ISSN: 2321-7782 (Online)
Volume 3, Issue 7, July 2015
International Journal of Advance Research in Computer Science and Management Studies
Research Article / Survey Paper / Case Study
Available online at: www.ijarcsms.com

User load balancing based on Hypergraph Community partitioning in social network

Vishwas Punjaji Badhe[1]
Student of ME Computer
G. S. Moze College of Engineering
Pune, India

Pradnya Velhal[2]
Assistant Professor
G. S. Moze College of Engineering
Pune, India

Abstract: The Major issue of User Load balancing is commonly observed in Social Networking Applications. The proposed system will handle User load balance issue by optimal workload partitioning to improve on efficiency parameters like accuracy, Community degree, time analysis. The Proposed system will dynamically analyze the Multi-way relations in Multi-user operational system as well as this algorithm effectively implements unique partitioning method. To handle the scalability issue, new users coming in the system is allocated to server efficiently. In account of performance issue effective allocation of users and balancing User workload could be considered as two sides of one issue. The proposed system is to be implemented to form communities in hyper graph. Firstly the users are classified on the basis of their degree of communication. The multiway communication is taken in consideration for formation of communities. These communities are then separated. Ultimately the Community hyper graph analysis (CGA) system will minimize total response time of social network system and also provides one convenient & secure way for the user transactions.

Keywords: User Load balancing, optimal workload Partitioning, Multi-way relations in data, communities in hypergraph, user allocation, Communication Degree (CD).

I. INTRODUCTION

In today's many online applications that have very elemental importance to database because whole application runs on data stored or real time data. Online social network is also a kind of application which have been increasing database availability requirement. Some database systems like Not only SQL [2] implements data partitioning and replication to achieve Scalability and availability [3] [4]. Database most of the time ignores the relations among the data and leads to redundant replications and significant communication overheads during query processing [2]. which degrades the performance. To overcome the problem of performance degradation number of approaches based on modeling the social network structure and user interactions have been proposed. These approaches emphasize on interactions between social network users via two way relations. Whereas some operations like multicasting the feeds to all connected users are more expensive than operations that only involve bilateral interactions.

This proposed system claims that hyper-graphs are more suitable for modeling these multi-user operations and multi way interactions. We can also say that performing partitioning in a single phase enables more accurate cost prediction and better load balancing. In this approach modeling the span (number of activated servers) is highlighted [5] [7].

We focus on hyper-graph for community formation in partitioning, and hence observe true balancing and cost estimations at the end of our partitioning scheme. In analysis, we observe that Server load imbalance the total number of servers processing a query (query span), have direct correlation with the performance of the system. Hence, we focus on these metrics for possible improvements in performance. After selective partitioning method for user distribution in social networks by utilizing the workload and time information. This approach values the time of interactions between users and targeted users are classified in different communities in hyper graph. We can say that simultaneous partitioning of this hyper-graph model can accurately target the objective of reducing the span of multi-user operations subject to load balance. After doing partitioning, system merges the communities with high degree of partition[20].

II. LITERATURE SURVEY

Many of NoSQL systems use either hash-based or range based or combination of both partitioning schemes. Rangebased partitioning divides key-space into ranges and each range is assigned to a server and if required this key space replicated to other servers. So this become an advantage of range-partitioning by which two consecutive keys are likely to come in the same partition, which is beneficial when range scan type queries are frequent. It generally maintains a map that stores information about which servers are responsible for which key ranges. Hash-based partitioning simply uses the hash of data to determine the responsible server for storing that data. Consistent hash rings are a blend of range- and hashbased partitioning schemes. Many of NoSQL [2] systems such as Cassandra, Dynamo, Voldemort, and Riak uses either hashbased or range-based scheme.

