SCRIBO: a Graph Visualizer & Analytical Tool

E Ajith and N Uma

June 19, 2022

SCRIBO: A Graph Visualizer & Analytical Tool

Ajith E

PG Student, New Horizon College of Engineering

Bangalore

Email:ajithyashas1615@gmail.com

Ms.Uma N

Sr. Assistant Professor

New Horizon College of Engineering,Bangalore

Email:nhce.uma2021@gmail.com

Abstract---Data science is a very visual field of computer science where everything needs to be depicted graphically in order to derive new forms of data and conclusions from it. For statistics we have bar charts and pie charts which represent the distribution of numbers across various scales, plots allow to distribute two or more data sets over a 2D or even 3D space to show the relationship between these sets and the parameters on the plot, maps allow locating elements on relevant objects and areas such as building plans, website layouts, etc. In Mathematics, a graph is a pictorial representation of any data in an organized manner. The graph shows the relationship between variable quantities. In a graph theory, the graph represents the set of objects, that are related in some sense to each other. The objects are basically mathematical concepts, expressed by vertices or nodes and the relation between the pair of nodes, are expressed by edges. Among the most popular map visualizations are heat maps, dot distribution maps, cartograms, etc. But for relational data such as given a group of people who have contacts with other important business or given a cluster of cities and where we have to calculate the shortest path between two cities or in the game of football what is the shortest path of a player from his current position to the goal post where the state of data is dependent on other data, depicting these relationship's graphically requires complex visualization software where current implementations are either not user friendly , hard to operate and expensive. This paper introduces new visualization techniques where graphs can be better rendered using advance techniques such as vertex and edge clipping against the view port , rendering a large drawing area by panning the view port to focus on a specific region , labelling vertex data with numeric , text and images and advanced analytic algorithms such as finding the shortest path between two vertices using Djikstra's ,Floyd ,finding the minimum spanning tree using Prims and kruskals and traversing all vertices in a graph using Breadth first search and Depth first search. This paper explains the above mentioned techniques in detail and how they can work together to create an advanced visualization tool for data scientists working with bidirectional weighted relational data i.e graphs and also aims to create graph's visually using a point and drag interface and employ a variety of algorithms to visually see the results of those algorithms.

Keywords – Graph, Relational , Visualization , Analytics

I. INTRODUCTION

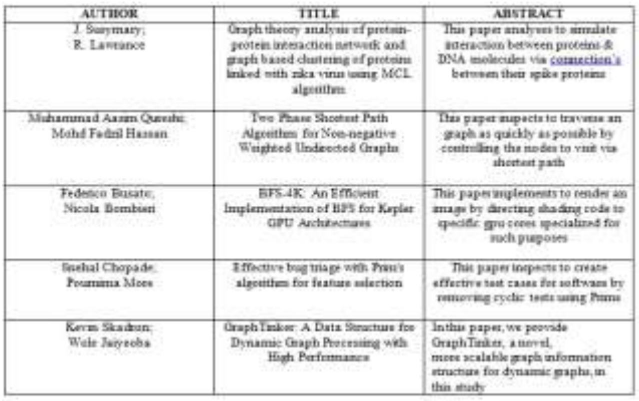
Graph Theory is the study of graphs in discrete mathematics. A graph is defined as a mathematical structure that connects a group of points to express a certain function. It is employed to establish a pairwise link between items. The graph is made up of vertices and edges .The linear graph has applications not just in mathematics, but also in computer science, physics and chemistry, linguistics, biology, and other sciences. The finest illustration of graph structure in real life is GPS, which allows you to track a journey or determine the direction of travel.

A graph is a collection of objects connected in some way, according to graph theory.. The objects A graph X (A, B), includes two sets A and B. The elements of A are the vertices of graph X whereas Graphs have a natural linear representation in which each vertex is represented by a point and each edge by a line joining two points. To have a better understanding of graphs, we should are essentially mathematical notions that are represented by vertices or nodes, with edges expressing the relationship between the nodes.. It is a visual depiction of mathematical truth. A graph is officially referred to as a pair G. (V, E). The finite set vertices are represented by V, while the finite set edges are represented by E. As a result, we may state that a graph has a non-empty set of vertices V and a set of edges. Edge is defined as the pair of vertices{1, 2, 3, 4, 5}, and set B {1, 2}, {2, 3}, {3, 4}, {4, 5} defines a graph with 5 vertices and 4 edges respectively.

