ISSN (ONLINE): 2349-7084
GLOBAL IMPACT FACTOR 0.238
ISRA JIF 0.351

INTERNATIONAL JOURNAL OF COMPUTER ENGINEERING IN RESEARCH TRENDS
VOLUME 2, ISSUE 5, MAY 2015, PP 288-292

Social Mining to Improve the Computational Efficiency Using MapReduce

Ms. Babbitha.M[1] Dr. Angelina Geetha[2] Mr. Mohammed Jaffer.A.R[3]

[1]*M.Tech, Software Engineering (CSE), B.S. AbdurRahman University, Chennai, India*
[2]*Associate Professor, Computer Science and Engineering, B.S. AbdurRahman University, Chennai, India*
[3]*M.Tech, Software Engineering (CSE), B.S. AbdurRahman University, Chennai, India*

Abstract-Graphs are widely used in large scale social network analysis. Graph mining increasingly important in modelling complicated structures such as circuits, images, web, biological networks and social networks. The major problems occur in this graph mining are computational efficiency (CE) and frequent sub graph mining (FSM). Computational Efficiency describes the extent to which the time, effort or efficiency which use computing technology in information processing. Frequent Subgraph Mining is the mechanism of candidate generation without duplicates. FSM faces the problem on counting the instances of the patterns in the dataset and counting of instances for graphs. The main objective of this project is to address CE and FSM problems. The paper cited in the reference proposes an algorithm called Mirage algorithm to solve queries using sub graph mining. The proposed work focuses on enhancing An Iterative MapReduce based Frequent Subgraph Mining Algorithm (MIRAGE) to consider optimum computational efficiency. The test data to be considered for this mining algorithm can be from any domains such as medical, text and social data's (twitter).The major contributions are: an iterative MapReduce based frequent subgraph mining algorithm called MIRAGE used to address the frequent subgraph mining problem. Computational Efficiency will be increased through MIRAGE algorithm over Matrix Vector Multiplication. Performance of the MIRAGE will be demonstrated through different synthetic as well as real world datasets. The main aim is to improvise the existing algorithm to enhance Computational Efficiency.

Index Terms—Computational Efficiency, Data Mining, Frequent SubgraphMining, Graphs, Map Reduce, Text Mining, Social Networks.

1. INTRODUCTION

1.1 OVERVIEW

Data mining is the computational process of discovering patterns in large datasets involving methods at the intersection of artificial intelligence, machine learning, statistics, and database systems. Data Mining is an analytic process designed to explore data (usually large amounts of data typically business or market related are known as "big data") in search of consistent patterns and/or systematic relationships between variables, and then to validate the findings by applying the detected patterns to new subsets of data. The process of data mining consists of three stages: the initial exploration, model building or pattern identification with validation/verification and deployment.

The overall goal of the data mining process is to extract information from a data set and transform it into an understandable structure for further use. It is used to extract patterns and knowledge from large amount of data. Aside from the raw analysis step, it involves database aspects, data pre-processing model and inference considerations, post-processing of discovered structures, visualization and online updating [1]. The actual data mining task is the automatic or semi-automatic analysis of large quantities of data to extract previously unknown interesting patterns such as groups of data records (cluster analysis), unusual records (anomaly detection) and dependencies (association rule mining).

1.2 OBJECTIVE

In recent years, online social network services such as Facebook and Twitter are becoming increasingly popular and have created huge amount of social network data. It is very difficult to store massive data in the computer memory. Many out-dated methods are not designed to handle massive amount of data. To address this problem all the outdate methods are re-designed under the computing framework that is well known of big data syndrome.

The main objective of the FSM is to extract the entire frequent subgraph in the given data set, whose occurrence counts will be specified above the specified threshold. This completely focus on effective mechanism for generating candidate subgraph (without duplicates) and to identify the frequent subgraph. Computational Efficiency (CE) emphasis on the efficiency extends to which time, effort or space is well used for the intended task [2]. A scalable method for frequent sub graph mining and computation efficiency is of huge demand since many disciplines such as social networks, bioinformatics, cheminformatics and semantic web are much focused on this Frequent SubGraph Mining and Computation Efficiency factors.

