An Science Gateway with Cost Adaptive Resource Management Schemes

Woojoong Kim(✉), Seung-Hwan Kim, and Chan-Hyun Youn

Department of Electrical Engineering, KAIST, Daejeon, Korea
{w.j.kim, s.h_kim, chyoun}@kaist.ac.kr

Abstract. In order to process heavy data and computation applications such as scientific application in cloud computing environment, it is important to do an efficient resource schedule that decrease the resource usage cost while guaranteeing users' Service Level Agreement. To resolve this issue, we propose and implement Science Gateway, which execute the scientific application efficiently on heterogeneous cloud service providers instead of users. Especially, we propose a cost-adaptive resource management schemes (i.e. VM pool management scheme that decreases the resource management cost significantly based on the long-term payment plans of cloud resource providers and VM placement management scheme that guarantee the performance of communication between VMs). Finally, we demonstrate that our proposed system improves the performance of existing cloud systems through some experimental and simulation results.

Keywords: Cloud computing · Scientific workflow application · Cloud broker

1 Introduction

Heavy data and computation applications usually require the huge amount of computing and storage resource. With the emergence of cloud computing, users can utilize the resource as a cloud service based on pay-per-usage base to execute their own heavy applications [1]. However, it is still difficult for users to decide how much cloud service should be leased in order to minimize the amount of cloud resource while guaranteeing the certain performance of the application. In addition, there are a lot of cloud services from various cloud providers and they have various policies on cloud service and service charge. In this case, users can make non-optimal decision with the limited information on cloud services and waste the execution time and cost because of the inefficient decision. Thus, third party entity (i.e. cloud broker) presented between user and Cloud Service Provider (CSP) is defined to make an optimal decision with rich information on cloud services instead of users.

The key issue of the cloud broker is to decrease the resource usage cost from cloud environment in order to increase the profit while guaranteeing the certain performance of cloud service required by user. In this paper, to resolve this problem, we propose and implement the cloud broker called Science Gateway with the cost adaptive resource schemes including VM pool management scheme and VM placement management scheme.

Y. Zhang et al. (Eds.): CloudComp 2015, LNICST 167, pp. 136–144, 2016.
DOI: 10.1007/978-3-319-38904-2_14

The VM pool management scheme utilizes the reserved VM payment policy(RVM) which reserve some virtual machines(VM) on long-term in a discount price. Organizing a RVM pool by estimating the amount of resource demand on the future with this scheme, the cloud broker decreases the resource usage cost by providing the available RVM in the RVM pool instead of On-demand VM(OVM).

The VM placement management scheme decides a certain physical node within the cloud server structure to lease new VM instance in order to guarantee the network capacity between leased VMs and decrease the data transmission delay between VMs.

2 A Model Description of the Science Gateway

The object of science gateway is to minimize a resource usage cost while satisfying user's SLA for cloud resource providers instead of users. In our model, the science gateway concentrates on scientific applications and users only request for their application execution with their SLA to the science gateway.

The scientific application can be represented as a workflow $A(T, E, D)$ where A is the set of tasks, E is the set of edges and D is deadline. We assume that the total execution time of each tasks in A are known. By the science gateway, each task is allocated to their proper VM instance and processed in order of their starting time decided by SLA constraints such as deadline D.

