CH08-320201

# Algorithms and Data Structures

ADS

## Lecture 23

Dr. Kinga Lipskoch

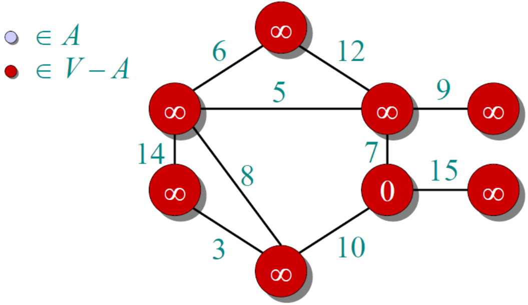
Spring 2019

# Prim's Algorithm Pseudocode

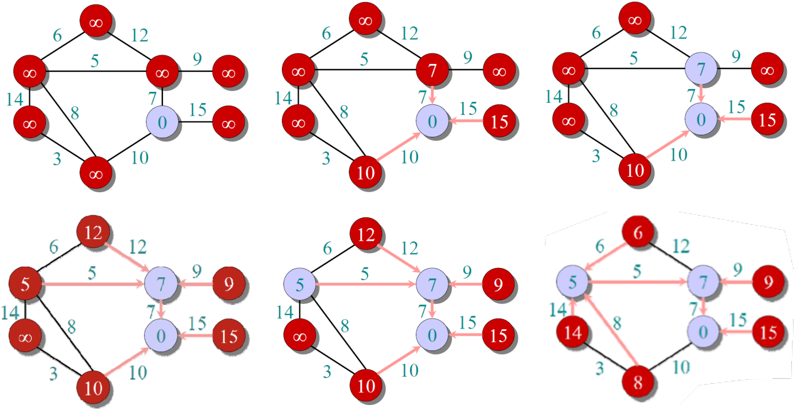
```
Q ← V
key[v] ← ∞ for all v ∈ V
key[s] ← 0 for some arbitrary s ∈ V
while Q ≠ ∅
    do u ← EXTRACT-MIN(Q)
        for each v ∈ Adj[u]
            do if v ∈ Q and w(u, v) < key[v]
                then key[v] ← w(u, v)
                     π[v] ← u
```

- The output is provided by storing predecessors $\pi[v]$ of each node $v$.
- The set $\{(v, \pi[v]) | v \in V\}$ forms the MST.

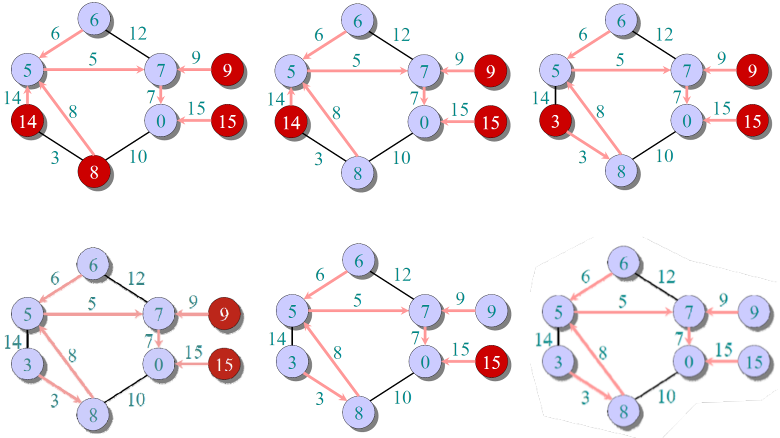
# Example (1)



# Example (2)


∞ 6 12 ∞ 5 ∞ 9 ∞ 14 7 8 15 ∞ 0 ∞ 3 10 ∞
∞ 6 12 ∞ 5 7 9 ∞ 14 7 8 15 ∞ 0 15 3 10 10
∞ 6 12 ∞ 5 7 9 ∞ 14 7 8 15 ∞ 0 15 3 10 10
12 6 12 5 5 7 9 9 14 7 8 15 ∞ 0 15 3 10 10
12 6 12 5 5 7 9 9 14 7 8 15 ∞ 0 15 3 10 10
6 6 12 5 5 7 9 9 14 7 8 15 14 0 15 3 10 8

