More on Graph Algorithms

Andrew C. Lee

EECS, Syracuse

Contents

1. Reading: CLRS: Chapter 24, Section 1 and 3; Chapter 25, Section 1 and 2. Course notes from past semesters (CNPS: sections on graphs).
2. Single pair shortest path problem
3. All pair shortest path problem
4. Dynamic Programming

Dijkstra's Algorithm Revisited

1. It works for both directed and undirected graphs with non-negative weights
2. It uses the priority queue data structure to support its greedy strategy
3. It will not work for graphs with negative edges
4. A more general method can be used to find shortest path from a single source (How?)

Bellman Ford Algorithm I

Idea: Give a graph with n vertices, apply the *relax* procedure to each edge $n-1$ times (rely on **Path-relaxation property**)
The *Relax* procedure

$$
\begin{array}{l}
\textsc{Relax}(u, v, w) \\
\quad \textbf{if } v.d > u.d + w(u, v) \\
\qquad v.d = u.d + w(u, v) \\
\qquad v.\pi = u
\end{array}
$$

Figure : The Relax Procedure

Bellman Ford Algorithm II

Path-relaxation property (See CLRS, Ch 24, introduction) if $p = \langle v_0, v_1, \ldots, v_k \rangle$ is a path from $s = v_0$ to v_k and we relax the edges of p in the order of $(v_0, v_1), (v_1, v_2), \ldots, (v_{k-1}, v_k)$, then $d.v_k$ is the distance of the shortest path from s to v_k. This property holds regardless of any other relaxation steps that occur, even if they are intermixed with relaxations of the edges of p.

The Algorithm

```
BELLMAN-FORD(G, w, s)
  INIT-SINGLE-SOURCE(G, s)
  for i = 1 to |G.V| - 1
      for each edge (u, v) ∈ G.E
          RELAX(u, v, w)
  for each edge (u, v) ∈ G.E
      if v.d > u.d + w(u, v)
          return FALSE
  return TRUE
```

Figure : Pseudocode for Bellman Ford Algorithm

Note Bellman Ford detects negative cycle (How?)

Analysis

1. Let $G = (V, E)$, $|V| = n$, $|E| = m$.
2. $\Theta(nm)$ (detecting an negative cycle takes $O(m)$ time)

Floyd Warshall Algorithm I

1. It is a all pair shortest path algorithm
2. We assume there is no negative weight edges
3. It uses adjacency matrix representation
4. It uses a method called *dynamic programming*

Floyd Warshall Algorithm I

pseudocode

FLOYD-WARSHALL(W, n)

$D^{(0)} = W$

for $k = 1$ **to** n

  let $D^{(k)} = \left(d_{ij}^{(k)}\right)$ 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 $D^{(n)}$

Figure : Pseudocode for Floyd Warshall Algorithm

W is the *weight* matrix $(w_{i,j})$ defined as

$$w_{i,j} = \begin{cases} 0 & i = j \\ \text{The weight of the directed edge } (i,j) & i \neq j; (i,j) \in E \\ \infty & i \neq j; (i,j) \notin E \end{cases}$$

Floyd Warshall Algorithm II

ideas "Parameterize" the intermediate vertices $\{1, \ldots, k\}$ starting from $k = 1$ etc.

$$d_{i,j}^{k} = \text{ length of the shortest path from vertex } i \text{ to } j$$

