ITCS 6114 Algorithms and Data Structures
Programming Project 3: Design of Demonstration Algorithms

Nevetha Balasubramanian (800899379)
Raghunath Vishwanath Nallapeta (800909795)
Sriharish Ranganathan (800901667)

Dijkstra's Shortest Path

To find the shortest path between points, the weight or length of a path is calculated as the sum of the weights of the edges in the path. A path is a shortest path is there is no path from x to y with lower weight.

Dijkstra's algorithm finds the shortest path from x to y in order of increasing distance from x. That is, it chooses the first minimum edge, stores this value and adds the next minimum value from the next edge it selects.

It starts out at one vertex and branches out by selecting certain edges that lead to new vertices. It is similar to the minimum spanning tree algorithm, in that it is "greedy", always choosing the closest edge in hopes of an optimal solution.

Working of Dijkstra's algorithm

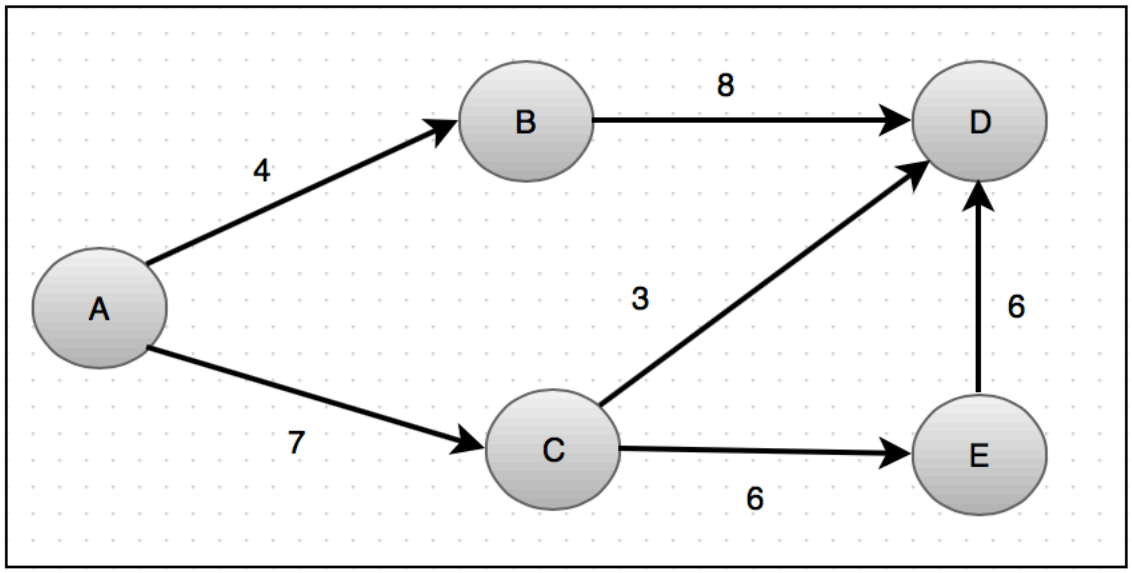
Step 0:
Temporarily assign C(A) = 0 and C(x) = infinity for all other x.
C(A) means the Cost of A
C(x) means the current cost of getting to node x

Step 1:
Find the node x with the smallest temporary value of c(x).
If there are no temporary nodes or if c(x) = infinity, then stop.
Node x is now labeled as permanent. Node x is now labeled as the current node. C(x) and parent of x will not change again.

