Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 48 (2015) 686 – 691

Procedia Computer Science

International Conference on Intelligent Computing, Communication & Convergence

(ICCC-2015)

Conference Organized by Interscience Institute of Management and Technology,

Bhubaneswar, Odisha, India

A NOVEL APPROACH FOR ANALYZING THE SOCIAL NETWORK

Saranya Balaguru [1], Rachel Nallathamby[2], C.R.Rene Robin[3]

[1]*PG Student, Jerusalem Engineering College, Chennai*

[2]*Research Scholar, Jerusalem Engineering College, Chennai*

[3]*Professor and Head of the Department, Jerusalem Engineering College, Chennai*

Abstract

Massive datasets are becoming more prevalent. In this paper, we propose an algorithm to process a large symmetric matrix of billion scale graph in order to extract knowledge from graph dataset. For example, interesting patterns like the people who frequently visit your page and the most number of participating triangles can be obtained using the algorithm. These interesting patterns are discovered by computation of several eigen values and eigen vectors. The main challenge in analyzing the graph data are simplifying the graph, counting the triangles, finding trusses. These challenges are addressed in the proposed algorithm by using orthogonalization, parallelization and blocking techniques. The proposed algorithm is able to run on highly scalable MapReduce environment. we use a social network dataset (facebook approximately 2 to 7 TB of data) to evaluate the algorithm. we also show experimental results to prove that the proposed algorithm scale well and efficiently process the billion scale graph.

*Keywords:*Big data; Graph Mining; Social Network Analysis.

1. Introduction

The Internet, the World Wide Web (WWW), social networks, protein interaction networks and many structures are modeled as graph. Graph is a representation of relationship collection which provide the human interpretation and

Peer-review under responsibility of scientific committee of International Conference on Computer, Communication and Convergence (ICCC 2015)
doi:10.1016/j.procs.2015.04.202

simple mechanical analysis. Graphs with billions of nodes and edges are more common, for example social network like Facebook , Twitter are having billions of active users in a large countries. Given a billion scale graph , finding interesting pattern[1] is a more challenging task. The interesting patterns in the graph are discovered by the computation of several eigen value and eigen vector. The main challenge[2] in analyzing the graph data are

- Simplifying the graph
- Counting the triangles
- Finding Trusses

The proposed algorithm called Multi-Dimensional Parallel Eigen Solver(MPES) addresses these challenges and able to run on highly scalable MapReduce environment.

1.1 Simplifying the graph

The analysis of graph starts from simplifying the graph that means removing duplicate edges and loops. For example ,In an undirected graph the edge (A,B) duplicates (B,A). A single MapReduce job is enough to simplify the graphs. First step is removing the loops and the MapReduce function finds the duplicate edges.

1.2 Counting triangle

The basis for analyzing the graph and subgraph is locating triangle in the graph. Triangles are very important in analyzing the complex network and also play a significant role in graph mining applications. Triangles are most basic and non-trivial subgraph. for example, In facebook friend of friend tend to become friend themselves and forms a triangle. This phenomenon is also observed in other types of networks such as biological and online network etc. The problem of counting the triangle [3],[4],[5] is addressed in the proposed algorithm.

1.3 Finding trusses

Trusses are subgraphs with high connectivity. Cliques and triangles are captures by identifying the trusses. If k-trusses are in the graph then it is proved that there should k-member cliques and k-1 triangles. Simplifying the graph is the first step to find trusses, triangles etc. To discover all these pattern, a promising tool is parallelism and more specifically MapReduce[6].

2. Related work

The related work forms two groups.

- Graph Mining
- MapReduce

2.1 Graph mining:

There are enormous number of graph mining algorithm are there to compute the eigen value and eigen vector and to find many interesting patterns. The tasks performed by graph mining algorithms are computation of communities(eg [7],[8]),subgraph discovery(eg [9],[10]), identifying a single node(eg PageRank[11]),counting number of triangle[12] and too-many. Most of the algorithm do not scale with billions of nodes and edges. There are several algorithms for connected components like Breadth-First search, Depth First Search and Random search. These algorithms are limited with shared memory model. The base for all algorithms are finding several eigen value and eigen vector.

2.2 Mapreduce:

MapReduce is programming paradigm for processing enormous amount of unstructured, semi structured data in parallel[13],[14]. The advantages of MapReduce are the concepts is familiar and data distribution, replication , load balancing are handled automatically. There are only two functions map and reduce need to be defined.