# Example (3)


6 12 6 5 5 7 9 9 14 8 7 15 14 0 15 3 8 10
6 12 6 5 5 7 9 9 14 8 7 15 14 0 15 3 8 10
6 12 6 5 5 7 9 9 14 8 7 15 3 0 15 3 8 10
6 12 6 5 5 7 9 9 14 8 7 15 3 0 15 3 8 10
6 12 6 5 5 7 9 9 14 8 7 15 3 0 15 3 8 10
6 12 6 5 5 7 9 9 14 8 7 15 3 0 15 3 8 10

# Complexity Analysis (1)

$\Theta(V)$ total:

$Q \leftarrow V$

$key[v] \leftarrow \infty$ for all $v \in V$

$key[s] \leftarrow 0$ for some arbitrary $s \in V$

$|V|$ times:

**while** $Q \neq \emptyset$

  **do** $u \leftarrow$ EXTRACT-MIN$(Q)$

  $degree(u)$ times:

   **for each** $v \in Adj[u]$

    **do if** $v \in Q$ and $w(u, v) < key[v]$

     **then** $\boxed{key[v] \leftarrow w(u, v)}$

      $\pi[v] \leftarrow u$

$\Theta(E)$ implicit DECREASE-KEY's.

Notation Θ(V) means Θ(|V|).

# Complexity Analysis (2)

$$\text{Time} = \Theta(V) \cdot T_{\text{EXTRACT-MIN}} + \Theta(E) \cdot T_{\text{DECREASE-KEY}}$$

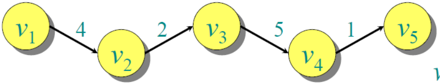
| $Q$ | $T_{\text{EXTRACT-MIN}}$ | $T_{\text{DECREASE-KEY}}$ | Total |
|---|---|---|---|
| min-heap | $O(\lg V)$ | $O(\lg V)$ | $O(E \lg V)$ |
| array | $O(V)$ | $O(1)$ | $O(V^2)$ |

# Definition: Path

- Consider a directed graph $G = (V, E)$, where each edge $e \in E$ is assigned a non-negative weight $w : E \to \mathbb{R}^+$.
- A path is a sequence of vertices in the graph, where two consecutive vertices are connected by a respective edge.
- The weight of a path $p = (v_1, ..., v_k)$ is defined by

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

- Example:




$w(p) = 12$

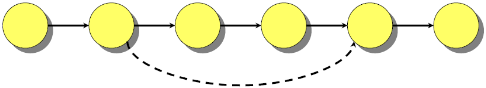
# Definition: Shortest Path

- A shortest path from a vertex $u$ to a vertex $v$ in a graph $G$ is a path of minimum weight.
- The weight of a shortest path from $u$ to $v$ is defined as $\delta(u, v) = \min\{w(p) : p \text{ is a path from } u \text{ to } v\}$.
- Note that $\delta(u, v) = \infty$, if no path from $u$ to $v$ exists.
- Why of interest?
  One example is finding a shortest route in a road network.

# Optimal Substructure

Theorem:
A subpath of a shortest path is a shortest path.
Proof:

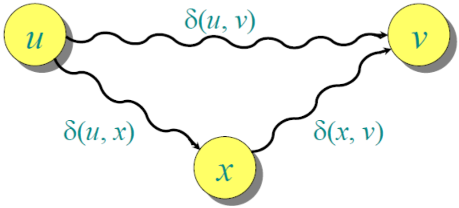
- Let $p = (v_1, \ldots, v_k)$ be a shortest path and $q = (v_i, \ldots, v_j)$ a subpath of $p$.
- Assume that $q$ is not a shortest path.
- Then, there exists a shorter path from $v_i$ to $v_j$ than $q$.
- But then, there is also a shorter path from $v_1$ to $v_k$ than $p$. Contradiction.



# Triangle Inequality

Theorem:
For all $u, v, x \in V$, we have that $\delta(u, v) \leq \delta(u, x) + \delta(x, v)$.

Proof:



# (Single-Source) Shortest Paths

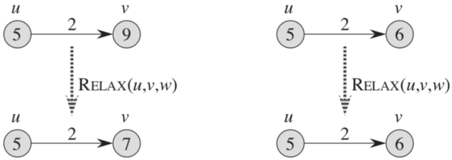
Problem:
Given a source vertex $s \in V$, find for all $v \in V$ the shortest-path weights $\delta(s, v)$.

Idea: Greedy approach.