1.3 SCOPE OF THE PROJECT

MIRAGE is the MapReduce algorithm for the frequent subgraph mining. This is mainly used for the creation of complete set of frequent subgraph for a given minimum support threshold. In map phase, it builds and recalls all patterns that have non-zero and in reducer phase it decides on which the pattern is frequent by aggregating their support through different computing nodes in order to ensure completeness. Mirage runs in an iterative manner such that output of the reducers of iteration i-1(where i denotes number of terms) is used as an input for the mappers in the iteration i where it will also compute the local support of candidate pattern. Reducers i then find the true frequent subgraph by aggregating their local supports [3]. The proposed system will perform the data mining in an efficient way using the algorithm. An overview of the proposed work contains the below modules: Data Collection, Removal of duplicate sets, Establishing Iterative MapReduce framework and Comparison and analysis of results.

The major contributions are: an iterative MapReduce based frequent subgraph mining algorithm called MIRAGE used to address the frequent subgraph mining problem. Computational Efficiency will be increased through MIRAGE algorithm over Matrix Vector Multiplication. Performance of the MIRAGE will be demonstrated through different synthetic as well as real world datasets. The main aim is to improvise the existing algorithm to enhance Computational Efficiency.

The Section 2 gives the related research work. Section 3 discusses the materials and methodologies and Section 4 presents our results and discussed them in section 5.Section 6 concludes the paper with future enhancements.

2. RELATED RESEARCH

Mansurulet al. [1] in their paper proposed the new algorithm for the frequent subgraph mining that address the key mechanism of candidate subgraph and is used to identify the subgraph. This paper clearly illustrates the Iterative MapReduce based algorithm to rectify the Frequent Subgraph Mining (FSM) problem in a very efficient manner. This algorithm gives the appropriate way to identify the frequent dataset and removes duplication. Social Graph Mining uses the same MIRAGE algorithm to address the Computational Efficiency (CE) problem.

Yi-Chen Lo et al. [2] in their paper represented the current need of the computational efficiency in mining large scaled social networks. This work presents the Computational Efficiency problem through the open source graph mining library called MapReduce Graph Mining Framework (MGMF). It deals with the large scaled social network mining tasks containing billions of entities where cloud computing is the solution. Author completely uses Matrix Vector Multiplication algorithms to resolve the Computational Efficiency problem.

SabaSehrish et al. [3] in their paper discussed about the high performance computing problems through MapReduce with Access Patterns (MRAP) which will be a unique combination of the data access semantics and the programming framework used in implementing High Performance Computing (HPC) analytics application. This paper is referred to know the basic ideas of scheduling in MapReduce.

3. MATERIALS AND METHODS

3.1 MAP REDUCE MODEL

Map Reduce, proposed by Google, is a distributed model for processing large-scale data. Users specify a map function and a reduce function. MapReduce takes in a list of key value pairs, splits them among the possible map tasks and then each map function produces any number of intermediate key-value pairs. Pairs with similar keys are gathered together at the reduce tasks, and then each reduce function performs computations before outputting values, which are either the final results, or possibly input for the next iteration. Ideally, MapReduce frameworks consist of several computers, usually referred to nodes, on the scale of tens to thousands. Processing occurs on data stored in the file system. Computation should be parallelized across the cluster, fault tolerant, and scheduled efficiently.

3.2 MAP FUNCTION

The mapper’s job is to take in a key-value pair. This key-value pair often comes from a partition of data specified by the Map Reduce architecture. After processing, the map function will emit another key-value pair. An added bonus comes in the form of an in-mapper combiner, which can do local computations to lessen the burden on the file system by acting as a mini-reducer. After all mappers have finished, all of the results are shuffled, sorted, and sent to the reducers.Hadoop sends single lines from the input file to the mappers, to which each applies a map function to those lines.

