Agent based Scheduling of Real Time Tasks in Cloud Computing using Bidirectional Bidding Mechanism

Y. R. Himadyuti
M.Tech, ISE Department,
AMC Engineering College,
Bangalore, India.

Prof. J Selvin Paul Peter
HOD, ISE Department
AMC Engineering College
Bangalore, India.

***Abstract*—The cloud computing has become an efficient paradigm to run real-time and complex applications to store and manage the voluminous amount of data. Scheduling plays a major role in managing the execution of real-time applications in cloud computing environment. Cloud provider provides services to run the tasks in cloud and maintain the quality of service for its customers. To ensure the cloud provider's QoS, we propose a novel agent-based scheduling mechanism to run the customers' tasks in cloud and provide dynamic resources. In agent-based scheduling, we introduce a bidirectional announcement based bidding mechanism to allocate tasks and resources dynamically. In addition, it consists of three phases, i.e., basic matching phase, forward announcement-bidding phase and backward announcement-bidding phase. However, the scalability is necessary when scheduling is performed in dynamically adding virtual machines which gives elasticity. We have designed calculation rules for both forward and backward announcement bidding mechanism which helps for selecting contractors. The experiments are evaluated for both random synthetic workload and Google cloud trace logs. The experiment results show that our agent-based scheduling mechanism gives better performance compared to existing methods.**

Keywords*— Agent - Based Scheduling, Bidirectional Announcement Bidding Mechanism, Cloud Computing, Dynamic Resources.

I. INTRODUCTION

The computing world is rapidly transforming towards developing software for millions to consume as a service, rather than to run on their individual computers. In cloud computing, most of the services are based on pay-per -use [1]. The virtualization technology is generally used in clouds, which is used to separate a single physical machine into multiple virtual machines in a cost-effective way [2]. Virtual machines should be dynamically provisioned to match actual application demands, rather than the peak requirement. Cloud service providers have to ensure that they can be flexible in their service delivery to meet various consumer requirements, while keeping the consumers isolated from the underlying infrastructure [3]. Using the server virtualization technology, a single host can run multiple Virtual Machines (VMs).The VMs will achieve fine-grained optimization of computing resources which dynamically relocated the live VM operations such as creation, migration and deletion [4].

To obtain high performance for real-time heterogeneous systems, scheduling algorithms play an important role. Examples of such applications include aircraft control systems, transportation systems and medical electronics [5]. In a commercial multi cloud environment, it consists of a lot of entrusted cloud providers with private information about their resources. The problem of scheduling of workflows in heterogeneous computer systems is known to be NP-Complete.

The agent-based technology deals with task allocation issue in distributed systems which derived from Distributed Artificial Intelligence (DAI) [7]. According to Wooldridge and Jennings, an agent has capable of controlling its own decision making in the cloud environment which provides a way to allocate tasks.

In agent-based scheduling technique, it has multiple individual agents which are capable of handling multi-agent system. The agents are interacting with each other in a system which has the ability to cooperate, coordinate, and negotiate.

II. RELATED WORK

Scheduling strategies are used in many applications and in various areas. Qin *et al.* proposes scheduling algorithms can be either static or dynamic [5]. The Quality of Service (QoS) requirements must be addressed in various hard and soft real-time systems. In fault tolerant, it introduces primary-backup model which executes the backup scheduler simultaneously while executing primary scheduler [9]. It achieves high flexibility, availability and schedulability of the resources.

VM consumes many resources which reduce the resource consumption; this issue is solved by Genetic Algorithm (GA) [4]. Kong *et al.* uses the type-I and type-II fuzzy logic systems predict the resource availability and workloads to improve the system availability and performance. Rodriguez and Buyya suggested a resource provisioning and scheduling strategy for real-time workflow on Iaas cloud, in which the swarm optimization technique minimize the overall workflow execution within timing constraint [10].