Understand its base - Graph Theory. For example, Euler observed the four bodies of land and the seven bridges. This helped him to draw out the first known visual representation of a modern graph. A modern graph is represented by a set of points, known as vertices or nodes are joined by a set of connecting lines known as edges. Euler first made an attempt to construct the path of the graph. Later, while experimenting with different theoretical graphs with alternative numbers of vertices and edges, he predicted a general rule. He concluded that in order to be able to walk in the Euler path, a graph should have none or two odd numbers of nodes. From there, the concept of graph theory was introduced So using this concept of graph theory we can create an application that can create graph data structures from scratch , we can build an application that can analyse the graph using a variety of algorithms and to load graphs from text files and save them back to it for easy backup and restore and most importantly, to visualise and alter graph data structures in order to solve complex real-world problems.

II.RELATED WORKS

The below mentioned work mentions the technologies used in graph theory



AUTHOR	TITLE	ABSTRACT
J. Susymary; R. Lawrance	Graph theory analysis of protein-protein interaction network and graph based clustering of proteins linked with zika virus using MCL algorithm	This paper analyses to simulate interaction between proteins & DNA molecules via connection's between their spike proteins
Muhammad Aasim Qureshi; Mohd Fadzil Hassan	Two Phase Shortest Path Algorithm for Non-negative Weighted Undirected Graphs	This paper inspects to traverse an graph as quickly as possible by controlling the nodes to visit via shortest path
Federico Busato; Nicola Bombieri	BFS-4K: An Efficient Implementation of BFS for Kepler GPU Architectures	This paper implements to render an image by directing shading code to specific gpu cores specialized for such purposes
Snehal Chopade; Pournima More	Effective bug triage with Prim's algorithm for feature selection	This paper inspects to create effective test cases for software by removing cyclic tests using Prims
Kevin Skadron; Wole Jaiyeoba	GraphTinker: A Data Structure for Dynamic Graph Processing with High Performance	In this paper, we provide GraphTinker, a novel, more scalable graph information structure for dynamic graphs, in this study

III.EMPLOYED ALGORITHMS

The algorithms and concepts used above have already been published in peer-reviewed journals.

A. Shortest Path*:* Dijikstra's : Analyse all of a vertex's outgoing edges to find the one with the least weight, then return to that vertex and repeat the process. Dijkstra's algorithm is very similar to Prim's minimum spanning tree algorithm. We generate an SPT (shortest path tree) with a given source as the root. We keep two sets: one contains vertices that are already in the shortest-path tree, and the other contains vertices that aren't yet in the shortest-path tree. We find a vertex in the other set (set of not yet included) that is closest to the source at each step of the algorithm.

Floyd's : Store n x n[n is number of vertices] matrix of the shortest path of every vertex in the graph to another vertex and reconstruct the matrix only if graph has changed else cache it. As a first step, we initialize the solution matrix to the same values as the input graph matrix. The solution matrix is then updated by treating all vertices as intermediate vertex. The idea is to pick all vertices one by one and update all shortest paths that include the picked vertex as an intermediate vertex.

B. **Graph Traversal**: Breadth first search (BFS): Push all of a vertex's neighbours into a queue, print every vertex in the queue, and repeat the process for each vertex. When a dead end occurs in any iteration, the Breadth First Search (BFS) algorithm traverses a graph breadth ward and uses a queue to remember to get the next vertex to start a search.

However, in order to understand the final infrastructure, a basic understanding of each of these algorithms and how they work together for this project is required.

Depth first search (DFS): Print the start vertex, pass it as a parameter to the algorithm, and call it on each of its neighbour's recursively. When a dead end occurs in any iteration, the Depth First Search (DFS) algorithm traverses a graph in a depth ward motion and uses a stack to remember to get the next vertex to start a search.

C. **Minimum Spanning Tree:** Prims :- Take each edge and add its start and end vertex to a set; if any two vertices pairs repeat in the set, it is considered to have a cycle and should be discarded. Prim's algorithm is a greedy algorithm for finding a minimum spanning tree between two points in a weighted undirected graph. This means it finds a subset of the edges that forms a tree that contains every vertex and has the least total weight of all the edges in the tree.

Kruskal's : Take a look at a starting vertex. All of its neighbours should be visited and added to a queue. If a vertex has already been visited, discard the edge connecting the current and destination vertex. The concept of Kruskal's algorithm is introduced in discrete mathematics' graph theory. In a connected weighted graph, it's used to find the shortest path between two points. This algorithm turns a graph into a forest, treating each node as a distinct tree.

IV. FINAL INFRASTRUCTURE

Using the above-mentioned algorithms we can summarize the implementation of graph visualizer and analytical tool with rendering pipeline and flow chart.

