Semester VI (B.Tech)

Jaypee University of Engineering and Technology, Guna

Advanced Programming Lab-3 18B17CI673

Lab Exercise □: Graph Algorithms

Title: Implementation of Graph Algorithms for Shortest Path Calculation

Mode: Self Learning

Outcomes:

1. To implement and compare different algorithms for finding the shortest path in a graph
2. To evaluate the performance of the algorithms based on their running time and memory usage
3. To gain a better understanding of graph theory and its applications in computer science

Methodology:

1. Selection of algorithms: Dijkstra's algorithm, Bellman-Ford algorithm, and Floyd-Warshall algorithm
2. Selection of programming language and libraries: Python and NetworkX
3. Selection of datasets: Randomly generated graphs with different sizes and densities
4. Implementation of algorithms using the selected programming language and libraries
5. Running the implemented algorithms on the selected datasets
6. Measuring the running time and memory usage of the algorithms using appropriate tools
7. Analysis of the results and comparison of the algorithms based on their performance

Steps:

1. Selection of algorithms: Dijkstra's algorithm, Bellman-Ford algorithm, and Floyd-Warshall algorithm
2. Selection of programming language and libraries: Python and NetworkX
3. Generation of random graphs with different sizes and densities using NetworkX
4. Implementation of Dijkstra's algorithm for shortest path calculation using Python and NetworkX
5. Implementation of Bellman-Ford algorithm for shortest path calculation using Python and NetworkX
6. Implementation of Floyd-Warshall algorithm for shortest path calculation using Python and NetworkX
7. Running the implemented algorithms on the generated graphs
8. Measuring the running time and memory usage of the algorithms using appropriate tools such as the Python time module and memory_profiler library

9. Analysis of the results and comparison of the algorithms based on their performance using appropriate statistical analysis techniques
10. Writing a report summarizing the methodology, results, and conclusions of the experiment.

Note: Specific steps for each problem need to be designed based on the problem's requirements and properties. For example, for the Knapsack Problem, the steps could include generating random items and weights, implementing the Naive and Dynamic Programming algorithms, running the algorithms on the generated items and weights, measuring the running time and memory usage of the algorithms, and analyzing the results. Similarly, for Longest Common Subsequence Problem, the steps could include generating random strings, implementing the Naive and Dynamic Programming algorithms, running the algorithms on the generated strings, measuring the running time and memory usage of the algorithms, and analyzing the results.

Experiments:

1. Shortest Path Calculation using Dijkstra's Algorithm
2. Minimum Spanning Tree using Kruskal's Algorithm
3. Graph Traversal using Breadth-First Search (BFS)
4. Graph Traversal using Depth-First Search (DFS)
5. All Pairs Shortest Path using Floyd-Warshall Algorithm
6. Topological Sorting of Directed Acyclic Graphs (DAGs)
7. Maximum Flow in a Flow Network using Ford-Fulkerson Algorithm
8. Traveling Salesman Problem using Brute Force and Dynamic Programming
9. Graph Coloring using Greedy Algorithm
10. Strongly Connected Components using Kosaraju's Algorithm.