International Journal of Hybrid Information Technology
Vol.8, No.1 (2015), pp. 249-256
http://dx.doi.org/10.14257/ijhit.2015.8.1.22

An Analysis of Swarm Intelligence based Load Balancing Algorithms in a Cloud Computing Environment

Uma Singhal[1] and Sanjeev Jain[2]

Madhav Institute of Technology and Science
Gwalior, India
[1]umasinghal5@gmail.com, [2]dr_sanjeevjain@yahoo.com

Abstract

The current advances in cloud computing layout well established research in Distributed computing, virtualization, web services, utility computing, have provided many advantages in scalability, cost and efficiency for cloud service users. These advantages are further expected to satisfy the demands of cloud users for cloud services in efficient and effective manner. This brings the issue of fault tolerance, scalability, efficiency, high availability. Central to these problems require the dynamic and efficient load balancing techniques. In this paper, we survey a special group of Swarm intelligence based load balancing algorithm and discuss the advantages and issues of these algorithms for cloud computing environment.

Keywords: *Cloud computing; Load balancing; Swarm Intelligence*

1. Introduction

Cloud computing is a new model of large-scale virtualization, distributed computing, software, network and web services. It has stimulated computing and data resides into large data centers [1] and away from desktop and manageable PCs. It has the ability to connect the power of Internet and wide area network (WAN) to make use of resources that are available remotely, hence providing cost-effective solution to most of the real life requirements [2]. It gives the scalable IT resources such as applications and services, coupled with the infrastructure on which they monitor over the Internet, on pay-as-you-go basis to maintain the capacity rapidly and easily. It supports to occupy changes in demand and helps any organization from investing too much capital cost for its hardware and software need [3-4]. Thereby, cloud computing is a structure for providing a suitable, on-demand network access to a common pool of computing resources. Cloud service model are of three types, as shown in Figure 1.

Cloud Software as a service (SaaS): The competence provided to the consumer is to make use of the provider's applications consecutively running on a cloud communications. The applications are genial to get from various client devices over a thin client interface for example a web browser. The consumer does not require dealing with the basic cloud infrastructure.

Cloud Platform as a Service (PaaS): The service provided to the consumer is to setup its own application on the cloud network. Consumer formed or obtained applications created by means of programming languages and tools sustained by the service provider. The customer does not need to supervise or control the basic cloud structure, but has command over the deployed applications and perhaps configuration setting for cloud domain.

Cloud Infrastructure as a Service (IaaS): The facility given to the customer is to provision processing, storage space, networks, and other basic computing resources where the consumer is capable to deploy and run random software, which contains operating systems

ISSN: 1738-9968 IJHIT

and placed applications. The consumer does not need to supervise the underlying cloud communications but has command over operating systems, placed applications, storage and perhaps limited control of select networking components.

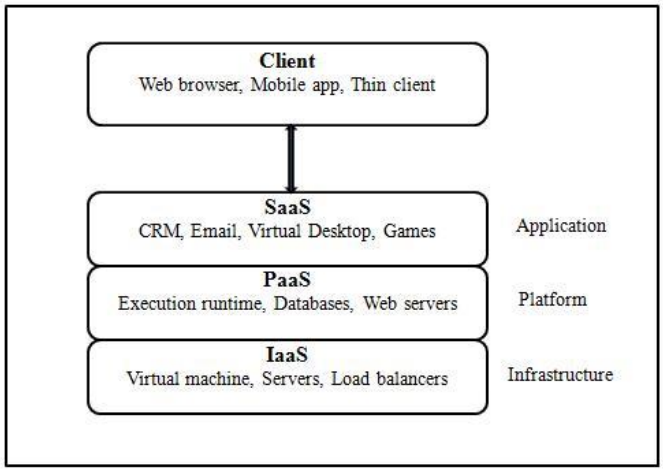

Figure 1. Cloud Service

The rest of this paper is organized as follows. In Section II, we discuss the various challenges and issues related to swarm intelligence based load balancing algorithms in cloud environment. Afterwards in Section III, we review various swarm intelligence based load balancing algorithms that are currently present in the literature. Section IV concludes the paper and accentuates future enhancements suitable for Load Balancing in cloud.

