Cloud Based Gateway Clustering of Cloud Data Retrieval with GCD Recovery Scheme

Padmakumari.P [#1], Umamakeswari.A [*2], Shanthi.P [#3]

[#] CSE Department, SASTRA University
Thanjavur, Tamilnadu, India
[1] padmalec.sastra@cse.sastra.edu
[3] shanthip@cse.sastra.edu
[*] CSE Department, SASTRA University
Thanjavur, Tamilnadu, India
[2] aum@cse.sastra.edu

***Abstract*—Different parallelization approaches are available to cluster the computers and to scale the data and resources. With them, parallel clustering mechanism is used to faster the computation time of data access with minimal cost. But, this mechanism is not feasible for efficient data retrieve over cloud environment and is not provided with fault tolerant design. The proposed paper attempts to develop data scalability in a cloud environment by implementing Cloud based Gateway Clustering (CBGC) mechanism with fault tolerant design. The Client accesses the data from the cloud using Cloud core storage gateway, which is an API. This paper proposes a core gateway which is divided into multiple sub virtual gateways to improve the data access speed. Also an enhanced Gateway Cut Detection (GCD) Algorithm is proposed to detect and recover sub-gateway failures and make sub- gateways to be available throughout the communication. The proposed CBGC mechanism with GCD is highly scalable and reliable to cloud resources.**

Keyword: Cloud storage, Datacenters, Gateway failure, Clustering, Scalablity, Throughput

I. INTRODUCTION

The process of systematic collection of data is known as clustering. Many clustering mechanisms are suggested in literature [5, 6]. Among these the hierarchical and partition method play a vital role. These customary mechanisms are not robust. Some others are extremely expensive in computation when dealing with large datasets. To tackle these problems, many new clustering mechanisms have been initiated, such as the Grid-based mechanism [7] and neural network based mechanism [8]. Major setbacks iin these mechanisms are concerned with time and space complexity in large Datasets. To prevail over computational complexity, Power Iteration Clustering (PIC) [4] mechanism is introduced. PIC has better ability in handling huge datasets, but computer memory is not enough to store data and its relevant similarity matrix when large dataset are used. Inspired the burly need for clustering of massive datasets, PIC using parallelism was implemented.

Parallel clustering mechanism [2] focuses on memory problem for storing similarity matrix in PIC mechanism. It can be overcome by dividing data into small sub-data and dispensing to multiple processors. The Message passing interface framework is used for implementation. Starting and ending indices of the sub-data has determined by the master processor and broadcast those indices to slave processors. Each slave processor retrieves the sub-data and performs similarity calculation to find the sub-matrixes and order the resultant row sum. The resultant row sum from all the processors is sent to master processor and combined to form global row sum of requests dataset. This mechanism does not explore with node failure. Motivation of parallel clustering expands awareness to a cloud environment.

In cloud storage, data are stored in a virtualized pool of third party dataset. Data can be accessed by clients using Cloud storage gateway (CSG) API. To improve the reliability of the data, it can be stored in multiple datacenters in the cloud. Cloud Gateway clustering (CGC) mechanism is implemented in this paper for fast access of data from cloud environment. Gateway failure can be overcome by using GCD Algorithm.

II. PARALLEL CLUSTERING

Among the many parallel programming frameworks available, MPI is used to implement parallel clustering mechanism. This mechanism focuses on matrix calculation and normalization. Let be n data points {d1, d2…dn} and p Processors. Master processor finds out the starting and ending indices of data, divides and broadcasts to slave processors. Slave processors read the given data from a stored file (d/p), perform similarity calculation to find sub-matrix and order the resultant row sum. Let s_j and e_j be starting and ending indices of processor ***j***. The procedure to find similarity matrix and normalization are given below:

For R=s_j to e_j

Read Rth case, d_R , from the stored file

For c = 1 to n

Read cth case, d_c, from the stored file

$M_j(R, c) = \frac{d_R . d_c}{\|d_R\|2.\|d_c\|2}$ where $R \neq c$

Endfor

$M_j(R, :) = M_j(R, :) / \sum_c M_j(R, c)$//normalizes by row sum

Endfor

In this mechanism, node or gateway failures are not addressed. Until the gateways are available, communication among the gateways is efficient. If any gateway fails, there is lack of communication which leads to data loss and improper Indexing. The CbGC mechanism enhances the fault tolerant design and also this mechanism improvises cloud environment. Data access in the cloud storage becomes more efficient with gateway failure recovery method called Gateway Cut Detection (GCD).

