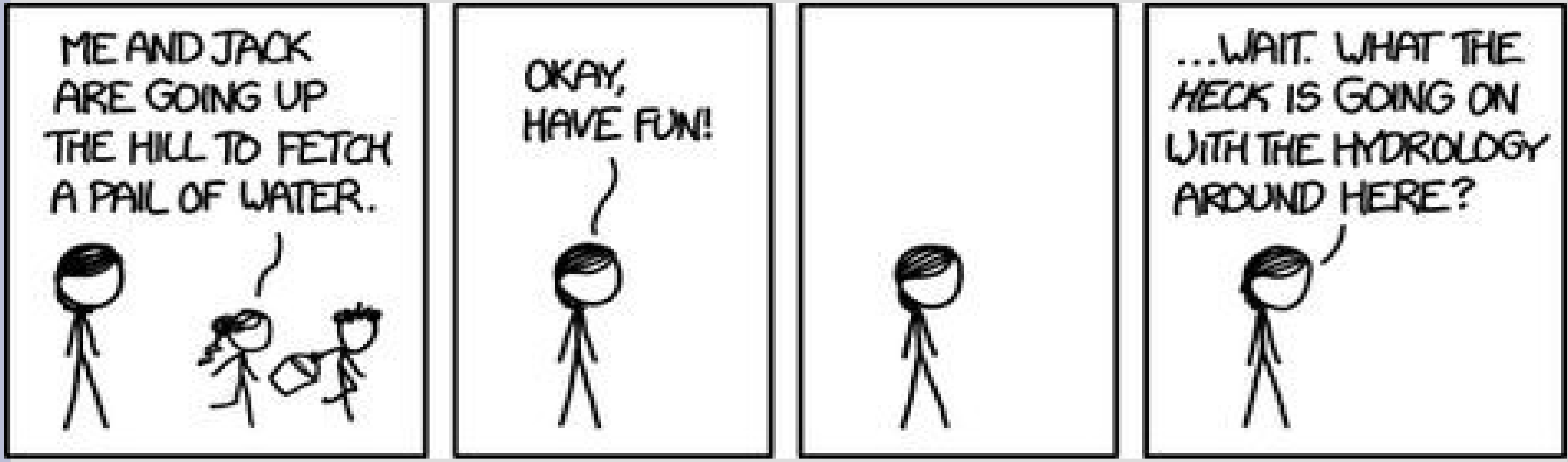
Weighted graphs



Announcements

HW 2 posted

Will post first part of programming assignment this weekend

Skipping Ch. 23 (geometic algs)

Relaxation

We will only do <u>relaxation</u> on the values v.d (min weight) for vertex v

Relax(u,v,w) (i.e. min() function)

if(v.d > u.d + w(u,v))

v.d = u.d+w(u,v)

v.π=u

Relaxation

Relaxation properties:

1. $\delta(s,v) \leq \delta(s,u) + \delta(u,v)$ (triangle inequality)
2. $v.d \geq \delta(s,v)$, v.d is monotonically decreasing
3. if no path, $v.d = \delta(s,v) = \infty$
4. if $\delta(s,v)$, when $(v.\pi).d = \delta(s,v.\pi)$ then relax($v.\pi$,v,w) causes $v.d = \delta(s,v)$
5. if $\delta(v_0,v_k) = p_{0,k}$, then when relaxed in order (v_0, v_1), (v_1, v_2), ... (v_{k-1},v_k) then $v_k.d = \delta(v_0,v_k)$ even if other relax happen
6. when $v.d = \delta(s,v)$ for all v in V, G_π is shortest path tree rooted at s

