International Journal of Research
Available at https://edupediapublications.org/journals
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 03 Issue 18
December 2016

SAE: Towards Productive Details Search Utility for Widely Used Social Networks

BAGATHI.KIRAN KUMAR
PG SCHOLAR,
DEPARTMENT OF CSE,
PYDAH COLLEGE OF ENGINEERING AND TECHNOLOGY,
VISAKHAPATNAM, AP

DR.RAMESH CHALLAGUNDLA
PROFESSOR& PRINCIPAL,
PYDAH COLLEGE OF ENGINEERING AND TECHNOLOGY,
VISAKHAPATNAM, AP

A. SRI SATYA KALYANI
ASSISTANT PROFESSOR
DEPARTMENT OF CSE
PYDAH COLLEGE OF ENGINEERING AND TECHNOLOGY,
VISAKHAPATNAM, AP

ABSTRACT

Social network analysis is used to extract features of human communities and proves to be very instrumental in a variety of scientific domains. The dataset of a social network is often so large that a cloud data analysis service, in which the computation is performed on a parallel platform in the cloud, becomes a good choice for researchers not experienced in parallel programming. In the cloud, a primary challenge to efficient data analysis is the computation and communication skew (i.e., load imbalance) among computers caused by humanity's group behavior (e.g., bandwagon effect). Traditional load balancing techniques either require significant effort to re-balance loads on the nodes, or cannot well cope with stragglers. In this paper, we propose a general straggler-aware execution approach, SAE, to support the analysis service in the cloud. It offers a novel computational decomposition method that factors straggling feature extraction processes into more fine-grained sub-processes, which are then distributed over clusters of computers for parallel execution. Experimental results show that SAE can speed up the analysis by up to 1.77 times compared with state-of-the-art solutions.

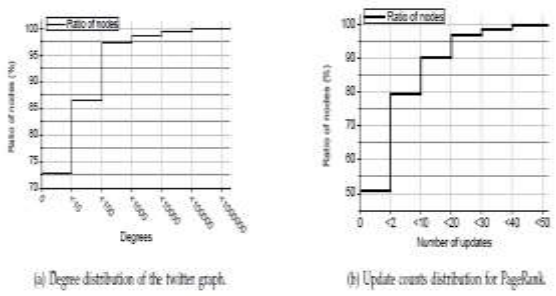
1. INTRODUCTION

SOCIAL network analysis is used to extract features, such as neighbors and ranking scores, from social network datasets, which help understand human societies. With the emergence and rapid development of social applications and models, such as disease modeling, marketing, recommender systems, search engines and propagation of influence in social network, social network analysis is becoming an increasingly important service in the cloud. For example, k-NN is employed in proximity search, statistical classification, recommendation systems, internet marketing and so on. Another example is k-means , which is widely used in market segmentation, decision support and so on. Other algorithms include connected component katz metric adsorption PageRank SSSP and so on. These algorithms often need to repeat the same process round by round until the computing satisfies a convergence or stopping condition. In order to

International Journal of Research
Available at https://edupediapublications.org/journals
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 03 Issue 18
December 2016

accelerate the execution, the data objects are distributed over clusters to achieve parallelism. However, because of the humanity's group behavior the key routine of social net-work analysis, namely feature extraction process (FEP), suffers from serious computational and communication skew in the cloud. Specifically, some FEPs need much more computation and communication in each iteration than others.

Power Graph can only statically partition computation for graphs with fixed dependencies and therefore cannot adaptively redistribute sub-processes over nodes to maximize the utilization of computation resources.



(a) Degree distribution of the twitter graph. (b) Update counts distribution for PageRank.

