INTERNATIONAL JOURNAL FOR TRENDS IN ENGINEERING & TECHNOLOGY
VOLUME 3 ISSUE 1 -JANUARY 2015 - ISSN: 2349 - 9303

Parallel Particle Swarm Optimization for Reducing Data Redundancy in Heterogeneous Cloud Storage

M.Vidhya[1]
[1]Department of Computer Science and Engineering,
Bannari Amman Institute of Technology,
Sathyamangalam .
vidhyam284@gmail.com

Mr.N.Sadhasivam[2],
[2]Department of Computer Science and Engineering,
Bannari Amman Institute of Technology,
Sathyamangalam.
sadhasivamn82@gmail.com

Abstract— Cloud storage is usually distributed infrastructure, where data is not stored in a single device but is spread to several storage nodes which are located in different areas. To ensure data availability some amount of redundancy has to be maintained. But introduction of data redundancy leads to additional costs such as extra storage space and communication bandwidth which required for restoring data blocks. In the existing system, the storage infrastructure is considered as homogeneous where all nodes in the system have same online availability which leads to efficiency losses. The proposed system considers that distributed storage system is heterogeneous where each node exhibit different online availability. Monte Carlo Sampling is used to measure the online availability of storage nodes. The parallel version of Particle Swarm Optimization is used to assign redundant data blocks according to their online availability. The optimal data assignment policy reduces the redundancy and their associated cost.

Index Terms— cloud storage, data redundancy, online availability, optimal data assignment, particle swarm optimization

1 INTRODUCTION

CLOUD storage systems are built upon storage resources from different computers or different dedicated storage devices to build a large storage service which provides more reliable, scalable and efficient storage service. The cloud storage services such as Amsazon, Facebook, Gmail use distributed storage systems. The cloud service providers render small capacity of storage for free and additional storage capacity for pay on monthly or annual basis. The cloud service providers and the users agreed upon the service level agreement which provide the quality of service measures such as response time, data availability, data reliability and reasonable costs. The amount of data that is replicated depends on the service level a customer chooses.

In the storage system the redundancy is maintained to ensure data availability. The two mechanisms to maintain data redundancy are replication and erasure coding. In replication mechanism, the entire file is stored multiple times in different storage nodes which increases the storage space for which the user have to pay additional costs. When the storage node fails, the entire file has to be transmitted to restore the lost data over the communication links which cause additional communication costs. Another mechanism is erasure coding in which the data file is divided into fixed number of data blocks which are further encoded into number of redundant blocks. The cloud storage system uses the (k, n) erasure code where the file is divided into k data blocks which are encoded into n redundant blocks. At any time k redundant blocks are enough to construct original file.

The storage system consists of set of storage nodes N. At any time the set of nodes M is chosen from N which is a random sample of N to store data blocks. In the set of nodes M, there may be some nodes available and some may not be available. By monitoring every node in the system for longer period of time, the mean node availability is calculated which is based on how much time the node active on the network.

The data assignment function is used to assign redundant data blocks to set of storage nodes based on their online availability. the number of possible combinations to assign redundant data blocks is computationally hard for large scale storage systems. to solve such large scale combinatorial problems, the heuristic algorithms are used. The optimization algorithm works by defining the search space and to maximize the specified function. There are numerous algorithms available but particle swarm optimization is proved to have better performance for distributed environment. The computational speed of particle swarm optimization algorithm is faster compared to other evolutionary techniques such as genetic algorithm. The PSO algorithm is easy to implement because it has few parameters and convergence speed is faster.

The particle swarm optimization algorithm may suffer from premature convergence that is the chance of falling into local optimum is higher. The algorithm falls into local optimum due to loss of diversity in population because each particle updates and follows the single global best particle. If the particle falls into local optimum, then the algorithm can't overcome from the local best

solution. So to overcome such problems the parallel version of particle swarm optimization algorithm is used. In this the entire population is divided into subgroups and each subgroup performs independently for some time. After some time interval, the best of each swarm is chosen to find the best particle which has maximum value. Based on that value, all particles update their position. Here the convergence rate of the algorithm is improved because the particles moved towards the global best particle which is the best of all the sub swarms. But when the convergence rate is reduced, the algorithm's computational speed also gets reduced. Here all the sub swarms are independent. So they are executed in parallel on different machines which improve the computational time.

The optimal data redundancy is the minimum amount of redundancy required to the data availability for accessing data at any time. The redundancy ratio decides how many times the data blocks are replicated. The value of redundancy ratio should be low to keep minimum redundancy.