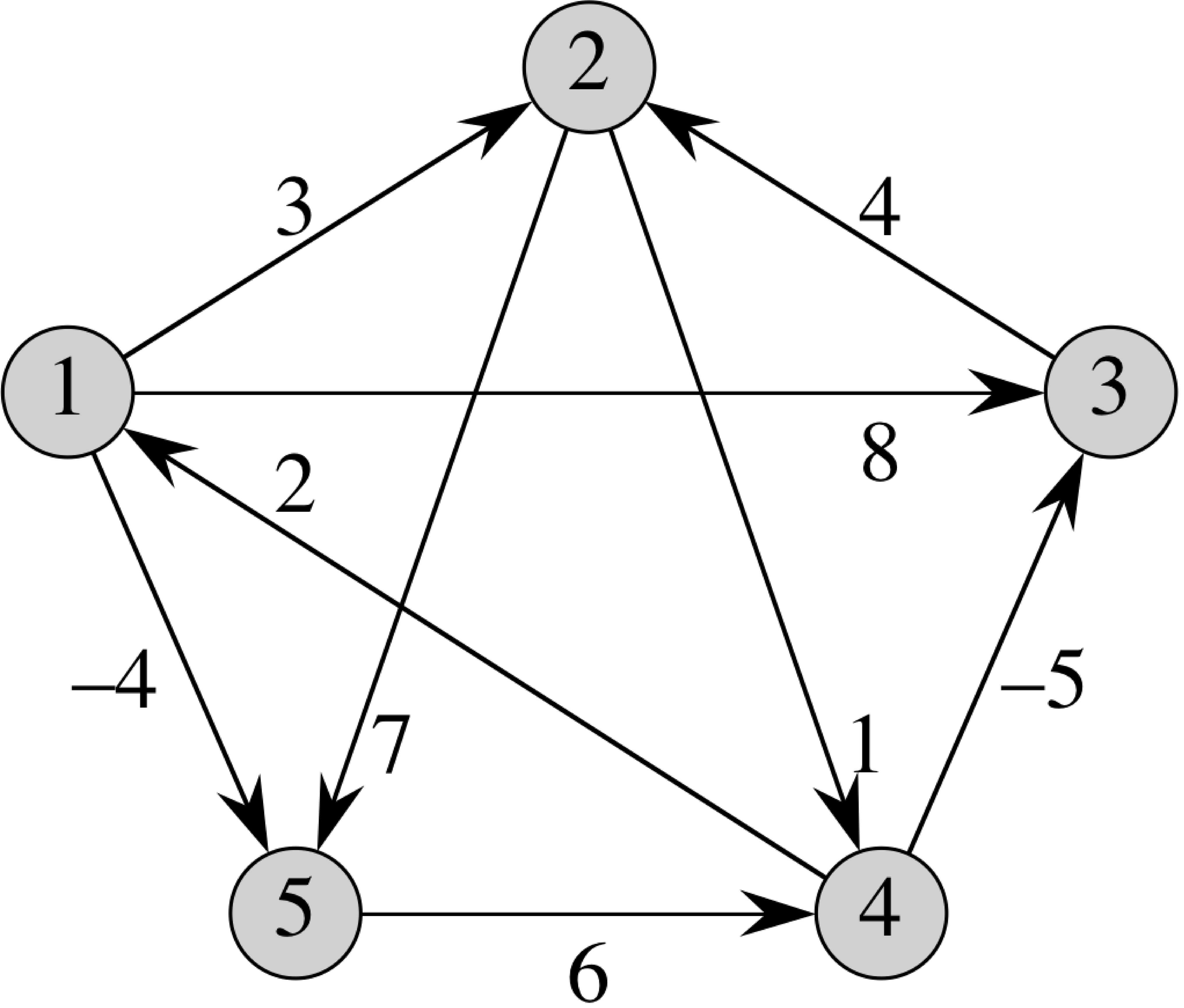
via intermediate vertices from {1, . . . , k}.

$$d_{i,j}^{k} = \begin{cases} w_{i,j} & k = 0 \\ \min\left(d_{i,j}^{k-1}, d_{i,k}^{k-1} + d_{k,j}^{k-1}\right) & k \geq 1 \end{cases}$$

