International Journal of Engineering Research and Technology. ISSN 0974-3154, Volume 13, Number 12 (2020), pp. 5386-5391


# Comparative Performance Analysis of Kruskal and Prim MST Algorithms

**Peace O. Ayegba[1], Aderemi E. Okeyinka[1], Marion Adebiyi[2], Emmanuel O. Asani[1], Joyce A. Ayoola[1], Goodness C. Ben[1]**

[1] *Department of Computer Science, Landmark University, Nigeria.*

[2] *DICT and Society Research Group, South Africa Luban Workshop, Durban University of Technology, P.O Box 1334, Durban 4000, South Africa.*

*ORCID: 0000-0002-0830-7811 (Peace O. Ayegba)*

**Abstract**

With the increased growth of large-scale networks and advancements in GIS applications, the use of the minimal spanning trees has become equally important. The minimal spanning tree for a given graph is a subgraph whose weight is minimal, and can be obtained using classical algorithms such and Kruskal and Prim's algorithm. Several studies have focused on improvements made on either Prims or Kruskal algorithm but very few have analyzed the complexities of these algorithms. In this study, we investigated the computational speed complexities of both algorithms to provide an informed report on which MST algorithm is computationally superior. Road network datasets comprising 82, 309, 1899, 4039, 6105 and 21048 nodes in succession have been used to simulate both algorithms, and results showed that Kruskal's algorithm performed better than Prim's algorithm in terms of computational speed.

**Keywords:** Minimal spanning trees, Kruskal, Prim, Performance Analysis, Complexity

## I. INTRODUCTION

Algorithmic complexity is concerned with how fast or slow algorithms perform. Complexity analysis is a tool that allows us to explain how an algorithm behaves as the input grows larger. It also helps us analyze the behavior of an algorithm when the input changes. In order to analyze the complexities of an algorithm, the algorithm must be implemented and then its results analyzed. The minimal spanning tree (MST) is a well-known combinatorial optimization problem that has significant applications in graph theory, with diverse algorithms proposed to solve it. A minimal spanning tree is a subgraph of a graph whose weights are minimal. Kruskal and Prim algorithms are classical algorithms used to provide the optimal MST; Optimal, based on time or the cost of paths, depending on which is required. For a graph with $v$ vertices and $e$ edges, both algorithms run in $O(e \log n)$ time, although, Prim's relies on a priority queue that supports priority modification while Kruskal's relies on a disjoint-set that supports find/union operations [1]. Suppose a hospital ambulance is to deliver pharmaceutical drugs to all hospitals in vicinity and the driver must take these drugs to every hospital exactly once during the trip. He/she must start from a hospital and end up in a destination with a minimum total distance covered. In order to determine the fastest route between hospitals, a minimal spanning tree is required. Selecting the best route from a number of alternatives is a problem often encountered. Such a problem can be modelled as a graph problem. Kruskal and Prim's algorithm are popular algorithms for constructing minimal spanning trees and they have both been used and modified extensively [2,3,4,5,6,7]. The minimum spanning tree problem arises in many applications, such as in clustering problems, image processing and road network transportation problems [8]. A very unique illustration of MST's is greedy algorithms, which always deliver an optimal solution, and use clever data structures where necessary to make it work efficiently. A less obvious application is that the minimum spanning tree can be used to approximately solve the traveling salesman problem.

In this study, Kruskal and Prim algorithms have been implemented on a road network graph of major hospitals in Ibadan, Oyo state, and the computational speed performance of both algorithms were documented. Different road network datasets have been used to simulate both algorithms to show the computational speed superiority of these algorithms.

