Chapter 25: All-Pairs Shortest-Paths

Some Algorithms

- ✓ When no negative edges
  - Using Dijkstra's algorithm with array: $O(V^3)$
  - Using Binary heap implementation: $O(VE \lg V)$
  - Using Fibonacci heap: $O(VE + V^2 \log V)$
- When no negative cycles
  - Floyd-Warshall [1962]: $O(V^3)$ time
- When negative cycles
  - Using Bellman-Ford algorithm: $O(V^2 E) = O(V^4)$
  - Johnson [1977]: $O(VE + V^2 \log V)$ time based on a clever combination of Bellman-Ford and Dijkstra

A dynamic programming approach

- Optimal Substructure (allows recursion)
  - ✓ solution to the problem contains optimal solutions to subproblems.
- Overlapping Subproblems (allows speed up)
  - ✓ a recursive algorithm revisits the same subproblem over and over again.
- Compute the value of an optimal solution in a bottom-up fashion.

The structure of an optimal solution

- Consider a shortest path p from vertex i to vertex j, and suppose that p contains at most m edges. Assume that there are no negative-weight cycles. Hence, $m \le n-1$ is finite.

The structure of an optimal solution (cont.)

- If i = j, then p has weight 0 and no edge
- If i≠j, we decompose p into $i \overset{p'}{\sim} k \to j$ where p′ contains at most m-1 edges.
- Moreover, p′ is a shortest path from i to k and $\delta(i,j) = \delta(i,k) + w_{kj}$, where $\delta(i,j)$ denote the shortest weight path from i to j

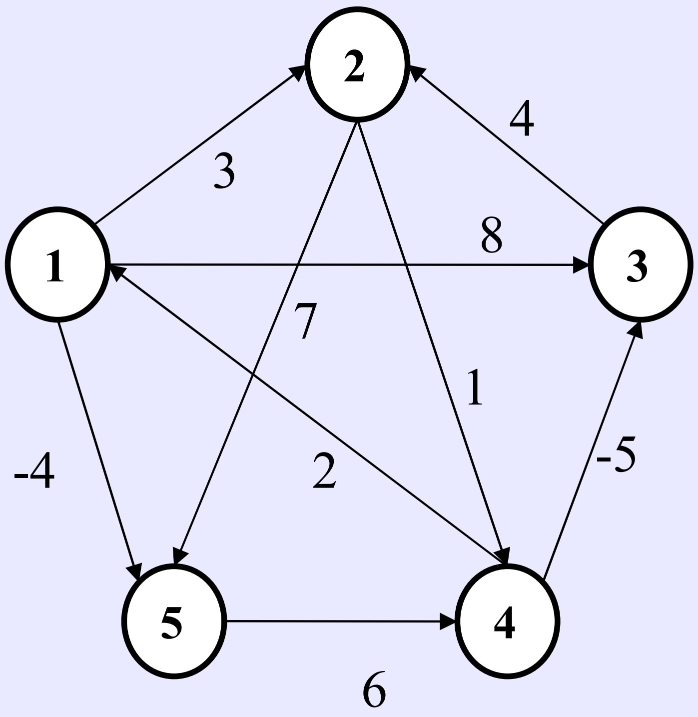
Recursive solution to the all-pairs shortest-path problem

- Define: $l_{ij}^{(m)}$ = minimum weight of any path from i to j that contains at most m edges.
- $$l_{ij}^{(0)} = \begin{cases} 0 & \text{if } i = j \\ \infty & \text{if } i \neq j \end{cases}$$
- Then $l_{ij}^{(m)} = \min\{ l_{ij}^{(m-1)}, \min_{1 \leq k \leq n} \{l_{ik}^{(m-1)} + w_{kj}\}\} = \min_{1 \leq k \leq n} \{l_{ik}^{(m-1)} + w_{kj}\}$ (why?)

Recursive solution to the all-pairs shortest-path problem

- Since shortest path from i to j contains at most n-1 edges,
  $\delta(i, j) = l_{ij}^{(n-1)}$
- Computing the shortest-path weight bottom up:
  - Compute $L^{(1)}, L^{(2)}, \ldots, L^{(n-1)}$, where $L^{(m)}=(l_{ij}^{(m)})$ for all i and j
  - Note that $L^{(1)} = W$.

Example:



$$W = L^{(1)} = \begin{pmatrix} 0 & 3 & 8 & \infty & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & \infty & \infty \\ 2 & \infty & -5 & 0 & \infty \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix}$$

$$L^{(2)} = \begin{pmatrix} 0 & 3 & 8 & 2 & -4 \\ 3 & 0 & -4 & 1 & 7 \\ \infty & 4 & 0 & 5 & 11 \\ 2 & -1 & -5 & 0 & -2 \\ 8 & \infty & 1 & 6 & 0 \end{pmatrix}$$

$$L^{(2)} = L^{(1)} \times W$$

$$L^{(1)} = \begin{pmatrix} 0 & 3 & 8 & \infty & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & \infty & \infty \\ 2 & \infty & -5 & 0 & \infty \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix} \times W = \begin{pmatrix} 0 & 3 & 8 & \infty & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & \infty & \infty \\ 2 & \infty & -5 & 0 & \infty \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix} = L^{(2)} = \begin{pmatrix} 0 & 3 & 8 & 2 & -4 \\ 3 & 0 & -4 & 1 & 7 \\ \infty & 4 & 0 & 5 & 11 \\ 2 & -1 & -5 & 0 & -2 \\ 8 & \infty & 1 & 6 & 0 \end{pmatrix}$$

$$l_{ij}^{(2)} = \min_{1 \le k \le 5} \{l_{ik}^{(1)} + w_{kj}\}$$

$$= \min \{l_{i1}^{(1)} + w_{1j},\ l_{i2}^{(1)} + w_{2j},\ l_{i3}^{(1)} + w_{3j},\ l_{i4}^{(1)} + w_{4j},\ l_{i5}^{(1)} + w_{5j}\}$$

$$l_{14}^{(2)} = \min \{l_{11}^{(1)} + w_{14},\ l_{12}^{(1)} + w_{24},\ l_{13}^{(1)} + w_{34},\ l_{14}^{(1)} + w_{44},\ l_{15}^{(1)} + w_{54}\} = \min\{\infty, 4, \infty, \infty, 2\} = 2$$

Matrix Multiplication

Let $l^{(m-1)}$ -> a

w -> b

$l^{(m)}$ -> c

min -> Σ

$+$ -> $\cdot$

$c_{ij} = \Sigma_{k=1 \text{ to } n} \; a_{ik} \cdot b_{kj}$

$l_{ij}^{(m)} = \min_{1 \le k \le n} \{l_{ik}^{(m-1)} + w_{kj}\}$

time complexity : $O(n^3)$

$$L^{(3)} = \begin{pmatrix} 0 & 3 & -3 & 2 & -4 \\ 3 & 0 & -4 & 1 & -1 \\ 7 & 4 & 0 & 5 & 11 \\ 2 & -1 & -5 & 0 & -2 \\ 8 & 5 & 1 & 6 & 0 \end{pmatrix}$$

$$L^{(4)} = \begin{pmatrix} 0 & 1 & -3 & 2 & -4 \\ 3 & 0 & -4 & 1 & -1 \\ 7 & 4 & 0 & 5 & 3 \\ 2 & -1 & -5 & 0 & -2 \\ 8 & 5 & 1 & 6 & 0 \end{pmatrix}$$

