AN EMPIRICAL ANALYSIS OF GRAPH ALGORITHM IN CONTEXT OF FREQUENT SUBGRAPH MINING ON GIRAPH SYSTEM

Sadhana Priyadarshini

Department of Computer Science and Engineering, GIT, GITAM
Gandhi Nagar, Rushikonda, Visakhapatnam-530045, Andhra Pradesh, INDIA.
sadhana.priyadarshini00@gmail.com

Sireesha Rodda

Department of Computer Science and Engineering, GIT, GITAM
Gandhi Nagar, Rushikonda, Visakhapatnam-530045, Andhra Pradesh, INDIA.
sireesharodda@gmail.com

Abstract

The Recurrent Subgraph Extraction plays a key role in the Graph Mining field when our data is distributed over networks. This paper emphasizes different types of graph mining algorithms with the Giraph Distributed System to get more desirable and valuable results than existing methods. We discuss how our proposed model MapReduce Geometric Multi-way Advanced Optimized Frequent Subgraph Mining (MGMAOFSM) impacts different graph mining mechanisms for centralized and distributed systems. The comparison is done for different criteria such as memory requirement or execution time with real four datasets (Facebook Social Network, Coronavirus (COVID-19) tweets, Google web graph, Patent Citation Network) with different threshold values. We implement various algorithms such as Triangle Closing, Shortest Path, Connected Components, and PageRank algorithms, and find out our proposed algorithm that requires less memory with the Triangle closing algorithm whereas in the case of PageRank is lowest with all threshold values.

Keywords: **Support count; Mapper and Reducer; Recurrent subgraph extraction; Graph Distribution.**

1. Introduction

The victorious approach of data mining is extremely noticeable in the area of citation graphs, social networks, chemical structure, and web data mining. With the expanding demand for graphical data, generating recurrent patterns can be applied to find out biological characteristics in the chemical dataset, which can be helped to reduce the cost and time required for manufacturing drugs. It can also be helpful in retail and shopping databases to attract customers based on demand items.

Graph Mining is a part of Data Mining that extracts interesting information from graph datasets. The performance of such an algorithm purely depends on how we divide the whole dataset and load distribution so that we can execute subgraph extraction algorithms parallel to reduce time and space complexity. Rapid transposition of shape and structure inside the dataset leads to improvement in existing recurrent subgraph extraction [1]. The upgrade is also challenging due to its dependency on structural features for different applications such as link survey, synthetic blend classification, and VLSI back-pedal engineering. However, these techniques have the most significant implementation in various areas like constructional and relational features of the ecosystem, chemistry model, etc. The community-based network consists of huge graphs where graph extraction can make it a problem for community identification. We can link the association node clustering to community issues [2…6].

The existing algorithm for the centralized graph database has both horizontal and vertical approaches to search recurrent subgraphs in a breadth-first way. Graph partition before candidate generation can reduce the algorithm complexity, space shortage, and increase time complexity. Our objective is based on the partition of the entire graph before the candidate production by using the proposed Geometric Multi-way Advance Frequent Subgraph Extraction mechanism and direct individual nodes in case of centralized and distributed graph datasets respectively. Our designed methodology MapReduce Advanced Optimized Frequent Subgraph Mining (MAOFSM) for different graph mining algorithms gives us the upgrade results of approximately nearly one-third to half of existing algorithms.

2. Related Work

Saeed Salem et. al proposed a RASMA algorithm to extract frequent and maximal recurrent subgraphs by implementing a reverse-search strategy to specify attached subgraphs of an indirect graph. It also uses various pruning methodologies to improve runtime performance [8]. Lan. B. Q. Nugen et. al designed CloGraMi(Closed frequent subgraph Mining) methods to extract all closed recurrent subgraphs from a Single Large graph dataset. There are two strategies used, the initial one is the latest level order traversal strategy to faster find closed subgraphs are searching, and then, we need to find a condition for premature pruning of a large portion of open candidates [9].

SaifUr.Rehman et. al developed A RAnked Frequent pattern-growth Framework (A-RAFF) that eliminated twin and extensive recurrent subgraphs. Various benchmark and synthetic graph datasets were used to validate their effectiveness by extensive experimental analysis [10].

Giulia Preti et. al designed a sample-foundation randomized algorithm called MaNIACS, used to calculate top-quality estimates of the gathering of subgraph patterns that a recurrent in a graph dataset. They use Minimum Node Image-based (MNI) for the frequency count of subgraphs. Hence time requirement is drastically reduced [11]. Vaclav Snasel et al made a survey on the difficulties of existing frequent subgraph mining technologies, their vital phases of it, the main clusters of FSM, and their association with the modern research field. It is based on association rules and structure discovery to generate algorithmic development in the modern world [12].

