ISSN(Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Special Issue 6, May 2015

Enhanced Fault Tolerance and Cost Reduction using Task Replication using Spot Instances in Cloud

Mohana Saranya S[1], Srimathi T[2], Ramanathan C[3], Venkadesan T[4]

Assistant Professor, Dept of Computer Engineering, Kongu Engineering College, Perundurai, Tamil Nadu, India[1]

UG Student, Dept of Computer Engineering, Kongu Engineering College, Perundurai, Tamil Nadu, India[2,3,4]

ABSTRACT: In recent years, Cloud computing is fast evolving due to its vast usage and its features. There are many pricing models in cloud. Instances in cloud are classified based on those pricing models. Out of those instances, on-demand instances have more QoS and high cost. Spot instances provide resources at reduced cost. But they are more prone to Out-of-failures. In this paper, we propose a method to enhance the fault tolerance for spot instances. This method uses task replication method to provide improved fault tolerance. Experimental results show that our method reduces execution cost to a great extend since it uses only spot instances.

KEYWORDS: Spot instances, Cloud, Fault tolerance, Task migration, Task replication

I. INTRODUCTION

Cloud computing is a distributed computing method providing computing resources as services. These resources are readily scalable and highly available. They are provided in an on-demand manner and also they are location transparent. Cloud computing can be a faster and cheaper alternative to an on-premises solution. You can get powerful software and massive computing resources without any infrastructure investments. Over the last few years, Cloud computing has generated a major impact on the global IT sector, giving rise to new user communities and new markets.

Cloud provides large scale services with the help of cloud servers. The services can be either computation service or storage service. Virtualization is used for dynamic configuration of these services. Any application in cloud computing environment can be represented by means of a workflow. However, this computing environment still cannot provide the quality, robustness and reliability that are needed for the execution of workflows because of different failures like failure of server providing the service, link failure, malicious code in the executing node, datasets required by the task may be acquired by other tasks etc.

II. PRICING MODELS

Dedicated Instance pricing has two components: (1) a dedicated per region fee (note that one need to pay this once per hour regardless of how many Dedicated Instances running) and (2) an hourly per instance usage fee. On-Demand Instances allow you pay for compute capacity by the hour with no long-term commitments. Spot Instances allow you to name your own price for the instance. You simply bid on spare instances and run them whenever your bid exceeds the current Spot Price, which based on demand. The Spot Instance pricing model complements the On-Demand and Reserved Instance models, often providing the cost-effective computing capacity.

The provider determines the price of a Spot Instance (spot price) based on the instance type and demand within the data center. Spot price of a instance changes with time, it is different for different instance types. The price also changes between regions and availability zones. Here, the users participate in a bidding market and bid a maximum price they are ready to pay for SIs. The user is aware of the number of bidders and their bid prices. The user is provided the resource or instance whenever their bid value is higher than or equal to the spot price. However, when the spot price

ISSN(Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Special Issue 6, May 2015

becomes higher than the user bid, provider terminates the resources.

III. RELATED WORKS

The existing scheduling algorithm maps ready tasks submitted by the task dispatcher to Cloud resources. This algorithm along with a suitable instance type also selects a suitable pricing model to minimize the overall cost. The objective of the algorithm is to map tasks that arrive to SIs and those that are prematurely terminated (out-of-bid failure) to on-demand instances. In this approach, a single Spot Instance type is used. This instance has low cost. The basis behind this is to reduce the overall execution cost. On the other hand, multiple types of on-demand instances are used. This helps to slow down and speed up execution.

Run time of a given task change with different instance types. The scheduling decision changes based on the instance types used. There are two scheduling algorithms developed by considering, Conservative and Aggressive algorithm.

Conservative algorithm: It uses Spot Instances only when the deadlines are relaxed. Under tight deadlines, it does not generate required slack time to utilize Spot Instances and therefore maps tasks mostly to on-demand instances. It is conservative in its approach and utilizes Spot Instances in a cautious manner only under relaxed deadline.
Aggressive algorithm: This approach generates high slack time than the previous algorithm and therefore uses Spot Instances even with a very strict and moderate deadline. This offers efficient reduction in cost under moderately low deadline. Under less deadline both algorithms work similarly. When the market is fragile inducing failures, this approach has low slack time. Hence, it has to map to on-demand instances that are expensive, increasing the overall execution cost.

Spot instances offer the compute instance at a much reduced price. These will be terminated prematurely if the bid price value goes below the spot price value. The failure of Spot Instances is governed by the bid price. Therefore, an intelligently calculated bid price minimizes the risks of failures. The bid price is provided by any one of the bidding strategies. If the bid price is higher than the on-demand price value, the algorithm maps on-demand instances as they offer high QoS. Also, the bid price varies with the spot price. Therefore, the algorithm verifies that the bid price is higher than the previous bid cost, if not the previous bid price value is used. The algorithm also calculates the failure probability of a bid price based on the history. Failure probability of the current bid price value is estimated with the help of failure estimator. If the failure probability is higher than the user defined threshold, the algorithm chooses on-demand instance instead of Spot Instances.

