International Journal of Computer Applications (0975 – 8887)
Volume 152 – No.2, October 2016

IAR: Improved Advance Reservation in IaaS Clouds

Vivek Shrivastava
International Institute of Professional Studies
Devi Ahilya University
Indore, India

Payal Gupta
John Hopkins University
3101 Wyman Park Dr.,
Baltimore, MD

D. S. Bhilare
Computer Centre
Devi Ahilya University
Indore, India

ABSTRACT

Cloud data centers have a large number of resources. Management of such huge amount of resources for a large number of consumers requires fail-safe algorithms and leasing policies. Advance Reservation (AR) leasing policy is a rigid policy, which needs resource and consumer locking at a very early point of time, while advanced reserved lease can be rejected at actual point of time when resources are required. This problem can be dealt with proposed Improved Advance Reservation (IAR) algorithm and leasing policy , which uses negotiation and provide half capacity of the requested number of resources, instead of rejecting a lease if consumer agrees for the same. Experimental results show that the proposed work maximize resource utilization and acceptance of requests in comparison with existing algorithms in Haizea.

General Terms

Cloud computing, Scheduling, IaaS Cloud, Algorithms

Keywords

IAR, Leasing Policies, Resource Management, IaaS Cloud

1. INTRODUCTION

Cloud computing is gaining popularity as it provides, on demand services over the Internet in a very less amount of time and at a very low cost. Users are not required to purchase and install software or hardware in cloud computing model. This supports minimum capital expenditures and operational expenditures as users are free from cooling cost, maintaining state-of-the-art software and hardware, and space required to put required hardware and software.

Clouds computing let users free to do their intended job because, users are not required to hire and maintain a big IT department for their computational requirements. Cloud supports three service models with the help of four deployment models. Three service models namely Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS) can be provided to users as single individual service or as a package of combination of more than one service models.

Deployment models of cloud provide it liberty to provide services as a private service to public services at large. Private deployment model of cloud confines services to only one user or organization and that may be on-premise or off-premise. Public deployment model of cloud provides better utilization of resources at cloud service provider side by using same resources for more than one users or organization so this model provides multi-tenancy. Hybrid deployment model is a combination of private and public deployment model, while community deployment model allows sharing cloud services among a group of users or organizations, which share same concerns.

2. RESOURCE ALLOCATION IN CLOUD

Cloud exhibits a pool of resources, sometimes infinite, put together to serve a large number of users. Managing these resources requires: registering new resources, allocating and reallocating resources, monitoring and securing resources, providing resources to consumers in a fail-safe and a zero downtime manner. Resource management also requires load balancing and load sharing so that resource utilization can be optimized, throughput can be maximized and response time can be minimized.

Computing resources can be allocated to users in the form of virtual machines (VMs) in cloud computing. These VMs can be provided by executing different kinds of leases. These leases works under a service level agreement (SLA). Leases have a nature based on their leasing policy and resource management algorithms.

Most of the cloud providers work majorly on Best Effort (BE) leases, immediate leases and a very few on AR leases [1]. Service providers like Amazon EC2 [2] provides a pay per use model to general public for providing computing resources on public cloud, Google Cloud platform [3] offers hosting of infrastructure to its end users and provides a set of modular cloud-based services, Microsoft Azure [4] provides platform and infrastructure for designing, implementing and managing applications and services through a global network on cloud. Eucalyptus [5], Nimbus [6] and OpenNebula [1] are cloud toolkits which are helpful in setting up a cloud on local infrastructure.

Sometimes it is not possible for the cloud providers to satisfy all the requirements of the user's request due to cases such as lack of resources (computing capacity and storage) which eventually leads to rejection of such request and increases the rejection rate of the system. Haizea is an open sources virtual machine based lease management architecture which tries to address such issues [7], [8], [9], [10]. It performs resource allocation and implements user's computational resource requests in the form of one or more virtual machines called as lease. This lease is accepted by Haizea if it can assure the resources allocation policy requested by the user. In order to assure the resource allocation it requires requested resources for a finite duration and the start time. It then reserves the resources for the particular lease in the specified time interval. Whenever the start time of the lease comes it allocates resources to the user in form of VMs, which are deployed on physical machines. The requested lease is stored using two main data structure called Lease and SlotTable. Lease stores the information regarding an accommodated lease and SlotTable will store information about the physical nodes and the reservation made on it. The submission of a lease causes the scheduler adding entries to the slot table. The start and the end of an allocation in the slot table are also treated as an event that causes the scheduler to re-evaluate the schedule.

