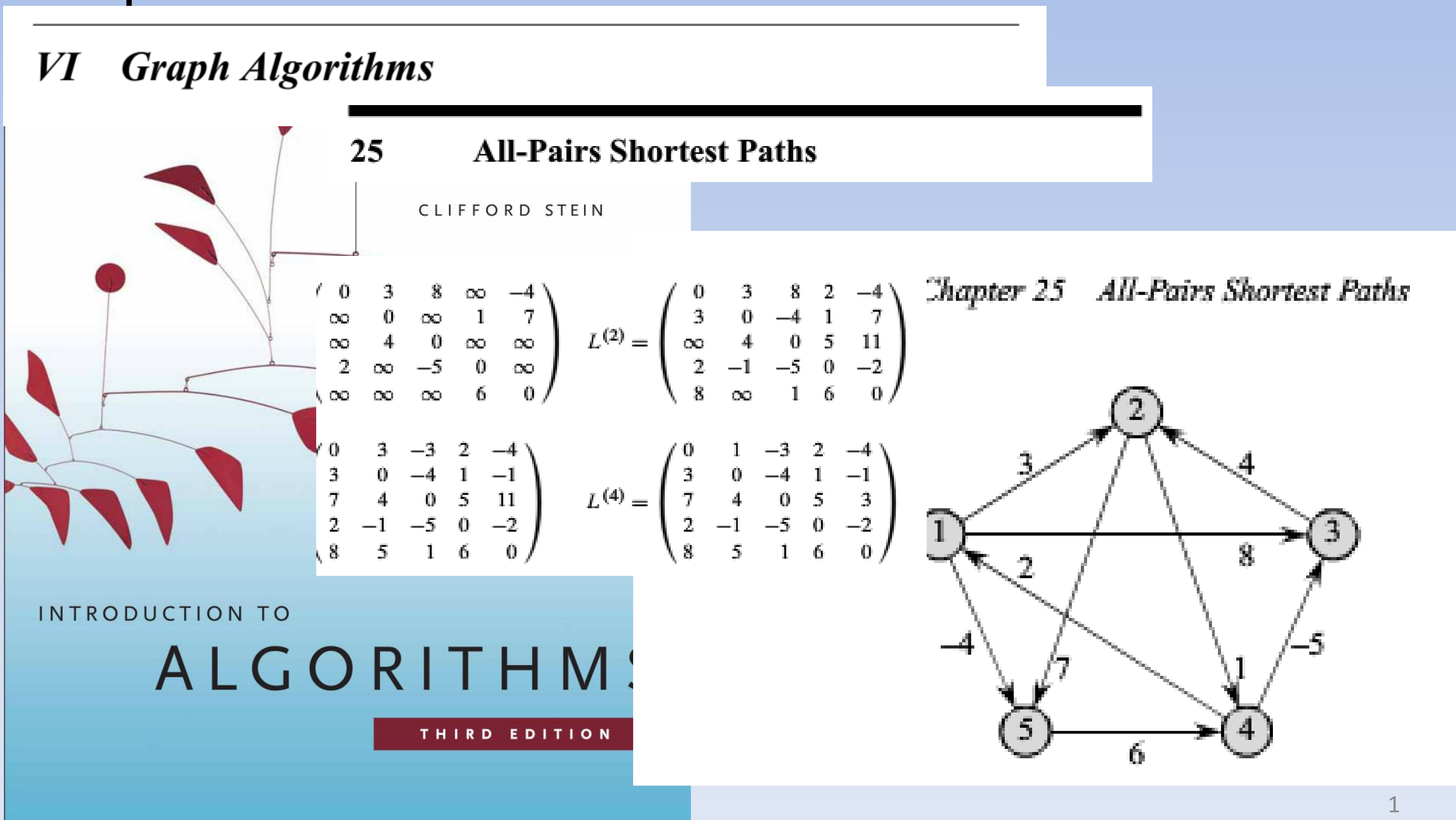
Design and Analysis of Algorithms

Section VI : Graph Algorithms

Chapter 25: All-Pairs Shortest Paths



All-Pairs Shortest Paths Problem

- Given:
  - Weighted, Directed Graph G=(V, E)
  - Weight Function w: E ->
    - Edges -> Real-Valued Weights