3. Problem Definition

Day by day, we are continuously generating digital data from different sources, as a result, we can't define a perfect mechanism to extract the desired pattern of information per our dataset. Big Data production leads the scientist to think again to solve the problems that arise. In many companies, everyday huge complex, semi-structured, unstructured data is collected and existing algorithms are unable to extract the frequent subgraphs. The main goal of data analysts in such companies is to achieve the ability to analyse and understand small-scale parts of the dataset. In this paper, we implement various graph algorithms with existing MapReduce Geometric Multi-way Advanced Optimized Frequent Subgraph Mining (MGMAOFSM) and make a comparative study that uses the algorithm as per the dataset given to us to get better results with different threshold (support) value[10…14].

4. Proposed Approach:

In this section, we narrate our four contributions in detail with various examples with respect to various threshold values.

4.1. Triangle closing graph algorithm for frequent subgraph mining:

The triangle closing is a method to search nodes that are not connected directly to it but are part of its social neighbour node. It helps us to avoid data loss during candidate generation [15]. This type of algorithm best suits a social network dataset. This method consists of the following steps:

Step-1: Each node (V_i) dispatches its adjacent record to all its neighbour and polls to halt.
Step-2: Each node (V_i) collects the adjacent list of neighbour. Now v starts to maintain 2-hop adjacency details. Each vertex exercises the 2-hop neighbour details and finds out the node that can generate associations to the node. The record of current nodes is classified in a descending sequence of a number of interconnections to the neighbour to the node.

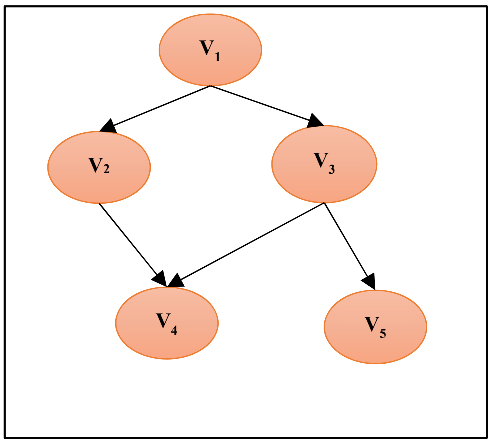


Figure 1. Facebook social sites dataset where user represented by vertex (V_i) and relationship represented by edges.

The social graph example is shown in fig.1, the vertex represents the user and edges represents the relationship exist between users. We implement the algorithm with the sample data shown in fig.1. In first superstep, each vertex dispatch its complete neighbour records to each and every neighbour. For example, V_1 dispatches its neighbour list (i.e. V_2, V_3) to both nearest V_2 and V_3 [16]. In next superstep, V_2 knows the presence v_3 and vice versa. Now the three vertex altogether form a triangle that we like to close. During second superstep, each vertex collect all adjacent ids as message that contain neighbours; v_4, v_5, v_1. Presume that V_1 will disregard the message that represent itself (i.e. V_1), then it got v_4 and v_5.This will generate its ranked list of neighbour that V_1 may be know. Alike V_3 will have V_2 as a possible friend and vice versa. The methodology will generate V_1 as new possible familiar for V_5 who currently is a familiar of V_3 only [17, 18].

Superstep-1: Superstep-2:

