Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 50 (2015) 307 – 315

Procedia Computer Science

2nd International Symposium on Big Data and Cloud Computing (ISBCC'15)

An Efficient Job Scheduling in Isometric HPCLOUD using ZBLA Optimization

Doreen Hephzibah Miriam D[a], Prabha B[b], Felicia Lilian J[c]

[a]*Professor & Head,,*[b&c] *Assistant Professor,Department of Information Technology*
LICET,Loyola college Campus,Nungambakkam,Chennai 600034,India

Abstract

Efficient scheduling of jobs in the cloud environment has become the important criteria to determine the system performance. To improve the user's satisfaction in scheduling of jobs there should be an efficient optimization algorithm to distribute the jobs in cloud environment which eventually leads to dynamic allocation of resources in distributed environment. An efficient optimization algorithm named Zone-Based Bee Life Algorithm (ZBLA) has been proposed on a Hyper Cubic Peer-to-Peer cloud (HPCLOUD) in a structured peer-to-peer framework implemented on the cloud environment. We have compared ZBLA with the Bees life algorithm (BLA) which allocates the jobs faster in addition the network is fault tolerant and scalable with the Isometric partitioned distributed computing paradigm.

Peer-review under responsibility of scientific committee of 2nd International Symposium on Big Data and Cloud Computing (ISBCC'15)

*Keywords:*Peer-to-Peer; ZBLA; BLA; Cloud; Job scheduling

1. Introduction

Cloud computing is an attractive technology which brings challenging growth to the IT industry by providing rich services. NIST gave a definition of cloud computing that, Cloud Computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction [1]. Cloud Computing provides an efficient and scalable environment for the scheduling mechanism. The main confrontation in distributed system is to stabilize the load which is the main key issue in today's research areas. This leads to distribute the load across the enterprise boundaries in a heterogeneous environment. Even though, an intelligent and adjustable load balancing is a

Peer-review under responsibility of scientific committee of 2nd International Symposium on Big Data and Cloud Computing (ISBCC'15)
doi:10.1016/j.procs.2015.04.043

composite problem and it uses the best intelligent optimization algorithm and approaches. The model that is introduced in this paper is designed to suite the public cloud.

Public cloud holds a large number of nodes (VM's) with distributed resources in many different geographic locations. Hence this proposed model segregates the public cloud into numerous isometric cloud divisions. When the public environment is very huge and complex the isometric partitions can simplify the process of balancing the load. [2].

Interconnection network is the main domain of research for high scalable parallel computing systems that makes the message to be sent in a rapid fashion. The advancement in recent technologies has supported the massive computing elements to process the resources in the parallel distribution manner. This makes an eye opener for the researchers to throw light on the communication network topology. A critically designed massive parallel system can be easily interconnected with the help of well-defined network. The network topology that has a very large diameter and a very low message passing paths with more number of nodes is very expensive. So to overcome all the above- mentioned scenarios, we are considering the variant of the Hypercube network, the Hyper cubic Peer-to-peer Cloud (HPCLOUD), which imbibes isometric partitioning of the topology. It consists of $\mathbf{N = 2^n}$ node where, **n** is the Network dimension and **N** denotes the Total Nodes in the Network.

In this paper, we analyse node arrival and node departure characteristics using Bees life Algorithm [BLA][1] along with the following criteria, resource utilization and load balancing for efficient resource management on cloud. The rest of the paper is organized as follows. Section 2 gives an overview of the related work; section 3 explains the Topological structure of HPCLOUD. Section 4 describes the implementation of HPCLOUD load balancing using Bees Life Algorithm. Section 5 explains the job scheduling mechanism and the experimental characteristics of HPCLOUD. Section 6 presents the performance analysis of HPCLOUD in comparison with bee's life algorithm and zone based bee's life algorithm. Section 7 presents the conclusion and future enhancements.

