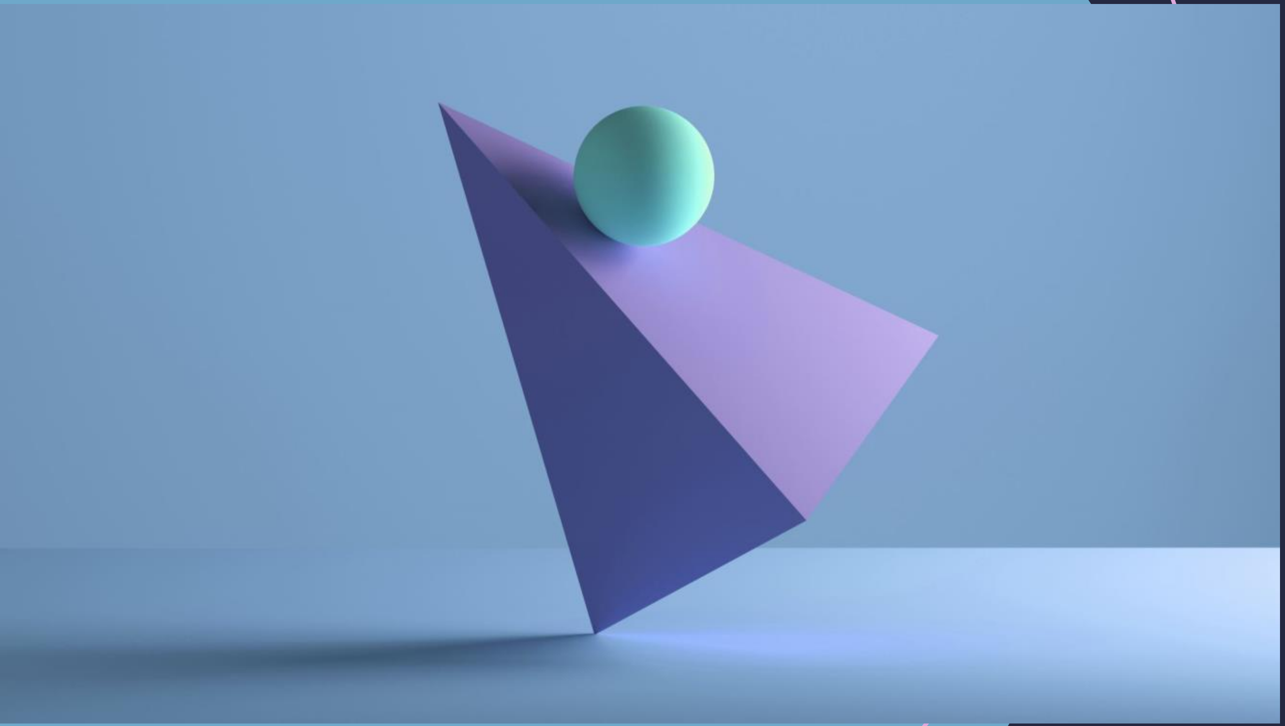

# LABEL-SETTING AND LABEL-CORRECTING ALGORITHMS

**GOLKONDA RAMANANJALI MOUNICA**

(AM.EN.P2ARI20028)

**NARENDRA SINGH BISHT**

(AM.EN.P2ARI20043)

**URVI SHARMA**

(AM.EN.P2ARI20022)

# Label-Setting Algorithms

In label-setting algorithms, the final optimal shortest path distance from the source node to the destination node is determined once the destination node is scanned and permanently labeled.

# Label-Correcting Algorithms

A label-correcting algorithm treats the shortest path distance estimates of all nodes as temporary and converges to the final one-to-all optimal shortest path distances at its final step when the shortest paths from the source node to all other nodes are determined.

# A Common Misconception

It has been a general belief that label-setting algorithms are better choices than label-correcting algorithms for computing the shortest path distance between a source node and a destination node (one-to-one) on a network.

The reason is that iterations in label-setting algorithms can be terminated whenever the destination node is scanned and permanently labeled.

In a study conducted by Zhan and Noon [1998], a fast-performing implementation of each type of algorithms was selected and compared for computing one-to-all shortest paths using 10 large real road networks.

