Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Technology 4 (2012) 783 – 789

Procedia Technology

C3IT-2012

Load Balancing in Cloud Computing using Stochastic Hill Climbing-A Soft Computing Approach

Brototi Mondal[a,*], Kousik Dasgupta[a], Paramartha Dutta[b]

[a]*Dept. of Computer Science and Engineering, Kalyani Government Engineering College, Kalyani, India*
[b]*Dept. of Computer and System Sciences, Visva-Bharati University, West Bengal, India*

Abstract

Cloud Computing, a new concept is a pool of virtualized computer resources. An Internet-based development where dynamically scalable and often virtualized resources are provided as a service over the Internet has become a significant issue. Cloud computing describes both a platform and type of application. A cloud computing platform dynamically provisions, configures, reconfigures, and deprovisions servers as needed. Servers in the cloud can be physical machines or virtual machines spanned across the network. Thus it utilizes the computing resources (service nodes) on the network to facilitate the execution of complicated tasks that require large-scale computation. Selecting nodes (load balancing) for executing a task in the cloud computing must be considered, and to exploit the effectiveness of the resources, they have to be properly selected according to the properties of the task. In this paper, a soft computing based load balancing approach has been proposed. A local optimization approach Stochastic Hill climbing is used for allocation of incoming jobs to the servers or virtual machines(VMs). Performance of the algorithm is analyzed both qualitatively and quantitatively using CloudAnalyst. CloudAnalyst is a CloudSim-based Visual Modeller for analyzing cloud computing environments and applications. A comparison is also made with Round Robin and First Come First Serve (FCFS) algorithms.

Keywords: Cloud computing, load balancing, soft computing, stochastic hill climbing, CloudAnalyst

1. Introduction

Internet technologies is fast growing and is being used extensively, with it *Cloud Computing* became a hot topic of industry and academia as an emerging new computing mechanism. It is supposed to provide computing as the utility to meet the everyday needs of the general community [1, 2]. Its infrastructure is used by businesses and users to access application services from anywhere in the world on demand. Thus it represents as a new paradigm for the dynamic provisioning of computing services, typically supported by state-of-the-art data centers containing ensembles of networked Virtual Machines [3]. It is a distributing computing mechanism that utilizes the high speed of the internet to move jobs from private PC to the remote computer clusters (big data centers owned by the cloud service providers) for data processing.

*Corresponding author
Email address: brototi.snp@gmail.com (Brototi Mondal)

2212-0173 © 2012 Published by Elsevier Ltd. Open access under CC BY-NC-ND license.
doi:10.1016/j.protcy.2012.05.128

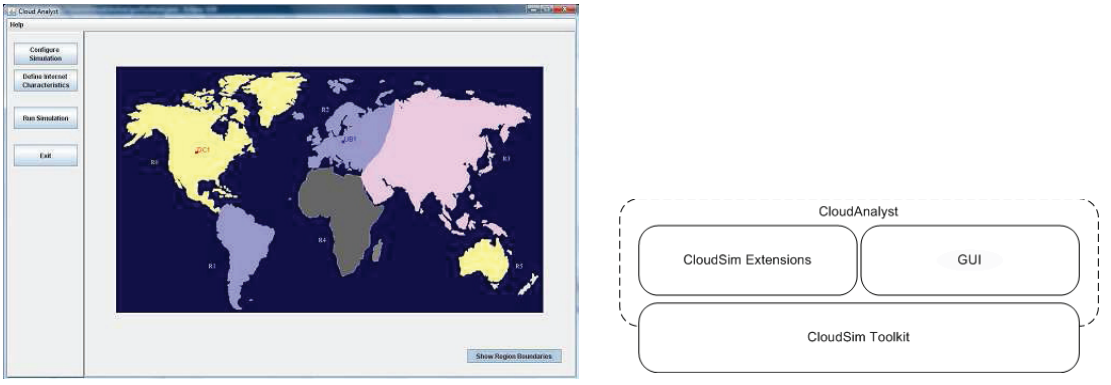
Though there is a glorious future of Cloud Computing, many crucial problems still need to be solved for the realization of cloud computing. Load balancing is one of these problems, it plays a very important role in the realization of Cloud Computing. Load balancing in cloud computing is to distribute the local workload evenly to the whole cloud. in fact it has become indispensable for cloud computing. It is used by Cloud service provider (CSP) in its own cloud computing platform to provide a high efficient solution for the user. Also, a inter CSP load balancing mechanism is needed to construct a low cost and infinite resource pool for the consumer. Thus Load balancing in cloud computing 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.