$L^{(3)} = L^{(2)} \times W$

$L^{(4)} = L^{(3)} \times W$

EXTENDED-SHORTEST-PATHS(L, W)

- Given matrices $L^{(m-1)}$ and W return $L^{(m)}$

$n \leftarrow L.row$

Let $L' = (l'_{ij})$ be a new $n \times n$ matrix

for $i = 1$ to n

    for $j = 1$ to n

        $l'_{ij} = \infty$

        for $k = 1$ to n

            $l'_{ij} = \min(l'_{ij}, l_{ik} + w_{kj})$

return L'

SLOW-ALL-PAIRS-SHORTEST-PATHS(W)

$$L^{(1)} = L^{(0)} \cdot W = W$$

$$L^{(2)} = L^{(1)} \cdot W = W^2$$

$$L^{(3)} = L^{(2)} \cdot W = W^3$$

$$\vdots$$

$$L^{(n-1)} = L^{(n-2)} \cdot W = W^{n-1}$$

SLOW-ALL-PAIRS-SHORTEST-PATHS(W)

```
n = W.rows
L^(1) = W
for m = 2 to n-1
        let L^(m) be a new n x n matrix
        L^(m) = EXTENDED-SHORTEST-PATHS( L^(m-1), W )
return L^(n-1)
```

Time complexity : $O(n^4)$

Improving the running time

$$L^{(1)} = W$$
$$L^{(2)} = W^2 = W \cdot W$$
$$L^{(4)} = W^4 = W^2 \cdot W \cdot W = W^2 \cdot W^2$$
$$\vdots$$

$$L^{(n)} = W^n = W^{n/2} \cdot W^{n/2}$$

i.e., using repeating squaring!

Time complexity: $O(n^3 \lg n)$

FASTER-ALL-PAIRS-SHORTEST-PATHS

FASTER-ALL-PAIRS-SHORTEST-PATHS(W)

1. n = W.row
2. $L^{(1)} = W$
3. m = 1
4. while m < n-1
5. let $L^{(2m)}$ be a new n x n matrix
6. $L^{(2m)}$ = Extend-Shorest-Path($L^{(m)}$, $L^{(m)}$)
7. m = 2m
8. return $L^{(m)}$

Some Algorithms

- ✓ When no negative edges
  - Using Dijkstra's algorithm with array: $O(V^3)$
  - Using Binary heap implementation: $O(VE \lg V)$
  - Using Fibonacci heap: $O(VE + V^2 \log V)$
- When no negative cycles
  - Floyd-Warshall [1962]: $O(V^3)$ time
- When negative cycles
  - Using Bellman-Ford algorithm: $O(V^2 E) = O(V^4)$
  - Johnson [1977]: $O(VE + V^2 \log V)$ time based on a clever combination of Bellman-Ford and Dijkstra

The Floyd-Warshall algorithm

- A different dynamic programming formulation
- **The structure of a shortest path:**

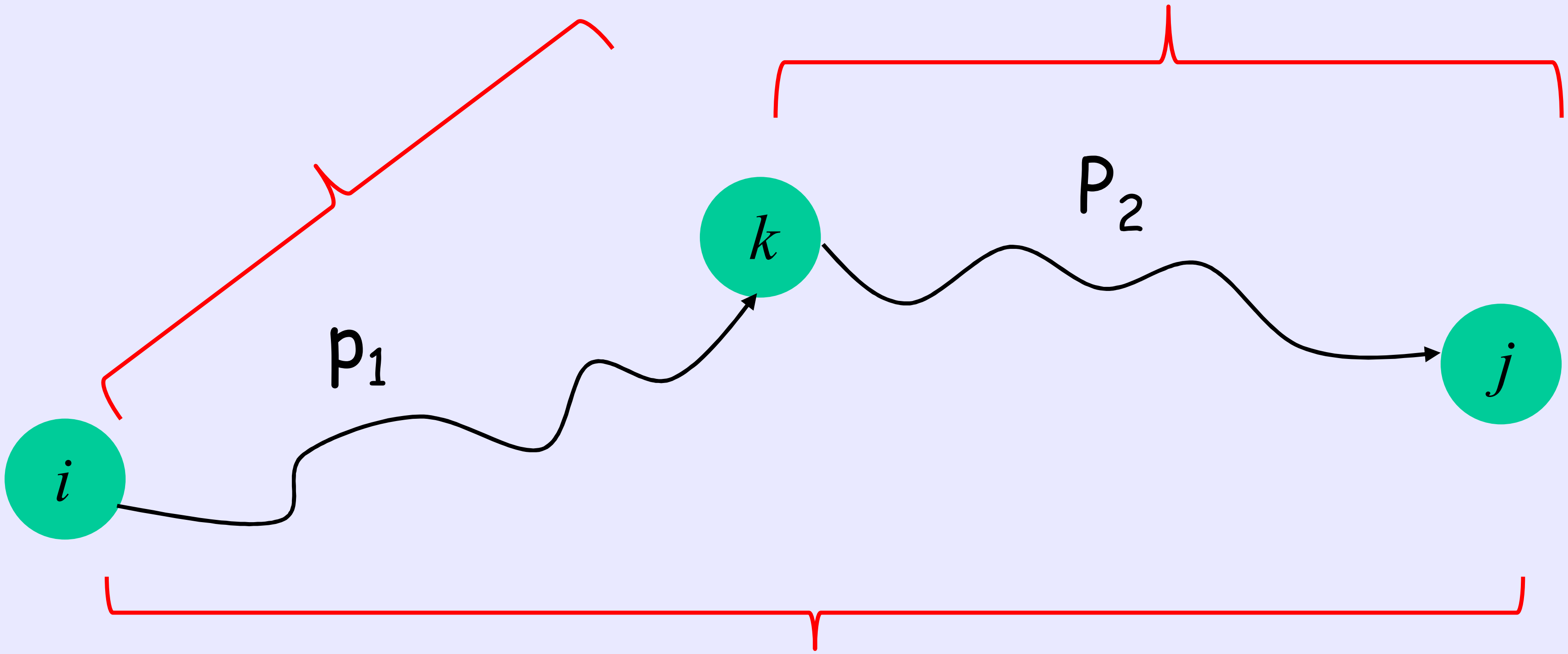
Let V(G)={1, 2, …, n}. For any pair of vertices i, j ϵV, consider all paths from i to j whose intermediate vertices are drawn from {1, 2, …, k}, and let p be a minimum weight path among them.