Servers in Cassandra cluster are arranged in ring format and data stored is according to ring analogy. Individual ring is divided into several ranges and each server is responsible for one or more ranges. As new server joins Cassandra, it is assigned a new token, which determines its position on the ring and the range of data, to which this server is responsible for. Cassandra uses Random Partitioning, which uses the MD5 hash of each row key to partition, and Ordered Partitioning, that uses row key for a sorter order in various servers. In database partitioning data items are represented via nodes, transactions are modeled via edges. The partitioning objective is to minimize the number of distributed transactions. The major key areas of proposed social network partitioning schemes are graph-partitioning, modular-optimization and random partitioning. There are different parameters to measured Partition qualities such as the number of internal messages or dialogs. Many times real time data is taken for test. For small partition counts, graph based approaches are shown to perform superior, whereas for large partition counts, modular optimization algorithms perform slightly better. Replication in this scheme enforces much replication and can lead to high I/O loads.

III. SYSTEM ARCHITECTURE

The proposed algorithm analyses synthetic communication among users to calculate degree of communication. Using this system perform Community formation in hypergraph[20]. The proposed system distributes users among available servers. After efficient allocation of user to server system focuses on the data which is generated or used by users. The synthetic communication of users then taken in consideration for the calculation of degree of communication. Basically proposed scheme determines individual load in distributed environment.

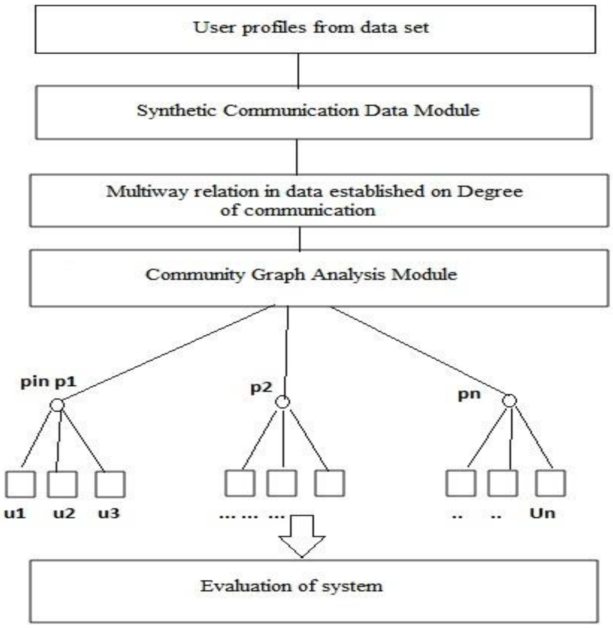
User profiles from data set
Synthetic Communication Data Module
Multiway relation in data established on Degree of communication
Community Graph Analysis Module
pin p1
p2
pn
u1
u2
u3
... ...
..
..
Un
Evaluation of system

Fig 1: Architecture of User load balancing system using Hyper graph

Above mentioned read/write operations take long time to execute so minimizing the total response time, balancing data load to different distributed servers, implement unique partitioning and selective replication method effectively, are some key objectives of proposed system.

Modules:

1. Dataset Module

The dataset module contains user profiles from soc-pokec-profiles.txt. A large number of user profiles is taken from dataset and user relations are taken from soc-pokec-relationships.txt. We can select required number of users (the number may be in thousands).

2. Synthetic communication module

The synthetic communication is established in this module. The Communication is of two types i.e. messages and call in social network.

3. Communication degree module

As we discussed in earlier section that User Profiles from dataset is selected and allocation of users to available servers is the first task. The users are randomly distributed among servers and the user load equilibrium is maintained while distribution.

After successful allocation of user to servers, system need to focus on data which users is either generating in the form of communication. The communication degree is calculated for each individual user. By which the load incurred in system by that user can be estimated. The direct and indirect communication degree(CD) is calculated on basis of direct and indirect communication pattern.

4. Hyper Community Graph Analysis

The community formation is based on entropy variable which is based on threshold communication between two users.

Analyzing the communication degree the merging of users initiated to form a community. The user with higher CD will be virtually at the center of community. As users are merged in community, similarly the users with lo CD are taken out i.e. they does not puts load on system.

