Dijkstra's Algorithm

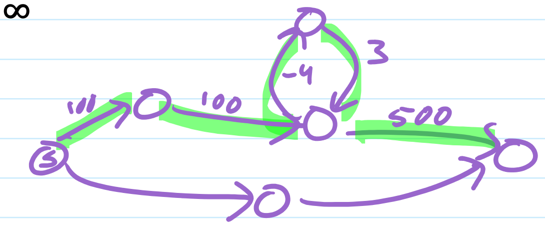
PRE: no negative weight edges

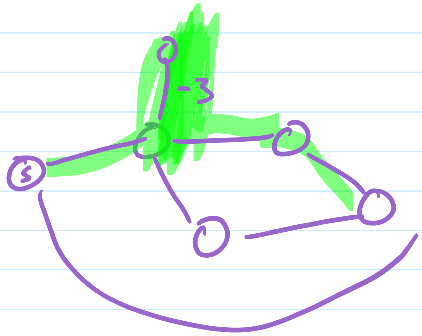
POST: d gives total weight of shortest paths, pred gives edges in shortest paths (∞ to mean unreachable)

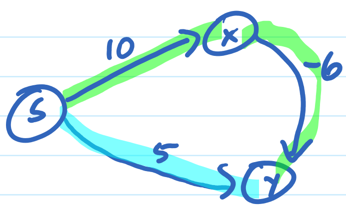
```
for each v
  color[v], pred[v], d[v] ← IN_QUEUE, NIL, ∞
d[s] ← 0

Q ← new PriorityQueue(d)

while Q not empty
  u ← dequeue(Q)
  for each outneighbor v of u
    if color[v] = IN_QUEUE and d[v] > d[u] + w(u, v)
      change_priority(Q, v, d[u] + w(u, v))
      d[v] ← d[u] + w(u, v)
      pred[v] ← u
  color[u] ← DONE
```







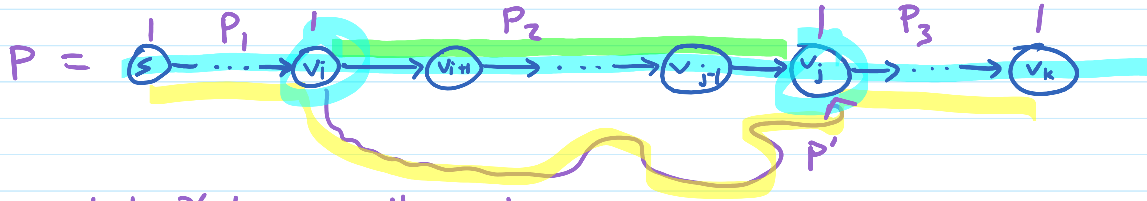
Think of an application where you want to minimize the sum of weights along your path and negative weight edges make sense.

Single Destination vs All Destinations

Optimal Substructure

If $s, v_1, v_2, \ldots, v_k$ is a shortest path $s \rightsquigarrow v_k$ then

for each v_i, v_j $\quad v_i, v_{i+1}, \ldots, v_j$ is shortest path $v_i \rightsquigarrow v_j$