The structure of a shortest path

- If k is not in p, then all intermediate vertices are in {1, 2,…, k-1}.
- If k is an intermediate vertex of p, then p can be decomposed into $i \overset{p_1}{\sim} k \overset{p_2}{\sim} j$ where p_1 is a shortest path from i to k with all the intermediate vertices in {1, 2, …, k-1} and p_2 is a shortest path from k to j with all the intermediate vertices in {1, 2, …, k-1}.

p_1: all intermediate vertices in {1, 2, 3…., k-1}

p_2: all intermediate vertices in {1, 2, 3…., k-1}



P: all intermediate vertices in {1, 2, 3…., k}

A recursive solution to the all-pairs shortest path

- Let $d_{ij}^{(k)}$ =the weight of a shortest path from vertex i to vertex j with all intermediate vertices in the set {1, 2, …, k}.

$$
\begin{aligned}
d_{ij}^{(k)} &= w_{ij} && \text{if } k = 0 \\
&= \min(d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)}) && \text{if } k \geq 1
\end{aligned}
$$

$D^{(n)} = (d_{ij}^{(n)})$ the final solution!

FLOYD-WARSHALL(W)

$n = W.rows$

$D^{(0)} = W$

for $k = 1$ to n

    Let $D^{(k)} = (d_{ij}^{(k)})$ be a new $n \times n$ matrix

    for $i = 1$ to n

        for $j = 1$ to n

            $d_{ij}^{(k)} = \min\left(d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)}\right)$

return $D^{(n)}$

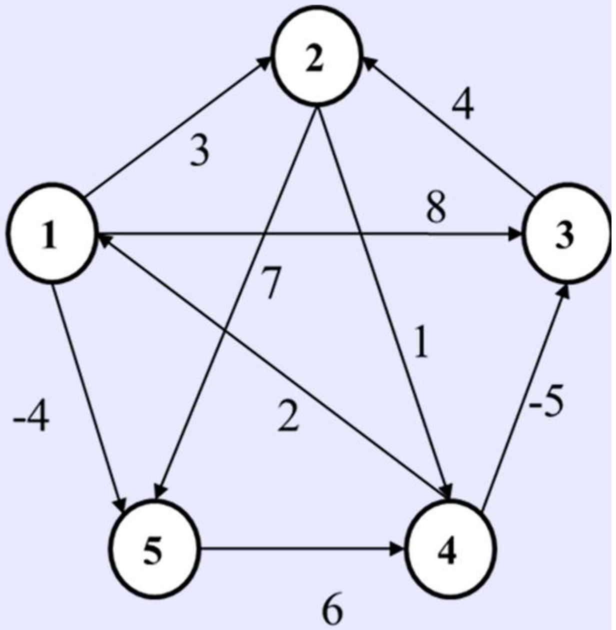
Complexity: $O(n^3)$

Example

$$D^{(0)} = \begin{pmatrix} 0 & 3 & 8 & \infty & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & \infty & \infty \\ 2 & \infty & -5 & 0 & \infty \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix} \qquad D^{(1)} = \begin{pmatrix} 0 & 3 & 8 & \infty & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & \infty & \infty \\ 2 & 5 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix}$$

$d_{ij}^{(1)} = \min(d_{ij}^{(0)}, d_{ik}^{(0)} + d_{kj}^{(0)}) \quad k = 1$

$d_{41}^{(1)} = \min(d_{41}^{(0)}, d_{41}^{(0)} + d_{11}^{(0)}) = 2$
$d_{42}^{(1)} = \min(d_{42}^{(0)}, d_{41}^{(0)} + d_{12}^{(0)}) = 5$
$d_{43}^{(1)} = \min(d_{43}^{(0)}, d_{41}^{(0)} + d_{13}^{(0)}) = -5$
$d_{44}^{(1)} = \min(d_{44}^{(0)}, d_{41}^{(0)} + d_{14}^{(0)}) = 0$
$d_{45}^{(1)} = \min(d_{45}^{(0)}, d_{41}^{(0)} + d_{15}^{(0)}) = -2$



$$D^{(1)} = \begin{pmatrix} 0 & 3 & 8 & \infty & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & \infty & \infty \\ 2 & 5 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix} \qquad D^{(2)} = \begin{pmatrix} 0 & 3 & 8 & 4 & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & 5 & 11 \\ 2 & 5 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix}$$

k = 1 k = 2

$d_{11}^{(2)} = \min (d_{11}^{(1)}, d_{12}^{(1)} + d_{21}^{(1)}) = 0$

$d_{12}^{(2)} = \min (d_{12}^{(1)}, d_{12}^{(1)} + d_{22}^{(1)}) = 3$

$d_{13}^{(2)} = \min (d_{13}^{(1)}, d_{12}^{(1)} + d_{23}^{(1)}) = 8$

$d_{14}^{(2)} = \min (d_{14}^{(1)}, d_{12}^{(1)} + d_{24}^{(1)}) = 4$

$d_{15}^{(2)} = \min (d_{15}^{(1)}, d_{12}^{(1)} + d_{25}^{(1)}) = -4$

$$D^{(2)} = \begin{pmatrix} 0 & 3 & 8 & 4 & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & 5 & 11 \\ 2 & 5 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix} \qquad D^{(3)} = \begin{pmatrix} 0 & 3 & 8 & 4 & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & 5 & 11 \\ 2 & -1 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix}$$

k = 2 k = 3

$d_{41}^{(3)} = \min (d_{41}^{(2)}, d_{43}^{(2)} + d_{31}^{(2)}) = 2$

$D_{42}^{(3)} = \min (d_{42}^{(2)}, d_{43}^{(2)} + d_{32}^{(2)}) = -1$

$d_{43}^{(3)} = \min (d_{43}^{(2)}, d_{43}^{(2)} + d_{33}^{(2)}) = -5$

$d_{44}^{(3)} = \min (d_{44}^{(2)}, d_{43}^{(2)} + d_{34}^{(2)}) = 0$

$d_{45}^{(3)} = \min (d_{45}^{(2)}, d_{43}^{(2)} + d_{35}^{(2)}) = -2$

Constructing a shortest path

- $\pi^{(0)}, \pi^{(1)}, \ldots, \pi^{(n)}$
- $\pi_{ij}^{(k)}$: is the predecessor of the vertex j on a shortest path from vertex i with all intermediate vertices in the set {1,2,…,k}.

$$\pi_{ij}^{(0)} = \text{NIL} \quad \text{if } i=j \text{ or } w_{ij} = \infty$$
$$= i \qquad \text{if } i \neq j \text{ and } w_{ij} < \infty$$

$$\pi_{ij}^{(k)} = \pi_{ij}^{(k-1)} \quad \text{if } d_{ij}^{(k-1)} \leq d_{ik}^{(k-1)} + d_{kj}^{(k-1)}$$
$$= \pi_{kj}^{(k-1)} \quad \text{if } d_{ij}^{(k-1)} > d_{ik}^{(k-1)} + d_{kj}^{(k-1)}$$