It was concluded that, in some situations, label-correcting algorithms are a better choice for computing one-to-all shortest paths on real road networks.

**Thus, contradicting the general belief!!**

Image source – Google Images

# Network

A network consisting of a set of nodes N and a set of arcs A is represented as a directed graph G = (N, A), with

- N denoting the number of nodes
- A denoting the number of directed arcs

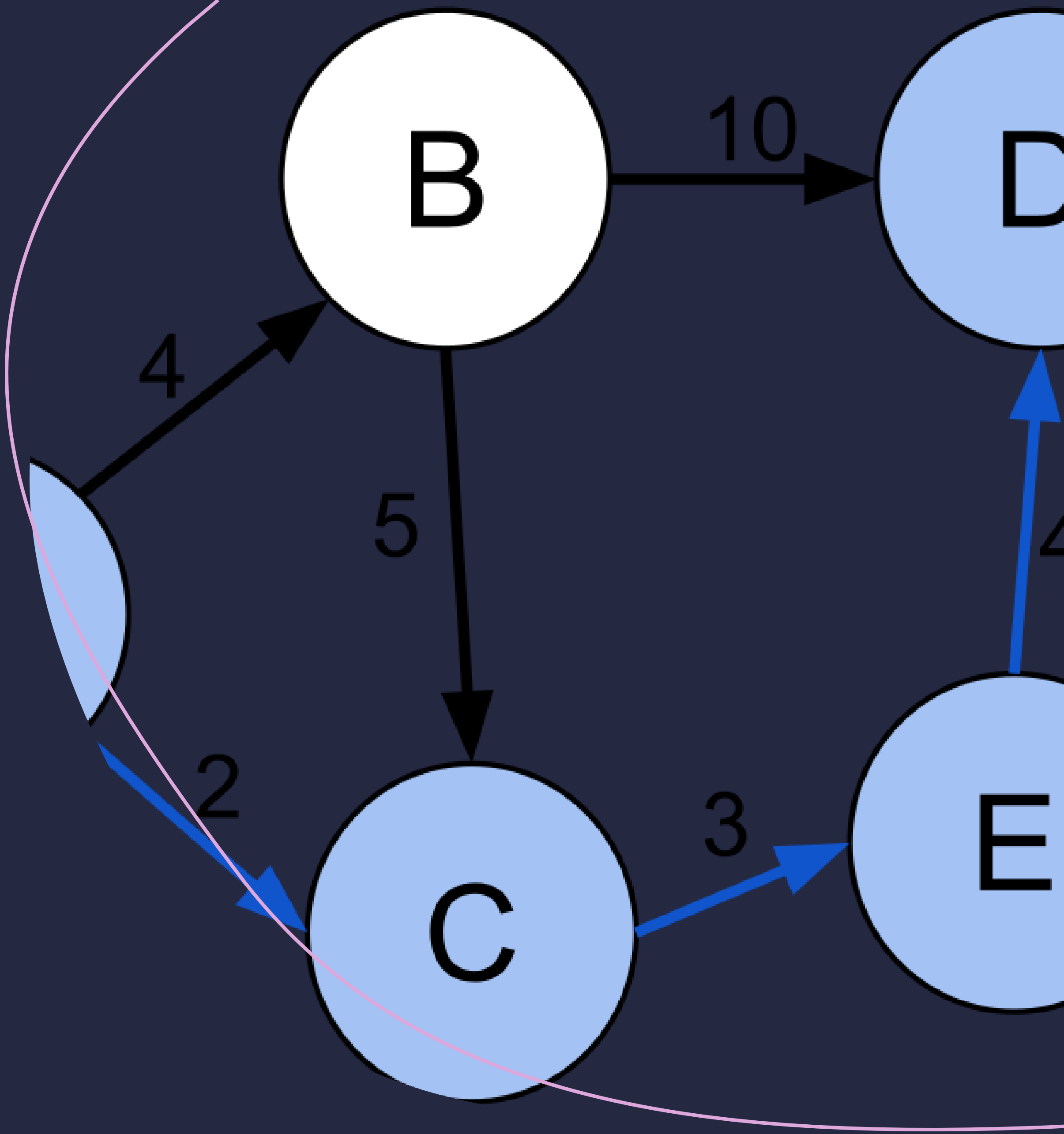


Image source - Wikipedia

# An Arc in a Network

- Each arc (i, j) is represented as an ordered pair of nodes traveling from node "i" to node "j".
- Each arc (i, j) has an associated numerical value, $d_{ij}$, representing the distance or cost incurred by traversing the arc.

