# Course: Advanced Algorithm and Problem Solving

## WEEK 8 Shortest Path Algorithms

**Lemlem Kassa (Ph.D.)**

**Addis Ababa Science and Technology University, Ethiopia**

**May, 2025**

# WEEK 8 : Shortest Path Algorithms

## Content

- Introduction to Shortest Path Algorithms
- Single Source Shortest Path Algorithms
  - Dijkstra's Algorithm
  - Bellman-Ford Algorithm
- Bellman-Ford vs Dijkstra Algorithm

# Lecture Learning Outcome

- Understand Shortest Path Algorithms
- Understand Single Source Shortest Path Algorithms
- Understand Bellman-Ford Algorithm and Dijkstra Algorithms
- Understand applications of Bellman-Ford Algorithm and Dijkstra Algorithms and how they works

# Introduction to Shortest Path Algorithms

**What are the Shortest Path Algorithms?**

- The shortest path algorithms are the ones that focuses on calculating the minimum travelling cost from source node to destination node of a graph in optimal time and space complexities.
- There are various types of graphs (weighted, unweighted, negative, cyclic, etc.) therefore having a single algorithm that handles all of them efficiently is not possible.
- Shortest-path algorithms categorized into two :-Single Source Shortest Path Algorithms and all Pair Shortest Path Algorithms

[1]. Shortest Path Algorithm Tutorial with Problems, GeeksforGeeks, November 23, 2023.https://www.geeksforgeeks.org/shortest-path-algorithms-a-complete-guide/

# Introduction to Shortest Path Algorithms …cont'd

**Types of shortest-path algorithms**

**A) Single Source Shortest Path Algorithms**:

- In this algorithm we determine the shortest path of all the nodes in the graph with respect to a single node i.e. we have only one source node in this algorithms.
  - **Depth-First Search (DFS)**
  - **Breadth-First Search (BFS)**
  - **Dijkstra's algorithm**
  - **Bellman-Ford algorithm**
  - Etc..

[1]. Shortest Path Algorithm Tutorial with Problems, GeeksforGeeks, November 23, 2023.https://www.geeksforgeeks.org/shortest-path-algorithms-a-complete-guide/

# Introduction to Shortest Path Algorithms ..cont'd

## 1. Shortest Path Algorithm using Depth-First Search(DFS):

- DFS algorithm recursively explores the adjacent nodes until it reaches to the depth where no more valid recursive calls are possible.
- For DFS to be used as a shortest path algorithm, the graph needs to be acyclic i.e. a TREE, the reason it won't work for cyclic graphs is because due to cycles, the destination node can have multiple paths from the source node and **DFS** will not be able to choose the best path.
- Time Complexity: O(N) where N is the number of nodes in the tree.

**Algorithm**:

- Distance of source node to source node is initialized to 0.
- Start the DFS from the source node.
- As we go to a neighbouring nodes we set it's distance from source node = edge weight + distance of parent node.
- DFS ends when all the leaf nodes are visited.

[1]. Shortest Path Algorithm Tutorial with Problems, GeeksforGeeks, November 23, 2023.https://www.geeksforgeeks.org/shortest-path-algorithms-a-complete-guide/

# Introduction to Shortest Path Algorithms

**2. Breadth-First Search (BFS) for Shortest Path Algorithm:**

- BFS is a great shortest path algorithm for all graphs, the path found by breadth first search to any node is the shortest path to that node, i.e. the path that contains the smallest number of edges in unweighted graphs.
- This is due to the fact that unlike DFS, BFS visits all the neighbouring nodes before exploring a single node to its neighbour.
- As a result, all nodes with distance 'd' from the source are visited after all the nodes with distance smaller than d.
- **Time Complexity**: O(N) where N is the number of nodes in the graph

[1]. Shortest Path Algorithm Tutorial with Problems, GeeksforGeeks, November 23, 2023.https://www.geeksforgeeks.org/shortest-path-algorithms-a-complete-guide/

# Single Source Shortest Path Algorithms

## Dijkstra’s Algorithm

- The  Dijkstra’s Algorithm is a greedy algorithm that is used to find the minimum distance between a node and all other nodes in a given graph.
- Here we can consider node as a router and graph as a network. It uses weight of edge .ie, distance between the nodes to find a minimum distance route.
- The goal of Dijkstra's algorithm is to find the quickest or easiest way to get from a starting point to an ending point by checking all possible routes and choosing the one with the smallest total cost.

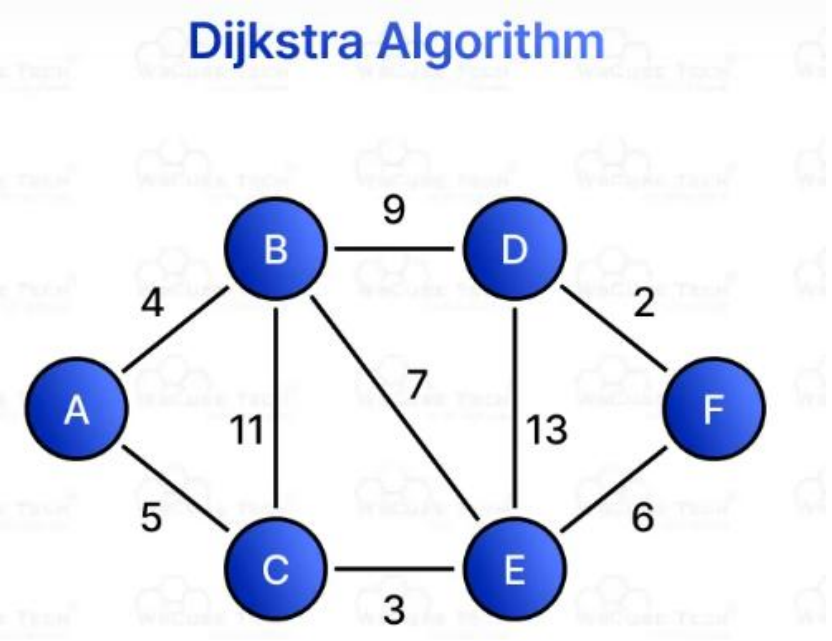
[2]. Dijkstra Algorithm: Example, Time Complexity, Code, WsCube Tech.
https://www.wscubetech.com/resources/dsa/dijkstra-algorithm

# Single Source Shortest Path Algorithms ..cont'd

**Example : -**Imagine we are planning a trip in a city where we need to visit different places, like a park, a museum, and a library.