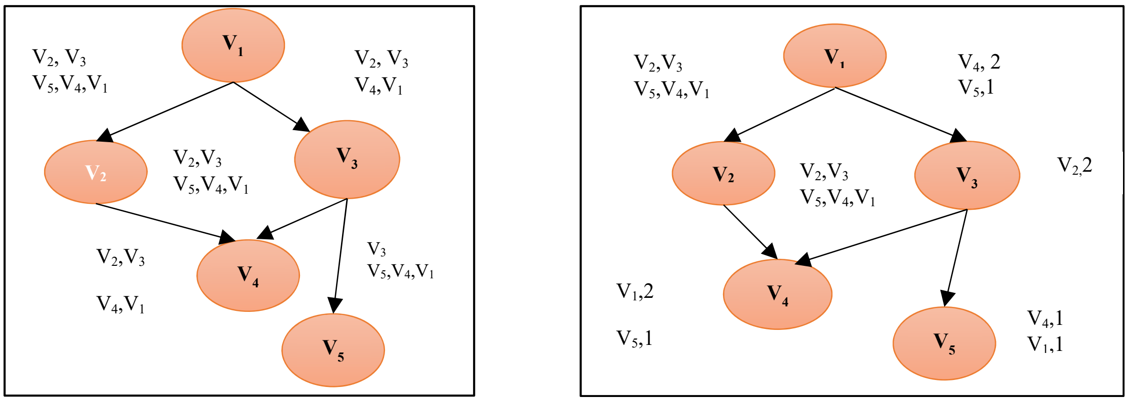


Figure 2. Implementing triangle algorithm using Facebook Social Sites

4.2. Shortest path graph algorithm for frequent subgraph mining:

In this page, the main objective to use the shortest path algorithm to overcome the problem arises for optimize the network problem and graph mining. The smallest distance between a set of nodes is the path that begins with the source node and terminates at the destination node such that the grant total of weights of its compose edge is keep downed. In this paper, we use it a little bit differently to way together frequent patterns (i.e. Single Source Shortage Path (SSSP)) [19...23].

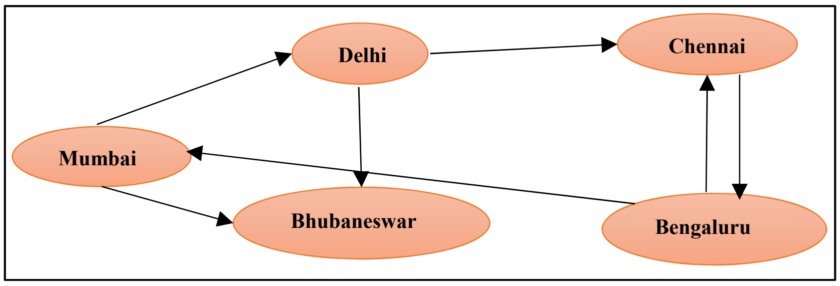


Figure 3. Sample of Network Graph Dataset (node represents cities and edge represents the road between them)

The fig.3 represents an example of road network with five nodes representing various cities Mumbai, Delhi, Bhubaneswar, Chennai, and Bengaluru. Each node has a direction to indicate the source and destination and weight that represents the distance between two cities [11]. Let us assume that a person in Mumbai desires to know what the possibilities of the smallest paths to all five cities are. In the first loop, all nodes have the smallest distance of infinity except the origin node Mumbai which has an interval of zero to itself. If the smallest distance is less than the node value, the node will generate the lowest distance in addition to edge weight. Once we see in fig 3 is then the one that satisfies this condition. Then, the Mumbai node will dispatches costs of 10(0+10) along with 5(0+5) to Delhi and Bhubaneswar, respectively. At that moment, the Mumbai node poles to halt. In the second loop, only nodes Bhubaneswar and Delhi are active [24]. For Bhubaneswar, it gets only one message (5) which is less than its value (infinity). Hence, it will update its value to the smallest distance (5) and generate 8(3+5), 14(5+9), and 7(5+2) to nodes Delhi, Chennai, Bengaluru, respectively. Similarly, Delhi will 1 generate 12 and 11 nodes in Bhubaneswar and Chennai, respectively. Then both nodes Bhubaneswar and Delhi will pole to halt. In the next loop, nodes Delhi, Bengaluru, and Chennai are active as they get a message [18]. Each node will receive the smallest distance between its value and received messages and generate the corresponding distances [22]. The loop continues till all nodes become idle at once all nodes pole to halt, each node maintains the distance required to travel from a source node (i.e. Mumbai). Hence shortest distance from Mumbai to Delhi has cost 8 (i.e. path from Mumbai to Bhubaneswar cost 5 and then Bhubaneswar to Delhi cost is 3).

4.3. Connected components graph algorithm for frequent subgraph mining:

In a connected component algorithm, two nodes are called to be attached if there is an edge or path between them; i.e. they are attachable to each other. In the case of a connected graph only one connected component (i.e. graph itself) whereas in case of an undirected graph, an attached component is a subgraph in which any two nodes are connectable to each other. In this paper, we use this algorithm to find out different groups of nodes that share the same subgraphs. If a node has no edge then it generates a connected component by itself.