III. CLOUD BASED GATEWAY CLUSTERING (CBGC)

Cloud storage is form of storing enterprise data of an organization in virtualized pool of storage. Those data can be accessed by the client from cloud storage using web service application programming interface (Web-API). Cloud Storage gateway is a network application reside in the client and act as bridge among client and cloud data storage. To make fast access of data from cloud storage, Cloud Based Gateway Clustering (CBGC) mechanism is proposed. Usually, data in the cloud storage is available in multiple data centers for reliable access.

In proposed mechanism, Cloud Core Gateway is split into multiple virtual sub-storage gateways. Cloud Core Storage gateway receives the starting and ending indices of dataset to be retrieving from cloud storage. The core-gateway split the index into sub indices and broadcast the starting and ending indices to each virtual sub-gateway. Virtual sub-gateways choose different datacenters and retrieve the data according to their starting and ending indices. The sub –gateways broadcast the dataset to the Core. The core gateway links them and gives it to the client.Fig.1 explains about the overall architecture of CBGC mechanism.

Let S and E be the starting and ending indices of the datasets given by the client to cloud core storage gateway. Core gateway splits the indices into $S_1, S_2 \ldots S_n$ and $E_1, E_2, \ldots E_n$ and broadcast to multiple virtual gateways. Virtual gateways search the data by using starting and ending indices given by the core gateway in different datacenters. After receiving the data, sub gateway broad cast to core and core gateway concatenate the data and give it to the client. Following steps explain clearly about how cloud core gateway and virtual sub-gateways retrieve the data from cloud storage.

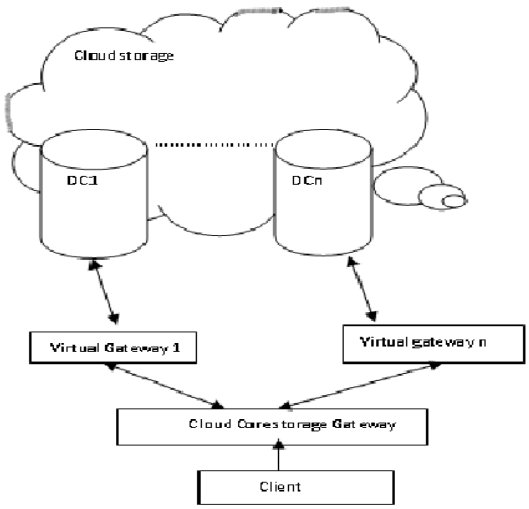

Fig.1. Overall Infrastructure of CBGC

Core gateway receives the starting and ending indices from the client, it starts with splitting the indices and broad- cast to multiple virtual gateways. Following steps make clear about the processing mechanism of cloud core gateway

Step-wise method for Cloud Core Storage Gateway

1. Cloud core storage gateway gets starting and ending indices from the client.
2. Partition the indices and transmit to different virtual sub- storage gateways to access the data from cloud storage.
3. Collect $R_{sei,}$ sei=1,2,3,……n from all virtual sub-storage gateways and link them to obtain the overall Row sum ,R
4. Obtain the initial vector,$iv^0=R/\|R\|_1$
5. Transmit the vector ,iv^0 to all virtual sub gateways
6. Collect sub vectors from sub gateways iv^t_i =1,2,3…..n and link them to obtain a new vector ,iv^t ,and normalize it by its element sum
7. Check the stop criteria ,$| \delta^t - \delta^{t-1} | \approx 0$
8. If no ,move to step 5
9. If OK, End

Matrix–vector multiplication is carrying out to revise the Virtual sub gateway. The Cloud core gateway gather all revised vectors from sub-gateways and stop the process once all data retrieve. Steps given below make clear about the working mechanism of sub-gateways to collect data from different data centers.

