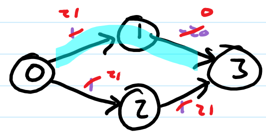
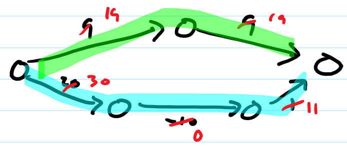
Negative Weights

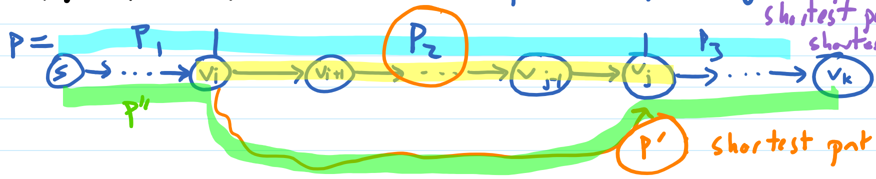
Negative weights → Dijkstra





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

for each v_i, v_j: $v_i, v_{i+1}, \ldots, v_j$ is shortest path $v_i \rightsquigarrow v_j$ — subpaths of shortest paths are shortest paths



Suppose $v_i, v_{i+1}, \ldots, v_j$ not shortest path $v_i \rightsquigarrow v_j$

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

$$\ell(P_2) \le \ell(P') \le \ell(P_2)$$

Want $d_s(n-1, v)$

Let $d_s(i, v)$ = total weight of min-weight path $s \rightsquigarrow v$ using $\le i$ edges

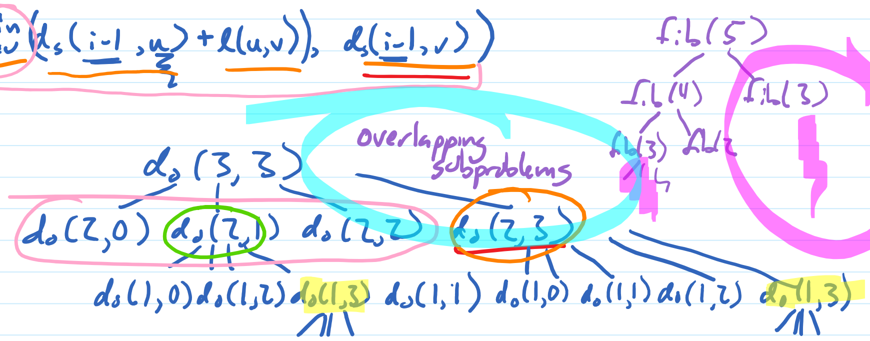
base cases

$$= \begin{cases} 0 & \text{if } v = s \\ \infty & \text{if } i = 0 \text{ and } v \ne s \quad (\infty \text{ to mean no path}) \\ \min\left(\min_{u \ne v}\left(d_s(i-1, u) + \ell(u,v)\right), d_s(i-1, v)\right) \end{cases}$$

Memoize

SHORTEST-PATHS(s, i, v)

- if $v = s$ return 0
- else if $i = 0$ return ∞
- else if (i, v) in Memo return Memo$[i, v]$
- else
  - dist ←
  - Memo$[i, v]$ ← dist
  - return dist



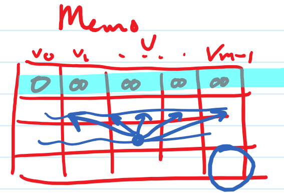
total time = # subproblems · time to solve each (given solns to subproblems)

$= n \cdot n \cdot n$

$O(n^3)$

SHORTEST-PATHS(n, s) — dynamic programming — $s = v_0$

Memo



max # edges

- $M \leftarrow n \times n$ array
- base case: for $v = 0$ to $n-1$: $M[0, v] \leftarrow \infty$
- $M[0, s] \leftarrow 0$

row-by-row: for $i = 1$ to $n-1$
- for $v = 0$ to $n-1$
  - $M[i, v] = \min\left(M[i-1, v], \min_{u = 0 \ldots n-1} M[i-1, u] + \ell(u, v)\right)$

$\Theta(n^3)$

$\Theta(n^2)$ space

SHORTEST-PATHS(n, s)

$M \leftarrow n \times n$ array
for $v = 0$ to $n-1$ $\quad M[\cancel{0}, v] \leftarrow \infty$
$M[\cancel{0}, s] \leftarrow 0$