Goiri *et al.* proposed an energy-efficient and multifaceted scheduling policy, in this approach for VM allocation it consider the maximum provider's profit [11]. Grauber *et al.* proposed energy-efficient scheduling algorithm, to achieve energy saving in cloud environment it performs live migrations in virtual machines. However, the aforesaid algorithm does not solve the issue of scalability in dynamic scheduling efficiently. Meanwhile, the scheduling of tasks and resources should be performed dynamically in cloud environment.

In agent-based scheduling, the jobs are scheduled dynamically in an efficient manner. The agent-based distributed manufacturing scheduling are based mainly on the combination of market-based mechanisms used by the multi-agent systems. Further, scheduling is done in distributed system which allows parallel computing and improves the performance of the system.

In this study, we investigated a novel agent-based scheduling to improve the scheduling based on CNP model. The proposed strategy adopts the bidirectional-bidding mechanism where both tasks and resources are announcers and bidders.

III. SYSTEM MODEL

In this section, we deal with the system model, notation and proposed model architecture used in our work.

A. *Preliminaries of ABSBB*

The host set is represented as $H= \{ h_1 , h_2 ,.., h_k \}$ where h_k is the number of host .and the active host set is represented as H_a with n elements, $H_a \subseteq H$. For each host $h_k \in H_a$ it has a VM set which represent in the form $V = \{V_1, V2, ...V_{|H_a|} \}$. A set of task is represented in the form $T = \{t_1, t_2 ...\}$ where the tasks are independent, non-preemptive, and aperiodic and essentially with deadlines. A task t_i submitted by a user can be modeled by a collection of parameters i.e., $t_i = \{a_i ,l_i , d_i , p_i \}$ where a_i ,l_i , d_i and p_i are the arrival time, task length, deadline, and priority respectively. Let s_{ijk} represents the start time and f_{ijk} represents the finish time of task t_i on v_{jk} .

In this paper, we followed some constraints to run the task on VM. Each task can only be allocated to one VM which cannot be shared by other task

The start time s_{ijk} can be calculated as,

$$s_{ijk} = \max\{a_i, r_{jk}\}, \tag{1}$$

The ready time r_{jk} can be calculated as,

$$r_{jk} = \begin{cases} c_{jk} & if\ \forall x_{pik} = 0 | a_p \le a_i, \\ \max \{f_{ijk}\} & if\ \exists x_{pjk} = 1 | a_p \le a_i. \end{cases} \tag{2}$$

The finish time f_{ijk} can be calculated as,

$$f_{ijk} = s_{ijk} + e_{ijk} \tag{3}$$

The finish time determine whether the task's deadline can be satisfied. If both the preceding task and succeeding task are allocated to same VM, which cannot be execute at same time. This comes to the following constraint C_1.

$$c_1 : \begin{matrix} (s_{ijk} \ge f_{pjk}) V \\ (f_{ijk} \le s_{pjk}) \end{matrix} \quad if_{\, x_{ijk} = x_{pjk} = 1}^{\, \forall i \ne p, j \in [1, |V_k|], \exists k \in [1, |H_a|],} \tag{4}$$

B. *System Architecture*

Our system architecture is depicted in fig. 1. The Agent based bidirectional bidding mechanism is designed with three main agents namely VM Agent, Task Agent, and Manager Agent. The basic interaction between agents is performed by the following steps,

Initially, the process starts with the VM Set. From the VM Set, a new VM is generated and updated in VM Agent. Then, the VM information sends to Manager Agent. From the Task Set, a new task is generated in Task Agent. Task Agent sends task information to Manager Agent. Later, Manager Agent filters matching result and sends to Task Agent. Then, Task Agent sends forward announcement to VM Agent. VM Agent calculates forward bidding and sends to Task Agent. Meanwhile, Task Agent awarded VMs to VM Agent. However, VM Agent sends backward announcement to Task Agent. Then, Task Agent calculates backward bidding and sends to VM Agent. Finally, bidirectional contract is awarded between task and VM Agent. In this process, VM can be added dynamically when a new task is arrived.