2. Related work

2.1 Distributed Cloud computing environment maintaining the Integrity of the Specifications

The rapid growth of cloud computing has made IT sector a wide spread of opportunities in the domain. The system that is parallel and also distributed contains a cluster of interconnected and virtualized computers that are provisioned dynamically in unified computing resources [4]. Due to the exponential growth of cloud computing, it has been widely adopted by the industry and thus making a rapid expansion in availability of resource in the Internet. As the size of cloud scales up cloud computing service providers requires handling of massive requests. The primary challenge then becomes to keep the performance same or better whenever such an outburst occurs. Thus in spite of glorious future of Cloud Computing, many critical problems still need to be explored for its perfect realization [3]. The main objective of load balancing methods is to speed up the execution of applications on resources whose workload varies at run time in unpredictable way [10]. Load balancing techniques are widely discussed in homogeneous as well as heterogeneous environments such as grids and Cloud balancing moves global server load balancing from traditional routing options based on static data to context aware distribution across cloud based services, both internal and external. Cloud balancing provides an organization with the ability to distribute application requests across any number of application deployments located in data centers and through cloud-computing providers

Load Balancing allows distribution of workload across one or more servers, datacentres, hard drives, or other computing resources, thereby providing Cloud Service Providers (CSP) a mechanism to distribute application requests across any number of application deployments located in data centres. Load balancing mechanisms can be broadly categorized as centralized or decentralized, dynamic or static, and periodic or non-periodic Scheduling of tasks in cloud computing is an NP-hard optimization problem. Load balancing of non preemptive independent tasks on virtual machines (VMs) is an important aspect of task scheduling in clouds The main aim of cloud computing is to provide services to use the resources and gain maximum profit effectively. This service leads us to scheduling of the task as a core and competing issue in cloud computing. The processing of this service in the cloud environments

Identify applicable sponsor/s here. If no sponsors, delete this text box (*sponsors*).

are known to us as virtual machines (VMs). In user's point of view, the virtual machines should run the tasks efficiently as early as possible while they run in parallel. This is the prime area, which has taken us to problems in scheduling the customer tasks within the available resources [5]. The scheduler is responsible to do one task that is assigned to one or more VMs simultaneously which is called as load balancing problem [4][5].

The load-balancing algorithm tries to increase the response time of user's and hence ensuring that the resources are utilized to the maximum. The main purpose of load balancing is to speed up the execution of applications on the resources whose work-load varies during run time in an unpredictable way. Load balancing algorithm can be categorized into two different ways [5].

1) Static load balancing algorithm: The simplicity with respect to both implementation and over head is the main advantage of this algorithm, hence there is no necessity to monitor the performance of the nodes constantly. This algorithm is applicable when there is low variation in the load for the VMs. Therefore this algorithm is not suitable for grid and cloud computing environments, where the load will be varying at various points of time.

2) Dynamic load balancing algorithm: At run-time, this algorithm make changes to the distribution of work-load among the nodes and makes use of current load information while making distributed decisions.

2.2 Motivations of HPCLOUD

To inter connect a large number of processor in a multi-computer system we are proposing to use the hierarchical inter connection networks, which is also a cost effective way to inter connect massively parallel computers[5].

1) Hyper cube inter connection networks: The critical work is to determine the network topology that contains a numerous parallel systems in an inter connected network this affects the system performance drastically. The hypercube network also known as binary n-cube has been stated to be a very powerful topology [3],[6],[5]. The attractiveness of this Hypercube P2P topology as shown in Figure 1, is its small diameter, which is the maximum number of links (or hops) a message has to travel to reach its final destination between any two nodes. For hyper cubic network the diameter is identical to the degree of a node $n=\log_2 N$. There are 2^n nodes contained in the hypercube, each is uniquely represented by a binary sequence $b_{n-1}b_{n-2}..b_0$ of length n[6]. The nodes in the hypercube are adjacent if and only if they differ at exactly one bit position. This property greatly facilitates the routing of messages through the network. In addition, the regular symmetric nature of the network provides fault tolerance [5].

2) High performance computing in HPCLOUD: For past few years, the job-scheduling problem in cloud computing plays an attractive role among the researchers. The main advantage of job scheduling is to achieve a high performance.

