Design and Analysis of Algorithms

Mohammad GANJTABESH

mgtabesh@ut.ac.ir

School of Mathematics, Statistics and Computer Science,
University of Tehran,
Tehran, Iran.

Graph Theoretical Problems

- Basec Definitions
- Graph Representation
- Graph Traversal (BFS, DFS)
- Topological Sort
- Strongly Connected Components
- Shortest Paths
  - Single-Source All Destination (Dijkstra and Bellman-Ford Algorithms)
  - All-Pairs (Matrix Multiplication, Floyd-Warshall, and Johnson's Algorithms)
- Minimum Spanning Tree (Kruskal, Prim)

All-Pairs Shortest Paths

Suppose that we are given a weighted directed graph $G = (V, E)$ with a weight function $w : E \mapsto R$. The All-Pairs shortest path problem is to find, for every pair of vertices $u, v \in V$, a shortest path from u to v.

The following solutions are available for this problem:

1. Running a single-source shortest-paths algorithm $|V|$ times, once for each vertex as a source.
   - Using Dijkstra's algorithm requires $O(|V|3)$ time complexity.
   - Using BELLMAN-FORD algorithm requires $O(|V|^2 \times |E|)$ time complexity, which for dense graphs it is $O(|V|^4)$.
2. Using Matrix Multiplication
3. Using FLOYD-WARSHALL algorithm

All-Pairs Shortest Paths: Matrix Multiplication

let $l_{ij}^{(m)}$ be the minimum weight of any path from vertex i to vertex j that contains at most m edges.

- For $m = 0$ we have:

$$l_{ij}^{(0)} = \begin{cases} 0 & \text{if } i = j, \\ \infty & \text{if } i \neq j. \end{cases}$$

- For $m \geq 1$ we have:

$$\begin{aligned} l_{ij}^{(m)} &= \min\left(l_{ij}^{(m-1)}, \min_{1 \leq k \leq n} \{ l_{ik}^{(m-1)} + w_{kj} \} \right) \\ &= \min_{1 \leq k \leq n} \{ l_{ik}^{(m-1)} + w_{kj} \} \end{aligned}$$

Now, any shortest path from i to j contains at most $n-1$ edges. Also, any longer path can not have lower weight. So, $\delta(i,j) = l_{ij}^{(n-1)} = l_{ij}^{(n)} = l_{ij}^{(n+1)} = \cdots$.

All-Pairs Shortest Paths: Matrix Multiplication

Suppose that $W = (w_{ij})$ is the weight matrix for graph $G = (V, E)$. In order to solve All-Pairs shortest path problem, we compute a series of matrices $L^{(1)}, L^{(2)}, \cdots, L^{(n-1)}$, where for $m = 1, 2, \cdots, n-1$, we have $L^{(m)} = (l_{ij}^{(m)})$.

- The final matrix $L^{(n-1)}$ contains the actual shortest-path weights.
- $l_{ij}^{(1)} = w_{ij}$ for all vertices $i, j \in V$, and so $L^{(1)} = W$.

```
EXTEND-SHORTEST-PATHS(L, W)
n ← rows[L]
let L′ = (l′_ij) be an n × n matrix
for i ← 1 to n
    do for j ← 1 to n
           do l′_ij ← ∞
              for k ← 1 to n
                  do l′_ij ← min(l′_ij, l_ik + w_kj)
return L′
```

Relation with Matrix Multiplication: $\begin{cases} + \longrightarrow \min \\ \cdot \longrightarrow + \end{cases}$. The time Complexity is $O(n^3)$.

All-Pairs Shortest Paths: Matrix Multiplication

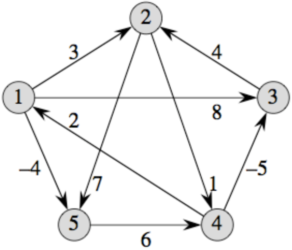
SLOW-ALL-PAIRS-SHORTEST-PATHS(W)

$n \leftarrow rows[W]$
$L^{(1)} \leftarrow W$
for $m \leftarrow 2$ **to** $n - 1$
    do $L^{(m)} \leftarrow$ EXTEND-SHORTEST-PATHS$(L^{(m-1)}, W)$
return $L^{(n-1)}$

The total time complexity is $O(n^4)$.

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

All-Pairs Shortest Paths: Matrix Multiplication



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

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

All-Pairs Shortest Paths: Matrix Multiplication

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

$n \leftarrow rows[W]$
$L^{(1)} \leftarrow W$
$m \leftarrow 1$
while $m < n - 1$
    do $L^{(2m)} \leftarrow$ EXTEND-SHORTEST-PATHS$(L^{(m)}, L^{(m)})$
        $m \leftarrow 2m$
return $L^{(m)}$

The total time complexity is $O(n^3 \log n)$.

