A comparative study of Kruskal's Algorithm and Prim's Algorithm

Era Islam and Faria Zaman
Department of Computer Science and Engineering
Rajshahi University of Engineering and Technology

***Abstract*—Both Prim's algorithm and Kruskal's algorithm are greedy algoriithms for finding minimum spanning tree. Prim's algorithm can be used on connected, weighted and undirected graph [1], whereas Kruskal's algorithm can be used on weighted, undirected and both connected and disconnected graph [2]. Previously, Prim's algorithm was found to run faster in dense graphs with more number of edges than vertices, whereas Kruskal's algorithm was found to run faster in sparse graphs [3]. The answer to the question which one is superior is not definite. In this paper, we are proposing a new prediction where Kruskal's algorithm is faster than Prim's algorithm in case of time complexity for finding minimun spanning tree. Next an experiment is conducted in which the order of the starting graph is changed and each algorithm's runtime is then measured. From the result it can be shown that Kruskal's algorithm is faster than Prim's algorithm.**

I. Introduction

Graph theory is the study of graphs, where mathematical structures are used to model pairwise relations between objects in mathematics. The first paper in the history of graph theory written by Leonhard Euler on the Seven Bridges of Königsberg was published in 1736 [4]. Euler's formula relating the number of edges, vertices, and faces of a convex polyhedron was studied and generalized by Cauchy and L'Huillier, and came to the conclusion that no solution exists and this consequently laid the foundations to the study of graphs.A spanning tree of a graph is just a subgraph that contains all the vertices of that graph and it is also a tree [5]. A graph may have many spanning trees within it.

Previously, an experiment was performed to compare Prim's algorithm and Kruskal's algorithm on Shanghai and Shenzhen 300 Index Hierarchical Structure Tree. The algorithms were used to find the minimum spanning tree in building up super metric space. After analyzing the sample data of Shanghai and Shenzhen 300 Index Hierarchical Structure Tree, it was shown that when the number of shares is less than 100, in case of space complexity, Kruskal's algorithm is more efficient than Prim's algorithm. However, when the number of shares is greater than 100, in case of time complexity, Prim's algorithm is more efficient than Kruskal's algorithm [6] The result came from the study that matrix of Shanghai and Shenzhen 300 stock the inter-continental distance formed by the edge map more, so Prim better than Kruskal. However, in case of time complexity, they did not claim the representation to work out when the number of shares is less than 100.

II. Description of the algorithms

Algorithm is a set of self-contained operations in Mathematics and Computer Science which is performed step by step [7]. It performs multiples tasks like calculation, data processing, automated reasoning tasks.

A. Graph

A graph is defined as a set of points, which are called vertices, some or all of which are connected with a set of lines, or edges.
A graph G is denoted as follows:

$G = \{V, E\}$

Where V is the set of vertices and E is the set of edges. Often vertices are also called nodes. A graph is called undirected when there is no distinction between the two vertices associated with each edge, or it is called directed when there is direction from one vertex to another [4]. A graph is connected when there is a path between every of vertices. A graph is called disconnected where it is not connected [8]. There are many forms of graphs, which may be connected in different ways, in this study we'll use a special class of graphs, called trees. A tree is an undirected graph where any two vertices are connected by exactly one edge, we can say that any acyclic connected graph is a tree, where a forest is a disjoint union of trees [9]. 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 [5]. In Fig. 1. the minimum spanning of the graph is shown by the bold lines.

B. Prim's Algorithm

An algorithm is a procedure for solving a problem, based on conducting a sequence of specified actions. A computer program can be viewed as an elaborate algorithm. In mathematics and computer science, an algorithm usually means a small procedure that solves a recurrent problem. Algorithms are widely used throughout all areas of IT (information technology) [10].

In computer science, Prim's algorithm is a greedy algorithm,it finds a minimum spanning tree for a weighted undirected graph [1]. But it can not find a minimum spanning tree in disconnected graph. That means it does not work in a forest. However, Prim's algorithm can run separately for each connected component in a forest to find minimum spanning tree.
Prim' s algorithm starts at any vertex of a graph. This works, because all of the vertices are part of the minimum spanning tree, but it is not necessary for all the edges to be part of it. At first, we choose the edge with the smallest weight that is not chosen before and add it to the minimum spanning tree. Then we consider the adjacent edges. Again the edge with the smallest weight is chosen. If there are two edges of the same weight, we can choose either one of them. The step is followed repeatedly until all of the vertices of the graphs in part of the minimum spanning tree. The running time of prim's algorithm using an adjacency matrix or an adjacency list is O(|V |2) to find the array of weights of the minimum spanning tree. However, this running time can be improved using heaps in the algorithms in a loop [1]. In Fig. 2. the minimum spanning tree by using Prim's Algorithm is shown.

C. Kruskal's Algorithm

Kruskal's algorithm starts with the edge with the smallest weight which is the first part of the spanning tree. Then the next smallest edge will be selected. If it does not complete a cycle we add it to set of edges of the minimum spanning tree, otherwise we discard it. Thus, we keep selecting new edges this way until we have n – 1 successfully chosen edges to get a minimum spanning tree. Kruskal's algorithm generates a minimum-cost spanning tree for every undirected graph.
The computing time of Kruskal Algorithm is O(|E |log |E |), where E is the edge set of G, which is the worst case [12]. As, the edges have to be sorted first and it takes O(|E |log |E |) where it dominates the runtime for verifying whether the edge in consideration is a safe edge or not which would take O(|E |log |V |). In Fig. 3. the minimum spanning tree by Kruskal's Algorithm is shown.