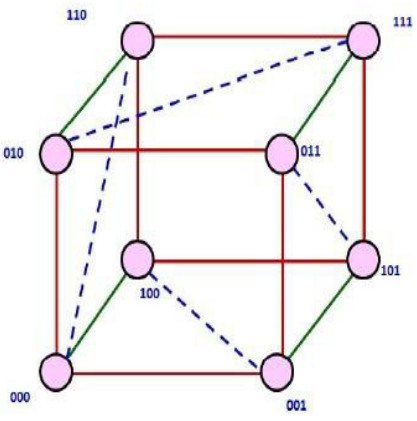

Fig.1. Topology of Hypercube P2P

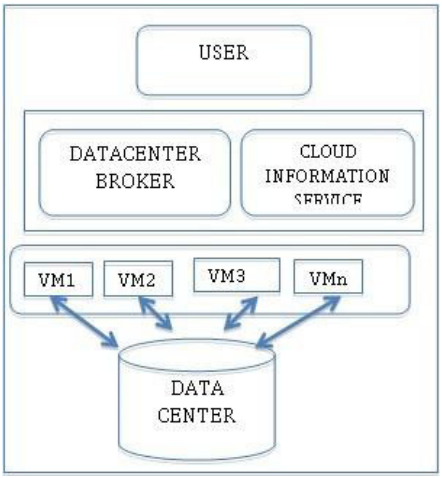

Fig.2. Job Scheduling in cloud computing

Computing and system throughput simultaneously and maintaining the efficiency and equality among the jobs and diminish the cost of execution [7],[8].

The job-scheduling algorithm in cloud can be categorized into two main classes,

1) Batch mode heuristic scheduling (BMHA)
2) Online mode heuristics algorithms (OMHA)

In BMHA, the jobs are taken into a collection when they arrive in the system (E.g.-Round Robin algorithm). While in OMHA, the jobs are scheduled in dividedly as soon as they arrive in the system (E.g.-Most fit task scheduling algorithm) [9] Some of the other job scheduling algorithms are First Come First Serve, Round Robin, Random algorithm, max-principle in the bee colony has been proposed in [1]. It is called marriage in honey bee optimization algorithm. The previous job scheduling algorithm provides the general method for reconfiguring a system that faces a dynamic work-load change, but doesn't explain about the high performance computing based on shared computations and storage over long distances [9].

A new algorithm known as bee swam optimization algorithm also called as Bees life Algorithm (BLA) is applied effectively to schedule the computations jobs among processing the resources into cloud data centres. This scenario is considered as NP-Complete problem as it aims to bring all the workloads among the processing resources in optimal fashion to reduce the total execution time of jobs and then to improve the effectiveness of the whole computing services [1],[7].

The performance of the system is an important aspect in cloud environment which is determined through the Load balancing of tasks. This makes cloud computing environment more efficient and thereby improves user satisfaction [12].

3. Topology of HPCLOUD

3.1 Structure of Hypercube P2P Cloud

A hyper cubic structure is a cube that has eight nodes named in order starting from 000 to111. Any node in the cube has three neighbours that it is directly connected to. There exists four extra links in the cube. These additional links minimize the distance of traversal between any two nodes to be almost two hops. The data centres are the nodes of the hypercube and they are mapped onto a cloud to form the Hyper Cubic peer-to-peer cloud. The names of the data centres are the three least significant bits of their binary representation. Thus the zeroth data centre is named 000 and the seventh data centre is 111. The Additional Neighbourhood Link (ANL) are introduced using the following algorithm [13]:

1) Flipping anyone of the n bits returns a neighbour of a given data centres. Each bit corresponds to a link in the network.
2) Consider the bits in position (2,1,0) LSB for additional neighbour link in n-Dimensional HPGRID system.
   - If the number of zero's in the last three bits of a node's bit address is Odd, then flip the bits in position 1 and 2.
   - If the number of zero's in the last three bits of a node's bit address is Even, then flip the bits in position 0 and 2.

3.2. Job scheduling using Zone based Bees Life Algorithm [ZBLA]

Scheduling of Jobs in cloud is analyzed through the behaviour of the bees by mutation and food-foraging process which is called as Bees Algorithm. This population-based meta heuristic process starts with a placing the scout bees in random manner in each search space. The best sites visited in point of view of fitness values are chosen as selected bees which will be foraged their close sites to carry out a neighbourhood search [1].