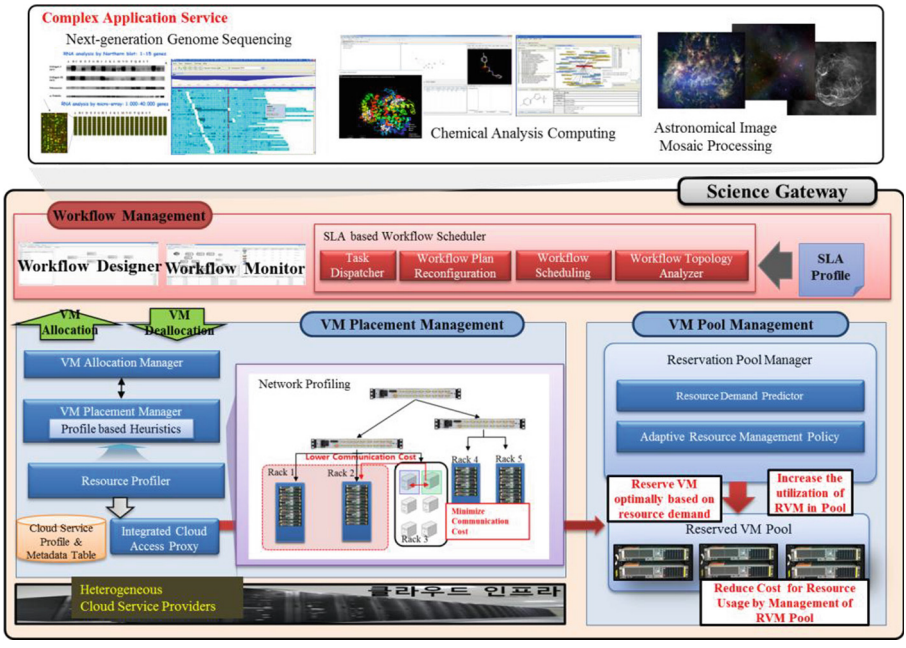

Fig. 1. Architecture of the proposed science gateway

Different types of VM instances have the different computing performance. We assume that CSP provides only three types of cloud resource service: small, medium, large. Each type of VM is charged in proportion to their capacity with Billing Time Unit (BTU). BTU is the base time unit to charge for resource usage time and partial-BTU resource usage time is rounded up to one BTU. A VM type is represented as $VT_i = \{VT_{c_i}, VT_{m_i}, VT_{s_i}\}, i \in \{1, 2, \ldots, K\}$: *number of VCPU (#)* VT_{c_i}, *memory size (GBs)* VT_{m_i}, *storage space (GBs)* VT_{s_i}. CSP provides OVM and RVM payment plans [2]. By RVM plan, VM instance can be leased for long BTU (e.g., monthly or yearly) with the discount price.

The science gateway is described in Fig. 1. It receives the scientific application request from users through GUI based workflow designer. Workflow management schedules the resource plan for each task within the requested application and processes all tasks on the VM instances provided by VM Placement Management Scheme and VM Pool Management Scheme.

3 Cost Adaptive Cloud Resource Management Schemes

VM Pool Management Scheme is shown in Algorithm 1. This scheme decides the appropriate amount of RVMs to be leased in the heuristic way in order to decrease the resource usage cost. This scheme works in the period of time interval T. The amount of RVMs is dependent on the resource demand. The historical data on all the executed tasks and their allocated VM types during the previous time interval T' is used for this scheme. We assume that the request trend in the current time interval T will be similar with the one in T'. As a result, we can decide the appropriate amount of i-type RVM, N_i for current time interval T. First, we organize clusters for each task within A based on their assigned VM instance type VT_i. Eventually, all the tasks within A are classified into some clusters Cl_{VT_i}. On each cluster Cl_{VT_i}, we organize groups g_m. It is a batch of non-overlapped tasks. After the tasks of Cl_{VT_i} are sorted in order of their starting time, each task of Cl_{VT_i} is picked into g_m in sequence if its start time *st* is later than the finish time *ft* of last task in g_m. This procedure is repeated until there is no available task in Cl_{VT_i} anymore. Finally, with $gct(g_m)$ total execution time of g_m and $VT(g_m)$ allocated VM instance type of g_m, we derive a RVM lease duration $RVM_{VT(g_m),gct(g_m)}$ by finding the size of BTU closest to the $gct(g_m)$.

We determine whether to lease $RVM_{VT(g_m),gct(g_m)}$ from CSP or not with the following condition.

$$\frac{C\left(RVM_{VT(g_m),gct(g_m)}\right)}{\sum_{\forall t \in g_m} et(t) \cdot C\left(OVM_{VT(t)}\right)} < 1 \tag{1}$$

The denominator of Eq. (1) is the total resource usage cost on OVMs for the tasks having the execution time *et* in g_m. The numerator of Eq. (1) is the total resource usage cost of RVM for g_m. If Eq. (1) is satisfied, it means that RVM is more efficient on cost than OVMs for g_m. As the value of Eq. (1) is decreased, leasing RVM is more efficient.

Algorithm 1. VM Pool Management Scheme

Input: *historical data including* $A = \{\forall t_{i,j}\}$ *and* $\forall VT(t_{i,j})$ *during previous time interval* T'

Output: N_i *during current time interval* T

```
For VT_i, ∀ i ∈ {1,2, … , K}.
   For ∀t ∈ S
      Cl_VT_i = Cl_VT_i ∪← t ∈ S if VT(t) = VT_i
   End for
End for
m = 0
For Cl_VT_i ∀ i ∈ {1,2, … , K}.
  sort tasks in Cl_VT_i in order of their starting time
   While available tasks exists in Cl_VT_i do
      For ∀t' ∈ Cl_VT_i
         If st(t') ≥ ft(t''), t'' = last task in g_m
            g_m = g_m ∪ t'
         End If
      End For
   remove tasks in g_m from Cl_VT_i
   m = m + 1
   End while
End for
For ∀g_m
   gct(g_m) = ft(g_m) − st(g_m)
   If C(RVM_VT(g_m),gct(g_m)) / Σ_{vt∈g_m} et(t)·C(OVM_VT(t)) < 1 then
  lease RVM_VT(g_m),gct(g_m)
                    from cloud resource provider
   End if
End for
```