Example

$$D^{(0)} = \begin{pmatrix} 0 & 3 & 8 & \infty & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & \infty & \infty \\ 2 & \infty & -5 & 0 & \infty \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix} \qquad D^{(1)} = \begin{pmatrix} 0 & 3 & 8 & \infty & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & \infty & \infty \\ 2 & 5 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix}$$

$$\Pi^{(0)} = \begin{pmatrix} N & \boxed{1} & 1 & N & \boxed{1} \\ N & N & N & 2 & 2 \\ N & 3 & N & N & N \\ 4 & N & 4 & N & N \\ N & N & N & 5 & N \end{pmatrix} \qquad \Pi^{(1)} = \begin{pmatrix} N & 1 & 1 & N & 1 \\ N & N & N & 2 & 2 \\ N & 3 & N & N & N \\ 4 & \boxed{1} & 4 & N & \boxed{1} \\ N & N & N & 5 & N \end{pmatrix}$$

K =1

Example

$$D^{(1)} = \begin{pmatrix} 0 & 3 & 8 & \infty & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & \infty & \infty \\ 2 & 5 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix} \qquad D^{(2)} = \begin{pmatrix} 0 & 3 & 8 & 4 & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & 5 & 11 \\ 2 & 5 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix}$$

$$\Pi^{(1)} = \begin{pmatrix} N & 1 & 1 & N & 1 \\ N & N & N & \boxed{2} & \boxed{2} \\ N & 3 & N & N & N \\ 4 & 1 & 4 & N & 1 \\ N & N & N & 5 & N \end{pmatrix} \qquad \Pi^{(2)} = \begin{pmatrix} N & 1 & 1 & \boxed{2} & 1 \\ N & N & N & 2 & 2 \\ N & 3 & N & \boxed{2} & \boxed{2} \\ 4 & 1 & 4 & N & 1 \\ N & N & N & 5 & N \end{pmatrix}$$

K =2

$$D^{(2)} = \begin{pmatrix} 0 & 3 & 8 & 4 & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & 5 & 11 \\ 2 & 5 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix} \qquad D^{(3)} = \begin{pmatrix} 0 & 3 & 8 & 4 & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & 5 & 11 \\ 2 & -1 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix}$$

$$\Pi^{(2)} = \begin{pmatrix} N & 1 & 1 & 2 & 1 \\ N & N & N & 2 & 2 \\ N & \boxed{3} & N & 2 & 2 \\ 4 & 1 & 4 & N & 1 \\ N & N & N & 5 & N \end{pmatrix} \quad \text{K = 2}$$

$$\Pi^{(3)} = \begin{pmatrix} N & 1 & 1 & 2 & 1 \\ N & N & N & 2 & 2 \\ N & 3 & N & 2 & 2 \\ 4 & \boxed{3} & 4 & N & 1 \\ N & N & N & 5 & N \end{pmatrix} \quad \text{K = 3}$$

$$D^{(4)} = \begin{pmatrix} 0 & 3 & -1 & 4 & -4 \\ 3 & 0 & -4 & 1 & -1 \\ 7 & 4 & 0 & 5 & 3 \\ 2 & -1 & -5 & 0 & -2 \\ 8 & 5 & 1 & 6 & 0 \end{pmatrix} \qquad D^{(5)} = \begin{pmatrix} 0 & 1 & -3 & 2 & -4 \\ 3 & 0 & -4 & 1 & -1 \\ 7 & 4 & 0 & 5 & 3 \\ 2 & -1 & -5 & 0 & -2 \\ 8 & 5 & 1 & 6 & 0 \end{pmatrix}$$

$$\Pi^{(4)} = \begin{pmatrix} N & 1 & 4 & 2 & 1 \\ 4 & N & 4 & 2 & 1 \\ 4 & 3 & N & 2 & 1 \\ 4 & 3 & 4 & N & 1 \\ 4 & \boxed{3} & 4 & \boxed{5} & N \end{pmatrix} \quad \text{K = 4} \quad \Pi^{(5)} = \begin{pmatrix} N & \boxed{3} & 4 & \boxed{5} & 1 \\ 4 & N & 4 & 2 & 1 \\ 4 & 3 & N & 2 & 1 \\ 4 & 3 & 4 & N & 1 \\ 4 & 3 & 4 & 5 & N \end{pmatrix} \quad \text{K = 5}$$

For example:

The shortest path from 1 -> 2 is: 1 -> 5 -> 4 -> 3 -> 2

The shortest path from 2 -> 1 is: 2 -> 4 -> 1

The shortest path from 5 -> 2 is: 5 -> 4 -> 3 -> 2

Transitive-Closure of a Directed Graph

- Given a directed graph $G=(V, E)$ with $V=\{1, 2, \ldots n\}$
- We might wish to determine whether G contains a path from i to j for all vertex pairs $i, j \in V$.
- We define the transitive closure of G as the graph $G^* = (V, E^*)$, where $E^* = \{(i, j)$: there is a path from vertex i to vertex j in G$\}$

Transitive closure of a directed graph

- Given a directed graph $G = (V, E)$ with $V = \{1,2,\ldots, n\}$
- The transitive closure of G is $G^* = (V, E^*)$ where $E^*=\{(i, j) \mid \text{there is a path from } i \text{ to } j \text{ in } G\}$.

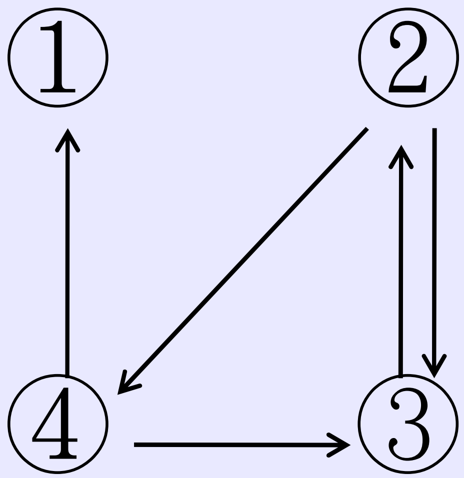
Modify FLOYD-WARSHALL algorithm:

$$t_{ij}^{(0)} = \begin{cases} 0 & \text{if } i \neq j \text{ and } (i,j) \notin E \\ 1 & \text{if } i=j \text{ or } (i,j) \in E \end{cases}$$

for $k \geq 1$

$$t_{ij}^{(k)} = t_{ij}^{(k-1)} \vee (t_{ik}^{(k-1)} \wedge t_{kj}^{(k-1)})$$

Example



$$T^{(0)} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{pmatrix}$$

$$T^{(1)} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{pmatrix}$$ K=1

$$T^{(2)} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{pmatrix}$$ K= 2

$$T^{(3)} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$ K = 3

$$T^{(4)} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$ K = 4

$$t_{ij}^{(k)} = t_{ij}^{(k-1)} \vee (t_{ik}^{(k-1)} \wedge t_{kj}^{(k-1)})$$

TRANSITIVE-CLOSURE(G)

n = |G.V|
Let $T^{(0)} = (t_{ij}^{(0)})$ be a new n x n matrix
for i = 1 to n
    for j =1 to n
        if i == j or $(i, j) \in G.E$
            $t_{ij}^{(0)} = 1$
        else $t_{ij}^{(0)} = 0$
for k =1 to n
    Let $T^{(k)} = (t_{ij}^{(k)})$ be a new n x n matrix
    for i = 1 to n
        for j =1 to n
            $t_{ij}^{(k)} = t_{ij}^{(k-1)} \vee (t_{ik}^{(k-1)} \wedge t_{kj}^{(k-1)})$
return $T^{(n)}$