1) Bees Behaviour: The native bees that came from the continents of Africa and Europe, gain energy by feeding the nectar in their own hives and they use the pollens as a main source of protein in the rearing larvae. Generally a single breeding female bee called as Queen will be available in the Bee colony along with the Drones, the male bees that are thousand in number and the Workers, the sterile females and also the young larvae known as Broods. The communications between the bees are determined precisely by

two kinds of dances: One is called the round dance where the food is very near. This dance is performed when the bees search for their food. The Queen guarantees for the reproduction of bees'. The Queen bee in full flight mates with male bees, until her spermtheca becomes full. The egg that is unfertilized will produce the drone, while the egg that is fertilized produce the worker or queen depending on food quality [1],[7].

2) Description of Zone based Bees life Algorithm: The initial step in Zone Bees Life Algorithm initiates the bees population which carries 'N' bees (individuals) that are randomly chosen from the search space. In the second step fitness population fitness is evaluated. Generally the population of bee contains a queen (1), drones (D) and workers (W) in which the fittest bee will be the queen, the D fittest bees are called as drones and the left over bees are the workers. Consequently, the population size N will be the sum of the bees 1, D and W. The drones (D) and workers (W) are considered as user-defined parameters. In every cycle the bee population life contains two bee behaviours: reproduction and food foraging. In the first reproduction behaviour, the queen mates in the space by 'mating-flight' with the drones using the operator's crossover and mutation. In the next behaviour, the queen starts breeding N broods in step4. Then, the brood fitness is evaluated (steps5). If the fittest brood is fitter than the queen, it will be considered as the new queen for the next population. Moreover, D best bee individuals are chosen among the D fittest following broods and the drones of the current population to form the drones of the next population.

4. Implementation

4.1 Steps for implementing Bees Life Algorithm in HPCLOUD

This section we discuss about the implementation of Zone Bees Life Algorithm [ZBLA]. The efficient job scheduling in Peer-to-Peer environment can be studied with the help of this algorithm. Initially we have created 8 data centres each is capable of handling a job at any instance, so totally we have 8 jobs. The population in the environment is generated and the zone bees' life algorithm is implemented by the following algorithm:

```
Algorithm: ZBLA
    Begin
        Initial Population [N Jobs][M DC's], i=0;
            Fitness [DC_i] = population[1][1] + population [2][1]+…+population[N][M]
        SelectBestQueen(fitness[DC_i]);
        While QueenMate Drones do
            CreateBroods();
        End While
        NewPopulationCreated;
    End
```

```
Algorithm: Createbroods()
    Begin
        Initial i=0, j=1;
        While (Fitness [DC_i] > Fitness [DC_j]) do
            Swap(Fitness [DC_i], Fitness [DC_j])
            i++, j++;

        Reproduction Phase
            CROSSOVER
                rand1() and rand2() //Two-Point Crossover
                While Queen crossover DC_i do
                    crossover();
                new_broods;
            MUTATION
```

```
Select_task(DCi)

While (Fitness [DCi]> Fitness [DCj]) do
        Swap(Fitness [DCi], Fitness [DCj])
        i++, j++;
Threshold_value();

Food Foraging Phase
        Segregate_zone();
        Nearest_neighbour();
```

5. Experimental Results

5.1. Results of ZBLA

The initial step in Zone based bees life algorithm is initialization of population. Each population contains N individuals (solutions) where each one is represented by a set of datacentres. Each datacentres carried out a set of job tasks as follows:

$$DCTasks = DC_1Tasks, DC_2Tasks, \ldots\ldots, DC_mTasks: \text{set of datacentres}.$$

Each datacentre contains a set of job tasks as follows:

$$DC_jTasks = JTask_{j1}, JTask_{j2}, \ldots, JTask_{jN}$$

In order to evaluate the performance and the effectiveness of ZBLA to schedule tasks between datacentres in the cloud, we propose a set of tests carried on virtual cloud that contains 8 data centres. We assume that there are 8 jobs to be executed by the cloud datacentres after scheduling of their parts (Tasks).

