Data Structures and Algorithms

Lab 12
Dijkstra algorithm

Today’s Objectives

- Recap
- Dijkstra’s Algorithm Example
- Dijkstra’s Algorithm Implementation
- Dijkstra’s Algorithm Applications
- Coding Exercise

Shortest Path

- Given a weighted graph G and two vertices u and v, we want to find a path of minimum total weight between u and v.

Dijkstra's Algorithm

- Finds the shortest path from a given a vertex s to every other vertex in G
- Works on the same idea as the Prim's algorithm, with a small difference
- We grow a "tree" of vertices, beginning with $\boldsymbol{s}$ and eventually covering all the vertices
- We store (in a PQ) with each vertex $\boldsymbol{v}$ a key $\boldsymbol{d}(\boldsymbol{v})$ representing the distance of $\boldsymbol{v}$ from $\boldsymbol{s}$
- At each step
  - We add to the tree the vertex $\boldsymbol{u}$ outside the tree with the smallest distance key, $\boldsymbol{d}(\boldsymbol{u})$
  - We update the keys of the vertices adjacent to $\boldsymbol{u}$

Pseudocode

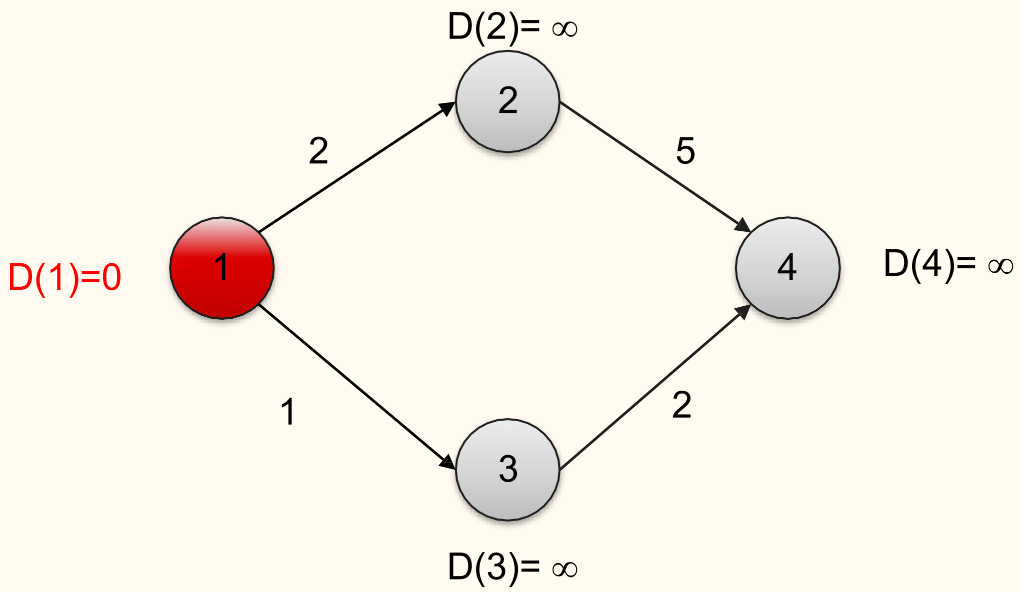
Algorithm ShortestPath(G, s):

Input: A weighted graph G with nonnegative edge weights, and a distinguished vertex s of G.

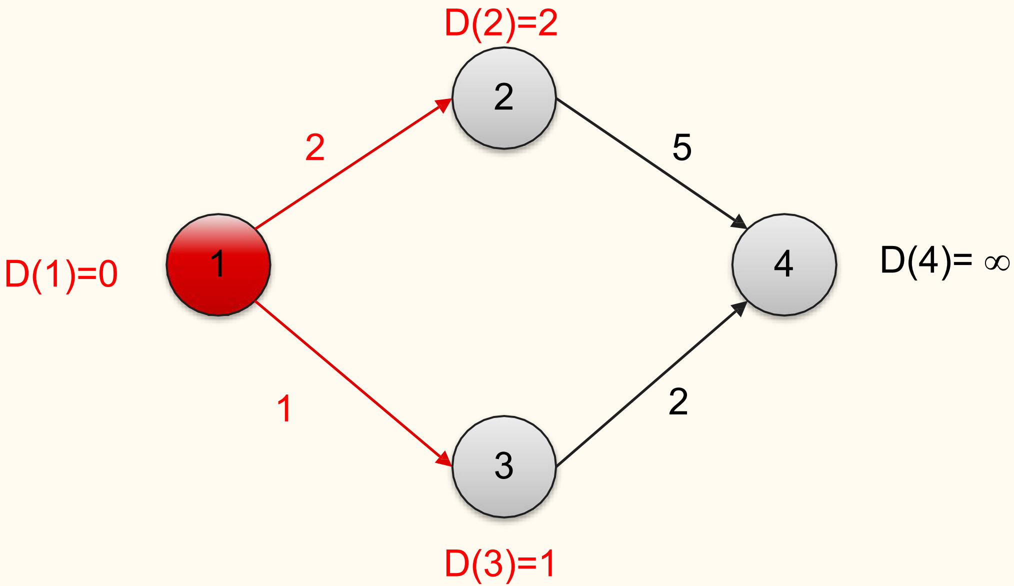
Output: The length of a shortest path from s to v for each vertex v of G.