## II. RELATED WORK

[2] compared the performance of Prim and Kruskal's algorithm on the Shanghai and Shenzhen 300 index daily stock exchange data between 2005 and 2007, that is, they compared the performance of both MST algorithms in constructing the super metric space of the sample stocks within this period. Comparison of the space complexity of both algorithms showed that Kruskal's algorithm was superior with an input data less than 100, whereas the time complexity of prims algorithm was superior to Kruskal's with an input data size greater than 100 but had little difference when a data size less than 100 was used. [9] designed an optimal route for the water supply system in the Kwahu South District of Ghana using Prim's algorithm. The map used consisted of 12 nodes and the distances between the towns were the weights of the graph. It was seen from the results that the total cost of pipeline construction was significantly reduced when the MST was used. [10] also applied the Prims algorithm in designing an optimized LAN network for Chuka University, Kenya. The algorithm was used to determine the shortest length of cables to directly link a set of network nodes supposing all buildings in the university are connected. The graph used consisted of 11 nodes and the distance between the buildings were the weights of the graph. The result showed that the Prims algorithm produced a route with minimal distance and therefore optimizes the installation

cost of the optic fiber in the LAN network. [11] also studied the use of MSTs based on optimization of time and cost for temporal graphs. The MST based on cost optimization for temporal graphs was treated as a Directed Steiner tree problem, that is, it was transformed into a static graph. Then, a linear-time algorithm for finding the optimal cost MST was introduced. Prim's algorithm for minimum spanning tree has been used by [12] in the transportation system of the Odeda local government in Nigeria. The graph used consisted of 88 nodes and 96 paths where the nodes and paths represent the villages and the connecting roads. It was concluded that Prim's algorithm proved effective in providing the shortest routes and in reducing the fuel cost for transportation in that LGA. Also, [13] performed a similar implementation but made use of Prim's algorithm in producing the minimum spanning tree of the 36 state capitals in Nigeria with Yenagoa as the starting node. They proposed the construction of the optimal road network discovered, in the design of telecommunication networks, transportation and even petroleum pipelines. Also, [14] proposed the use of neutrosophic numbers instead of a real/fuzzy number as weights on the graph in generating the minimal spanning tree. The proposed algorithm is a modification of Kruskal's algorithm and produces both the cost and MST of a neutrosophic graph. The algorithm was compared with existing algorithms and it was shown that the proposed scheme was simple and efficient and suitable for real-world supply chain and transportation problems. [15] developed a heuristic algorithm that combines reinforcement learning and graph embedding in solving optimization problems. The algorithm proposed, incrementally constructs a minimal spanning tree solution, acting like a greedy meta-algorithm. The deep learning architecture used, S*tructure2vec deep Q-learning*, was shown to be efficient and effective in learning greedy heuristics. Thus, the proposed method catered for difficulties that occur when applying conventional MST algorithms on large data sets.

## III. METHOD

The system design consists of five segments namely,

- Kruskal's algorithm module
- Prims algorithm module,
- The time function calculating the execution times
- Complexity analysis of both algorithms
- Comparison of resulting MSTs.

The data used for this analysis, consisted of 82 nodes representing major hospitals from all regions in Ibadan. The services provided by each hospital and their infrastructure was also included in the dataset. This dataset was used to analyze the complexities of the Kruskal and Prims algorithm using the same programming environment and conditions. Furthermore, to show the varying time complexities of the algorithms, road network datasets gotten from SNAP (Stanford Network Analysis Project) were used.

### III.I Kruskal's Algorithm

Kruskal's algorithm uses the greedy approach for finding a minimum spanning tree. Kruskal's algorithm treats the graph as a forest in which every node as an independent tree and connects one with another only if it has the lowest cost compared to all other options available. It is known as a greedy algorithm in graph theory as it finds a minimum spanning tree for a connected weighted graph adding increasing cost arcs at each step. This means it finds a subset of the edges that forms a tree that includes every vertex, where the total weight of all the edges in the tree is minimized. If the graph is not connected, then it finds a minimum spanning forest.

It works as follows:

- Sort the graph edges with respect to their weights.
- Start adding edges to the minimum spanning tree from the edge with the smallest weight until the edge of the largest weight.
- Only add edges which don't form a cycle—edges which connect only disconnected components

The flowchart depicting the steps for Kruskal's algorithm is shown in figure 1.