3.3 REDUCE FUNCTION

The reducer takes in a list of values corresponding to a specific key. Here, the reduce function can perform many operations, such as aggregations and summations. Since all the values we need have been grouped, bulk computations on those values become trivial.

3.4 REDUCER FOR CONSTRUCTING SUBGRAPHS

Subgraphs of size k − 1 with the same graph id are gathered for the reducer function. Note all of the single edges in these subgraphs and use that information to generate the next generation of possible subgraphs of size k. Encodes this subgraph as a string just as was outputted from the previous map function. All labels are alphabetized and use special markers to designate differing nodes with the same labels. The results of this step are written out to the Hadoop File System.

3.5 MAP FUNCTION FOR GATHERING SUBGRAPH STRUCTURES

Similar to the process involving the first map function, Hadoop sends lines of input to the mappers. This second map function will have the responsibility of outputting the label-only subgraph encodings as a key and the node identification numbers and graph ids as values.

3.6 ARCHITECTURE AND MODULE DESIGN

The basic architecture diagram of the entire system is given in figure 3.1.The above architecture illustrates the complete flow of social graph mining. Dataset are given as graph input data (social media, biological dataset) .Graph data’s are used to perform the data mining process. The process starts with frequent subgraph mining where all the duplicate sets are removed. MapReduce exactly perform mapping and reducing functions with the file system. Then FSM with mining process is carried out with the MIRAGE algorithm with three different phases such as partition phase, preparation phase and mining phase. FSM analysing is carried out for three different factors candidate generation, graph isomorphism and support counting. Final output is produced with the comparison result that increases computational efficiency.

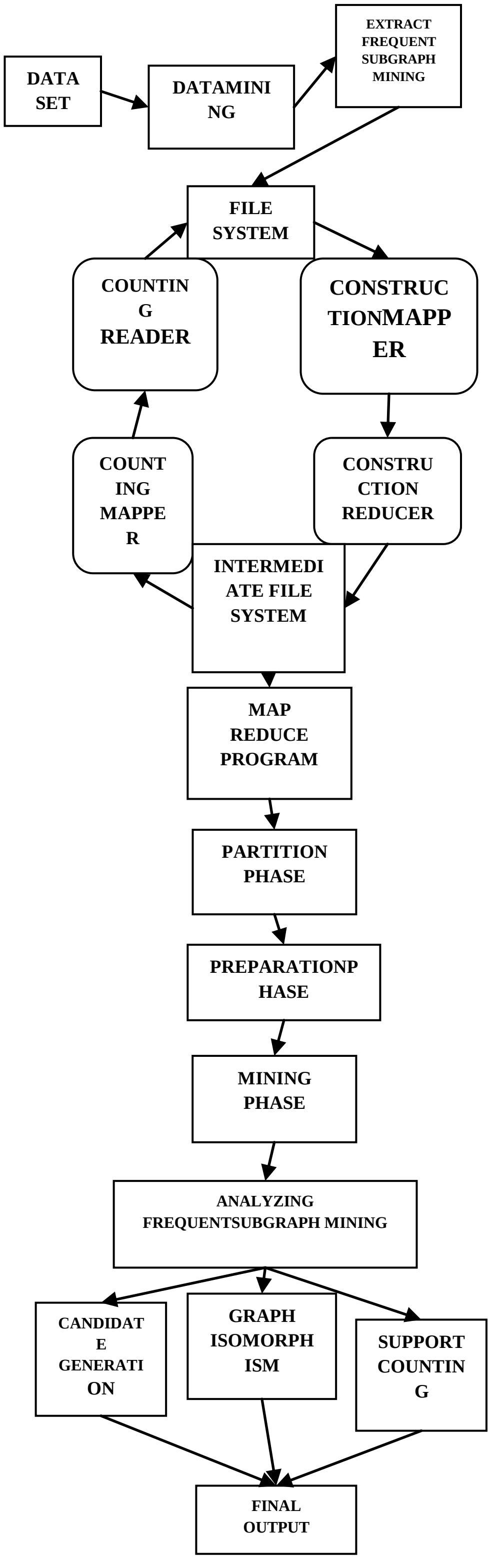
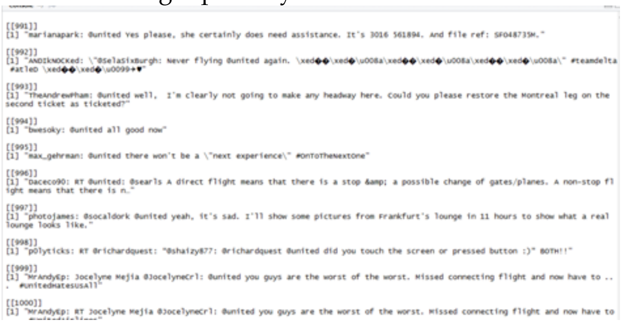