Initialize $D[s] = 0$ and $D[v] = \infty$ for each vertex $v \neq s$.
Let a priority queue Q contain all the vertices of G using the D labels as keys.
while Q is not empty **do**
  {pull a new vertex u into the cloud}
  u = value returned by Q.remove_min()
  for each vertex v adjacent to u such that v is in Q **do**
    {perform the ***relaxation*** procedure on edge (u, v)}
    if $D[u] + w(u, v) < D[v]$ **then**
      $D[v] = D[u] + w(u, v)$
      Change to $D[v]$ the key of vertex v in Q.
return the label $D[v]$ of each vertex v

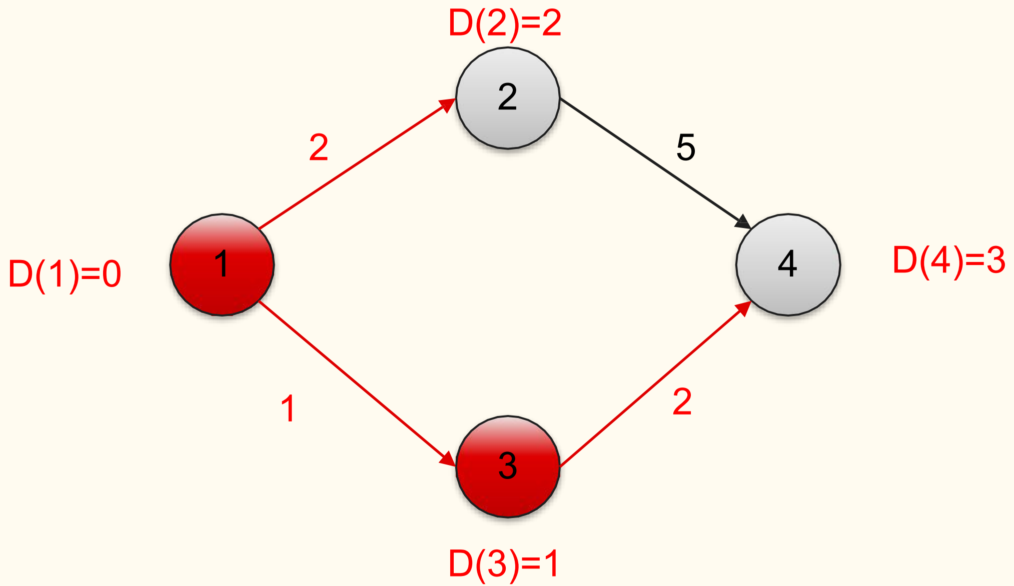
Example



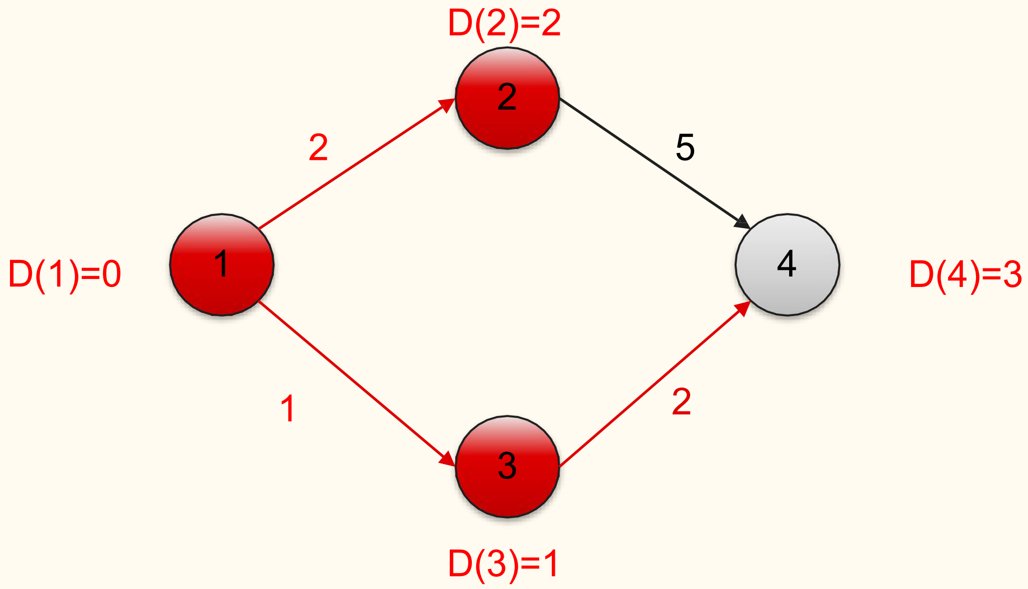
Example



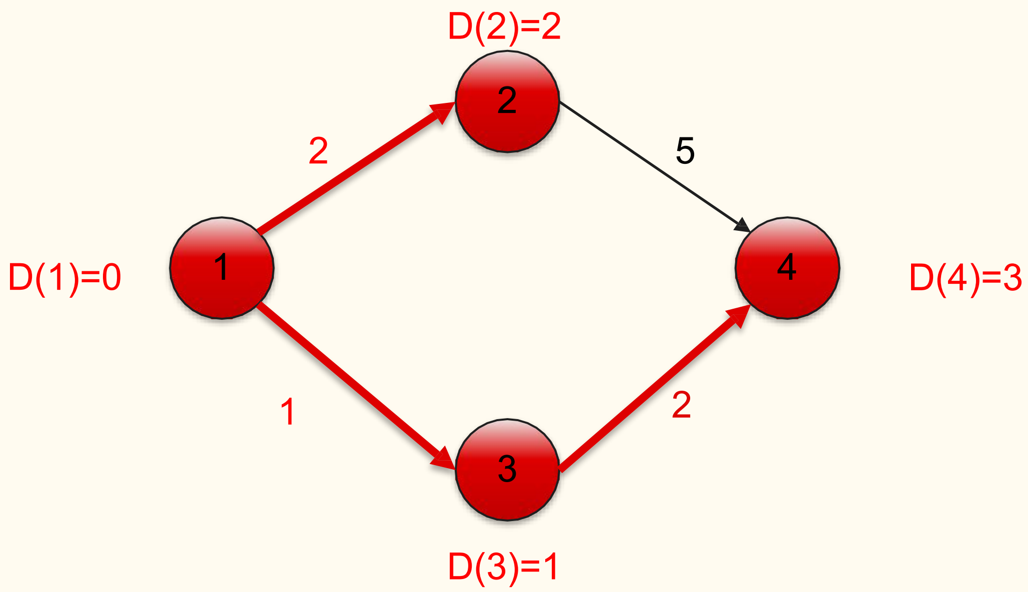
Example



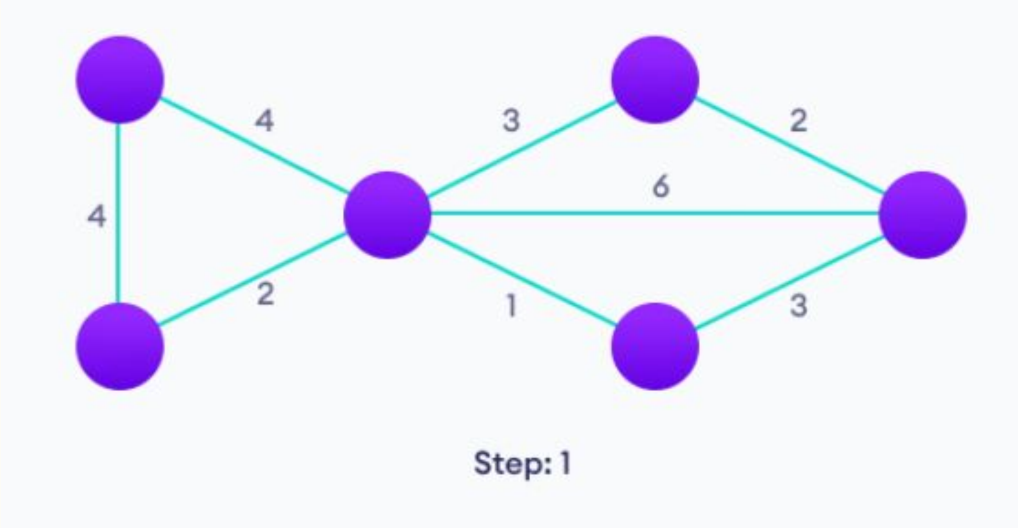
Example



Example

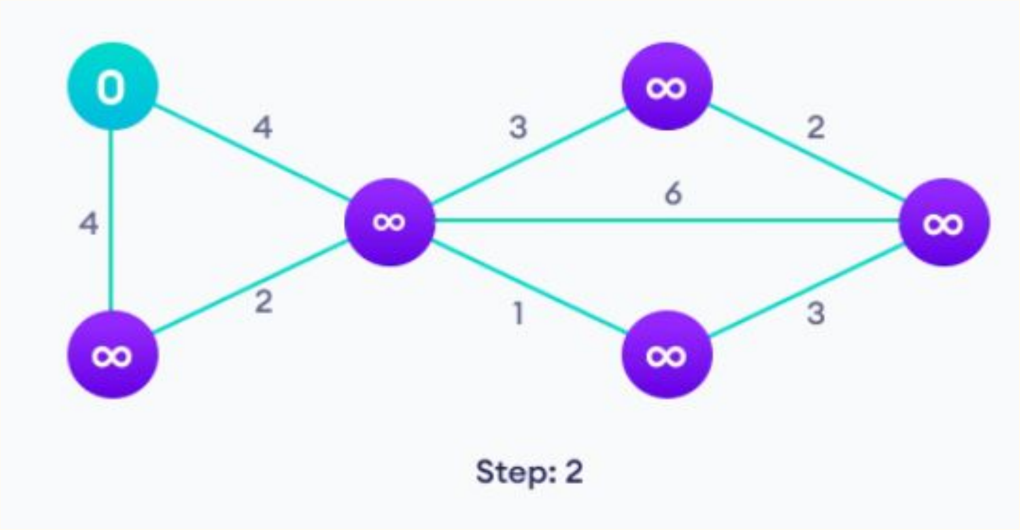


Example2



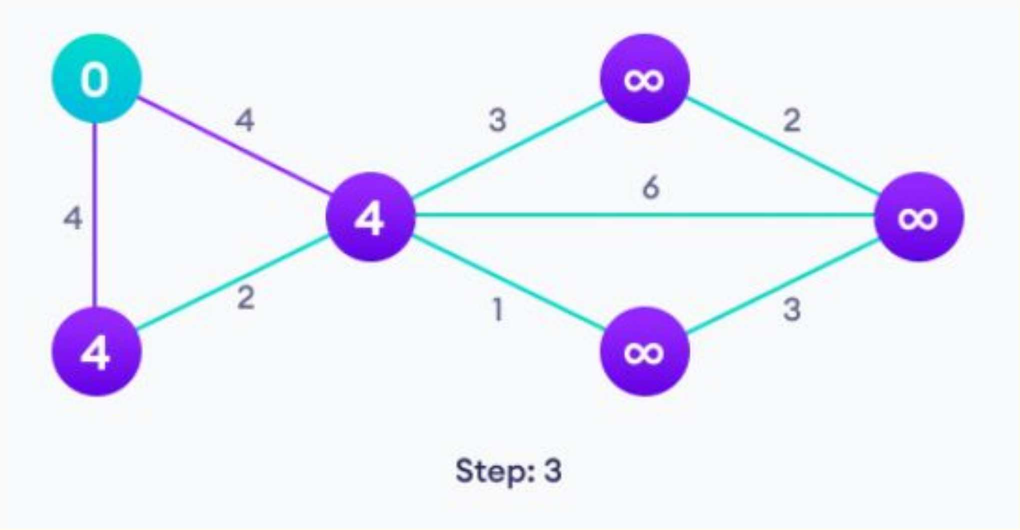