Several approaches of Load Balancing exist in literature. In [4]Minimum Execution Time (MET) is used to assign each job in arbitrary order to the nodes on which it is expected to be executed fastest, regardless of the current load on that node. Whereas in [5] Min-Min scheduling algorithm the minimum completion time for every unscheduled job is calculated. Then the jobs are assigned with the minimum completion time to the node that offers it this time. A Round-robin algorithm uses simple distribution of jobs among all data centers or processing units. This paper proposes a Stochastic hill climbing approach for the load balancing for maximum optimization of available resources. CloudAnalyst [6]-A CloudSim based Visual Modeler has been used for simulation and analyzing of the algorithm. A comparative study is also done with Round-robin algorithm and First Come First Serve (FCFS) and results are found to be encouraging.

The remaining part of the paper is arranged as follows. A brief introduction to the simulation tool CloudAnalyst is made in Section 2 and Section 3 describes the proposed algorithm. In Section 4 simulation results are given. Whereas Section 5 gives conclusion and future work.

2. A Brief review on CloudAnalyst- The simulation tool

As Cloud computing allows deployment of large scale applications easier and cheaper, it also creates new issues for researchers. To test these new issues researchers needs some testbed. Also Cloud infrastructures are distributed, applications can be deployed in different geographic locations, and its impact on performance is felt far from the data center. Quantifying impact of number of simultaneous users, geographic location of relevant components, and network in applications is hard to achieve in real testbeds, because of the presence of elements that cannot be predicted nor controlled by developers. Although simulation can be done using CloudSim [7] but it requires building of the environment and its related properties. **CloudAnalyst**[9] allows us to separate the simulation experimentation exercise from a programming exercise, so a researcher can focus on the simulation complexities without spending too much time on the technicalities of programming. A snapshot of the CloudAnalyst simulation toolkit is shown in figure 1(a) and its architecture is in figure 1(b).



(a) (b)

Fig. 1. (a) A Snapshot of CloudAnalyst (b)CloudAnalyst Architecture

3. Load Balancing using A Stochastic Hill Climbing approach

Load Balancing is a process to make effective resource utilization by reassigning the total load to the individual nodes of the collective system and to improve the response time of the job. For developing strategy for load balancing the main points to be considered are estimation of load, comparison of load, stability of different system, performance of system, interaction between the nodes, nature of work to be transferred, selecting of nodes [8]. This load considered can be in terms of CPU load, amount of memory used, delay or Network load.

Load Balancing algorithms can be be distributed or non-distributed (centralized). Our approach is in the later type where the load balancing algorithm is executed only by a single node in the whole system (the central node). This node is solely responsible for load balancing of the whole system. The other nodes interact only with the central node. Centralized load balancing takes fewer messages to reach a decision, as the number of overall interactions in the system decreases drastically as compared to the distributed case. However, centralized algorithms can cause a bottleneck in the system at the central node and also the load balancing process is rendered useless once the central node crashes. The first disadvantage can be taken care if we make the load distribution more effective. This can be considered as an optimization problem where loads are distributed among the available servers to achieve an effective throughput. Over the years soft computing has been used as a effective optimization tool in the next section we describe our proposed approach.

3.1. Stochastic Hill Climbing Algorithm for Load balancing in Cloud Computing

There are two main families of procedures for solving a optimization problem. *Complete methods* which guarantees either to find a valid assignment of values to variables or prove that no such assignment exists. These methods frequently exhibit good performance, and guarantee a correct and optimal answer for all inputs. Unfortunately, they require exponential time in the worst case, which is not acceptable in the cloud computing domain. The other *Incomplete methods* may not guarantee correct answers for all inputs. Rather these methods finds satisfying assignments for solvable problems with high probability. These algorithms have gained popularity in recent years, due to their simplicity, speed and observed effectiveness at solving certain types of problems. A variant of Hill Climbing algorithm **Stochastic Hill Climbing** [9](SHC) is one of the incomplete approaches for solving such optimization problems. A stochastic and Local Optimization algorithm is simply a loop that continuously moves in the direction of increasing value, which is uphill. It stops when it reaches a ”peak” where no neighbor has a higher value. This variant chooses at random from among the uphill moves and the probability of selection can vary with the steepness of the uphill move. Thus it maps assignments to a set of assignments by making minor changes to the original assignment. Each element of the set is evaluated according to some criteria designed to move closer to a valid assignment to improve the evaluation score of the state. The best element of the set is made the next assignment. This basic operation is repeated until either a solution is found or a stopping criteria is reached. So it has two main components a candidate generator which maps one solution candidate to a set of possible successors, and a evaluation criteria which ranks each valid solution (or invalid full assignments), such that improving the evaluation leads to better (or closer to valid) solutions.