1. Receive starting and end indices from core storage gateway
2. Read the chunk of data
3. Calculate the row sum, $_{Rsei,}$ of the sub- matrix and send it to cloud core storage gateway
4. Normalize sub-matrix by using the row sum,$N_t=DM_i^{-1}$ A
5. Receive the initial vector from cloud core storage gateway,iv^{t-1}
6. Obtain sub-vector by performing matrix-vector multiplication,$iv^t_i=$?$W_i iv^{t-1}$,
7. Send the sub-vector $^t_{i,}$to cloud core storage gateway

IV. GATEWAY CUT DETECTION

The gateway failures in the network will decrease the multi-hop paths. Gateways which fail can be cut from network and allow other to carry on. This paper has proposed with an algorithm to identify the sub- gateways failure and recover it, permit uninterrupted flow of data among other sub- gateways. Steps follow in GCD algorithm. In Step 1, identify the Sub-gateways which are all cut off from the Core Gateway in the client by making DOS event to be takes place. Step 2, find out the gateways which lie close to cuts but are still linked to the core-gateway to detect CCOS events and alert the core gateway. In GCD Algorithm, specially designed Gateway List is active always. Let GL (IP, SF) denotes Gateway list that consists of all Gateways ip addresses that are active at time K, where k=0, 1, 2…. is a repetitive counter. If Gateway list perform the GCD algorithm with initial condition as GL (0) =0, if failure not occur or any gateway is not cut, then the state flag SF is set to positive interger.if any failure occurs at time T>0 disconnects the gateway failed and set flag SF to negative integer. Implementation of an algorithm is given below. The advantage of GCD Algorithm is easily implement in cloud environment. It is robust when failure occurs in sub-gateways. Core-gateway has the ability to find out the failed sub-gateway.so that communication link to access the data will increase.

Let G be Core Gateway; VG1, VG2… are virtual gateways; Set Status flag SF =positive integer

Let ACK=1, DACK=0

1. If the sub-gateway VG1 is active, Send Keep-Alive message is sent by core-gateway.
2. Remain for Keep- Alive message RESPONSE
3. Receives reply, set SF=Positive integer in GL (Gateway List)
4. Else if the sub-Gateway VG1 is inactive (gateway failed)
5. Set SF=negative integer in GL, DACK=0
6 .Select the nearest gateway.
7. Repeat the steps

Fig. 2 has shown the gateway recovery mechanism using GCD Algorithm

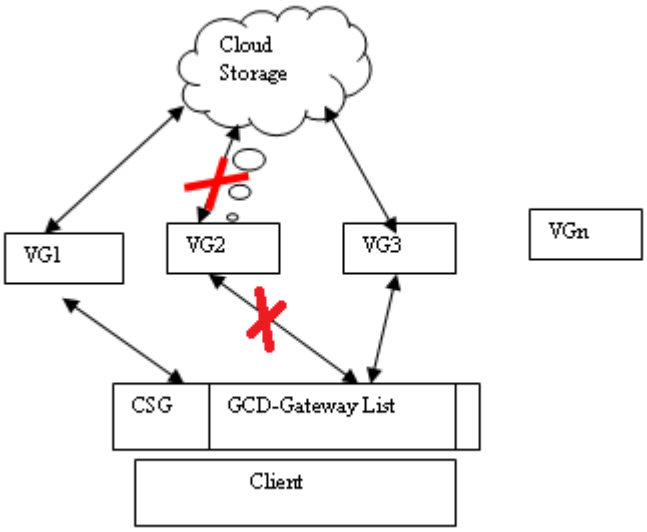

Fig 2. GCD Gateway Failure Recovery Scheme

V. RESULT AND ANALYSIS

The efficiency of the Parallel clustering mechanism has been approved [2] .This paper will spot light on representing the efficiency of C_bGC mechanism in cloud environments.