- Each road between these places has a distance marked on it.
- We want to find the shortest route that lets us travel the least distance.

- Dijkstra's algorithm helps to do this by starting at current location, checking all the roads we can take, and picking the one with the shortest distance.
- Then, from the next place, it does the same thing, always choosing the shortest available path, until we reach to the destination.



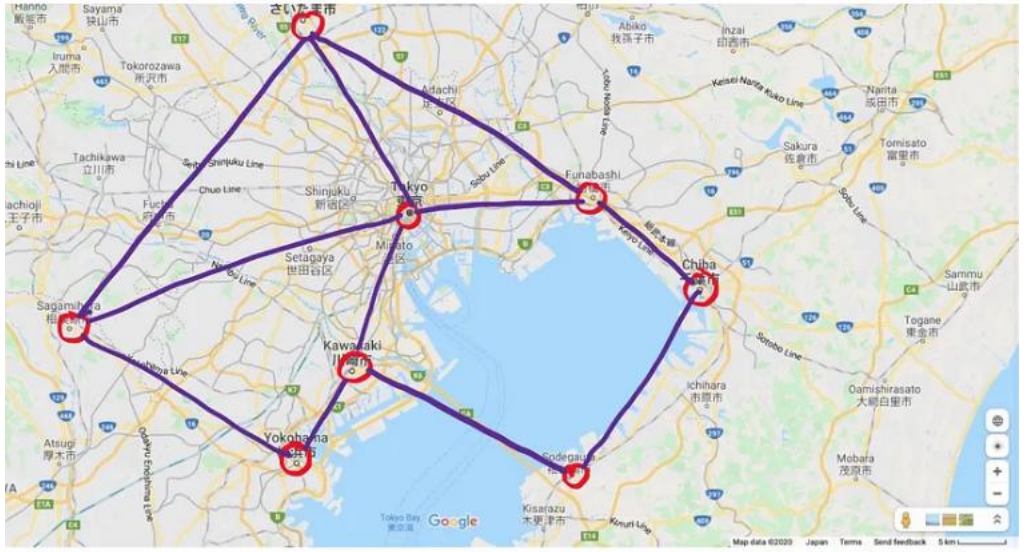

[2]. Dijkstra Algorithm: Example, Time Complexity, Code, WsCube Tech. https://www.wscubetech.com/resources/dsa/dijkstra-algorithm

# Single Source Shortest Path Algorithms ..cont'd

**Dijkstra Algorithm Example Problems**

Problem 1: Shortest Path in a Road Network

- **Problem:** Given a map of a city with various intersections (nodes) and roads (edges with distances), find the shortest path from a starting intersection to all other intersections in the city.
- **Solution:** Use Dijkstra's algorithm to compute the shortest path from the starting intersection to all others, ensuring efficient travel across the city.




https://medium.com/swlh/network-optimization-1-shortest-path-problem-3757a67a129c

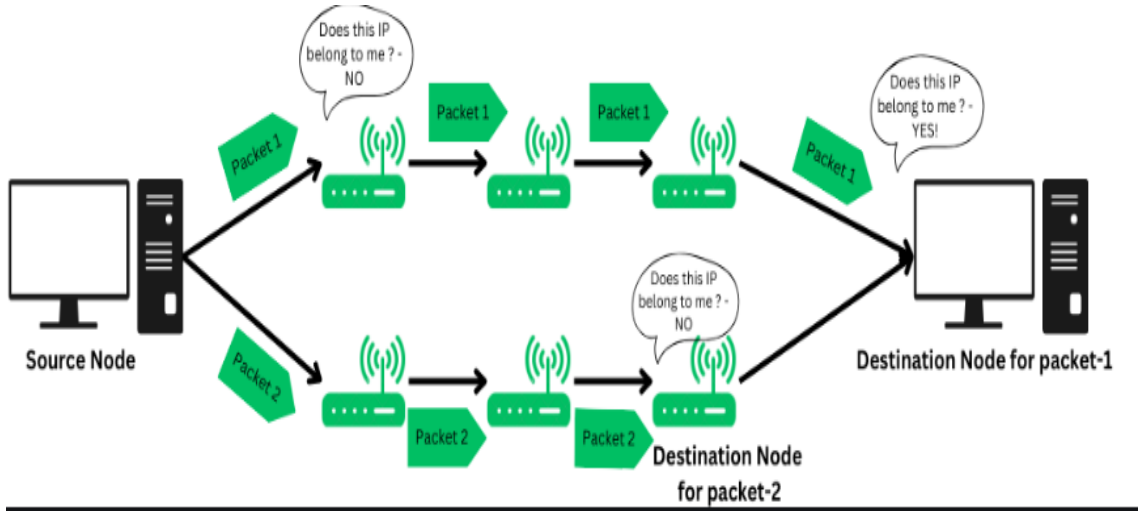
[2]. Dijkstra Algorithm: Example, Time Complexity, Code, WsCube Tech. https://www.wscubetech.com/resources/dsa/dijkstra-algorithm