Figure 3.1 System Architecture

4. EXPERIMENTAL RESULTS

4.1 COLLECTGRAPH DATA

This file contains the synthetic datasets and real world large graph dataset (social media). The graph data's are collected from twitter social media networking whereas synthetic data's are collected from the UCI machine learning repository.



```
[[991]]
[1] "marianapark: @united Yes please, she certainly does need assistance. It's 3016 561894. And file ref: SF048735M."

[[992]]
[1] "ANDIKNOCKed: \"@SelaSixBurgh: Never flying @united again. \xed��\xed�\u008a\xed��\xed�\u008a\xed��\xed�\u008a\" #teamdelta
 #atleD \xed��\xed�\u0099+♥"

[[993]]
[1] "TheAndrewPham: @united well,  I'm clearly not going to make any headway here. Could you please restore the Montreal leg on the
second ticket as ticketed?"

[[994]]
[1] "bwesoky: @united all good now"

[[995]]
[1] "max_gehrman: @united there won't be a \"next experience\" #OnToTheNextOne"

[[996]]
[1] "Daceco90: RT @united: @searls A direct flight means that there is a stop &amp; a possible change of gates/planes. A non-stop fl
ight means that there is n…"

[[997]]
[1] "photojames: @socaldork @united yeah, it's sad. I'll show some pictures from Frankfurt's lounge in 11 hours to show what a real
lounge looks like."

[[998]]
[1] "p0lyticks: RT @richardquest: "@shaizy877: @richardquest @united did you touch the screen or pressed button :)" BOTH!!"

[[999]]
[1] "MrAndyEp: Jocelyne Mejia @JocelyneCrl: @united you guys are the worst of the worst. Missed connecting flight and now have to ..
.  #UnitedHatesUSAll"

[[1000]]
[1] "MrAndyEp: RT Jocelyne Mejia @JocelyneCrl: @united you guys are the worst of the worst. Missed connecting flight and now have to
```

Figure 4.1 Dataset Preparation

Figure 4.1 shows the dataset preparation of the real time twitter data's.

4.2 REMOVAL OF DUPLICATE SETS

Frequent Sub graph Mining is a relation between the object's elements that is recurringover and over again which are represented as patterns. FSM will generate candidate sub graphs (without generating duplicates).

4.3 ESTABLISHING ITERATIVE MAPREDUCE FRAMEWORK

Frequent sub graph mining is a very well-studied area in graph mining research because of its wide range of applications in the above areas. Frequent patterns can help understand different functions and relations. For example, in a protein-protein interaction network (PPI), a frequent pattern could uncover unknown functions of a protein. Similarly, in a social network, a frequent pattern could show a friend clique. There are two different aspects of mining frequent subgraphs. The first category deals with a single large graph. The second category deals with a set of graphs. Memory-based algorithms do fairly well on small datasets, but as the data size increases, memory becomes a bottleneck. It progress the MapReduce programming with three phases.