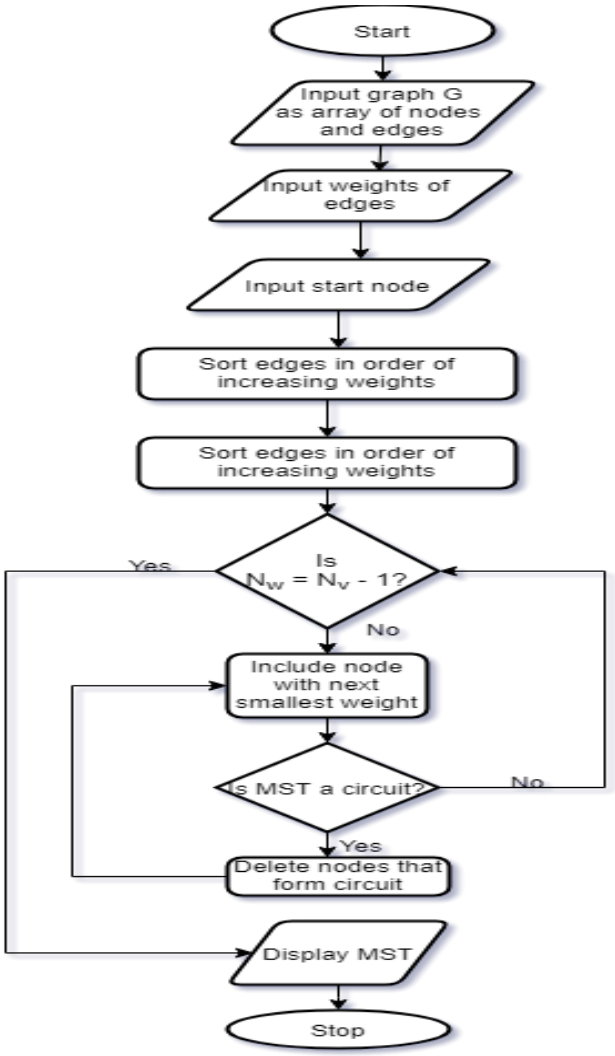


**Fig. 1.** Flowchart of Kruskal's algorithm

### III.II Prim's Algorithm

Prim's calculation is a calculation in diagram hypothesis that finds a base spreading over tree for an associated weighted chart. The calculation finds a subset of the edges that shapes a tree that incorporates each vertex where the complete load of the considerable number of edges in the tree is limited. On the off chance that the diagram isn't associated, at that point it will just locate a base traversing tree for one of the associated parts. The calculation was found in 1930 by mathematician Vojtech Jarnik and later freely by PC researcher Robert Prim and rediscovered by Dijkstra. In this manner, it is some of the time called DJP calculation or Jarnik calculation. It functions as follows;

- Create a tree containing a solitary vertex, picked self-assertively from the diagram
- Create a set containing all the edges in the chart
- Loop until each edge in the set interfaces two vertices in the tree
- Remove from the set and edge with least weight that interfaces a vertex in the tree with the vertex not in the tree.

The flowchart depicting Prim's algorithm is shown in Figure 2.