Start with a weighted graph

Example2



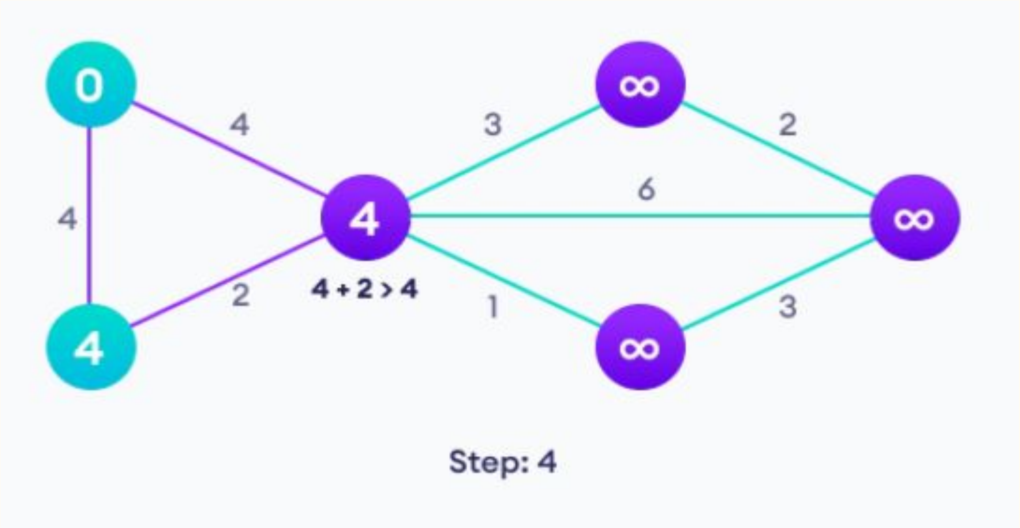
Choose a starting vertex and assign infinity path values to all other devices

Example2



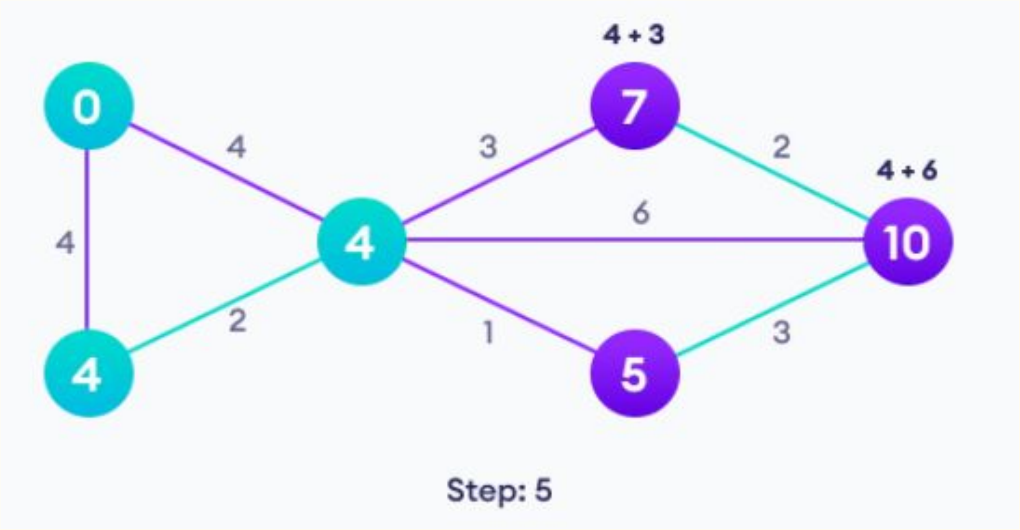
Go to each vertex and update its path length

