Cloud Resource Allocation Schemes Using Drip Irrigation Approach

R. Madhumathi

R. Radhakrishnan

K. Veningston

P. Sengottaiyan

Abstract

Managing resources in cloud introduce the challenge of on-demand and optimistic resource allocation in response to service requests. Typically cloud computing offers an elastic infrastructure that resource provider agent can use to obtain resources (e.g., hardware, software resources) that meet the demand. Resource provider agents are charged for the amount of resources allocated in the cloud. In this respect, an issue is to decide on the right amount of resources reserved in the cloud, and its reservation time so that the financial cost on the resource provider agent is minimized. This paper proposes a simple and easy method to implement algorithms for efficient resource allocation schemes that exploit optimistic rates while ensuring that adequate resources are reserved in the cloud for the clients. Based on the prediction of demand for the request, proposed algorithms are designed by simulating drip irrigation approach in order to mitigate the wrong resource allocation decisions. The results of experimental evaluations show that the proposed algorithms significantly reduce the financial cost of resource allocations in the cloud as compared to other conventional schemes.

Keywords *cloud computing; resouce provisioning; resource reservation; scheduling; drip irrigation*

I. Introduction

The cloud computing environment consists of a collection of interconnected and virtualized computers that are dynamically provisioned as a unified computing resource. Cloud provisioning is the process of allocating computing resources to a customer by the cloud provider. Cloud computing services have been termed into three major categories, namely, Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) [1]. It allows the users to access large pool of resources from the remote

Dept. of Computer Science and Engineering, Sri Ramakrishna Engineering College, Coimbatore, Tamil Nadu, India. Email: madhumathi.r@srec.ac.in

Dept. of Electronics and Communication Engineering, Vidhya Mandir Institute of Technology, Tamil Nadu, India. Email: rlgs14466@rediffmail.com

Dept. of Computer Science and Engineering, Madanapalle Institute of Technology and Sciences, Andhra Pradesh, India. Email: drveningstonk@mits.ac.in

Assistant Manager, Jain Irrigation Systems, Tamil Nadu, India. Email: Sengs249@gmail.com

locations. The computing resources can be allocated dynamically upon the demands and preferences of the consumers.

Resource allocation is a process of assigning available resources to the needed cloud applications in order to achieve a high level of user satisfaction and resource utilization ratio thereby to improve the overall performance and resource utility of the system [2, 3, 4]. It is an essential part in data management applications such as virtual machine placement in data centers, network virtualization, and so on. Most of the existing systems work well for the homogenous environment in data or resource management system. However, it is difficult to extend them to dynamic environment where demand is highly variable. In this paper, we present novel schemes to resource allocation that allows resources to be allocated according to drip irrigation model successfully applied in meeting issues in current agricultural needs. A generic model has been proposed that assist cloud environments to solve their dynamic allocation problems.

The application of allocation schemes includes capacity planning for cloud services, VM placement in data centers, network virtualization, and data replication management. All these applications utilize significant resource allocation components to address the issues associated with it. In order to build a cloud computing environment, there are many open source platforms available which are Eucalyptus [5], Open-Nebula [6], Openstack [7], Nimbus [8], and so on. The resource scheduler of the virtual machine is typically used to manage the resource of each platform effectively.

Motivation of this paper is to use drip irrigation approach applied in agriculture. Drip irrigation is an application of artificial water to crops rather than rain water to overcome it is daily water requirement to full its growth stages. These assumptions are considered and modeled for addressing problems in resource scheduling in cloud environment.

The paper has been organized as follows; the brief background study is presented in section 2. Followed by, section 3 provides an outline of the drip irrigation system model which has been successfully applied in present day agriculture and proposes the resource allocation schemes which exploit the drip irrigation approach. Further, experimental setup and result evaluation are discussed in section 4 and subsequently section 5 concludes the paper with further extension.

II. Related Work

Several research efforts address dynamic resource allocation in cloud computing [9, 10, 11, 12]. Lee et al proposed a resource allocation model based on the performance of node [14]. The general model of resource allocation works as follows; upon accepting a request from a customer, resource provider must create the appropriate number of virtual machines (VMs) and allocate resources to meet their demand. The process is conducted in

ISBN: 978-0-9948937-1-0

several different ways: advance provisioning, dynamic provisioning and user self-provisioning. In this respect, the term resource provisioning means that providing resources. The approach proposed in [13] solves the problem of optimization in resource allocation. In this approach, incoming jobs are allocated to the virtual machines using Ant Colony Optimization (ACO) approach which minimizes the makespan by comparing with First Come First Served (FCFS) and Round-Robin (RR) scheduling algorithms.

