**CONSTRUCT A MINIMUM SPANNING TREE USING PRIM’S ALGORITHM**

**Project-1 Submitted in Partial fulfillment for the award of**

**B.Sc.Degree in Computer Science**

**Madurai Kamaraj University , Madurai.**

**NAME :M.NAVEENKUMAR**

**ROLL.NO :19AUCS006**

**CLASS : III-B.Sc.COMPUTER SCIENCE**

**SUBMITTED ON :26-04-2022**

**PROJECT GUIDE : DR.T.KATHIRVALAVAKUMAR**

****

Research Centre in Computer Science

V.H.N.SENTHIKUMARA NADAR COLLEGE (Autonomous)

Virudhunagar.

**CONSTRUCT A MINIMUM SPANNING TREE USING PRIM’S ALGORITHM**

**Project-1 Submitted in Partial fulfillment for the award of**

**B.Sc.Degree in Computer Science**

**Madurai Kamaraj University , Madurai.**

**NAME :M.NAVEENKUMAR**

**ROLL.NO :19AUCS006**

**CLASS : III-B.Sc.COMPUTER SCIENCE**

**SUBMITTED ON :26-04-2022**

**PROJECT GUIDE : DR.T.KATHIRVALAVAKUMAR**

****

**Project Guide Signature HOD Signature**

Research Centre in Computer Science

V.H.N.SENTHIKUMARA NADAR COLLEGE (Autonomous)

Virudhunagar.

**CONSTRUCT A MINIMUM SPANNING TREE USING PRIM’S ALGORITHM**

**Objective**

Generating Minimum Cost Spanning tree using Prim’s Algorithm identifying the Optimal route for Delivering the Courier Parcels to Different Places.

**Problem Description**

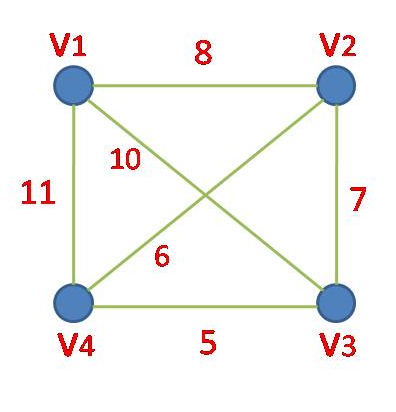
**Courier service Companies** deliver the courier parcels an daily basis,It has been Observed That they are not following the optimal route plan. It leads to more petrol cost and more transportation time for delivering the courier parcels. To solve this problem and to generate optimal path to different places from courier office , distances between cities where courier parcels to be Delivered are to be collected and its graph structure ,to be drawn, from this drawn graph structure ,minimum cost spanning tree can be generated to get the optimal route to different places from a courier office.

**Procedure**

* Construct adjacent matrix w to represent a graph with n places.
* Select one starting place assume v1.
* Construct two arrays “nearest” and “distance for the places.
* Generate solution set w which contains places nearer to starting place.
* Initialy solution set y will have starting place only nearest{i}.
* Weight of i vi to the place nearer to nearest [i].
* Initialy nearest of [i] or v1 for all places.
* Initialy distance of i is w[1][i].
* Find the minimum value in distance of [i] if the minimum value is corresponding to the place k (that is distance of k is minimum),then add the place vk to the solution set with the corresponding edge.
* V1 to vk (here v1 is nearer to vk).
* Update the array nearest [] and distance [] for the places not included in the solution set.
* do the above two steps to find nearer place and the corresponding edge.
* Repeat the above three steps (n-1) time to obtain minimum spanning tree.

**Experimental Results**

First Graph



Adjacency Matrix

V1 V2 V3 V4

V1 0 8 10 11

V2 8 0 7 6

V3 10 7 0 5

V4 11 6 5 0

Assume starting place v1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | V1 | V2 | V3 | V4 |
| Nearer place | ̶ | V1 | V1 | V1 |
| distance | ̶ | 8 | 10 | 11 |

Resultant selected vertices f={v1,v2}

Resultant selected corresponding Edge y={v1 →v2=8}

Calculate nearer vertex from f to the remaining vertices of the graph.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | V1 | V2 | V3 | V4 |
| Nearer place | ̶ | ̶ | V2 | V2 |
| distance | ̶ | ̶ | 7 | 6 |

Resultant selected vertices f={v1,v2,v4}

Resultant selected corresponding Edges y={v1 →v2=8, v2 →v4=6}

Calculate nearer vertex from f to the remaining vertices of the graph.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | V1 | V2 | V3 | V4 |
| Nearer place | ̶ | ̶ | V4 | ̶ |
| distance | ̶ | ̶ | 5 | ̶ |