C. *Agent Based Model*

Our Agent based model constitutes of three agents, i.e., task agent, VM agent and manager agent. An agent is a self-reliable program that is capable of controlling its own acting and decision making, based on its perception of its environment, in pursuit of one or more objectives. A task agent is created with the arrival of the task and disappears with the end of the task. A VM agent is created when the VM is established, and dies out once the VM is destroyed. The manger agent is usually existent. The manager agent receives information from both the task agents and VM agents. The agent model is designed as follows,

- $T^A = \{ti^A , i = 1,2,...,| T |\}$ is a task agent set, where t_i^A represents the ith task agent inT^A .
- $V^A = \{v^A_{jk} , j = 1,2,... | v_k |, k = 1,2,...,| H_a |\}$ is a VM agent set, where V_{jk}^A donates the jth VM on h_k .
- m^A represents a manager agent.

IV. AGENT BASED BIDIRECTIONAL BIDDING MECHANISM

The bidirectional bidding mechanism which deals with three phases, i.e., basic matching phase, forward-announcement bidding phase, backward-announcement bidding phase.

A. *Basic Matching Phase*

In this phase, VM need to satisfy the basic requirement of tasks. If it satisfies, the manager agent will filter the task and VM. The process can be explained in detailed as follows:

1. When a new task arrives, the new task agent is generated. The task agent sends the task information to manager agent.
2. After a new task arrives, dynamically a new VM is generated by VM agent. The VM agent sends the VM information to manager agent.
3. The manager agent receives the task and VM information, then matches the information and filters the results.
4. Then the manager agent sends the selected VM information to task agent.

B. Forward-Announcement Bidding Phase

In this phase, the task agent selects the VM to execute the task in it. The VM is selected based on the bidding value. The process can be explained in detailed as follows:

1. Firstly, the task agent receives the VM information from the manager agent.
2. Then each task agent sends the task information with its parameter such as arrival time, length, priority, deadline to the VM agent.
3. After making forward announcement, VM Agent calculates the bidding value.
4. Task agent receives the bidding value of VM agent and make forward contract.

The task agents negotiate with VM agent to select one or more VMs for a task on situation with the intention of the VMs can guarantee the timing constraint.

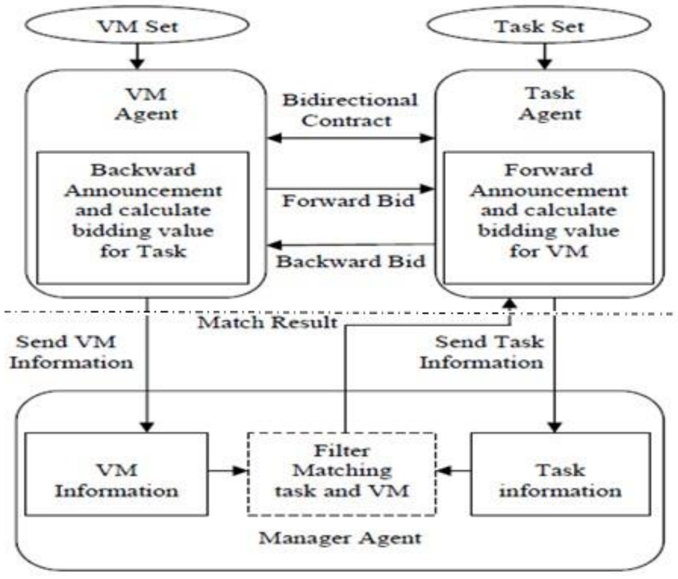

Fig. 1. System Architecture

C. Backward-Announcement Bidding Phase

In this phase, the VM agent selects the task to run on it. It selects the task based on the bidding value. The process can be explained in detailed as follows:

1. VM agent sends backward announcement to task agent.
2. Then the task agent receives the VM agent information and calculates the bidding value.
3. After calculation, task agent sends the bidding value to VM agent.
4. Then the VM agent receives the bidding value and make backward contract.
5. Later, bidirectional contract is perform between task and VM agent.