$\mathbb{R}$

- Weight of path P=<v_0, v_1, …, v_k>
  - $w(p) = \sum w(v_{i-1}, v_i)$
- Shortest-Path Weight δ(u,v) is the minimum weight path w(p) that goes from u to v, otherwise ∞
- The shortest path from u to v is any path p with a weight of δ(u,v)
- **ALL-PAIRS SHORTEST PATHS**
  - For all pairs of vertices u,v ∈ V, δ(u,v)

All-Pairs Shortest Paths

- Adjacency-List Representation
- Assume vertices are numbered 1, 2, … ,|V|
- Input: n x n weight matrix W of an n-vertex directed graph G=(V,E)
- $W = (w_{ij})$, w_{ij}=
  - 0, if i=j
  - the weight of directed edge (i, j), if $i \neq j$ & $(i,j) \in E$
  - ∞ if $i \neq j$ & $(i,j) \notin E$

All-Pairs Shortest Paths Output

- *n* x *n* matrix D = (d_{ij})
  - ➢ d_{ij}= weight of shortest path from vertex i to vertex j.
  - ➢ $\delta(i,j)$
- Predecessor Matrix $\Pi = (\pi_{ij})$
- π_{ij} =
  - ➢ NIL, if i=j or no path from i to j.
  - ➢ predecessor to j on some shortest path from i to j.

Predecessor Subgraph

- $G_{\pi,i}=(V_{\pi,i}, E_{\pi,i})$
- $V_{\pi,i} = \{j \in V : \pi_{ij} \neq nil\} \cup \{i\}$
- $E_{\pi,i} = \{(\pi_{ij,} j) : j \in V_{\pi,l} - \{i\}$

Print-All-Pairs-Shortest-Path

PRINT-ALL-PAIRS-SHORTEST-PATH(Π, i, j)

```
if i == j
    print i
elseif π_ij == NIL
    print "no path from" i "to" j "exists"
else PRINT-ALL-PAIRS-SHORTEST-PATH(Π, i, π_ij)
    print j
```

Dynamic Programming Steps

- Characterize the structure of optimal solution
- Recursively define the value of an optimal solution
- Compute the value of an optimal solution bottom-up.
- Construct the optimal solution from computed information.

Recursive Solution

- Define $l_{ij}^{(m)}$ as the minimum weight of any path from vertex *i* to *j* that contains at most *m* edges.
- $l_{ij}^{(0)} =$
  - 0 if i=j
  - ∞ if i≠j