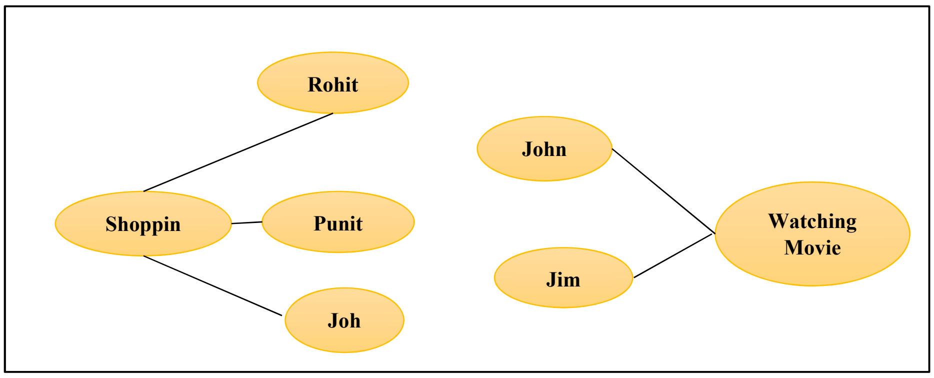


Figure 4. Sample Facebook social network: User associate with their interaction

The fig.4 represents a sample of our Facebook social graph dataset that consists of a set of users and their hobbies. As shown in fig. there are two connected components. The first component consists of the nodes Rohit, Punit, John whose connected throw Shopping, the component consist of John, Jim that connected throw Watching Movie. One assumption we made in this paper, that each node has a unique ID. To implement the algorithm we require four supersteps [23].

- Superstep-1: Individual nodes are assigned their attached component id (cID) with its unique ID. Then each node dispatches its attached component id to all its adjacent nodes. For example Shopping node dispatches its cID (5) to nodes Rohit, Punit, John. Similarly, it gets from their cID.
- Superstep-2: Individual node compares all the received ids with its cID to be the lowest value among all of them. For example, Rohit received 5 and its own cID is 1, so its cID remains 1. As cID did not update,

the node will not be dispatched to adjacent nodes and will be idle. The shopping node will change cID to the smallest between 3,2,15. It then will dispatch cID(1) to its adjacency[22].

- Superstep-3: Punit and John nodes receive a small-scale cID from the shopping node and change its cID accordingly.
- Superstep-4: The watching movie node gets cID of 6 which is not less than its cID. In this case, this node becomes idle. So the program will end. Once the program terminates, we can output the connected component id of all nodes and group similar ids together.

4.4. PageRank graph algorithm for frequent subgraph mining:

We use the PageRank algorithm for finding out the significance of the authority of nodes in graphs. It is used to classify the internet web pages such that major pages with higher PageRank values are shown first in the search result. Our main objective to use this algorithm in frequent subgraph mining is that once a threshold value is given it can filter out easily as major websites are likely to receive more links than others. In this algorithm, a user gets a particular page by pressing randomly on links. The algorithm works as follows: Initially, the user is assumed to visit a fixed webpage; represented in our dataset V1 from a set of webpages (V_i). The probability of getting through a fixed webpage edge in E_1= 1/ number of webpages vertices. Once the first edge is chosen, the next pressed edge can be any webpage vertex V_1 is referring to. If V_1 has E_1 edges and one of the links is referring to E_2, then E_1 will generate $1/V_1$ of its significance to E_2. If E_2 has many edges pointing to it, then its significance I_2 will be the total of the significance values of these incoming edges to $E_{(i)}$. In this paper, we use a random surfer model for PageRank [24].

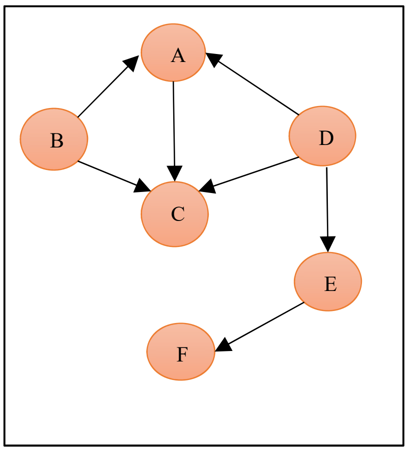


Figure 5. Social web graph: vertices are webpages and edges are departing links between users.

The fig.5 represents a social site web graph datasets where A, B, C, D, E, F are represented by nodes with their links. In the first loop, all nodes have equal probability to reach any node 0.16(1/ number of nodes). Each node dispatches a message to all its adjacency nodes. On the basis of vertex value and number of departing edges, the message value is fixed. For example, node B has a value of 0.16 and 2 departing edges; both A and C will be sent a message value 0.08(0.16/2). In the second loop, each node will add all messages it gets. So A node 0.08and 0.006, so total is 0.086. Now the new node value is 0.196. In the third loop B and D did not receive any message, so it became idle. The receiving and implementing process of the PageRank formulae is repeated till each node maintains its PageRank value.

5. Experimental Result and Comparison

In succeeding, we represent a relative experimental study on four different graph mining methods in Giraph system for recurrent subgraph extraction with four graph datasets.