The proposed algorithm is described as given below:

Step 1: Maintain an index table of Virtual Machine servers (VMs) and the state of the VM BUSY/AVAILABLE At the start all VMs are available.

Step 2: A new job arrives in the cloud.

Step 3: Generate query for the next allocation.

Step 4: Generate a VM id randomly.

Step 5: Parse the allocation table from to get the status of the particular VM.
If the VM is found unallocated:

Step 5a: Return the VM id.

Step 5b: Send the request to the VM identified by that id.

Step 5c: Update the allocation table accordingly.
If the VM is found to be allocated:
Step 5d: Use a random function to generated a random VM.
Step 5e: Select the VM for allocation to the job with a probability such that this VM will be able to handle the job efficiently.
Step 5f: Keep account of performance of the VM if it does not perform according to expectation (cost value) decrease its probability for assignment in next iteration.
Step 5g: Update the allocation table accordingly.
Step 6: When the VM finishes processing the request, and the response cloudlet is received. Generate a notification of VM de-allocation..
Step 7: Continue from Step 2 for next allocation.

4. Simulation results and analysis

For testing the algorithm CloadAnalyst has been used. A hypothetical configuration has been generated keeping in mind the application of cloud in e-auction and social networking sites like Facebook, google+ etc.

4.1. Simulation configuration

Six user bases representing the six major continents of the world is considered. Further for simplicity each user base is contained within a single time zone and it is assumed out of the total registered users 5 % are online simultaneously during the peak time and only one tenth is on line during the off-peak hours. Furthermore, each user makes a new request every 5 minutes. Each simulated data center hosts a particular amount of virtual machines dedicated to the application. Machines have 4 GB of RAM and 100GB of storage and each machine has 4 CPUs, and each CPU has a capacity power of 10000 MIPS. Such user bases used for experimentation are described in Table 1.

Table 1. Simulation configuration

S.No	User Base	Region	Simultaneous Online Users During Peak Hrs	Simultaneous Online Users During Off-peak Hrs
1	UB1	0-N.America	6,000	600
2	UB2	1-S.America	2,000	200
3	UB3	2-Europe	5,000	500
4	UB4	3-Asia	7,000	700
5	UB5	4-Africa	1,000	100
6	UB6	5-Oceania	1,500	150

4.2. Simulation Scenarios

For simulation purpose several scenarios are considered. To start with a single single centralized Cloud Data Center (DC) is considered to host the social network application. So all requests from all users around the world are processed by this single DC. This DC has 25,50 and 75 VMs allocated to the application at each cloud configuration(CCs). The Simulation scenario is described in Table 2(a).

In the next scenario we consider two DCs with each having initially 25,50 and 75 VMs allocated to the application in each CCs. Then each DCs have 25 and 50 VMs, 25 and 75 VMs and 50 and 75 VMs allocated to the application in each CCs as given int Table 2(b). Next three DCs are considered with initially each having 25,50 and 75 VMs for each CCs. Further it is extended to a mixture of 25,50 and 75 VMs for each DCs as given in table 3(a).

Similarly four, five and six DCs are considered with configuration as given in tables 3(b),4(a) and4(b).

4.3. Results

With the scenario and configuration as mentioned in the previous subsections, overall average response time (RT)in (milliseconds) is calculated. The results are taken for the Stochastic Hill Climbing Algorithm (SHC), Round-Robin (RR) and First Come First Serve ($FCFS$) and given in Tables 2(a),2(b),3(a),3(b),4(a) and 4(b). Figures 2(a),2(b),3(a),3(b),4(a) and4(b) depicts a graphical overview of the performance of the Stochastic Hill Climbing Algorithm with respect to RR and $FCFS$. The results show that in most of the case Stochastic Hill Climbing (SHC) outperforms the other two approaches.