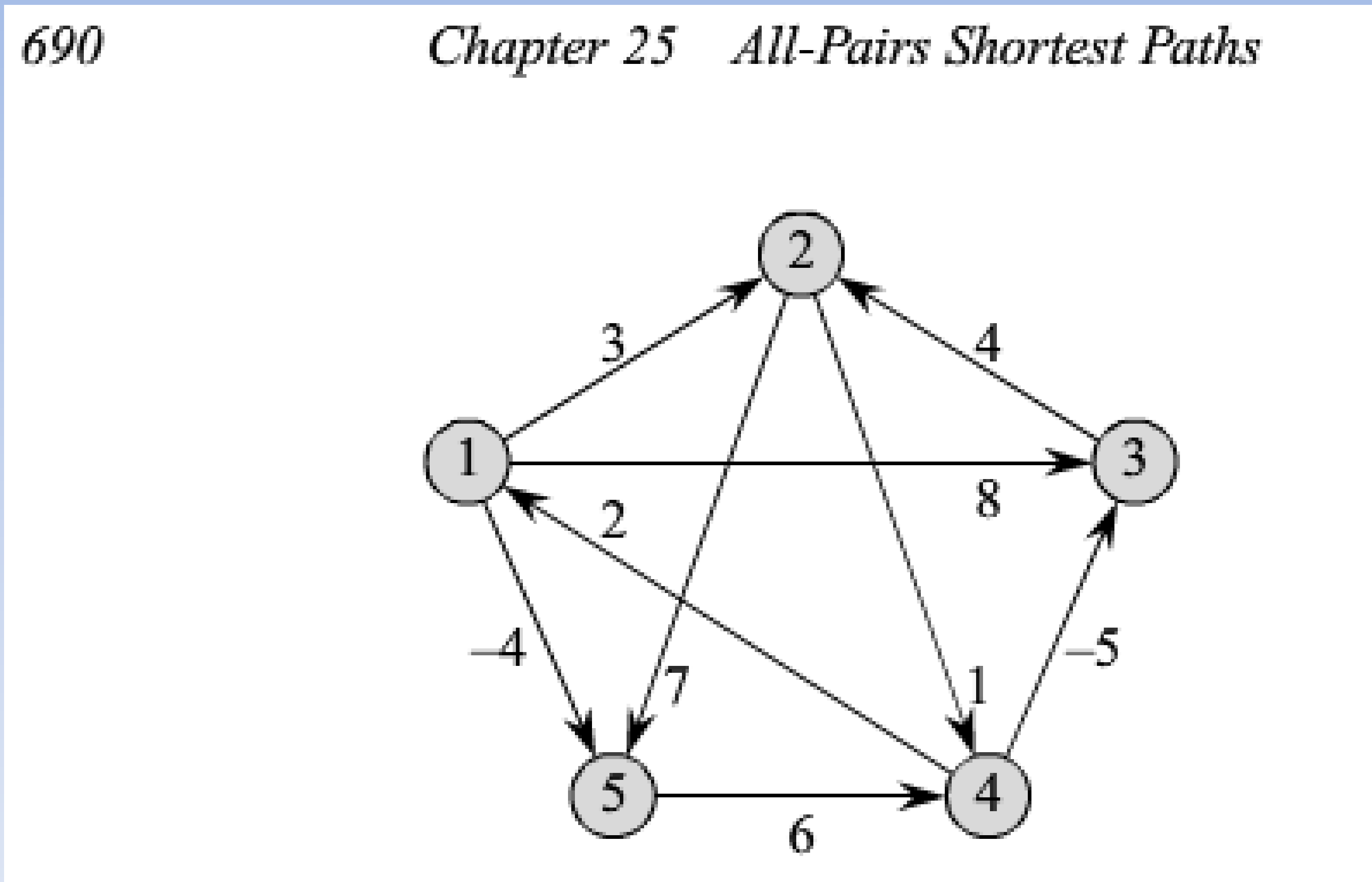
Single Source Relaxation Process

- Relax Edge (u, v) By:
  - Testing possible shortest path improvement to **v** by using current path to **u**
  - When improvements are possible update:
    - v.d: estimated shortest-path weight
    - v.π: v's parent

Recursive Solution

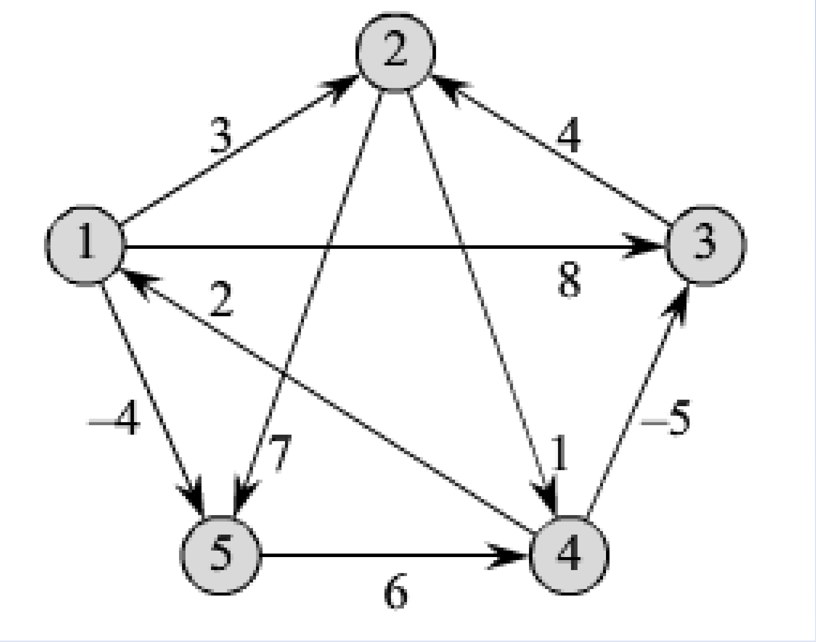
- Define $l_{ij}^{(m)}$ as the minimum weight of any path from vertex i to j that contains at most m edges.
- $l_{ij}^{(0)}$ =
  - 0 if i=j
  - ∞ if i≠j
- **Compute $l_{ij}^{(m)}$ as the minimum**
  - **$l_{ij}^{(m-1)}$ the minimum weight path from i to j with m-1 edges**
  - **$l_{ik}^{(m)} + w_{kj}$: for all possible k's.**

