Lecture 30: Wednesday November 21, 2001

today

- single source shortest path: Dijkstra

announcements

- Assignment 8 due Monday November 26

shortest path problem variants

- single source?   or all pairs?
- graph: directed?   (or undirected)
- edges: weighted?   (or unweighted)
- weights: non-negative?   (or may have negative weights)
- directed: acyclic?   (or may have di-cycles)

undirected unweighted single-source

- distance(x,y): min'm number of edges in x-y path ∞ if none
- have seen: solved by BFS

directed, non-negative weighted, single-source

- distance(x,y): min'm sum of edge weights in x-y path ∞ if none
- solved by Dijkstra's algorithm (next)

Dijkstra's SSSP algorithm   [CLRS 24.3]

- greedy, similar to Prim/Dijkstra/Boruvka MST algorithm
- also works for undirected graphs
- also works for unweighted (make all weights 1)

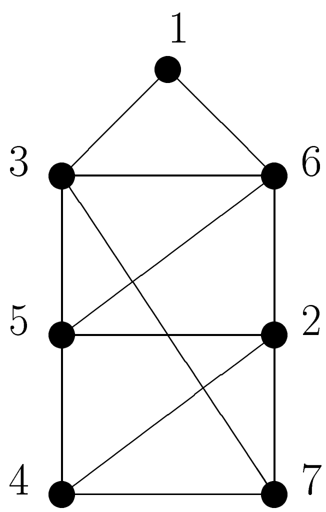
Dijkstra SSSP

```
Relax(u,v,w)        { shorter to v via (u,v)? { [CLRS p 586]
if d[v] >  d[u] + w(u,v) then
   d[v] <- d[u] + w(u,v)
   parent[v] <- u

InitSS(G,s)                         { [CLRS p 585]
for each vertex v in V[G] do
  d[v] <- infinity        { shortest (s,v) distance so far
  parent[v] <- NIL        { allows path construction
d[s] <- 0                 { s is source

Dijkstra(G,w,s)  { digraph, weights non-neg. [CLRS p595]
InitSS(G,s)
S <- empty     { vertices with d[v] final
Q <- V[G]      { min priority queue
while not isEmpty(Q) do
  u <- ExtractMin(Q)
  add u to S
  for each vertex v in Adj[u] do
    Relax(u,v,w)
```

weights	1	2	3	4	5	6	7
1:			04				
2:				11			31
3:							05
4:					03		14
5:		14	42			07	
6:	03	05	13				
7:							




```
*********
*trace  *
*SSSP(4)*
*********
```

v	d[v]	p[v]	in S?
1			
2			
3			
4	0	*	yes
5			
6			
7			

correctness of Dijkstra SSSP

- claim: at termination, d[v] is dist(s,v) for each v in V
- loop invariant: at start of Line 4, d[v] is dist(s,v) for each v in S

proof of loop invariant

- init'n: S empty, so invariant holds vaccuously
- next step $S = \{s\}$, and d[s]=0=dist(s,s), so invariant holds
- maintenance ?
- term'n: S=V, so invariant implies claim

proof of loop invariant: maintenance

- only change to S in loop: add u
- so must show: at loop body end, d[u] is dist(s,u)

showing d[u] is dist(s,u)

- consider any shortest s-u path $P = (s = v_0, v_1, \ldots, v_k = u)$
- $y = v_j$: first vertex in P not in S possibly $y = u$
- $x = v_{j-1}$ possibly $x = s$
- d[y] = dist(s,y) when u added to S why?
  - any subpath of a shortest path is a shortest path
  - so $(s, \ldots, x, y)$ is a shortest path
  - x is in S
  - so (x, y) was relaxed when $d[x] = dist(s, x)$
  - $d[v]$ never increases, so $d[x] = dist(s, x)$ still
  - so $d[y] \leq length(s, \ldots, y) = dist(s, y)$
- y precedes (or equals) u in shortest s, u path
- so dist(s,y) $\leq$ dist(s,u)
- y and u both in $V - S$ when u chosen, so $d[u] \leq d[y]$
- …but also $d[y] = dist(s, y) \leq dist(s, u) \leq d[u]$
- so $d[y] = dist(s, y) = dist(s, u) = d[u]$ ☺

Dijkstra SSSP run time

- same as PDB MST
- WC $\Theta(n^2)$ with priority queue: list imp'n
- WC $\Theta(m \lg n)$ with priority queue: heap imp'n