$\Theta(n^3)$ time
$\Theta(n)$ space

for $i = 1$ to $n-1$
for $v = 0$ to $n-1$
$M[\cancel{i}, v] = \min\left(M[\cancel{i-1}, v], \min\limits_{u=0 \ldots n-1} M[\cancel{i-1}, u] + \ell(u,v) \right)$

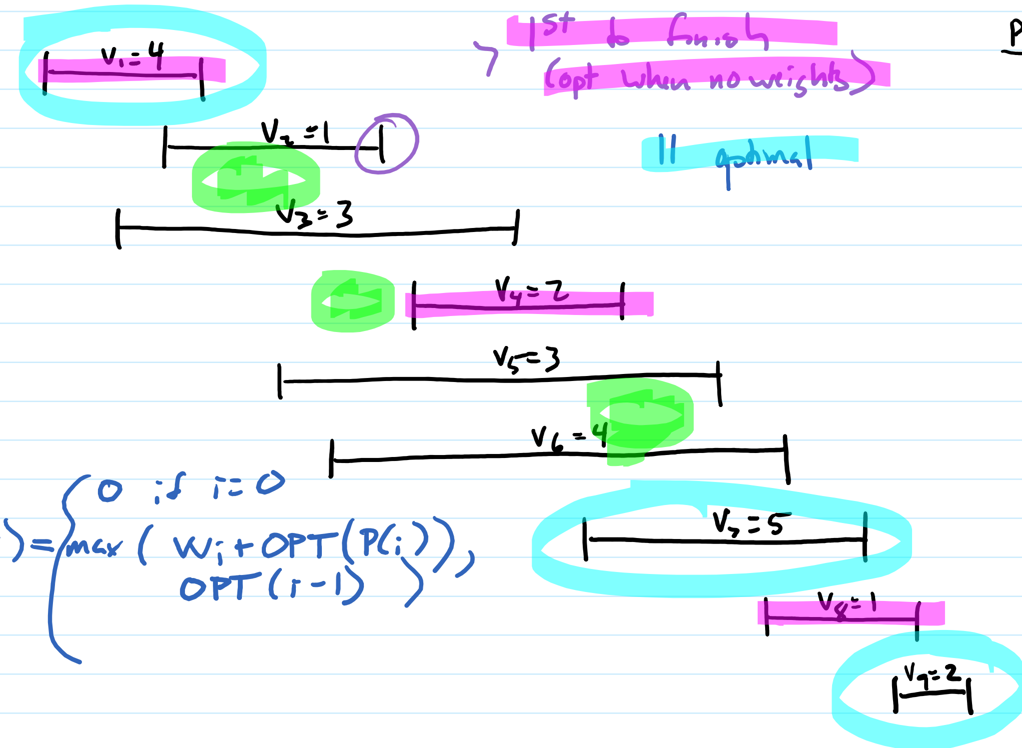
SHORTEST-PATHS(n, s)

$M \leftarrow n \times n$ array
for $v = 0$ to $n-1$ $\quad M[0, v] \leftarrow \infty$
$M[0, s] \leftarrow 0$

$\Theta(n \cdot m)$

for $i = 1$ to $n-1$
~~for $v = 0$ to $n-1$~~ for each edge (u, v)
$M[i, v] = \min\left(M[i-1, v], \cancel{\min\limits_{u=0 \ldots n-1}} M[i-1, u] + \ell(u,v) \right)$

Weighted Interval Selection



→ 1st to finish (opt when no weights)

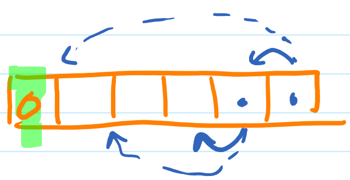
|| optimal

$P(i)$ = index of last to finish before i starts

0
0
0
2
1
1
3
5
7

$$OPT(i) = \begin{cases} 0 & \text{if } i = 0 \\ \max\left(w_i + OPT(P(i)),\ OPT(i-1)\right) \end{cases}$$

$OPT(i)$ = value of optimal selection using activities $\leq i$



$$= \begin{cases} 0 & \text{if } i = 0 \\ \max\left(\underline{w_i + OPT(P(i))},\ \underline{OPT(i-1)}\right) & \text{otherwise} \end{cases}$$

```
COMPUTE-OPT(n, v, P)
    OPT ← (n+1)-elt array
    OPT[0] ← 0
    for i = 1 to n
        OPT[i] =
        USE[i] = OPT[i] ≠ OPT[i-1]
```

$\Theta(n)$

SELECT(OPT, j)