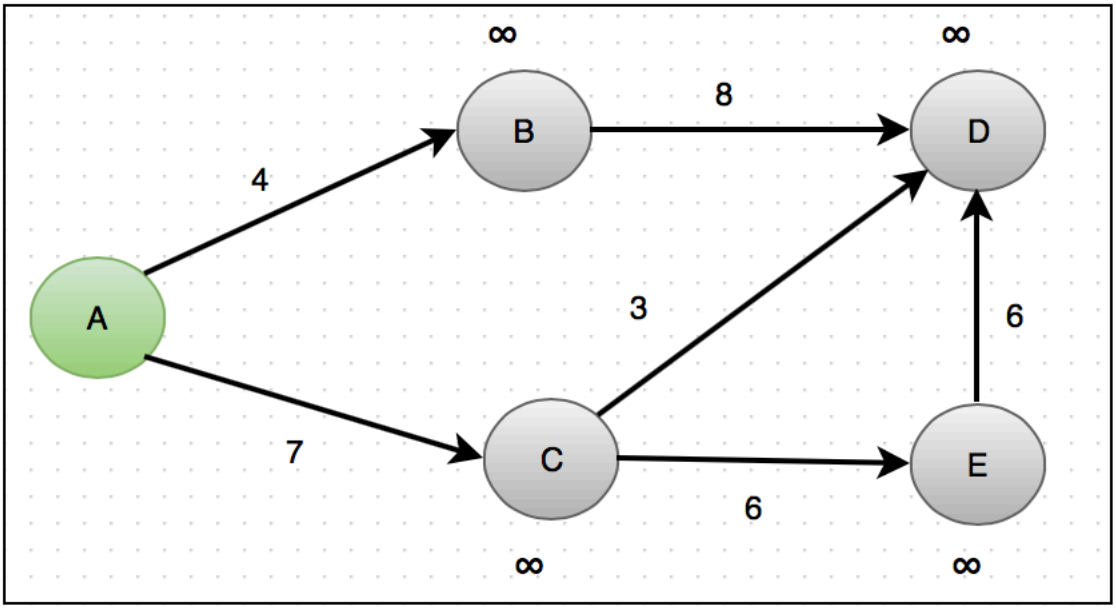
Step 2:
For each temporary node labeled vertex y adjacent to x, make the following comparison:
if c(x) + Wxy < c(y), then c(y) is changed to c(x) + Wxy
assign y to have parent x

Step 3:
Return to step 1.

Consider below graph and let's apply Dijkstra's algorithm to find shortest path. Assume each node represents a particular place and the numbers mentioned is the cost to travel between the nodes that is between the places.

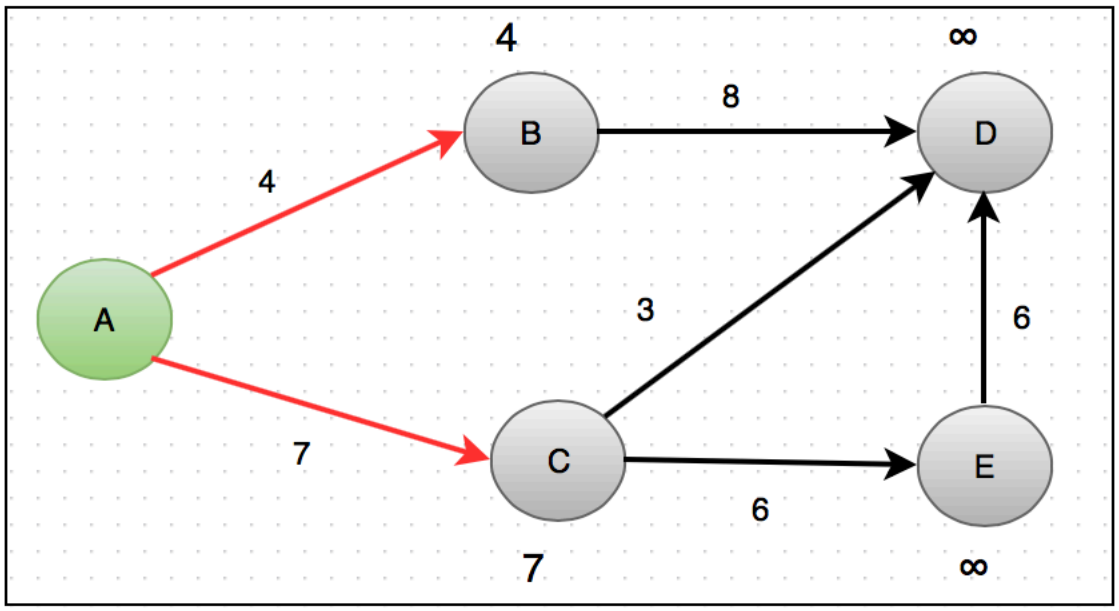


Step 1: Consider node A as source for travelling. In the first step all the cost for travelling to node from source are initialized to infinity (∞) as shown in the below diagram.

