**CHAPTER ONE: INTRODUCTION**

**1.0 Introduction**

A minimum spanning tree (MST) or minimum weight spanning tree is a subset of the edges of a connected, edge-weighted undirected graph that connects all the vertices together, without any cycles and with the minimum possible total edge weight. MST problem is of high importance in network optimization, but it is also difficult for the traditional network optimization technique to deal with. A good example would be a road construction company charged to construct streets to connect houses in a particular city, constructing the roads (vertices) requires a graph containing the houses (nodes) to be connected by the road. Some of the roads might be too expensive, either because they are longer, or the topography of the land (a valley or mountain), these roads would be represented with larger weight. Having in mind the cost of constructing these roads, larger weights on roads implies higher costs, a spanning tree for this graph would be as subset of those roads that has no cycles but still connects every house; there might be several spanning trees possible. A minimum spanning tree would be one with the lowest total cost, representing the least expensive path for constructing the road.

This is exactly what this project is set to achieve, by developing a fast and reliable computer program that could compute at a great level a cost-effective MTS of any undirected graph network supplied by a user.

* 1. **Statement of the problem**

This system intends to solve the following problems:

1. Cost: Cost is of great essence in every endeavor, since the MST of a network of graph is a subset of the graph, this means that the cost of implementing the graph is reduced.
2. Time: The time taken to implement the MST of a graph G is way less than the time needed to implement that of the original graph.
3. Speed: In a graph of network concerned with data dissemination, speed is of essence, since it takes lesser time to go through all nodes in a minimized network, which makes the dissemination of information faster throughout the nodes.
4. Man Power: Given a network of graph G of a road network, a road construction company requires a specific number of workers to work on the road, this number of workers and the number of hours each works depends on the size of the region to be worked on, if the graph reduces, the number of men and working time reduces also.
   1. **Aim and Objectives of the Study**

The aim of this project is:

To develop A Cost-Effective Network Minimizer.

The objectives of this project are:

The application should be able:

1. To accept any given network (edges and nodes) from the user as input
2. To transform the network to the barest cost-effective form
3. To display the reduced and transformed network

**1.3 Significance of the Problem**

There is some many use of this project to humanity, some of which are:

1. Road Construction: As stated in the introduction, MST can used to compute the minimal and most cost-effective path through which the road should be constructed.
2. Telecommunication Companies: Suppose MTN is laying cable in a new city (e.g. Enugu). If it is constrained to bury the cable only along certain paths (e.g. roads), then there would be a graph containing the points (e.g. villages) connected by those paths. Some of the paths might be more expensive, due to too much mountains or valley which will require more man power or the cable be buried deeper respectively, these paths will cost much since there will have higher weights. MST will help reduce the graph to produce path that has no cycles but still connects all the villages, which will in turn reduce the number of resources and man power needed to lay the cables and also reduce the cost of laying them.
3. Oil and Water Transportation: Finding the minimal path to transport crude oil from wells to refinery will minimize cost. Also, a cost effect

**1.4 Scope of the Project**

The scope of this project is limited to undirected graphs only.

**CHAPTER TWO: LITERATURE REVIEW**

**2.0 Introduction**

The minimal spanning tree problem (MSTP) is a notable problem of combinatorial optimization. It deals with the problem of obtaining a tree of minimum weight that spans all the vertices of a weighted, undirected and connected graph, where the weight of the tree corresponds to the sum of weights of its edges. It is widely applied in various fields of science and technology ranging from computer and communication networks, knowledge engineering, wiring connections, VLSI circuits design to a large class of optimization problems. Recent approaches in analyzing various biomedical problems like medical imaging, bio-terrorism, etc have made an extensive use of the concepts of minimal spanning tree (MST). In fact, recent advances in clustering algorithms also deploy the concepts of MST [1].

This chapter discuss the theoretical background and the literature review of the project work. The theoretical background discusses the programming language(s) used in writing the project and also explaining reasons they were used. The literature review discusses the contributions of many scholars who have written articles, books etc about MST.