Fig. 1. Skewed computation load distribution for a twitter follower graph.

mize the utilization of computation resources. At the worker level, the state-of-the-art solutions, namely persistence-based load balancers (PLB) and retentive work stealing (RWS) , can dynamically balance load via tasks redistribution/stealing according to the profiled load from the previous iterations. However, they cannot support the computation decomposition of straggling FEPs. The task partitioning for them mainly considers evenness of data size, and so the corresponding tasks may not be balanced in load. This may cause serious computational and communication skew during the execution of program

2. SYSTEM ANALYSIS

Existing System

Task-level load balancing. Skew Reduce is a state-of-the-art solution for reducing load imbalance among tasks, in view that in some scientific applications, different partitions of the data object set take vastly different amounts of time to run even if they have an equal size. It proposes to employ user defined cost function to guide the division of the data object set into equally-loaded, rather than equally-sized, data partitions. However, in order to ensure low load imbalance for social network analysis, it has to pay significant overhead to periodically profile load cost for each data object and to divide the whole data set in iterations.

Worker-level load balancing. Persistence-based load balancers and retentive work stealing represent the approaches to balance loads among workers for iterative applications. Persistence based load balancers redistributes the work to be
performed in a given iteration based on measured performance profiled from previous iterations. Retentive work stealing is used for applications with significant load imbalance within individual phases, or applications with workloads that cannot be easily profiled.

Disadvantages:

- It cannot support the computation decomposition of straggling FEPs.

- High load imbalance will be generated for the analysis program.
- Twitter follower graph containing 41.7 million vertices and 1.47 billion edges. Shows the degree distribution of the graph. It can be seen that less than one percent of the vertices in the graph are adjacent to nearly half of the edges

Proposed system

- SAE addresses the problem of computational and communication skews at both task and worker levels for social network analysis in a different way based on the fact that the computation of FEP is largely decomposable.
- it proposes an efficient approach for social network analysis to factor straggling FEPs into several sub-processes.
- adaptively distribute these sub-processes over computers, aiming to parallelize the decomposable part of straggling FEP and to accelerate its convergence.

Advantages

- The decomposable part, of the computation task in an FEP is decomposable, because each feature can be seen as a total contribution from several data objects.
- Each FEP can be factored into several sub-processes which calculate the value of each data object separately.
- This allows us to design a general approach to avoid the impact of load skew via a straggler-aware execution method.

3. ALGORITHM

Redistribution algorithm

1: =_ Executed on the master. _=
2: **procedure** REDISTRIBUTION(W_{Set}, F_{idle})
3: W_d.Load 0
4: =_ Calculate all workers' mean load. W_{Set} contains
the profiled remaining load of all workers. _=
5: $L_{Average}$ GetMeanLoad(W_{Set})
6: **while** (F_{idle}^ W_d.Load _ $L_{Average}$) _L_c _ " **do**
7: **if** F_{idle} **then** ==Triggered by an idle worker.
8: =_ Get the idle worker. _=
9: W_d GetIdleWorker()
10: **else**
11: =_ Get the worker with the least load. _=
12: W_d GetFastestWorker(W_{Set})
13: **end if**
14: =_ Get the slowest worker. _=
15: W_s GetSlowestWorker(W_{Set})
16: L_{0c} $L_{Average}$ W_d.Load
17: $L_{00\ c}$ W_s.Load $L_{Average}$
18: =_ Get the minimum value. _=
19: L_c Min(L_{0c};$L_{00\ c}$)
20: Migration(W_s, W_d, L_c)
21: W_s.Load W_s.Load L_c
22: W_d.Load W_d.Load + L_c
23: **end while**
24: **end procedure**

Migration algorithm

1: =_ Executed on the worker. It is employed to migrate L_c of load from worker W_s to W_d. _=
2: **procedure** MIGRATION(W_s, W_d, L_c)
3: NeedDecomposition False
4: L_t 0
5: **while** L_t < L_c **do**
6: =_ Select the unprocessed feature in this
7: iteration and with load less than L_c L_t. _=
8: F_t W_s.SelectFeature(L_c L_t)
9: **if** F_t = ; **then** ==No more suitable features
10: NeedDecomposition Ture
11: **end if**
12: **if** NeedDecomposition = False **then**
13: =_ Insert the values needed to be