INITIAL								
STEP-1	DC1	DC2	DC3	DC4	DC5	DC6	DC7	DC8
J1	10	11	12	10	10	11	13	8
J2	11	10	11	8	11	12	11	11
J3	13	11	13	9	12	8	12	12
J4	8	13	12	12	8	9	10	13
J5	9	12	13	8	9	10	10	9
J6	11	9	10	13	13	13	13	8
J7	10	12	10	11	9	10	8	10
J8	12	8	8	10	8	10	9	10
	84	**86**	**89**	**81**	**80**	**83**	**86**	**81**
	Drone1	Drone2	Drone3	Drone4	Queen	Drone6	Drone7	Drone8

Fig.3. Initial Population of ZBLA

Each job can be divided into tasks relevant to the existent data centres. Each task is defined by its identifier and the execution time. The above mentioned table describes about population of 8 datacentres and 8 jobs assigned. The subsequently step is to identify the Queen. This can be achieved by selecting the drone, which is having least fitness value. So here DC5 is identified as Queen.

To ensure the diversity in the generation of populations, two genetic operators are applied in the reproduction phase: crossover and mutation. Nevertheless, in the foraging phase, we define a greedy approach to ensure a neighbourhood search to in order optimize the found individual drones among its neighbours. In the following sections, these operators are illustrated.

1) Crossover:

It is a binary operator in which the queen mates with one drone randomly chosen. This process is repeated until reaching N individuals with crossover probability of queen.

2) Mutation:

It is a unitary operator in which the child has the chance to be mutated according to the mutation probability of queen. We propose the mutation operator in which a random selected task will be chosen and replaced by another random selected task in the same datacentre.

Threshold value is calculated as

Threshold value :(max_fitvalue+min_fitvalue)/2

In food foraging stage, new population is generated. Here we segregate the DC's into 4 different zones. There by find out the nearest neighbour for each DC that reaches in 1hop. Group the min 2's and find out whether it is lesser (or) equal to the threshold, if so then proceed further else skip this step.

Algorithm: Segregate_zone();
Initial:
DataCenter = 8 { 000, 001, 010, 011, 100, 101, 110, 111};
Nearest_neighbour(1HOP)
DC[1] → DC[5]
DC[2] → DC[6]
DC[3] → DC[7]
DC[4] → DC[8]
Zone = 4 { zone1[000,100], zone2[011,111], zone3[101,001], zone4[110,010]};
End

Select the weakest individual from the low fittest DC and the strongest individual from the high fittest DC. Then Replace them. Now, the generated broods form the next generation. Repeat the step still the fittest value reduces.

Table 1
Datacenter according to zones

Zone	Datacenter
Zone1	000,100
Zone2	011,111
Zone3	001,101
Zone4	010,110

Table 2.
Makespan of ZBLA and BLA

Makespan Calculation			
	Initial	After Population -1	After Population -2
BLA	89	90	89
ZBLA	89	86	86

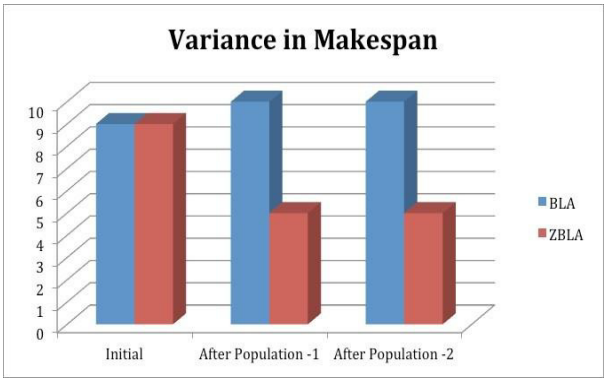

Fig .4. Variance of Makespan

6. Conclusion

In this paper, we have proposed the Zone based Bee Life Algorithm (ZBLA) which works efficiently to reduce the makespan. This indeed leads to effective job scheduling in cloud datacentre with load balancing. To prove the efficiency of this ZBLA algorithm, we have compared with the BLA algorithm and evaluated that the makespan varies along with the fitness range in the Data Centres. The segregation of zones made the makespan to be stabilized. The experimental results shows that ZBLA increases the efficiency and the performance in comparison with BLA. Specifically comparing the convergence rate, ZBLA is faster than BLA. The future work involves optimising the ZBLA for higher dimension of the network.

