Shortest Paths

Jie Wang

University of Massachusetts Lowell
Department of Computer Science

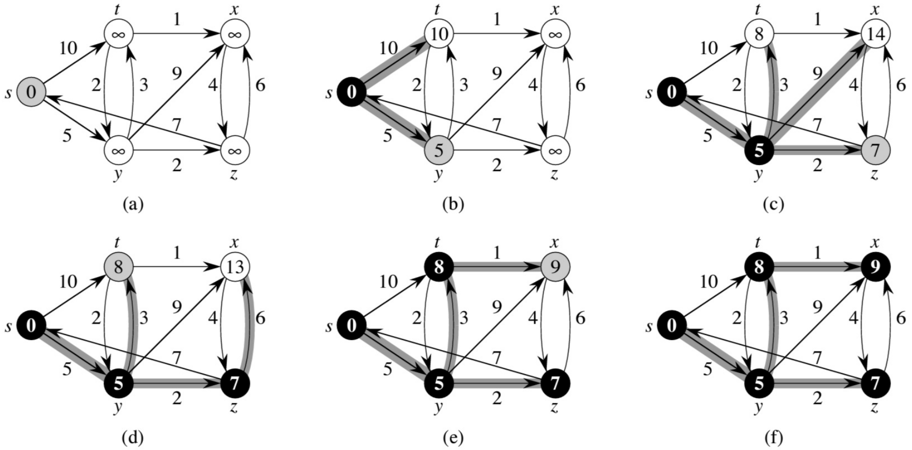
Greedy Dijkstra's Algorithm

Let $G = (V, E)$ be a weighted, directed graph with nonnegative weights.

$\textsc{Dijkstra}(G, w, s)$

for each vertex $v \in G.V$
    $v.d = \infty$
    $v.\pi = \text{NIL}$
$s.d = 0$
$S = \emptyset$
$Q = G.V$
while $Q \neq \emptyset$
    $u = \textsc{Extract-Min}(Q)$
    $S = S \cup \{u\}$
    for each vertex $v \in G.Adj[u]$
        if $v.d > u.d + w(u, v)$
            $v.d = u.d + w(u, v)$
            $v.\pi = u$

Execution of Dijkstra



Correctness Proof of Dijkstra

Theorem 24.6. Dijkstra's Algorithm terminates with $u.d = \delta(s, u)$ for all $u \in V$, where $\delta(s, u)$ is the length of the shortest path from s to u.

Proof. By contradiction: u is added to S but $u.d \neq \delta(s, u)$. That is, there is a path from s to u but when u is selected, $u.d > \delta(s, u)$.

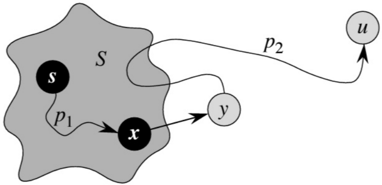


Figure 24.7 The proof of Theorem 24.6. Set S is nonempty just before vertex u is added to it. We decompose a shortest path p from source s to vertex u into $s \overset{p_1}{\leadsto} x \rightarrow y \overset{p_2}{\leadsto} u$, where y is the first vertex on the path that is not in S and $x \in S$ immediately precedes y. Vertices x and y are distinct, but we may have $s = x$ or $y = u$. Path p_2 may or may not reenter set S.

Correctness Proof of Dijkstra Continued

- Note that for all $v \in V$: $v.d \geq \delta(s, v)$.
- Let p be a shortest path from s to u, and y the first node $\in V - S$ on p. That is,
$$y.d = \delta(s, v).$$
- Thus, $u.d > \delta(s, u) = \delta(s, y) + \delta(y, u) = y.d + \delta(y, u) \geq y.d$.
- Hence, u should have not been chosen, a contradiction.

Runtime: $O(|V| \log |V| + |E|)$.

Floyd-Warshall's Algorithm

Finding all-pairs shortest paths on a directed graph $G = (V, E)$.