VM Placement Management Scheme guarantees the network capacity for the continuous created VM considering the network interference occurred between co-location VMs, so that reduce the occurred data transmission delay between VMs for executing the scientific application efficiently. To do this, the science gateway manages the last used resource table *LastUsedResourceTable* for each user in order to save the physical node used lastly to create the VM. When a user requests a VM in the first time, there is no available last used resource information. In this case, the VM is created in a physical node *maxCapacityPN* having maximum free resource capacity within cloud server structure in order to increase the probability that the physical node keep a free resource capacity for the next VM creation of the user. After creating the VM, the information on the physical node of the created VM with the user id *uid* are saved to the last used resource table. When the user requests again, the last used resource information is available in the last used resource table. Therefore, a VM is created in the same physical node with the last used resource if possible in order to guarantee the maximum network bandwidth. In the case that the resource capacity of this physical node is not enough for the flavor type *f* requested by the user, available physical nodes *availablePNs* are sorted in the closest order from the physical node of the last used resource. The new physical node close to the physical node of the last used resource while having enough free resource capacity for flavor type *f* and having the smallest network traffic is chosen to reduce the network interference with other VMs on the physical node. The network traffic of each physical node is monitored in real-time. After finding the new physical node and creating the VM, the resource information on the new physical node is updated to the last used

resource table for the corresponding user. Algorithm 2 shows the procedure of the proposed network aware VM placement algorithm.

Algorithm 2. VM Placement Management Scheme

Input *:* uid^k, f (uid^k : *request id*, f : *flavor type*)

Output *: created VM*

```
01: if LastUsedResource of uid^k is available
02:     physicalNode p^k ← get LastUsedResource of uid^k
03:     if physicalNode p^k is available for flavor type f
04:       createdVM = createNewVM(p^k, f)
05:       return createdVM
06:     else
07:       sort availablePNs in closest order from p^k
            and split availablePNs into sameLevelPNs (∈ availablePNs )
08:       for sameLevelPNs ∈ availablePNs
            sort sameLevelPNs in smallest traffic order
          for p^i ∈ sameLevelPNs
09:         if p^i is available for flavor type f
10:           createdVM = createNewVM(p^i, f)
11:           record {uid^k, p^k} into LastUsedResource
12:           return createdVM
13:         end if
          end for
14:       end for
15:       return null
16:     end if
17:   else
18:   maxCapacityPN ← max( remainCapacity(p^j) )   ···  ( p^j ∈ PNSet )
19:   if maxCapacityPN is available for flavor type f
20:     createdVM = createNewVM(maxCapacityPN, f)
21:     record {uid^k, p^k} into LastUsedResource
22:     return createdVM
23:   else
24:     return null
25:   end if
26: end if
```

4 Test Environments and Performance Evaluation

For the evaluation, we built a science gateway, which is a solution platform to execute workflow typed scientific applications for solving complicated scientific problems by organizing the distributed heterogeneous cloud resources in this paper. The request from the user is dealt through the workflow designer as scientific application service provider. The workflow engine demands cloud resources to resource provisioning manager and process the scientific application. We applied a phased workflow scheduling scheme with division policy [3] as scheduler. Also, we organized Next Generation Sequencing (NGS) with Burrows-Wheeler Aligner (BWA) as a scientific application [4].

To evaluate the performance of the proposed schemes, we built the test environment with the science gateway as shown in Fig. 2 with the specific configuration on