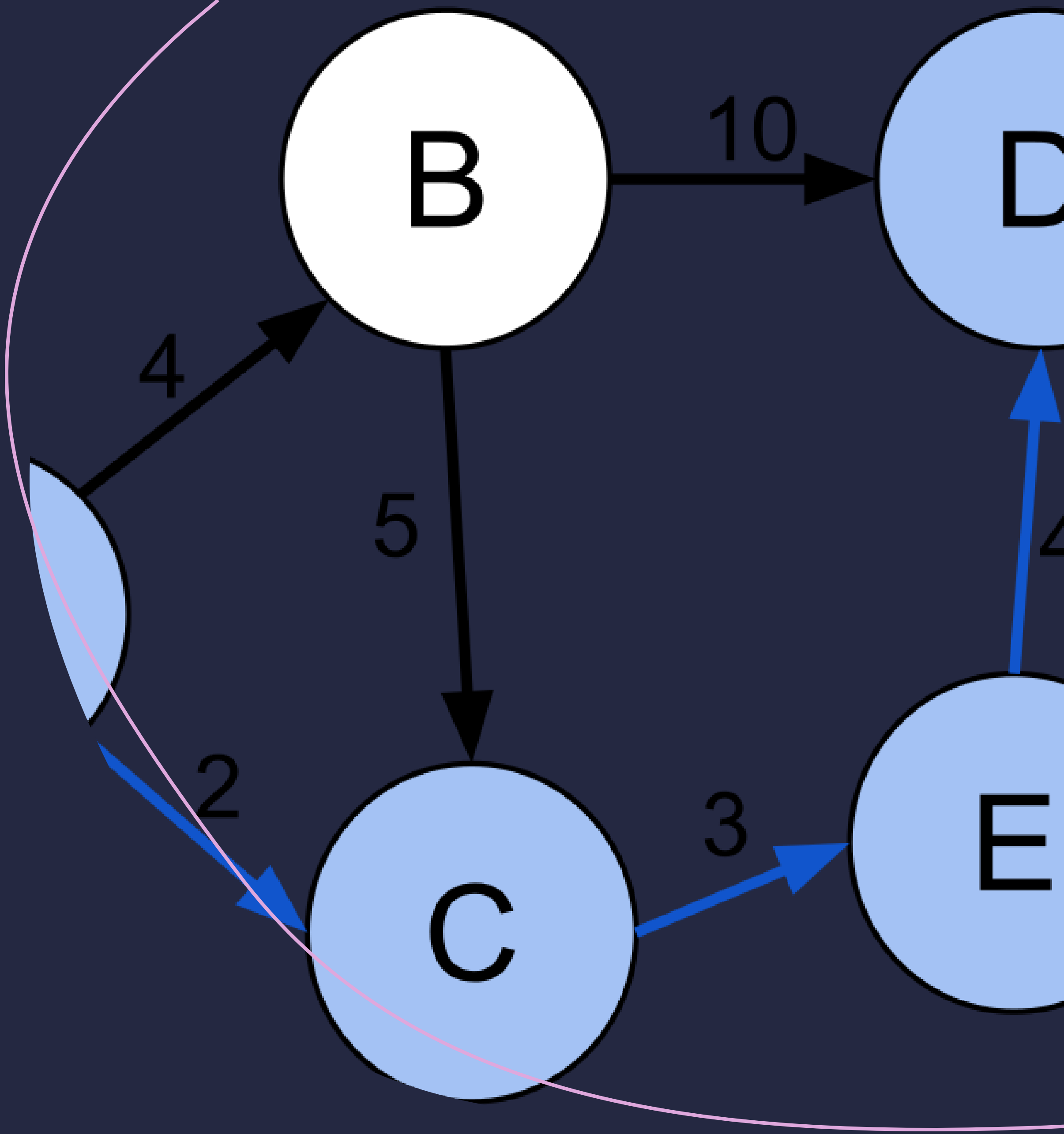


Image source - Wikipedia

# Shortest Path Tree

The one-to-all shortest paths are normally represented as a directed out-tree rooted at the source nodes.