Table 2. Simulation scenarios and calculated overall average response time (RT) (a) using one data center (b) using two data centers

(a)

S.No	Cloud Configuration	DC specification	RT in (*ms*) using SHC	RT in (*ms*) using RR	RT in (*ms*) using FCFS
1	CC1	Each with 25 VMs	328.02	329	329.23
2	CC2	Each with 50 VMs	290.1	290.23	290.37
3	CC3	Each with 75 VMs	245.34	245.67	245.94

(b)

S.No	Cloud Configuration	DC specification	RT in (*ms*) using SHC	RT in (*ms*) using RR	RT in (*ms*) using FCFS
1	CC1	Each with 25 VMs	302.34	308.17	311.34
2	CC2	Each with 50 VMs	292.25	306.49	309.52
3	CC3	Each with 75 VMs	289.73	301.78	304.56
4	CC4	Each with 25,50 VMs	290.01	300.21	304.87
5	CC5	Each with 25,75 VMs	287.23	294.45	297.23
6	CC6	Each with 50,75 VMs	285.14	289.61	291.01

Table 3. Simulation scenarios and calculated overall average response time (RT) (a) using three data centers (b) using four data centers

(a)

S.No	Cloud Configuration	DC specification	RT in (*ms*) using SHC	RT in (*ms*) using RR	RT in (*ms*) using FCFS
1	CC1	Each with 25 VMs	293.82	303.17	305.34
2	CC2	Each with 50 VMs	289.25	291.49	293.52
3	CC3	Each with 75 VMs	286.73	289.78	290.56
4	CC4	Each with 25,50,75 VMs	290.01	300.21	304.87

(b)

S.No	Cloud Configuration	DC specification	RT in (*ms*) using SHC	RT in (*ms*) using RR	RT in (*ms*) using FCFS
1	CC1	Each with 25 VMs	264.35	277.35	281.05
2	CC2	Each with 50 VMs	261.71	268.93	275.27
3	CC3	Each with 75 VMs	257.46	261.09	267.79
4	CC4	Each with 25,50,75 VMs	251.31	258.21	263.94

Table 4. Simulation scenarios and calculated overall average response time (RT) (a) using five data centers (b) using six data centers

(a)

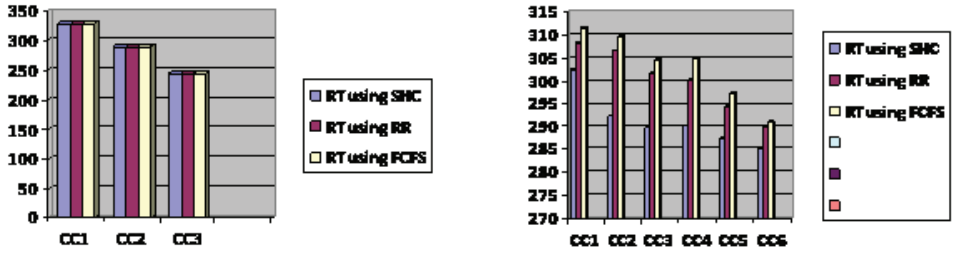
S.No	Cloud Configuration	DC specification	RT in (*ms*) using SHC	RT in (*ms*) using RR	RT in (*ms*) using FCFS
1	CC1	Each with 25 VMs	235.86	243.57	251.03
2	CC2	Each with 50 VMs	230.84	238.06	245.44
3	CC3	Each with 75 VMs	229.46	233.88	242.79
4	CC4	each with 25,50,75 VMs	225.64	231.16	238.01

(b)

S.No	Cloud Configuration	DC specification	RT in (*ms*) using SHC	RT in (*ms*) using RR	RT in (*ms*) using FCFS
1	CC1	Each with 25 VMs	228.96	240.97	249.26
2	CC2	Each with 50 VMs	227.56	237.34	244.04
3	CC3	Each with 75 VMs	225.78	231.67	239.87
3	CC4	Each with 25,50,75 VMs	223.56	231.496	238.97