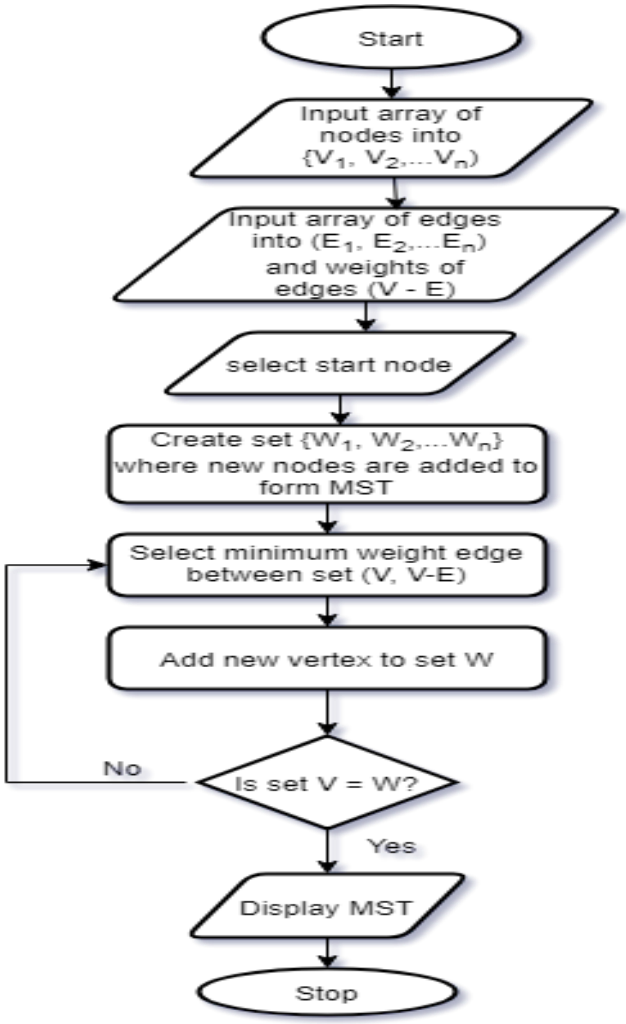


**Fig. 2.** Flowchart of Prim's algorithm

### III.III System Design

In the design for the two algorithms, the following steps are carried out:

- A text file containing the nodes, edges and weights of the graph is created first.
- When the program is executed, the contents of the text file is read character by character, and the Kruskal or Prim function is initiated.
- The time function on the algorithm starts also.
- After execution, an output showing the Graph and MST is displayed; The time of execution and minimum distance produced for each algorithm is shown also.
- To simulate the speed performance of both algorithms, 6 different road network datasets, containing 9, 36, 81, 6105 and 21,048 nodes respectively, will be used.

## IV. IMPLEMENTATION

In implementing and simulating both algorithms, the following tools were used.

- Python 3.0 programming language running in PyCharm IDE (Integrated development environment).
- 82 node network graph representing major hospitals in Ibadan, Oyo state, Nigeria.
- Road network dataset gotten from SNAP (Stanford network analysis project) database to further simulate both algorithms (Kliment et al., 2014).

### IV.I 82-Node Dataset

Figure 3 shows a screenshot of hospitals and their Node Ids from Node 0 to 81. There are 91 edges, representing roads, connecting the nodes and respective weights/distances in Km. Figures 4, 5 and 6 show screenshots of the implemented Kruskal and prims algorithm in PyCharm IDE.

| Node ID | Hospital | Services/ Infrastructure |
|---|---|---|
| 0 | University Health Center, Jaja | Ambulance, Antenatal, Dentistry, Family planning, General Medicine, General Surgery, Immunization, In-Patient, Laboratory Services, Occupation health services, Optometry, Paediatrics, Pharmacy, Psychiatry, Psychotheraphy, Ultrasound |
| 1 | UCH Annex, Ikolaba | Antenatal, Family planning, General Medicine, Immunization, Obstetrics and Gynaecology |
| 2 | University College Hospital UCH | Accident and Emergency, Ambulance, Antenatal, Blood Banking, Blood Transfusion, Cardiology, Dentistry, Dermatology, Dietetics, ECG, Endocrinology, ENT clinic, Family planning, Gastroenterology, General Medicine, General Surgery, Haematology, HIV/AIDS Screening, Immunization, In-Patient, Laboratory Services, Morgue, Nephrology, Neuro-Surgery, Obstetrics and Gynaecology, Occupation health services, Oncology (Tumour treatment), Optometry, Orthopaedics, Paediatrics, Pathology, Pharmacy, Psychiatry, Psychotheraphy, Radiotheraphy, rheumatology, Telemedicine, Training, Ultrasound, X-Ray Services |
| 3 | St Gregory Specialict Diagnostic clinic, Yemetu | Antenatal, General Medicine, General Surgery, Ultrasound |
| 4 | Highland Specialist Hospital, Yemetu | General Medicine, General Surgery, In-Patient |