# Single Source Shortest Path Algorithms ..cont'd

**Dijkstra Algorithm Example Problems**

Problem 2: Network Packet Routing

- **Problem:** In a computer network, we need to determine the most efficient route for data to travel from one server to another, minimizing the total latency.
- **Solution:** Apply Dijkstra's algorithm to calculate the shortest path based on the latency (weight) of each connection between servers.




https://www.geeksforgeeks.org/what-is-routing/

[2]. Dijkstra Algorithm: Example, Time Complexity, Code, WsCube Tech. https://www.wscubetech.com/resources/dsa/dijkstra-algorithm

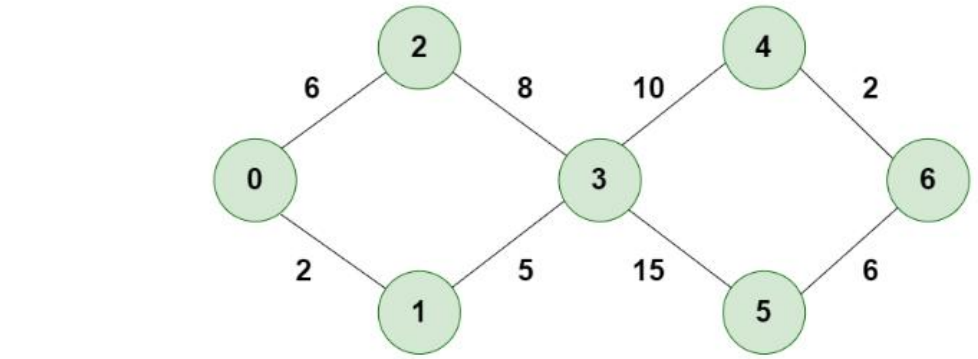
# Single Source Shortest Path Algorithms ..cont'd

**Algorithm for Dijkstra's Algorithm**

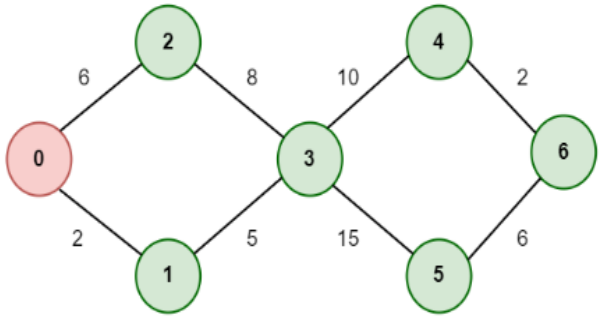
- Mark the source node with a current distance of 0 and the rest with infinity.
- Set the non-visited node with the smallest current distance as the current node.
- For each neighbor, N of the current node adds the current distance of the adjacent node with the weight of the edge connecting 0->1.
- If it is smaller than the current distance of Node, set it as the new current distance of N.
- Mark the current node 1 as visited.
- Go to step 2 if there are any nodes are unvisited.

# Single Source Shortest Path Algorithms ..cont'd

**Example :** Dijkstra's Algorithm will generate the shortest path from Node 0 to all other Nodes in the graph.




**STEP 1** Start from Node 0 and mark Node 0 as Visited and check for adjacent nodes

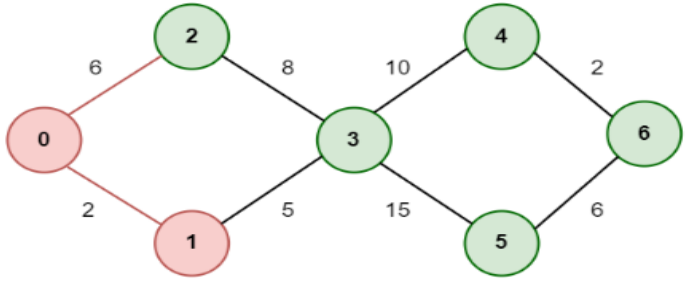



**Unvisited Nodes**
{0,1,2,3,4,5,6}

**Distance:**
0: 0 ✓
1: ∞
2: ∞
3: ∞
4: ∞
5: ∞
6: ∞

**Dijkstra's Algorithm**

**STEP 2** Mark Node 1 as Visited and add the Distance




**Unvisited Nodes**
{0,1,2,3,4,5,6}

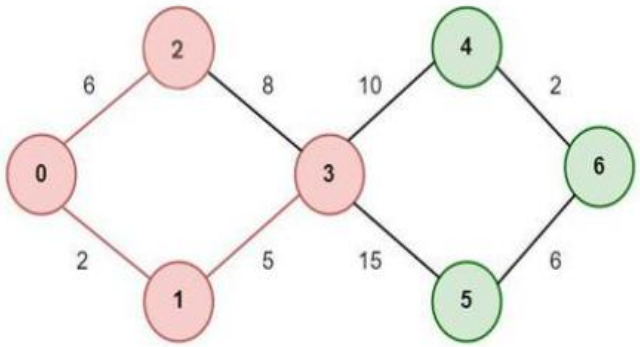
**Distance:**
0: 0 ✓
1: 2 ✓
2: ∞
3: ∞
4: ∞
5: ∞
6: ∞