This directed tree is referred to as a shortest path tree.

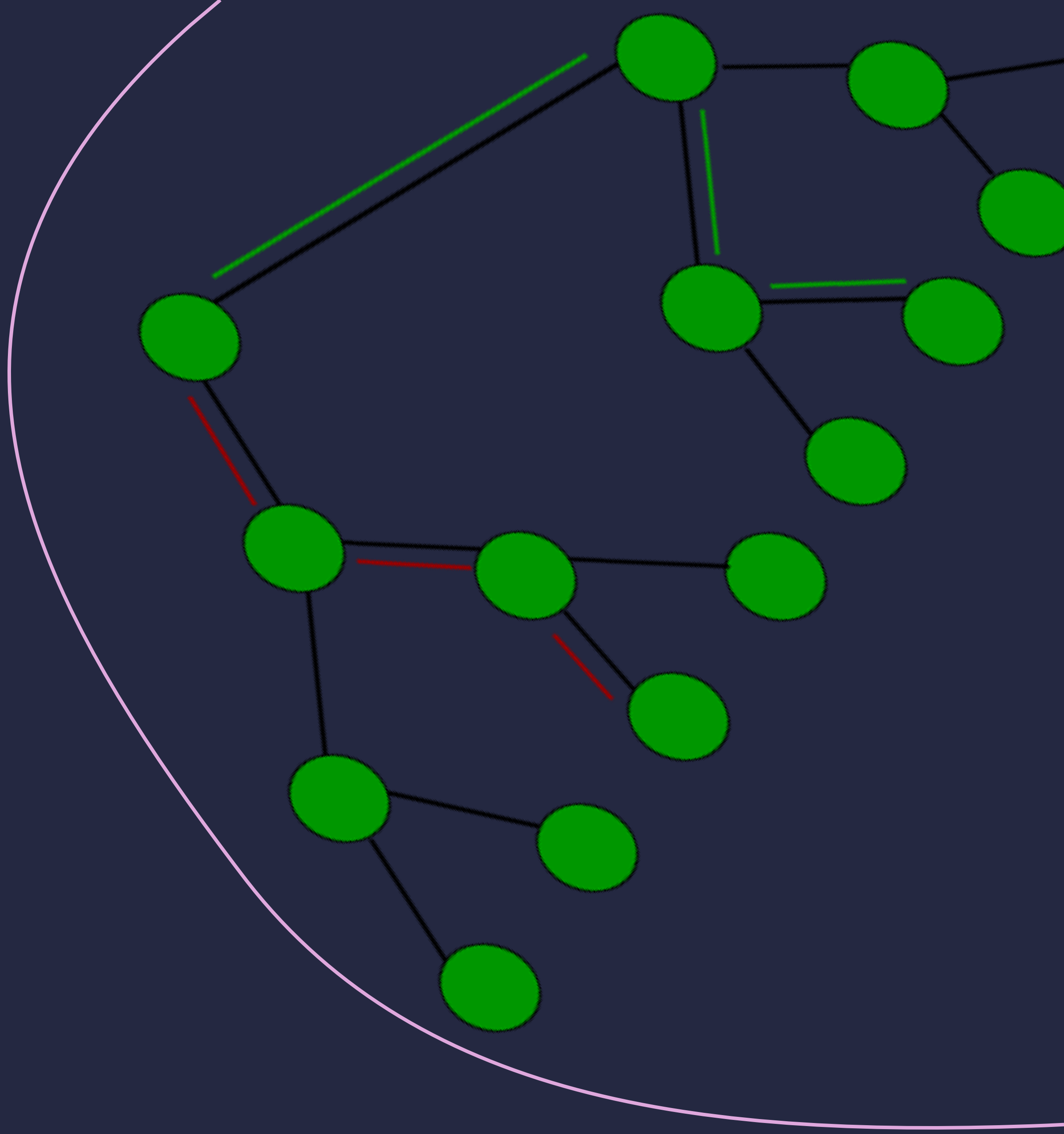

Image source – GeekforGeeks

# Node Status

- Unreached: If its distance label has a value of infinite (sufficiently large).
- Labeled: If its distance label has been updated at least once, that is, its distance label is different from infinite.
- Permanently Labeled: If its distance label represents the final and optimal shortest path from the source node.
- Scanned: If it has undergone a scanning operation and its distance label has subsequently not been updated.

# Labeling Method

- Central to all shortest path algorithms is the labeling Method.
- Both the label-setting algorithms and label-correcting algorithms iteratively employ the labeling method to construct a shortest path tree.
- In the labelling method, an out tree from source node "s" is constructed and improved until no further improvements can be made.
- At termination, the shortest path from “s” to node “i” is represented by a unique path from “s” to “i” on the out-tree.

# A Node in an Out-tree

While constructing and improving the out-tree, three pieces of information are maintained for each node "i" in the labeling method:

- a distance label d (i)
- a parent node p (i)
- a node status S (i) ∈ {unreached, labeled, scanned}

# Terminology

- Distance Label: As the labeling method progresses, the distance label d(i) is used to record the upper bound of the s-to-i path distance in the out-tree.
- Parent Node: The parent node p(i) records the node that immediately precedes node "i" in the out-tree.

