Review: Single Source Shortest Paths with Non-Negative Weights

Yufei Tao

Department of Computer Science and Engineering
Chinese University of Hong Kong

We will now commence our discussion on the **single source shortest path** (SSSP) problem. This lecture will start with **Dijkstra's algorithm**, which should have been covered in CSCI2100.

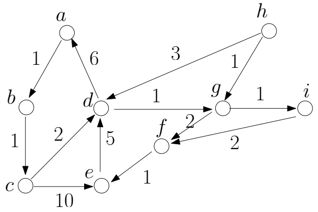
Weighted Graphs

Let $G = (V, E)$ be a simple directed graph.

Let w be a function that maps each edge $e \in E$ to a **non-negative** integer value $w(e)$, which we call the **weight** of e.

G and w together define a **weighted simple directed graph**.

Example



The integer on each edge indicates its weight. For example, $w(d, g) = 1$, $w(g, f) = 2$, and $w(c, e) = 10$.

Shortest Path

Consider a path in G: $(v_1, v_2), (v_2, v_3), ..., (v_\ell, v_{\ell+1})$, for some integer $\ell \geq 1$. We define the path's **length** as

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

A **shortest path** from u to v has the minimum length among all the paths from u to v. Denote by $spdist(u, v)$ the length of a shortest path from u to v.

If v is unreachable from u, $spdist(u, v) = \infty$.

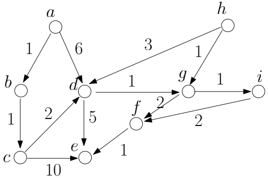
Single Source Shortest Path (SSSP) with Non-Negative Weights

Let $G = (V, E)$ be a simple directed graph, where function w maps every edge of E to a non-negative value. Given a **source vertex** s in V, we want to find a shortest path from s to t for **every** vertex $t \in V$ reachable from s.

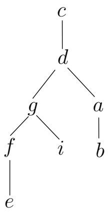
The output is a **shortest path tree** T:

- The vertex set of T is V.
- The root of T is s.
- For each node $u \in V$, the root-to-u path of T is a shortest path from s to u in G.

Example



A shortest path tree for source vertex c:



Edge Relaxation

For every vertex $v \in V$, we will — at all times — maintain a value $dist(v)$ equal to the shortest path length from s to v **found so far**.

Relaxing an edge (u, v) means:

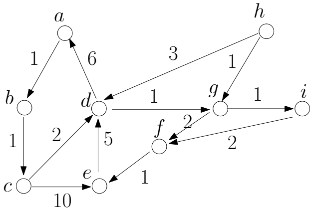
- If $dist(v) \leq dist(u) + w(u, v)$, do nothing;
- Otherwise, reduce $dist(v)$ to $dist(u) + w(u, v)$.

Dijkstra's Algorithm

1. Set $parent(v) \leftarrow$ nil for all vertices $v \in V$
2. Set $dist(s) \leftarrow 0$ and $dist(v) \leftarrow \infty$ for each vertex $v \in V \setminus \{s\}$
3. Set $S \leftarrow V$
4. Repeat the following until S is empty:
   - Remove from S the vertex u with the **smallest** $dist(u)$.
   - Relax every outgoing edge (u, v) of u.
     If $dist(v)$ drops after the relaxation, set $parent(v) \leftarrow u$.

Example

Suppose that the source vertex is c.

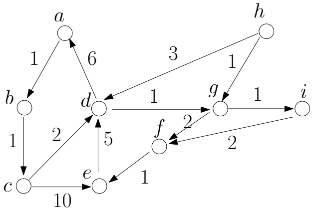


vertex v	$dist(v)$	$parent(v)$
a	∞	nil
b	∞	nil
c	0	nil
d	∞	nil
e	∞	nil
f	∞	nil
g	∞	nil
h	∞	nil
i	∞	nil

$S = \{a, b, c, d, e, f, g, h, i\}$.

Example

Relax the out-going edges of c.

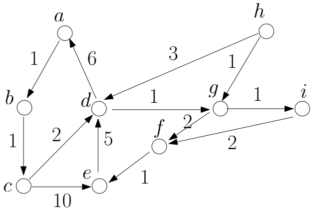


vertex v	$dist(v)$	$parent(v)$
a	∞	nil
b	∞	nil
c	0	nil
d	2	c
e	10	c
f	∞	nil
g	∞	nil
h	∞	nil
i	∞	nil

$S = \{a, b, d, e, f, g, h, i\}$.

Example

Relax the out-going edges of d.