Example Graph



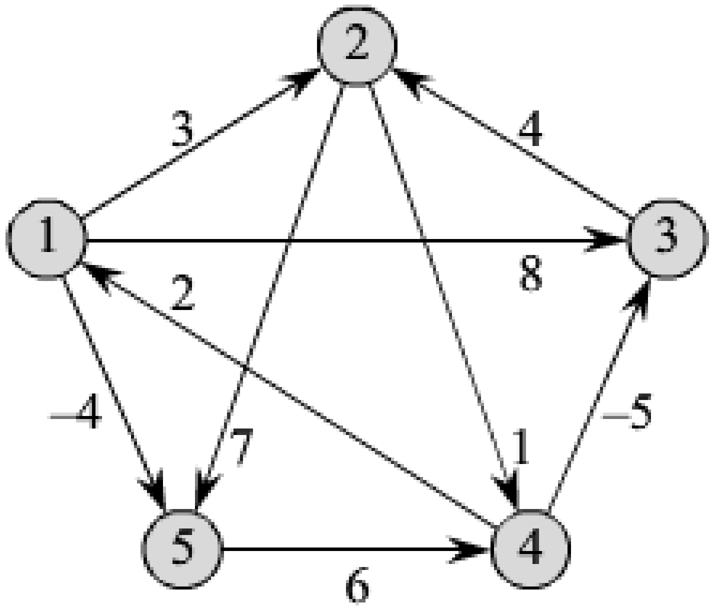
$L^{(1)}$

$$\begin{pmatrix} 0 & 3 & 8 & \infty & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & \infty & \infty \\ 2 & \infty & -5 & 0 & \infty \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix}$$



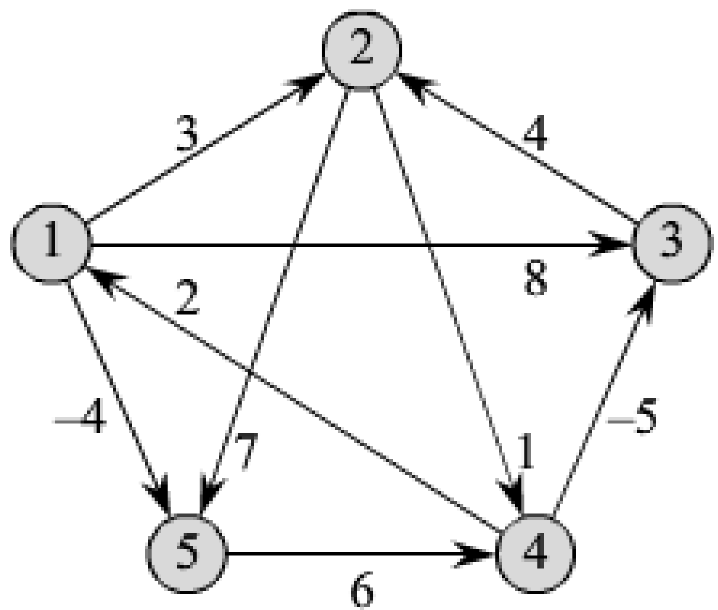
$L^{(1)}$

$$\left(\begin{array}{cc} 0 & 3 \\ \infty & 0 \\ \infty & 4 \\ 2 & \infty \\ \infty & \infty \end{array} \right.$$



