Shortest Paths

Given weighted dir. graph $G=(V,E)$, $s,t \in V$,
$w: E \to \mathbb{R}^+$

find path p from s to t
minimizing $\sum_{e \in p} w(e)$

Special Case 1: Unweighted

use BFS in $O(m+n)$ time
solves single-source shortest paths (SSSP)
(s to v $\forall v \in V$)

Special Case 2: DAG

by dyn prog., also solves SSSP

Define subproblems: $\forall v \in V$
$d(v)$ = shortest-path dist. from s to v

Ans: $d(t)$

Base case: $d(s) = 0$

Recursive formula:

take choices over last edge

$$\boxed{d(v) = \min_{u:(u,v)\in E} \left(d(u) + w(u,v) \right)}$$

Evaluation order:

topological order!

& $O(m+n)$ time

Runtime:

$$O\left(n + \sum_{v \in V} \text{in-deg}(v) \right)$$

$$= \boxed{O(n+m)}$$

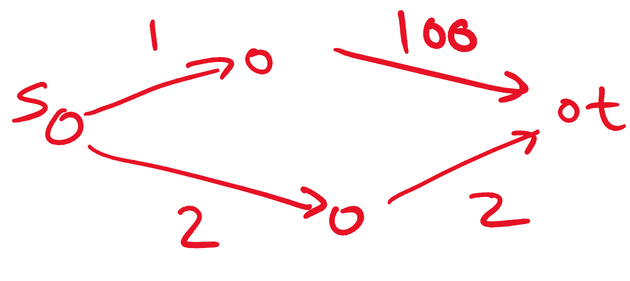
Rmk:
- fails for general graphs
- works for negative wts

Dijkstra's Alg'm (1959)

- assume no negative wts
- solves SSSP

  compute $d[v]$ = shortest-path dist from s to v $\forall v \in V$.

idea: smart greedy!



- consider vertices in increas. order of dist from s.

High-Level Version:

// maintain set S of "known" vertices

$S = \{s\}$, $d[s] = 0$

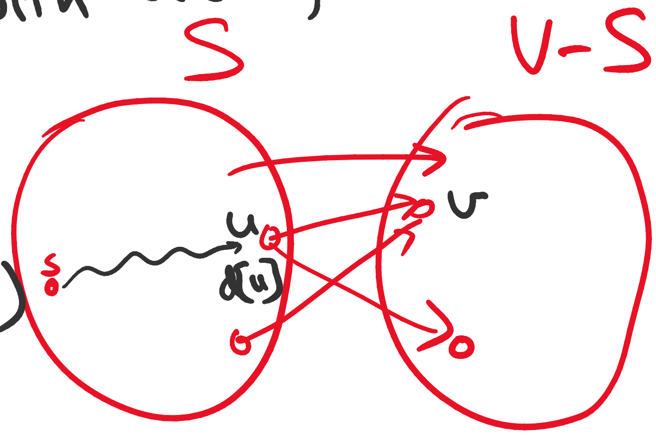
while $S \neq V$ {

  pick edge (u,v) with $u \in S$, $v \in V - S$
  minimizing
  $d[u] + w(u,v)$

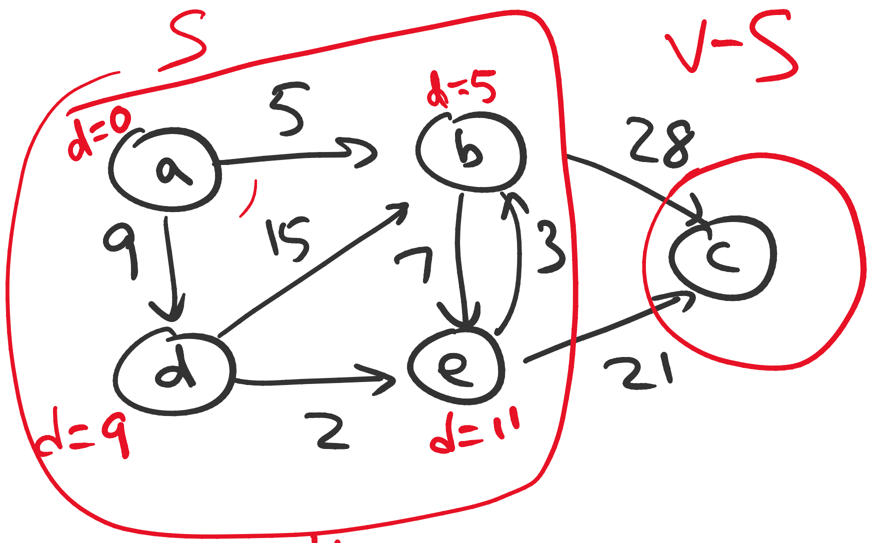
  set $d[v] = d[u] + w(u,v)$

  insert v to S

}



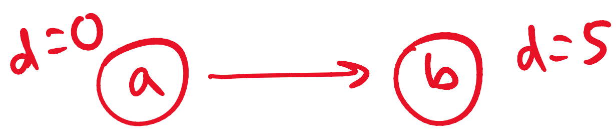
Ex



$s = a$ $t = c$

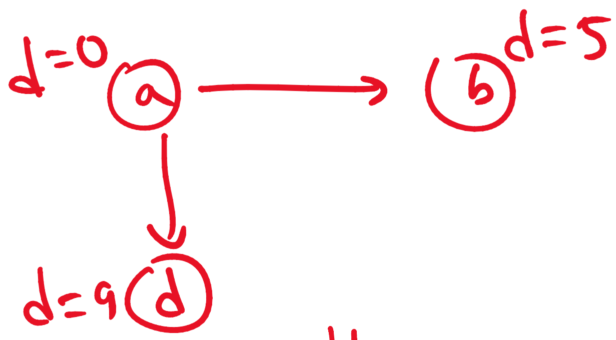
$0 + 5$
$0 + 9$

select (a,b)

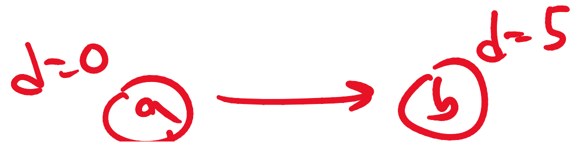


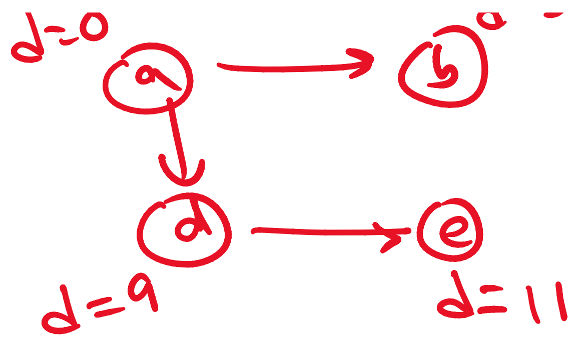
$0 + 9$
$5 + 7$
$5 + 28$

select (a,d)