References

1. S. Bitam, "Bees life algorithm for job scheduling in cloud computing," in ICCIT, 2012, pp. 186–191.

2. H.-C. Hsiao, H.-Y. Chung, H. Shen, and Y.-C. Chao, "Load rebalancing for distributed file systems in clouds," IEEE Transactions on Parallel and Distributed Systems, vol. 24, no. 5, pp. 951–962, 2013.

3. D. D. H. Miriam and K. S. Easwarakumar, "HPGRID: A new resource management architecture with its topological properties for massively parallel systems," Int. J. Comput. Appl. Technol., vol. 43, no. 2, pp. 155–167, Mar. 2012.

4. M. A. T. Arabi E.Keshk, Ashraf B.El-Sisi, "Cloud task scheduling for load balancing on intelligent strategy," Int. J. Intelligent Systems and Applications, vol. 05, pp. 25–36,

5. M. Katyal and A. Mishra, "A comparative study of load bal- ancing algorithms in cloud computing environment," CoRR, vol. abs/1403.6918, 2014.

6. D. D. H. Miriam and K. S. Easwarakumar, "An efficient resource discovery methodology for HPGRID system," International Journal of Advanced Computer Science and Applications, vol. 2, no. 1, pp. 59–68, January 2011.

7. S.-J. Huang, X.-Z. Liu, W.-F. Su, and T.-C. Ou, "Application of enhanced honey-bee mating optimization algorithm to fault section estimation in power systems," Power Delivery, IEEE Transactions on, vol. 28, no. 3, pp. 1944–1951, July 2013.

8. N. R. Sabar, M. Ayob, G. Kendall, and R. Qu, "A honey- bee mating optimization algorithm for educational timetabling problems," European Journal of Operational Research, vol. 216, no. 3, pp. 533 – 543, 2012.

9. D. B. L.D. and P. Venkata Krishna, "Honey bee behavior inspired load balancing of tasks in cloud computing environ- ments," Appl. Soft Comput., vol. 13, no. 5, pp. 2292–2303, May 2013.

10. R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging {IT} platforms: Vision, hype, and reality for delivering computing as the 5th utility," Future Generation Computer Systems, vol. 25, no. 6, pp. 599 – 616, 2009.
11. I. A. Mohialdeen, "Comparative study of scheduling algo- rithms in cloud computing," Journal of Computer Science, vol. 09, no. 2, pp. 252–263, 2014.

12. G. Xu, J. Pang, and X. Fu, "A load balancing model based on cloud partitioning for the public cloud," Tsinghua Science and Technology, vol. 18, no. 1, pp. 34–39, Feb 2013.

13. R. Gangeshwari, J. Subbiah, K. Malathy, and D. Miriam, "Hpcloud: A novel fault tolerant architectural model for hierarchical mapreduce," in Recent Trends In Information Technology (ICRTIT), 2012 Insternational Conference on, April 2012, pp. 179–184.

14. Kousik Dasguptaa, Brototi Mandalb, Paramartha, Jyotsna Kumar Mondald, Santanu Dame, "A Genetic Algorithm (GA) based Load Balancing Strategy for Cloud Computing", Procedia Technology 10 (2013) 340 – 347

15. Dhinesh Babu L.D. a,∗, P. Venkata Krishna b, "Honey bee behavior inspired load balancing of tasks in cloud computing environments", Applied Soft Computing 13 (2013) 2292–2303

16. Ioannis A. Moschakis∗, Helen D. Karatza, “Multi-criteria scheduling of Bag-of-Tasks applications on heterogeneous interlinked clouds with simulated annealing”, The Journal of Systems and Software 101 (2015) 1–14

17. Fan Zhang,Junwei Ca,Keqin Li ,Samee U. Khand, Kai Hwang, “Multi-objective scheduling of many tasks in cloud platforms”, Future Generation Computer Systems 37 (2014) 309–320