- Shortest Path of length 1 weights =
  - (1, 2) w/ w(1,2) = 3

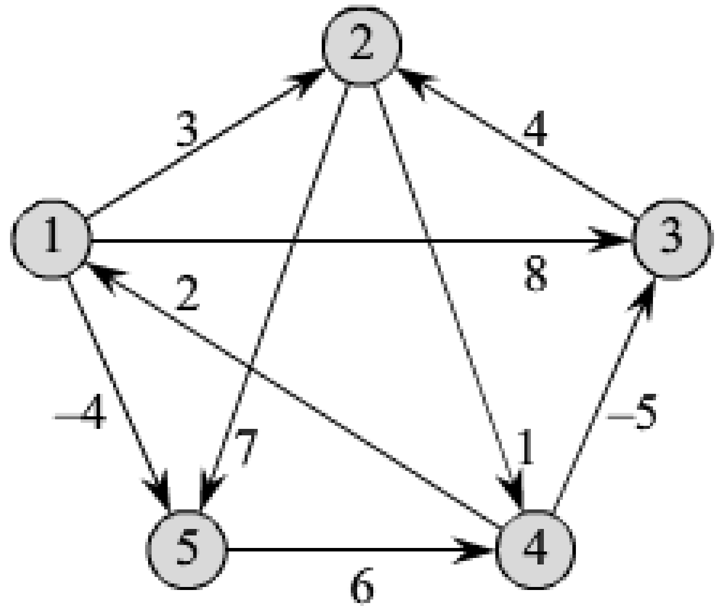
$$L^{(1)} \left(\begin{matrix} 0 & 3 \\ \infty & 0 \\ \infty & 4 \\ 2 & \infty \\ \infty & \infty \end{matrix} \right.$$



- Shortest Path of length 1 weights =
  - (1, 2) w/ w(1,2) = 3
  - (4, 1) w/ w(4,1) = 2

$L^{(1)}$

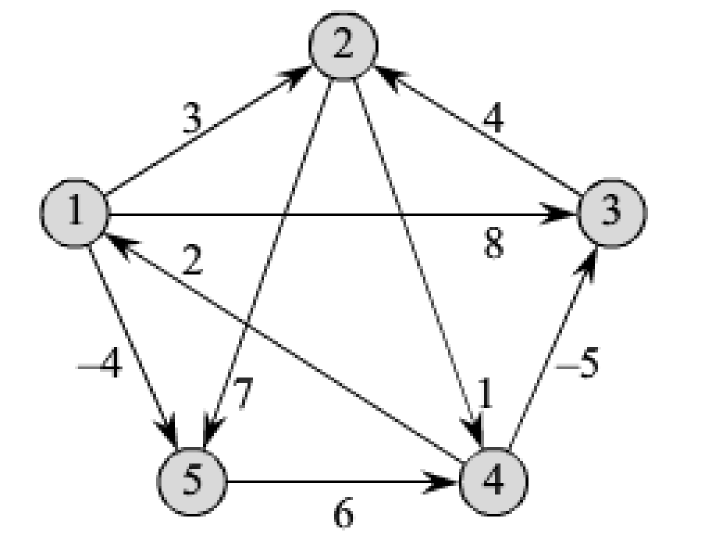
$$\left(\begin{matrix} 0 & 3 \\ \infty & 0 \\ \infty & 4 \\ 2 & \infty \\ \infty & \infty \end{matrix}\right.$$



- Shortest Path of length 1 weights =
  - (1, 2) w/ w(1,2) = 3
  - (4, 1) w/ w(4,1) = 2
  - (3, 2) w/ w(3,2) = 4