1. Rendering pipeline(System architecture)

Tool Panel[Graph Creation Tool]

- Right Click to add nodes
- Left click & drag to add edge
- Select 2 or more nodes & right click to use algorithms

Tool Panel[Graph Creation Tool]

- Right Click to add nodes
- Left click & drag to add edge
- Select 2 or more nodes & right click to use algorithms

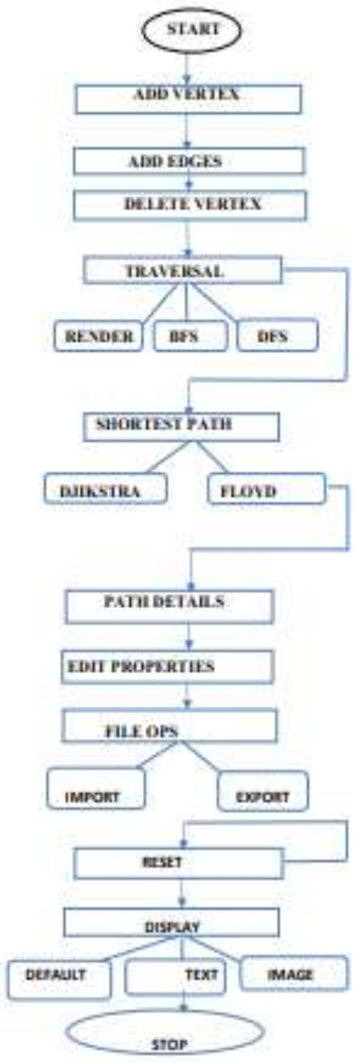
Tool Panel[Graph Creation Tool]

- Right Click to add nodes
- Left click & drag to add edge
- Select 2 or more nodes & right click to use algorithms

In summary , the functionalities of the graph visualizer and analytical tool

1. ADD VERTEX : Use this option to add vertex(nodes) to the drawing area.
2. ADD EDGE : We may add weights to edges, which are used to link one or more nodes.
3. DELETE VERTEX : You may delete a vertex using this option (node).
4. TRAVERSAL : There are three submenus under traversal.
   - Render : This option allows you to change the graph's colour and latency.

1. Flow chart of graph visualizer and analytical tool



   - BFS: This submenu is used to push all of a vertex's neighbours into a queue, print every vertex on the queue, and then repeat the operation for each vertex.
   - DFS: This submenu is used to print the start vertex, send it as a parameter to the method, then call the algorithm on each of its neighbours in a recursive fashion.
5. MST : There are two submenus under MST.
   - Prims :- For each edge, add its start and end vertex to a set, and

if any two vertices pairs repeat in the set, it is regarded to have a cycle, thus remove it.

- Kruskal's :- Select a starting vertex. All of its neighbours should be visited and added to a queue. If a vertex has previously been visited, discard the edge linking the current and destination vertices

6. PATH DETAILS : This option displays all of the graph's paths, as well as their weights and edges.

7. EDIT PROPERTIES : This option lets you change the edge and weight weights as well as see the vertex number.

8. FILE OPS : We have two subcategories under file ops.

- IMPORT : A pre-existing file may be imported into the drawing area and utilised to access the menus.
- EXPORT : The Drawing file may be exported as a bin file, which can then be imported whenever needed, reducing the amount of time spent on it.

9. RESET : This option allows you to clear the drawing area and start again with a new vertex.

10. DISPLAY : Under display we have 3 sub menus

- DEFAULT: Numbers can be used to add the vertex
- TEXT: The vertex is entered as a string of characters.
- IMAGE : The vertex is represented as an image that may be linked to edges and weighted with other images.