International Journal of Research
Available at https://edupediapublications.org/journals
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 03 Issue 18
December 2016

14: processed for it into a suitable block. _=
15: B_{set}.InsertNonStraggling(F_t)
16: L_t L_t+ F_t.Load
17: **else**
18: =_ Randomly get a straggling feature. _=
19: F_t SelectFeature(Max)
20: $B_{max}(F_t)$ ($T_{origin}(F_t)$
_)1=2
21: L_m Min(L_c L_t, F_t:$Load_{Remaining}$)
22: $N(F_t)$ b L_m
$T_{origin}(F_t)$ c _ $B_{max}(F_t)$
23: =_ Insert $N(F_t)$ numbers of F_t' s
24: unprocessed blocks into B_{set}. _=
25: B_{set}.InsertStraggling(F_t, $N(F_t)$)
26: L_t L_t+ L_m
27: **end if**
28: **end while**
29: =_ Transform all blocks contained in B_{set}. _=
30: Transform(W_d, B_{set})
31: **end procedure**

PageRank algorithm with SAE model

1: **procedure** EXTRA(FeatureSet O_{set})
2: **for** each R(k, j) 2 O_{set} **do**
3: V (j).sum V (j).sum+ R(k, j)
4: **end for**
5: **end procedure**
6: **procedure** ACC(R(j))
7: ValueSet Get(R(j))
8: R(j).sum 0
9: **for** each V (j) 2 ValueSet **do**
10: R(j).sum R(j).sum+ V (j).sum
11: **end for**
12: R(j) (1- d)+ d _ R(j).sum
13: hlinksi look up outlinks of j
14: **for** each link < j, i > 2 hlinksi **do**
15: R(j, i) R(j)/deg(j)
16: Diffuse(i, R(j, i))
17: **end for**
18: **end procedure**

4 I

MPLEMENTATION

- **Data Owner**

In this module, the data owner Registers, Login's the owner by registered details, requesting the resources in cloud like Virtual Machine (VM), Memory, Threshold, and uploads their data in the selective cloud server. For the security purpose the Trust manager encrypts the data file and then store in the Cloud Server.

- **Cloud Server**

The Data Owner sends a request to Virtual Master to provide services by assigning the task for any one cloud like Cs1, CS2, and CS3. The cloud service provider manages multiple clouds to provide data storage service via Virtual Master. To access the shared data files, data consumers download encrypted data files of their interest from the specified cloud and then decrypt them and The Cloud server can attack the files in the cloud Server

- **Data Integrity**

Data Integrity is very important in database operations in particular and Data warehousing and Business intelligence in general. Because Data Integrity ensured that data is of high quality, correct, consistent and accessible.

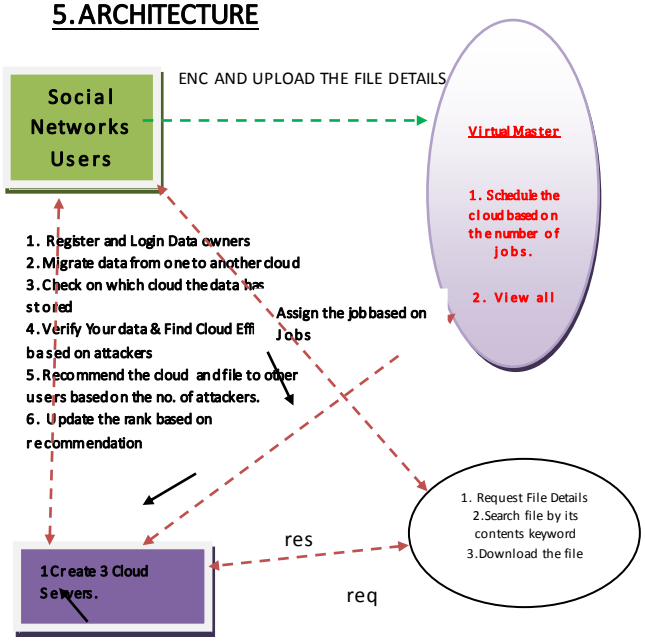
- **Virtual Master** :The Virtual Master Will Schedule the cloud based on the number of jobs and View all transactions (Upload and downloads), View all Cloud Schedules
- **Data Consumer(End User)**