Time complexity: $O(n^3)$

Some Algorithms

- ✓ When no negative edges
  - Using Dijkstra's algorithm: $O(V^3)$
  - Using Binary heap implementation: $O(VE \lg V)$
  - Using Fibonacci heap: $O(VE + V^2 \log V)$
- ✓ When no negative cycles
  - Floyd-Warshall [1962]: $O(V^3)$ time
- When negative cycles
  - Using Bellman-Ford algorithm: $O(V^2 E) = O(V^4)$
  - Johnson [1977]: $O(VE + V^2 \log V)$ time based on a clever combination of Bellman-Ford and Dijkstra

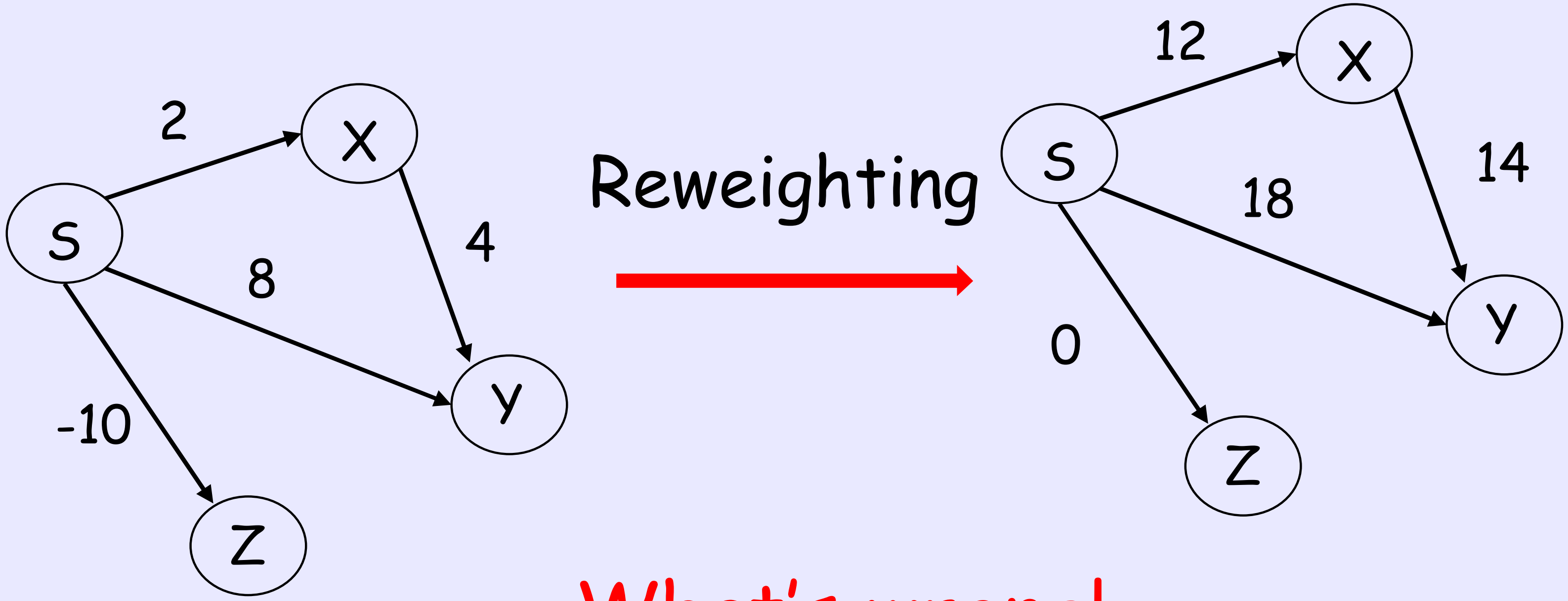
Johnson's algorithm for sparse graphs

- For sparse graphs, it is asympototically faster than either repeated squaring of matrices or the Floyd-Warshall algorithm
- Using reweighting technique

Reweighting technique

- If G has negative-weighted edge, we compute a new set of nonnegative weight that allows us to use the same method. The new set of edge weight $\hat{w}$ satisfies:
- 1. For all pairs of vertices $u, v \in V$, a shortest path from u to v using weight function w is also a shortest path from u to v using the weight function $\hat{w}$
- 2. $\forall (u,v) \in E(G)$, $\hat{w}(u,v)$ is nonnegative