Experimental Set up

We take four datasets and for experimental analysis we use a personal computer with an AMD Ryzen 5 3500U with Radeon Vega Mobile Gfx 2.10 GHz, 8 GB RAM, Windows 10 66-bit operating system. The details of our four datasets are as follows:

- Facebook Social Network: This is downloaded from the website SNAP: Network datasets: Social circles (stanford.edu) and the dataset has circles (or friend list) from Facebook which is considered a node of our data set. There are 4039 nodes, 88234 edges, nodes in the largest WCC is 4039(1.000), nodes in the largest SCC is 4039(1.000), edges in the largest SCC is 88234 (1.000), average clustering coefficient

0.0655, number of triangles 1612010, the fraction of closed triangles 0.2647 with longest shortest path 8 and effective diameter is 4.7(90%).

- Coronavirus (COVID-19) tweets: this dataset was downloaded from https://ieee-dataport.org/ open-access/ Coronavirus (COVID-19) Tweets Dataset | IEEE DataPort (IEEE-dataport.org)The dataset has a CSV file that consists of IDs and sentiment scores of tweets connected to the COVID-19 pandemic. There are a total of 1,965,835,614 tweets the in COVID-19 tweets dataset. All the tweets before 20th march 2020 have a Greek alphabet name rather than a number counter to make a convention to update the CSV file. In this paper, we start the 'beta' name instead of 'gama' and proceed ahead.
- Google web graph: Like the Facebook social Network the dataset is downloaded from https://SNAP: Network datasets: Google web graph (stanford.edu). The dataset consists of web pages that are represented by nodes and directed edges represented by hyperlinks between them. There are 875713 nodes, 5105039 edges, the total numbers largest WCC nodes, WCC nodes, SCC edges are 855802(0.977), 5066842(0.993), 434818(0.497), 3419124(0.670) respectively. The average clustering coefficient is 0.5143, the longest shortest path is 1 with an effective diameter of 8.1(90%).
- Patent Citation Network: The dataset is downloaded from http:// SNAP: Network datasets: the US Patent citation network (stanford.edu). The dataset is carried by the National Bureau of Economic Research based on US patents datasets for 37 years. It consists of all utilities granted during the span of time including 3923922 patents. The entire graph dataset consists of 3774768 nodes and 16518948 edges. The total number of nodes in the largest WCC, and edges in the largest WCC are 3764117(0.997) and the effective diameter 9.4(90%).

By using both directed and undirected graph datasets, we perform the experiment and the analysis using different thresholds with the time required to execute different graph mining algorithms and overall memory requirements for it. To run the connected components on the datasets, we need to perform four supersteps till the method intersects. To overcome the problems related to graph theory and network optimization, we analyse the shortest path graph mining algorithm, to select the shortest path between specific nodes to all other nodes in the graph. To implement the triangle closing algorithm we performs two supersteps to get the number of are ranked in descending order of the number of connections neighbours of the vertex.

In this section, we used our proposed algorithm MapReduce Giraph Advanced Optimized Frequent Subgraph Mining (MGAOFSM) with different graph mining methods like Triangle Closing, Shortest Path, Connected Components, Page Rank to analyse it performance. Initially, we perform the MGAOFSM algorithm with different graph mining methodologies with different support or threshold values (i.e. 25, 20, 15, 10) percentage shown in fig. 6.