**Dijkstra's Algorithm**

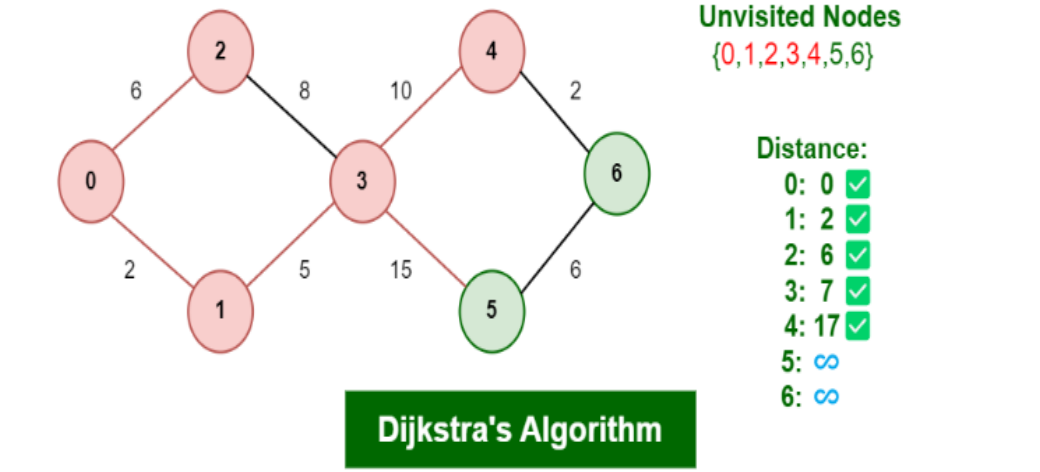

[3]. What is Dijkstra's Algorithm? | Introduction to Dijkstra's Shortest Path Algorithm, GeeksforGeeks, 2025. https://www.geeksforgeeks.org/introduction-to-dijkstras-shortest-path-algorithm/

# Single Source Shortest Path Algorithms ..cont'd

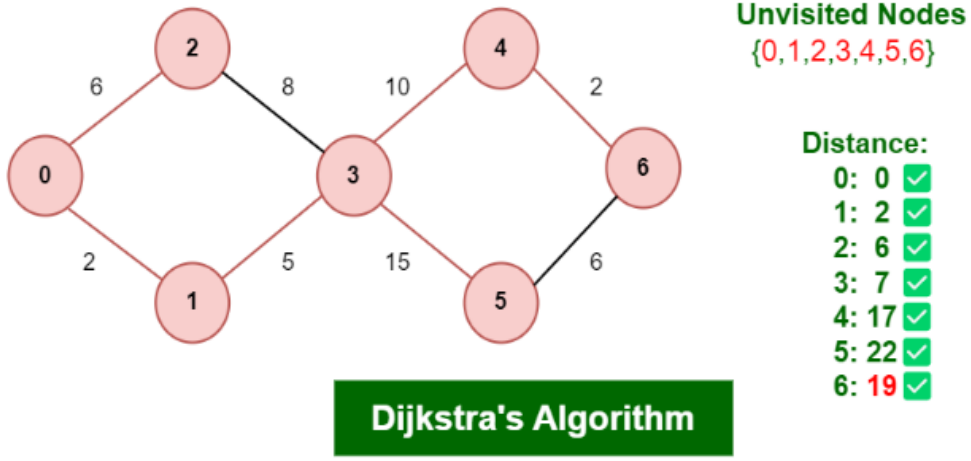
**STEP 3** Mark Node 3 as Visited after considering the Optimal path and add the Distance




**STEP 4** Mark Node 4 as Visited after considering the Optimal path and add the Distance




**STEP 5** Mark Node 6 as Visited and add the Distance




- ***So, the Shortest Distance from the Source Vertex is 19 which is optimal one***

# Single Source Shortest Path Algorithms ...cont'd

**Dijkstra's Algorithm in Dense vs. Sparse Graphs**

- The performance of Dijkstra's algorithm varies depending on the density of the graph:
  - **Dense Graphs**: Dijkstra's algorithm is more computationally intensive due to the higher number of edges, but the priority queue optimization can mitigate some of these issues.
    - Priority queue is a data structure that always extracts the node with the smallest distance.
  - **Sparse Graphs:** Dijkstra's algorithm is highly efficient, especially with priority queue optimization, making it a strong choice for these types of graphs.

[4]. Dijkstra Algorithm - Shortest Path Algorithm in Data Structures, WsCube Tech Team, WsCube Tech. https://www.wscubetech.com/resources/dsa/dijkstra-algorithm

# Single Source Shortest Path Algorithms ...cont'd

## Bellman-Ford algorithm

- A fundamental algorithm used to find the shortest path in a graph, even when some edges have negative weights.
- Unlike Dijkstra's algorithm, the Bellman-Ford algorithm can handle graphs with negative weight edges, making it a versatile choice for various applications
- A graph is like a network of points connected by lines, and each line has a number (or weight) that shows how long or difficult it is to travel between two points.
- The Bellman-Ford algorithm works even if some of these lines have negative numbers, which can represent situations where moving from one point to another might actually reduce the overall cost or distance.

[5]. Bellman-Ford Algorithm: Example, Time Complexity, Code , WsCube Tech, WsCube Tech. https://www.wscubetech.com/resources/dsa/bellman-ford-algorithm

# Single Source Shortest Path Algorithms ...cont'd

**Bellman-Ford algorithm with an easy Example:**

- Imagine we are planning a road trip.
- We have a map where towns are connected by roads, and each road has a number showing how long it takes to travel.
- Some roads might have a shortcut or a faster way that reduces our travel time.
  - The Bellman-Ford algorithm helps us find the quickest way to reach each town from our starting point, even if some shortcuts might initially seem longer or take on a longer route.
  - It checks all possible paths and updates our travel plan until it finds the best route to each town, considering all the shortcuts and different road lengths.

[5]. Bellman-Ford Algorithm: Example, Time Complexity, Code , WsCube Tech, WsCube Tech. https://www.wscubetech.com/resources/dsa/bellman-ford-algorithm

# Single Source Shortest Path Algorithms ..cont'd

## Principle of Relaxation of Edges

- Relaxation means **updating** the shortest distance to a node if a shorter path is found through another node. For an edge **(u, v)** with weight **w**:
  - If going through **u** gives a shorter path to v from the source node (i.e., **distance[v] > distance[u] + w**), we update the distance[v] as distance[u] + w.