Haizea supports four kinds of leases: BE, Immediate, AR and Deadline Sensitive (DLS) [11]. BE lease are preemptable lease and do not have a time constraint. These leases are splitable. Immediate and AR leases are non- preemptable and have a specific time constraint like start time and end time. A DLS lease is also a preemptable lease but with certain time constraint i.e. it must be executed before its deadline is reached. In order to execute an AR and an immediate lease, BE leases are pre-empted to provide resources to these leases. There is no guarantee when a BE lease will have enough resources for its execution. When the system is flooded with number of AR and immediate leases then BE lease have to wait for long to be executed. In order to avoid longer waiting period the user prefers to resubmit its lease in the form of an AR lease. This increase in number of AR leases will cause the system to reach a bottleneck condition where the system performance will degrade. In such situation, most AR lease will be rejected due to unavailability of demanded resources in a given time period. In order to handle this type of situations, an option to run the lease at half capacity can be associated with the AR leases. Those AR leases, which can be executed at full capacity will be executed as such like in the original system, but those leases which are rejected at full capacity are given an option to run at half capacity with the user's consent while keeping the other parameters same. Thus, it assures the user that their requests are executed even when the system is flooded with a lot of leases.

3. LITERATURE SURVEY

To minimize rejection of deadline sensitive resource scheduling in IaaS cloud computing by experimenting in Haizea, Dynamic planning based scheduling algorithm has been proposed in [12] which can admit new leases and prepare the schedule whenever a new lease can be accommodated.

Decision making algorithms and extending the current AR algorithms in IaaS cloud computing, a negotiation based allocation of resources has been proposed in [12]. If the system has lots of AR and immediate leases then priority will be given to these leases. BE leases have to wait for a long time and will be served when resources are free from AR & immediate leases.

To protect BE leases from starvation of resources which is the main problem when dealing with heterogeneous request environment, Starvation-removal algorithm was proposed in [13]. This algorithm controls increased rate of lease suspensions by increasing priority of leases by aging mechanism. BE leases take long period of time to complete due to presence of so many AR and immediate leases. This may lead to more requests to AR leases which in turn may cause internal fragmentation of free resources.

Proper load should be provided with virtual machines on a server, so that they can be protected from overloading. Measurement of computing power can be done by CBUD Micro [15] for very little computing power devices.

Resource request and acceptance rate also fall due to heavy request traffic for resources and slow response, and the completion time of requests for resources. These situations are handled by consumer rating index (CRI) as given in [16] and modified Earliest Deadline First algorithm (mEDF) as given in [17]. In [16] an algorithm and a leasing policy for prioritizing consumers on the basis of CRI score was provided. Authors showed policy to maintain order of execution of users' task on the basis of some parameters and claimed their result is an improvement over existing policies and algorithms under some conditions. A security aware leasing policy and resource management algorithm: SAFETY was proposed and implemented in [18], [19] for isolation of users' task in multitenant cloud.

4. NEED OF CONVERSION OF AR LEASE TO IAR LEASE

AR is the process of requesting resources to be used at any specified interval in future. Users submit a request i.e. a non preemptable lease by specifying a series of parameters such as number of resources, and time constraints such as start time and duration time. The lease manager checks for the feasibility of each request. If one or more parameter cannot be satisfied then the request is rejected. This type of lease offers rigidity and does not allow the user to change the parameters. The rigidity thus offered increases the rate of rejection for this type of lease. In order to overcome this problem, negotiation can help by decreasing the number of resources to half if the user agrees for the same. This will allow the AR to be accepted at half capacity thus eventually increase the acceptance rate. Moreover, a lease requesting more than the maximum number of hardware resources can also be accepted if they run at half capacity.