- the map stage reads the input and produces the (key, value) pairs
- the shuffling stage sorts the output and sends it to reduce stage
- the reduce stage processes the (key, value) pairs and produces single (key, value) pairs as a final output.

3. Proposed method

The main aim is to design a eigen solver that finds the top-k eigen values of large symmetric matrix formed from billion scale graph in a parallel environment. The successful parallel environment for processing the web scale graph is HADOOP([15],[16],[17]). The design of eigen solver needs careful attention in choosing the algorithm. We choose the sequential method and design it in such a way that it will run in parallel. Orthogonalization[18] [19], parallelization and blocking techniques are incorporated in the eigen solver to scale well. The proposed system architecture is shown in Figure 1.

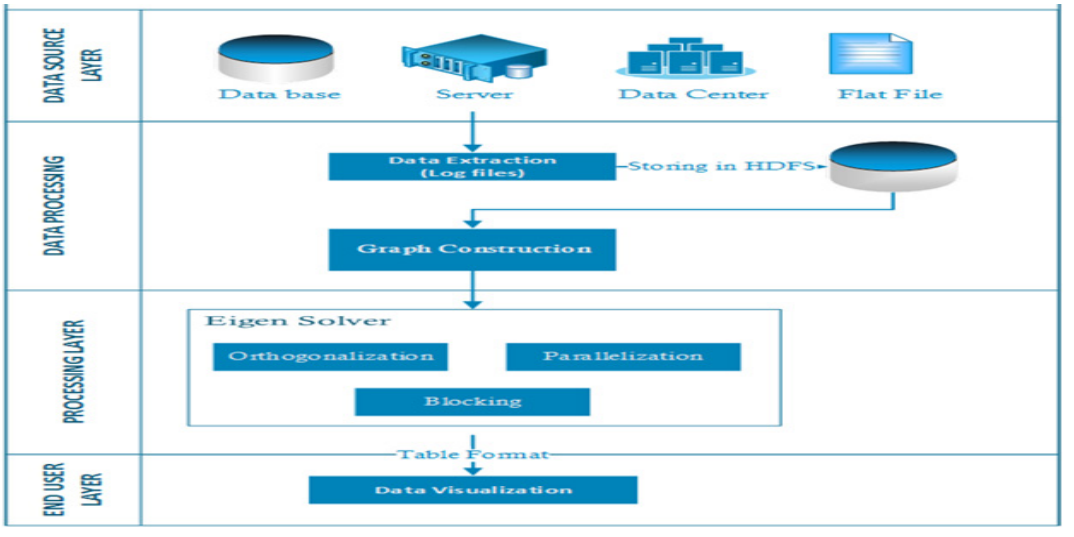

Figure 1: Architecture of Hadoop Eigen Solver.

3.1 Orthogonalization

Orthogonalization of vectors gives good approximation to find the top several eigen value and eigen vectors. The main idea of orthogonalization is as follows: The first step is choosing a random basis vector a. In each iteration new basis vector is computed and that is orthogonalized with the previous vector. Let y be the number of iteration, X_y be the x × y matrix and A be the matrix whose eigenvalues we need to compute. The proposed algorithm(MPES) is shown in Algorithm 1.

Algorithm 1. Multi-Dimensional Parallel Eigen Solver

Input: Matrix $A^{x \times x}$, vector a,

Number of iteration y,

Number of eigen values n

Output: n eigenvalues $\lambda_{1...n,}$

Eigen Vector.

1: $\alpha_0 \leftarrow 0$, $v_0 \leftarrow 0$, $v_1 \leftarrow a$;

2: **for** i = 1, ..y **do**

3: $v \leftarrow Av_i$;

4: $\alpha i \leftarrow v_i^T v$

5: $v \leftarrow v - \alpha_{i-1} v_{i-1} - \alpha_i v_i$; //Orthogonalization

6: $\alpha i \leftarrow ||v||$;

7: **if** $\alpha i = 0$ **then**

8: break for loop;

9: **end if**

10: $vi+1 \leftarrow v/\alpha i$;

11:end for

12: $\lambda 1::n \leftarrow$ top n diagonal elements of the Matrix// Eigenvalues

13:$Av=\lambda v$ // Eigenvector

3.2 Parallelization:

Parallelization is achieved by processing the graph using the proposed algorithm in MapReduce environment. Parallelization means running all the operations on multiple machines in parallel. Parallelizing the operations is shown in Algorithm 2.