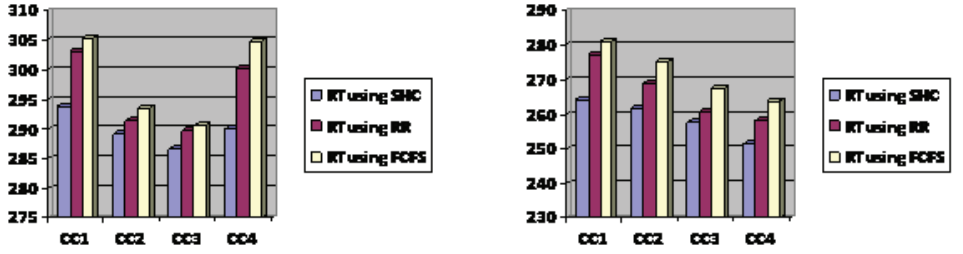
5. Conclusion

In this paper a stochastic hill climbing approach has been used for load distribution in Cloud computing environment. The soft computing based approach has been compared with two approaches Round Robin and First Come First Serve. The results are quite encouraging however use of other soft computing techniques are needed to be studied for further improvement. The authors are presently working on the above.



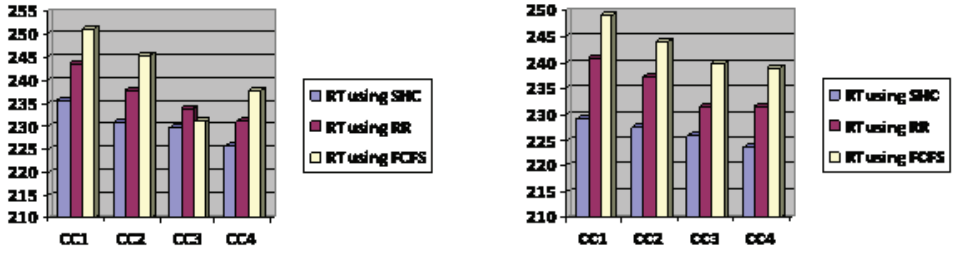
(a) (b)

Fig. 2. Performance analysis using SHC, RR and FCFS (a)for one data center (b)two data centers



(a) (b)

Fig. 3. Performance analysis using SHC, RR and FCFS (a)for three data centers (b)four data centers



(a) (b)

Fig. 4. Performance analysis using SHC, RR and FCFS (a)for five data centers (b)six data centers

References

[1] R.Buyya, C. Yeo, S.Venugopal, J.Broberg, I.Brandic, Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility, in: Future Generation Computer Systems, vo1.25, 2009, pp. 599–616.
[2] G. Boss, P. Malladi, D. Quan, L. Legregn, Cloud computing, in: High Performance On Demand Solutions (HiPODS), IBM, 2007.
[3] R. R.Buyya, R.Ranjan, Intercloud: Utility-oriented federation of cloud computing environments for scaling of application services, in: ICA3PP 2010, Part I, LNCS 6081., 2010, pp. 13–31.
[4] T. R.Armstrong, D.Hensgen, The relative performance of various mapping algorithms is independent of sizable variances in run-time predictions, in: 7th IEEE Heterogeneous Computing Workshop (HCW '98), 1998, pp. 79–87.
[5] A. Vouk, Cloud computing- issues, research and implementations, in: Information Technology Interfaces, 2008, pp. 31–40.
[6] B.Wickremasinghe, R.N.Calheiros, R. Buyya, Cloudanalyst: A cloudsim-based visual modeller for analysing cloud computing

environments and applications, in: Proceedings of the 24th International Conference on Advanced Information Networking and Applications (AINA 2010), Perth, Australia,, 2010.

[7] R.N.Calheiros, R. Ranjan, A. Beloglazov, C. Rose, R. Buyya, Cloudsim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms, in: Software: Practice and Experience (SPE), Volume 41, Number 1, ISSN: 0038-0644, Wiley Press, New York, USA., 2011, pp. 23–50.

[8] A. M. Alakeel, A guide to dynamic load balancing in distributed computer systems, in: IJCSNS International Journal of Computer Science and Network Security,VOL.10 No.6., 2010, pp. 153–160.

[9] S. Russell, P. Norvig, Artificial intelligence: A modern approach 3/e, in: Pearson Publication, ISBN-10: 0136042597., 2010.