Table 1 Lease Requests Arrivals

Lease Number	Nodes	Arrival Time	Start Time	Duration Time
1	2	11:10	12:00	30
2	3	11:20	12:30	10
3	4	11:30	12:00	20
4	6	11:40	12:40	40
5	1	11:50	12:40	10

4.1 System Description

To request resources from Haizea, user has to submit lease in specific format. Main parameters must include virtual nodes, amount of physical resources for each node, start time and duration time. CPU and memory are two computational resources which are requested by user. It also has a software field, where a user can specify the software to be deployed on allocated resources. In this experiment, only those leases are considered in which CPU and memory are fixed for all nodes. It is assumed that the lease request number of virtual nodes with the same hardware for each node. This is done for simplicity of experiment. The proposed model can handle the leases demanding more than the available/ fixed (4) number of virtual nodes. The software specified in software field of the request is considered as VM image. These VM are stored in a central repository. If a lease is pre-empted, it is suspended by suspending its VM. These VMs may be resumed on the same nodes or on other nodes. When a VM is suspended, a memory stale file is created on the node where the VM was running. When a VM is resumed, the VM image is transferred to the original node on which it was running and the memory stale file is read into memory whereas when a suspended VM is migrated to a different node, its image gets transferred to the new node on which it will run. Suspending multiple VMs communicating with each other can cause the system to enter into a deadlock state. All these assumptions are made based on the assumptions made in combining batch execution.

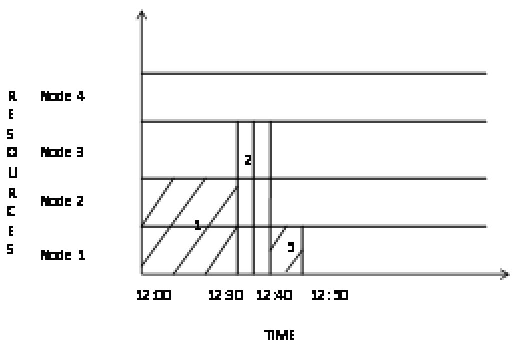

Figure 1 Resource Allocation in 4 nodes.

4.2 Task Characteristics

- As previously defined Haizea supports four type of lease (tasks) to schedule.
- All tasks are periodic.
- Deadline and BE tasks can be executed in parts.
- All tasks are independent and the scheduling algorithm assumes no communication between them.
- Haizea accepts new tasks if all the old tasks that are accepted can be feasibly scheduled in accordance with the new task.

5. THE HAIZEA VM SCHEDULER

This class is responsible for taking a lease and scheduling VMs to satisfy the requirements of that lease. This class has an algorithm named schedule exact in VM Scheduler. This algorithm schedules VMs/ leases that must start at an exact time i.e. AR lease. This type of lease is easy to schedule because exact start time is known, which means that's the only starting time that have to check for resources. This method is responsible to call the mapper. If no mapping is possible, it indicates that there are not sufficient resources for the requested lease. This raises an exception and rejects the lease. The proposed algorithm checks if mapping is possible or not. If the mapping is not possible it creates a dictionary with the same lease number but reduces the number of virtual nodes to half. This new AR thus formed is called IAR (Improved Advance Reservation). This new dictionary is again given to the mapper for mapping. Now if no mapping is possible again this lease gets rejected.

5.1.1 Algorithm 1.

```
athalf = 0
 # a new parameter that indicates
if the dictionary is modified.
if mapping == None
new_dict = dict(lease.numnodes/2)
athalf =1
#indicates that there is a change
in dictionary
#Marks that an AR has been
converted into a IAR.
mapping, actualend, preemptions =
self.mapper.map(lease,
requested_resources, start, end,
strictend = True)
#call the mapper function
if mapping == none
  raise exception;
```

This class has one more algorithm under the Slot table Event Handler named handle start vm. It is responsible for handling the start of a VM Resource Reservation. This is the part where Slot table event handler works. It marks the beginning of a resources reservation from a dictionary. In the proposed algorithm it checks if for a lease a new dictionary is formed then Resource Reservation (rr) requested resource is also updated.