  - In the bellman-ford algorithm, this process is repeated **(V – 1)** times for all the edges.
  - bellman-ford algorithm applies dynamic programming i.e., it tries all possible solution and pick up the best solution.

[5]. Bellman-Ford Algorithm: Example, Time Complexity, Code , WsCube Tech, WsCube Tech. https://www.wscubetech.com/resources/dsa/bellman-ford-algorithm

# Single Source Shortest Path Algorithms ..cont'd

**Step-by-step to understand the Bellman-Ford algorithm:**

- Initializing the source vertex with distance to itself as 0 and all other vertices as infinity. Creating the array with size N.
- Calculate the shortest distance from the source vertex. Following this process for N-1 times where N is the number of vertices in the graph.
  - For relaxing the path lengths for the vertices for each edge m-n:

  *if distance[n] > distance[m] + weight of edge mn, then*

  *distance[n] = distance[m] + weight of edge mn*

- Even after minimizing the path lengths for each vertex after N-1 times, if we can still relax the path length where distance[n] > distance[m] + weight of edge mn, then we can say that the graph contains the **negative edge cycle.**

# Single Source Shortest Path Algorithms ..cont'd

**Handling Negative Weight Cycles**

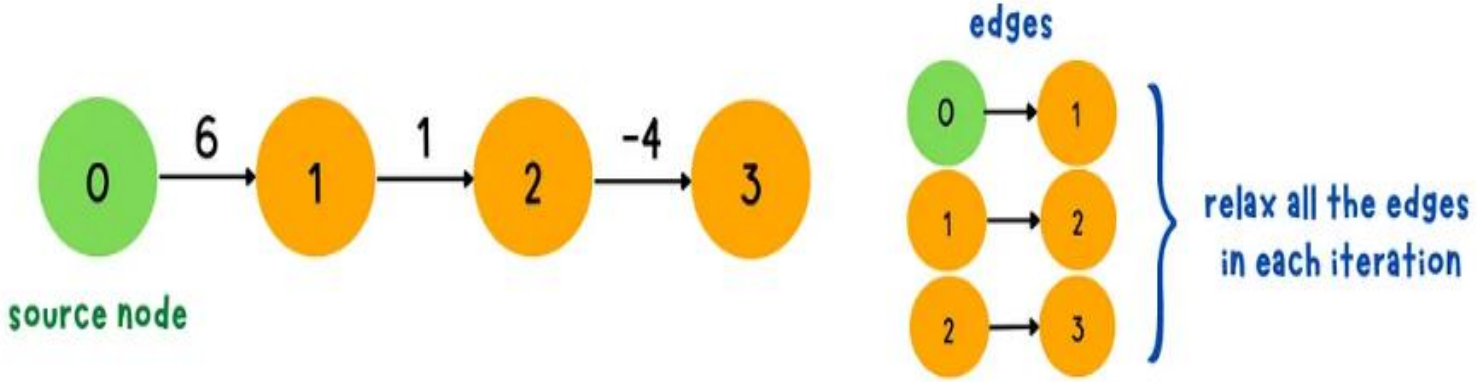
- The Bellman-Ford algorithm can detect negative weight cycles.
- After relaxing all the edges V-1 times (where V is the number of vertices), the algorithm performs one more pass through all the edges.
- If it finds that any edge can still be relaxed (i.e., if a shorter path is still found), it means there is a negative weight cycle in the graph.
- This is because if there were no negative weight cycles, the shortest paths would have been finalized after V-1 passes. The algorithm then reports the presence of a negative cycle, which **indicates that no valid shortest path exists for some nodes.**
- This capability to handle negative weights and cycles makes the Bellman-Ford algorithm particularly useful in scenarios where such conditions might exist, ensuring that we still get accurate and reliable path calculations.

[5]. Bellman-Ford Algorithm: Example, Time Complexity, Code , WsCube Tech, WsCube Tech. https://www.wscubetech.com/resources/dsa/bellman-ford-algorithm

# Single Source Shortest Path Algorithms ..cont'd

**Why Iterate (n-1) times in Bellman-Ford Algorithm?**

- In Bellman-Ford Algorithm, we can iterate all the edges in any order and relax them.
- Take the following graph as an example. There are four nodes in the graph and we will visit all the edges.