2 RELATED WORK

Cloud storage services have obtained importance in the recent years [1, 6, 7, 23]. These services allow users to store data outside of their storage infrastructure. To store large amounts of data from millions of users, cloud storage systems build their services over distributed storage infrastructures that are more scalable and reliable than centralized storage systems. Different works have proposed distributed data storage infrastructures, some of which built on grid infrastructures [1, 7], and some others built on P2P networks [20]. To achieve the required data availability, all the storage systems need to store data with redundancy. But the use of redundancy also increases storage and communication costs. To maintain the storage system accessible, the redundancy has to be reduced.

The peer-to-peer (P2P) storage system such as Wuala [21] allows users to share part of their hard drives to obtain a reliable and online storage capacity. Chandra and Weissman [5] proposed the idea of letting users to contribute their unused storage resources to satisfy the storage requirements of these research projects. A similar approach was advocated by the developers of Open Cirrus [4] to aggregate unused resources from different small and heterogeneous datacenters.

The existing storage solutions treat all nodes in the system are equal and they have same online availability. But the nodes in distributed storage infrastructure are heterogeneous. They exhibit different online availabilities. So the proposed system uses heterogeneity to calculate data availability, the Particle Swarm Optimization algorithm is used to place redundant data blocks according to node's availability and minimum data redundancy is obtained that satisfies the targeted availability.

By reducing redundancy, distributed storage systems can reduce storage and communication costs. In [2], Blakes and Fodrigues described a limit beyond which communication costs cannot be reduced without increasing storage costs and the reverse is also true. The data redundancy schemes [17, 22, 23] provide better communication-storage trade-offs. Unfortunately, all these works have assumed homogeneous properties for all the storage nodes in the system.

The main objective of this paper is to optimize distributed storage systems by considering the real heterogeneities present in cloud [4] systems, or any other existing distributed systems. In [7], Deng and Wang considered node heterogeneities in grid storage systems. However, they did not use such heterogeneities to reduce the amount of data redundancy required. In [15], Pamies-Juarez et al. describe heterogeneities using simple heuristic approaches to reduce data redundancy. In [16], Pamies-Juarez et al. provided the method to calculate data availability for any scale of storage system and use optimization method to assign data blocks to set of nodes. They provide the simple measure to find optimal data redundancy. In [11], Gonsalves & Egashira describes the parallel version of PSO algorithm without introducing any complexity. In this paper, the parallel version of algorithm is used to improve the data availability and to reduce the data redundancy of nodes compared to standard Particle Swarm Optimization algorithm.

3 PROBLEM DESCRIPTION

In distributed storage system in order to maintain data availability some amount of redundancy has to be maintained .But introducing redundancy leads to additional storage and communication costs. So the need for effective redundancy scheme is required which balances the both data availability and its associated costs. In literature [9, 14, 17, 20], the erasure codes are proved to provide better redundancy for distributed storage systems. To improve data availability the data objects should be stored in high reliable nodes. To calculate node's availability for any scale Monte Carlo Sampling method is used.

The existing system use Particle Swarm Optimization algorithm to assign data blocks to different nodes based on their availability. In original PSO, each particle updates its position based on its personal best and the global best. Here the global best is the best of entire swarm. All the particles in the swarm are moved towards on single position called global best. So the algorithm converges faster which is called premature convergence leads to poor performance. That is the algorithm fails to find a global optimal solution or even sometimes best local optimal solution. This has occurred due to decline of population diversity. When the algorithm parameters are not tuned properly there is a chance of falling in the local optima. In general, maintaining diversity will reduce convergence rate which is used to remove swarm stagnation in a poor local optimal solutions. To overcome such problems a parallel approach called Parallel Particle Swarm Optimization is used.

4 REDUCING DATA REDUNDANCY USING PARTICLE SWARM OPTIMIZATION

4.1 Monte Carlo method to measure data availability

Data availability is referred as the degree to which data can be instantly accessed. The term is specified in service level agreements of cloud storage provider. In distributed storage infrastructure, data availability is defined as the possibility of detecting k blocks out of the n redundant blocks so that erasure codes construct original data object from k redundant blocks. In [14], Lluis et al. defined the data availability d, as a function D (k, n, g, N) which relies on erasure

code parameters k and n, the data assignment function g and the set of nodes N.

Monte Carlo Sampling technique is used to measure the data availability of storage system in heterogeneous distributed infrastructure. The distributed storage system consists of N nodes. So there are 2^N possible combinations to store n redundant blocks. The data assignment function g assigns n redundant blocks to storage nodes based on their online availability.