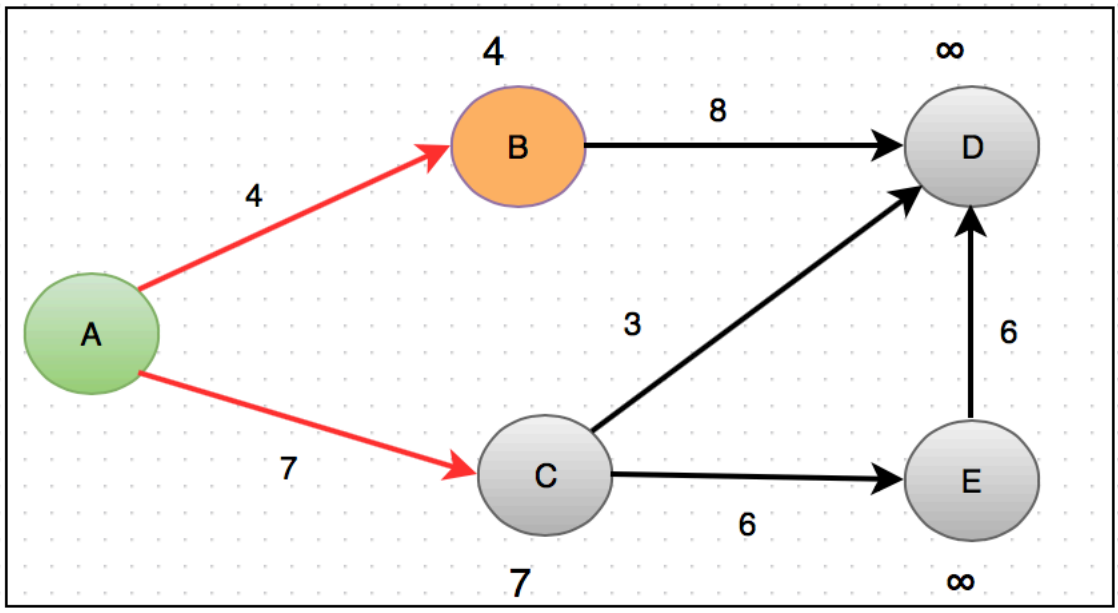


Step 2: In second step the immediate adjacent node to the source are selected and their cost of traversal from the source is calculated. In our example node B and node C are immediate adjacent nodes to source A. As mentioned in the graph the cost of traversal from node A to node B is 4 and it is greater than (∞) so the cost is

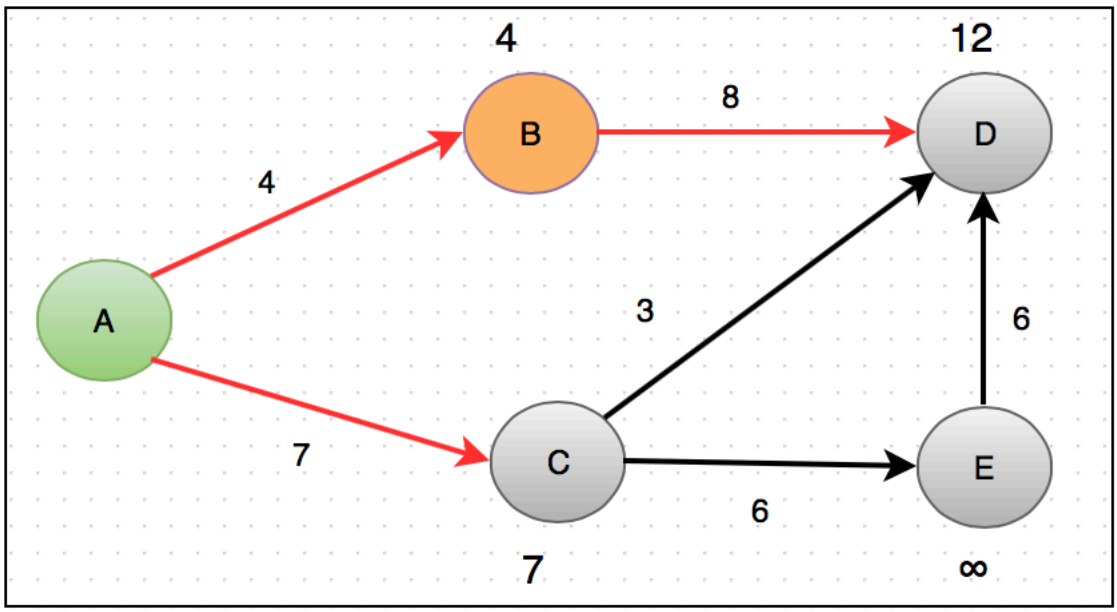
replaced to 4. This applies to node C and cost is 7. Below figure shows the cost calculated. The red arrow marks indicate that the path has been travelled.



