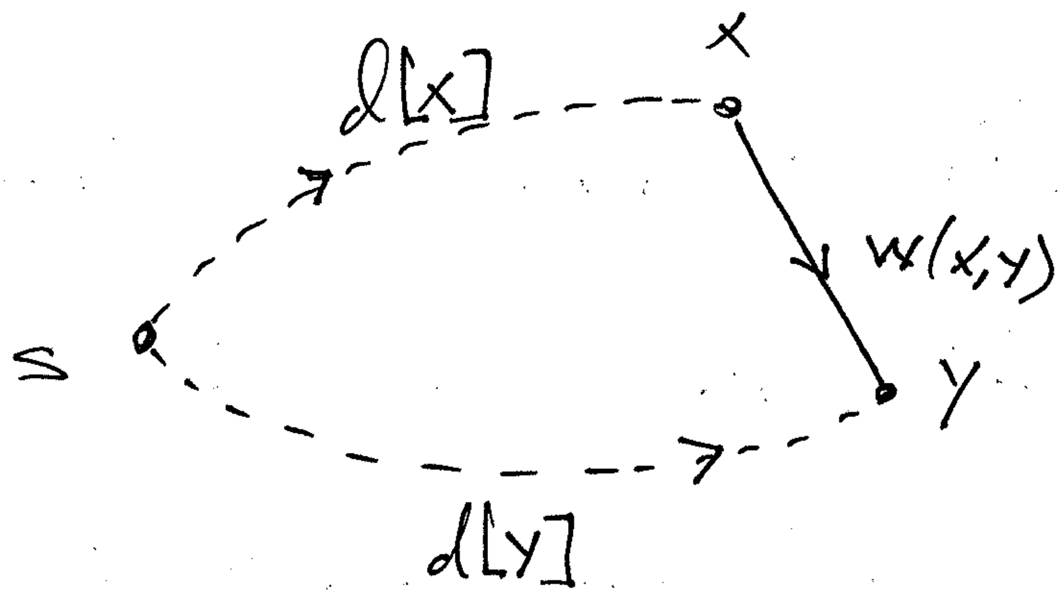
cse 101 6-4-26

Final: Wed. June 10 4:00-6:00 PM

SETS: closes Sun. 6/7 11:59 PM

Pa8: ext. 2 days (last)

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



<u>note</u>

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

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

- d-values only go down

Lemma 1

Let $x \in V(G)$ 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 $d[x]$.

Proof

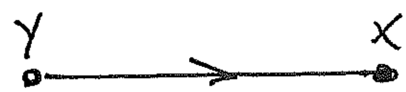
We use induction on the number of calls to Relax( ). Let n = # such calls in the 'relaxation sequence'.

If $n=0$ then the only vertex with finite d-value is s, so it must be that $x=s$. Indeed there is an s-s path of weight $d[s]=0$, namely the trivial path. The base case is therefore satisfied.

Let $n>0$ and suppose that for any vertex u, if $d[u]$ becomes finite after fewer than n relaxations, then G contains an s-u path of weight $d[u]$.

must show: if a seq. of n relaxations causes $d[x]$ to become finite, then G contains an s-x path of weight $d[x]$.

Now suppose a sequence of n relaxations causes $d[x]$ to become finite. Then some edge with x as terminus was relaxed in that sequence.

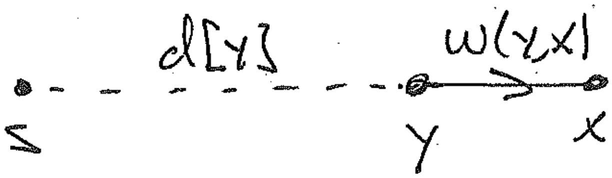


Call the origin of that edge y.

On the call to Relax(y, x), $d[x]$ was set to

$$d[x] = d[y] + w(y, x).$$

Since we suppose this number is finite, $d[y]$ must have been finite before the call. Thus $d[y]$ became finite after a sequence of fewer than n relaxations. By the induction hypothesis G contains an s-y path of weight $d[y]$



That path followed by edge (y,x) constitutes an s-x path in G of weight $d[x]=d[y]+w(y,x)$.

This completes the proof. ///

<u>Lemma 2</u>

After Initialize(G,s), the inequality

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

is maintained over any sequence of calls to Relax( ).

<u>Proof</u> (contradiction)

If $d[x] < \delta(s,x)$ were to become true after some relaxation sequence, then $d[x]$ would be finite, and by Lemma 1, G would contain an s-x path of weight $d[x]$. This contradicts the very definition of $\delta(s,x)$ as the weight of a <u>minimum weight</u> s-x path.

