Department of
Computer Science and Engineering

Title: Implement Prim's Algorithm

Data Structure Lab
CSE 106

Green University of Bangladesh

1 Objective(s)

- To learn Prim's algorithm to find MST of a graph.

2 Problem Analysis

2.1 Prim's Algorithm

Prim's algorithm is a minimum spanning tree algorithm that takes a graph as input and finds the subset of the edges of that graph which

- form a tree that includes every vertex.
- has the minimum sum of weights among all the trees that can be formed from the graph.

2.2 How Prim's algorithm works

It falls under a class of algorithms called greedy algorithms that find the local optimum in the hopes of finding a global optimum. We start from one vertex and keep adding edges with the lowest weight until we reach our goal. The steps for implementing Prim's algorithm are as follows:

- Initialize the minimum spanning tree with a vertex chosen at random.
- Find all the edges that connect the tree to new vertices, find the minimum and add it to the tree.
- Keep repeating step 2 until we get a minimum spanning tree.

2.3 Example of Prim's algorithm

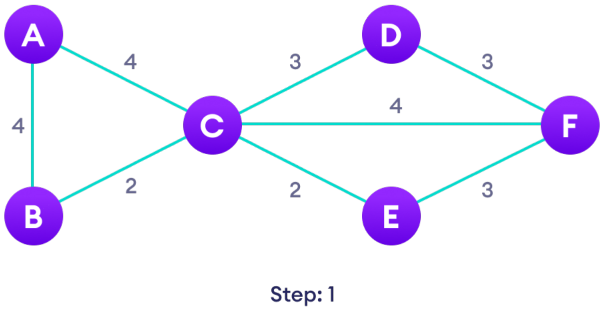
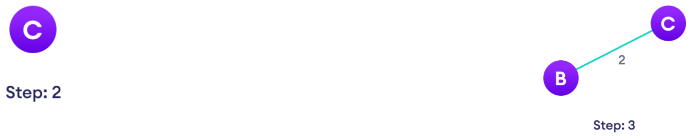


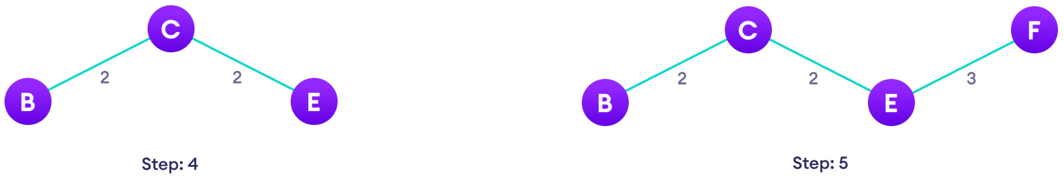
Figure 1: Start with a weighted graph



(a) Choose the edge with the least weight, if there are more than 1, choose anyone

(b) Choose the next shortest edge and add it

Figure 2: Step 2 and 3



(a) Choose the next shortest edge that doesn't create a cycle and add it

(b) Choose the next shortest edge that doesn't create a cycle and add it

Figure 3: Step 4 and 5

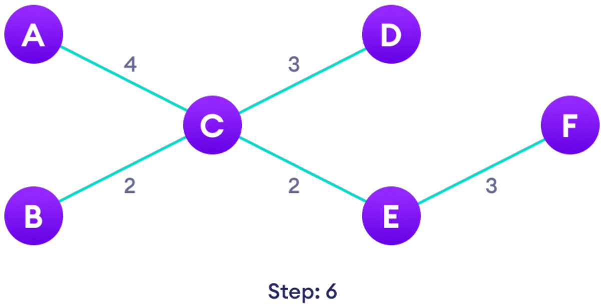


Figure 4: Repeat until you have a spanning tree

3 Algorithm

Algorithm 1: Prim's Algorithm

```
T = $\emptyset$;
U =  1 ;
while $(U \neq V)$ do
    let (u, v) be the lowest cost edge such that u $\in$ U and v $\in$ V - U;
    T = T $\cup$ (u, v)
    U = U $\cup$ v
end
```

4 Implementation in C

```c
// Prim's Algorithm in C

#include<stdio.h>
#include<stdbool.h>

#define INF 9999999

// number of vertices in graph
#define V 6

// create a 2d array of size 6x6
//for adjacency matrix to represent graph

int G[V][V] = {
    {0,4,4,0,0,0},
    {4,0,2,0,0,0},
    {4,2,0,3,2,4},
    {0,0,3,0,0,3},
    {0,0,2,0,0,3},
    {0,0,4,3,3,0}
};

int main() {
  int no_edge;  // number of edge

  // create a array to track selected vertex
  // selected will become true otherwise false
  int selected[V];

  // set selected false initially
  memset(selected, false, sizeof(selected));

  // set number of edge to 0
  no_edge = 0;

  // the number of egde in minimum spanning tree will be
  // always less than (V -1), where V is number of vertices in
  //graph

  // choose 0th vertex and make it true
  selected[0] = true;

  int x;  //  row number
  int y;  //  col number

  // print for edge and weight
  printf("Edge : Weight\n");

  while (no_edge < V - 1) {
    //For every vertex in the set S, find the all adjacent vertices
    // , calculate the distance from the vertex selected at step 1.
    // if the vertex is already in the set S, discard it otherwise
    //choose another vertex nearest to selected vertex  at step 1.

    int min = INF;
    x = 0;
    y = 0;

    for (int i = 0; i < V; i++) {
      if (selected[i]) {
        for (int j = 0; j < V; j++) {
          if (!selected[j] && G[i][j]) {  // not in selected and there is an edge
            if (min > G[i][j]) {
              min = G[i][j];
              x = i;
              y = j;
            }
          }
        }
      }
    }
    printf("%d - %d : %d\n", x, y, G[x][y]);
    selected[y] = true;
    no_edge++;
  }

  return 0;
}
```

5 Sample Input/Output (Compilation, Debugging & Testing)

Input: Weight edges graph like figure 1

Output:

```
0 - 1 => 4
1 - 2 => 2
2 - 4 => 2
2 - 3 => 3
3 - 5 => 3
```

6 Discussion & Conclusion

Based on the focused objective(s) to understand about the MST algorithms, the additional lab exercise made me more confident towards the fulfilment of the objectives(s).

7 Lab Task (Please implement yourself and show the output to the instructor)

1. Write a Program in java to find the Second Best Minimum Spanning Tree using Prim's Algorithm.

7.1 Problem analysis

A Minimum Spanning Tree T is a tree for the given graph G which spans over all vertices of the given graph and has the minimum weight sum of all the edges, from all the possible spanning trees. A second best MST T′ is a spanning tree, that has the second minimum weight sum of all the edges, from all the possible spanning trees of the graph G.

8 Lab Exercise (Submit as a report)

- Find the number of distinct minimum spanning trees for a given weighted graph.

9 Policy

Copying from internet, classmate, seniors, or from any other source is strongly prohibited. 100% marks will be *deducted* if any such copying is detected.