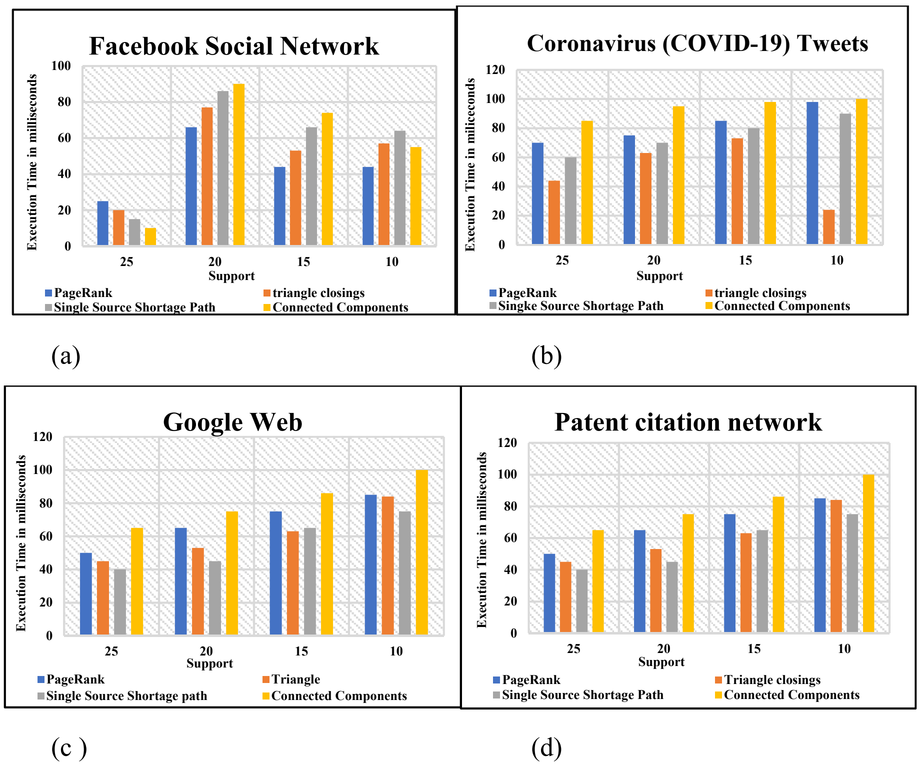
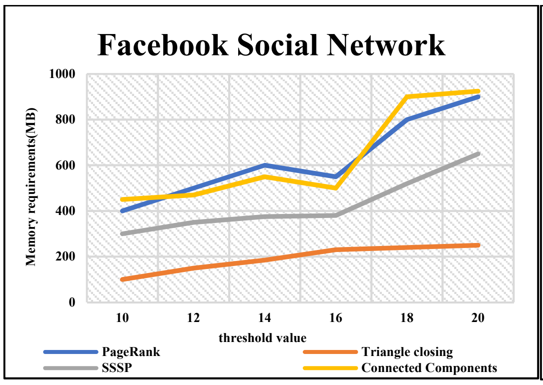


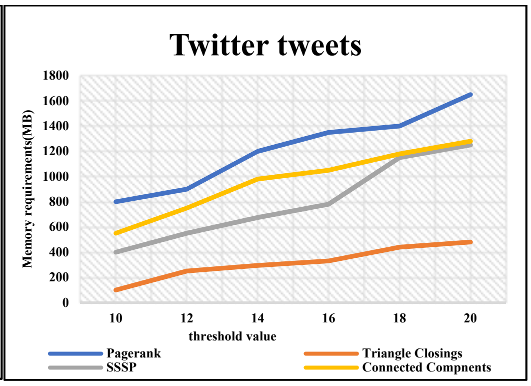
Figure 6: Execution time vs Support value

- For the Facebook Social Network (Figure 6.a): In these datasets, the Connected Components algorithm with MGAOFSM performs well for high threshold value (25) whereas PageRank algorithm performs better with all four datasets. Triangle closing and Single Source Shortage Path algorithm performance same with 20 and 15 support value, but it drastically reduces with 25 threshold value due to optimization at this level. If we evaluate on the basis of execution time with different threshold value triangle closing algorithms for MGAOFSM with Facebook Social Network is better performance.
- Coronavirus (COVID-19) tweets (Figure 6.b): For this dataset, the PageRank, Connected Component, Single Source Shortest Path (SSSP) algorithm has the highest execution time with the lowest value for triangle closing algorithm for threshold 10. On an average Triangle Closing is the smallest execution time for all thresholds. The Connected Component has the highest average execution time in the dataset. For the PageRank, Connected Component, Single Source Shortest Path algorithms, the execution time with the decreasing threshold value.
- Google Web (Figure 6.c): For PageRank, Connected component, Single Source Shortest Path algorithms, the execution time increases with a decrease in threshold value. The average execution time for all threshold values is the smallest i.e. (56). Whereas connected components have maximum average execution time.
- Patent citation network (Figure 6.d): In these datasets, the SSSP has the lowest execution time with all values of thresholds and the highest with connected components. On average Triangle Closing was the lowest among all algorithms we tried to implement with our proposed methods.

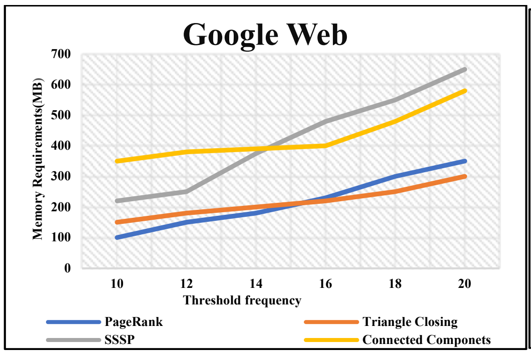
Secondly, we consider our comparison with execution time with four algorithms with our proposed methodology.