# Labelling Method - Procedure

- Begins by setting d (i) = $\infty$, p (i) = $\emptyset$, and S (i) = unreached for every node i.
- Upon initialization, information associated with a source node s are set as d (s) = 0 and S (s) = labeled.
- Progresses by applying a scanning operation to labeled nodes.
- Terminates with a shortest path tree over either a subset of nodes (DIKBA) or the entire set of nodes (TWO-Q).

# Labelling Method - Pseudo Code

```
for all (i, j ) ∈ A do
        begin
        if d (i) + d ij < d (j) then
            begin
            D (j) = d (i) + d ij
            P (j) = i
            S (j) = labeled
            end
        end
set S (i) = scanned
```

# Types of Shortest Paths

Depending on the nature of the problem, it is sometimes necessary to compute shortest paths in following ways :

- One-to-All: From a source node to every other node.
- All-to-All: From every node to every other node on a network.
- One-to-One: From a source node to a destination node.
- One-to-Some: From a source node to some destination nodes on a network.

# Label Setting vs Label Correcting

Label-setting and label-correcting algorithms are based on the labeling method and employ the scanning operation as described.

The two algorithms differ primarily in the data structures used for

- Managing the set of labeled nodes
- Selecting nodes for scanning

# A Dequeue Algorithm

- The nodes are stored in either stack or queue depending on whether they are already seen in the data structure or not.
- Both BFS and DFS strategies are used.
- Time complexity is O $(n * 2^n)$ ,

  which is huge!

