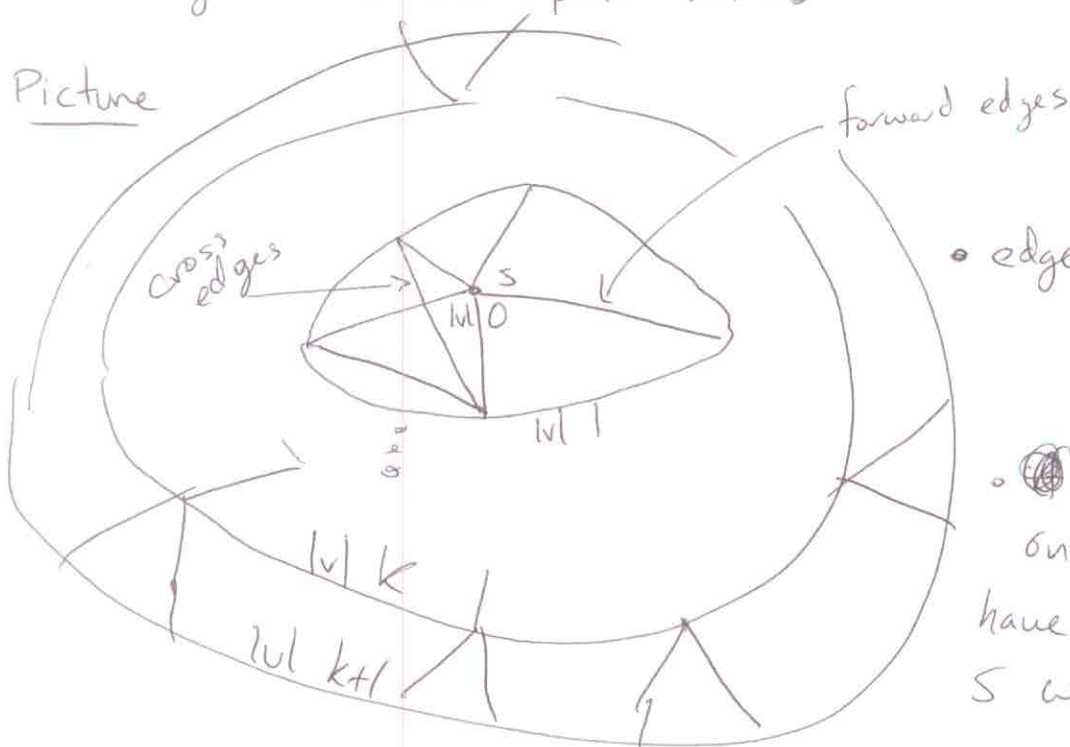


Agenda

- 1] BFS as a shortest path alg.
- 2] Dijkstra
- 3] • Which road to add?

1] BFS (Breadth-First Search)

• Defn - BFS ~~is~~ on an unweighted ~~an undirected~~ $G=(V,E)$ starting at $s \in V$, assigns each node $v \in V$ a $\text{level}(v) = \# \text{ of edges in shortest path from } s$.

Picture

Ex : 6-degrees of Kevin Bacon

Consider the friend graph of the world

- $V = \{ \text{vertex for every person ever to live in the world} \}$
- $E = \{ (u,v) : u \text{ and } v \text{ are friends (or know each other)} \}$

Say you are node $x \in V$. How far are you from Kevin Bacon? Use BFS! What level is KB in $\text{BFS}(G,x)$?

[2] Dijkstra

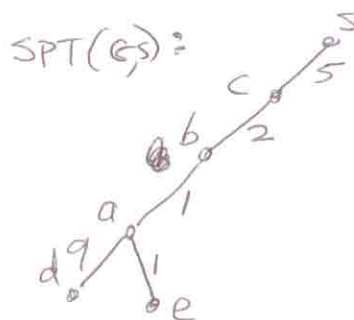
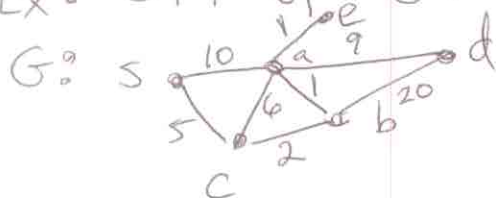
• What about weighted graphs?

• Dijkstra ~~finds~~ finds the following

Input: $G=(V,E)$; start node s ; weights $w_e \geq 0 \forall e \in E$.

Output: Shortest Path Tree rooted at s to all $v \in V$.

Ex: SPT of G :



• Note: Bellman-Ford finds SPs when w_e is arbitrary, but no negative cycles in G . Why no neg. cycles?

• ~~Dijkstra~~ Dijkstra uses a priority queue + BFS-like search.

[3] Which road to add? (4.20)

Input: $G=(V,E)$ cities & roads; $w_e \geq 0$ length of road; $s, t \in V, E'$

Output: Find $e' \in E'$ whose addition to G could result in the largest decrease in SP distance between s and t .

Idea: Learn $SPT(G,s)$ and $SPT(G,t)$.

$\downarrow$
 $\delta_s(v)$ = length SP from s to v

$\downarrow$
 $\delta_t(v)$

For $e'=(u,v) \in E'$
Compare $\delta_s(t), \delta_t(s), \delta_s(u) + w_{e'} + \delta_t(v)$
and $\delta_s(v) + w_{e'} + \delta_t(u)$. Find smallest.

Alg Road(G, l_e, E', s, t)

3

- 1) Run Dijkstra(G, l_e, s) and Dijkstra(G, l_e, t) to get $\delta_s(v)$ and $\delta_t(v)$ (as defined earlier)
- 2) $\text{min_sp} = \min(\delta_s(t), \delta_t(s))$ and $\text{ret_edge} = \emptyset$
- 3) For each $e' = (u, v) \in E'$
 - if $(\delta_s(u) + l_{e'} + \delta_t(v) < \text{min_sp})$ then $\text{min_sp} = \delta_s(u) + l_{e'} + \delta_t(v)$ and $\text{ret_edge} = e'$
 - if $(\delta_s(v) + l_{e'} + \delta_t(u) < \text{min_sp})$ then $\text{min_sp} = \delta_s(v) + l_{e'} + \delta_t(u)$ and $\text{ret_edge} = e'$
- 4) return ret_edge .

RT: $O((|V| + |E|) \log |V|)$ for Dijkstra with binary heap.

For loop is $O(|E'|)$ so $O((|V| + |E|) \log |V| + |E'|)$

Correctness Claim: Road($\cdot$) returns e' iff e' is the road whose addition minimizes the SP dist from s to t .

" $\Rightarrow$ " Assume Road($\cdot$) returns $e' = (u, v)$. Then either $\delta_s(v) + l_{e'} + \delta_t(v)$ or $\delta_s(v) + l_{e'} + \delta_t(u)$ is smallest of all E' , both of which are ~~paths~~ from s to t . ~~So, it is the new SP for~~ They are also both SPs since δ_s and δ_t are also $\sim$ SPs.

" $\Leftarrow$ " Let $e' = (u, v)$ be the road whose addition minimizes SP dist between s, t . Then $\delta_s(u) + l_{e'} + \delta_t(v)$ or $\delta_t(u) + l_{e'} + \delta_s(v)$ must be the ^{length of the} shortest ~~path~~ between s, t . This is what Road($\cdot$) tests for and minimizes.