Shortest path Algorithm

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

- The shortest path between two vertices is a path with the shortest length (least number of edges)
- Breadth-first-search is an algorithm for finding shortest (link-distance) paths from a single source vertex to all other vertices.

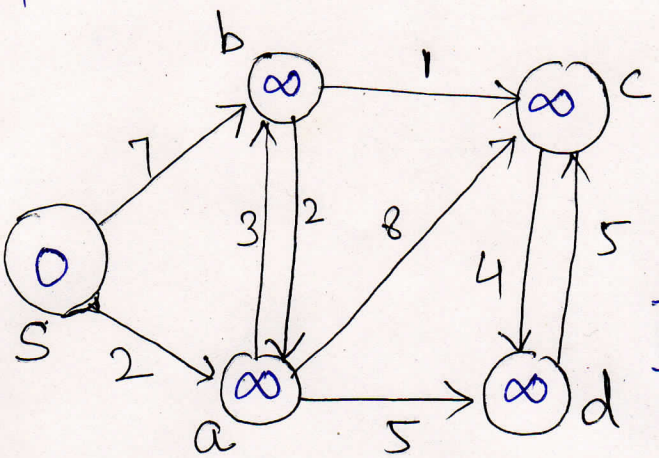
Dijkstra's Algorithm

- Maintain an estimate $d[v]$ of the length $\delta(s,v)$ of the shortest path for each vertex v.
- Always $d[v] \geq \delta(s,v)$ and $d[v]$ equals the length of a known path
  ($d[v]=\infty$ if we have no paths so far)
- Initially $d[s]=0$ and all the other $d[v]$ values are set to ∞. The algorithm will then process the vertices one by one in some order.

  The processed vertex's estimate will be valided as being real shortest distance, i.e. $d[v]=\delta(s,v)$

Here "processing a vertex u" means finding new paths and updating $d[v]$ for all $v \in Adj[u]$ if necessary
The process by which an estimate is updated is called relaxation.

Example:

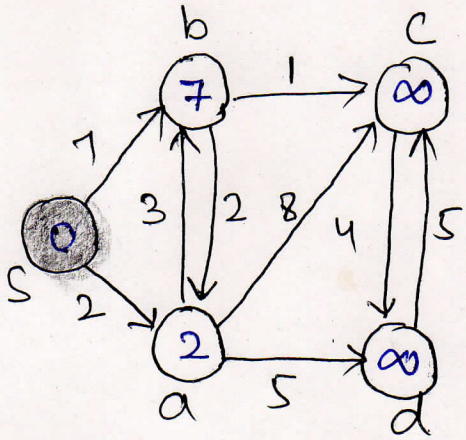


Step0: Initialization

v	s	a	b	c	d
d[v]	0	∞	∞	∞	∞
pred[v]	nil	nil	nil	nil	nil
color[v]	W	W	W	W	W

Priority Queue

v	s	a	b	c	d
d[v]	0	∞	∞	∞	∞

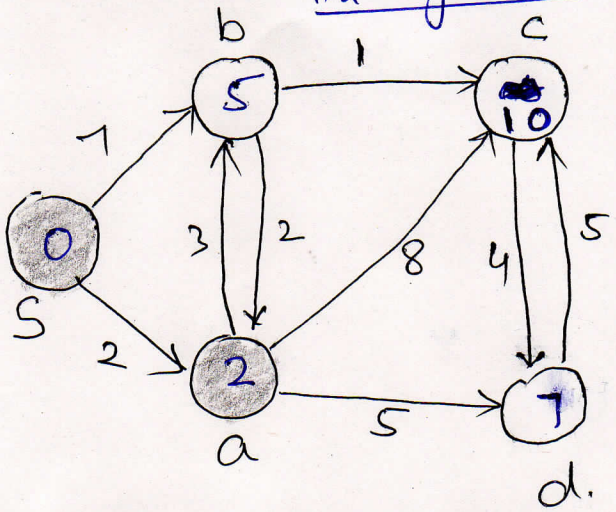


Step1:

v	s	a	b	c	d
d[v]	0	2	7	∞	∞
pred[v]	nil	s	s	nil	nil
Color[v]	B	W	W	W	W

Priority Queue

v	s	a	b	c	d
d[v]		2	7	∞	∞

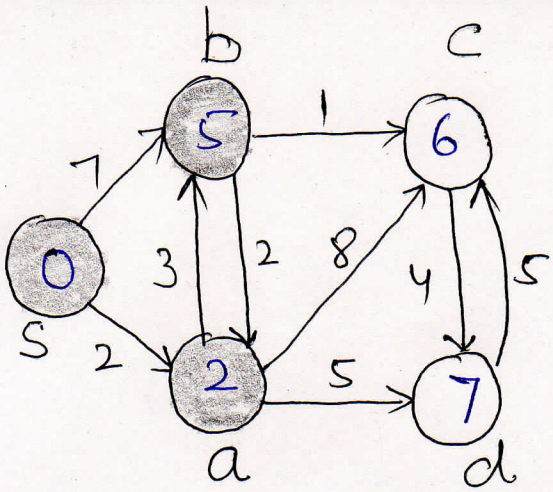


Step2

v	s	a	b	c	d
d[v]	0	2	5	10	7
pred[v]	nil	s	a	a	a
Color[v]	B	B	W	W	W

Priority Queue

v	b	c	d
d[v]	5	10	7

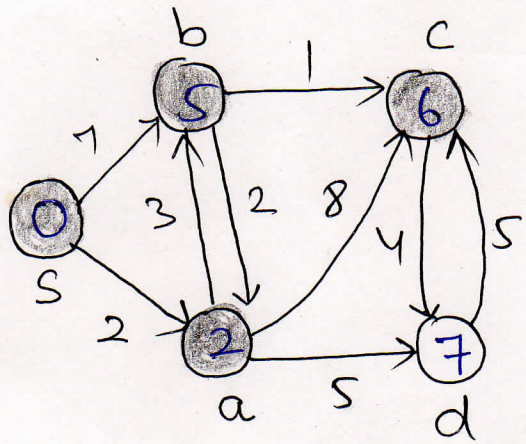


Step3

v	s	a	b	c	d
d[v]	0	2	5	6	7
pred[v]	nil	s	a	b	a
Color[v]	B	B	B	W	W

Priority Queue

v	c	d
d[v]	6	7

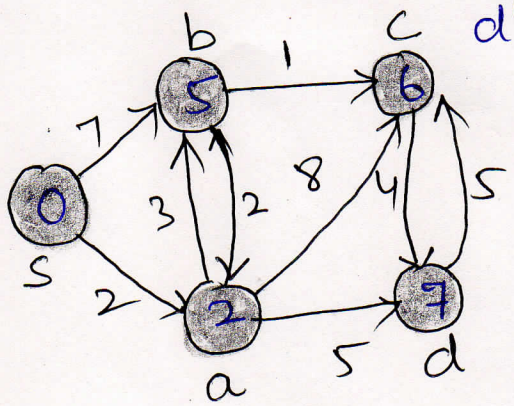


Step4

v	s	a	b	c	d
d[v]	0	2	5	6	7
pred[v]	nil	s	a	b	a
Color[v]	B	B	B	B	W

Priority Queue

v	d
d[v]	7



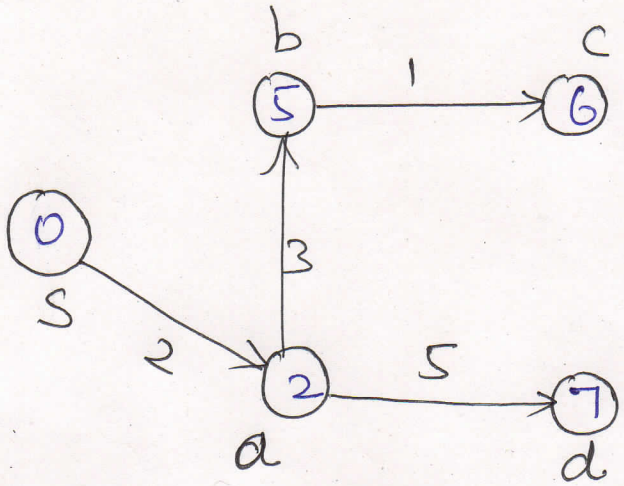
Step5

v	s	a	b	c	d
d[v]	0	2	5	6	7
pred[v]	nil	s	a	b	a
Color[v]	B	B	B	B	B

Priority Queue

$Q = \phi$

Therefore, Shortest path tree



v	s	a	b	c	d
d[v]	0	2	5	6	7
pred[v]	nil	s	a	b	a