Algorithm Analysis and Design- II Lab Manual

TextBook: Problem Solving in Data Structures & Algorithm using JAVA- Hemant Jain

B.Tech 4 Semester

Section 23412A1, 23412C3
ITER, SOA University

May 12, 2025

Overview

Lab Assignment 1: Graph Traversal Algorithms (Chapter 12)

1. Write a JAVA program to represent a graph using adjacency matrix. **(Example 12.1)**
2. Write a JAVA program to represent a graph using adjacency matrix. **(Example 12.2)**
3. Write a JAVA program to implement Depth First Search algorithm using stack. **(Example 12.3)**
4. Write a JAVA program to implement Depth First Search algorithm using recursion. **(Example 12.4)**
5. Write a JAVA program to implement the Breadth First Search (BFS) algorithm. **(Example 12.5)**
6. Write a JAVA program to find if there exists a path between any two vertices of a graph. **(Example 12.7)**
7. Given a source vertex and a destination vertex, write a JAVA program to find all the possible paths from source to destination. **(Example 12.8)**

Lab Assignment 2: Graph Algorithms (Chapter 12)

1. Write a JAVA program to print all the paths from a source vertex to a destination vertex. **(Example 12.9)**
2. Write a JAVA program to find the distance between source and destination vertex in a graph. **(Example 12.13)**
3. Given a directed graph, write a JAVA program to find if there exists a cycle in the graph. **(Example 12.14)**
4. Given an undirected graph, write a JAVA program to find if there exists a cycle in the graph using disjoint sets data structure. **(Example 12.18)**
5. Given a directed graph, write a JAVA program to find if this graph is strongly connected or not. **(Example 12.21)**
6. Write a JAVA program to implement Prims Algorithm using adjacency matrix representation of a graph. **(Example 12.24)**

Lab Assignment 3: Hashing

Hashing

- Hash Table: Data Structure for storing and retrieving data in O(1) time
- Terms: search key (e.g. Roll number, Aadhar number etc..), hash table (key-value data structure), hash function (function to map placement of values at the key positions)
- Example- Given 57, 92, 51, 27, 65, 68
- Examples of hash function- k mod n, mid square, folding function
- Problem- Collision (element already present at the key index) Example- Given 57, 92, 51, 27, 65, 68, 97, 22

Collision Resolution Techniques

1. Separate Chaining (Open hashing)
2. Open addressing (closed hashing)
    1. Linear Probing
    2. Quadratic Probing
    3. Double hashing

Lab Assignment 3

Collision Resolution Technique

1. Separate Chaining (Open hashing)
   - Create a linked list for values to be inserted at a key
   - Example- Given 57, 92, 51, 27, 65, 68, 97, 22, hash function (k mod 5)
   - Requires extra space
   - Pros- Insertion in O(1) time
   - Cons- Deletion and Searching in O(n) time, requires extra space despite of having space in hash table
2. Open addressing (closed hashing)
   1. Linear Probing
      - Inserts the colliding value at the next available location in hash table
      - Hash function h'(k) = (h(k) + i)mod n where i is collision number/probe number
      - Example- Given 57, 92, 51, 27, 65, 68, 97, 22, hash function h(k)= (k mod 10), h'(k) = (h(k)+i)mod 10
      - Pros- No extra space requirements
      - Cons- Search time O(n)-worst case, deletion is difficult as it creates empty spaces, so search of next element is difficult, primary clustering (increases the search time), secondary clustering (searching the keys again and again for placing elements)

Lab Assignment 3

Collision Resolution Technique

1. Open addressing (closed hashing)
    1. Quadratic Probing
        - Hash function $h'(k) = (h(k) + i^2) mod n$ where i is collision number/probe number
        - Example = 52, 96, 11, 43, 98, 37, 26, 22
        - Pros- No extra space required, primary clustering resolved
        - Cons- Search Time- O(n), Secondary clustering, no guarantee of finding slot
    2. Double hashing (not in syllabus, only for reading purpose)
        - Two hash functions are used
        - $h_1(k) = k mod 11$, $h_2(k) = 8 - (k mod 8)$, $h(k) = (h_1(k) + i * h_2(k)) mod 11$
        - Example- 10, 44, 85, 20, 66
        - Pros- No extra space, no primary clustering, no secondary clustering
        - Cons- Search complexity- O(n)

Lab Assignment 3

- Write a JAVA program to implement hashing using linear probing.
- Write a JAVA program to implement hashing using quadratic probing.
- Write a JAVA program to implement hashing using separate chaining.

Lab Assignment 4: String Matching Algorithms

1. Write a JAVA program to match two strings (using BRUTE Force approach)
2. Write a JAVA program to match two strings using Rabin Karp algorithm.

***String Matching**- Given a string T= "abcdefghij" and a pattern P="def", you have to find whether substring/pattern "def" is present in given string or not.

Lab Assignment 5: String Algorithms

1. Write a JAVA program to find the presence of matching order of string. (e.g. T="ABCDEFGHIJ", M="BEFJ"- matching order- yes, M="BHDJ"-matching order-no.
2. Write a JAVA program to find if a string has unique characters or not.
3. Write a JAVA program to convert the case of string (upper to lower case and lower to upper case). e.g.- abcdEFGH converted to ABCDefgh
4. Write a JAVA program to check if a string is palindrome or not.
5. Write a JAVA program to compare the equality of two strings.
6. Write a JAVA program to reverse a string.
7. Write a JAVA program to contatenate two strings. e.g. A= abc, B= DEF, concatenated string A.B = abcDEF

Lab Assignment 6: Greedy Algorithms

1. Write a JAVA program to implement fractional knapsack problem using greedy heuristic of profit/weight.
2. Write a JAVA program to implement interval scheduling problem.

Lab Assignment 7: Greedy Algorithms

1. Write a JAVA program to implement huffman coding algorithm.

Lab Assignment 8: Divide and Conquer

1. Write a JAVA program to implement merge sort algorithm.
2. Write a JAVA program to implement counting inversion algorithm.

Lab Assignment 9: Divide and Conquer

1. Write a JAVA program to implement quick sort algorithm.

Lab Assignment 10: Dynamic Programming

1. Write a JAVA program to implement coin exchange problem.
2. Write a JAVA program to implement weighted interval scheduling.

The End