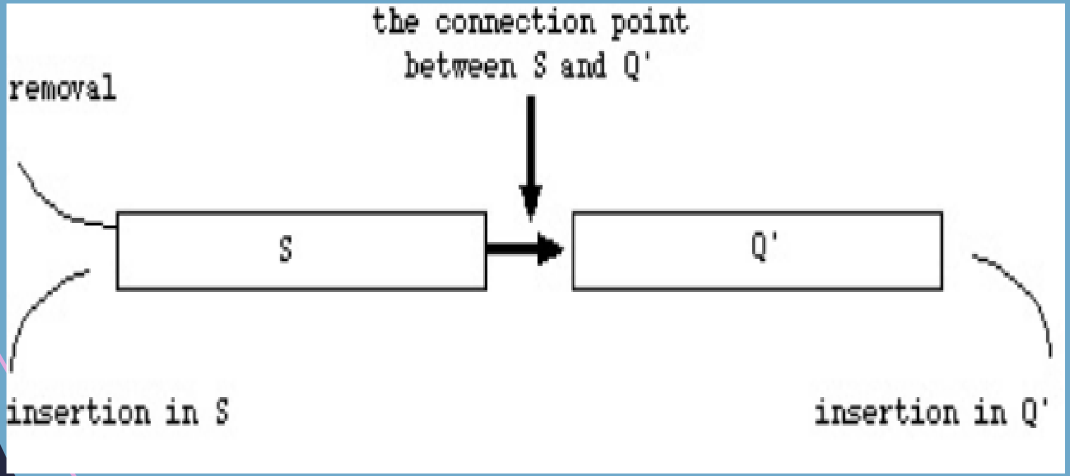


Image Source: Three Fastest Shortest Path Algorithms on Real Road Networks: Data Structures and Procedures : JGIDA vol.1, no.1, pp. 69-82

# Pallottino's Graph Growth Algorithm - (TWO-Q)

- Same as that of Dequeue, except two queue are used.
- By definition, a node becomes labeled if its distance label is changed during a scanning operation.
- Time complexity is O $(n^2 m)$ , reduced in comparison.
- Number of scans are more, but the effort for scanning selection is minimal.

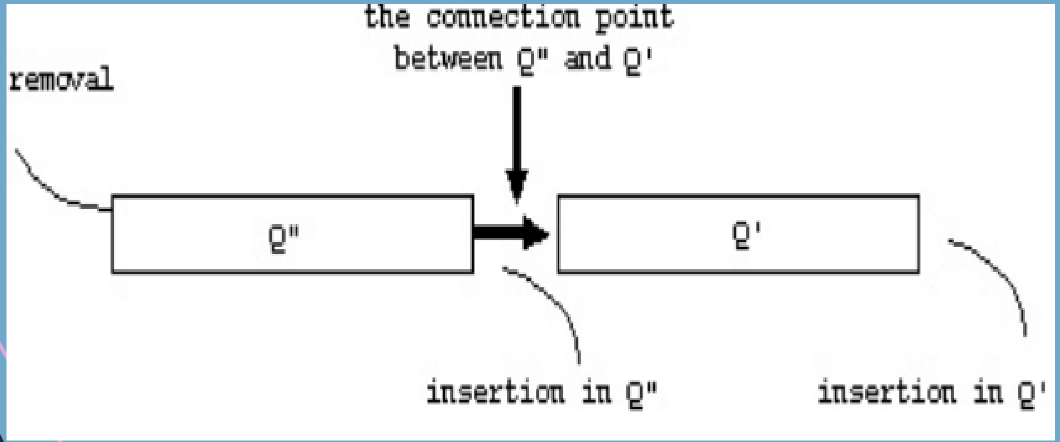


Image Source: Three Fastest Shortest Path Algorithms on Real Road Networks: Data Structures and Procedures : JGIDA vol.1, no.1, pp. 69-82

# Dijkstra's Algorithm Original Implementation

In the original Dijkstra algorithm, a labeled node with the minimum distance label is selected for scanning.

The set of labeled nodes is managed as an unordered list, which is clearly a bottleneck since all labeled nodes must be checked during an iteration in order to select the node with the minimum distance label.