V.RESULTS & DISCUSSION

Figure 1: Graph

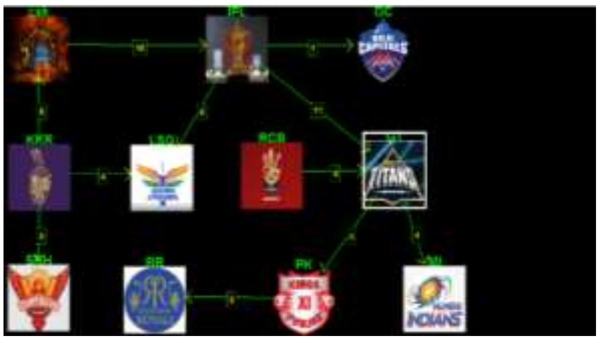


Figure 2: Shortest path using Floyd's algorithm

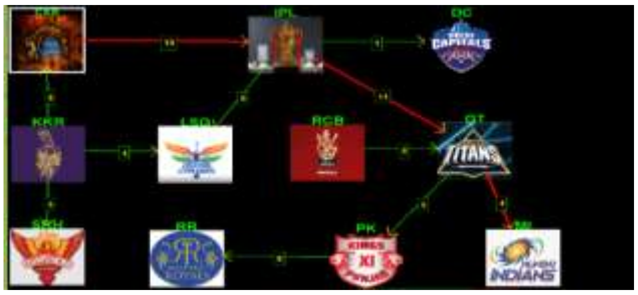
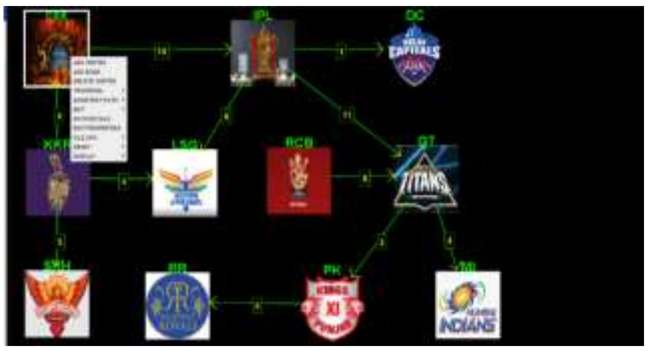


Figure 3 : Menu in graph visualizer and analytical tool



VI.CONCLUSION

Thus with this application we can analyse graph's visually and employ a large variety of algorithms to get the desired results we want. Existing applications use complex text sheets to display the results which can make the whole process tedious to see and process and on top of that use ineffective rendering techniques to render the graph causing large amount of lag and other mishaps. Our applications aims to simplify the whole process and make the whole process visual and reduce lag and improve overall speed and efficiency with comprising on the accuracy of algorithms thus allowing for large graph data structures to be loaded from text files without losing time. With large real world problems condensed into an graph data structure we can overall reduce the need for high end hardware to visualize these results and improve overall performance in the future.

VII.REFERENCES

[1] L. Bass, P. Clements, and R. Kazman, Software Architecture in Practice, 2nd ed. Reading, MA: Addison Wesley, 2009. [E-book] Available: Safari e-book.

[2] D. Ince, “Acoustic coupler,” in A Dictionary of the Internet. Oxford University Press, [online document], 2019. Available: Oxford Reference Online, http://www.oxfordreference.com [Accessed: May 24, 2007].

[3] M. T. Kimour and D. Meslati, “Deriving objects from use cases in real -time embedded systems,” Information and Software Technology, vol. 47, no. 8, p. 533, June 2009. [Abstract]. Available: ProQuest, http://www.umi.com/proquest/. [Accessed November12, 2017].

[4] A. Altun, “Understanding hypertext in the context of reading on the web: Language learners’ experience,” Current Issues in Education, vol. 6, no. 12, July, 2005. [Online serial]. Available: http://cie.ed.asu.edu/volume6/number12/. [Accessed Dec. 2, 2007].

[5] C. Wilson-Clark, “Computers ranked as key literacy,” The Atlanta Journal Constitution, para. 3, March 29, 2017. [Online], Available: http://www.thewest.com.au. [Accessed Sept. 18, 2017].

[6] Z. Boyan and C. Qingyu, "A New Method for Electronic Tracking Beam Stabilization," Proceedings of CIE International Conference of Radar, pp.507-510, 1996.

[7] B. L. Diamond, et al, The ARIES Program Coordinates, Transformations, Trajectories and Tracking, ESD-TR -75-255, Lincoln Laboratory, MIT, 1975.

[8] Ramachandra, K. V., Kalman Filtering Techniques for Radar Tracking, Marcel Dekker, INC. 2000.

[9] S. S. Blackman and R. Popoli, Design and Analysis of Modern Tracking Systems, Artech House, Norwood, MA, 2018.

[10] D. Lerro and Y. Bar-Shalom, "Tracking with debiased consistent converted measurements versus EKF," IEEE Trans. on Aerospace and Electronic Systems, vol. 29, no. 3, pp. 1015-1022, July, 2019.

[11] T. Bohné, I. Heine, Ö. Gürerk, C. Rieger, L. Kemmer and L. Y. Cao, "Perception engineering learning with virtual reality," in *IEEE Trans. on Learn. Technologies*, vol. 14, no. 4, pp. 500-514, 1 Aug. 2021. doi: 10.1109/TLT.2021.3107407.

[12] R. H. Byor, “Nanotechnology in water and wastewater treatment,” Ph.D. dissertation, College of Eng. and Sci., Victoria Univ., Melbourne, 2016, p. 84.