Let P' be any other path $v_i \rightsquigarrow v_j$

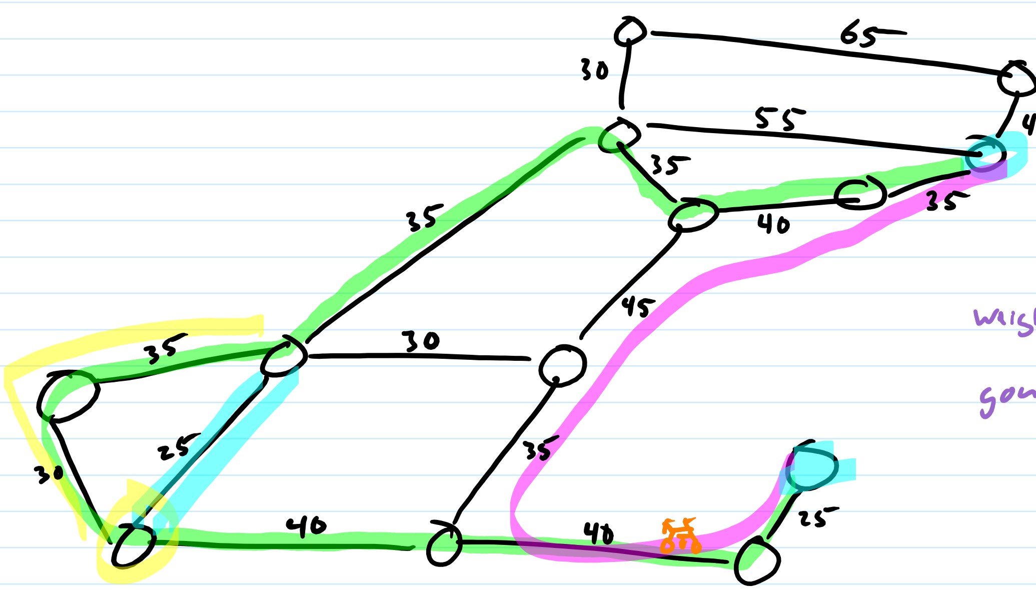
ℓ = total weight along path

$$\cancel{\ell(P_1)} + \ell(P_2) + \cancel{\ell(P_3)} = \ell(P) \le \cancel{\ell(P_1)} + \ell(P') + \cancel{\ell(P_3)}$$

Weaker: If $P = s, v_1, \ldots, v_i, \ldots, v_j, \ldots, v_k$ and P_{ij} is best path $v_i \rightsquigarrow v_j$ then

let $P' = s, v_1, \ldots, v_i, P_{ij}, v_j, \ldots, v_k$

$\ell(P') \le \ell(P)$



weights = speed limit

goal = minimize max weight edge

adding edges to a path can't make it better $\ell(P_1) \leq \ell(P_1 P_2)$

Dijkstra's Algorithm

```
for each v
  color[v], pi[v], d[v] ← IN_QUEUE, NIL, ∞
d[s] ← 0

Q ← new PriorityQueue(d)
S ← empty

while Q not empty
  u ← dequeue(Q)
  S ← S union {u}
  for each outneighbor v of u
    if color[v] = IN_QUEUE and d[v] > d[u] + w(u, v)
      change_priority(Q, v, d[u] + w(u, v))
      d[v] ← d[u] + w(u, v)
      pi[v] ← u
  color[u] ← DONE
```

a) $|S| \geq 1 \rightarrow s \in S$

b) S, Q partition the vertices

c) $|Q| = n - \#$ iterations of loop

d) for $v \in S$ $d[v] = \delta(s,v)$ — tot weight of shortest path $s \rightsquigarrow v$

$\pi[v]$ = next-to-last on that path

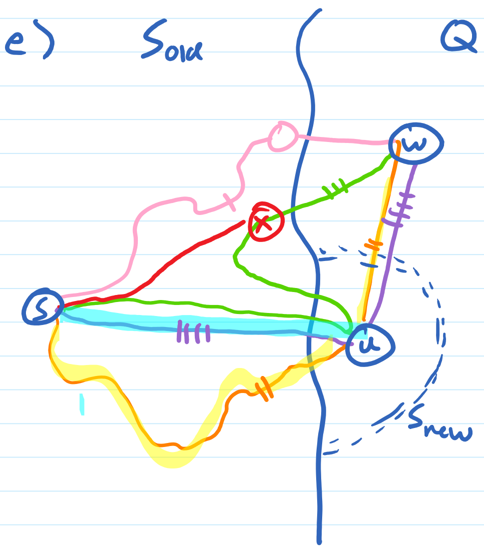
e) for $v \in Q$ $d[v]$ = tot weight of shortest path $s \overset{S}{\rightsquigarrow} v$ — s to v using intermediate vertices in S