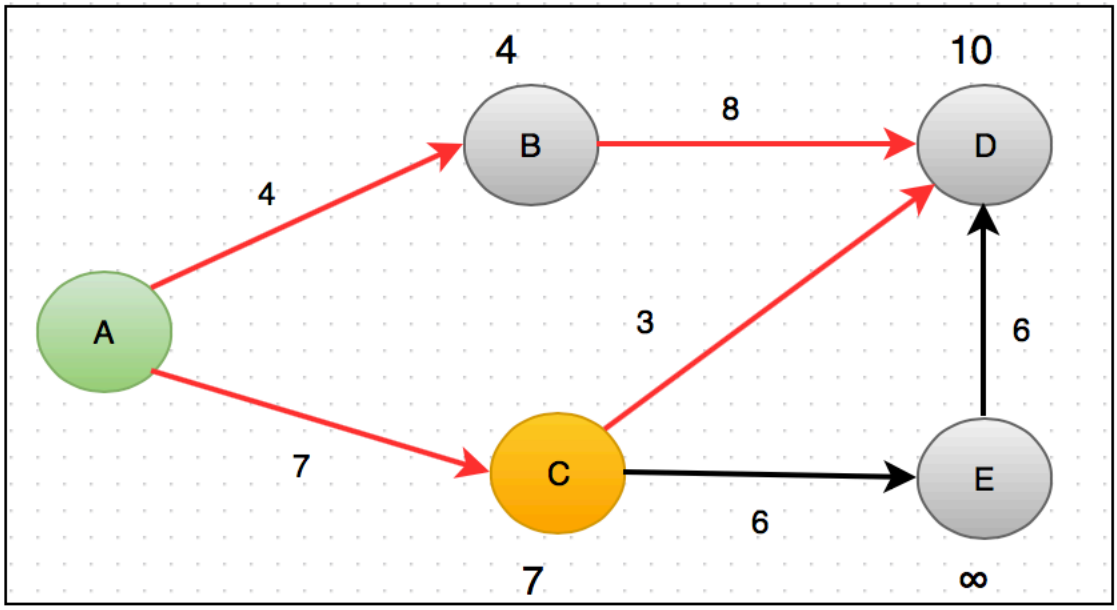
Step 3: Now the node B is selected as temporary source and further nodes cost are calculated from node B.



Now the node B is selected and distance from B to D is calculated as shown below.