Reweighting Example

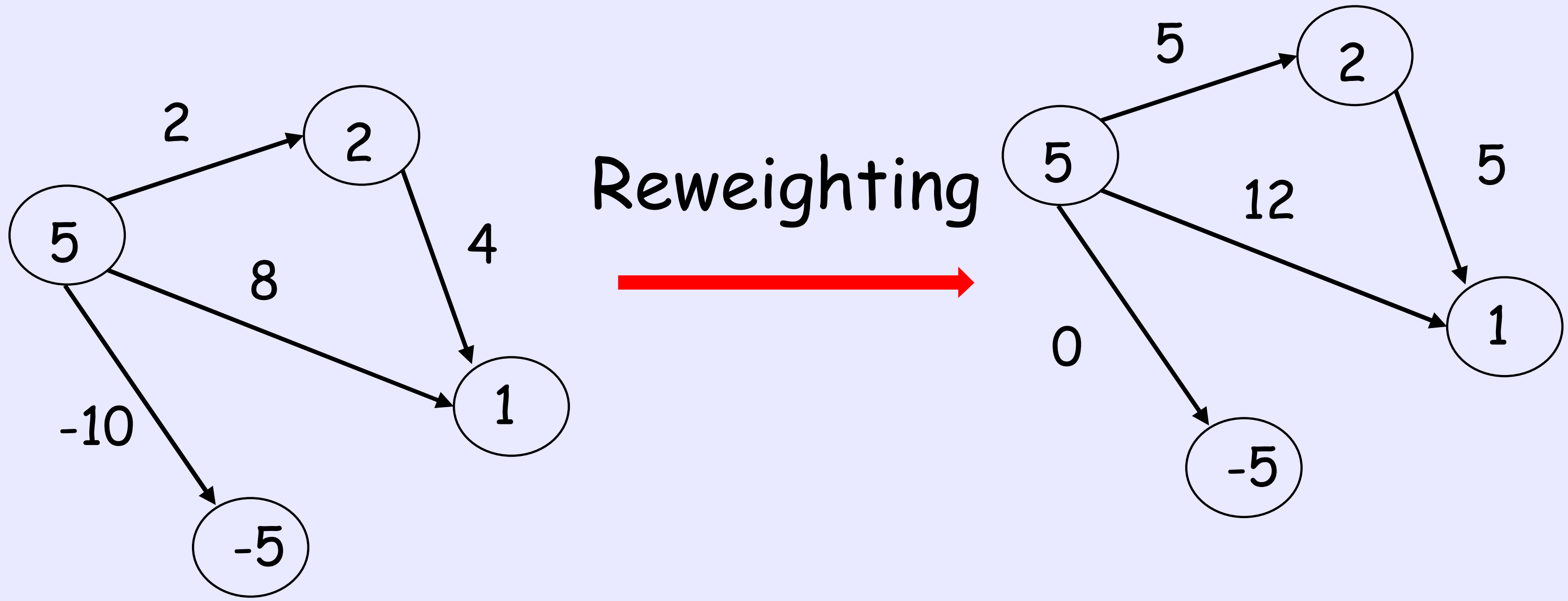


What's wrong!

Lemma: (Reweighting doesn't change shortest paths)

- Given a weighted directed graph $G = (V, E)$ with weight function $w:E \rightarrow R$, let $h:V \rightarrow R$ be any function mapping vertices to real numbers. For each edge $(u,v) \in E$, $\hat{w}(u,v) = w(u,v) + h(u) - h(v)$

e.g.



Lemma: (Reweighting doesn't change shortest paths)

- Let $p = \langle v_0, v_1, \ldots, v_k \rangle$
- Then $w(p) = \delta(v_0,v_k)$ if and only if $\hat{w}(p) = \hat{\delta}(v_0,v_k)$ Also, G has a negative-weight cycle using weight function w iff G has a negative weight cycle using weight function $\hat{w}$.
- Question: how to setting the value of $h(v_i)$ for all i?

Proof

- First we show that $\hat{w}(p) = w(p) + h(v_0) - h(v_k)$

$$\hat{w}(p) = \sum_{i=1}^{k} \hat{w}(v_{i-1}, v_i)$$

$$= \sum_{i=1}^{k} (w(v_{i-1}, v_i) + h(v_{i-1}) - h(v_i))$$

$$= \sum_{i=1}^{k} w(v_{i-1}, v_i) + h(v_0) - h(v_k)$$

$$= w(p) + h(v_0) - h(v_k)$$

Proof

- Because $h(v_0)$ and $h(v_k)$ do not depend on the path, if one path from v_0 to v_k is shorter than another using weight function w, then it is also shorter using $\hat{w}$. Thus,

  $w(p) = \delta(v_0, v_k)$ if and only if $\hat{w}(p) = \bar{\delta}(v_0, v_k)$ $= w(p) + h(v_0) - h(v_k)$

Proof

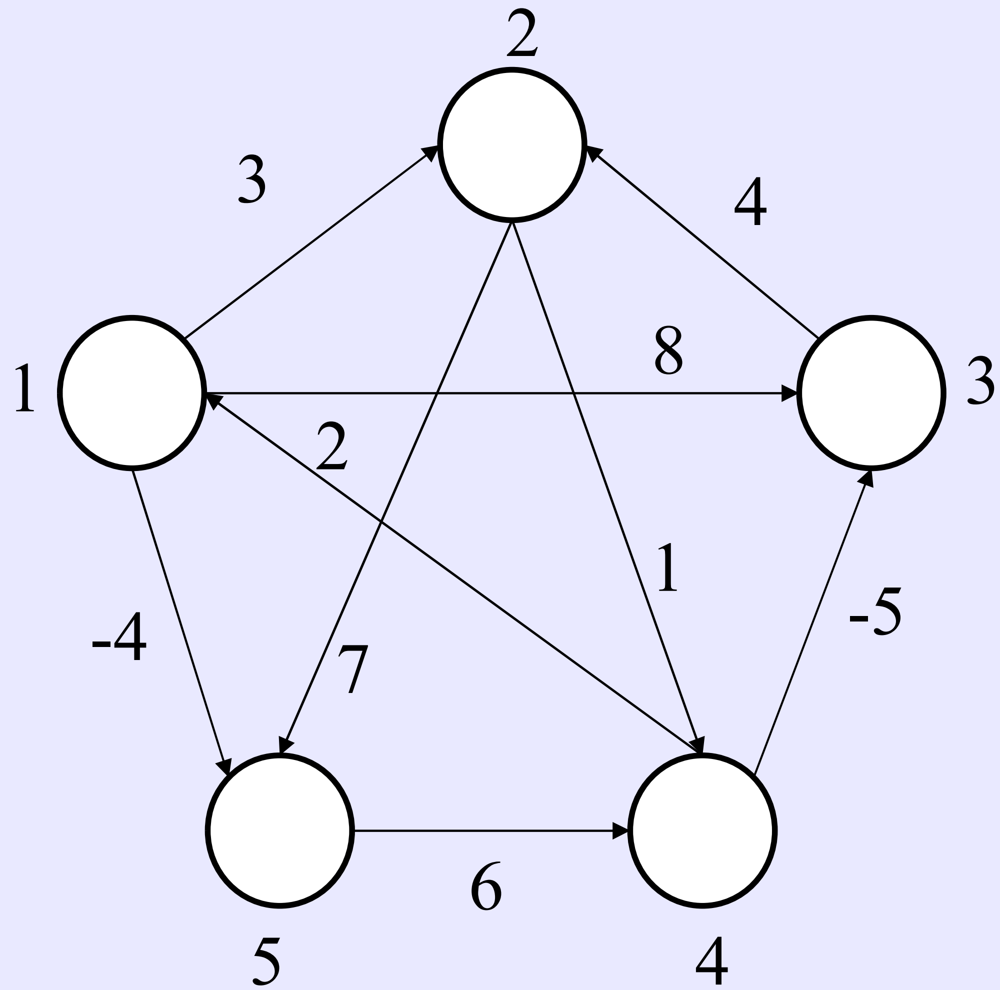
- G has a negative-weight cycle using w iff G has a negative-weight cycle using $\hat{w}$.
- Consider any cycle $C=\langle v_0, v_1, \ldots, v_k \rangle$ with $v_0=v_k$. Then $\hat{w}(C) = w(C) + h(v_0) - h(v_k) = w(C)$.