In this module, the user has to get Registered to Trust Manager to access the Cloud services and need to

International Journal of Research
Available at https://edupediapublications.org/journals
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 03 Issue 18
December 2016

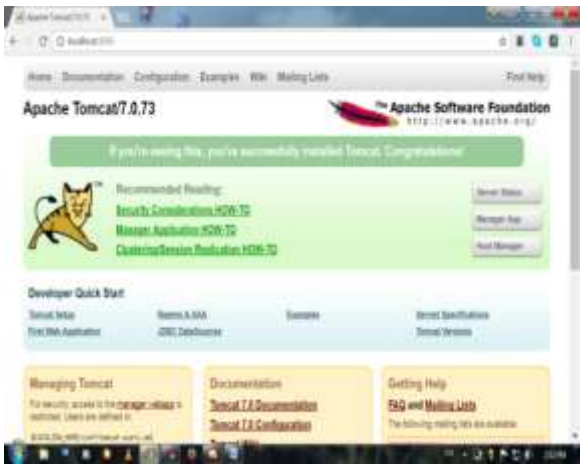
Authenticate the user by Logging in by providing the User Name and auto Generated Password by Virtual Master then the Data Consumer can access the data file with the encrypted key, so if User access the file by wrong Key then the user will consider as malicious users and blocked the User

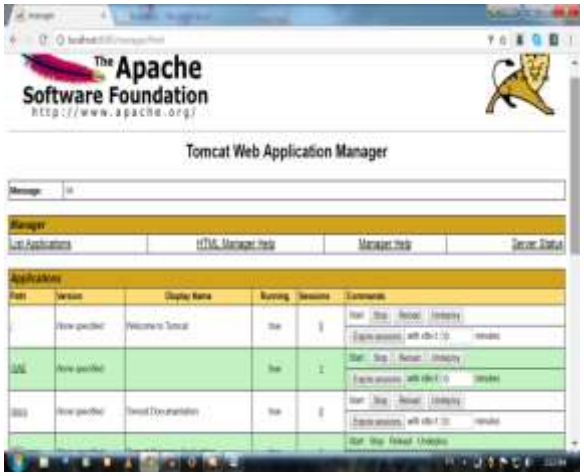
5. ARCHITECTURE



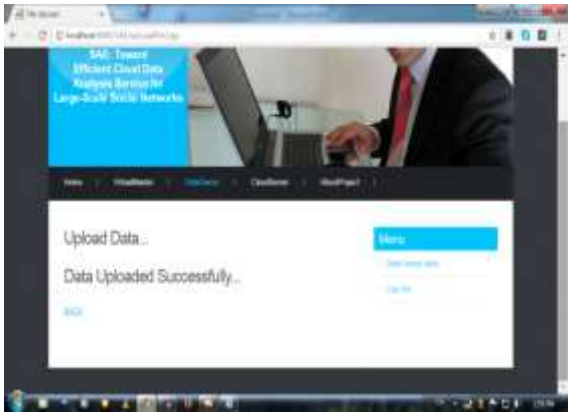
1. Receive file information.
2. View all File Information
3. View all data owners and End Users.
4. View all attackers
5. View all transactions(Upload ,Download)
6 View all Recommendation

6. Results





International Journal of Research
Available at https://edupediapublications.org/journals
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 03 Issue 18
December 2016



CONCLUSION

For social network analysis, the convergence of straggling FEP may need to experience significant numbers of iterations and also needs very large amounts of computation and communication in each iteration, inducing serious load imbalance. However, for this problem, current solutions either require significant overhead, or can not exploit underutilized computers when some features converged in early iterations, or perform poorly because of the high load imbalance among initial tasks. This paper identifies that the most computational part of straggling FEP is decomposable. Based on this observation, it proposes a general approach to factor straggling FEP into several sub-processes along with a method to adaptively distribute these sub-processes over workers in order to accelerate its convergence. Later, this paper also provides a programming model along with an efficient runtime to support this approach. Experimental results show that it can greatly improve the performance of social network analysis against state-of-the-art approaches. As shown in Section 5, the master in our approach may become a bottleneck. In future work, we will study how to employ our approach in a hierarchical way to reduce the memory overhead and evaluate its performance gain.