2. Issues Related to Swarm Intelligence based Load balancing Algorithms

In this section we give an introduction to the major challenges related to swarm intelligence based Cloud Load Balancing Algorithm that must be addressed before it is implemented in the cloud. These issues if not tackled properly may down the performance of the algorithm. These challenges are summarized as follows:

- If static swarm intelligence based load balancing algorithms are employed for static clouds. These algorithms perform well in stable cloud environment. However, Cloud environments are mostly dynamic and it is a challenging problem to design dynamic and flexible swarm intelligence based load balancing algorithm.
- Cloud environment comprises of heterogeneous resources but swarm intelligence based load balancing algorithms are of homogenous resources Resource aware load balancing algorithm can improve performance of the system so developing resource aware swarm intelligence based load balancing algorithm is a challenging task.
- Swarm intelligence based load balancing algorithms are generally centralized. Large scale cloud environment with single point of failure is a fearsome problem. Designing distributed swarm intelligence based load balancing algorithms is a challenging task.
- Swarm intelligence based load balancing algorithms generates large number of agents like particle swarm, honey bee or artificial ants. These agents help in performing the load balancing task. These agents also monitor the cloud. Establishing synchronization among these agents is a difficult task.

- As swarm intelligence based load balancing algorithms produce large number of agents. These agents distributed in the network and monitor the cloud nodes. This large number of agents adds the overhead in the cloud network which can down the system performance.
- Cloud computing platform is very huge in general. Therefore, swarm intelligence based algorithm must consider the scalability factor and the algorithm must be able to cope up with dynamic resource allocation without affecting the system performance.

3. Review of Swarm Intelligence based Load Balancing Algorithms

In this section we give a review of Swarm intelligence based load balancing algorithms that are currently implemented in the literature. Swarm intelligence based load balancing algorithms can be categorized in to two classes namely, Artificial Bee Colony Inspired Algorithms and Particle Swarm Optimization Inspired Algorithms. We first discuss the Artificial Bee colony Inspired Algorithms that have been developed for cloud load balancing. Then later discuss the Particle Swarm Optimization Inspired cloud load balancing algorithms.

3.1. Artificial Bee Colony Inspired Load Balancing Algorithms

Karaboga & Basturk [5-6] designed the artificial bee colony (ABC) algorithm, which is inspired by the intelligence behavior of bee colonies. They also concluded performance of ABC is better than PSO, Evolutionary Algorithm (EA) and Differential Evolution (DE) in case of numerical function optimization.

ABC is a population based optimization that simulates the behavior of honey bee for forging of food. It is a random search technique, used to find out the optimal solution of the various optimization problems. This algorithm work in the three phases: employed bees phase, onlookers and scouts phase. These phases are categorized on the basis of work done by them for searching of food. The search is carried out by the artificial bees can be summarized as follows: Employed bees find out a food source within the neighborhood, update the food source in their memory if they find something better otherwise count the trial of the search around the food source. They share their information with onlooker bees and then the onlookers select food sources within neighborhood on the basis of some probability. They update the food source if they find a better solution than employed bees otherwise increment the count of search around food source. If the number of trial count increases the predefined limit then scout bee phase start. In this phase, scout bee select new food source randomly to become employed again.

Here, food source (FS) represents a possible solution to the problem domain, which is half of the colony size (CS). FS is defined as in (3.1).

$$X = \{x_1, x_2, ..., x_N\} \quad x_i = \{x_{i,1,,}, x_{i,2}, ..., x_{i,D}\} \tag{3.1}$$

where N is total number of agent and D is the dimension of each agent. Randomly initialize each food source using following equation:

$$x_{i,d} = x_d^{\min} + rand\ (0,1)(x_d^{\max} - x_d^{\min}) \tag{3.2}$$

The employed bees select new solution randomly at tth iteration as given in:

$$z_{i,d}(t) = x_{i,d}(t) + \phi(x_{i,d}(t) - x_{k,d}(t)) \quad (3.3)$$

If fitness corresponding to new solution is better than before then update previous solution. When the employed phase has finished, employed bees share their best fitness with onlooker bees.