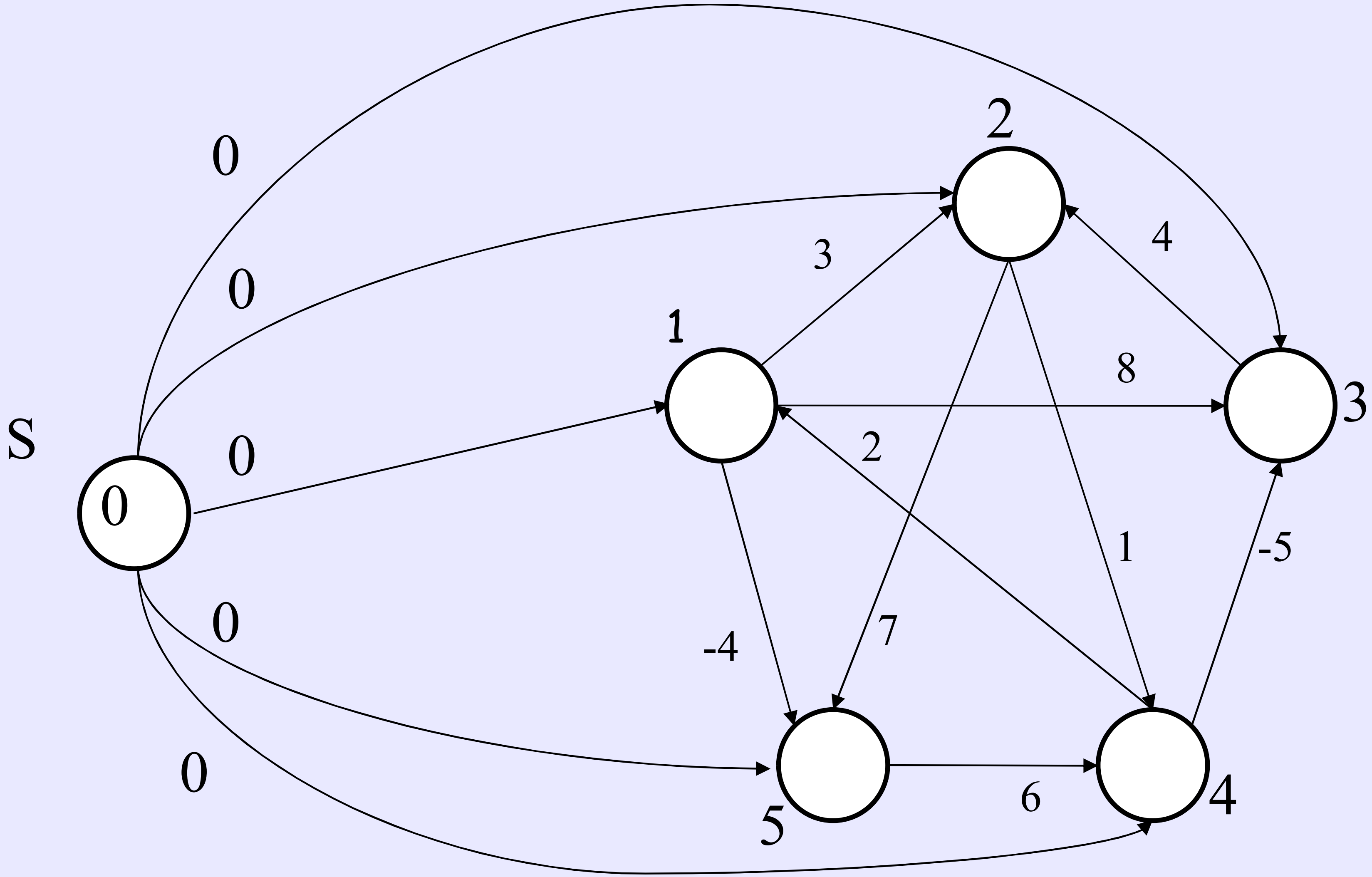
Question: how to setting the value of $h(v_i)$ for all i?

Producing nonnegative weight by reweighting

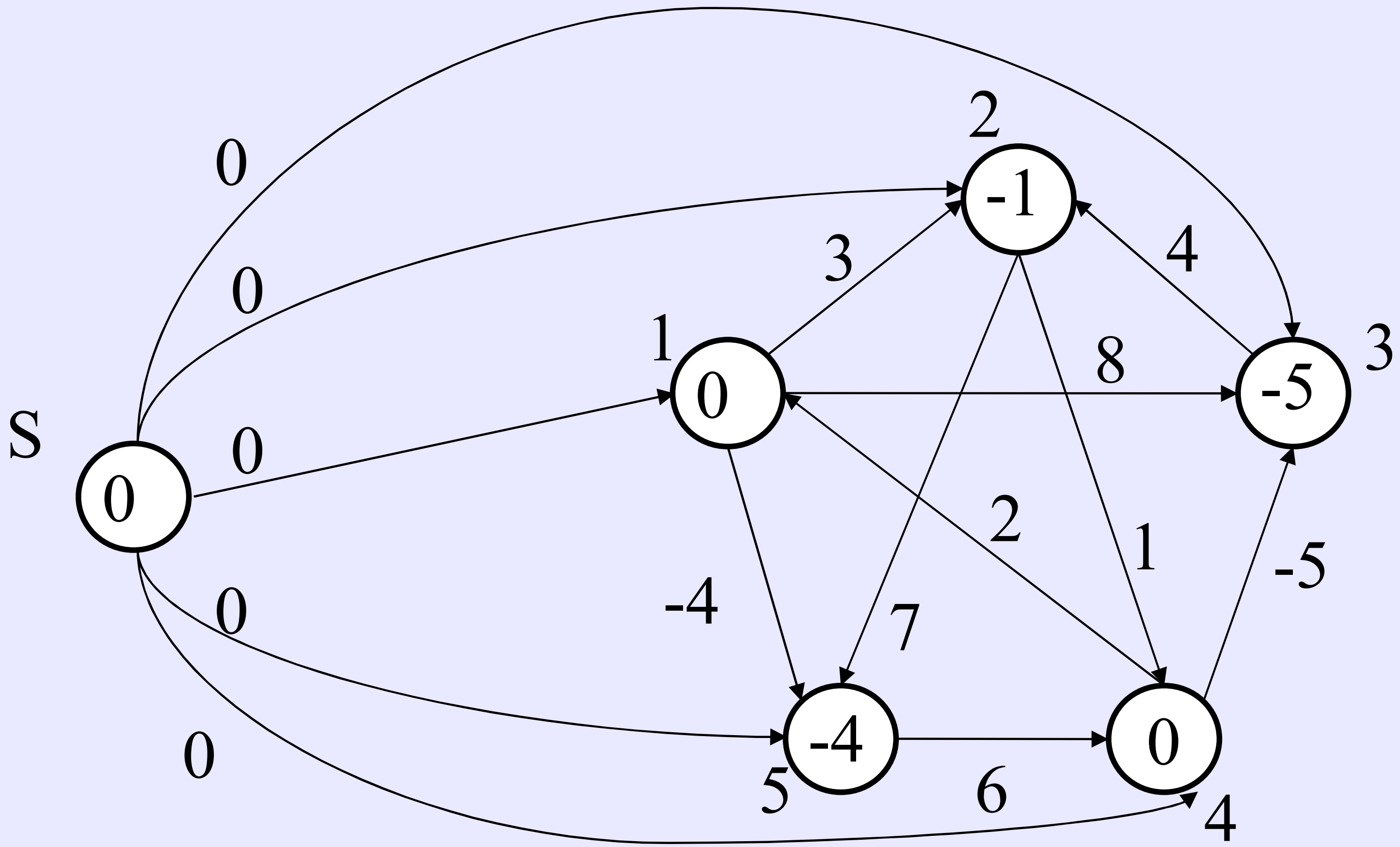
- Given a weighted directed graph $G = (V, E)$
- We make a new graph $G' = (V', E')$, $V' = V \cup \{s\}$, $E' = E \cup \{(s,v): v \in V\}$ and $w(s,v) = 0$, for all v in V
- Let $h(v) = \delta(s, v)$ for all $v \in V$
- We have $h(v) \leq h(u) + w(u, v)$ and there is no negative weight cycle (why?)
- $\hat{w}(u, v) = w(u, v) + h(u) - h(v) \geq 0$