Example2



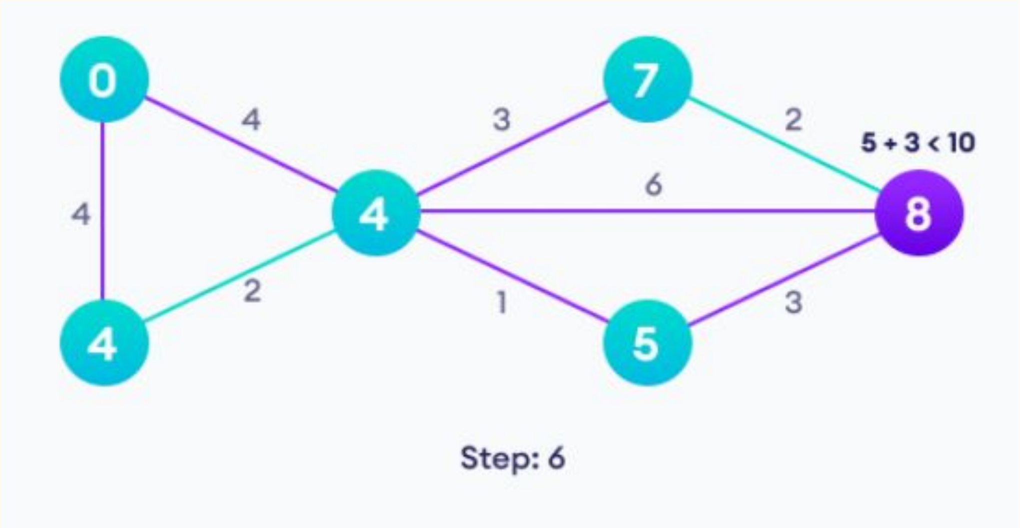
If the path length of the adjacent vertex is lesser than new path length, don't update it

Example2



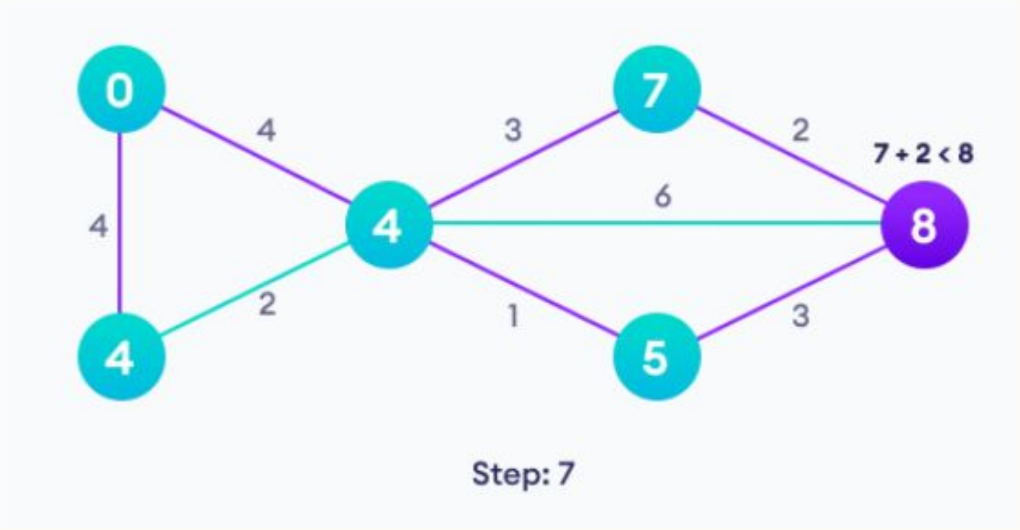
Avoid updating path lengths of already visited vertices

Example2



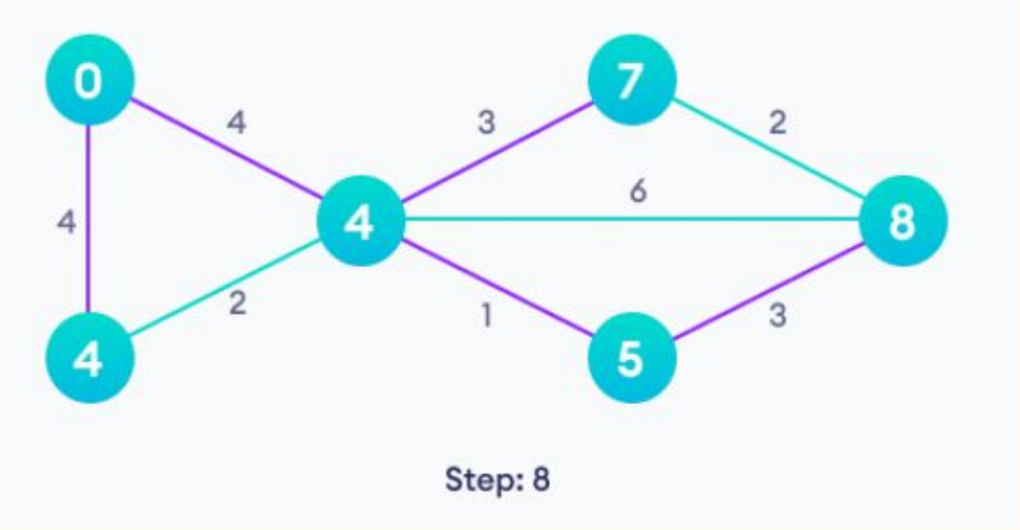
After each iteration, we pick the unvisited vertex with the least path length. So we choose 5 before 7