Algorithm 2: Parallelization

Input: Matrix $M = \{(src, (dst,val))\}$,

vector $v_i = \{(id, vval)\}$

Output: vector x

1: Map(key k, value v, Vector y):

2: $src \leftarrow k$;

3: $(dst,val) \leftarrow v$;

4: Output(src, ($val \times x[dst]$));

5: Reduce(key k, values V []):

7: $s \leftarrow 0$;

8: **for** $v \in V$ **do**

9: $s \leftarrow s + v$;

10: **end for**

11: Output(k, s); // Output a vector element

3.3 Blocking

Blocking is used to minimize the network traffic by reducing the volume of information exchanged between the nodes. In HADOOP, each line of input is assumed as a single map function. We divide the large matrix in to blocks , one key value of each block is single line of map function. Likewise, each blocks are assigned with map function and run in parallel. The processing of individual elements takes time than running it in blocks. So, the blocks are processed more faster and more efficiently.

4. Conclusion:

We have presented an efficient algorithm to process a billion scale graph and to analyze a real world graph to identify important patterns. Our approach that is Multi-Dimensional Parallel Eigen solver(MPES) is applicable in large dataset with relationship connectivity. The main contribution is to improve the scalability, flexibility and efficiency compared to other eigen solver. In Future, we can extend our analysis with multiple relationships of the participating node that is tensor based analysis.

References:

[1] U.Kang , C.Faloutsos ,"Big Graph Mining: Algorithm and Discoveries" , SIGKDD Volume 14, 2012 , pp 29-36.

[2] Dunren Che,Mejdl Safran and Zhiyong Peng,"From Big Data to Big Data Mining: Challenges, Issues and Opportunities“ , Database Systems for Advanced Applications Lecture Notes in Computer Science Volume 7827, 2013, pp 1-15.

[3] C. E. Tsourakakis, “Fast counting of triangles in large real networks without counting: Algorithms and laws,” in ICDM, pp. 608–617, 2008.

[4] L. Becchetti, P. Boldi, C. Castillo, and A. Gionis. Efficient semi-streaming algorithms for local triangle counting in massive graphs. In Proceedings of ACM KDD, Las Vegas, NV, USA, August 2008.

[5] L. S. Buriol, G. Frahling, S. Leonardi, A. Marchetti-Spaccamela, and C. Sohler. Counting triangles in data streams. In PODS '06: Proceedings of the twenty-fifth ACM SIGMOD-

SIGACT-SIGART symposium on Principles of database systems, pages 253{262, New York, NY, USA, 2006. ACM.

[6] J. Dean and S. Ghemawat, "Mapreduce: Simplified data processing on large clusters," OSDI, 2004.

[7] J. Chen, O. R. Zaiane, and R. Goebel, "Detecting communities in social networks using max-min modularity," *SDM*, 2009.

[8] T. Falkowski, A. Barth, and M. Spiliopoulou, "Dengraph: A density-based community detection algorithm," *Web Intelligence*, 2007.

[9] S. Ranu and A. K. Singh, "Graphsig: A scalable approach to mining significant subgraphs in large graph databases," ICDE, 2009.

[10] J. Cheng, J. X. Yu, B. Ding, P. S. Yu, and H. Wang, "Fast graph pattern matching," *ICDE*, 2008.

[11] S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine," c*omputer networks and ISDN systems*, vol. 30, no. 1-7, pp. 107–117, 1998.

[12] C. E. Tsourakakis, U. Kang, G. L. Miller, and C. Faloutsos, "Doulion: Counting triangles in massive graphs with a coin," *KDD*, 2009.

[13] G. Aggarwal, M. Data, S. Rajagopalan, and M. Ruhl, "On the streaming model augmented with a sorting primitive," *Proceedings of FOCS*, 2004.

[14] S. Papadimitriou and J. Sun, "Disco: Distributed co-clustering with map-reduce," *ICDM*, 2008.

[15] U. Kang, C. E. Tsourakakis, A. P. Appel, C. Faloutsos, and J. Leskovec, "Radius plots for mining tera-byte scale graphs: Algorithms, patterns, and observations," in *SDM*, 2010, pp. 548–558.

[16] U. Kang, C. Tsourakakis, and C. Faloutsos, "Pegasus: A petascale graph mining system - implementation and observations," *ICDM*, 2009.

[17] "Hadoop information," http://hadoop.apache.org/.

[18] K. Wu and H. Simon, "A parallel lanczos method for symmetric generalized eigenvalue problems," *Computing and Visualization in Science*, 1999.

[19] G. H. Golub and C. F. V. Loan, "Matrix computations," *Johns Hopkins University Press*, 1996.