D. Bidding Value Computation:

In previous section, the selection is mainly based on the bidding values. The forward-announcement phase deals with forward bidding computation and the backward-announcement phase deals with backward bidding computation.

1) Forward Bidding Computation: In forward-announcement phase, the task agent selects the VM based on the calculation of forward bidding value. The calculation rule for forward bidding is as follows:

$$Fb_{ijk} = d_i - e_{ijk} - f_{pjk} \qquad (7)$$

The f_{pjk} represents the preceding task finish time on the same VM. If $fb_{ijk} > 0$, it can able to complete the task before deadline. Else, it cannot finfish the task before deadline.

2) Backward Bidding Computation: In backward-announcement phase, the VM agent selects the task based on the calculation of backward bidding value. The calculation rule for backward bidding is as follows:

$$bb_{ijk} = \frac{p_i^{\theta} . e_{ijk}}{\left(d_i - f_{ij}\right) \sum_{k=k_i}^{k} \sum_{j=j_i}^{j} fb_{ijk}} \qquad (8)$$

Where bb_{ijk} is used to represent the bidding value of task to v_{jk} and the parameter θ represents the weight of priority. While allocating the task, highest priority is considered.

E. Strategies for Bidding

In both forward and backward bidding, we use selection strategies for awarding contract between task and VM agent. Two selection strategies are used for awarding contract which is MAX strategy and P Strategy.

- AN = {an_i,i = 1,2,. . n} denote an announcer set.
- BI_i = {bi_{ij},j = 1,2, … m_i} denote a bidder set for an announcer.
- BV = {br_{ij}, i = 1, 2 …n, j = 1,2,.. m_i}denote a bidding value set.

1) MAX Strategy: When more than one bidder bid at the same time, the announcer will choose the maximum bidding value $\forall an_i \in AN$, if bi_{ik} is selected, it must meet the constraint:

$$\forall bv_{ij} \in BV \Rightarrow bv_{ijk} \geq bvij \mid j \neq k$$

2) P Strategy: P Strategy is a probability selection strategy. When more than one bidder bid at the same time, the announcer will choose the bidding value based on the probability policy.

$\forall an_i \in AN$, the winning probability pr_j of bidder br_{ij} can be calculated as:

$$pr_j = bv_{ij} / \sum_{k=1}^{m_i} bv_{ik} \qquad (9)$$

If the randomly generated number satisfies the following formula (10), then the bidder br_{ij} is selected as a contractor.

$$\sum_{m=1}^{j-1} pr_m < pr \leq \sum_{n=1}^{j} pr_n \qquad (10)$$

V. AGENT BASED SCHEDULING USING BIDIRECTIONAL BIDDING MECHANISM

We present our novel agent based scheduling algorithm in cloud computing – ABSBB on the basis of our agent-based scheduling model for independent, aperiodic, real-time tasks.

A. Role of Manager Agent

In Algorithm 1, it shows the pseudo code of manager agent. The manager receives the new tasks and chooses the VM which satisfy the basic requirement of task agents. Then it sends the selected VM to task agent to run the task.

Algorithm 1: Process of Manager Agent

```
1. $T_{waiting}$ the tasks that arrive at the same time instant;
2. foreach $t_i^A$ in $T_{waiting}$ do
3.     $V_i^A$ the VM agents that satisfy the basic requirements of $t_i^A$;
4.     Send the $V_i^A$ to task agent $t_i^A$;
5. while $T_{waiting} \neq \emptyset$ do
6.     The task agents start the process by algorithm 2;
7.     The VM agent start the process by algorithm 3;
```

Line 1 represents the new arrival of task is in task queue. Line 2- 4 represents that the manager agent chooses the VM for task agent. Line 5-7 performs the task allocation.

B. Role of Task Agent

Task Agent sends the announcement information to VM Agent. According to forward bidding value which is calculated by VM Agent, Task Agent selects a VM based on selection strategy and makes forward contract for VM.