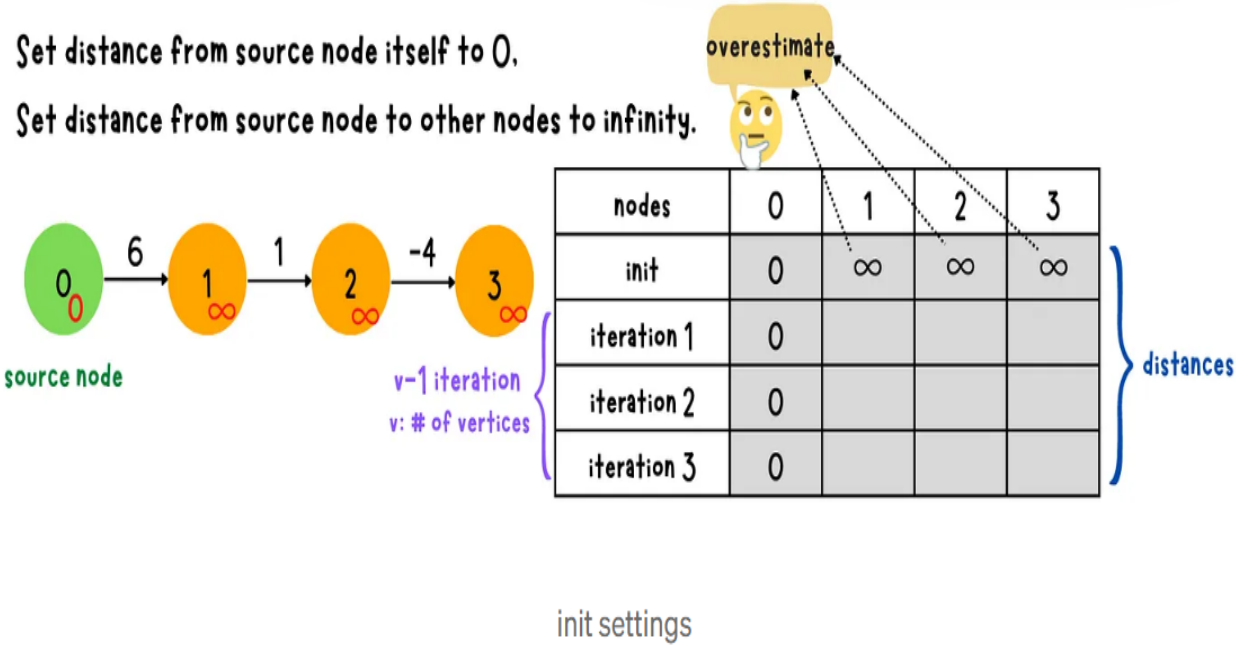



[6]. Bellman-Ford Algorithm, Claire Lee, YuminLee2, Sep. 2022.
https://yuminlee2.medium.com/bellman-ford-algorithm-9f9f331b4291

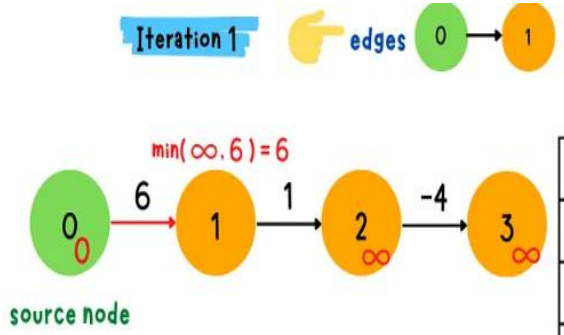
# Single Source Shortest Path Algorithms ..cont'd

**1. order: 0 → 1, 1 → 2, 2 → 3**

- If we visited edges in (0 → 1, 1 → 2, 2 → 3) order, we can get the shortest path after two iterations. There is no update occurred at the end of second iteration, so we stop the algorithm.



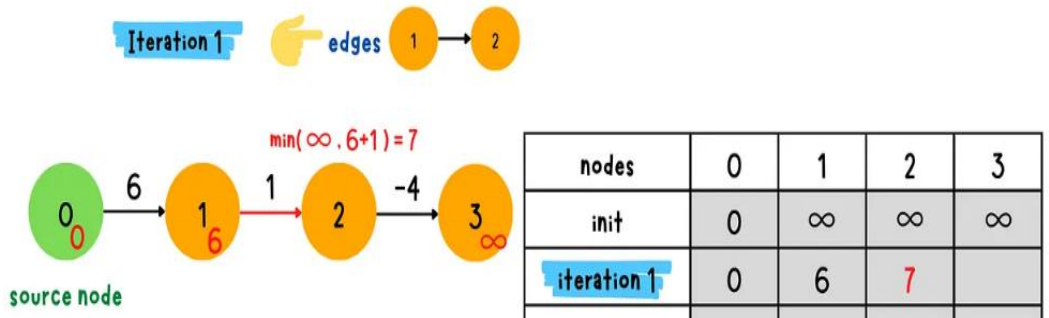

| nodes | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| init | 0 | ∞ | ∞ | ∞ |
| iteration 1 | 0 | | | |
| iteration 2 | 0 | | | |
| iteration 3 | 0 | | | |




| nodes | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| init | 0 | ∞ | ∞ | ∞ |
| iteration 1 | 0 | 6 | | |
| iteration 2 | 0 | | | |
| iteration 3 | 0 | | | |

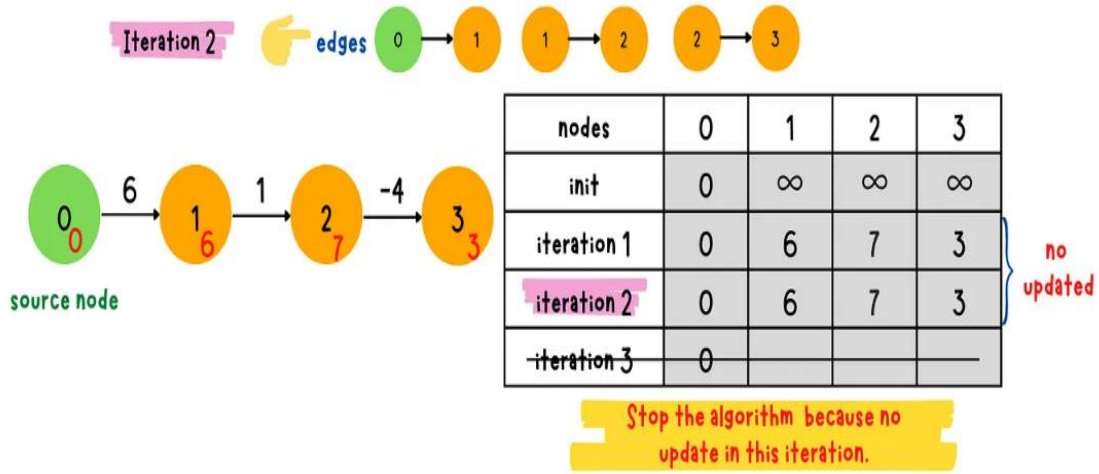

[6]. Bellman-Ford Algorithm, Claire Lee, YuminLee2, Sep. 2022.
https://yuminlee2.medium.com/bellman-ford-algorithm-9f9f331b4291