The simple bin-packing scheme solves many issues in cloud resource allocation task. The idea of bin-packing allocation is to fit a set of balls into a given set of bins, while satisfying constraints defined upon the different properties of the balls and bins i.e. demand and availability. The approach proposed by Buyya et al in [15] addresses the several challenges involved in Service Level Agreement (SLA)-oriented resource allocation in order to differentiate and satisfy service requests based on the desired utility of users.

Ergu et al consider the resource allocation mechanism based on task nature [16]. That is the nature of users request also considers a major role in cloud resource allocation. An approach presented in [17] allocates resources by considering the importance of tasks which is notified by Saaty Rating Scale. The analytical hierarchy process is used to obtain the information about the resources which can be mapped to tasks based on task rank. Chen et al constructed the logical view of resources from the physical view [18]. Logical resources are formed by considering many physical resources like resource virtualization. It ensures high utilization rate, less waiting time for users and high meeting rate. Xiao et al proposed a model for minimizing the problem of skewness and unevenness in resource utilization [19]. The model achieves the overload avoidance through two parameters namely cold spot and hot spot. Less number of machines are used to handle more number of incoming requests thereby supports green computing at some extent. The high throughput and minimal waiting time have also been considered in few algorithms.

III. Proposed resource allocation schemes

A. Drip Irrigation Approach for Agriculture

"'Drip irrigation is an irrigation method that saves water and fertilizer by allowing water to drip slowly to the roots of plants, either onto the soil surface or directly onto the root zone, through a network of valves, pipes, and emitters". It is typically done through narrow tubes that deliver water directly to the base of the plant [23]. A drip-irrigation system can be a production asset for a small and large farm. Using drip irrigation approach for cultivation of vegetable, farmer could be benefited enormously in terms of money saving, water saving, and less manpower requirement for watering plants [20]. With this motivation, this paper proposes a few models using drip irrigation approach. The following are the various approaches presented in this paper which exploits the drip irrigation model so as to meet the emerging issues in cloud resource allocation task.

ISBN: 978-0-9948937-1-0

1) Sequential drip based resource allocation for demand on memory (SDRAmem)
2) Sequential drip based resource allocation for demand on CPU load (SDRAcpu)
3) Prioritized drip based resource allocation for demand on memory (PDRAmem)
4) Prioritized drip based resource allocation for demand on CPU load (PDRAcpu)
5) Sequential hybrid drip based resource allocation (considering both memory and CPU load) (SHDRA)
6) Prioritized hybrid drip based resource allocation (considering both memory and CPU) (PHDRA)

Procedure Drip_Irrigation_Approach [22]

1. *Measure or get the details about the area of land to be irrigated.*
2. *Consider different land parameters.*
3. *Rank the fields in the targeted area which will be the sequence to be followed for watering.*
4. *Measure the quantity of water available at present and to be added in due course*
5. *Irrigate the respective fields in the sequence specified.*
6. *Reassess the details regarding water availability, land parameters.*
7. *Repeat the above steps till the agricultural field is completely irrigated.*

B. Drip irrigation enabled Resource Allocation schemes

- *Problem Definition*

The requirement of resources for completing a task may be different to different task. This paper classifies the utility of the task to be either low or high. In this respect, algorithms have been devised.

- *Theorem 1*

If the utility of the task is low, prove that the drip based approach is suitable for cloud resource allocation. If the resource availability at the resource provider is low, then drip based approach is enabled to allocate (or) use resource at the maximum utility. Here, drip irrigation approach has been utilized to solve the problem at some extent.

If the resource availability is high, no need for enabling drip irrigation approach for resource allocation because enabling drip system may be an expensive task. If this is being

ISBN: 978-0-9948937-1-0

the case, allocate the requested resource directly and assign all the available resource to the needy tasks.

if the *utility requirement of task* is *low*
if the *resource availability* is *low*
Enable *drip based approach* for resource allocation
Else Allocate resources directly without influencing drip based approach

Thus it is inferred that the utility requirement of the tasks is high, and then the drip enabled approach may not be suitable for resource allocation.