Algorithm 2: Process of Task Agent

```
1. $valueList \leftarrow \emptyset$;
2. foreach $v_{jk}^A$ in $v_i$ do
3.     Task agent $t_i^A$ sends the announcement information to $v_{jk}^A$;
4.     $fb_{ijk} \leftarrow$ VM agent $v_{jk}$ calculates the forward bidding value;
5. if $fb_{ijk} \geq 0$ then
6.     $valueList.add\ (fb_{ijk})$;
7. if $valueList \neq \phi$ then
8.     Task agent $t_i$ selects a bidder based on the values in valueList $v_{select}$ using the MAX/P strategy
9. else
10.    $v_{new} \leftarrow$ scaleUpResources();
11. if $v_{new} \neq NULL$ then
12.     The new VM generates a new VM agent and send the information to manager agent;
13.     Allocate $t_i$ to $v_{new}$;
14.     $T_{waiting}.remove\ (t_i^A)$;
15. else
16.     Reject $t_i$;
17.   $T_{waiting}.remove\ (t_i^A)$;
```

Line 2-4 represents the task agent announcement information is send to VM agent and receives the forward bidding value. Line 5 represents whether the task can be completed before its deadline or not. Line 7 and 8 represents the selection strategy. Line 9-17 represents a new VM creation which performed by algorithm 4 and represents the allocation and rejection of task.

C. Role of VM Agent

VM Agent sends the announcement information to Task Agent. According to backward bidding value which is calculated by Task Agent, VM Agent selects a Task based on selection strategy and makes bidirectional contract between Task and VM.

Algorithm 3: Process of VM Agent

```
1. $T_{candidate} \leftarrow$ the task agents that send the forward contract with the VM agent $v_{jk}^A$;
2. if $T_{candidate} \neq NULL$
3.     $valueList \leftarrow \emptyset$;
4. foreach $t_i^A$ in $T_{candidate}$ do
5.     $bb_{ijk} \leftarrow t_i^A$ 's backward bidding value;
6.     $valueList.add\ (bb_{ijk})$;
7.   VM agent $v_{jk}^A$ selects a bidder $t^A_{select}$ based on the values in valueList using the MAX/P strategy
8.   Build a bidirectional contract between the task and VM;
9. $T_{waiting}.remove\ (t_i^A)$;
```

Line 4-6 represents the VM agents backward announcement information which sends to task agent and receives the backward bidding value calculated by task agent. Line 7 represents that the VM agent select the task based on the selection strategy. Line 8 represents the bidirectional contract between the task and VM.

D. Dynamic Resource Provisioning

Algorithm 4: Function scaleUpResources ()

```
1. Create newVM with processing power $P_{new}$;
2. find $\leftarrow$ FALSE ; $v_{new} \leftarrow$ NULL ;
3. Foreach $h_k$ in $H_a$ do
4.     if $h_k$ can accommodate newVM then
5.         Allocate newVM to $h_k$;
6.         $v_{new} \leftarrow$ newVM; find $\leftarrow$ TRUE ;
7.         break;
8. if find == FALSE then
9.   sourceHost $\leftarrow$ Migrate VMs among the hosts to make room for newVm;
10.if sourceHost $\neq$ NULL then
11.    Allocate newVm to sourceHost;
12.    $v_{new} \leftarrow$ newVM ; find $\leftarrow$ TRUE ;
13. if find==FALSE then
14.    Turn on a host $h_{new}$ in $H - H_a$ ;
15. if the capability of $h_{new}$ satisfies $P_{new}$ then
16.    Allocate newVM to $h_{new}$ ;
17.    $v_{new} \leftarrow$ newVM ;
```

In this algorithm, it creates a new VM when a new task arrives. Line 3-7 represents that the host can able to accommodate the new VM. If it accommodate, then allocate the new VM to host. Line 9 represents that migration of VM when the host doesn't accommodate the new VM. Line 14 represents if migration also gets fail, and then turn the passive host into active host.