5. Evaluation of System Performance

The Community Graph Analysis (CGA) Algorithm is tested against Hypergraph (HG) Algorithm and these results are as Follows:

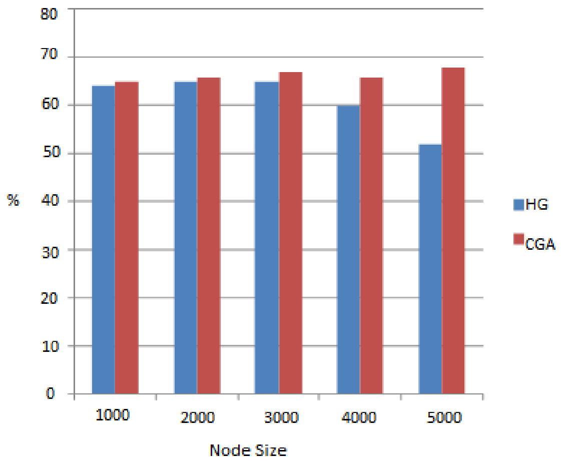

Fig 2: (a) Effect of increased load on accuracy
(Accuracy in % against node size)

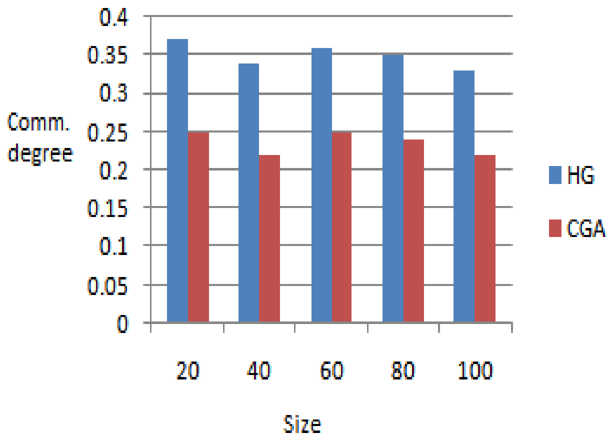

Fig 2: (b) Effect of increased load on Community degree
(Size Vs Community degree: CD is difference between common
Communication of two users, its good to be low i.e. CD id lower)

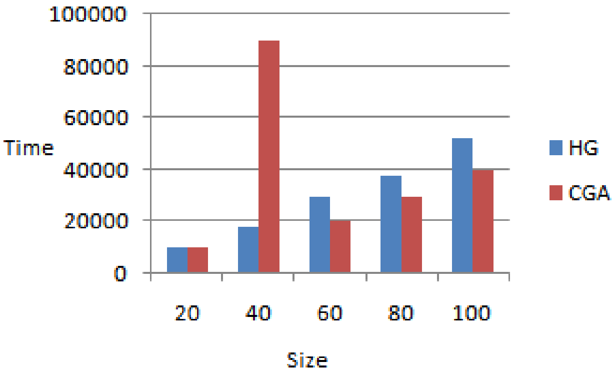

Fig 2: (c) Effect of increased load on Community Graph Analysis(CGA) Time
(Time Vs Size)

IV. ALGORITHM APPROACH

Step 1: The Users are Selected from Dataset and these users are distributed among available servers and total load on each server is calculated(i.e. The Load equilibrium is established). Total number of active servers is termed here as span of servers.

Step 2: users are selected and their synthetic communication is established in the form of Message and calls.

Step 3: The CD is calculated for each individual user to form a community (higher Message sender is treated as virtual center of the system). These user are given precedence according to their CD.