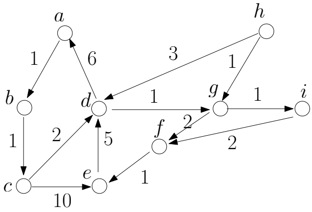


vertex v	$dist(v)$	$parent(v)$
a	8	d
b	∞	nil
c	0	nil
d	2	c
e	10	c
f	∞	nil
g	3	d
h	∞	nil
i	∞	nil

$S = \{a, b, e, f, g, h, i\}$.

Example

Relax the out-going edges of g.

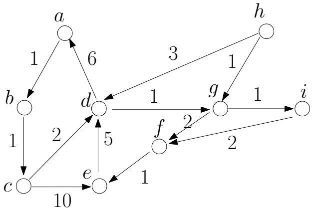


vertex v	$dist(v)$	$parent(v)$
a	8	d
b	∞	nil
c	0	nil
d	2	c
e	10	c
f	5	g
g	3	d
h	∞	nil
i	4	g

$S = \{a, b, e, f, h, i\}$.

Example

Relax the out-going edges of i.

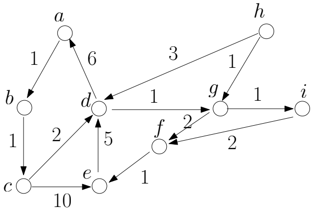


vertex v	$dist(v)$	$parent(v)$
a	8	d
b	∞	nil
c	0	nil
d	2	c
e	10	c
f	5	g
g	3	d
h	∞	nil
i	4	g

$S = \{a, b, e, f, h\}$.

Example

Relax the out-going edges of f.

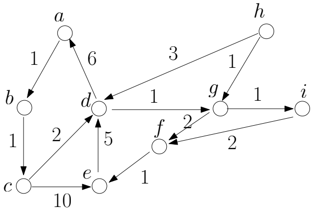


vertex v	$dist(v)$	$parent(v)$
a	8	d
b	∞	nil
c	0	nil
d	2	c
e	6	f
f	5	g
g	3	d
h	∞	nil
i	4	g

$S = \{a, b, e, h\}$.

Example

Relax the out-going edges of e.

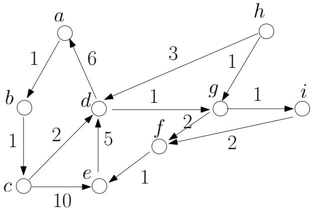


vertex v	$dist(v)$	$parent(v)$
a	8	d
b	∞	nil
c	0	nil
d	2	c
e	6	f
f	5	g
g	3	d
h	∞	nil
i	4	g

$S = \{a, b, h\}$.

Example

Relax the out-going edges of a.

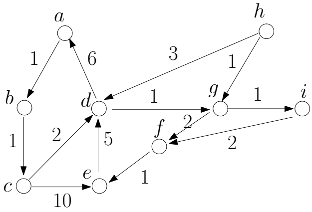


vertex v	$dist(v)$	$parent(v)$
a	8	d
b	9	a
c	0	nil
d	2	c
e	6	f
f	5	g
g	3	d
h	∞	nil
i	4	g

$S = \{b, h\}$.

Example

Relax the out-going edges of b.

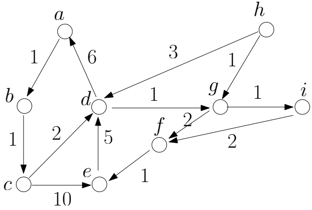


vertex v	$dist(v)$	$parent(v)$
a	8	d
b	9	a
c	0	nil
d	2	c
e	6	f
f	5	g
g	3	d
h	∞	nil
i	4	g

$S = \{h\}$.

Example

Relax the out-going edges of h.

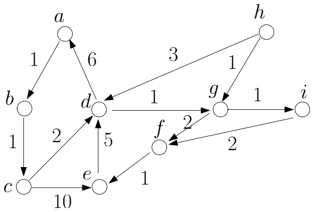


vertex v	$dist(v)$	$parent(v)$
a	8	d
b	9	a
c	0	nil
d	2	c
e	6	f
f	5	g
g	3	d
h	∞	nil
i	4	g

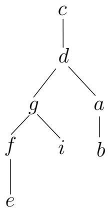
$S = \{\}$.
All the shortest path distances are now final.

Constructing the Shortest Path Tree

For every vertex v, if $u = parent(v)$ is not nil, then make v a child of u.



vertex v	$parent(v)$
a	d
b	a
c	nil
d	c
e	f
f	g
g	d
h	nil
i	g



You should be able to implement Dijkstra's algorithm to make sure that it runs in $O((|V| + |E|) \cdot \log |V|)$ time.

- Using advanced (graduate-level) data structures, we can reduce the time to $O(|V| \log |V| + |E|)$.

Dijkstra's algorithm does **not** work if edges can take negative weights.