Semester VI (B.Tech)

Jaypee University of Engineering and Technology, Guna

Advanced Programming Lab-3 18B17CI673

Lab Exercise 9: Dynamic Programming Algorithms

Title: Dynamic Programming Algorithms: A Hands-On Exploration

Mode: Self Learning

Outcomes:

1. To understand the principles of dynamic programming and its applications in various domains
2. To implement and compare different dynamic programming algorithms from different domains
3. To evaluate the performance of the algorithms based on their running time and memory usage
4. To gain hands-on experience in algorithm design and analysis

Methodology:

1. Selection of problems from different domains: Knapsack Problem, Longest Common Subsequence Problem, Matrix Chain Multiplication Problem, Shortest Path Problem, and Edit Distance Problem
2. Selection of programming language and libraries: Python and NumPy
3. Generation of random inputs with different sizes and distributions using NumPy
4. Implementation of the selected algorithms using Python and NumPy
5. Running the implemented algorithms on the generated inputs
6. Measuring the running time and memory usage of the algorithms using appropriate tools such as the Python time module and memory_profiler library
7. Analysis of the results and comparison of the algorithms based on their performance using appropriate statistical analysis techniques
8. Writing a report summarizing the methodology, results, and conclusions of the experiment.

Steps:

1. Select a problem from the selected domains, such as the Knapsack Problem.
2. Generate random inputs with different sizes and distributions using NumPy.
3. Implement the Naive algorithm for the problem.
4. Implement the Dynamic Programming algorithm for the problem.
5. Run the implemented algorithms on the generated inputs.
6. Measure the running time and memory usage of the algorithms using appropriate tools such as the Python time module and memory_profiler library.

7. Analyze the results and compare the algorithms based on their performance using appropriate statistical analysis techniques.
8. Repeat steps 1-7 for each problem in the selected domains.
9. Write a report summarizing the methodology, results, and conclusions of the experiment.
10. Measuring the running time and memory usage of the algorithms using appropriate tools such as the Python time module and memory_profiler library
11. Analysis of the results and comparison of the algorithms based on their performance using appropriate statistical analysis techniques
12. Writing a report summarizing the methodology, results, and conclusions of the experiment.

Experiments:

1. Knapsack Problem: generate random items and weights, implement Naive algorithm, implement Dynamic Programming algorithm, run algorithms on generated items and weights, measure running time and memory usage, analyze results.
2. Longest Common Subsequence Problem: generate random strings, implement Naive algorithm, implement Dynamic Programming algorithm, run algorithms on generated strings, measure running time and memory usage, analyze results.
3. Matrix Chain Multiplication Problem: generate random matrices, implement Naive algorithm, implement Dynamic Programming algorithm, run algorithms on generated matrices, measure running time and memory usage, analyze results.
4. Shortest Path Problem: generate random graphs, implement Naive algorithm, implement Dynamic Programming algorithm, run algorithms on generated graphs, measure running time and memory usage, analyze results.
5. Edit Distance Problem: generate random strings, implement Naive algorithm, implement Dynamic Programming algorithm, run algorithms on generated strings, measure running time and memory usage, analyze results.