4.2 Steps to measure data availability

a) The set S_V of V samples is taken from 2^N (i.e. $S_V \in 2^N$) randomly

b) The set A is the combination which together store k or more redundant data blocks is selected from S_V samples (i.e. $A \in S_V$) based on individual node's availability

c) The data availability d_V is measured by average of number of samples together store k or more redundant blocks to number of samples taken from 2^N i.e. $\left(d_V = \frac{A}{V} \right)$

4.3 Data assignment using particle swarm optimization

The data assignment function is used to assign redundant data blocks to set of storage nodes based on their online availability. The number of possible combinations to assign redundant data blocks is computationally hard for large scale storage systems. To solve best assignment of redundant blocks to storage nodes which maximizes the data availability, the Particle Swarm Optimization algorithm is used. The PSO is proved to be very efficient in distributed environment for performance. Optimization algorithms work by defining the workspace and to maximize the specified function. The search space is the set of possible combinations of storage nodes and the function to be maximized is data availability.

4.4 Algorithm for PSO

a) Initialize the swarm from the solution space that is randomly distributing n redundant blocks to set of storage nodes N

b) Evaluate the fitness function data availability d=(D, n, k, g)

c) Update individual and global bests

d) Update velocity and position of particles

The velocity is given by

$$V_i^{t+1} = wV_i^t + c_1 r_1 (lbest_i - X_i^t) + c_2 r_2 (gbest - X_i^t) \quad (1)$$

Here w is the inertia weight, r_1 and r_2 are random variables between [0, 1] and c_1 , c_2 are acceleration constants.

The position is given by

$$X_i^{t+1} = X_i^t + V_i^{t+1} \quad (2)$$

e) The step b is performed until the minimum data availability is met.

4.5 Finding optimal data redundancy

The optimal data redundancy is minimum data redundancy that is required to satisfy required data availability $\hat{d}$. The required data availability is taken as some nine's such as 0.9, 0.99, 0.9999 etc. The redundancy ratio r = $\frac{k}{n}$ which satisfies the minimum data availability is the optimal data redundancy. To find optimal redundancy r, the n value is fixed (i.e. n=|N|. β for large β) and the k value is changed from k=n to 1.

4.6 Algorithm for finding r

a) Set k=|N|

b) Calculate data availability D(n, k, g, N)

c) while D(n ,k, g, N) < $\hat{d}$ and k≠0

d) k=k-1

e) end while

5 PARALLEL PARTICLE SWARM OPTIMIZATION ALGORITHM FOR DATA ASSIGNMENT FUNCTION

The existing system uses PSO algorithm for data assignment function but the original PSO algorithm suffers from premature convergence and it can't suitable for parallel operation. The parallel version of particle swarm optimization algorithm divides the population into sub-population and applies the algorithm individually to these sub-populations to reduce computation time.

5.1 Algorithm for PPSO

a) Divide the population into k independent swarms randomly and initialize the individual swarm

b) The fitness function of each independent swarm is evaluated

c) The particle best and the swarm best of individual swarm is determined

d) The velocity and position of each particle in every swarm is updated

e) The steps 2 through 4 is repeated for N iterations

f) The global best is determined by comparing the swarm best of all the swarms. The fitness function data availability should be maximized for the global best. It is specified by the following equation

$$gbest = \max(sbest_1, sbest_2, sbest_3, ..., sbest_k) \quad (3)$$

g) The independent swarms are interacted using gbest value. The velocities of particles are updated according to the following equation

$$\begin{aligned} v_{i,j}(t) &= wv_{i,j}(t-1) \\ &+ c_1 r_1 (p_{i,j}(t-1) - x_{i,j}(t-1)) \\ &+ c_2 r_2 (p_{s,j}(t-1) - x_{i,j}(t-1)) \\ &+ c_3 r_3 (p_{g,j}(t-1) - x_{i,j}(t-1)) \end{aligned} \quad (4)$$

where $x_{i,j}$ is the position of ith particle in jth swarm, $v_{i,j}$ is the velocity of ith particle in jth swarm, $p_{i,j}$ is the pbest of ith particle in jth swarm, $p_{s,j}$ is the swarm best of jth swarm, p_g is the global best among all the sub-swarms, c_1, c_2, c_3 are acceleration parameters and r_1, r_2, r_3 are the random variables.