Example2



Notice how the rightmost vertex has its path length updated twice

Example2



Repeat until all the vertices have been visited

Dijkstra’s Algorithm Implementation

Live code

Dijkstra’s Algorithm Applications

- Internet packet routing
- Flight reservations
- Driving directions

Widest Path Problem(Telephone or computer network path)

- In networks, each line has a bandwidth, BW.
- We want to route the phone call or the packet via the highest BW.
- Finding a path between two vertices of the graph maximizing the weight of the minimum-weight edge in the path
- The vertices represents switching station or routers.
- The edges represents the transmission line, and the weight of edges represents the BW.

Widest Path Problem(Telephone or computer network path)

- The BW of a path or route is:

  $BW(P) = \min\{BW(e) \mid \text{edge } e, \text{ of } P\}$ (the weakest link)

- We want the highest BW so we use a max priority queue .

- For the relaxation, we use:

  If $\min(BW(u, z), BW(u)) > BW(z)$ then

  $BW(z) = \min(BW(u, z), BW(u))$

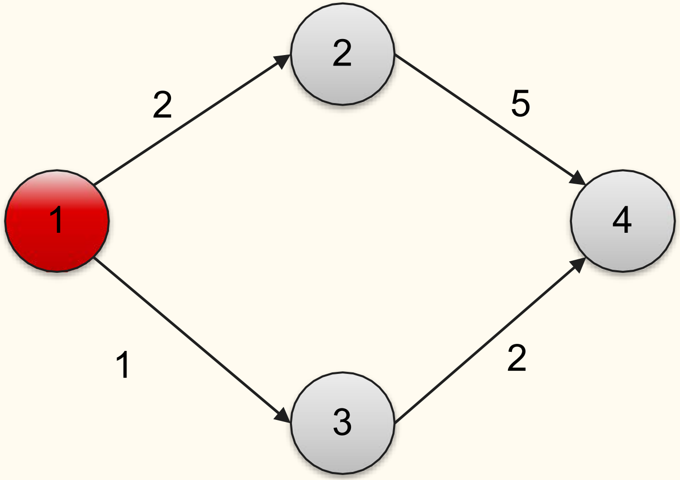
- Initialize the BW by
  - $BW(v) = \inf$, v the start vertex.
  - $BW(u) = 0$ for $u \neq v$

Widest Path Problem(Telephone or computer network path)

Algorithm: HighestBW(G, v)

Initialize $BW(v) = \infty$ and $BW(u) = 0$ for u != v
Initialize priority queue Q of vertices using BW as key.
while Q is not empty **do**
$u = Q$.removeMax()
for each vertex z adjacent to u **and** in Q **do**
  if min($BW(u, z)$, $BW(u)$) > $BW(z)$ **then**
    $BW(z)$ = min($BW(u, z)$, $BW(u)$)
    update z in Q
return BW

Widest Path Problem(Telephone or computer network path)



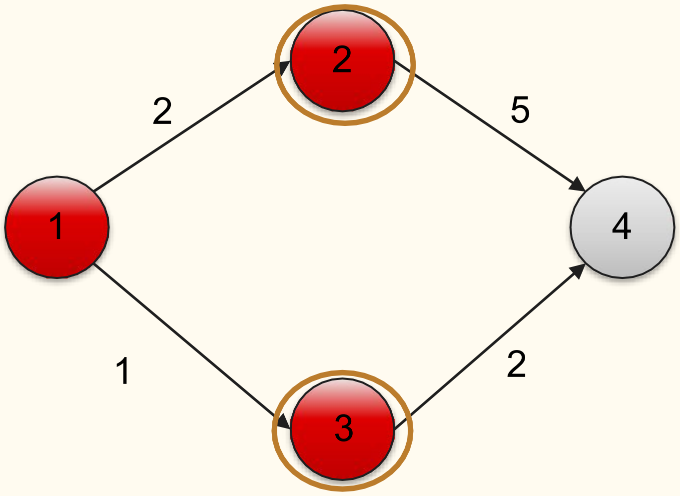
PQ

(0,4)
(0,3)
(0,2)
(+∞,1)

BW array	+∞	**0**	**0**	**0**

We will start from source vertex and then travel all the vertex connected to it and add in priority queue according to relaxation condition.