Onlooker bees calculate probability of estimating new solution around food source using following equation:

$$p_i(t) = \frac{fit_i(t)}{\sum_{i=1}^{FS} fit_i(t)} \quad (3.4)$$

If fitness evaluated in this phase is better than employed phase, store it otherwise increment number of trials. In both the employed bee and the onlooker bee phases, if number of trails exceeds the predetermined limit then source is supposed to be exhausted and algorithm enter into the scout bees phase.

In the employed phase, it is assumed that the current solutions are exhausted, while the scout bee phase is implemented in the algorithm to search the new solution for the problem. In scout bee phase, a scout bee finds a new random solution produced by equation (3.2) to be replaced with the exhausted food source.

The above mentioned procedure continues till maximum number of iteration reached or stopping condition reached. Pseudo code and flowchart of ABC algorithm is discussed below:

Table 3.3. Pseudocode of ABC Algorithm

a)	Randomly initialize food source (FS) with dimension D using equation (3.2) and other control parameters such as number of iteration, predetermined limit
b)	Evaluate fitness for each FS and also initialize trial corresponding to each FS to zero
c)	In employed phase, create new solution for each FS by (3.3) and evaluate them
d)	In onlooker phase, calculate probability using (3.4) for each solution, select solution according to calculated probability and evaluate fitness.
e)	If trail is greater than predetermined limit then randomly initialize solution using equation (3.3), evaluate fitness and set trial to zero.
f)	Record the best solution
g)	Repeat steps c) to f) until stopping condition reached

B. Kruekaew and W. Kimpan [7] have proposed a Virtual machine scheduling management on cloud computing using Artificial Bee Colony. In this work proposed ABC algorithm is combined with other algorithm as scheduling based on task size and Longest Job First (LJF) scheduling algorithm. Experimental results show that Combined ABC algorithms perform better in dynamic cloud environment and balance the workload which can minimize the total execution time and provide better resource utilization. This algorithm is suitable for

stability and scalability. However the algorithm suffers replication and network overhead problems.

Martin Randles, David Lamb and A. Taleb-Bendiab [8] have proposed a Honey bee inspired Load Balancing algorithm. The working of the algorithm can be explained as follows. The servers in the cloud capture the job of either foragers or scout. An advert board is maintained that simulates the waggle dance of the bees. A server successfully processing a request will display on the board. Any server that reads the board proceeds towards the chosen advert, and then fulfills the request; hence displaying the scout bee. If the server not reading the advert board makes a random virtual server's queue request; thus displaying the forager bee. A server process a request, evaluate its profit and compare with colony profit. If this server's profit was high than colony profit it continues with current virtual server and broadcast this information. Otherwise, the server continues to act as forgers or scouts. The algorithm is good for large scale, heterogeneous and changing cloud environments. The algorithm suffers with Network Overhead issue.

Jing Yao and Ju-hou He [9], have proposed a Load Balancing Strategy for cloud computing based on Artificial Bee algorithm. This algorithm is an enhanced version of the algorithm proposed in [8]. The previous algorithm was balancing only lightly loaded nodes where the mechanism in [9] examines all resources in the cloud iteratively. The algorithm improves the quality of service by using some extra operation. Experimental analysis conclude that in a system with a fixed number of host and virtual machine and growing number of requests the algorithm produces an improved throughput whereas a change in number of servers and definite number of requests the original ABC algorithm executes well. In case of hundreds of thousands of request load balancing is declined. The improved ABC algorithm is dynamic, scalable, distributed, and appropriate for heterogeneous cloud systems have proposed.