# Single Source Shortest Path Algorithms ..cont'd

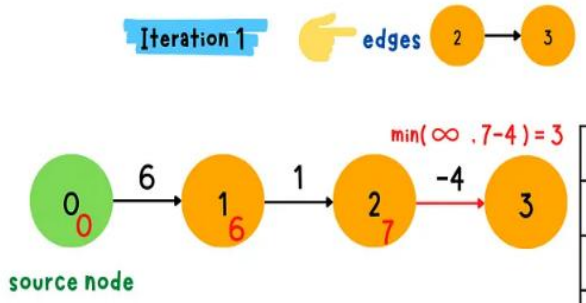



| nodes | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| init | 0 | ∞ | ∞ | ∞ |
| iteration 1 | 0 | 6 | 7 | |
| iteration 2 | 0 | | | |
| iteration 3 | 0 | | | |

iteration 1–2




| nodes | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| init | 0 | ∞ | ∞ | ∞ |
| iteration 1 | 0 | 6 | 7 | 3 |
| iteration 2 | 0 | 6 | 7 | 3 |
| ~~iteration 3~~ | 0 | | | |




| nodes | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| init | 0 | ∞ | ∞ | ∞ |
| iteration 1 | 0 | 6 | 7 | 3 |
| iteration 2 | 0 | | | |
| iteration 3 | 0 | | | |

iteration 1–3

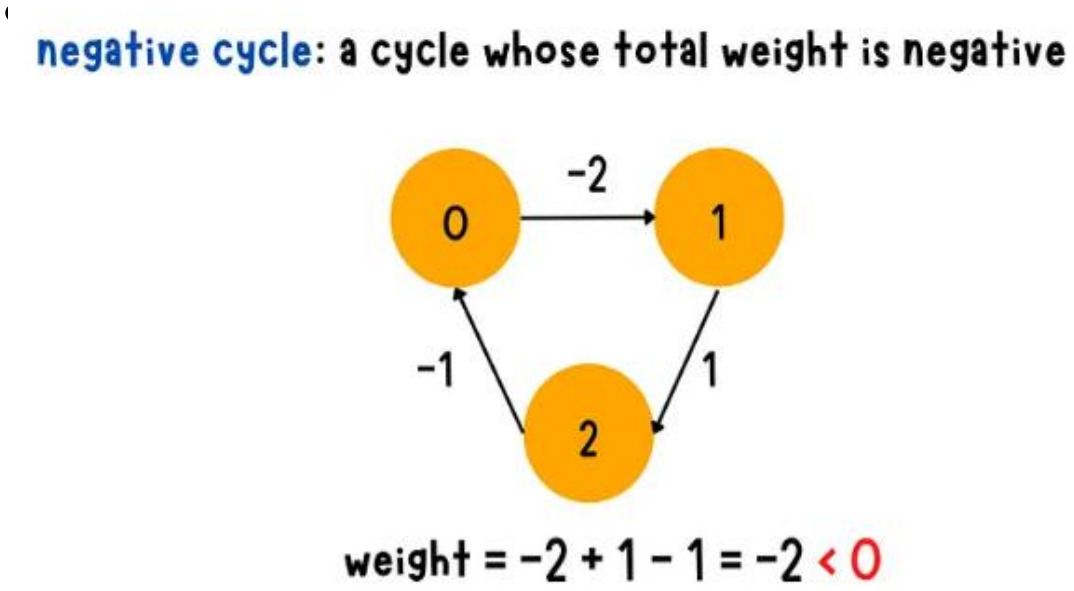
**Final shortest path on the last relaxation is :-**

distance[0] = 0
distance[1] = 6
distance[2] = 7
distance[3] = 3

# Single Source Shortest Path Algorithms ..cont'd

## Detect negative cycles

- When a graph contain negative cycles, it is not possible to find the shortest path from the source node to all other nodes. When a negative weight cycle is existed in a graph, every iteration of the cycle will give a shorter path.




- Bellman Ford algorithm provides a way to detect negative weight cycles in a graph. After we completed n-1 iterations, we perform the nth iteration. If any distance estimate is updated, a negative weight cycle must exist.

[6]. Bellman-Ford Algorithm, Claire Lee, YuminLee2, Sep. 2022. https://yuminlee2.medium.com/bellman-ford-algorithm-9f9f331b4291

# Single Source Shortest Path Algorithms ..cont'd

## Example Problems Using Bellman-Ford Algorithm

### 1. Single Source Shortest Path in a Graph with Negative Weights:

- **Problem:** Given a graph with negative weight edges, find the shortest path from a single source node to all other nodes.
- **Solution:** Use the Bellman-Ford algorithm to handle the negative weights and detect any negative weight cycles.

### 2. Detecting Negative Weight Cycles in a Graph:

- **Problem:** Given a graph, determine if it contains any negative weight cycles.
- **Solution:** After running the Bellman-Ford algorithm, check for further relaxations to identify negative weight cycles.

[5]. Bellman-Ford Algorithm: Example, Time Complexity, Code , WsCube Tech, WsCube Tech. https://www.wscubetech.com/resources/dsa/bellman-ford-algorithm

# Single Source Shortest Path Algorithms ..cont'd

## 3. Currency Arbitrage Detection:

- **Problem:** Given currency exchange rates, detect if there is a possibility of arbitrage (profit by converting one currency to another and back to the original).
  - **Solution:** Model the currency exchange as a graph with negative weights representing the algorithm of exchange rates, then use Bellman-Ford to detect negative cycles.