h) The position of all the particles are updated according to the following equation

$$x_{i,j}(t) = x_{i,j}(t-1) + v_{i,j}(t) \quad (5)$$

i) The steps 2 to 8 are repeated for M iterations and the output is the best optimal assignment of data blocks

6 EXPERIMENTAL RESULTS

6.1 Parameter Settings

The number of particles is 120 and the maximum number of iterations is 50. The acceleration parameters for PSO is c_1, c_2 =1 and the inertia weight parameter value for both PSO and PPSO is 0.7. The number of subgroups for parallel particle swarm optimization algorithm is 8. The acceleration parameters for parallel particle swarm optimization algorithm are c_1 =1, c_2 =1 and c_3 =1. The fitness function is data availability d= (D, k, n, g) which must meet the minimum data availability $\hat{d}$ = {0.9, 0.99, 0.999}.

6.2 Performance Metrics

a) Speedup

It is the measure used to compute the ratio between the average execution time on single processor $A[t_1]$ and the average execution time on multiple processors $A[t_k]$.The speedup due to k processors is defined by

$$S_k = \frac{A[t_1]}{A[t_k]} \quad (6)$$

TABLE 1
COMPARISON OF EXECUTION TIMES OF BOTH PSO AND PPSO

Cloudlet ID	Execution time of PSO	Execution time of PPSO
0	35.56	9.2838
1	108.8438	13.2625
2	234.1064	23.8726
3	340.7790	4.0867
4	378.0658	10.7202
5	426.2260	8.9963
6	472.4224	5.5739
7	512.396	6.9915

Here the k value chosen is 8. The execution times of both PSO and PPSO are shown in the table 1. The speedup of Parallel Particle Swarm Optimization is 3.2 times faster than original Particle Swarm Optimization.

b) Efficiency

The efficiency is defined as the percentage of speedup caused by k processors. It is given by

$$E_k = \frac{S_k}{k} * 100\% \quad (7)$$

From the experiments the Parallel Particle Swarm Optimization algorithm is 40% efficient than Particle Swarm Optimization algorithm. The results prove that the parallel version of particle swarm optimization algorithm is efficient for data assignment function in distributed cloud storage which reduces the computation time of data assignment function.

6.3 Redundancy savings

Redundancy Saving Ratio (RSR) is the metric used to measure the redundancy savings caused by distributed heterogeneous environment instead of homogeneous environment.

The data redundancies r_{homo} and r_{hetero} are caused by

homogeneous and heterogeneous storage systems. The Redundancy Saving Ratio is defined as

$$RSR = \left(1 - \frac{r_{homo}}{r_{hetero}}\right) * 100 \quad (8)$$

TABLE 2
THE REDUNDANCY SAVING RATIOS OF REAL AVAILABILITY DATASETS

Traces	Mean	Variance	|N|=100		
			$\hat{d}$=0.9	$\hat{d}$=0.99	$\hat{d}$=0.999
Skype	0.543	0.102	29.18%	32.09%	35.67%
PlanetLab	0.810	0.068	11.00%	12.03%	14.75%
Microsoft	0.834	0.075	15.32%	16.89%	20.78%
Seti@Home	0.664	0.110	20.01%	22.45%	31.67%

The number of storage nodes is 100. The fixed value for redundant blocks is n=100.|N| (i.e. β=100). The homogeneous environment uses unitary assignment function that is each block gets exactly one block. The heterogeneous environment uses parallel particle swarm optimization algorithm to assign data blocks to nodes based on their online availabilities. The four real time availability datasets are used for the experiments. The availability datasets are Planetlab nodes [19], Skype super-nodes [10], Microsoft desktop PCs [3], and Seti@-Home's desktop grid nodes[18]

The results from table 2 shows that the redundancy savings are high for highly heterogeneous environments

7 CONCLUSION AND FUTURE WORK

The proposed system uses parallel version of particle swarm optimization which reduces the computation time of the algorithm. The parallel particle swarm optimization algorithm assigns data blocks according to their online availabilities. By considering heterogeneous environment where each node have different online availabilities, the data assignment function assigns data blocks according to them. The redundancy savings are high almost 75% of redundancies are removed which also reduce storage and communication costs.

In the future work, the dynamic storage systems are considered. In this the number of nodes and the number of stored objects are continuously changed. To reduce redundancy for such environment a new scheme is needed.