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

- Shortest Path of length 1 weights =
  - (1, 2) w/ w(1,2) = 3
  - (4, 1) w/ w(4,1) = 2
  - (3, 2) w/ w(3,2) = 4



$L^{(2)}$

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

$L^{(3)}$

$$\begin{pmatrix} 0 & 3 & -3 & 2 & -4 \\ 3 & 0 & -4 & 1 & -1 \\ 7 & 4 & 0 & 5 & 11 \\ 2 & -1 & -5 & 0 & -2 \\ 8 & 5 & 1 & 6 & 0 \end{pmatrix}$$

$L^{(4)}$

$$\begin{pmatrix} 0 & 1 & -3 & 2 & -4 \\ 3 & 0 & -4 & 1 & -1 \\ 7 & 4 & 0 & 5 & 3 \\ 2 & -1 & -5 & 0 & -2 \\ 8 & 5 & 1 & 6 & 0 \end{pmatrix}$$

$L^{(4)}$

- $L^{(m)} = L^{(4)}$, for all $m \geq 4$

$$\begin{pmatrix} 0 & 1 & -3 & 2 & -4 \\ 3 & 0 & -4 & 1 & -1 \\ 7 & 4 & 0 & 5 & 3 \\ 2 & -1 & -5 & 0 & -2 \\ 8 & 5 & 1 & 6 & 0 \end{pmatrix}$$

Algorithm

EXTEND-SHORTEST-PATHS(L, W)

$n = L.rows$
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'

Algorithm

RELAX(u, v, w)

if $v.d > u.d + w(u, v)$
    $v.d = u.d + w(u, v)$
    $v.\pi = u$

EXTEND-SHORTEST-PATHS(L, W)

$n = L.rows$
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'

Algorithm

SQUARE-MATRIX-MULTIPLY(A, B)

$n = A.rows$
let C be a new $n \times n$ matrix
for $i = 1$ **to** n
    for $j = 1$ **to** n
        $c_{ij} = 0$
        for $k = 1$ **to** n
            $c_{ij} = c_{ij} + a_{ik} \cdot b_{kj}$
return C

EXTEND-SHORTEST-PA

$n = L.rows$
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'

Algorithm

EXTEND-SHORTEST-PATHS(L, W)

```
n = L.rows
let L' = (l'_ij) be a new n × n matrix
for i = 1 to n
    for j = 1 to n
        l'_ij = ∞
        for k = 1 to n
            l'_ij = min(l'_ij, l_ik + w_kj)
return L'
```

$$
\begin{aligned}
L^{(1)} &= L^{(0)} \cdot W &= W \,, \\
L^{(2)} &= L^{(1)} \cdot W &= W^{2} \,, \\
L^{(3)} &= L^{(2)} \cdot W &= W^{3} \,, \\
& \vdots & \\
L^{(n-1)} &= L^{(n-2)} \cdot W &= W^{n-1} \,.
\end{aligned}
$$

Algorithm – O(n^3)

EXTEND-SHORTEST-PATHS(L, W)

$n = L.rows$
let $L' = (l'_{ij})$ be a new $n \times n$ matrix
for $i = 1$ **to** n

$$
\begin{aligned}
L^{(1)} &= L^{(0)} \cdot W &&= W\,, \\
L^{(2)} &= L^{(1)} \cdot W &&= W^{2}\,, \\
L^{(3)} &= L^{(2)} \cdot W &&= W^{3}\,, \\
&\ \ \vdots \\
L^{(n-1)} &= L^{(n-2)} \cdot W &&= W^{n-1}\,.
\end{aligned}
$$