[5]. Bellman-Ford Algorithm: Example, Time Complexity, Code , WsCube Tech, WsCube Tech. https://www.wscubetech.com/resources/dsa/bellman-ford-algorithm

# Bellman-Ford vs Dijkstra Algorithm

| Feature | Bellman-Ford Algorithm | Dijkstra's Algorithm |
|---|---|---|
| Purpose | Finds the shortest path from a single source to all vertices | Finds the shortest path from a single source to all vertices |
| Handles Negative Weights | Yes, can handle graphs with negative weight edges | No, does not work with negative weight edges |
| Handles Negative Weight Cycles | Yes, detects and reports negative weight cycles | No, cannot handle graphs with negative weight cycles |
| Time Complexity | O(V * E), where V is the number of vertices, and E is edges | O(V^2) using an adjacency matrix, O(E + V log V) with a priority queue |
| Space Complexity | O(V) for the distance array, O(V + E) for storing edges | O(V) for the distance array, O(V + E) with a priority queue |
| Graph Type | Works on both directed and undirected graphs | Works on both directed and undirected graphs |
| Best Suited For | Graphs with negative weights or the need to detect negative cycles | Graphs with non-negative weights and efficient shortest path computation |
| Algorithm Type | Dynamic Programming | Greedy Algorithm |



[5]. Bellman-Ford Algorithm: Example, Time Complexity, Code , WsCube Tech, WsCube Tech. https://www.wscubetech.com/resources/dsa/bellman-ford-algorithm

# Bellman-Ford vs Dijkstra Algorithm

| Feature | Bellman-Ford Algorithm | Dijkstra's Algorithm |
|---|---|---|
| Iterations | Relaxes all edges (V-1) times | Relaxes each vertex once |
| Initial Distance Values | Initializes all distances to infinity except the source | Initializes all distances to infinity except the source |
| Edge Relaxation | Relaxes each edge in every iteration, updating the shortest path | Relaxes edges based on the closest vertex not yet processed |
| Complexity in Dense Graphs | Performs better in sparse graphs but slower in dense graphs due to O(V * E) complexity | Performs better in dense graphs with a priority queue |
| Optimal for Real-Time Systems | No, because of its higher time complexity | Yes, due to its efficiency and speed in non-negative graphs |
| Implementation Difficulty | Moderate complexity, more steps involved | Relatively straightforward with priority queue optimization |
| Example Use Cases | Currency arbitrage detection, network routing with potential negative weights | GPS navigation systems, real-time pathfinding in games |

[5]. Bellman-Ford Algorithm: Example, Time Complexity, Code , WsCube Tech, WsCube Tech. https://www.wscubetech.com/resources/dsa/bellman-ford-algorithm

# Applications of Bellman-Ford Algorithm

- **Finding Shortest Paths:** Computes shortest paths from a single source in graphs with negative weight edges.
- **Network Routing Protocols:** Used in distance-vector routing protocols like RIP (Routing Information Protocol) to find the shortest paths in network routing.
- **Arbitrage Detection:** Detects negative weight cycles in financial graphs, indicating the possibility of arbitrage opportunities.
- **Detecting Negative Weight Cycles:** Identifies negative weight cycles in a graph, which can indicate issues like unsustainable operations or financial arbitrage.
- **Geographic Information Systems (GIS):** Calculates shortest paths in maps where certain routes might have penalties or benefits, represented by negative weights.

[5]. Bellman-Ford Algorithm: Example, Time Complexity, Code , WsCube Tech, WsCube Tech. https://www.wscubetech.com/resources/dsa/bellman-ford-algorithm

# Real World Applications of Shortest Path Algorithms

- GPS Navigation
- Network Routing
- Transportation Planning
- Airline Flight Planning
- Social Network Analysis
- Circuit Design

[7]. Shortest Path Algorithms - A Complete Guide, GeeksforGeeks Team, GeeksforGeeks. https://www.geeksforgeeks.org/shortest-path-algorithms-a-complete-guide/

# Summary

- The shortest path algorithms are the ones that focuses on calculating the minimum travelling cost from source node to destination node of a graph in optimal time and space complexities.
- Shortest-path algorithms categorized into two :-Single Source Shortest Path Algorithms and all Pair Shortest Path Algorithms
- Dijkstra's algorithm helps to do this by starting at current location, checking all the roads we can take, and picking the one with the shortest distance.
- Bellman-Ford algorithm is a fundamental algorithm used to find the shortest path in a graph, even when some edges have negative weights.
- Unlike Dijkstra's algorithm, the Bellman-Ford algorithm can handle graphs with negative weight edges, making it a versatile choice for various applications.
- Real World Applications of Shortest Path Algorithms includes  GPS Navigation, Network Routing, Transportation Planning, Airline Flight Planning, Social Network Analysis, Circuit Design

# References


1. Shortest Path Algorithm Tutorial with Problems, GeeksforGeeks, November 23, 2023.https://www.geeksforgeeks.org/shortest-path-algorithms-a-complete-guide.
2. Dijkstra Algorithm: Example, Time Complexity, Code, WsCube Tech. https://www.wscubetech.com/resources/dsa/dijkstra-algorithm
3. What is Dijkstra's Algorithm? | Introduction to Dijkstra's Shortest Path Algorithm, GeeksforGeeks, 2025. https://www.geeksforgeeks.org/introduction-to-dijkstras-shortest-path-algorithm
4. Dijkstra Algorithm - Shortest Path Algorithm in Data Structures, WsCube Tech Team, WsCube Tech. https://www.wscubetech.com/resources/dsa/dijkstra-algorithm
5. Bellman-Ford Algorithm: Example, Time Complexity, Code , WsCube Tech, WsCube Tech. https://www.wscubetech.com/resources/dsa/bellman-ford-algorithm
6. Bellman-Ford Algorithm, Claire Lee, YuminLee2, Sep. 2022. https://yuminlee2.medium.com/bellman-ford-algorithm-9f9f331b4291
7. Shortest Path Algorithms - A Complete Guide, GeeksforGeeks Team, GeeksforGeeks. https://www.geeksforgeeks.org/shortest-path-algorithms-a-complete-guide

# Thank You!

## For your attention