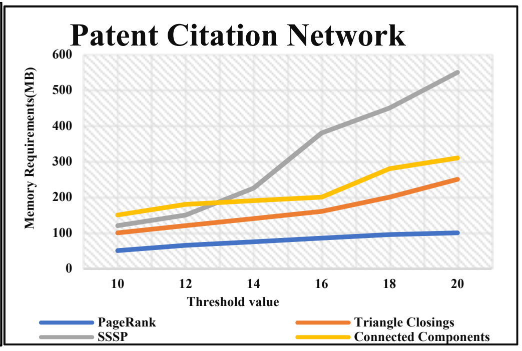
(a)



(b)



(c)



(d)

Figure 7. Memory requirement vs Threshold values

Except the Patent Citation Network, our proposed algorithm requires less memory with Triangle closing algorithm whereas in PageRank is lowest with all threshold values. The Single Source Shortest Path algorithm with GAOFSM has same memory requirement for 10, 12 percentage threshold value, then it increase with threshold value.

6. Conclusion and Future work

We investigated several graph mining algorithm for frequent subgraph with our proposed algorithm. The Giraph system has lowest execution time with Triangle Closing algorithm associated with MapReduce Geometric Multi-way Advanced Optimized Frequent Subgraph Mining. In this paper, we are able to merge both Giraph and

MapReduce frameworks to get better results as well as memory consumption less in a distributed system. PageRank methodology with our proposed algorithm required the lowest memory in all four graph datasets.

7. Conflicts of Interest

The authors declare no conflict of interest.

References

[1] Anchuri, P.,Zaki, M. J.,Barkol,O.,Golan,S. and Shamy M.(2013), 'Approximate graph mining with label costs'. In Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2013).

[2] D. Dai, W. Zhang, and Y. Chen, "Iogp: An incremental online graph partitioning algorithm for distributed graph databases," in Proceedings of the 26th International Symposium on High- Performance Parallel and Distributed Computing. ACM, 2017, pp. 219–230.

[3] Mccune, R. R., Weninger, T., and Madey, G.(2015) 'Thinking like a vertex: A survey of vertex-centric frameworks for large-scale distributed graph processing', ArXiv: 1507.04405 (2015).

[4] Tomasz Kajdanowicz, Przemyslaw Kazienko, and Wojciech Indyk. 2014. Parallel Processing of Large Graphs. Future Generation. Comput. Syst. 32 (March 2014), 324–337. DOI:http://dx.doi.org/ 10.1016/j.future.2013.08.007.

[5] Yue Zhao, Kenji Yoshigoe, Mengjun Xie, Suijian Zhou, Remzi Seker, and Jiang Bian. 2014. LightGraph: Lighten Communication in Distributed Graph-Parallel Processing. In Proceedings of the 2014 IEEE International Congress on Big Data (BIGDATACONGRESS '14). IEEE Computer Society, Washington, DC, USA.

[6] Bruce Hendrickson a,*, Tamara G. Kolda," Graph partitioning models for parallel computing", Parallel Computing 26 (2000) , www.elsevier.com/locate/parc.

[7] Saeed Salem, Mohammed Alokshiya , Mohammad Al Hasan "RASMA: a reverse search algorithm for mining maximal frequent subgraphs", BioData Mining 14, 19 (2021). https://doi.org/10.1186/s13040-021-00250-1

[8] Lam B. Q. Nguyen, Loan T. T. Nguyen, Ivan Zelnika, Vaclav Snasel, Hung Son Nguyen ,Bay Vo," A Method for Closed Frequent Subgraph Mining in a Single Large Graph" Digital Object IEEE, Identifier 10.1109/ACCESS.2021.313366, VOLUME 9, 2021,December 23, 2021.

[9] Lam B. Q. Nguyen , Ivan Zelinka, Quoc Bao Diep" CCGraMi: An Effective Method for Mining Frequent Subgraphs in a Single Large Graph" MENDEL. 27, 2 (Dec. 2021), 90-99. DOI:https://doi.org/10.13164/mendel.2021.2.090.

[10] Saif Ur Rehman, Kexing Liu, Tariq Ali, Asif Nawaz, Simon James Fong "A Graph Mining Approach for Ranking and Discovering the Interesting Frequent Subgraph Patterns" International Journal of Computational Intelligence Systems volume 14, Article number: 152 (2021) .

[11] Giulia PretiMatteo Riondato "MaNIACS: Approximate Mining of Frequent Subgraph Patterns through Sampling, KDD '21: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data MiningAugust 2021 Pages 1348–1358https://doi.org/10.1145/3447548.3467344.