Example:



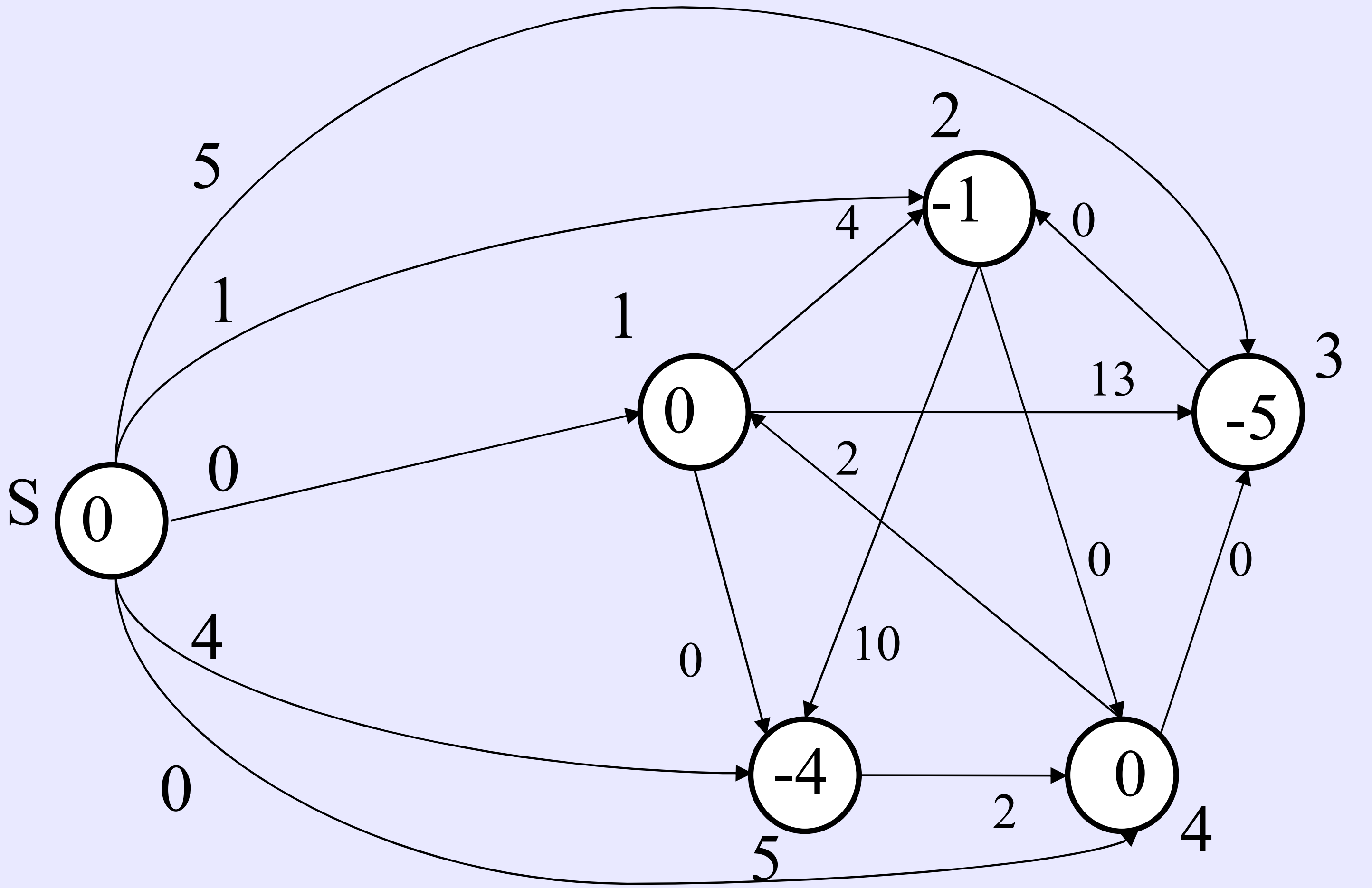


S
0
0
0
0
0
0
1
2
3
4
5
3
4
8
2
1
-5
-4
7
6


S
0
0
0
0
0
0
1
0
2
-1
3
-5
4
0
5
-4
3
4
8
2
1
-5
-4
7
6
0
0

$$h(v) = \delta(s, v)$$

$$\hat{w}(u, v) = w(u, v) + h(u) - h(v)$$



JOHNSON algorithm

1 Computing G', where $G'.V = G.V \cup \{\mathbf{s}\}$ and $G'.E = G.E \cup \{(s, v): v \in G.V\}$ and $w(s, v) = 0$.

2 if BELLMAN-FORD(G', w, s)= FALSE

3 print "the input graph contains negative weight cycle"

4 else for each vertex $v \in G'.V$

5 set $h(v)$ to be the value of $\delta(s, v)$ computed by the BF algorithm

6 for each edge $(u, v) \in G'.E$, $\hat{w}(u, v) = w(u, v) + h(u) - h(v)$

JOHNSON algorithm

Let $D = (d_{uv})$ be a new $n \times n$ matrix
for each vertex $u \in G.V$
    run DIJKSTRA $(G, \hat{w}, u)$ to compute $\hat{\delta}(u, v)$
        for all $v \in V[G]$.
for each vertex $v \in G.V$
        $d_{uv} = \hat{\delta}(u, v) + h(v) - h(u)$
return D

Complexity: Using Fibonacci heap $O(V^2 \lg V + VE)$

Using Binary heap implementation: $O(VE \lg V)$

Practice at home

- Exercises: 25.1-6, 25.1-9, 25.2-6, 25.2-8, 25.3-2, 25.3-3, 25.3-5, 25.3-6

Quiz

- For all-to-all shortest paths with no negative edges
  - The time complexity of using Dijkstra's algorithm with array is O(?)
  - The time complexity of using Binary heap implementation is O(?)
- When no negative cycles
  - The time complexity o Floyd-Warshall is O(?)