**2.1 Theorical Background**

The major technologies used in this project is HTML, CSS, JavaScript, NodeJs and MongoDb. The program is presented using browsers such as Firefox, Opera, Google Chrome etc. The design part of the program is written using Hyper-Text Markup Language (HTML) and Cascading Style Sheet (CSS). HTML is primarily used to create the frame work of the web-based program.

CSS is a style sheet language used for describing the presentation of a document written in a markup language like HTML, it is designed to enable the separation of presentation and content, including layout, colors, and fonts.

JavaScript often abbreviated as JS, is a high-level, interpreted programming language that confirms to the ECMAScript specification. It is a programming language that is characterized as dynamic, weakly typed, prototype-based and multi-paradigm. JavaScript is one of the core technologies of the World Wide Web (WWW), it enables interactive web pages and is an essential part of web applications.

NodeJs is an open-source, cross-platform JavaScript run-time environment that executes JavaScript code outside the browser. JavaScript is used primarily for client-side scripting, in which scripts written in JavaScript are embedded in a webpage's HTML and run client-side by a JavaScript engine in the user's web browser. Node.js lets developers use JavaScript to write command line tools and for server-side scripting—running scripts server-side to produce dynamic web page content before the page is sent to the user's web browser. Consequently, Node.js represents a "JavaScript everywhere" paradigm, unifying web application development around a single programming language, rather than different languages for server side and client-side scripts.

MongoDB is a cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with schemata. It stores data record as BSON documents. BSON is a binary representation of JSON documents, though it contains more data type that JSON. MongoDB documents are composed of field-and value pairs and hace the following structure:

{

field1: value1

field2: value2

field3: value3

…

fieldN: valueN

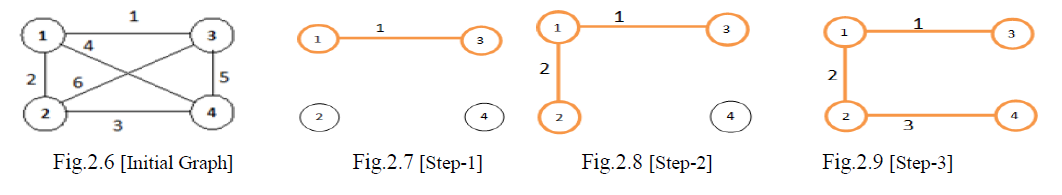
}

The value of a field can be any if the BSON data types, including other documents, arrays and arrays of documents.

**2.2 Review of Related Literature**

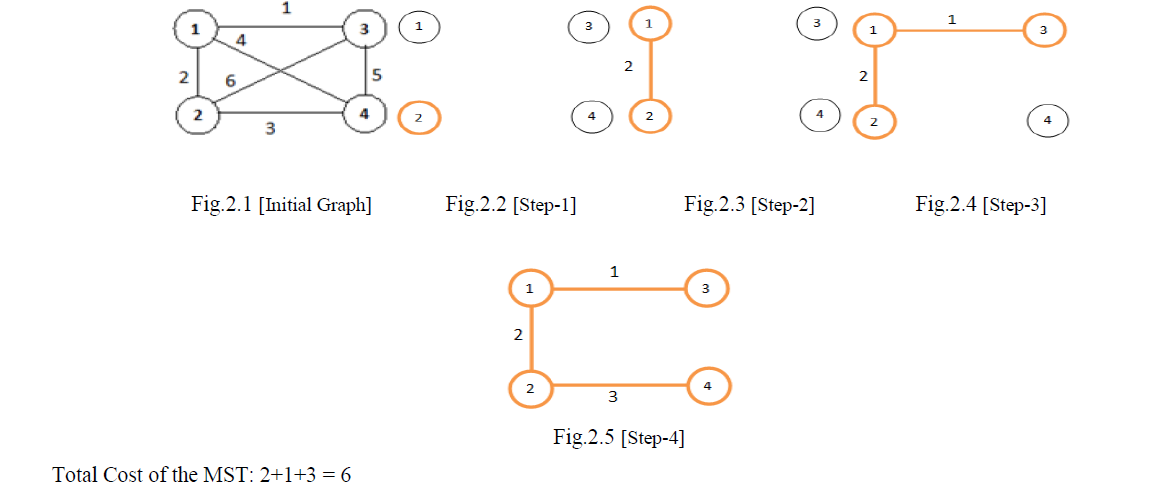
A lot of researches have been done on designing optimal network for purposes of distribution, communication, machine scheduling, gas and pipelines, etc. The tool commonly used for the optimal design of such networks is the Minimal Spanning Tree. A Minimal Spanning Tree problem is one of the most fundamental and intensively studied problems in network optimization problem with many theoretical and practical applications [2], [3], [4]. [5] used minimum spanning tree to design optimal offshore natural gas pipeline systems.