While creating a Spot Instance, it also estimates the risk propositions and bid intelligently. Spot Instance with the calculated bid price value is instantiated by the resource provider. The applications are more likely to exhibit different levels and patterns, and the distributed resources organize into various topologies for query dissemination. Fault identification and tolerance should be taken into consideration, to ensure effective performance. Checkpointing is the method to make the most of a diverse set of tasks with fault identification from the available resources in cloud efficiently. For the purpose of fault identification in the scheduling of tasks, checkpoints are added to identify the occurrence of fault with less time and less computation required.

IV. PROPOSED SYSTEM

A. TASK REPLICATION CONTROL FLOW CHART

The scheduling system in a cloud should overcome out-of-bid failures which can be provided by fault tolerant scheduling algorithms. Fault tolerant scheduling algorithms can be categorized based on check pointing, replication of tasks and resubmission of tasks. Each category has its own pros and cons. Workflow is a sequence of tasks processed in an order based on control or data dependency between those tasks.

ISSN(Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Special Issue 6, May 2015

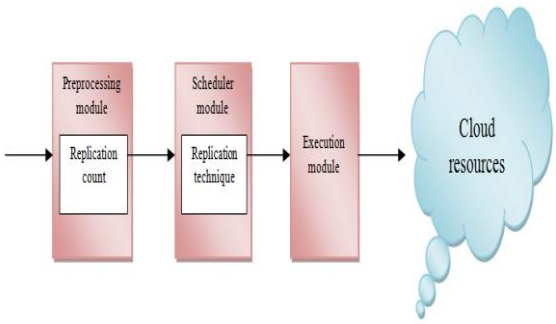

Fig 1 Control Flow Diagram

Task replication process is designed with three major modules: Preprocessor Module (PM), Replication based Scheduler Module (RSM) and Executor Module (ResEM) with Rescheduling if required. The control flow diagram of these modules is shown in Figure 1

There are two types of servers in cloud which are storage server and computational server. Storage server provides the services related to storage of data and modification which does not require any mapping of those services. Computational server provides the services related to computing resources which requires mapping of those services to a task. Task replication system replicates the task and schedules the tasks to meet the deadline of a workflow by replication of tasks. This replication of tasks is performed in the scheduling phase.

Users initially submit their workflows with deadline and replication factor in the form of abstract data structure format to PM (Preprocessor module). The PM discovers the services required for those tasks submitted and divides the tasks based on computation services and storage services. It also generates a threshold which helps in prioritizing the tasks and a heuristic metric which indicates replication count. Then RSM replicates the tasks based on the priority and replication count. It allocates the particular services to these tasks in cloud environment. After mapping, ResEM sends the tasks to the servers and starts the timer based on an expected execution time. If ResEM receives the successful output from the server within the expiry time then it activates all the other tasks which are dependent on the task. If it fails to receive successful output from server, then ResEM waits for other replicas. If all replicas also fail then it resubmits the task.

B. METHODOLOGY

Mapping between the tasks and services depends on the deadline of workflow and the computation power of those resources available. Execution time of these workflows can be hours, days or even weeks and hence the realization of workflow failure at the end of workflow may lead to missing of the deadline. The workflow execution system with low tolerance for failures may cause a situation that user may not realize the failure of workflow and so tasks may miss their deadline. Hence the workflow scheduling system should employ high levels of tolerance for failures.

In this paper, fault tolerance is achieved by compromising task replication and task resubmission methods. Tasks are replicated based on priority of the task. Tasks are prioritized based on length, out degree, earliest deadline or high resubmission impact. Tasks are scheduled to meet deadline by considering priority and thereby to reduce wastage of resources by restricting unnecessary replication of tasks. Here, the objective of these methods is to schedule the tasks within its deadline by tolerating the faults that may be present in a cloud computing environment to provide good success rate of scheduling.

Methods like replication and resubmission of tasks does not require any history of information. Few techniques provide fault tolerance in spot instances by making use of replication. All tasks are replicated to the maximum count which provides high fault tolerance but uses lot of Spot Instances. The tasks which can be executed on highly reliable

ISSN(Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Special Issue 6, May 2015

resources are also replicated and so the resources are wasted. If there is enough number of instances available then this method will give better fault tolerance power to the workflow scheduling system. Most of the times, number of tasks will be high when compared to the number of instances available and hence this method may lead to task serialization instead of parallel execution of tasks.

Another technique provides fault tolerance by using resubmission of tasks to instances. Here, tasks are resubmitted to the same or different instance after failure declaration which will not waste any resource but increases the overall execution time of a workflow. This may also lead to missing deadline of a workflow and hence gives the low success rate of scheduling when deadline is considered.

To overcome the disadvantages of replication and resubmission, few techniques are available which finds the tradeoff between the replication and the resubmission. Tasks are replicated without considering any other parameters except the heuristic metric calculated which may replicate all the tasks even if they are not important and may lead to resource wastage.

Hence, in this paper, prioritization of tasks is done along with heuristic metric. The tradeoff between replication and resubmission factor is used to find the heuristic metric that indicates the replication count for each task. Priority for the tasks is provided by using parameters like length of tasks, out degree, earliest deadline and high resubmission impact. Prioritizing the tasks helps to meet the deadline of workflow and reduces resource wastage along with providing fault tolerance for the workflow system.