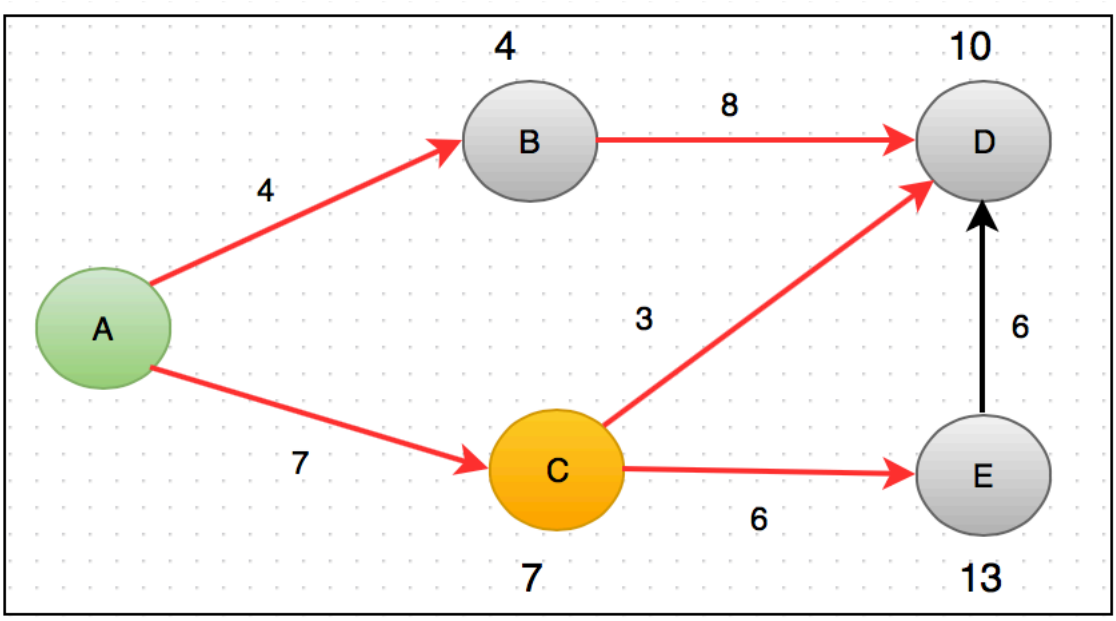


Step 4: Now the next adjacent node to source A is selected and calculated. In our case node C will be selected and further nodes distance will be calculated.

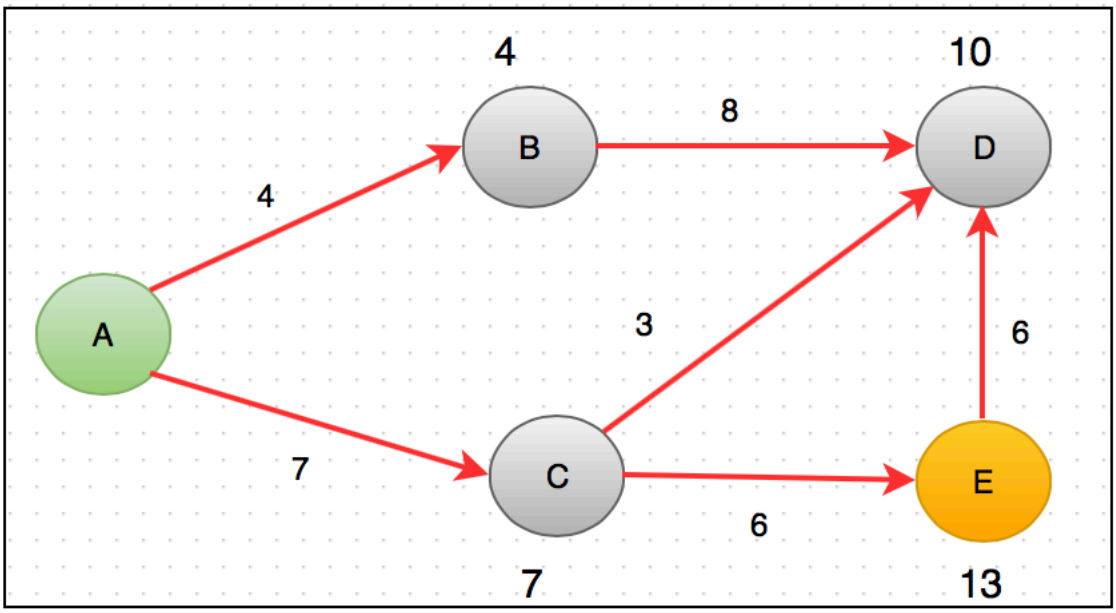


Node c has node D and node E as adjacent nodes and their distances are calculated. Cost from node C to node D is 3 and cost from A to B is 7 so total cost from A to D via C is 10 which is less than the previously calculated cost via B which was 12. Now the cost 12 will be replaced with 10 as shown in the above graph.