A number of algorithms exist for the determination of Minimum Spanning Tree; these include Kruskal algorithm, Prim’s algorithm and Boruvka algorithm, [6]. In [7], the author described Kruskal’s algorithm as one that allows all the edges to be arranged in increasing order of weight. These edges are placed in a priority queue. Then minimum weighted edge from the queue is selected and placed in to one new forest. This process is repeated until n-1 edges are selected. If addition of any edge in the graph creates cycle then this edge will be rejected.



Total Cost of the MST: 1+2+3 = 6

He also described Prim’s Algorithm as one which the starting node is selected randomly from the given graph, and a set that contains all the edges in the graph is created. Any edge that is adjacent to the randomly selected edge and has the minimum weight is selected from the set and add to new graph if it connects a vertex in the tree with a vertex not in the tree. This process is repeated until every edge in the set connects two vertices in the tree.



In [8], the authors gave a pseudocode for Prim’s algorithm which involves the following steps:

**Step 0**: set C = ø and C`= {1, 2, …, m}

**Step 1**: Start with any node, say node i, in 0 C and connect it to node j that is closest to node i. Set C = {i, j} and C` = {1, 2, …, m} \ {i, j}.

**Step 2**: Now choose node l in C’ that is closest to some node k in C. Then connects node k to node l. Set C = {i, j, k} and C’ = C’\C

**Step 3**: Repeat step 2 until C’ = ø. Ties for the closest node and arc to be included in the minimum spanning tree may be broken arbitrarily.

In [9], the authors introduced a pseudocode to calculate the minimum spanning tree of a graph using Matrix Algorithm.

Let, G = (V, E) be an undirected connected weighted graph with n vertices, where V is the set of vertices, E is the set of edges and W be the set of weights (cost) associated to respective edges of the graph. Where eijthe edge adjacent to vertices viand vj. wij= the weight associated to the edge eij. The Weight Matrix M of the graph G is constructed as follows: If there is an edge between the vertices vito vjin G then Set, M[i,j] = wij Else Set, M[i,j] = 0

Algorithm:

**Input**: the weight matrix M = [wij] n × n for the undirected weighted graph G

**Output**: Minimum Cost Spanning Tree T of G.

**Step 1**: Start

**Step 2:** Repeat Step 3 to Step 4 until all (n-1) elements matrix of M are either marked or set to zero or in other words all the nonzero elements are marked

**Step 3**: Search the weight matrix M either column-wise or row-wise to find the unmarked nonzero minimum element M[i,j], which is the weight of the corresponding edge eij in M.

**Step 4**: If the corresponding edge eij of selected M[i,j] forms cycle with the already marked elements in the elements of the M then Set M[i,j] = 0 Else Mark M[i,j]

**Step 5**: Construct the graph T including only the marked elements from the weight matrix M which shall be the desired Minimum cost spanning tree of G.

The above listed algorithms has their advantages and disadvantages, time complexity and space complexity, this paper is not to address this but to build a computer program that uses this algorithms (Prim’s and Kruskal’s) to determine the MST of any given non-directed weighted graph.