There are many useful ways represent graphs. One way to represent a graph without multiple edges is to list all the edges of the graph and the other way to represent a graph with no multiple edges is to use adjacency list, which specify the vertices to each vertex of the graph.
To simplify computation, graphs can be represented using matrices. Two types of matrices are commonly used to represent graphs. One is based on the adjacency of vertices, and the other is based on incidence of vertices and edges.
Suppose that, G=(V,E) is a simple graph where V is set of the vertices and E is the set of the edges. The adjacency matrix of G, with respect to the listing of the vertices, is the nxn zero-one matrix with 1 as its (i,j) the entry when I and j are adjacent and 0 as its (i,j)th entry when they are not adjacent. The adjacent matrix for a directed graph does not have to be symmetric, because there may not be an edge from ai to aj, when there is edge from ai to aj.
When a simple graph contains relatively few edges, that is sparse, is usually preferable to use adjacency matrix to represent the graph. A simple graph is dense, when it contains many edges, such as a graph that contains more than half of all possible edges [13]. Prim's algorithm is significantly faster in the limit when there is a dense graph with many more edges than vertices. Kruskal performs better in typical situations(sparse graphs) because it uses simpler data structures [10].

D. New Approach

In our experiment, we took some random graphs and applied Prim's algorithm and Kruskal's algorithm. We measured the running time gradually increasing the vertices of the graphs.
Here we implemented the algorithms in programming language C. We used rand() function to determine the weight of every edge of the graphs. We took the range of the rand() function 50, from 0 to 49.We declared the graph using adjacency matrix. In the end we calculated the running time of the program. Then we showed the result in graph representation.
Our expected benefit from this approach was to prove a algorithm superior than another algorithm. From this approach we found that Kruskal's algorithm is efficient than Prim's Algorithm.
In some cases our program returns value where Prim's algorithm takes less time than Kruskal's algorithm, which is rare. This happens because of language, editor, system etc which is a limitation of our experiment.

III. Experimental Procedure

The computer in which we did the experiment has operating system Windows 7, Intel(R) Core(TM) i5-5200U @2.20GHz Processor, Ram 4GB, Cache Memory 3MB, and the programming language was C, Editor was CodeBlocks.

We started our experiment using 10 vertices. Gradually we increased the vertices to 125. We did 3 experiments. For every experiment we took the data 3 times and noted the running time, then we calculated the average of them. In the end we draw 3 graphs for vertices versus time in seconds.
The benefit of this approach is we can compare Prim's and Kruskal's algorithm for random graphs. The graphs show the comparison between the algorithms.
Previously, Prim's algorithm and Kruskal's algorithm were compared on Shanghai and Shenzhen 300 Index Hierarchical Structure Tree. The algorithms were used to find the minimum spanning tree in building up

super metric space. We chose this approach because here they declared Prim's algorithm to be superior when number of shares are greater than 100.
Here we took the vertices number 10, 25, 50, 75, 100 and 125. We calculated the running time for Prim's algorithm and Kruskal's algorithm increasing the vertices to compare them for different number of vertices.

We took different number of vertices to do the experiment and measured the running time.
Number of vertices were taken in X-axis and Time in Seconds were taken in Y-axis. In one graph we compared two algorithms drawing there vertices versus time line.

TABLE I
FIRST EXPERIMENT

n	10	25	50	75	100	125
Kruskal	1.67	1.67	2.33	3.33	4.33	4.67
Prim	1.67	2.33	3.33	3.67	5.33	6.67

TABLE II
SECOND EXPERIMENT

n	10	25	50	75	100	125	150
Kruskal	1	1.33	2	2.67	3	3.67	5.67
Prim	2	2	3.33	4	7	6	7.67

TABLE III
THIRD EXPERIMENT

n	10	25	50	75	100	125
Kruskal	1	1.67	1.67	3	3.67	4.67
Prim	1.33	2	2.67	3.67	4.67	6

IV. CONCLUSION

The algorithms were implemented in programming language C. Graphs were represented by adjacency matrix and weights were assigned by rand() function to generate random graphs. At the end the running time of the algorithms were measured. From the result it is shown that, for random graphs, Kruskal's algorithm is efficient than Prim's algorithm for any number of vertices. Using this approach we can compare between the running time of Prim's algorithm and Kruskal's algorithm.

REFERENCES

[1] https://en.wikipedia.org/wiki/Prim
[2] https://en.wikipedia.org/wiki/Kruskal
[3] https://www.quora.com/What-is-the-difference-in-Kruskals-and-Prims-algorithm
[4] https://en.wikipedia.org/wiki/Graph
[5] https://www.ics.uci.edu/ eppstein/161/960206.html
[6] http://ieeexplore.ieee.org/document/5369459/?reload=true
[7] https://en.wikipedia.org/wiki/Algorithm
[8] http://mathworld.wolfram.com/ConnectedGraph.html
[9] http://www.personal.kent.edu/ rmuhamma/GraphTheory/MyGraphTheory/trees.htm
[10] https://www.quora.com/What-is-the-difference-in-Kruskals-and-Prims-algorithm
[11] http://www.cprogramming.com/tutorial/computersciencetheory/mst.html
[12] Horowitz, Sahni, Rajasekaran.2008.Algorithms:Fundamentals of Computer.Universities Press (India) Private Limited, India.773(pp).
[13] Rosen.2007.DISCRETE MATHEMATICS AND ITS APPLICATION:WITH COMBINATORICS AND GRAPH THEORY.McGraw Hill Education(India) Private Limited,India.839(pp).