5.1.2 Algorithm 2

```
if lease_state ==
Lease.STATE_READY:

l.set_state(Lease.STATE_ACTIVE)
            rr.state =
ResourceReservation.STATE_ACTIVE
            now_time =
get_clock().get_time()
           l.start.actual =
now_time
 if athalf  != 0

#indicating a IAR has been formed
and we need to update resource
reservation
       rr. requestedresources =
rr. requestedresources/2
            try:

self.resourcepool.start_vms(l, rr)
            except EnactmentError,
exc:

self.logger.error("Enactment error
when starting VMs.")
                # Right now, this
is a non-recoverable error, so we
just
                # propagate it
upwards to the lease scheduler
                # In the future,
it may be possible to react to
these
                # kind of errors.
                raise
```

5.1.3 The Haizea Action

Haizea has one more algorithm in Actions named from rr which overrides the function EnactmentAction. This algorithm performs the enactment operation. In the proposed algorithm if there is a difference found between the original lease submission dictionary and new dictionary for that particular lease, the enactment action is done by considering the IAR.

6. EXPERIMENT AND RESULTS

To evaluate both the algorithms following parameters have been used-

- System Throughput
- Number of leases accepted

- Number of leases rejected
- Number of leases running at half capacity

6.1 Experimental Setup

This proposed algorithm is implemented in Python and simulated on Haizea by modifying its VM Scheduler and Enactment Component. Haizea has been used in simulated mode. For experiment purpose, four physical nodes each having one CPU and 1024 MB memory are considered as the total available resources. The number of leases is varied and readings are noted for the parameters above. An AR lease has basically three important parameters, namely start time, duration and number of nodes. All the parameters are varied randomly and manually.

6.2 Experiment 1: accepted lease, rejected lease

The experiment is performed by considering leases in four sets of 5,10,15,20 and 25 leases. A comparison between the number of leases accepted and the number of leases rejected is drawn in both existing and proposed algorithms.

Table 2 Leases accepted and rejected in set of 5 leases

Set of 5 leases	Algorithm	Accepted	Rejected
Set 1	Existing	3	2
	Proposed	4	1
Set 2	Existing	4	1
	Proposed	4	1
Set 3	Existing	3	2
	Proposed	4	1
Set 4	Existing	3	2
	Proposed	4	1
Average	Existing	3.25	1.75
	Proposed	4	1

Table 3 Leases accepted and rejected in set of 10 leases

Set of 10 leases	Algorithm	Accepted	Rejected
Set 1	Existing	5	5
	Proposed	7	3
Set 2	Existing	5	5
	Proposed	7	3
Set 3	Existing	5	5
	Proposed	6	4
Set 4	Existing	5	5
	Proposed	6	4
Average	Existing	5	5
	Proposed	6.5	3.5

Table 4 Leases accepted and rejected in set of 15 leases

Set of 15 leases	Algorithm	Accepted	Rejected
Set 1	Existing	6	9
	Proposed	10	5
Set 2	Existing	8	7
	Proposed	10	5
Set 3	Existing	8	7
	Proposed	11	4
Set 4	Existing	9	6
	Proposed	10	5
Average	Existing	7.75	7.25
	Proposed	10.25	4.75

Table 5 Leases accepted and rejected in set of 20 leases

Set of 20 leases	Algorithm	Accepted	Rejected
Set 1	Existing	8	12
	Proposed	13	7
Set 2	Existing	10	10
	Proposed	12	8
Set 3	Existing	11	9
	Proposed	15	5
Set 4	Existing	10	10
	Proposed	12	8
Average	Existing	9.75	10.25
	Proposed	13	7

Table 6 Leases accepted and rejected in set of 25 leases

Set of 25 leases	Algorithm	Accepted	Rejected
Set 1	Existing	11	14
	Proposed	17	8
Set 2	Existing	13	12
	Proposed	16	9
Set 3	Existing	12	13
	Proposed	16	9
Set 4	Existing	13	12
	Proposed	15	10
Average	Existing	12.25	12.75
	Proposed	16	9