REFERENCES

[1] Z. Song and N. Roussopoulos, "K-nearest neighbor search for moving query point," Lecture

Notes in Computer Science, vol. 2121, pp. 79–96, July 2001.

[2] X. Yu, K. Q. Pu, and N. Koudas, "Monitoring k-nearest neighbor queries over moving objects," in Proceedings of the 21st International Conference on Data Engineering. IEEE, 2005, pp. 631–642.

[3] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu, "An efficient k-means clustering algorithm: Analysis and implementation," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 7, pp. 881–892, July 2002.

[4] L. Di Stefano and A. Bulgarelli, "A simple and efficient connected components labeling algorithm," in Proceedings of the International Conference on Image Analysis and Processing. IEEE, 1999, pp. 322–327.

[5] E. Deelman, G. Singh, M.-H. Su, J. Blythe, Y. Gil, C. Kesselman, G. Mehta, K. Vahi, G. B. Berriman, J. Good et al., "Pegasus: A framework for mapping complex scientific workflows onto distributed systems," Scientific Programming, vol. 13, no. 3, pp. 219–237, January 2006.

[6] L. Katz, "A new status index derived from sociometric analysis," Psychometrika, vol. 18, no. 1, pp. 39–43, March 1953.

[7] D. Liben-Nowell and J. Kleinberg, "The link prediction problem for social networks," in Proceedings of the 12th international conference on Information and knowledge management. ACM, 2003, pp. 556–559.

[8] S. Baluja, R. Seth, D. Sivakumar, Y. Jing, J. Yagnik, S. Kumar, D. Ravichandran, and M. Aly, "Video suggestion and discovery for youtube: taking random walks through the view graph," in Proceedings of the 17th international conference on World Wide Web. ACM, 2008, pp. 895–904.

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

[10] S. Baluja, R. Seth, D. Sivakumar, Y. Jing, J. Yagnik, S. Kumar, D. Ravichandran, and M. Aly, "Video suggestion and discovery for youtube: taking random walks through the view graph," in Proceedings of the 17th international conference on World Wide Web. ACM, 2008, pp. 895–904.

[11] H. H. Song, T. W. Cho, V. Dave, Y. Zhang, and L. Qiu, "Scalable proximity estimation and link prediction in online social networks," in Proceedings of the 9th ACM SIGCOMM conference on Internet Measurement Conference. ACM, 2009, pp. 322–335.

[12] Y. Zhang, X. Liao, H. Jin, and G. Min, "Resisting Skewaccumulation for Time-stepped Applications in the Cloud via Exploiting Parallelism," IEEE Transactions on Cloud Computing,

doi: 10.1109/TCC.2014.2328594, May 2014

AUTHOR'S PROFILE

BAGATHI.KIRAN KUMAR
PG SCHOLAR
DEPARTMENT OF CSE
PYDAH COLLEGE OF ENGINEERING AND TECHNOLOGY, VISAKHAPATNAM

A.SRI SATYA KALYANI
ASSISTENT PROFESSOR
DEPARTMENT OF CSE
PYDAH COLLEGE OF ENGINEERING AND TECHNOLOGY, VISAKHAPA

DR.RAMESH CHALLAGUDLA

M.E,PH.D,MIEEE(USA),FIETE(IND),MIE(IND),MIS TE is a former faculty of Birla Institute of Technology, Gitam University, ANITS and currently working as Professor and Principal of Pydah College of Engineering and Technology, Visakhapatnam