V. EVALUATION RESULTS

A. Simulation Setup

CloudSim, cloud simulator was used to simulate the Cloud environment. It was extended to support workflow applications in cloud environment. *Application Modeling:* Small workflow with size of 10 tasks was considered. *Resource Modeling:* A Cloud model with a single datacenter and a host is considered. We have considered two instance types for on-demand instances and one instance type for SI.

B. Analysis and Results

In this section, we discuss the execution cost incurred by both pricing models and the effect of task replication on existing model.

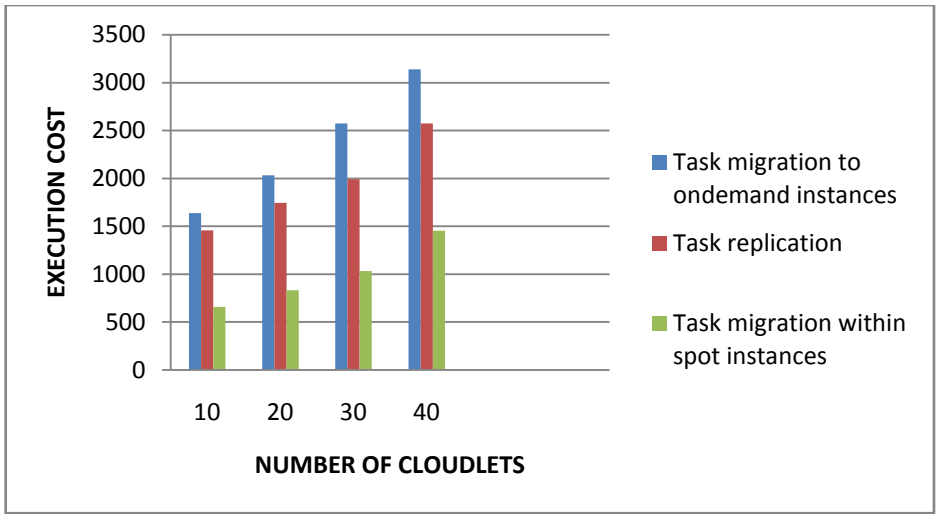

Fig 2 Sample line graph showing the reduction of cost

ISSN(Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Special Issue 6, May 2015

As shown in the figure 2, the execution cost of spot instances is much lower than that of on-demand instances. Spot Instances offers Cloud users reduction in costs of up to 60% for multiple applications. Significant cost reductions are achieved due to lower QoS which make them less reliable and prone to out-of-bid failures. This introduces a new aspect of reliability into the SLAs and the existing trade-offs thus making it challenging for Cloud users.

On using task replication and task resubmission instead of task migration the execution cost is much reduced because only spot instances are used in task replication. Also prioritization of tasks to find heuristic replication count reduces resource wastage.
Thus the task replication offers a better fault tolerance in cloud environment with less execution cost.

VI. CONCLUSIONS AND FUTURE WORK

In this paper, task replication and task resubmission mechanisms are used that handle the out-of-bid failure caused in spot instances are presented. They minimize the execution cost without degrading the effectiveness of fault tolerance. They are shown to be robust and fault-tolerant towards out-of-bid failures and performance variations of Cloud instances.

Task replication and task resubmission offers reduced execution cost compared to task migration. An issue associated with above methods is that, the resource wastage is high because when all tasks succeed or failure probability is low the replicated task are not needed.

In our future work, we would like to consider task migration inside spot instances to provide fault-tolerance. We are interested to investigate efficient failure prediction methods for Spot Instance along with multiple Spot Instance types.

REFERENCES

[1] Amazon EC2 Spot Instances. http://aws.amazon.com/ec2/spot-instances/, 2009.
[2]R.N. Calheiros, R. Ranjan, A. Beloglazov, C.A.F. De Rose, and R. Buyya. Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Software: Practice and Experience, 41(1):23–50, 2011.
[3]Anju Bala , Inderveer Chana 'Fault Tolerance- Challenges, Techniques and Implementation in Cloud Computing', 2012.
[4]Golam Moktader Nayeem, Mohammad Jahangir Alam," Analysis of Different Software Fault Tolerance Techniques", 2006.
[5]Chase.J.S, Anderson.D.C, Thakar.P.N, Vahdat.A.M and Doyle.R.P (2001) 'Managing energy and server resources in hosting centers', Proceedings of the 18th ACM Symposium on Operating Systems Principles, ACM, New York, USA, pp. 103–116.
[6]Antonina Litvinova, Christian Engelmann and Stephen L. Scott, 'A Proactive Fault Tolerance Framework for High Performance Computing', 2009.
[7]NM.Armbrust, A.Fox, R. Griffit,et al., 'A view of cloud computing', Communications of the ACM, vol. 53, no. 4, pp. 50–58, 2010.
[8]S. Ostermann and R. Prodan. Impact of variable priced cloud resources on scientific workflow scheduling. In Parallel Processing Euro-Par, volume 7484. Springer, 2012.