PARTITION PHASE

In this phase input graph data will be divided into many partitions. It then performs the filtration of data's. In data partition phase, MIRAGE splits the input graph dataset (G) into many partitions. One straightforward partition scheme is to distribute the graphs so that each partition contains the same number of graphs from G. This works well for most of the datasets. During the partition phase, input dataset also goes through a filtering procedure that removes the infrequent edges from all the input graphs.

PREPARATION PHASE

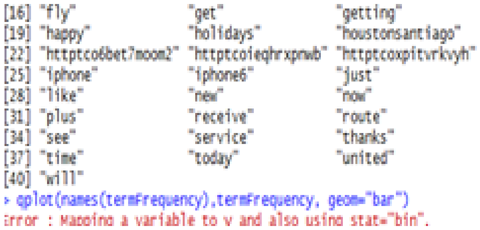
Mappers in this phase prepare some partitions specific data structures. This data structure is edge-extension-map. Reducer in this phase does nothing but write input key value pairs. The mappers in this phase prepare some partition specific data structures such that for each partition there is a distinct copy of these data structures. The first of such data structure is called edge-extension-map, which is used for any candidate generation that happens over the entire mining session. The second data structure is called edge; it stores the occurrence list of each of the edges that exist in a partition. Note that, since the partition phase have filtered out all the infrequent edges, all single edges that exist in any graph of any partition is frequent. Mappers in the preparation phase compute the min-dfs-code and create the pattern object for each single-edge patterns.

MINING PHASE

In this phase, mining process discovers all possible frequent subgraphs through iteration. Preparation phase populates all frequent subgraphs of size one and writes it in the distributed file system. It follows till n frequent patterns. In this phase, mining process discovers all possible frequent subgraphs through iteration. Preparation phase populates all frequent sub graphs of size one and writes it in the distributed file system.

CANDIDATE GENERATION

Candidate generation produce the frequent subgraphs without duplication. The joining of two frequent subgraphs can lead to multiple candidate sub graphs. Based on the parent-child relationship the set of candidate patterns of a mining task in a candidate generation tree can be arranged as like the below figure 4.2.



```
[16] "fly"               "get"               "getting"
[19] "happy"             "holidays"          "houstonsantiago"
[22] "httptco6bet7moom2" "httptcoieqhrxpnwb" "httptcoxpitvrkvyh"
[25] "iphone"            "iphone6"           "just"
[28] "like"              "new"               "now"
[31] "plus"              "receive"           "route"
[34] "see"               "service"           "thanks"
[37] "time"              "today"             "united"
[40] "will"
> qplot(names(termFrequency),termFrequency, geom="bar")
Error : Mapping a variable to y and also using stat="bin".
```

Figure 4.2 Candidate generation

SUBGRAPH ISOMORPHISM

Subgraph Isomorphism performs redundancy check. It obviously reduces the generation of same subgraph many number of times. It also downloads closure property. It also used for checking containment of a frequent subgraph.

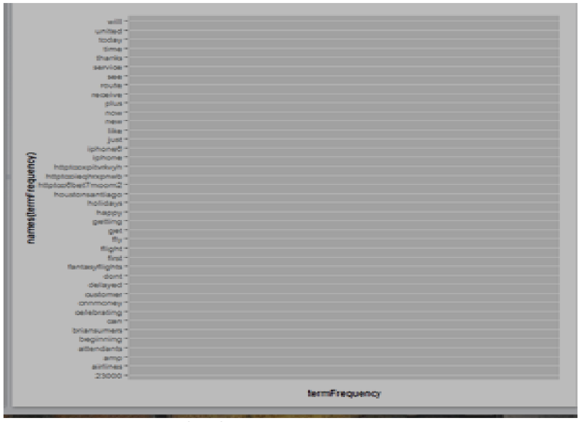


Figure 4.3 Bar Plot for frequent terms

Figure 4.3 describes the bar plot for the frequent terms that occurred in the overall data corpus.