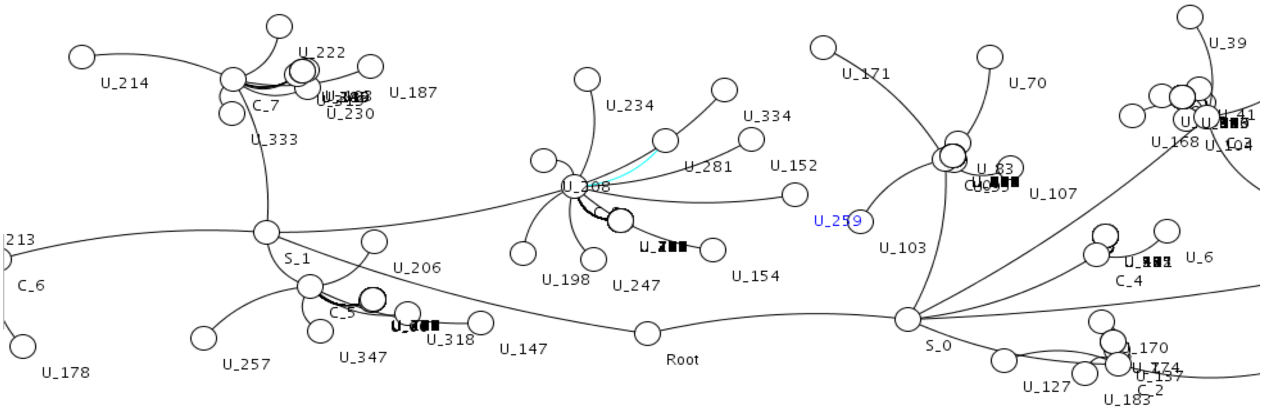

Fig 3: Community Hypergraph

Different users participate in read/write events in social network system. Here one virtual Root node is decided for community.

This Root contains available servers. In Fig 3, we have two servers *S0* and *S1* this user will contain newly formed communities under them. These Communities are dynamic i.e. they can changed as the CD [20] is updated for every individual.

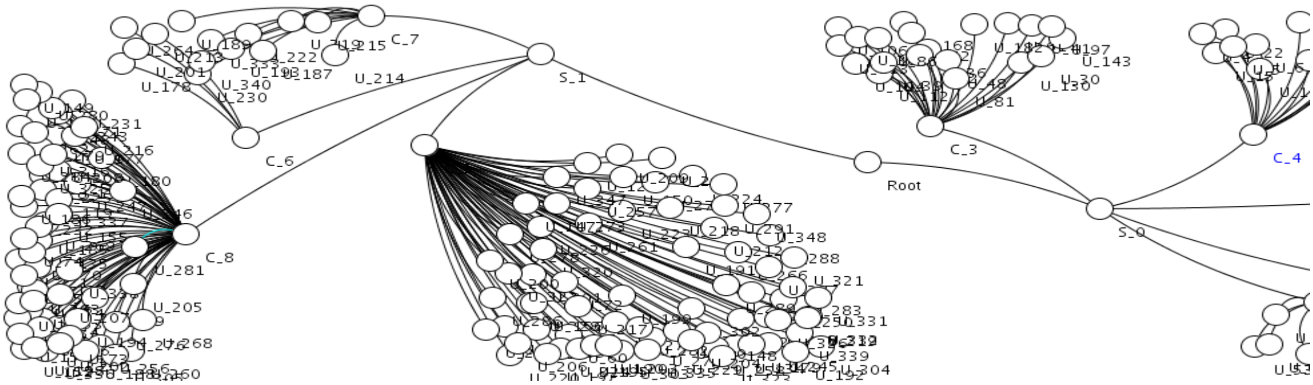

Fig 4: Community Hypergraph

Step 4: As shown in Fig 4: The Different Communities formed under server is given. The Precedence of users according to their CD is given. The user with Higher CD comes first in list.

V. Mathematical Modeling

The total load is distributed among servers Si is calculated according to threshold load on Si.

Hyper graph *H=(V,N)* is defined as a set *V* of vertices and a set *N* of nets (hyper edges), where each net connects a number of distinct vertices as well as the vertices connected by a net are said to be its pins (Pins(n_j)). A cost c(n_j) and a weight *w*(v_i) may be associated with a net n_j € N and a vertex v_i € *V*, respectively.