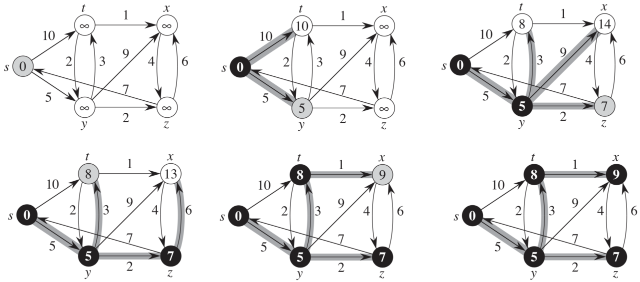
1. Maintain a set $S$ of vertices whose shortest-path distances from $s$ are known.
2. At each step, add to $S$ the vertex $v \in V \setminus S$ whose distance estimate from $s$ is minimal.
3. Update the distance estimates of vertices adjacent to $v$.

# Dijkstra's Algorithm

```
d[s] := 0
for each v ∈ V\{s}
  d[v] := infinity
S := Ø
Q := V   // min-priority queue maintaining V \ S.
while Q != Ø
    u := Extract-Min(Q)
    S := S U {u}
    for each v ∈ Adj[u]
        if d[v] > d[u] + w(u,v)       // *****
        then d[v] := d[u] + w(u,v)   // Relaxation
            pi[v] := u                 // *****
```



# Example Dijkstra's Algorithm




```
while Q != Ø
    u := Extract-Min(Q)
    S := S U {u}
    for each v ∊ Adj[u]
        if d[v] > d[u] + w(u,v)
        then d[v] := d[u] + w(u,v)
            pi[v] := u
```

S = {s, y, z, t, x}

# Correctness of Dijkstra's Algorithm

Correctness can be shown in 3 steps:

(i) $d[v] \geq \delta(s, v)$ at all steps (for all $v$)

(ii) $d[v] = \delta(s, v)$ after relaxation from $u$,

(iii) if $(u, v)$ on shortest path (for all $v$) algorithm terminates with $d[v] = \delta(s, v)$

# Correctness (i)

Lemma:

- Initializing $d[s] = 0$ and $d[v] = \infty$ for all $v \in V \setminus \{s\}$ establishes $d[v] \geq \delta(s, v)$ for all $v \in V$.
- This invariant is maintained over any sequence of relaxation steps.

Proof:

Suppose the Lemma is not true, then let $v$ be the first vertex for which $d[v] < \delta(s, v)$ and let $u$ be the vertex that caused $d[v]$ to change by $d[v] = d[u] + w(u, v)$. Then,

$$
\begin{aligned}
d[v] &< \delta(s, v) && \text{supposition} \\
&\leq \delta(s, u) + \delta(u, v) && \text{triangle inequality} \\
&\leq \delta(s,u) + w(u, v) && \text{sh. path} \leq \text{specific path} \\
&\leq d[u] + w(u, v) && v \text{ is first violation}
\end{aligned}
$$

Contradiction.

# Correctness (ii)

Lemma:

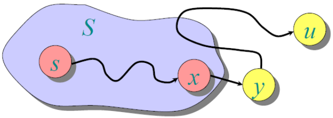
- Let $u$ be $v$'s predecessor on a shortest path from $s$ to $v$.
- Then, if $d[u] = \delta(s, u)$, we have $d[v] = \delta(s, v)$ after the relaxation of edge $(u, v)$.

Proof:

- Observe that $\delta(s, v) = \delta(s, u) + w(u, v)$.
- Suppose that $d[v] > \delta(s, v)$ before relaxation (else: done).
- Then, $d[v] > \delta(s, v) = \delta(s, u) + w(u, v) = d[u] + w(u, v)$ (if clause in the algorithm).
- Thus, the algorithm sets $d[v] = d[u] + w(u, v) = \delta(s, v)$.

# Correctness (iii)

**Theorem**:
Dijkstra's algorithm terminates with $d[v] = \delta(s, v)$ for all $v \in V$.




**Proof**:

- It suffices to show that $d[v] = \delta(s, v)$ for every $v \in V$ when $v$ is added to $S$.
- Suppose $u$ is the first vertex added to $S$ with $d[u] > \delta(s, u)$.
- Let $y$ be the first vertex in $V \setminus S$ along the shortest path from $s$ to $u$, and let $x$ be its predecessor.
- Then, $d[x] = \delta(s, x)$ and $d[y] = \delta(s, y) \leq \delta(s, u) < d[u]$.
- But we chose $u$ such that $d[u] \leq d[y]$. Contradiction.