Floyd Warshall Algorithm and dynamic programming

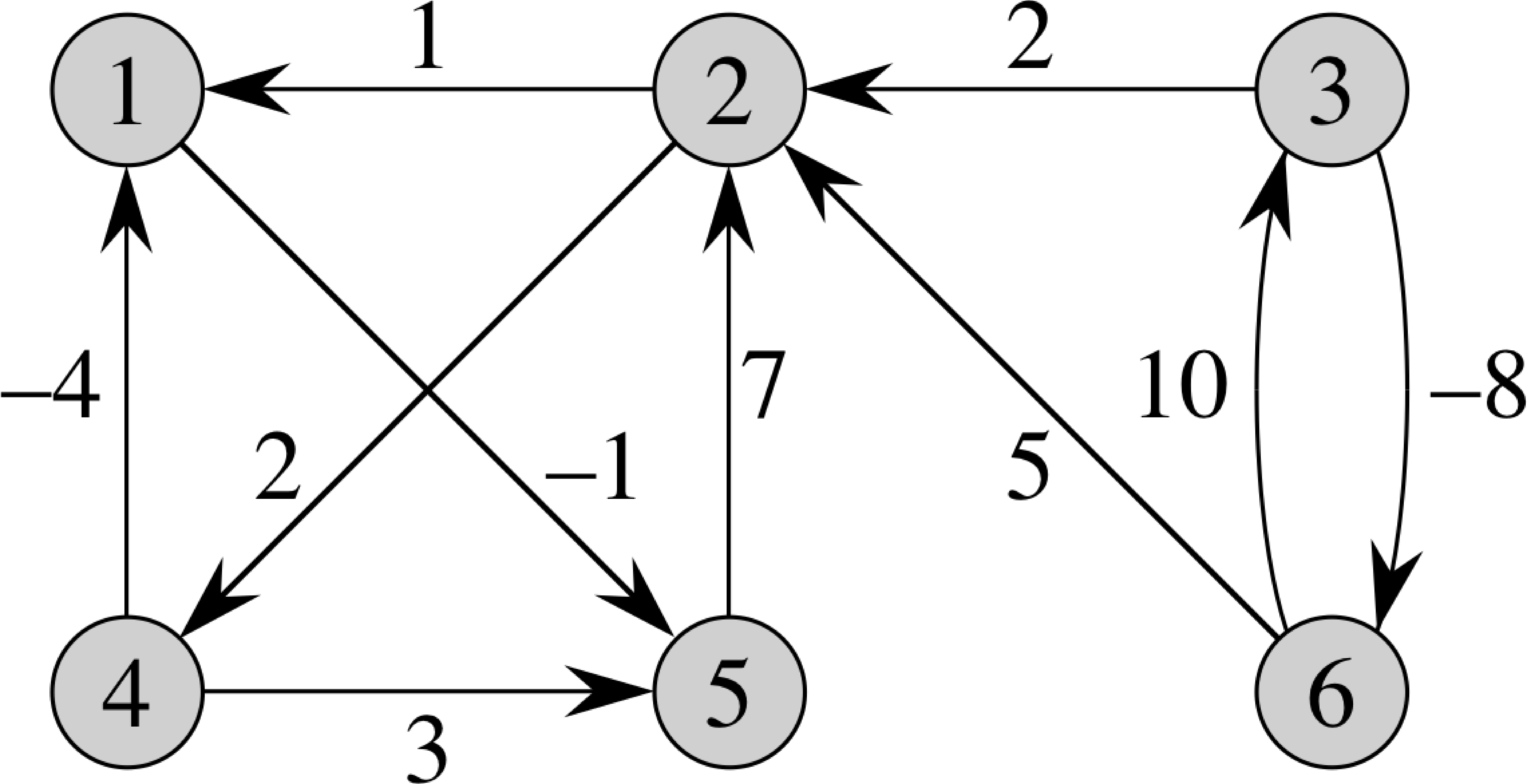
Discussions What are the main technique used in Floyd Warshall Algorithm ?

We will examine other examples that use the same technique (*dynamic programming*) in later lectures.



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

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


1
2
3
4
5
6
1
2
−4
2
−1
7
5
10
−8
3


all intermediate vertices in $\{1, 2, \ldots, k-1\}$
all intermediate vertices in $\{1, 2, \ldots, k-1\}$
k
p_1
p_2
i
j
p: all intermediate vertices in $\{1, 2, \ldots, k\}$