6.2.1 *Results of Experiment 1*

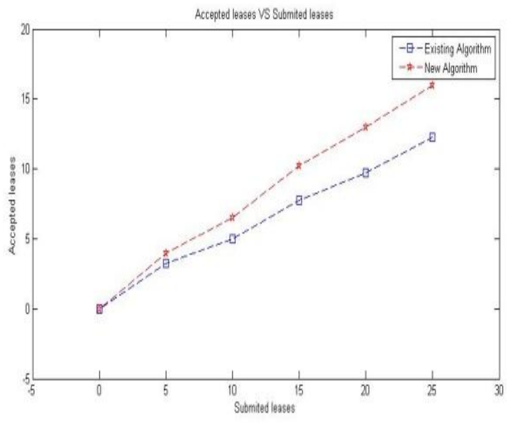

Figure 2 Accepted Leases VS Submitted Leases

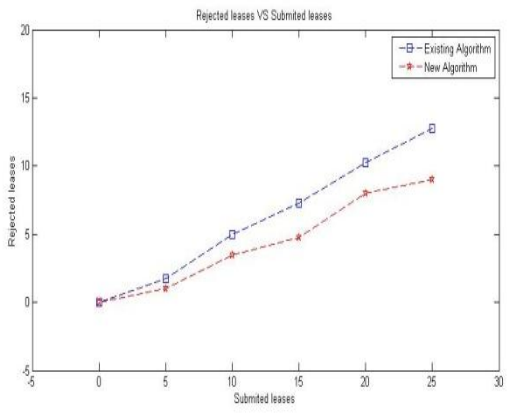

Figure 3 Rejected Leases VS Submitted Leases

6.2.2 Experiment 2: number of leases at half capacity

This experiment is performed for four sets of 5, 10, 15, 20, 25 leases. This experiment shows the number of leases that are accepted at half capacity.

Table 7 Leases that are running at half capacity

No. of AR leases	Leases that are running at half capacity				
	Set 1	Set 2	Set 3	Set 4	Average
5	2	3	2	3	2.5
10	6	4	4	6	5
15	8	6	6	8	7
20	12	8	8	12	10
25	14	12	12	14	13

Table 8.

6.2.3 Results of Experiment 2

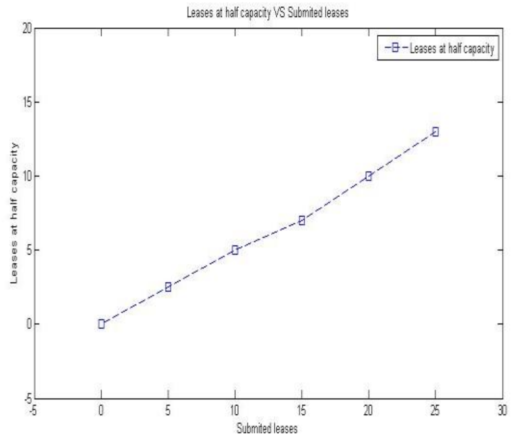

Figure 4 Leases at half capacity VS Submitted Leases

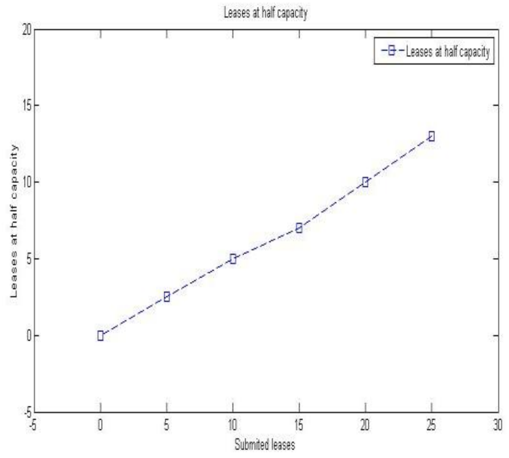

Figure 5 Leases at Half Capacity

7. CONCLUSION AND FUTURE WORK

The complexity of this procedure is dependent on start time, duration time, end time, number of requested resources, and number of leases running at that time. By running the lease along with a previous running lease on the remaining resources increases the acceptance rate by a second order polynomial function.