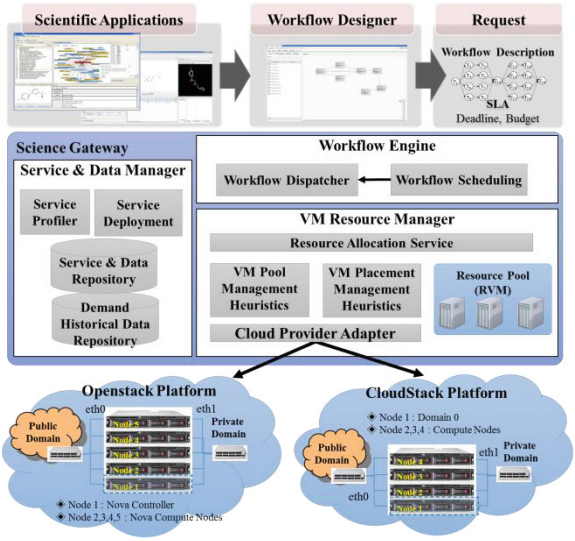

Fig. 2. An experimental testbed for cost adaptive resource schemes

Table 1. Specific configurations on testbed environment

		OpenStack Platform	CloudStack Platform
Hypervisor		KVM	XEN
H/W Specification		Intel Xeon E5620 2.40GHz, Core 16, MEM 16G, HDD 1T, 5 Node	Intel Core i7-3770 CPU 3.40GHz, Core 8, MEM 16G, HDD 1T, 4 Node
S/W Specification		OS: Ubuntu 14.04	OS: CentOS 6.0
VM Types	**small**	Spec: 1 VCPU, 2 GB MEM, 80GB Disk On-demand VM Unit Time Cost: 2 RC per second Reserved VM Unit Time Cost: 1,209,600 RC per week	
	medium	Spec: 2 VCPU, 4 GB MEM, 80GB Disk On-demand VM Unit Time Cost: 4 RC per second Reserved VM Unit Time Cost: 2,419,200 RC per week	
	large	Spec: 4 VCPU, 8 GB MEM, 80GB Disk On-demand VM Unit Time Cost: 8 RC per second Reserved VM Unit Time Cost: 4,838,400 RC per week	
	c4.small	Spec: 4 VCPU, 1 GB MEM, 80GB Disk Unit Time Cost: 4 RC per second Reserved VM Unit Time Cost: 2,419,200 RC per week	
	m8.small	Spec: 1 VCPU, 8 GB MEM, 80GB Disk Unit Time Cost: 4 RC per second Reserved VM Unit Time Cost: 2,419,200 RC per week	

testbed platform shown in Table 1. We organize the cloud platforms with 5 computing nodes on the OpenStack and 4 computing nodes on the CloudStack respectively [5, 6] to compose the heterogeneous resource.

We applied relative cost in order to evaluate the proposed scheme on cost performance in cloud environment. The definition of relative cost on the i^{th} cloud service is described as Eq. (2) where c_u^i is unit time cost and t_u^i *is* resource usage time.

For the convenience of this experiment, we assign unit time as a second for OVM and a week for RVM. The unit time cost on resource contract c_{ur}^{i} is calculated as Eq. (3) where the weight vector $\vec{w} = [w_c, w_m]$ means the effectiveness of each element such as CPU core and memory respectively. The weight vector for RVM is applied to half of OVM's with the reference of CSP GoGrid (on annual case) [7].

$$\mathrm{RC^i} = \mathrm{c_u^i} \cdot t_u^i \tag{2}$$

$$c_{ur}^{i} = w_c \cdot r_c^i + w_m \cdot r_m^i \tag{3}$$

In the experiment of the VM pool management scheme, we measured relative cost on the sum of OVM leasing cost and RVM leasing cost between the case without VM pool management, the case of the VM pool management scheme with static number of RVM (1, 2 respectively) and the case of the proposed scheme on various average interarrival time of workflow request in exponential distribution. This experiment is executed in scale-downed 4 weeks.

As shown in Fig. 3, the measured VM leasing costs for three cases are expressed in log scale. Our proposed scheme shows good adaptivity and cost efficiency on the resource management compared to the static pool management for all environment case. Therefore, we can decrease the VM leasing cost by the proposed scheme.

In the experiment of the VM placement management scheme, we evaluate the performance of proposed scheme compared to the existing VM placement scheme proposed by Alicherry and Lakshman [8].

The proposed scheme has the smaller total data transmission delay over the entire request interarrival time compared to the existing scheme as shown in Fig. 4. The existing scheme cannot guarantees the network capacity for the continuous created VM on BWA