Widest Path Problem(Telephone or computer network path)



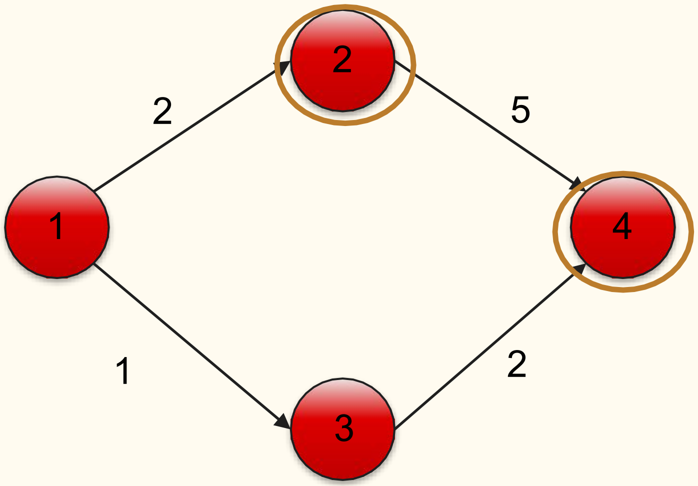
BW array

+∞	**2**	**1**	**0**

PQ

(0,4)
(1,3)
(2,2)

Widest Path Problem(Telephone or computer network path)



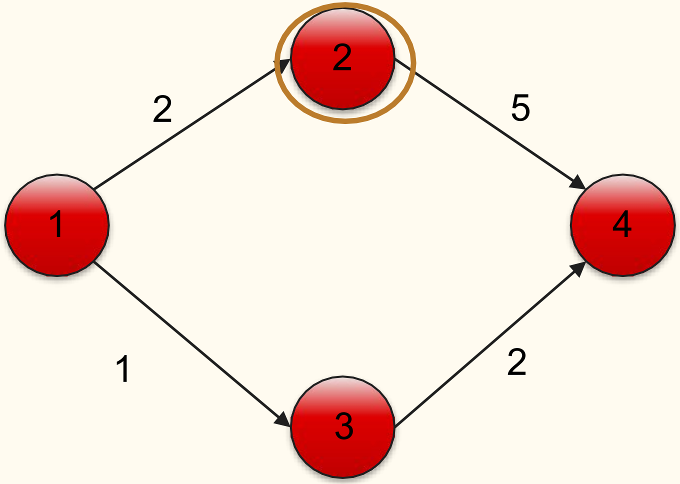
BW array

$+\infty$	**2**	**1**	**2**

PQ

(1,3)
(2,4)

Widest Path Problem(Telephone or computer network path)



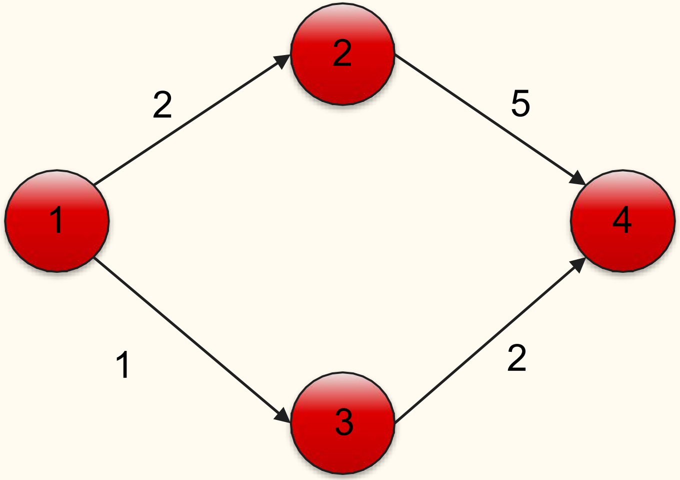
PQ

(1,3)

BW array

$+\infty$	**2**	**1**	**2**

Widest Path Problem(Telephone or computer network path)



PQ

BW array	$+\infty$	**2**	**1**	**2**

Coding Exercise

https://leetcode.com/problems/network-delay-time/

See You next week!