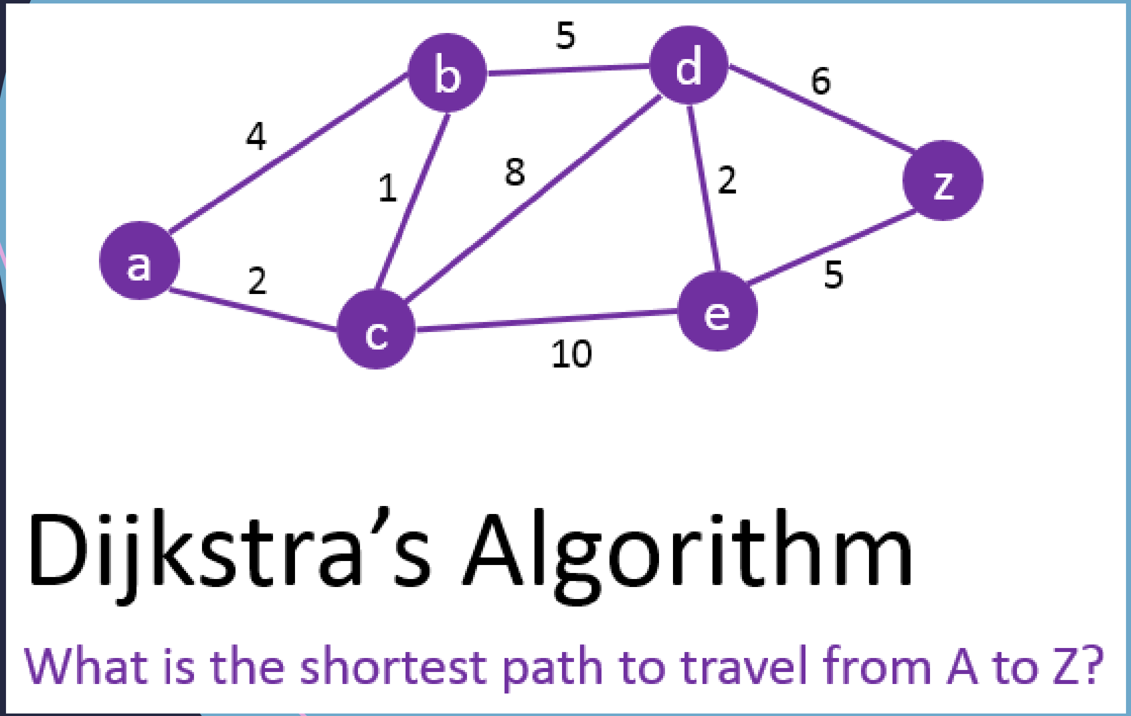


Image Source - www.101computing.net

# continued …

A natural enhancement of the original Dijkstra algorithm is to maintain the labeled nodes in a data structure such that the nodes are either exactly or approximately sorted according to distance label.

One such data structure is the bucket.

# Dijkstra's Algorithm Approximate Bucket Implementation

In the approximate bucket implementation of the Dijkstra algorithm (DIKBA), a bucket "i" contains those labeled nodes whose distance labels are within the range of

$[\, i * \beta, (i+1) * \beta - 1],$

where β is a chosen constant.