///.

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 the edges of P are relaxed in order:

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

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

Proof:
see lemma 24.15 p. 673.

(Induction on k = # edges in a shortest path.)

24.1

BellmanFord(G, s)

```
1.) Initialize(G, s)
2.) for i = 1 to |V|-1
3.)     for each (x,y) ∈ E
4.)         Relax(x,y)
5.) for each (x,y) ∈ E
6.)     if d[y] > d[x] + w(x,y)
7.)         return false
8.) return true
```

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

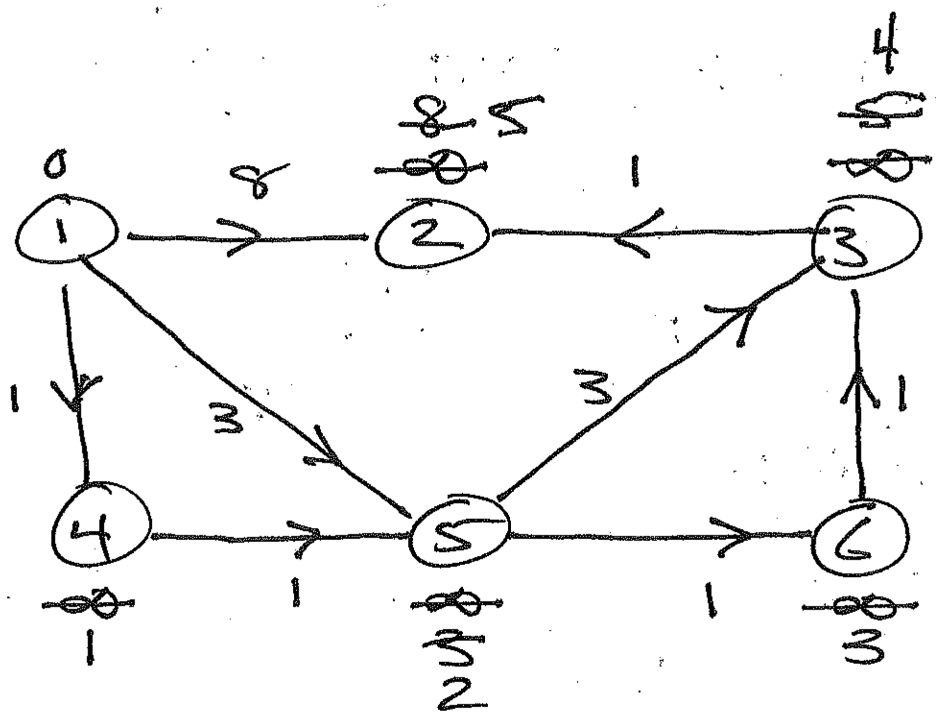
- Initialize : $\Theta(n)$
- Relax(·, ·) costs $\Theta(1)$
  each edge is relaxed $n-1$ times
  so loop 2-4 costs