$\rightarrow \pi[v]$ = next-to-last on that path

f) for all $v \in Q$, $d[v]$ = priority of v in Q

g) for all v, $\pi[v] = NIL$ or $\pi[v] \in S$

e) S_{old} Q



code : $d_{new}[w] = \min(d_{old}[w], \ell(\text{—|||—}))$

(goal : show all paths $s \overset{S_{new}}{\rightsquigarrow} w$ are at least as long as ·)

$\ell(\text{—|—}) \overset{\text{INV e}}{\geq} d_{old}[w] \overset{\min}{\geq} d_{new}[w]$

$\ell(\text{—||—}) = \ell(s \overset{||}{\rightsquigarrow} u) + w(u,w)$

$\overset{\text{INV e}}{\geq} d_{old}[u] + w(u,w) \overset{\min}{\geq} d_{new}[w]$

$\ell(\text{—|||—}) = \ell(s \overset{|||}{\rightsquigarrow} x) + w(x,w)$

$\geq \ell(s \rightsquigarrow x) + w(x,w)$

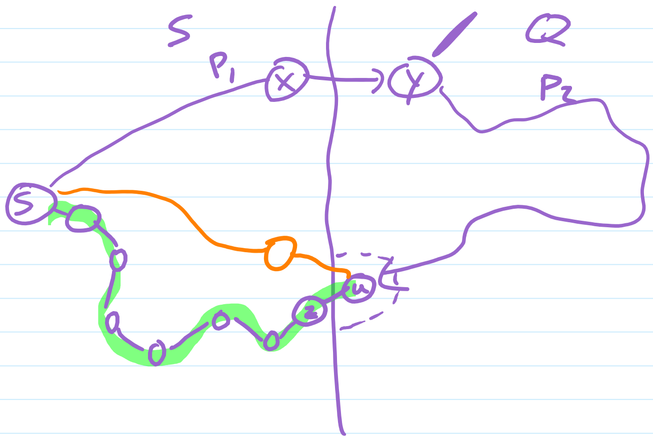
$\geq d_{old}[w]$

$\geq d_{new}[w]$

d) $d[v]$ is total weight of a shortest path $s \rightsquigarrow v$ (for all $v \in S$)

(min tot weight)

S | Q let P be a path



let P be a path $s \rightsquigarrow u$ using int verts in Q

Let y be 1st vertex on P in Q

$d[u] \le d[y]$ dequeue

$\le \ell(P_1)$

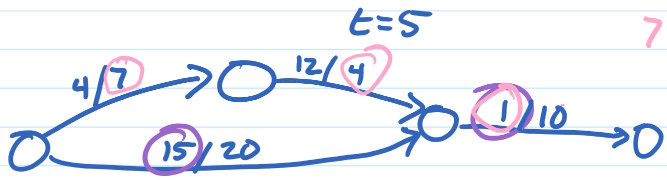
$\le \ell(P_1) + \ell(P_2)$ no neg weight edges

$= \ell(P)$

Bike Change

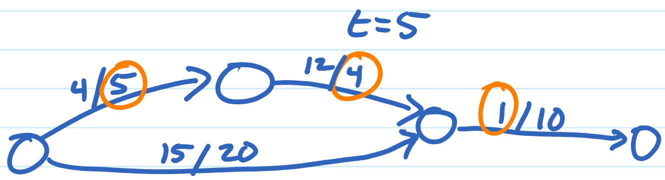
Given directed graph weighted with travel times using two modes $w_1(u,v) = $ time in mode 1
$w_2(u,v) = $ time in mode 2

with ability to change mode at any vertex with time penalty t,
find shortest paths from source s



$7+4+5+1 = 17$

$15+1 = 16$



$5+4+5+1 = 15$