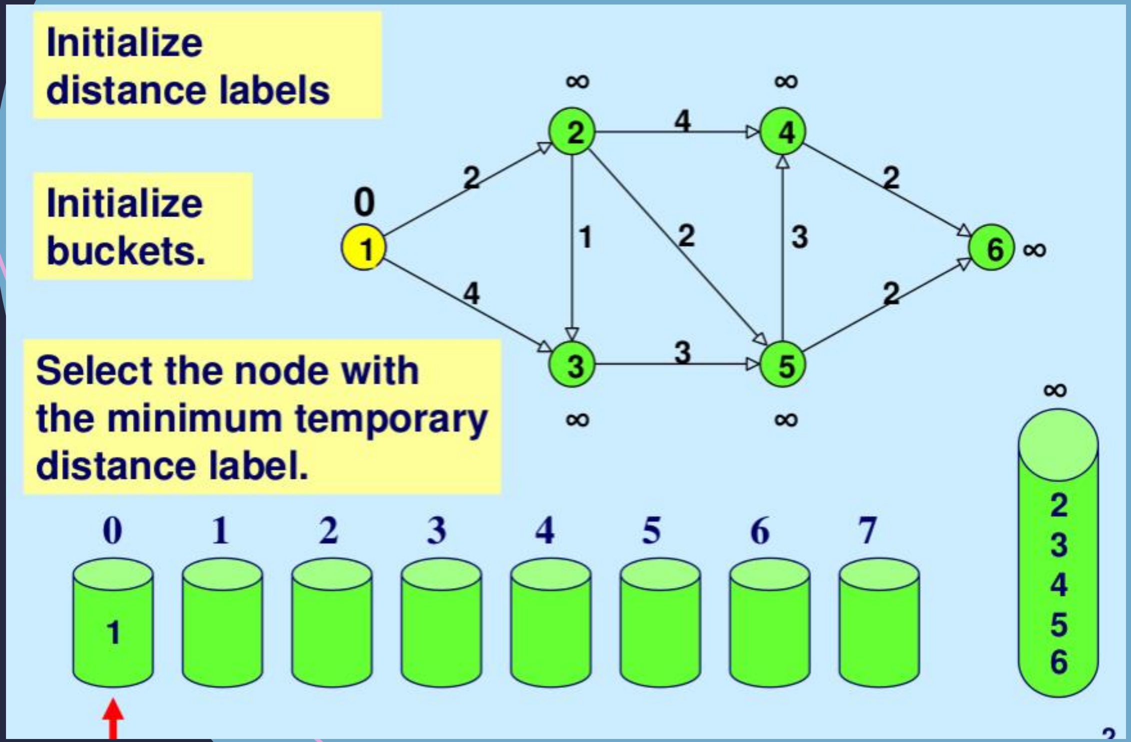


Image Source: www.ocw.mit.edu

# continued ...

- Nodes within each bucket are maintained in a FIFO ordered list.
- A node can be scanned more than once.
- When all nodes within a current bucket have been completely scanned out, their distance labels become permanent.

# Zhan and Noon's Study [1998]

It was concluded that

- Dijkstra’s algorithm implemented with approximate buckets (DIKBA) is the best-performing label-setting algorithm.
- Pallottino’s graph growth algorithm implemented with two queues (TWO-Q) is the best-performing label-correcting algorithm.

Additionally, DIKBA was observed to be over 50% slower than TWO-Q for computing one-to-all shortest paths on real road networks.

# continued …

The conclusion led to several interesting questions with respect to the performance of algorithms DIKBA and TWO-Q in computing one-to-one shortest paths.

These questions were both explored and answered in another study by Zhan and Noon.

# Zhan and Noon's Study [2000] Terminology

(Computation of one-to-one shortest paths -  DIKBA vs TWO-Q)

- Lower End Threshold Distance: The distance from the source node within which algorithm DIKBA is highly likely to be faster.
- Middle Threshold Distance: The distance from the source node at which the speed of the two algorithms DIKBA and TWO-Q is about equal.
- Upper End Threshold Distance: The distance from the source node beyond which algorithm TWO-Q is highly likely to be faster.

# Zhan and Noon's Study [2000] Observations

- For a set of randomly chosen pairs of source and destination nodes, algorithm TWO-Q was generally about 14% faster than DIKBA on average in computing a one-to-one shortest path.
- The lower end, middle and upper end threshold ratios were found to be 0.20, 0.40 and 0.70, respectively.
- The ratios suggested that distance range from a source node to a destination node, within which DIKBA was faster than TWO-Q, was less than the distance range within which TWO-Q was faster than DIKBA on any given network.

# continued ...

- Clearly, when the shortest path distances of all destination nodes have a threshold ratio of 0.40 or less, DIKBA would likely be faster than TWO-Q, on average.
- However, when at least one of the destination nodes falls beyond a threshold ratio of 0.40, TWO-Q would likely be faster.

Overall, there was no clear advantage to use DIKBA instead of TWO-Q for computing one-to-one shortest paths.

# Zhan and Noon's Study [2000] Conclusion

The TWO-Q algorithm is generally a better choice if one faces the task of computing one-to-one or one-to-some shortest path distances.

This was because middle threshold ratio of 0.40 favoured algorithm TWO-Q when the distribution nature of network nodes in a two-dimensional space was taken into consideration.

# Zhan and Noon's Study [2000] Recommendations

- When there is no prior knowledge about the shortest path distance between a source node and a destination node, relative to the longest shortest path distance on a shortest path tree rooted at the source node, use TWO-Q to compute one-to-one (some) shortest paths on a network.
- DIKBA may be chosen if it is known that the shortest path distance from a source node to a destination node is within 40% of the longest shortest path distance on the shortest path tree rooted at the source node.

# continued …

The choice of TWO-Q is perhaps most appropriate in situations where it is necessary to compute many relatively long routes.

Example: Setting up distribution system optimization models.

An obvious choice for DIKBA would be situations in which the paths are short relative to the extent of the network.

Example: Centralized dispatching system for emergency response.

# Questions?

# Thank you