A. Datasets:

To exhibit the data efficiency of the CBGC mechanism, it ran on different datasets varies from 10000 to 500000 rows of data from cloud storage. Consider each row of dataset is 5 dimensional vectors ([D1, D2, D3, D4, D5] and first couple of dimension sampled by Gaussian mixture G and the other eight dimensions drawn separately by using Gaussian distribution. First couple of data that reveal the clustering structure, while others are clatter falsely added to check the mechanism's robustness. Fig. 3 has shown the clear idea about the efficiency of dataset.

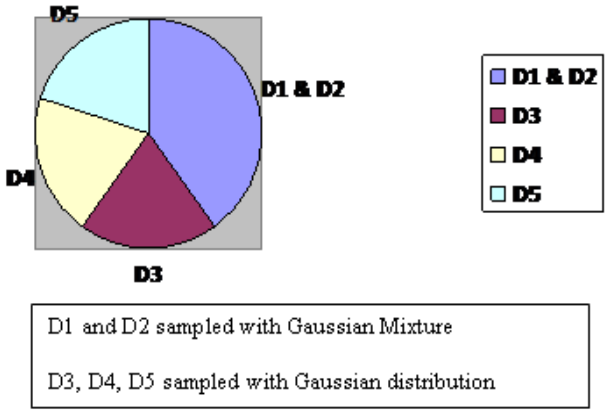

Fig. 3. Robustness and Efficiency of Dataset

B.Speed Up:

Speedup of proposed mechanism can be implemented by performing in Amazon S3 storage.Fig.4 shows that speedup increases when data access takes place in multiple virtual Gateways. This indicates that number of virtual gateways increase the speedup of data access. Amazon s3 storage machines are auto setup and configuration settings. Large set data can be process in Amazon S3.the proposed mechanism supports to process large datasets.

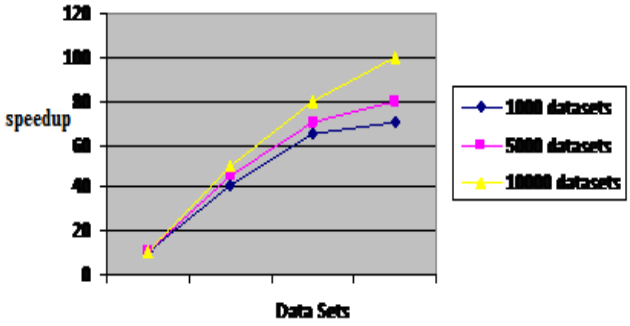

Fig.4. Speedup result obtain for C_bGC mechanism on Amazon S3 storage

C.Failover:

Gateway failover time consists of (a) finding the gateway failure (b) Change to nearest gateway (c) Start again access to data. To measure the failover time, intentionally caused gateways to failed. Measure the throughput between the client and cloud server. Fig 5 shows the throughput comparison between parallel power iteration clustering mechanism and cloud based gateway clustering mechanism and GCD Algorithm. It make clear that number of failure gateway recovery using CbGC mechanism with GCD Failure recovery method is high when compare to pPic