3.2. Particle Swarm Optimization (PSO) based Load Balancing Algorithms

Particle swarm optimization (PSO) [10] is stochastic optimization which was proposed by Eberhart and Kennedy in the middle 90's. It is population based evolutionary algorithm which is inspired by social behavior of birds flock. They search the food by communication among neighbours and from their own experience. PSO uses the same concept of bird's flock for the optimal solution of problem. Each particle (agent) flown through the problem search space towards the optimal solution by the learning from its own experience *i.e.*, cognitive component and learning from their neighbors *i.e.*, social component. Each particle keep track of its best previous location, known as pbesti, which is calculated by fitness function and best location among all particles that is the global best solution is denoted by gbest.

On each step of algorithm, every particle decides where to move next, considering its own experience, that is the memory of its best past position and the experience of its most successful particle in the swarm. So, velocity of each particle is updated by the following equation:

$$v_{i,d}(t+1) = w \times v_{i,d}(t) + c_1 \times r_1(pbest_{i,d} - x_{i,d}) + c_2 \times r_2(gbest - x_i) \quad (3.5)$$

$$x_{i,d}(t+1) = x_{i,d}(t) + v_{i,d}(t+1) \quad (3.6)$$

In the above equation (3.5), current velocity of a particle is calculated by previous velocity, previous best value of that particle (pbesti) and global best value of the population, c1 is coefficient of the cognitive component, c2 is coefficient of the social component, w is known

as inertia factor, r1, r2 are the random numbers uniformly distributed in the interval [0, 1]. In equation (3.6), current position of each particle is determined by previous position and current velocity of particle. Pseudocode and flowchart of PSO algorithm is given below:

Table 3.3. Pseudocode of ABC Algorithm

Randomly initialize food source (FS) with dimension D using equation (3.2) and other control parameters such as number of iteration, predetermined limit
Evaluate fitness for each FS and also initialize trial corresponding to each FS to zero
In employed phase, create new solution for each FS by (3.3) and evaluate them
In onlooker phase, calculate probability using (3.4) for each solution, select solution according to calculated probability and evaluate fitness.
If trail is greater than predetermined limit then randomly initialize solution using equation (3.3), evaluate fitness and set trial to zero.
Record the best solution
Repeat steps c) to f) until stopping condition reached

Elina Pacini, Cristian Mateos, and Carlos García Garino [11] have proposed dynamic scheduling based on Particle Swarm Optimization for cloud based Scientific Experiments. In this work, Cloud scheduler based on Particle Swarm Optimization (PSO) have proposed and evaluated. The main performance metrics that are examined in this work are total no of VMs created and the number of cloud users that this scheduler is able to efficiently serve. Experimental analysis is conducted with job data from real world scientific problem. Also comparison is made with other schedulers of Random assignment and Genetic algorithm. The results have shown that proposed scheduler successfully balance above examined metrics as compared to scheduler placed on Random assignment and Genetic algorithm. However this algorithm needs to be examined with heterogeneous machines.

Suraj Pandey, LinlinWu, Siddeswara Guru, and Rajkumar Buyya [12] have proposed A Particle Swarm Optimization (PSO)-based Heuristic for Scheduling Workflow Applications in Cloud Computing Environments. Cloud computing provide applications by virtualizing resources. Users are charged for its application based on pay-as-you-go model. User application may suffer large execution and data retrieval cost. In this work, proposed particle swarm optimization (PSO) based scheduling heuristic consider both computation and data transmission cost for data insistent applications. The proposed algorithm is examined with a workflow application by changing its communication and computation costs. The proposed PSO is compared with existing 'Best Resource Selection' (BRS) algorithm. Experimental analysis concluded that 3 times cost savings and equal distribution of load onto resources when using PSO as compared to existing BRS algorithm.

