph

Path

A weighted graph (directed or undirected) is a graph $G = (V, E)$ with a weight fcn. $w : E \longrightarrow \mathbb{R}$.

Defn

if P is a path in G

$$P: \quad u = x_0, x_1, x_2, \ldots, x_k = v$$

the Weight of P is

$$w(P) = \sum_{i=1}^{k} w(x_{i-1}, x_i)$$

Defn

Shortest Path Weight:

$$\delta(u,v) = \begin{cases} \text{weight of a min-weight } u\text{-}v \text{ path, if } v \text{ is reachable fr. } u \\ \infty \quad \text{otherwise} \end{cases}$$

SSSP Problem

Given a weighted graph $G=(V,E,w)$ and a source vertex $s \in V$, find the shortest path weight $\delta(s,x)$ for all $x \in V$. If $\delta(s,x) < \infty$, determine a shortest $s-x$ path.

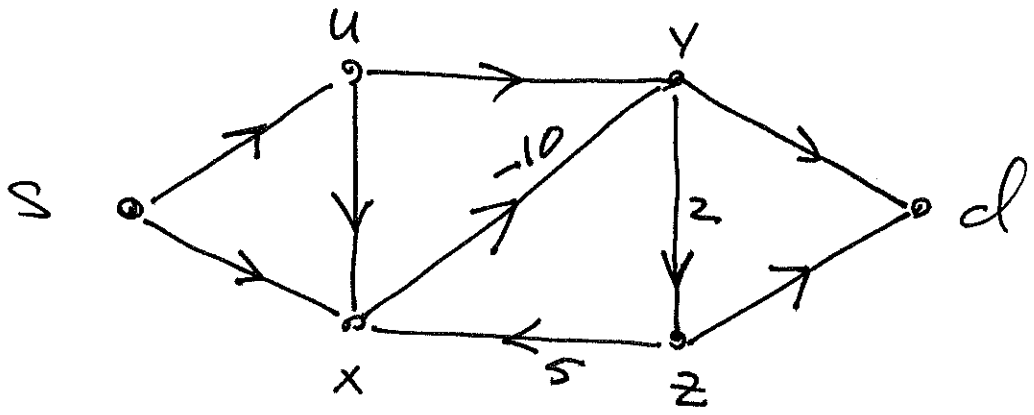
Assume: G is a digraph.

Two algorithms

- Dijkstra
- BellmanFord

Both algorithms actually find shortest walks. note though a shortest walk might not exict!

Ex.



neg. weight cycle : $w(x, y, z, x) = -3$

- Dijkstra: outlaws neg. weight edges.

- BellmanFord: detects neg. weight cycles reachable from s.

Common infrastructure

P[x]: Parent of x, encodes shortest path tree (PrintPath(G, s, x) prints path).

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

Predecessor subgraph: $G_p = (V_p, E_p)$

$$V_p = \{ x \in V \mid p[x] \neq nil \} \cup \{ s \}$$

$$E_p = \{ (p[x], x) \mid p[x] \neq nil \}$$

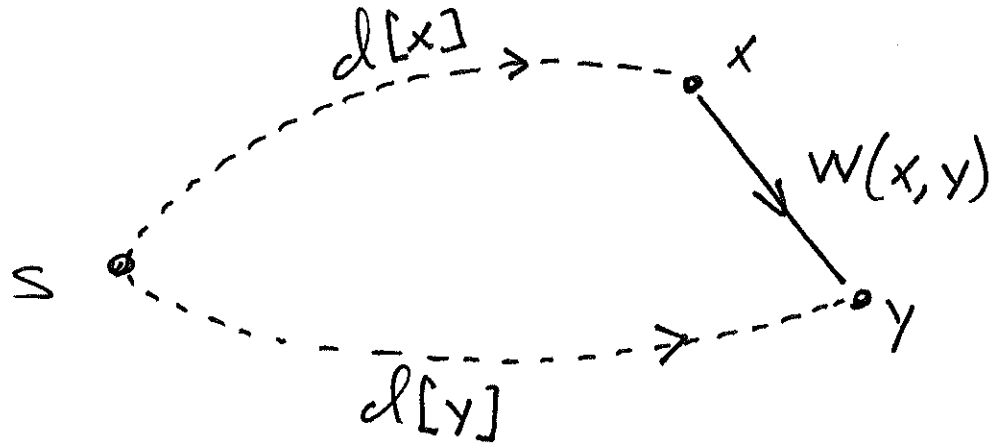
Initialize (G, s)

1. for all $x \in V$

   :

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

what does Relax(x,y) do?



note: Relax(x,y) changes only attributes of y.

after Relax(x,y) is called,

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

is true.

Lemma 1

Let $x \in V$ and suppose that after Initialize(G, s), some sequence of calls to Relax$(\cdot, \cdot)$ results in $d[x]$ being 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 Relaxation sequence.

Lemma 3 (Path Relaxation Property)

If

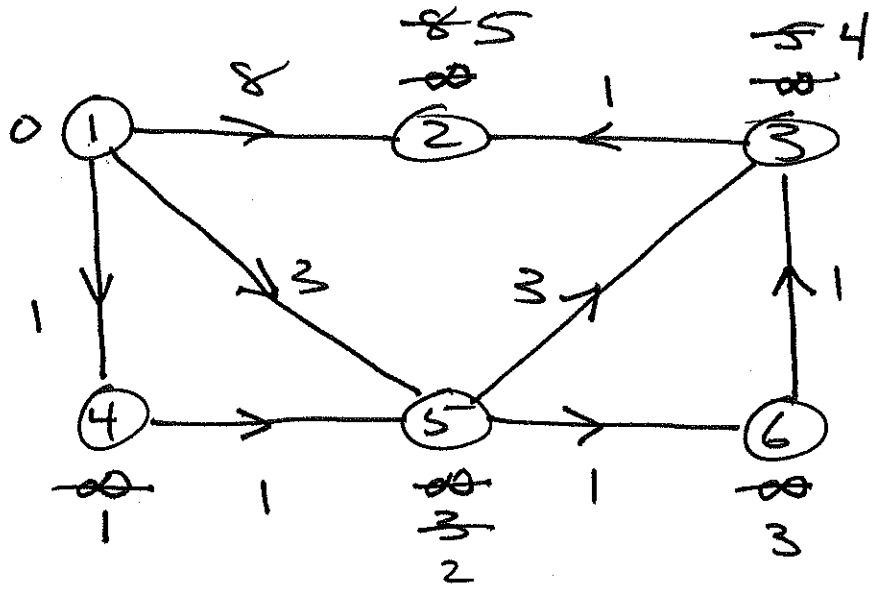
$P: s = x_0, x_1, x_2, \ldots, x_k$

is a min weight $s - x_k$ path, and its edges 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 relaxations that occur, even if interspersed with those above.

Ex. Dijkstra(G, 1) s=1



PQ: ~~1~~ ~~2~~ ~~3~~ ~~4~~ ~~5~~ ~~6~~

d: 0 ~~8~~ ~~5~~ 1 ~~3~~ 3
5 4 2