π = {V_1,V_2,… ,V_k} is said to be a *K*-way partition of a given hyper graph *H*, if parts are mutually disjoint and collectively exhaustive. In π , a net is said to connect a part if it has at least one pin in that part. Connectivity set ∧(V_k) of a net n_j is the set of parts connected by n_j:

$$(n_j\ ) = (V_k : V_k\ \cap Pins(n_j\ ) \neq \emptyset\ ) \qquad (1)$$

The connectivity λ(n_j) = |∧(n_j) |of a net n_j is the number of parts connected by n_j . In the Hyper graph Community, given a Hyper graph *H=(V,N)* is formed on the basis of CoEntropy variable and then precedence is assigned. If node *v* influences (activates) its neighbor *u (v and u* are both inactive at that time), we label the edge *evu* as live [21]. If *evu* is live and *v* belongs to community *Cm*, but *u* belongs to a different community *Cl*, we say that *u* is a live node of *Cm*. Let L[Cm] be the set of live nodes of *Cm*. The combination entropy of community *Cl to Cm* is defined as:

$$CoEntropy(C\mathcal{E}_m^l) = \max_{v\epsilon C_{m,u}\epsilon\ L[C_m],u\epsilon\ C_l} \frac{\overline{R_m}\ (\{u\})}{R_m(\{v\})} \qquad (2)$$

$$\frac{\overline{R_m}\ (\{u\})}{R_m(\{v\})}$$

In above ratio the Numerator denotes the influence degree increment of node u, and Denominator denotes the influence degree increment of node v in its community *Cm*. As we detect the communities based on diffusion model.

VI. Conclusion

User load balancing based on Community hyper-graph partitioning (CGA) model can be used for partitioning huge user community in social networks to reduce inter-server communication overhead. The proposed model basically focuses on multi-way interactions between users in social network. The issue of efficient assignment of users to distributed server is also focused in early steps of proposed system. The Community Graph Analysis (CGA) model improves results in terms of Accuracy, Communication Degree and Time analysis against simple Hyper graph (HG). The basic overhead in Hash-based approaches is communication overhead. Community graph partitioning approach can deal with maximum I/O in distributed environment, As the number of users increases in social network then HG performance reduces where as CGA handles huge user load efficiently.

Acknowledgement

With immense pleasure, I am presenting this Project research paper on “User load balancing based on Hypergraph Community partitioning in social network" as a part of the curriculum of M.E. Computer Engineering at G.S. MOZE COLLEGE OF ENGINEERING. It gives me proud privilege to complete this Project research paper Work under the valuable guidance of Prof. Mrs. Pradnya Velhal. I am also extremely grateful to Prof. Ratnaraj Kumar (HOD of Computer Department) and Principal for providing all facilities and help for smooth progress of Project Work. I would also like to thank my friends and my family members who have directly or indirectly guided and helped me for comlpetion of this work.