Resultant selected vertices f={v1,v2,v4,v3}

Resultant selected corresponding Edges y={v1→v2=8,v2→v4=6,v3→v4=5}

Minimum Spanning Tree

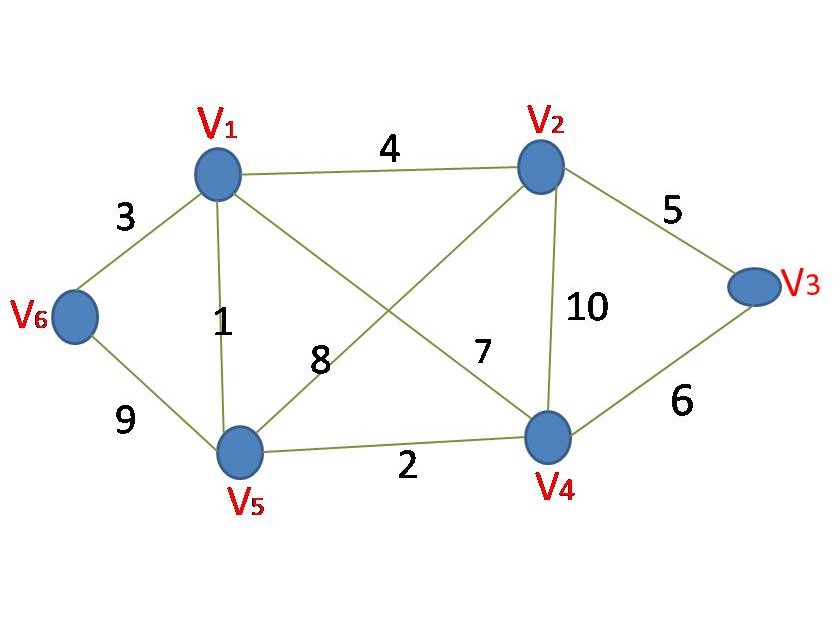
8

6

**Minimum Spanning Tree cost 19**

5

Second Graph

****

Adjacency Matrix

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | v1 | v2 | v3 | v4 | v5 | v6 |
| v1 | 0 | 4 | Ꚙ | 7 | 1 | 3 |
| v2 | 4 | 0 | 5 | 10 | 8 | Ꚙ |
| v3 | Ꚙ | 5 | 0 | 6 | Ꚙ | Ꚙ |
| v4 | 7 | 10 | 6 | 0 | 2 | Ꚙ |
| v5 | 1 | 8 | Ꚙ | 2 | 0 | 9 |
| v6 | 3 | Ꚙ | Ꚙ | Ꚙ | 9 | 0 |

Assumestarting place V1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | V1 | V2 | V3 | V4 | V5 | V6 |
| Nearer place | ̶ | V1 | V1 | V1 | V1 | V1 |
| distance | ̶ | 4 | Ꚙ | 7 | 1 | 3 |

Resultant selected vertices F={v1,v5}

Resultant selected corresponding Edge Y={v1→ v5=1 }

Calculate nearer vertex from f to the remaining vertices of the graph.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | V1 | V2 | V3 | V4 | V5 | V6 |
| Nearer place | ̶ | V1 | V1 | V5 | ̶ | V1 |
| distance | ̶ | 4 | Ꚙ | 2 | ̶ | 3 |

Resultant selected vertices F={v1,v4,v5}

Resultant selected corresponding Edges Y={v1→v5=1, v5→v4=2}

Calculate nearer vertex from f to the remaining vertices of the graph.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | V1 | V2 | V3 | V4 | V5 | V6 |
| Nearer place | ̶ | V1 | V4 | ̶ | ̶ | V1 |
| distance | ̶ | 4 | 6 | ̶ | ̶ | 3 |

Resultant selected vertices F={v1,v4,v5,v6}

Resultant selected corresponding Edges

Y={v1→v5=1 , v5→v4=2 , v1→v6=3 }

Calculate nearer vertex from f to the remaining vertices of the graph.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | V1 | V2 | V3 | V4 | V5 | V6 |
| Nearer place | ̶ | V1 | V4 | ̶ | ̶ | ̶ |
| distance | ̶ | 4 | 6 | ̶ | ̶ | ̶ |

Resultant selected vertices F={v1,v4,v5,v6,v2}

Resultant selected corresponding Edges

Y={v1→v5=1 , v5→v4=2 , v1→v6=3 , v1→ v2=4 }

Calculate nearer vertex from f to the remaining vertices of the graph.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | V1 | V2 | V3 | V4 | V5 | V6 |
| Nearer place | ̶ | ̶ | V2 | ̶ | ̶ | ̶ |
| distance | ̶ | ̶ | 5 | ̶ | ̶ | ̶ |

Resultant selected vertices F={v1,v4,v5,v6,v2,v3}

Resultant selected corresponding Edges

Y={v1→v5=1 , v5→v4=2 , v1→v6=3 , v1→ v2=4 , v2→v3=5 }

Minimum Spanning Tree

4

3

5

1

2

**Minimum Spanning Tree cost 15**

Two Different problems are Considered First program is with four Nodes and second problem is with six nodes. The results obtained on each stage for each example problem are tabulated. The obtained minimum spanning tree for each problem is drawn.

**Conclusion**

Prim’s Algorithm is generating minimum cost spanning tree. It can be used for water connection line problem and electric connection circuit.