$$\Theta(m(n-1)) = \Theta(mn)$$

- loop 5-7 costs $\Theta(m)$

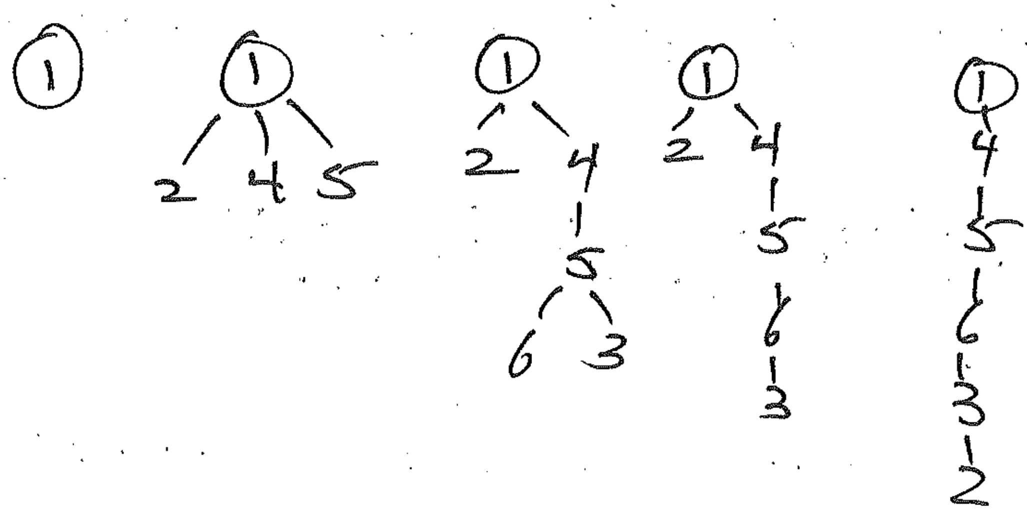
total cost : $\Theta(mn)$

Ex. Dijkstra : s = 1



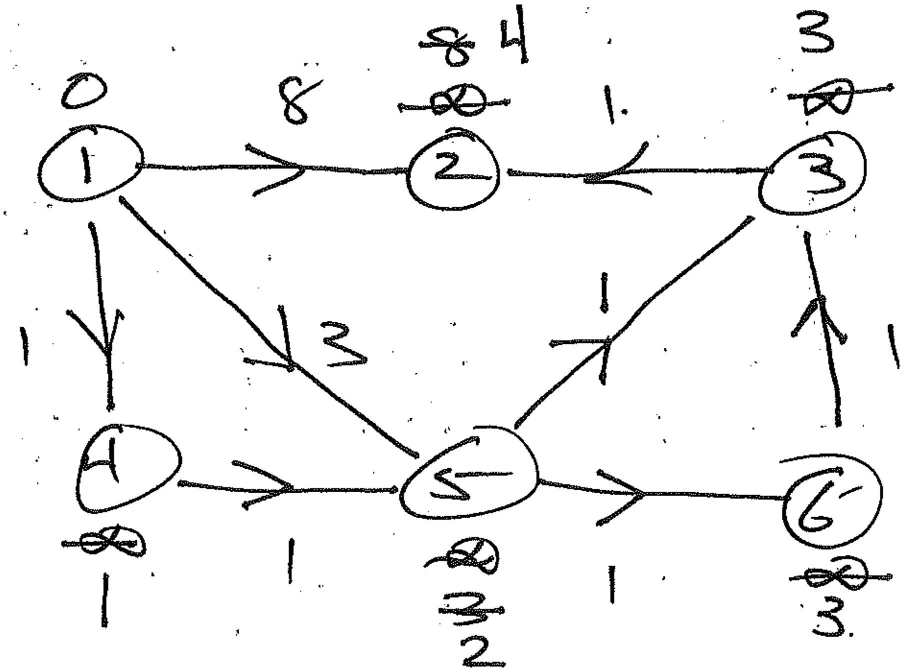
PQ :	~~1~~	~~2~~	~~3~~	~~4~~	~~5~~	~~6~~
d :	0	~~∞~~ ~~8~~ 5	~~∞~~ ~~5~~ 4	~~∞~~ 1	~~∞~~ ~~3~~ 2	~~∞~~ 3
p :	n	~~n~~ ~~1~~ 3	~~n~~ ~~5~~ 6	~~n~~ 1	~~n~~ ~~1~~ 4	~~n~~ 5

SP Tree



Ex. how to break ties?

s = 1 smallest vertex label



PQ	~~1~~	~~2~~	~~3~~	~~4~~	~~5~~	~~6~~
d	0	~~∞~~ ~~8~~ 4	~~∞~~ 3	~~∞~~ 1	~~∞~~ ~~3~~ 2	~~∞~~ 3
π	n	~~n~~ ~~1~~ 3	~~n~~ 5	~~n~~ 1	~~n~~ ~~1~~ 4	~~n~~ 5

24.3

Dijkstra(G, s)

1.) Initialize(G, s)
2.) $S = \emptyset$
3.) $Q = V$ // keys are d-values
4.) while $Q \neq \emptyset$
5.) $x = \text{ExtractMin}(Q)$ ← explain this
6.) $S = S \cup \{x\}$
7.) for all $y \in adj[x]$
8.) Relax(x, y)

observe: the call to Relax(·,·) contains an implicit call to DecreaseKey()

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

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

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

assume P.Q. implemented as a min-heap.

$\Theta((n+m)\log n)$ ← opposite Page

Ext. Min $\Theta(\log n)$

total : $\Theta(n \log n)$

Relax $\Theta(\log n)$

total $\Theta(m \log n)$

Runtime : $\Theta(m \log n + n \log n)$

$= \Theta((n+m) \log n)$

without P. Q.

Ext min $\Theta(n)$

total $\Theta(n^2)$

Relax $\Theta(1)$

total $\Theta(m)$

Runtime = $\Theta(m + n^2)$