1. Recall Shortest Path problems

S.S. ALL DEST
S.S. S. DEST
etc.

Assume we have a weighted directed graph. (can have negative wts)

We consider the Bellman-ford Algorithm what is the idea?

Relax all edges (|V|-1 times

what is Relax? if $d[v] > d[u] + w(u,v)$
replace.

```
Relax(u,v,w)
1. if d[v] > d[u] + w(u,v)
2.       d[v] ← d[u] + w(u,v).
```

```
Bellman-ford(V,E,s)
1. d[v] ← ∞  ∀v∈V
2. d[s] ← 0
3. for i ← 1 to |V|-1
4.       for each edge (u,v) ∈ E
5.             Relax(u,v,w).
```

Running Time? $\Theta(VE)$

for a dense graph? $\Theta(V^3)$.

If we do the Relaxation of all edges | more time
what can we determine? NWC.

Suppose we have a DAG (no cycles).
here we can get the order in which to relax the edges by a topological sort.

if v_1 is before v_2 before v_3 before v_4 in the topological sort., (Recall lecture from Mon 13/5). 2 WEEKS AGO.

what do we know about graph G.?

We know # of in edges from v_2 to v_1 or v_3 to v_2 or v_3 to v_1 etc.

RECALL: TOP. SORT $\Rightarrow$ if $(u,v) \in E$ then u comes before v in the top. order.

If we relax in topological order, we will get the shortest paths

1. TOPOLOGICAL SORT $\Theta(V+E)$
2. Relax edges in TOP. ORDER $\Theta(V+E)$

$\Theta(V+E)$.

] algorithm for shortest paths in a DAG

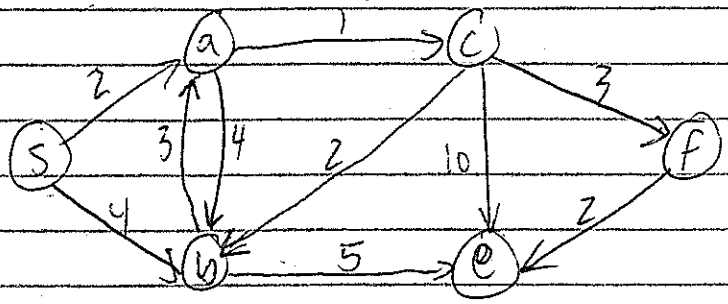
Expand 2.
- 2.1 $d[v] \leftarrow \infty$ if $v \in V$
- 2.2 $d[s] \leftarrow 0$
- 2.3 for each u in TOP ORDER
- 2.4 for each v adj to u
- 2.5 relax(u,v,w).

Recall Relax(u,v,w)
1. if $d[v] > d[u] + w(u,v)$
2. $d[v] \leftarrow d(u) + w(u,v)$

Recall Dijkstra's algorithm from CS226.

What D.S. does Dijkstra's use? What problem does it solve?
SSS? on weighted, directed graph (w/ non-neg. weights)

so $w(u,v) \geq 0 \quad \forall (u,v)$ edges.



PUT ALL ~~IN~~ PQ.

	PRED
~~s, 0~~	NIL
~~a, ∞ 2~~	s
~~b, ∞ 4~~	s
~~c, ∞ 3~~	a
e, ~~∞~~ 13 ~~9~~ 8	c b f
~~f, ∞ 6~~	c

will also keep track of predecessor.

CAN MAINTAIN PQ as an array of d's

s	a	b	c	e	f
0	~~∞~~	~~∞~~	~~∞~~	∞	∞
	2	4	3	~~13~~ 9 8	6

we extract min, get s, 0. look @ 2 edges $s \to a$, $s \to b$
both reduce $\infty \to 2$ & $\infty \to 4$ do decrease keys on both update predecessors of a, b to s.

extract min to get a, 2 (now s & a are done).
consider $a \to b$ & $a \to c$ $4 \not> 4+1$, $\infty > 2+1$.

250

for 11.28.2018

for SP. algorithms we have. for Directed Weighted Graphs:

1. BELLMAN-FORD : EDGE WTS CAN BE NEGATIVE
   Relax all edges V-1 times $\Rightarrow \Theta(VE)$
   can determine NWC if go one more turn & changes NO CYCLES,
2. DAG-SHORTEST-PATHS : REQUIRES ~~DAG~~, NEG WTS OK.
   TOP SORT
   RELAX in that order. $\Rightarrow \Theta(V+E)$

3. DIJKSTRA'S : REQUIRES WT NONNEG.
   Similar to PRIM's.
   what did PRIM's ALGORITHM SOLVE?
   KRUSKAL'S SOLVED IT TOO?

   MST on ~~an~~undirected, weighted, connected graph.

   DO YOU RECALL D.S. PRIM'S USED? PQ
   DIJKSTRA'S DOES TOO.

Show. 237, 238, 248, 251, 252, 253, 254, 255, 256, 257, 258

Dijkstra's is similar to Prim's for MST but Prim's doesn't ADD the weights.

```
Dijkstra's (V, E, s)
1. d[v] ← ∞ ∀v∈V
2. d[s] ← 0
3 Build a PQ on the d's
4. while PQ not empty
5.        u ← PQ.ExtractMin()
6.        for each v adj to u
7.                if d[v] > d[u] + w(u,v)
8.                        decrease key(v, d[u] + w(u,v))
9.                        v.pred ← u.
```

Prints doesn't do this.

Running Time?

1-3 $\Theta(V)$

4-9 happens V times

have V ExtractMins

6-9 goes E total times so we have E decrease keys.

So just Prim's RUNNING TIME depends on PQ implementation

	PQ UNSORTED ARRAY	PQ BIN. HEAP	PQ FIB. HEAP (Amortized)
V Extr. Mins	$V \cdot O(V)$	$V \cdot O(\lg V)$	$V \cdot O(\lg V)$
E Decrease Keys	$E \cdot O(1)$	$E \cdot O(\lg V)$	$E \cdot O(1)$
	$O(V^2 + E)$	$O(V \lg V + E \lg V)$	$O(V \lg V + E)$
	$= O(V^2)$	if connected $= O(E \lg V)$	

b/c E is $O(V^2)$

DENSE prefer Unsorted Array b/c $E \approx V^2$.
SPARSE " BIN. HEAP b/c $E \approx V$
MEDIUM " FIB. " .

Let's Consider solutions to ALL-PAIRS S.P. problem.

How can we use the Single Source S.P. algorithms to solve?

Any ideas? Assume Adj. List for edges (not adj. mtx).

recall: Bellman Ford $\Theta(VE)$

Dijkstra's $\Theta(V \lg V + E)$ using Fib. Heap for PQ.

Can do Repeated B-F. (once for each source V)
so $\Theta(V^2E)$

Can d Repeated Dijkstra's (if no neg. wt's.) $\Theta(V^2 \lg V + VE)$

We want to come up w/ something better (in Ram train) than these.

Consider an Adj. Matrix for these instead of Adj. List

Dynamic Programming Problems can often be formulated as shortest path graph problems.

S.P.'s

If $u \cdots v$ is a S.P. that goes through x

then $u \cdots x$ is shortest & $x \cdots y$ is shortest

$\Rightarrow$ OPTIMAL SUBSTRUCTURE