**Fig. 3.** Hospitals and Node Ids

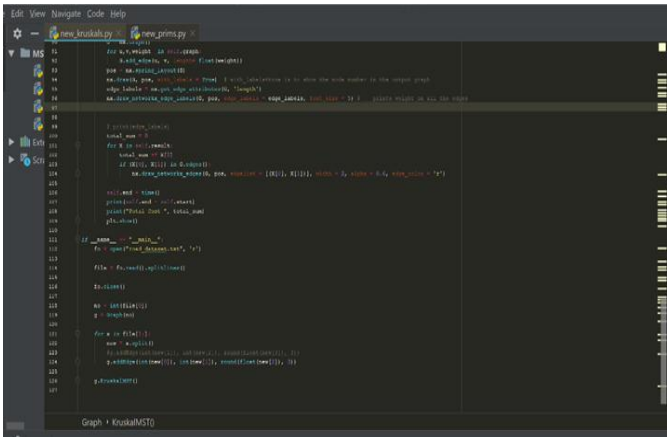

**Fig. 4.** Kruskal and Prims algorithm in development environment

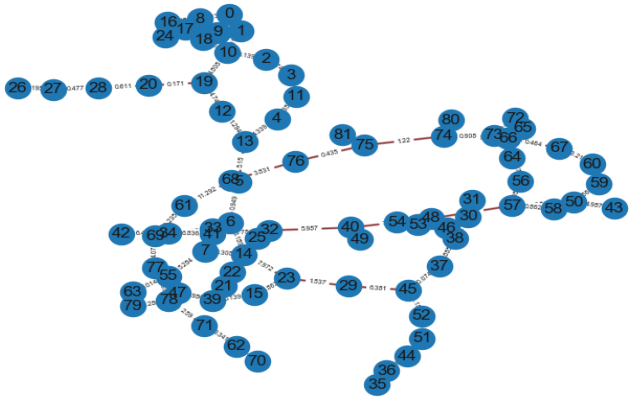

**Fig. 5.** Minimal spanning tree produced by Kruskal's algorithm for 82 nodes

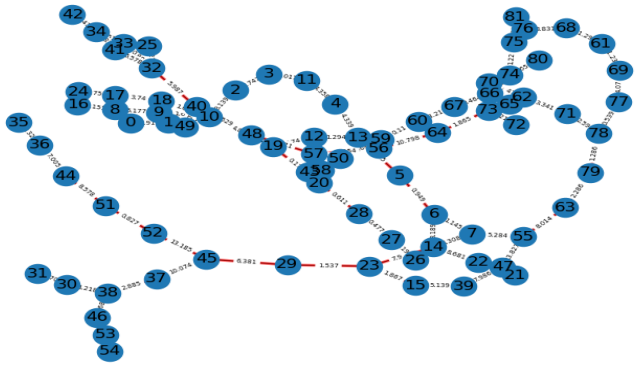

**Fig. 6.** Minimal spanning tree produced by Prim's algorithm for 82 nodes

The total weight of the spanning trees produced by both algorithms for the 82-node dataset was **298.447km.** However, the time taken for Kruskal's algorithm was 234.8 milliseconds while Prim's algorithm took 249.9 milliseconds. This shows that Kruskal's algorithm performed better in terms of speed when compared to Prim's algorithm.

### IV.II Data analysis with varying Dataset

Road network datasets containing 9, 36, 82, 309, 1899, 4039, 6105 and 21048 nodes were used in analyzing the time complexities of both algorithms. Figures 7, 8, 9 and 10 show the MST's produced on a 9-node and 36-node graph for both Kruskal and Prim's algorithm. The lines marked red show the spanning tree.