References

1. J. M. Pujol, V. Erramilli, G. Siganos, X. Yang, N. Laoutaris, P. Chhabra, and P. Rodriguez, “The little engine (s) that could: scaling online social networks,” ACM SIGCOMM Computer Communication Review, vol. 40, no. 4, pp. 375–386, 2010.
2. R. Hecht and S. Jablonski, “Nosql evaluation: A use case oriented survey,” in Cloud and Service Computing (CSC), 2011 International Conference on, pp. 336 –341, dec. 2011.
3. J. M. Pujol, G. Siganos, V. Erramilli, and P. Rodriguez, “Scaling online social networks without pains,” Proc of NETDB, 2009.
4. A. Lakshman and P. Malik, “Cassandra: a decentralized structured storage system,” SIGOPS Oper. Syst. Rev., vol. 44, no. 2, pp. 35–40, Apr. 2010. [Online]. Available: http://doi.acm.org/ 10.1145/1773912.1773922
5. C. Curino, E. Jones, Y. Zhang, and S. Madden, “Schism: a workload-driven approach to database replication and partitioning,” Proc. VLDB Endow., vol. 3, pp. 48–57, Sept. 2010.
6. E. Florenzano, “Twitter example for cassandra.” https://github. com/twissandra/twissandra.
7. O. R. M. Thomae, “Database partitioning strategies for social network data,” Master’s thesis, Massachusetts Inst. of Tech., 2012.
8. U. V. C¸ ataly ¨ urek and C. Aykanat, “PaToH: a multilevel hypergraph partitioning tool, version 3.0,” tech. rep., Bilkent University, Dept. of Comp. Eng., 1999.
9. R. Oguz Selvitopi, A. Turk, and C. Aykanat, “Replicated partitioning for undirected hypergraphs,” J. Parallel Distrib. Comput., vol. 72, pp. 547–563, Apr. 2012.
10. “Amazon ec2 instance types.” http://aws.amazon.com/ec2/ instance-types/.
11. G. Karypis and V. Kumar, “Metis - unstructured graph partitioning and sparse matrix ordering system, version 2.0,” tech. rep., Dept. of Comp. Sci. and Eng. at the University of Minnesota, 1995.
12. A. Tatarowicz, C. Curino, E. Jones, and S. Madden, “Lookup tables: Fine-grained partitioning for distributed databases,” in Data Engineering (ICDE), 2012 IEEE 28th International Conference on, pp. 102 –113, april 2012.
13. J. N. Alertbox, “Participation inequality: Encouraging more users to contribute.” http://www.nngroup.com/articles/ participation-inequality/, 2006.
14. A. Silberstein, J. Terrace, B. F. Cooper, and R. Ramakrishnan, “Feeding frenzy: selectively materializing users’ event feeds,” in Proceedings of the 2010 ACM SIGMOD International Conference on Management of data, pp. 831–842, 2010.
15. S. Yang, X. Yan, B. Zong, and A. Khan, “Towards effective partition management for large graphs,” in Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data, pp. 517–528, 2012.
16. Y. Huang, Q. Deng, and Y. Zhu, “Differentiating your friends for scaling online social networks,” in Cluster Computing (CLUSTER), 2012 IEEE International Conference on, pp. 411–419, sept. 2012.
17. R. Van Driessche and D. Roose. Dynamic load balancing with a spectral bisection algorithm for the constrained graph partitioning problem. In High-Performance Computing and Networking, number 919 in Lecture Notes in Computer Science, pages 392–397. Springer, 1995. Proc. Int’l Conf. and Exhibition, Milan, Italy, May 1995.
18. S. Kadambi, J. Chen, B. F. Cooper, D. Lomax, R. Ramakrishnan, A. Silberstein, E. Tam, and H. Garcia-Molina, “Where in the world is my data?,” PVLDB, vol. 4, no. 11, pp. 1040–1050, 2011.
19. Kallen 2006 N. Kallen, R. Pointer, E. Ceaser, and J. Kalucki. Introducing gizzard, a framework for creating distributed datastores. Twitter Engineering Website, 40, April 2006. URL http://engineering.twitter.com/2010/04/introducing-gizzard-framework-for.html.
20. Influence Maximization on Large-Scale Mobile Social Network: A Divide-and-Conquer Method by Guojie Song, Xiabing Zhou, Yu Wang and Kunqing Xie in IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS , 2014
21. D. Kempe, J. Kleinberg, and E. Tardos, “Maximizing the spread of influence through a social network,” in Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, 2003, pp. 137–146.

AUTHOR(S) PROFILE

Mr. Vishwas Punjaji Badhe, Received his BE degree (2012) from Department of Computer Engineering of Savitribai Phule Pune University (Maharashtra, India), he is now Student of ME (Computer) in G. S. Moze College of Engineering of Savitribai Phule Pune University. He is also working as a lecturer (Since 2013) in Y. B. patil Polytechnic of MSBTE Board, Mumbai, (Maharashtra, India).

Prof. Pradnya Velhal, Received her BE degree (2005) from Govt. college of Engineering, Karad (Maharashtra, India), she has also completed MTech. (Computer) from College of Engineering, Bharati Vidyapith. Now she is working as a Assistant Professor in G. S. Moze College of Engineering of Savitribai Phule Pune University (Maharashtra, India).