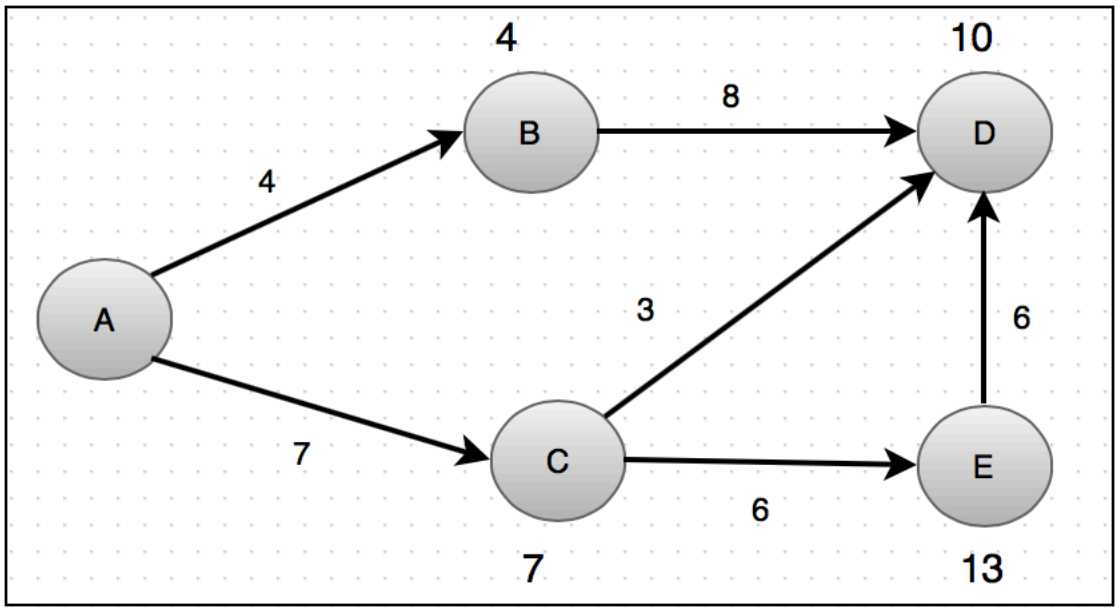
Now the node E distance has to be calculated and it is shown in the below graph.



Step 5: Further node E has adjacent node E for which distance has to be calculated. Now node E is selected and distance to D from E is calculated which is 13+6 = 19 which is greater than the current cost 10 so no changes are made to the cost of D.



Now all the paths in the graph are red in color which means all the paths have been travelled to calculate the cost. Final graph will be as below.



Now all the shortest distances between the nodes have been calculated.

How does your demonstration algorithm improve the understanding of the algorithm?

The demonstration algorithm which we implemented helps to understand the Dijkstra's algorithm. Most of the implementations only show the final cost between source and the nodes but we have designed to display the cost between each node to node which have a direct path between the nodes and also if the distance has to be calculated between the nodes which don't have a direct path our program shows the path that is the nodes to be covered to reach the destination node with cost.

Algorithm Analysis:

The time complexity of the algorithm is O(V^2) as there are two nested while loops. The inner loop statements are executed O(V+E) times The inner loop has operation which takes O(LogV) time. So overall time complexity is O(E+V)*O(LogV) which is O((E+V)*LogV) = O(ELogV) where E is the number of edges and V is the vertices.

Example with output of program:

Consider the below input file

```
Bangkok India 1.2
Bangkok USA 0.5
Denmark Bangkok 0.6
Bangkok Canada 0.25
Canada India 1.1
India Denmark 1.6
USA Canada 0.7
USA England 0.45
Canada England 1.3
Denmark England 0.3
Canada Denmark 0.67
```

Let's run our program for this input file and get the output file with the paths. Below is the screenshot of the output file.

```
Bangkok
   Canada  0.25
   Denmark  0.6
   India  1.2
   USA  0.5
Canada
   Bangkok  0.25
   Denmark  0.67
   England  1.3
   India  1.1
   USA  0.7
Denmark
   Bangkok  0.6
   Canada  0.67
   England  0.3
   India  1.6
England
   Canada  1.3
   Denmark  0.3
   USA  0.45
India
   Bangkok  1.2
   Canada  1.1
   Denmark  1.6
USA
   Bangkok  0.5
   Canada  0.7
   England  0.45
```

Distance between the nodes which has direct path

```
Please select..
1. for finding shortest path..
2. for printing graph..
Enter exit to quit the program..
1

Please enter vertices name in same case as in the input file. Space between source and destination
India Canada
India Canada 1.1
```

Distance between the nodes which doesn’t have direct path

```
Please select..
1. for finding shortest path..
2. for printing graph..
Enter exit to quit the program..
1

Please enter vertices name in same case as in the input file. Space between source and destination
USA India
USA Bangkok India 1.7
```