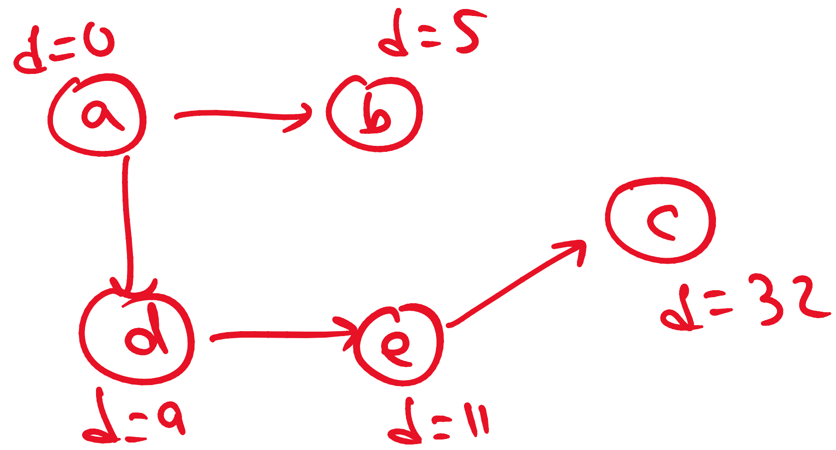
$9 + 2$
$5 + 7$
$5 + 28$





5+28

11+21



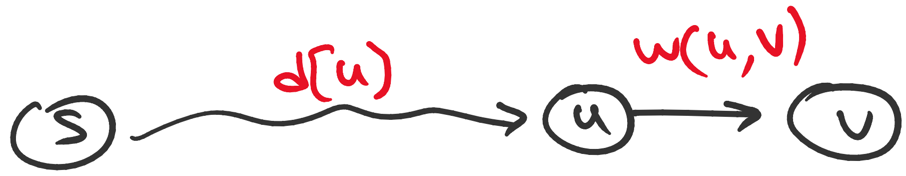
Correctness Pf:

Claim Assume all d[u] values are correct $\forall u \in S$.

If (u,v) is edge from S to V-S with smallest d[u] + w(u,v),

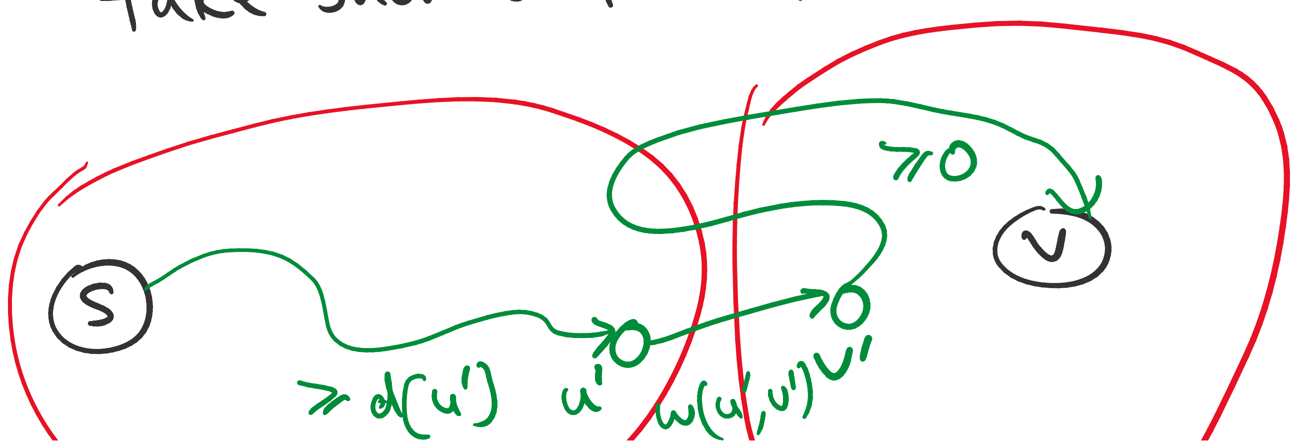
then min dist(s,v) = d[u] + w(u,v).

Pf: (≤) there is a path of dist d[u] + w(u,v)



$\Rightarrow$ min-dist(s,v) ≤ d[u] + w(u,v)

(≥) take shortest path p^* from s to v.



p^* must have some edge (u',v') from S to $V-S$

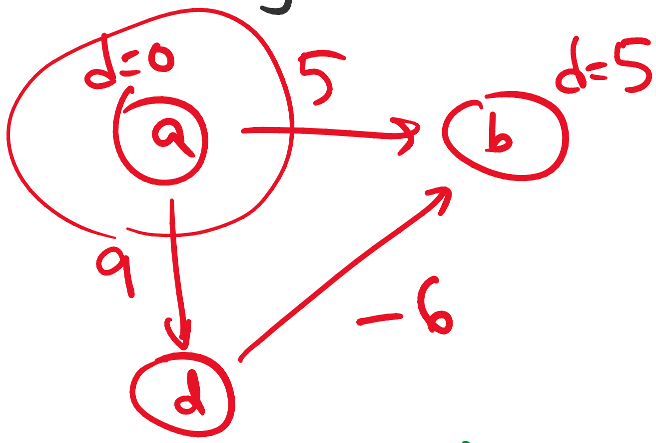
$$\Rightarrow \quad \text{min-dist}(s,v) \geq \underbrace{d[u'] + w(u',v')}_{} + 0$$

$$\geq d[u] + w(u,v)$$

. D

Rmk - not correct for negative wts

Counterex



$s = a$

Implementation: (Detailed Vers.)