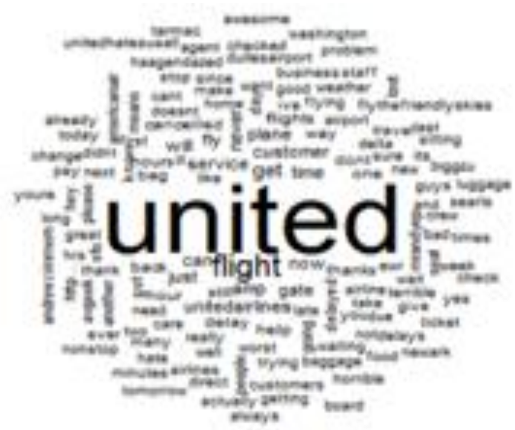


Figure 4.4 Word Cloud

Figure 4.4 illustrates the word cloud of the frequent terms identified in the data corpus.

5. COMPARITIVE ANALYSIS

The comparison of synthetic dataset and real world social dataset performance will be measured. These experimental results will be analyzed for the different runtime of MIRAGE.

Runtime of MIRAGE for different minimum support is conducted for biological datasets.

Runtime of MIRAGE for different number of database Graphs are analyzed through four different synthetic datasets.

Runtime of MIRAGE on varying number of data nodes are observed through Yeast dataset.

6. CONCLUSION

The existing system needs for more improvements in time and space complexity. It specifically provides solution for matrix vector multiplication based algorithms. It cannot be used with the other graph mining algorithm such as MIRAGE, Betweenness / closeness centrality and social network generation. This paper shows the enhancement of computation efficiency of graph data using matrix vector multiplication (MVM) method over Mirage algorithm.Thus the computational efficiency of the graph data would be increased using the MIRAGE over MVM and the speed of the social network will be increased by map reduce technique. In this paper we present a novel iterative MapReduce based frequent subgraph mining algorithm, called MIRAGE. We show the performance of MIRAGE over real life and large synthetic datasets for various system and input configurations. We also compare the execution time of MIRAGE with an existing method, which shows that MIRAGE is significantly better than the existing method.

ACKNOWLEDGEMENT

First offer our deepest gratitude to the almighty who has given us the strength and good health during complete this paper. I express profound gratitude to our respected dean Dr. K.M MEHATA Ph.D., for his encouragement and guidance. I articulate our gratitude to our head of the department Dr. SHARMILA SANKAR M.E.,Ph.D., who had taught me the way to do successful paper. I gratitude to my respected internal coordinator Dr. ANGELINA GEETHA M.C.A,,M.E.,Ph.D., who had the novel ideas for this paper and helped me a lot and given valuable suggestions to finish this paper in very successful and neat way. Without whose constant encouragement and whole idea, the paper would not have been possible.

REFERENCES

[1] Mansurul A Bhuiyan and Mohammad Al Hasan, “MIRAGE: An Iterative MapReduce based Frequent Subgraph Mining Algorithm”, ACM Computing Research Repository, arXiv: 1307.5894, Volume 1, 2013.

[2] Yi-Chen Lo, Hung-CheLai, Cheng-Te Li and Shou-De Lin,” Mining and Generating Large Scaled Social Networks via MapReduce”, Springer-Verlag Advances in Social Networks Analysis and Mining, pp - 1449–1469, 2013.

[3] SabaSehrish, Grant Mackey, Pengju Shang, Jun Wang and John Bent,”Supporting HPC Analytics Applications with Access Patterns Using Data Restructuringand Data-Centric Scheduling TechniquesinMapReduce” IEEE Transactions on Parallel and Distributed Systems, Volume 24, 2013.