- *Theorem 2*

If the utility of the task is high, prove that the drip based approach is suitable for cloud resource allocation.

if the *utility requirement of task* is *high*
if the *demand on memory* is *high*
Enable *drip based approach* for resource allocation by assigning 50% of resources to given request (task)
Set the *current demand* to be *current demand*-50% of *existing demand*
(i.e. % of request further needs to be satisfied)
Set the *current availability* to be *current availability*-50% of *existing demand*
(i.e. % of resources allocated to the request)
if the *demand on memory* is *medium*
Enable *drip based approach* for resource allocation by assigning 25% of resources to given request (task)
Set the *current demand* to be *current demand*-25% of *existing demand*
Set the *current availability* to be *current availability*-25% of *existing demand*
else (when the *demand on memory* is *low*)
Enable *drip based approach* for resource allocation by assigning 10% of resources to given request (task)
Set the *current demand* to be *current demand*-10% of *existing demand*
Set the *current availability* to be *current availability*-10% of *existing demand*

- *Algorithm Overview*

The following algorithm steps are the essential in implementing six variants of approaches presented in the paper.

1) Access the available resource at different VMs say,{VM_1,VM_2…….VM_n}. The computing resources are treated to CPU load and memory.

2) Measure the demand of requirement of tasks (*K*)
3) *for each* tasks, Measure the expected utility of tasks in terms of time *T*

Procedure Drip_Enabled_Resource_Scheduling_Approach

1. *Measure or get the details about the request for task to be completed.*
2. *Consider different task parameters such as expected time to be completed, demand of computing resources, and the level of demand (high, medium, low).*
3. *Prioritize the task according to the utility of the task thereby to ensure the sequence to be following for fair resource allocation.*
4. *Measure the quantity of resource available at present and to be added in due course*
5. *Allocate resources to the respective task in the sequence specified.*
6. *Reassess the details regarding availability of resources and pending demand to be fulfilled.*
7. *Repeat the above steps till the entire task schedule is completed and allocated with needed resources.*

- *Level of Demand*

The level of demand may be determined by the utility of the task. Typically when the utility is low, irrespective of level of demand on memory (low, medium, high), assign the resources to tasks. If the task is completed, remove the task from the queue.

IV. Results and Discussion

A. Dataset preparation

Since the unavailability of real dataset for evaluating the proposed schemes, the model dataset has been created with the variants of attributes of demand on computing resources such as memory requirement and CPU load requirement for experimental evaluation.

- Higher utility, lesser demand on memory
- Lower utility, lesser demand on memory
- Higher utility, higher demand on memory
- Lower utility, higher demand on memory
- Higher utility, lesser demand on CPU
- Lower utility, lesser demand on CPU
- Higher utility, higher demand on CPU
- Lower utility, lesser demand on CPU

Incorporating combination of utility requirement, memory demand and CPU demand, further dataset is enabled with entries of the following.

- High utility, less demand on memory, less CPU load

ISBN: 978-0-9948937-1-0

- Less utility, less demand on memory, less CPU load
- High utility, high demand on memory, less CPU load
- Less utility, high demand on memory, less CPU load
- High utility, high demand on memory, high CPU load
- Less utility, high demand on memory, high CPU load
- Less utility, less demand on memory, high CPU load
- High utility, less demand on memory, high CPU load

B. Experimental Evaluation

For testing the proposed algorithms, it is essential that the availability of computing resources should be greater at (all the) time (T_i) for {i=1……n}.

The following are the assumptions made for experimentation. It is required to keep track of the following variable at every time T_i. This is the core of the proposed drip enabled resource allocation schemes.

- Number of request
- Order in which resource requests are coming
- Availability of resources at T_i
- Number of resources allotted to tasks (may be 10%, 20%,...)
- In order to maintain *fairness* in allocating resources, all the requests are treated uniformly important.

Upon processing the resource requests K, measure the availability f resources at time (T_i) for both traditional and proposed drip based approaches. Following are the evaluation metrics used to test the effectiveness of the proposed resource allocation schemes.

1. ***Number of users utilized the available resources*** for the same available resources.
2. For particular time period, ***% of resources used***, without making the client in waiting state.
3. ***Makespan:*** The makespan is the total length of the schedule i.e. time spent in completing all the jobs to be processed. The schedule or task may comprise more than one job request.

C. Experimental Results

Based on the test bed designed above for evaluation, different metrics have been evaluated and results have been shown. The proposed approaches have been simulated using CloudSim toolkit [21] which is an environment for simulating the cloud computing applications. The proposed algorithms namely, SDRAmem, SDRAcpu, PDRAmem, PDRAcpu, SHDRA, and PHDRA are compared with the First Come First Served and Round Robin algorithm, by simulating them using the same toolkit. In order to simulate the

experimental environment, one datacenter with various host nodes and virtual machines of different specifications as given in [14].