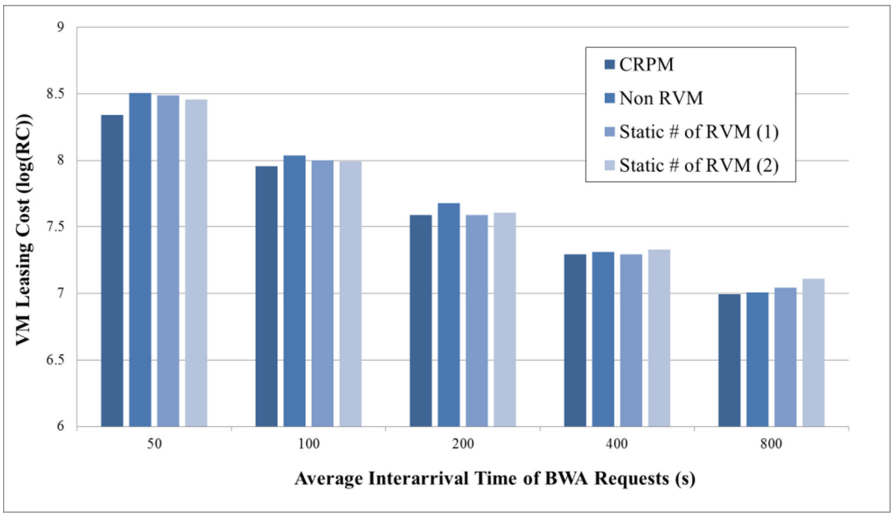

Fig. 3. Performance comparison of VM pool management scheme

service compared to the proposed scheme because it is only focused on the placement for a VM creation request and not considers the interference occurred by the traffic congestion.

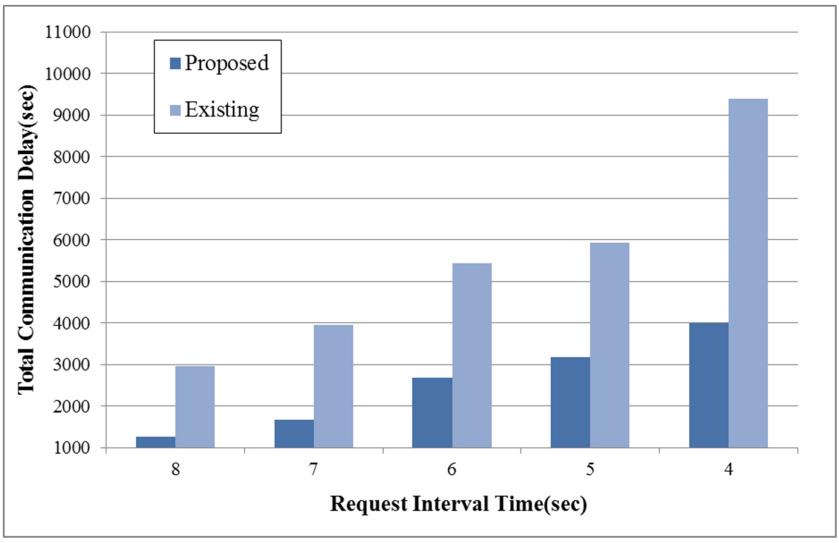

Fig. 4. The total data transmission delay of proposed scheme and existing scheme

5 Conclusion

In this paper, to decrease the resource usage cost while guaranteeing user's SLA on executing the scientific application in cloud environment, we propose and implement the science gateway with the cost adaptive resource management schemes. Experiment shows the proposed schemes decrease the expenditure of leasing VM instances and also guarantee the good performance. Finally, we demonstrate that our proposed schemes shows good adaptivity and cost efficiency compared to the existing schemes.

Acknowledgments. This work was supported by 'The Cross-Ministry Giga KOREA Project' grant from the Ministry of Science, ICT and Future Planning, Korea and the MSIP (Ministry of Science, ICT & Future Planning), Korea in the ICT R&D Program 2014.

References

1. Zhang, Q., Cheng, L., Boutaba, R.: Cloud computing: state-of-the art and research challenges. J. Internet Serv. Appl. **1**(1) (2010)
2. Chaisiri, S., Lee, B.S., Niyato, D.: Optimization of resource provisioning cost in cloud computing. IEEE Trans. Serv. Comput. **5**(2), 164–177 (2012)

3. Kim, S.-H., Joo, K.-N., Ha, Y.-G., Choi, G.-B., Youn, C.-H.: A phased workflow scheduling scheme with task division policy in cloud broker. In: Leung, V.C.M., Lai, R., Chen, M., Wan, J. (eds.) CloudComp 2014. LNICST, vol. 142, pp. 76–86. Springer, Heidelberg (2015)
4. BWA. http://bio-bwa.sourceforge.net
5. Openstack. http://www.openstack.org
6. Cloudstack. http://cloudstack.apache.org
7. GoGrid. http://www.gogrid.com/
8. Mansoor, A., Lakshman. T.V.: Network aware resource allocation in distributed clouds. In: 2012 Proceedings IEEE, INFOCOM. IEEE (2012)