Lecture 7 Paths in graphs II Dec. 30, 2020

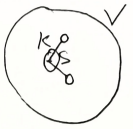
base: estimates

$v_1, v_2, \cdots, v_n$

dist[v_1] dist[v_2] dist[v_n]

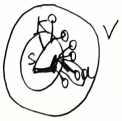
invariant

base case:



$K = \{s\}$

inductive step



~~1. shortest path~~

~~2. extend K~~

subset $K \subseteq V$

$\forall v \in K$ dist[v] = dist(s, v)

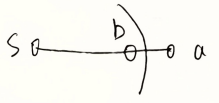
Inductive step:

how to extend K? (Extend a)

vertex in $V \setminus K$ that is closest to s.

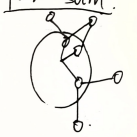
claim: a is 1-edge extension from K.

by contradiction: if b is 1-edge extension from K then edge $(b \to a) > 0$, then b is the shortest. (contradiction).



Shortest path:

pick the smallest path sum.



Dijkstra's Algorithm:

dist[s] = 0, dist[$V \setminus s$] = ∞

K = { } use priority queue

while ($K \neq V$) (queue $\neq \phi$)

pick $v \in V \setminus K$ with smallest dist[ ], (v = queue.poll())

add v to K

for all $(v, w) \in E$:

if dist[w] > dist[v] + l(v, w)

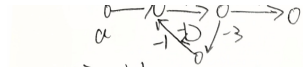
dist[w] = dist[v] + l(v, w).

change priority

Runtime:

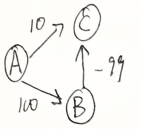
inserts	O|V|	binary heap
delete min	O|V|	binary heap
decrease keys (change priority)	O|E|	binary heap
overall	O((|V|+|E|) log |V|)	O(log|V|)

The case of negative edges:



ill-defined with negative cycles

⇒ look on graphs without negative cycles.



A ~~~ C in A->B->C

Update [u,v] ≡ dist[v] = min { dist[v], dist[u] + t(u,v) } "relax"

① update is safe

dist[u] ≥ d(s,u) dist[v] ≥ d(s,v)

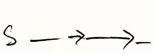
⇓

new dist[v] ≥ d(s,v)

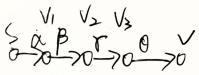
② if dist[u] is correct, then update [u,v] makes dist[v] correct

(Not for negative cases)

~~negative~~:

① 

then update [u,v] makes dist[v] correct



update (e_7)
update (e_{15}) — update (α)
update (e_3)
⋮ — update (β)

So we can update edge meaningfully.

Claim: if the sequence contains update (α), β, γ, θ, then dist[v] is correct.

① update (α) makes v_1 shortest

update (θ) makes v shortest

② update is safe, won't change other results.

---

e_1 e_2 ... e_3 ...

1: update (e_1), update (e_2) update (e_3) ... (e_i)

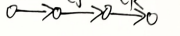
2: update (e_1) update (e_2) update (e_3) ... (e_j)

⋮

(e_k)

($|V|-1$) times makes every sequence found

$|V|-1$ times



for i = 1, ..., |V|-1:
  for edge: edge_set:
    update (edge)

T(n) = O(|V|·|E|)

[illegible]