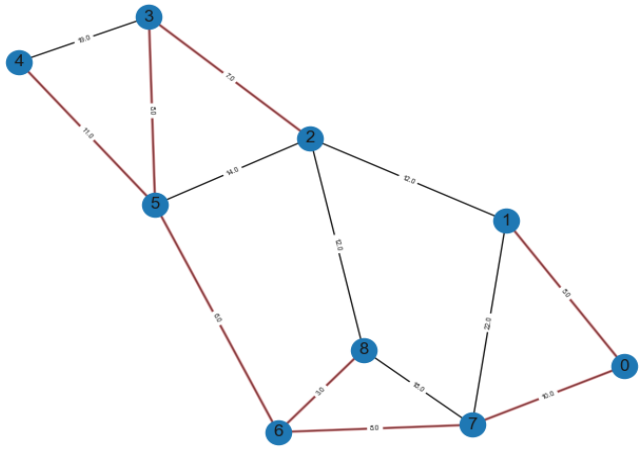

**Fig. 7.** Kruskal's MST for 9-node graph.

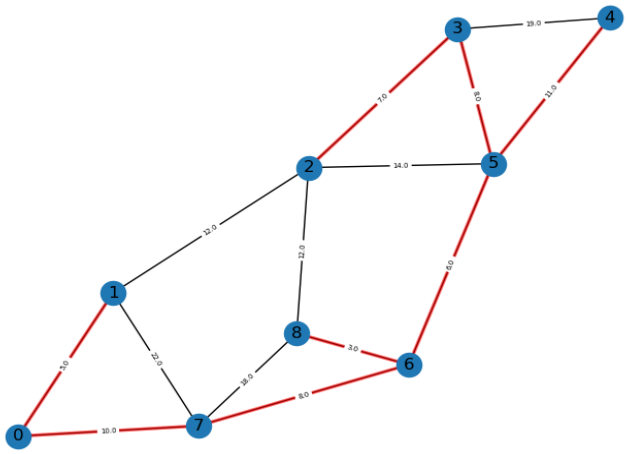

**Fig. 8.** Prim's MST for 9-node graph.

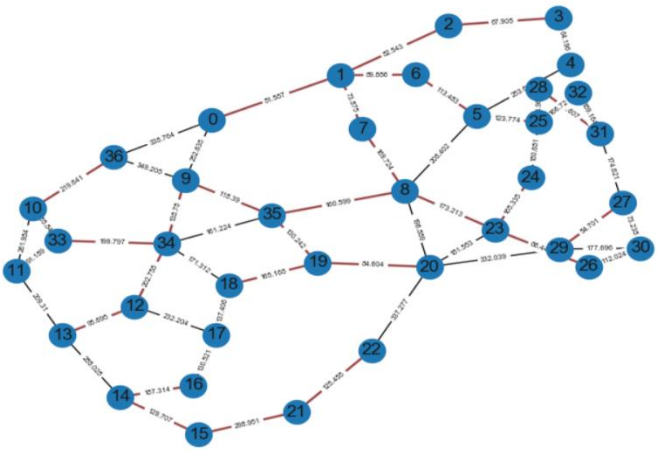

**Fig. 9.** Kruskal's MST for 36-node graph.

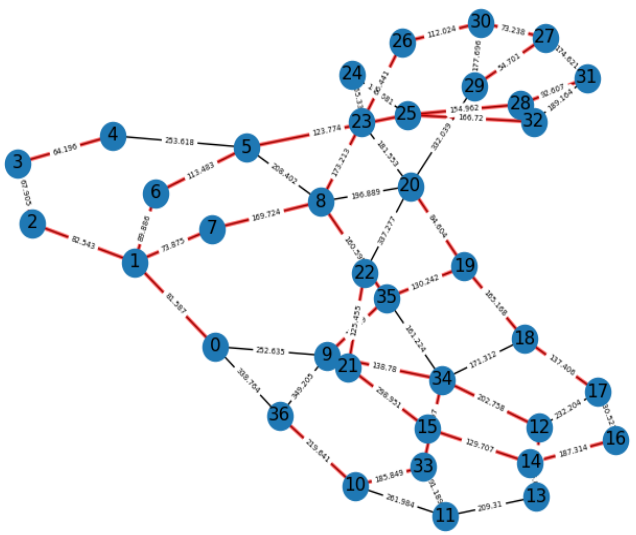

**Fig. 10.** Prim's MST for 36-node graph.

### IV.III Results

**Table 1.** Time and cost results for both algorithms

| | **Kruskal** | **Prim** |
|---|---|---|
| 9 nodes<br>• Time (milliseconds)<br>• Cost (km) | 174.9<br>58 | 174.9<br>58 |
| 37 nodes<br>• Time (milliseconds)<br>• Cost (km) | 194.8<br>4728 | 209<br>4728 |
| 82 nodes<br>• Time (milliseconds)<br>• Cost (km) | 234.8<br>298.447 | 249.9<br>298.447 |
| 309 nodes<br>• Time (milliseconds)<br>• Cost (km) | 587.6<br>7860.4 | 610.5<br>7860.3 |
| 1899 nodes<br>• Time (milliseconds)<br>• Cost (km) | 62567<br>879.1 | 67345<br>879.1 |
| 4039 nodes<br>• Time (milliseconds)<br>• Cost (km) | 105043.4<br>311156.36 | 145604<br>311156.36 |
| 6105 nodes<br>• Time (milliseconds)<br>• Cost (km) | 121523<br>378728.83 | 189437.7<br>378728.83 |
| 21048 nodes<br>• Time (milliseconds)<br>• Cost (km) | 1138750<br>307.69199 | 1873320<br>307.69199 |

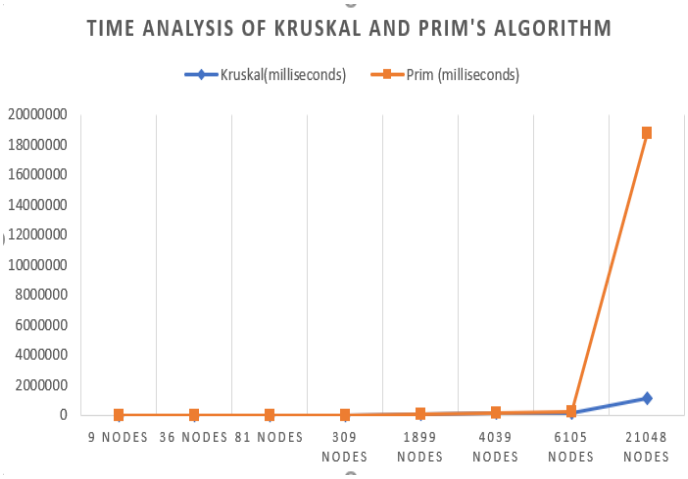