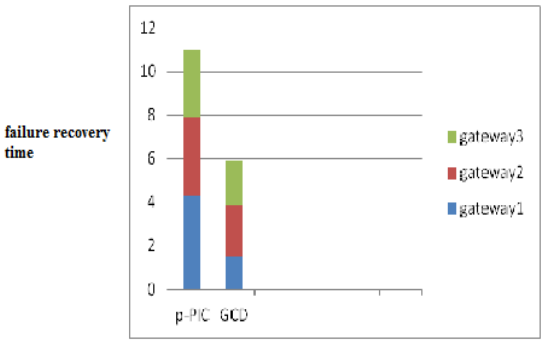

Fig.5.Comparsion of Failure Recovery Time between p-PIC and GCD

VI. CONCLUSION

Accessing large set of data from the cloud storage is a not easy task. This paper illustrates Cloud based gateway clustering mechanism to access data from the cloud storage in effective and efficient way. Dividing the indices and broadcast to sub virtual gateways, those gateways retrieve data from different datacenters in cloud storage. This Mechanism reduces the pardon of core gateway from where data can be retrieve from cloud storage. Core-gateway takes care of gateway failure detection and recovery by using GCD algorithm. Results and analysis show that CbGC mechanism using GCD failure recovery is very scalable and have high throughput. The proposed paper is not deal with security issues and efficient method to overcome that during retrieval of data from cloud storage. It also not addresses data loss during access. These issues can be heighted in future work.

REFERENCES

[1] Navneet N Tewani, Neeharika Ithapu, K Raghava Rao, Sheik Nissar Sami, B. Sai Pradeep,V.Krishna Deepak, *distributed Fault Tolerant Algorithm for Identifying Node Failures in Wireless Sensor Networks*, International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278-3075, Volume-2, Issue-5, April 2013

[2] Weizhong Yana, Umang Brahmakshatriya a, Ya Xuea, Mark Gilder b, Bowden Wisec, *p-PIC: Parallel power iteration clustering for big data*, journal of parallel and distributed computing, 2013

[3] Yohei Matsuhashi,Takahiro Shinagawa, Yoshiaki Ishii, Nobuyuki Hirook, Kazuhiko Kato , *Transparent VPN failure recovery with virtualization*, Future Generation Computer Systems 28 (2012) 78–84
[4] F. Lin, W.W. Cohen, *Power iteration clustering*, in: Proceedings of the 27th International Conference on Machine Learning, Haifa, Israel, 2010
[5] R. Xu, D. Wunsch, *Survey of clustering algorithms*, IEEE Transactions on Neural Networks 16 (3) (2009).
[6] A. Jain, M. Murty, P. Flynn, *Data clustering: a review, ACM Computing Surveys* 31 (3) (2006) 264–32
[7] A. Cuzzocrea, Il-Y. Song, K.C. Davis, Analytics *over large-scale multidimensional data: the big data revolution!* in: Proceedings of DOLAP 2011,pp. 101–104.
[8] D. Yan, L. Huang, M.I. Jordan, *Fast approximate spectral clustering*, in: KDD 2009, Paris, France, June 28–July 1, 2009
[9] W. Zhao, H. Ma, Q. He, *Parallel K-means clustering based on Map Reduce*, Cloud Computing 5931 (2009) 674–679
[10] C.-H. Hsu, A. Cuzzocrea, S.-C. Chen, *CAD: an efficient data management and migration scheme across clouds for data-intensive scientific applications*,in: Lecture Notes in Computer Science, vol. 6864, 2011, pp. 120–134
[11] V. Olman, F. Mao, H. Wu, Y. Xu, *Parallel clustering algorithm for large data sets with applications in bioinformatics*, IEEE Transactions on Computational Biology and Bioinformatics 6 (2) (2009) 344–352
[12] H. Wang, J. Zhao, H. Li, J. Wang, *Parallel clustering algorithms for image processing on multi-core CPUs*, in: 2008 International Conference on Computer Science and Software Engineering, 2011
[13] Jagdish Pimple, Prof.Yogadhar Pandey ,*Cut Detection in Wireless Sensor Network using Distributed Source Separation Detection (DSSD) Approach.*, International Journal of Scientific and Research Publications, Volume 2, Issue 12, December 2012 1 ISSN 2250-3153
[14] Jon Kleinberg, *Detecting a Network Failure*, Internet Mathematics Vol. 1, No. 1: 37-56
[15] G. Dini, M. Pelagatti, and I.M. Savino, *An Algorithm for Reconnecting Wireless Sensor Network Partitions*,Proc. European Conf. Wireless Sensor Networks, pp. 253-267, 2008