E. Improved Greedy Strategy

In Improved Greedy Algorithm it discuss about scheduling without any selection strategies.

Algorithm 5: The I-Greedy Algorithm

```
1. T ← newly arrived tasks and tasks in the waiting queue;
2. find ← FALSE;
3. foreach $t_i$ in $T$ do
4.     foreach $v_{jk}$ in $V$ do
5.         Calculate the bidding value $fb_{ijk}$ ;
6.         if $fb_{ijk} > 0$ then
7.             Allocate $t_i$ to $v_{jk}$ ;
8.             find ← TRUE;
9. if find == FALSE then
10.     Create a new VM $v_{pq}$ and calculates the bidding value
        $fb_{ipq}$ ;
11. if $fb_{ipq} > 0$ then
12.     Allocate $t_i$ to $v_{pq}$ ;
13. else
14.     Reject $t_i$ ;
```

Line 1 represents when a new task arrives from the waiting queue it stores the task information in a task list. Line 3-8 represents, when each new task arrives it calculates forward bidding and allocate the task to VM. Line 9-14 create a new VM and calculate the forward bidding value and allocate the task to VM. Else, it rejects the task.

VI. EXPERIMENTAL SETUP AND RESULT ANALYSIS

To evaluate the proposed method, we used cloudsim toolkit for simulation. The performance of ABSBB scheduling is observed by different constraints as follows.

The performance of the proposed system has been evaluated based on various numbers of VMs with different configurations and list of tasks with different deadlines.

In fig. 2, it is observed that the task completion time is less as the number of tasks increases. The number of tasks in each VM's Queue will be in waiting stage until it gets the CPU.

In fig. 3, it is observed that the average waiting time of the task in VM queue to get the CPU is very minimal when compared to the existing system. The waiting time increases as the number of tasks with the tight deadline arrives in the VM. Our proposed system can satisfy the task to complete within specified deadline.

The performance of real workload i.e., Google cloud trace logs run for 29 days which gives the results for 25 million tasks. On the day of 18, it gives the performance result for 955,626 tasks which is submitted to the cloud.

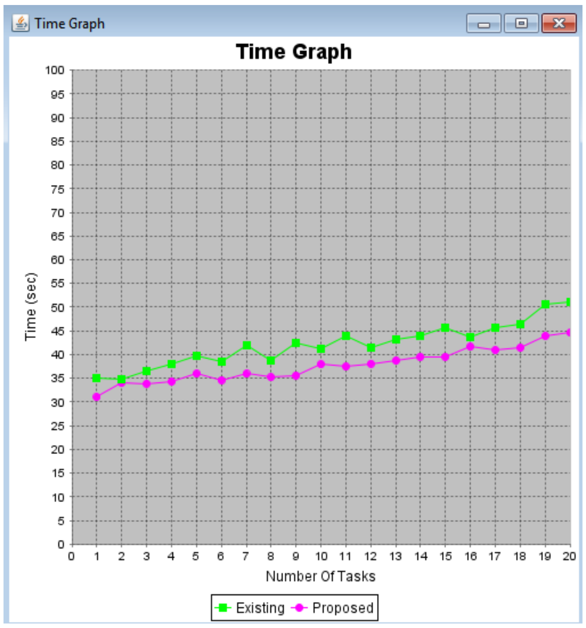

Fig. 2. Time Graph for ABSBB.

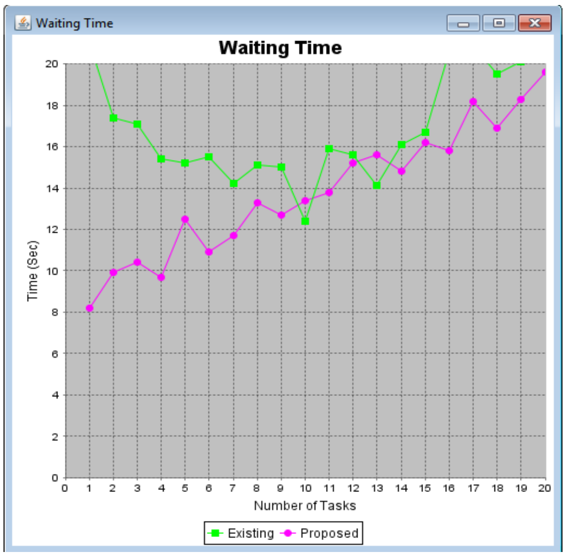

Fig. 3. Waiting Time of tasks in the VM's queue.