[12] Xiaohong Zhang, Zhiyong Zhong, Shengzhong Feng, Bibo Tu, Jianping Fan," Improving Data Locality of MapReduce by Scheduling in Homogeneous Computing Environments", Conference: IEEE International Symposium on Parallel and Distributed Processing with Applications, ISPA 2011, Busan, Korea, 26-28 May, 2011.

[13] Lam B. Q. Nguyen,Ivan Zelinka,Vaclav Snasel,Loan T. T. Nguyen,Bay Vo" Subgraph mining in a large graph: A review", Wires Data Mining and Knowledge Discovery.08 March 2022

[14] Zigang Cao,Gang Xiong,Yong Zhao,Zhenzhen Li,Li Guo," A Survey on Encrypted Traffic Classification", International Conference on Applications and Techniques in Information Security ATIS 2014: Applications and Techniques in Information Security pp 73-81.

[15] Yufei Gao,[1] Yanjie Zhou,[2] Bing Zhou,[3] Lei Shi,[4] and Jiacai Zhang "Handling Data Skew in MapReduce Cluster by Using Partition Tuning", Journal of Healthcare Engineering , Recent Advances and Developments in Mobile Health, Volume 2017 |Article ID 1425102 ,"

[16] Hill S, Srichandan B, and Sunder Raman R(2012). 'An iterative MapReduce approach to frequent subgraph mining in biological datasets'. In Proceedings of the ACM Conference on Bioinformatics, Computational Biology and Biomedicine (2012).

[17] Jaliya Ekanayake, Hui Li, Bingjing Zhang, Thilina Gunarathne, Seung-Hee Bae, Judy Qiu, and Geoffrey Fox. 2010. Twister: A Runtime for Iterative MapReduce. In Proceedings of the 19th ACM.International Symposium on High Performance Distributed Computing (HPDC '10). ACM, New York, NY, USA, 810–818.

[18] Jaliya Ekanayake, Hui Li, Bingjing Zhang, Thilina Gunarathne, Seung-Hee Bae, Judy Qiu, and Geoffrey Fox. 2010. Twister: A Runtime for Iterative MapReduce. In Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing (HPDC '10). ACM, New York, NY, USA, 810–818.

[19] Hamilton Wilfried Yves Adoni, Tarik Nahhal, Moez Krichen, A survey of current challenges in partitioning and processing of graph-structured data in parallel and distributed systems,Springer,june2020 , Distributed and Parallel Data Database.

[20] Charu C. Aggarwal, Haixun Wang," Managing and Mining Graph Data", IOP Conference Series Materials Science and Engineering 180(1):012065, March 2017.

[21] Minyang Han, Khuzaima Daudjee, Khaled Ammar, M Tamer Ozsu, Xingfang Wang, and Tianqi Jin. 2014. An Experimental Comparison of Pregellike Graph Processing Systems. Proceedings of the VLDB Endowment 7, 12 (2014), 1047–1058.

[22] Ribeiro, P., and Silva, F(2014),'G-Tries: A data structure for storing and finding subgraphs.'Data Mining and Knowledge Discovery 28, 2 (2014).

[23] Saeed Salem,Mohammed Alokshiya, Mohammad Al Hasan,RASMA: a reverse search algorithm for mining maximal frequent subgraphs,BMC Part of Springer nature,march 16,2021.

[24] C. Sakouhi, S. Aridhi, A. Guerrieri, S. Sassi, and A. Montresor, "Dynamicdfep: A distributed edge partitioning approach for large dynamic graphs," in Proceedings of the 20th International Database Engineering & Applications Symposium. ACM, 2016, pp. 142–147.

[25] D. Dai, W. Zhang, and Y. Chen, "Iogp: An incremental online graph partitioning algorithm for distributed graph databases," in Proceedings of the 26th International Symposium on High-Performance Parallel and Distributed Computing. ACM, 2017, pp. 219–230.

[26] Sergey Edunov,Dionysio Logothesis,Cheng Wang,Avery Ching and Maja Kabiljo,"Generating synthetic social graphs with Darwini" IEEE International Conference on Distributed Computing Systems, 2018.

Authors Profile

Ms Sadhana Priyadarshini is a Phd scholar in Department of Computer Science and Engineering at GITAM (Deemed to be University), Vishakhapatnam, India. She completed MTech (CSE) from SQA University in 2010.Her research interests in field of Data Mining.

Dr. Sireesha Rodda is a Head of Department of Computer Science & Engineering, GITAM (Deemed to be University). She has 20 years of research experience in the fields of Artificial Intelligence, Data Mining and Machine Learning. She has more sthan 45 papers published in referred journal.