Table1: HOST MACHINES specification

VM ID	Core	CPU	Memory
1	1	500	512
2	1	700	1024
3	1	300	512
4	1	500	1024
5	1	700	512

Table 2: Virtual Machines request specification

VM ID	Core	CPU	Memory
1	1	500	512
2	1	700	1024
3	1	300	512
4	1	500	1024
5	1	700	512
6	1	300	1024
7	1	500	512
8	1	700	1024
9	1	300	512
10	1	500	1024
11	1	700	512
12	1	300	1024
13	1	500	512
14	1	700	1024
15	1	300	512
16	1	500	1024
17	1	700	512
18	1	300	1024
19	1	500	512
20	1	700	1024

The specification of the host nodes and virtual machine requests are depicted in Tables 1 and 2. Figure 1 depicts the availability of memory on each host after all the virtual machines have been scheduled to proper hosts, which shows the memory utilization effectiveness in comparison with FCFS and RR algorithms. The mapping between host machine and virtual machine request is assessed for fair allocation of resources.

ISBN: 978-0-9948937-1-0

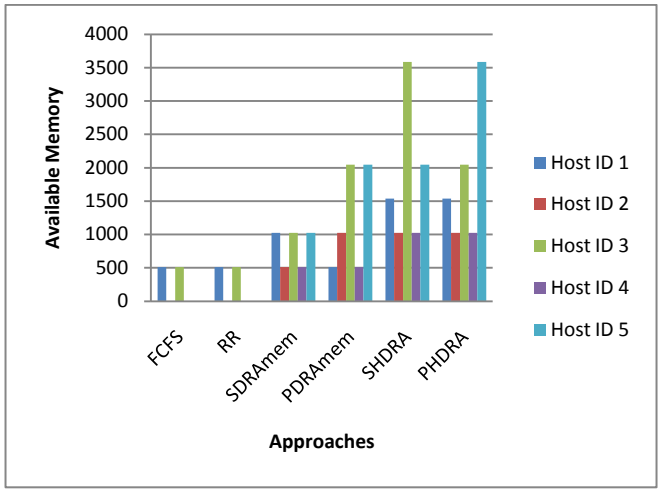

Figure 1: Availability of Memory

Figure 2 depicts the availability of CPU on each host after all the virtual machines have been scheduled to proper hosts, which shows the CPU utilization effectiveness in comparison with FCFS and RR algorithms.

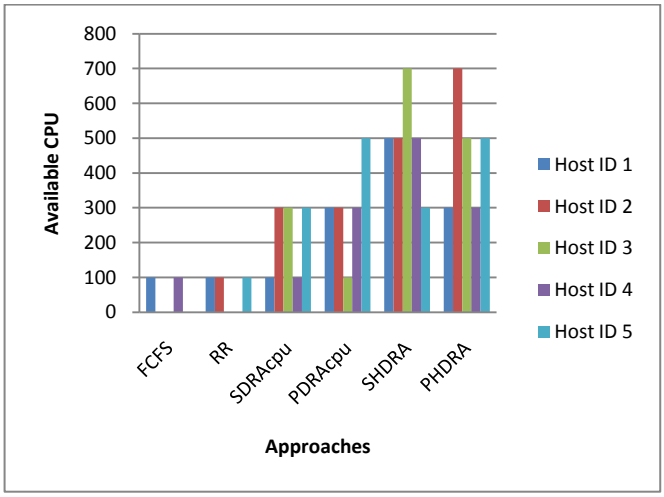

Figure 2: Availability of CPU

It is inferred that the experiment results shows that the proposed drip based resource allocation schemes can improve the resource utilization rate so as to serve a greater number of resource requests at a time without consuming the allocation time.

V. Conclusion

This paper addressed the problem of dynamic resource provisioning VM instances in cloud environment in order to generate high profit besides determining the VM allocation with various drip based resource allocation schemes namely, SDRAmem, SDRAcpu, PDRAmem, PDRAcpu, SHDRA, and PHDRA. An extensive simulation experiments have been performed in order to evaluate these approaches. The results show that hybrid version of proposed schemes can effectively capture the demand and provision the computing resources to meet the resource demand. The proposed approaches have been compared against FCFS and RR scheduling algorithm in terms of utilization rate. It is summarized that a highly effective VM instance provisioning and allocation system could be designed by taking other dynamic features in resource allocation scenario in cloud into consideration.