In this work number of resources has been divided to half in order to execute the lease. In future the division of resources can be done on dynamic or user selectable basis.

8. REFERENCES

[1] Borja, S., Ruben, M.S. and Ignacio, M.T., 2009. An Open Source Solution for Virtual Infrastructure Management in Private and Hybrid Clouds. IEEE Internet Computing.

[2] Amazon EC2, http://aws.amazon.com/ec2/ [accessed Sep. 8, 2014].

[3] Google Cloud Platform, https://cloud.google.com/ [accessed Oct. 12, 2014].

[4] Microsoft Azure , http://azure.microsoft.com/en-in / [accessed Aug. 18, 2014].

[5] Nurmi, D. Wolski, R. Grzegorczyk, C. Obertelli, G. Soman, S. Youseff, L. and Zagorodnov, D. 2009. The Eucalyptus Open-source Cloud-Computing System. In Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid.

[6] Nimbus, http://www.nimbusproject.org/ [accessed Sep. 8, 2014].

[7] Sotomayor, B. Keahey, K. and Foster, I. 2006. Overhead Matters: A Model for Virtual Resource Management. In Proceedings of the 2nd International Workshop on Virtualization Technology in Distributed Computing, IEEE Computer Society.

[8] Sotomayor, B. Montero, R. Llorente, I. and Foster, I. 2009. Resource leasing and the art of suspending virtual machines. In Proceedings of the IEEE International Conference on HPCC-09.

[9] Sotomayor, B. Montero, R. Llorente, and I. Foster, I. 2008. Capacity leasing in cloud systems using the OpenNebula engine. In Proceedings of the Workshop on Cloud Computing and Applications.

[10] Sotomayor, B. Keahey, K. and Foster, I. 2008. Combining Batch Execution and Leasing Using Virtual Machines. In Proceedings of the 17th International Symposium on High performance distributed computing (HPDC '08). ACM.

[11] Nathani, A. Chaudhary, S. and Somani, G. Policy based resource allocation in IaaS cloud. Future Generation Computer Systems, 28(1), (Jan. 2012), 94-103.

[12] Akhani, J. Chuadhary, S. and Somani, G. 2011. Negotiation for resource allocation in IaaS cloud. In Proceedings of the Fourth Annual ACM Bangalore Conference.

[13] Shrivastava, V. and Bhilare, D.S. Algorithms to Improve Resource Utilization and Request Acceptance Rate in IaaS Cloud Scheduling. International Journal of Advanced Networking & Applications, 3(5), (Nov. 2012), 1367-1374.

[14] Shrivastava, V. Bhilare, D. S. 2014. COMMA: A Cost Oriented Market and Migration Aware Leasing Policy and Algorithm in IaaS Clouds. In Proceedings of the 2014 International Conference on Information and Communication Technology for Competitive Strategies (ICTCS '14). ACM.

[15] Shrivastava, V. and Bhilare, D.S. CBUD Micro: A Micro Benchmark for Performance Measurement and Resource Management in IaaS Clouds. International Journal of Emerging Technology and Advanced Engineering 3(11), (Nov. 2013), 433-437.

[16] Shrivastava, V. and Bhilare, D.S. CRI: A Novel Rating Based Leasing Policy and Algorithm for Efficient Resource Management in IaaS Clouds. International Journal of Computer Science and Information Technologies, 3(2014), (Jun. 2014), 4226- 4230.

[17] Shrivastava, V. and Bhilare, D.S. mEDF: Deadline Driven Algorithm for Minimizing Response Time and Completion Time in IaaS Clouds. International Journal of Application or Innovation in Engineering and Management (Jun. 2014) 3(6), 16-22.

[18] Shrivastava, V. and Bhilare, D.S. 2015. SAFETY: A Framework for Secure IaaS Clouds. International Journal of Advanced Networking and Applications. (May 2015), 6(6), 2549-2555.

[19] Shrivastava, V. Bhilare, D.S. 2015. A Security Aware Leasing Policy and Algorithm for IaaS Clouds. International Journal of Engineering Research and Technology (Jun. 2015) 4(6), 886-891.