**Fig. 11.** Time analysis for both algorithms.

### IV.IV Findings and Summary

In bid to analyze the complexities of Kruskal and Prims algorithm, this study has obtained the minimum spanning tree of 82 major hospitals in Ibadan, Oyo state, using both algorithms. Access to hospital services is crucial for human survival and development, and therefore minimizing the distance and therefore travel time between these hospitals is equally important. However, the travel time may vary based on the means of transportation used (Vehicle, Motorcycle or Foot).

To further simulate the time complexities, road network dataset with different number of nodes was used. From the results shown in Table 1, Kruskal and Prims algorithm produced minimal spanning trees of the same cost for all sample datasets used. However, the speed analysis shows that for smaller graphs (9-node graph), Kruskal and Prim computed the MSTs at the same time, but as the number of nodes are increased gradually, Kruskal's algorithm performed better in terms of speed. Using a 37-node dataset, the time of execution for Kruskal's algorithm was 194.8 milliseconds, whereas Prim's algorithm completed its execution in 209 milliseconds. However, the difference in execution times become more obvious when the number of nodes is increased to more than 6000 nodes, with the execution times as 121523 milliseconds and 189437 milliseconds for Kruskal and Prim's algorithm respectively.

Figure 11 shows the time analysis on a graph and progression of both algorithms as the number of nodes increase.