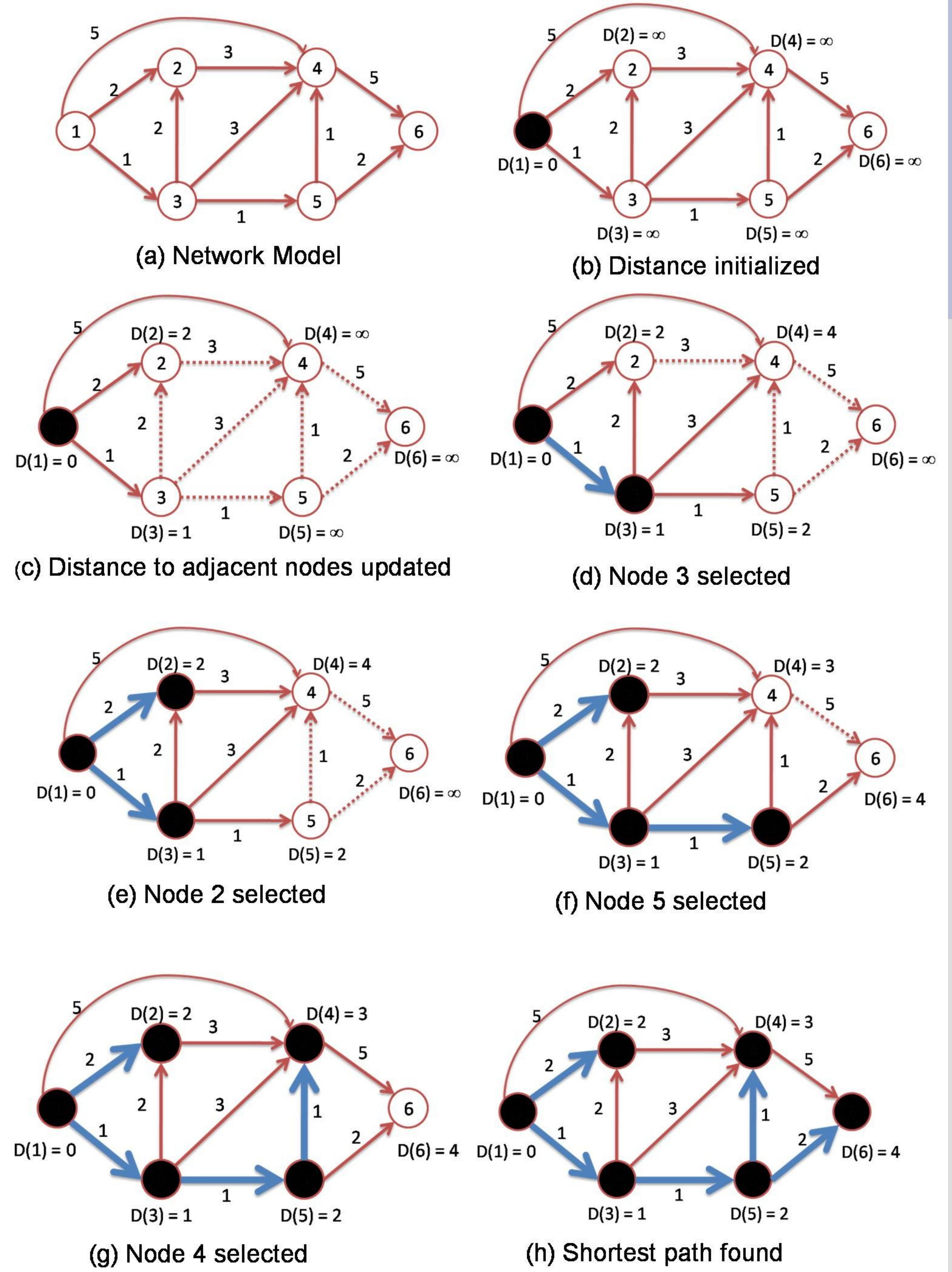
Dijkstra

Dijkstra's algorithm is the BFS equivalent for non-negative weight graphs

Dijkstra

```
Dijkstra(G,w,s)
initialize G from s
Q = G.V, S = empty
while Q not empty
   u = Extract-min(Q)
   S = S U {u}
   for each v in G.Adj[u]
      relax(u,v,w)
```

S optional


(a) Network Model
D(1) = 0
D(2) = ∞
D(3) = ∞
D(4) = ∞
D(5) = ∞
D(6) = ∞
(b) Distance initialized
D(1) = 0
D(2) = 2
D(3) = 1
D(4) = ∞
D(5) = ∞
D(6) = ∞
(c) Distance to adjacent nodes updated
D(1) = 0
D(2) = 2
D(3) = 1
D(4) = 4
D(5) = 2
D(6) = ∞
(d) Node 3 selected
D(1) = 0
D(2) = 2
D(3) = 1
D(4) = 4
D(5) = 2
D(6) = ∞
(e) Node 2 selected
D(1) = 0
D(2) = 2
D(3) = 1
D(4) = 3
D(5) = 2
D(6) = 4
(f) Node 5 selected
D(1) = 0
D(2) = 2
D(3) = 1
D(4) = 3
D(5) = 2
D(6) = 4
(g) Node 4 selected
D(1) = 0
D(2) = 2
D(3) = 1
D(4) = 3
D(5) = 2
D(6) = 4
(h) Shortest path found

Dijkstra

Runtime?

Dijkstra

Runtime:
Extract-min() run |V| times
Relax runs Decrease-key() |E| times
Both take O(lg n) time

So O((|V| + |E|) lg |V|) time
(can get to O(|V|lg|V| + E) using Fibonacci heaps)

Dijkstra

Runtime note:

If G is almost fully connected,

$|E| \approx |V|^2$

Use a simple array to store v.d

Extract-min() = $O(|V|)$

Decrease-key() = $O(1)$

total: $O(|V|^2 + E)$

Dijkstra

Correctness: (p.660)
Sufficient to prove when u added to S, $u.d = \delta(s,u)$

Base: s added to S first, $s.d=0=\delta(s,s)$

Termination: Loop ends after Q is empty, so V=S and we done

Dijkstra

Step: Assume v in S has $v.d = \delta(s,v)$
Let y be the first vertex outside S
on path of $\delta(s,u)$

We know by relaxation property 4,
that $\delta(s,y)=y.d$ (optimal sub-structure)

$y.d = \delta(s,y) \leq \delta(s,u) \leq u.d$, as $w(p)\geq 0$

Dijkstra

Step: Assume v in S has $v.d = \delta(s,v)$

But u was picked before y (pick min),

$u.d \leq y.d$, combined with $y.d \leq u.d$

$y.d=u.d$

Thus $y.d = \delta(s,y) = \delta(s,u) = u.d$