Acknowledgment

Authors wish to extend thanks to their institutions for research support and the anonymous reviewers for their helpful comments which improved the presentation of this paper.

References

1. Arfeen M., Pawlikowski K., Willig A., "A Framework for Resource Allocation Strategies in Cloud Computing Environment," Proceedings of 35th Annual IEEE Conference on Computer Software and Applications, pp. 261-266, 2011.
2. Gao K., Wang Q., Xi L., " Reduct algorithm based Execution Times prediction in Knowledge Discovery Cloud Computing Environment," The International Arab Journal of Information Technology (IAJIT), vol. 11, no. 3, 2014.
3. Li C., Li L., "Efficient resource allocation for optimizing objectives of cloud users, IaaS provider and SaaS provider in cloud environment," Journal of Super Computing, vol. 65, pp. 866–885, 2013.
4. Warneke D., Kao O., "Exploiting Dynamic Resource Allocation for Efficient Parallel Data Processing in the Cloud," IEEE Transactions on Parallel and Distributed Systems, vol. 22, no. 6, pp. 985-997, 2011.
5. The Eucalyptus. http://www.eucalyptus.com/eucalyptus-cloud.
6. OpenNebula. http://opennebula.org/about:about.
7. OpenStack. http://www.openstack.org.
8. Nimbus. http://www.nimbusproject.org.
9. Chaisiri S., Lee B., Niyato D., "Optimization of resource provisioning cost in cloud," IEEE Transactions on Services Computing, vol. 5, no. 2, pp.164-177, 2012.
10. Leivadeas A., Papagianni C., Papavassiliou S., "Efficient Resource Mapping Framework over Networked Clouds via Iterated Local Search-Based Request Partitioning," IEEE Transactions on Parallel and Distributed Systems, vol. 24, no. 6, 2013.

11. Hsiao H., Chung H., Shen H., Chao Y., "Load Rebalancing for Distributed File Systems in Cloud," IEEE Transaction on Parallel and Distributed Systems, vol. 24, no. 5, pp. 951 – 962, 2013.
12. Uhligetal R., "Intel virtualization technology," IEEE Transactions on Computers, vol. 8, no. 5, pp. 48 - 56, 2005.
13. Tawfeek M, EI-Sisi A, Keshk A, Torkey F, "Cloud Task Scheduling based on Ant Colony Optimization," The International Arab Journal of Information Technology (IAJIT), Accepted February 2014.
14. Lee H., Jeong Y., Jang H., "Performance analysis based resource allocation for green cloud computing," Journal of Super Computing, 2013.
15. Rajkumar Buyya, Saurabh Kumar Garg and Rodrigo N. Calheiros, SLA-Oriented Resource Provisioning for Cloud Computing: Challenges, Architecture, and Solutions, International Conference on Cloud and Service Computing, 2011.
16. Ergu D., Kou G., Peng Y., Shi Y., Shi Y., "The analytic hierarchy process: task scheduling and resource allocation in cloud computing environment," Journal of Super Computing, vol. 64, pp. 835–848, 2013.
17. Saaty TL., "How to make a decision: the analytic hierarchy process," European Journal Operation Research, vol. 48, no. 1, pp. 9-26, 1990.
18. Chen X., Zhang J., Li J., Li X., "Resource virtualization methodology for on-demand allocation in cloud computing systems," Service Oriented Computing and Applications (SOCA), vol. 7, pp. 77–100, 2013.
19. Xiao Z., Song W., Chen Q., "Dynamic Resource Allocation Using Virtual Machines for Cloud Computing Environment," IEEE Transactions on Parallel and Distributed systems, vol. 24, no. 6, pp. 1107-1117, 2013.
20. Eric Simonne, Robert Hochmuth, Jacque Breman, William Lamont, Danielle Treadwell, and Aparna Gazula, Drip-irrigation systems for small conventional vegetable farms and organic vegetable farms, University of Florida,
21. Calheiros RN, Ranjan R, Beloglazov A, De Rose CAF, Buyya R, CloudSim: a ToolKit for modeling and simulation of cloud computing environment and evaluation of resource provisioning algorithm, 2011.
22. R. Madhumathi and R. Radhakrishnan, Resource Allocation using Complexity Equalization Technique in Cloud Environment, International Journal of Applied Engineering Research, Volume 9, Number 24 pp. 25573-25585, 2014.
23. Drip Irrigation Systems Definition. https://en.wikipedia.org/wiki/Drip_irrigation