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

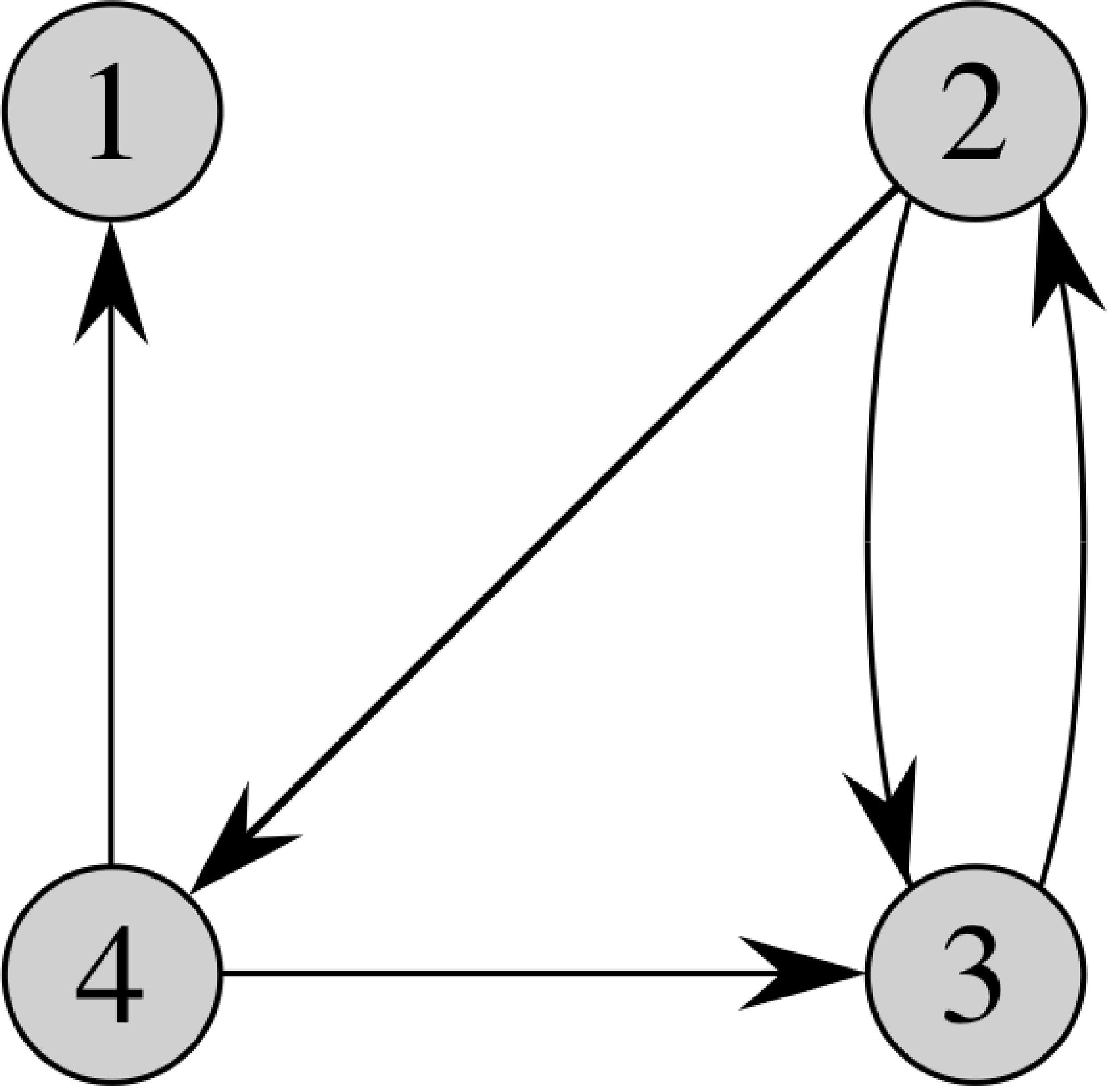
$$D^{(1)} = \begin{pmatrix} 0 & 3 & 8 & \infty & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & \infty & \infty \\ 2 & 5 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix} \qquad \Pi^{(1)} = \begin{pmatrix} \text{NIL} & 1 & 1 & \text{NIL} & 1 \\ \text{NIL} & \text{NIL} & \text{NIL}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

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

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

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

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



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

$$T^{(3)} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix} \quad T^{(4)} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$