**What are the differences between Bellman Ford’s and Dijkstra’s algorithms?**

* Difficulty Level : [Easy](https://www.geeksforgeeks.org/easy/)
* Last Updated : 23 Jun, 2022

[**Bellman Ford’s algorithm**](https://www.geeksforgeeks.org/bellman-ford-algorithm-dp-23/)   
Like other [Dynamic Programming Problems](https://www.geeksforgeeks.org/solve-dynamic-programming-problem/), the algorithm calculates shortest paths in a bottom-up manner. It first calculates the shortest distances which have at-most one edge in the path. Then, it calculates the shortest paths with at-most 2 edges, and so on. After the i-th iteration of outer loop, the shortest paths with at most i edges are calculated. There can be maximum |V| – 1 edge in any simple path, that is why the outer loop runs |v| – 1 time. The idea is, assuming that there is no negative weight cycle if we have calculated shortest paths with at most i edges, then an iteration over all edges guarantees to give the shortest path with at-most (i+1) edges

[**Dijkstra’s algorithm**](https://www.geeksforgeeks.org/dijkstras-shortest-path-algorithm-greedy-algo-7/)   
Dijkstra’s algorithm is very similar to Prim’s algorithm for minimum spanning tree. Like Prim’s MST, we generate an SPT (shortest path tree) with a given source as root. We maintain two sets, one set contains vertices included in the shortest-path tree, other set includes vertices not yet included in the shortest-path tree. At every step of the algorithm, we find a vertex which is in the other set (set of not yet included) and has a minimum distance from the source.

**Differences between Bellman Ford’s and Dijkstra’s algorithm:**

**Bellman Ford’s algorithm** and **Dijkstra’s algorithm** both are single-source shortest path algorithm, i.e. both determines the shortest distance of each vertex of a graph from a single source vertex. However, there are some key differences between them. We follow the Dynamic Programming approach in Bellman Ford’s algorithm and Greedy approach in Dijkstra’s algorithm. Let’s see the other major differences between these two techniques-

| Bellman Ford’s Algorithm | Dijkstra’s Algorithm |
| --- | --- |
| Bellman Ford’s Algorithm works when there is negative weight edge, it also detects the negative weight cycle. | Dijkstra’s Algorithm may or may not work when there is negative weight edge. But will definitely not work when there is a negative weight cycle. |
| The result contains the vertices which contains the information about the other vertices they are connected to. | The result contains the vertices containing whole information about the network, not only the vertices they are connected to. |
| It can easily be implemented in a distributed way. | It can not be implemented easily in a distributed way. |
| It is more time consuming than Dijkstra’s algorithm. Its time complexity is O(VE). | It is less time consuming. The time complexity is O(E logV). |
| Dynamic Programming approach is taken to implement the algorithm. | Greedy approach is taken to implement the algorithm. |
| Bellman Ford’s Algorithm have more overheads than Dijkstra’s Algorithm. | Dijkstra’s Algorithm have less overheads than Bellman Ford’s Algorithm. |
| Bellman Ford’s Algorithm have less scalability than Dijkstra’s Algorithm. | Dijkstra’s Algorithm have more scalability than Bellman Ford’s Algorithm.  . |