## V. CONCLUSION

In this study, the major hospitals in Ibadan, Oyo state was captured as a network graph and two algorithms (Kruskal and Prims algorithm) was used to construct the minimum spanning tree, that is, a sub graph of the map in Figure 1.2, containing all the nodal hospitals for which the total distance in kilometer is minimum. The study revealed that the minimum spanning tree distance, produced by both algorithms, for the 82 hospitals is **298.447km.** The time taken for Kruskal's algorithm to give this

result was 234.8 seconds whereas Prim's algorithm produced the MST for 82 nodes in 249.9 milliseconds.

Furthermore, we investigated the computational speed of both algorithms using different datasets, and the study has revealed that Kruskal's algorithm performed significantly better than Prim's algorithm in terms of computational speed, executing 21,048 nodes in **1,138.750 seconds** (approximately 19 minutes) while Prim's algorithm executed 21,048 nodes in **1,873.320 seconds** (approximately 31 minutes).

Future studies will focus on the improvements of both algorithms for very large graphs i.e. 1 million nodes or more.

## REFERENCES

[1] Erkan, A. (2018, July). The educational insights and opportunities afforded by the nuances of Prim's and Kruskal's MST algorithms. In *Proceedings of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education* (pp. 129-134).

[2] Huang, F., Gao, P., & Wang, Y. (2009, November). Comparison of Prim and Kruskal on Shanghai and Shenzhen 300 Index hierarchical structure tree. In *2009 International Conference on Web Information Systems and Mining* (pp. 237-241). IEEE.

[3] Zeng, X., Liu, Q., & Yao, S. (2019). An Improved Prim Algorithm for Connection Scheme of Last Train in Urban Mass Transit Network. *Symmetry*, *11*(5), 681.

[4] Li, H., Xia, Q., & Wang, Y. (2017). Research and improvement of kruskal algorithm. *Journal of Computer and Communications*, *5*(12), 63.

[5] Munier, B., Aleem, M., Islam, M. A., Iqbal, M. A., & Mehmood, W. (2017). A Fast Implementation of Minimum Spanning Tree Method and Applying it to Kruskal's and Prim's Algorithms. *Sukkur IBA Journal of Computing and Mathematical Sciences*, *1*(1), 58-66.

[6] Jothi, R., Sraban Kumar Mohanty, and Aparajita Ojha. "Fast approximate minimum spanning tree based clustering algorithm." *Neurocomputing* 272 (2018): 542-557.

[7] Saylı, A., & Alkhalissi, J. H. (2018). Negligence Minimum Spanning Tree Algorithm. *Avrupa Bilim ve Teknoloji Dergisi*, (14), 70-76.

[8] Ayegba et al., (2020). A Comparative Study of Minimal Spanning Tree Algorithms.

[9] Frempong, B. E. (2013). *Optimal Pipeline Connection for the Water Supply System in the Kwahu South District of Ghana*(Doctoral dissertation).

[10] Gitonga, C. K. (2015). Prims algorithm and its application in the design of university LAN networks. *International Journal*, *3*(10).

[11] Huang, X., Guo, W., & Chen, G. (2015, March). Fast obstacle-avoiding octilinear steiner minimal tree construction algorithm for VLSI design. In *Sixteenth International Symposium on Quality Electronic Design* (pp. 46-50). IEEE.

[12] Arogundade, O. T., Sobowale, B., & Akinwale, A. T. (2011). Prim Algorithm Approach to Improving Local Access Network in Rural Areas, International Journal of Computer Theory and Engineering, 3(3), 413–417.

[13] Effanga, E., & Uwe, E. (2016). Minimum Spanning Tree of City to City Road Network in Nigeria. IOSR Journal of Mathematics, 12(04), 41–45.

[14] Dey, A., Broumi, S., Bakali, A., Talea, M., & Smarandache, F. (2019). A new algorithm for finding minimum spanning trees with undirected neutrosophic graphs. *Granular Computing*, *4*(1), 63-69.

[15] Khalil, E., Dai, H., Zhang, Y., Dilkina, B., & Song, L. (2017). Learning combinatorial optimization algorithms over graphs. In *Advances in Neural Information Processing Systems* (pp. 6348-6358).