Algorithm

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 × n matrix
    L^(m) = EXTEND-SHORTEST-PATHS(L^(m-1), W)
return L^(n-1)
```

$$
\begin{aligned}
L^{(1)} &= L^{(0)} \cdot W = W \,, \\
L^{(2)} &= L^{(1)} \cdot W = W^2 \,, \\
L^{(3)} &= L^{(2)} \cdot W = W^3 \,, \\
&\vdots \\
L^{(n-1)} &= L^{(n-2)} \cdot W = W^{n-1} \,.
\end{aligned}
$$

Algorithm – O(n^4)

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 × n matrix
    L^(m) = EXTEND-SHORTEST-PATHS(L^(m-1), W)
return L^(n-1)
```

$$
\begin{aligned}
L^{(1)} &= L^{(0)} \cdot W &= W\,, \\
L^{(2)} &= L^{(1)} \cdot W &= W^{2}\,, \\
L^{(3)} &= L^{(2)} \cdot W &= W^{3}\,, \\
&\vdots \\
L^{(n-1)} &= L^{(n-2)} \cdot W &= W^{n-1}\,.
\end{aligned}
$$

Algorithm

$$\begin{aligned}
L^{(1)} &= L^{(0)} \cdot W &= W\,, \\
L^{(2)} &= L^{(1)} \cdot W &= W^{2}\,, \\
L^{(3)} &= L^{(2)} \cdot W &= W^{3}\,, \\
&\vdots \\
L^{(n-1)} &= L^{(n-2)} \cdot W &= W^{n-1}\,.
\end{aligned}$$

$$\begin{aligned}
L^{(1)} &= W\,, \\
L^{(2)} &= W^{2} &= W \cdot W\,, \\
L^{(4)} &= W^{4} &= W^{2} \cdot W^{2} \\
L^{(8)} &= W^{8} &= W^{4} \cdot W^{4}\,, \\
&\vdots \\
L^{(2^{\lceil \lg(n-1) \rceil})} &= W^{2^{\lceil \lg(n-1) \rceil}} &= W^{2^{\lceil \lg(n-1) \rceil - 1}} \cdot W^{2^{\lceil \lg(n-1) \rceil - 1}}\,.
\end{aligned}$$

Algorithm

$$
\begin{aligned}
L^{(1)} &= W, \\
L^{(2)} &= W^2 &&= W \cdot W, \\
L^{(4)} &= W^4 &&= W^2 \cdot W^2 \\
L^{(8)} &= W^8 &&= W^4 \cdot W^4, \\
&\vdots \\
L^{(2^{\lceil \lg(n-1) \rceil})} &= W^{2^{\lceil \lg(n-1) \rceil}} &&= W^{2^{\lceil \lg(n-1) \rceil - 1}} \cdot W^{2^{\lceil \lg(n-1) \rceil - 1}}
\end{aligned}
$$

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

$n = W.rows$
$L^{(1)} = W$
$m = 1$
while $m < n - 1$
    let $L^{(2m)}$ be a new $n \times n$ matrix
    $L^{(2m)} =$ EXTEND-SHORTEST-PATHS$(L^{(m)}, L^{(m)})$
    $m = 2m$
return $L^{(m)}$

Algorithm

$$
\begin{aligned}
L^{(1)} &= W, \\
L^{(2)} &= W^2 = W \cdot W, \\
L^{(4)} &= W^4 = W^2 \cdot W^2 \\
L^{(8)} &= W^8 = W^4 \cdot W^4, \\
&\vdots \\
L^{(2^{\lceil \lg(n-1) \rceil})} &= W^{2^{\lceil \lg(n-1) \rceil}} = W^{2^{\lceil \lg(n-1) \rceil - 1}} \cdot W^{2^{\lceil \lg(n-1) \rceil - 1}}
\end{aligned}
$$

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

```
n = W.rows
L^(1) = W
m = 1
while m < n - 1
    let L^(2m) be a new n × n matrix
    L^(2m) = EXTEND-SHORTEST-PATHS(L^(m), L^(m))
    m = 2m
return L^(m)
```

Algorithm – O(n^3 lg n)

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

$n = W.rows$
$L^{(1)} = W$
$m = 1$
while $m < n - 1$
  let $L^{(2m)}$ be a new $n \times n$ matrix
  $L^{(2m)} =$ EXTEND-SHORTEST-PATHS$(L^{(m)}, L^{(m)})$
  $m = 2m$
return $L^{(m)}$