REFERENCES

[1] D. P. Anderson (2004), 'Boinc: a system for public-resource computing and storage', Proceedings of the 5th IEEE/ACM International Workshop on Grid Computing

[2] C. Blake & R. Rodrigues (2003), 'High availability, scalable storage, dynamic peer networks', Proceedings of the 9th Workshop on Hot Topics in Operating Systems (HOTOS)

[3] W. J. Bolosky , J. R. Douceur, D. Ely & M. Theimer (2000), "Feasibility of a serverless distributed file system deployed on an existing set of desktop pcs", proceedings of the International Conference on Measurement and Modeling of Computer Systems ACM SIGMETRICS

[4] R. Campbell, I. Gupta, M. Heath, S. Y. Ko, M. Kozuch, M. Kunze, T. Kwan, K. Lai, H.Y. Lee, M. Lyons, D. Milojicic, D. O'Hallaron & Y. C. Soh (2009), ' Open cirrus cloud computing testbed: federated data centers for open source systems and services research', Proceedings of the Usenix Workshop on Hot Topics on Cloud Computing (HotCloud)

[5] A. Chandra & J. Weissman (2009), ' Nebulas: Using distributed voluntary resources to build clouds', Proceedings of the Usenix Workshop on Hot Topics on Cloud Computing (HotCloud)

[6] CleverSafe, 2010, Cleversafe, http://www.cleversafe.com

[7] Y. Deng & F. Wang (2007), 'A heterogeneous storage grid enabled by grid service', Proceedings on Small Interest Group on Operating Systems review (SIGOPS) , vol. 41, no. 1, pp. 7–13

[8] A. Dimakis, P. Godfrey, M. Wainwright & K. Ramchandran (2007), 'Network coding for distributed storage systems', Proceedings of the 26th IEEE International Conference on Computer Communications

[9] A. Duminuco & E. W. Biersack (2008), 'Hierarchical codes: how to make erasure codes attractive for peer-to-peer storage systems', Proceedings of the 8th International Conference on Peer-to-Peer Computing (P2P)

[10] B. Godfrey, Repository of availability traces,2010,<http://www.cs.berkeley.edu/pbg/availability/>

[11] T. Gonsalves & A. Egashira (2013), "Parallel Swarms Oriented Particle Swarm Optimization", Proceedings of Applied Computational Intelligence and Soft Computing 2013

[12] J. Kennedy & R. Eberhart (1995), 'Particle swarm optimization', Proceedings of the IEEE International Conference on Neural Networks

[13] J. Kubiatowicz, D. Bindel, Y. Chen, S. Czerwinski, P. Eaton, D. Geels, R. Gummad, S. Rhea, H. Weatherspoon, W. Weimer, C. Wells & B. Zhao (2000), 'Oceanstore: an architecture for global-scale persistent storage', Proceedings of the 9th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)

[14] W. K. Lin, D. M. Chiu & Y. B. Lee (2004), 'Erasure code replication revisited', Proceedings of the 4th International Conference on Peer-to-Peer Computing (P2P)

[15] P. Lluis, G. Pedro, S. Marc & H. Blas (2011), 'Towards the design of optimal data redundancy schemes for heterogeneous cloud storage infrastructures', Journal of Elsevier, vol. 55, no. 5, pp. 1100–1113

[16] L. Pamies-Juarez, P. García-López & M. Sánchez-Artigas, M (2009), 'Heterogeneity-aware erasure codes for peer-to-peer storage systems', Proceedings of the 38th IEEE International Conference on Parallel Processing (ICPP)

[17] R. Rodrigues & B. Liskov (2005), 'High availability in dhts: erasure coding vs replication', Proceedings of the 4th International Workshop on Peer-To-Peer Systems (IPTPS)

[18] Standford.edu, Folding@home: distributing computing project, 2009, <http://folding.stanford.edu>

[19] J. Stribling , Planetlab all pairs ping, 2010, <http://infospect.planetlab.org/pings>

[20] H. Weatherspoon & J. D. Kubiatowicz (2002),'Erasure coding vs. replication: a quantitative comparison', Proceedings of the 1st International Workshop on Peer-To-Peer Systems (IPTPS)

[21] WuaLa, 2010, Wuala, <http://www.wuala.com>.

[22] F. Wu, T. Qiu, Y. Chen & G. Chen (2005), 'Redundancy schemes for high availability in dhts', Proceedings of the 3rd International Symposium on Parallel and Distributed Processing and Applications (ISPA)

[23] Z. Zhang & Q. Lian (2002), 'Reperasure: replication protocol using erasure code in peer-to-peer storage network', Proceedings of the 21st Symposium on Reliable Distributed Systems (SRDS)