- Using DP.
- Formulation: Let $d_{ij}^{(k)}$ be the weight of a shortest path from node i to node j, where all intermediate nodes on the path are a subset of $\{1, 2, \cdots, k\}$.
  - There are $\Theta(n^3)$ subproblems.
  - Want to compute $d_{ij}^{(n)}$ for all pairs (i, j).
- Localization:

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

Bottom Up

Let $D^{(n)}$ denote the matrix of shortest-path weights.

$\text{Floyd-Warshall}(W)$

$n = W.rows$
$D^{(0)} = W$
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 $D^{(n)}$.

Runtime: $\Theta(n^3)$.

Shortest-Path Construction

- Let $\Pi^{(k)} = \left(\pi_{ij}^{(k)}\right)_{n\times n}$, where $\pi_{ij}^{(k)}$ is the predecessor of node j on a shortest path from node i with all immediate nodes in the path in $\{1, 2, \cdots, k\}$.
- Let

$$\pi_{ij}^{(0)} = \begin{cases} \text{NIL}, & \text{if } i = j \text{ or } w_{ij} = \infty, \\ i, & \text{if } i \neq j \text{ and } w_{ij} < \infty. \end{cases}$$

- Let

$$\pi_{ij}^{(k)} = \begin{cases} \pi_{ij}^{(k-1)}, & \text{if } d_{ij}^{(k-1)} \leq d_{ik}^{(k-1)} + d_{kj}^{(k-1)}, \\ \pi_{kj}^{(k-1)}, & \text{if } d_{ij}^{(k-1)} > d_{ik}^{(k-1)} + d_{kj}^{(k-1)}. \end{cases}$$

Transitive Closure

- Determine if G contains a path from i to j for all pairs (i, j).
- Define a **transitive closure** of G as the graph $G^* = (V, E^*)$, where

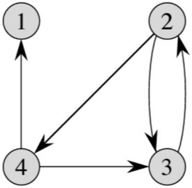
$$E^* = \{(i, j) \mid \text{there is a path from node } i \text{ to node } j \text{ in } V\}.$$

- Let

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

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

Example



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

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

Reweighting and Johnson's Algorithm

- Reweighting: Obtain nonnegative weight cycles while preserving shortest paths
- The following reweighting preserves shortest paths for any function $h : V \to R$:

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

  - Reason: Let $p = \langle v_0, v_1, \ldots, v_k \rangle$ be any path from v_0 to v_k. Then

$$\begin{aligned}\hat{w}(p) &= \sum_{i=1}^{k} (w(v_{i-1}, v_i) + h(v_{i-1}) - h(v_i)) \\ &= w(p) + h(v_0) - h(v_k).\end{aligned}$$

  Thus, $\hat{w}(p)$ is the smallest among all paths from v_0 to v_k under the new weights iff $w(p)$ is so under the old weights.

Nonnegative New Weight

- Add a dummy node s to every node $u \in G$ with $w(s, u) = 0$.
- Let $h(u) = \delta(s, u)$.
- By triangle inequality: $\delta(s, v) \leq \delta(s, u) + w(u, v)$, thus,

$$\hat{w}(u, v) = w(u, v) + h(u) - h(v) \geq 0.$$

Johnson's Algorithm

$\textsc{Johnson}(G, w)$

Add a dummy node s to every node $u \in G$ with $w(s, u) = 0$.
if $\textsc{Bellman-Ford}(G', w, s) ==$ FALSE
    Print "There is a negative-weight cycle"
else for each $v \in G'.V$
        Set $h(u) = \delta(s, u)$
    for each $(u, v) \in G'.E$
        Set $\hat{w}(u, v) = w(u, v) + h(u) - h(v)$
    for each $u \in G.V$
        run $\textsc{Dijkstra}(G, \hat{w}, u)$ to compute $\hat{\delta}(u, v)$ for all $v \in G.V$
        for each $v \in G.V$
            $d_{uv} = \hat{\delta}(u, v) + h(v) - h(u)$
return $D = (d_{uv})$