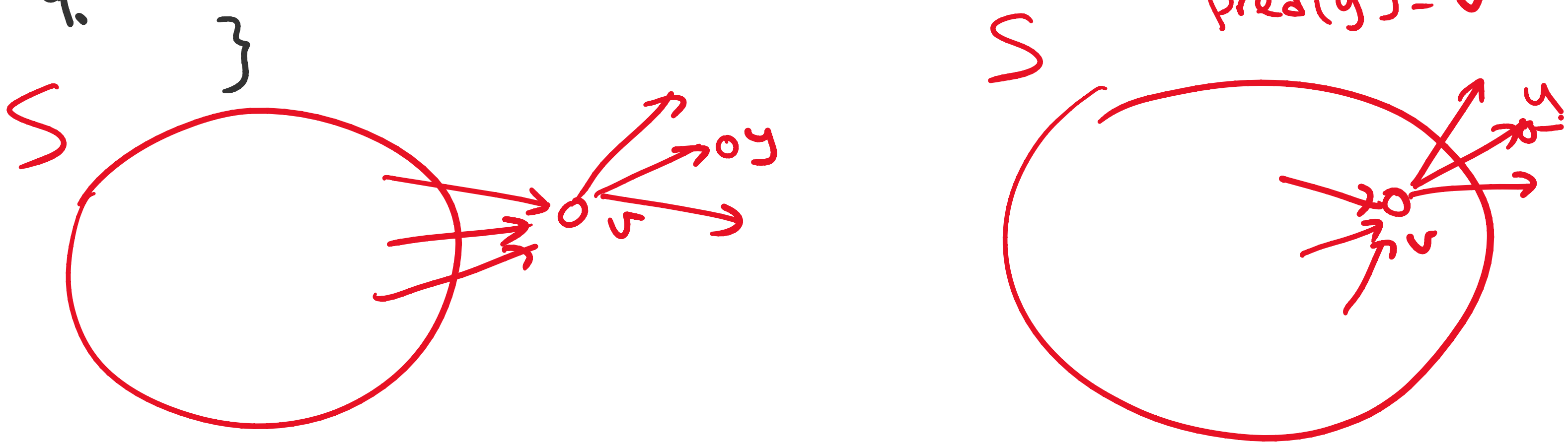
// maintain $key[v] = \min_{u \in S} (d[u] + w(u,v)) \quad \forall v \in V-S$

// $Q = V - S$

1. $Q = V$
2. $key[v] = \infty \quad \forall v \in V - \{s\}$, $key[s] = 0$
3. while $Q \neq \emptyset$ {
4. pick $v \in Q$ with smallest key
5. $d[v] = key[v]$
6. remove v from Q
7. for each out-neighbor y of v

6. ...
7. for each out-neighbor y of v
8. if $y \in Q$ and $d[v] + w(v,y) < key[y]$
9. $key[y] = d[v] + w(v,y)$, $pred[y] = v$

}



BEFORE　　　　AFTER

Runtime:

option 1: no data structure

line 4　$O(n)$ time $\leftarrow$

line 6　$O(1)$

line 9　$O(1)$

$\Rightarrow$ total time

$$O\Big(n \cdot n + \underbrace{\sum_{v \in V} \text{out-deg}(v) \cdot 1}\Big)$$

$$= O(n \cdot n + m \cdot 1)$$

$$= \boxed{O(n^2).}$$

option 2: heap / priority queue

line 4　$O(\log n)$

line 6　$O(\log n)$　(delete-min)

line 9　$O(\log n)$　(change-key)

$$\Rightarrow O(n \cdot \log n + \underline{m \cdot \log n})$$

$$\Rightarrow O(n \cdot \log n + m \cdot \log n)$$

$$= \boxed{O((m+n)\log n)}$$