VII. CONCLUSION AND FUTURE WORK

In this paper, we introduced an agent-based bidirectional bidding mechanism for task scheduling in cloud. We designed different agents to schedule the tasks; it includes the forward and backward announcement bidding mechanism. The calculation rules and selection strategies are used to award contract between task and VM for scheduling. In our future work, instead of MAX Strategy and P Strategy we are going to implement Game based bi-objective selection strategy. In Game based bi- objective selection strategy, the winner of task and resource is selected by the minimum product of time and cost across all agents.

REFERENCES

[1] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, I. Brandic, “Cloud Computing and Emerging IT Platforms: Vision, Hype, and Reality for Delivering Computing as the 5th Utility,” *Future Generation Comput.Syst.*, vol. 57, no. 3, pp. 599-616, 2009.

[2] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, and I. Stoica, “A View of Cloud Computing,” *Commu. the ACM*, vol. 53, no. 4, pp. 50-58, 2010.

[3] A. Beloglazov, J. Abawajy, and R. Buyya, “Energy-Aware Resource Allocation Heuristics for Efficient Management of Data Centers for Cloud Computing,”

Future Generation Comput. Syst., vol. 28, no. 5, pp. 755-768, 2012.

[4] L. He, D. Zou, Z. Zhang, C. Chen, H. Jin, and S. A. Jarvis, “Developing Resource Consolidation Frameworks for Moldable Virtual Machines in Clouds,” *Future Generation Comput. Syst.*, vol. 32, pp. 69-81, 2014.

[5] X. Qin and H. Jiang, “A Novel Fault-Tolerant Scheduling Algorithm for Precedence Constrained Tasks in Real-Time Heterogeneous Systems,” *Parallel Comput.*, vol. 32, no. 5, pp. 331-356, 2006.

[6] L. F. Bittencourt, E. R. M. Madeira, and N. L. S. da Fonseca, “Scheduling in Hybrid Clouds,” *IEEE Communications Magazine*, pp. 42-47, 2012.

[7] H. Goldingay and J. Mourik, “The Effect of Load on Agent-Based Algorithms for Distributed Task Allocation, Inf. Sci., vol. 222, pp. 66C80,Feb. 2013.

[8] K M. Sim, “Agent-Based Cloud Computing,” *IEEE Trans. ServecesComputing*, vol. 5, no. 4, pp. 564-577, 2012.

[9] X. Zhu, X. Qin, and M. Qiu, “QoS-Aware Fault-Tolerant Scheduling for Real-Time Tasks on Heterogeneous Clusters,” *IEEE Trans. Computers*, vol. 60, no. 6, pp. 800-812, 2011.

[10] M. Malawski, G. Juve, E. Deelman, and J. Nabrzyski, “Cost –and Deadline-Constrained Provisioning for Scientific workflow Ensembles in IaaS Clouds,” *Proc. Int’l Conf. High Performance Computing, Networking,Storage and Anal (SC ’12)*, pp. 1-11, 2012.

[11] I. Goiri, J. L. Berral, J. O. Fit_o, F. Juli_a, R. Nou, J. Guitart, R. Gavald_a, and J. Torres, “Energy-Efficient and Multifaceted Resource Management for Profit-Driven Virtualized Data Centers,” *Future Generation Comput. Syst.*, vol. 28, pp. 718-731, 2012.

[12] Google Cluster Data V2: http://code.google.com/p/google/clusterdata/wiki/ClusterData2011_1