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

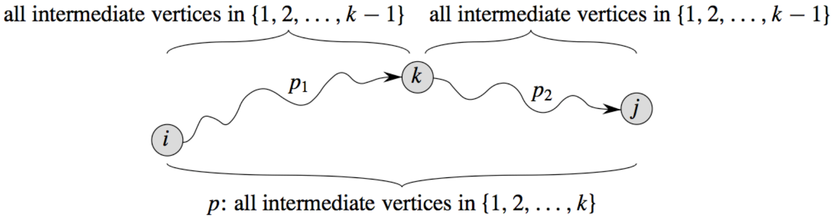
All-Pairs Shortest Paths: Floyd-Warshall algorithm

Definition

The **intermediate** vertex of a simple path $p = \langle v_1, v_2, \cdots, v_l \rangle$ is any vertex of p other than v_1 or v_l, that is, any vertex in the set $\{v_2, v_3, \cdots, v_{l-1}\}$.

- Suppose that $V = \{1, 2, \cdots, n\}$ and $\{1, 2, \cdots, k\} \subset V$.
- For any pair of vertices $i, j \in V$, consider all paths from i to j whose intermediate vertices are all drawn from $\{1, 2, \cdots, k\}$, and let p be a minimum-weight path among them.
  - If k is not an intermediate vertex of path p, then a shortest path from vertex i to vertex j with all intermediate vertices in the set $\{1, 2, \cdots, k-1\}$ is also a shortest path from i to j.
  - If k is an intermediate vertex of path p, then we break down p into $i \overset{p_1}{\leadsto} k \overset{p_2}{\leadsto} j$, where p_1 is a shortest path from i to k, p_2 is a shortest path from k to j, and their intermediate vertices are in the set $\{1, 2, \cdots, k-1\}$.

All-Pairs Shortest Paths: Floyd-Warshall algorithm



Let $d_{ij}^{(k)}$ be the weight of a shortest path from vertex i to vertex j for which all intermediate vertices are in the set $\{1, 2, \cdots, k\}$. A recursive definition following the previous discussion is given by:

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

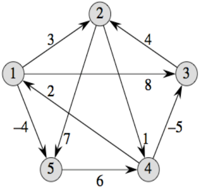
All-Pairs Shortest Paths: Floyd-Warshall algorithm

FLOYD-WARSHALL(W)

$n \leftarrow rows[W]$
$D^{(0)} \leftarrow W$
for $k \leftarrow 1$ **to** n
  do for $i \leftarrow 1$ **to** n
    do for $j \leftarrow 1$ **to** n
      do $d_{ij}^{(k)} \leftarrow \min\left(d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)}\right)$
return $D^{(n)}$

The total time complexity is $\Theta(n^3)$.

All-Pairs Shortest Paths: Floyd-Warshall algorithm



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

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

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

Exercises

1. Show how to express the single-source shortest-paths problem as a product of matrices and a vector. Describe how evaluating this product corresponds to a Bellman-Ford-like algorithm.
2. Suppose we also wish to compute the vertices on shortest paths in the algorithms of this lecture. Show how to compute the predecessor matrix Π from the completed matrix L of shortest-path weights in $O(n^3)$ time.
3. The vertices on shortest paths can also be computed at the same time as the shortest-path weights. Let us define $\pi_{ij}^{(m)}$ to be the predecessor of vertex j on any minimum-weight path from i to j that contains at most m edges. Modify EXTEND-SHORTEST-PATHS and SLOW-ALL-PAIRS-SHORTEST-PATHS to compute the matrices $\Pi^{(1)}, \Pi^{(2)}, \cdots, \Pi^{(n-1)}$ as the matrices $L^{(1)}, L^{(2)}, \cdots, L^{(n-1)}$ are computed.
4. Modify FASTER-ALL-PAIRS-SHORTEST-PATHS so that it can detect the presence of a negative-weight cycle.
5. How can the output of the Floyd-Warshall algorithm be used to detect the presence of a negative-weight cycle?
6. The transitive closure of $G = (V, E)$ is defined as the graph $G^* = (V, E^*)$, where

$$E^* = \{(i, j) \mid \text{ there is a path from vertex } i \text{ to vertex } j \text{ in } G\}.$$

   a) Describe how we can compute the transitive closure of a graph $G = (V, E)$.
   b) Give an $O(|V| \cdot |E|)$-time algorithm for computing the transitive closure of a directed graph $G = (V, E)$.