6. Conclusion and Future Work

Load balancing is the most important concern for delivering cloud services efficiently. In this paper, we present a survey on swarm intelligence based cloud load balancing algorithms and challenges caused by these algorithms when using for cloud environment. The entire algorithms analyzed in this paper suffer from one or two issues. Therefore, there is still room for improvement in the algorithms. Hence, in future we aim to introduce our own swarm intelligence based algorithm that will take care of discussed issues.

References

[1] N. J. Kansal and I. Chana, "Cloud Load balancing techniques: A step towards green computing", IJCSI International Journal of Computer Science Issues, vol. 9, no. 1, (**2012**) January.
[2] B. P. Rimal, E. Choi, and I. Lumb, "A Taxonomy, Survey, and Issues of Cloud Computing Ecosystems, Cloud Computing: Principles, Systems and Applications", Computer Communications and Networks, Chapter 2, pp. 21-46, DOI 10.1007/978- 1-84996-241-42, Springer – Verlag London Limited, (**2010**).
[3] R. W. Lucky, "Cloud computing", IEEE Journal of Spectrum, vol. 46, no. 5, (**2009**) May, pp. 27- 45.
[4] M. D. Dikaiakos, G. Pallis, D. Katsa, P. Mehra, and A. Vakali, "Cloud Computing: Distributed Internet Computing for IT and Scienti_c Research", IEEE Journal of Internet Computing, vol. 13, no. 5, (**2009**) September/October, pp. 10-13.
[5] B. Basturk and D. Karaboga, "An artificial bee colony (ABC) algorithm for numeric function optimization". In IEEE Swarm intelligence symposium, (**2006**), pp. 12–14.
[6] D. Karaboga and B. Basturk, "Artificial bee colony (abc) optimization algorithm forsolving constrained optimization problems". Foundations of Fuzzy Logic and Soft Computing, (**2007**), pp. 789–798.
[7] B. Kruekaew and W. Kimpan, "Virtual Machine Scheduling Management on Cloud Computing Using Artificial Bee Colony", Proceedings of the International Multi-Conference of Engineers and Computer Scientists 2014, vol. 1, IMECS 2014, (**2014**), March 12 - 14, Hong Kong.
[8] M. Randles, E. Odat, D. Lamb, O. Abu- Rahmeh and A. Taleb Bendiab, "A Comparative Experiment in Distributed Load Balancing", Second International Conference on Developments in eSystems Engineering (**2009**).
[9] J. Yao and J.-h. He, "Load Balancing Strateg y of Cloud Computing based on Artificial Bee Algorithm", 8th International conference on computi ng technology and Information Management, vol. 1, (**2012**).
[10] J. Kennedy and R. Eberhart. Particle swarm optimization. In Neural Networks, 1995. Proceedings, IEEE International Conference on, vol. 4, (**1995**), pp. 1942–1948.
[11] E. Pacini, C. Mateos and C. García Garino, "Dynamic Scheduling based on Particle Swarm Optimization for Cloud-based Scientific Experiments", CLEI Electronic Journal - Special Issue HPCLatam 2013. In press. CLEI (Latin-American Center for Informatics Studies), (**2014**), ISSN 0717-5000, [Latindex].
[12] S. Pandy, L. Wu, S. M. Guru and R. K. Buyya, "A Particle Swarm Optimization-Based Heuristic for Scheduling Workflow Application in Cloud Computing Environments", Advance Information Networking and Applications, IEEE International Conference, (**2010**), pp. 400-407.

Authors

Uma Singhal, has received the B.E. degree in Computer Science and Engineering, from MPCT, Gwalior, India in 2011. Currently she is pursuing M.Tech in Computer Science and Engineering from MITS, Gwalior and it will be completed in 2014. Her research include Cloud Computing and Adhoc Network.

Sanjeev Jain, he is a Director, MITS Gwalior, India. He has 27 years of teaching experience, including 7 year administrative experience as Director of MITS a Grant in aid Autonomous Engineering College of Madhya Pradesh. His teaching and research include Image Processing and Data Mining, Computer Network.

