International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Impact Factor (2012): 3.358

Efficient Virtual Pre-emption based on Local and External Requests in Cloud Computing

Dhara J.Patel[1], Bakul Panchal[2]

[1]Computer Department, LD College of Engineering, Gujarat Technological University

[2]Assistant Professor, Computer Department, LD College of Engineering, Gujarat Technological University

Abstract: ***Resource provisioning is one of the main challenges in large-scale resource sharing environments such as federated Grids. In resource sharing environments resource providers serve requests from external users along with their own local users. The problem arises when there is not sufficient resources for local users, who have higher priority than external ones, and need resources urgently. This problem could be solved by pre-empting leases from external users and allocating them to the local ones. However, pre-empting leases entails side-effects in terms of overhead time as well as increasing makespan of external requests. In our proposed work, we model the overhead of the pre-empting vms and calculate the number of leases to be pre-empted by proposing an efficient algorithm that considers various scenarios based on the type of the local and external requests and selects the appropriate request to be pre-empted in a manner that the more local requests are served and also reduces the rejection ratio of highly prioritized Requests.Thereby , providing efficient resource provisioning by providing prioritized queues one for each type of request.***

Keywords: Cloud computing, Pre-emption, Local Request, External Request, Cloudsim

1. Introduction

Managing and providing computational resources for user applications is called as resource provisioning. It is one of the challenges in the high performance computing community. Resource sharing environments enable sharing, selection, and Update aggregation of resources across several Resource Providers (RP), also known as sites that are connected through high bandwidth network connections. In a resource sharing environment computational resources in each RP are shared between external users as well as local users of the RP. Recently, Virtual Machine (VM) technology has been employed for resource provisioning in many resource sharing environments [1-3].

Resource provisioning and sharing is based on the lease abstraction. A lease is an agreement between a resource provider and a resource consumer whereby the provider agrees to allocate resources to the consumer according to the lease terms presented by the consumer [3]. Virtual machine technology has been used to implement lease-based resource provisioning [3]. The capabilities of VMs in getting suspended, resumed, stopped, or even migrated (when there is enough bandwidth) have been extensively studied and have shown to be useful in resource provisioning without major utilization loss. It makes one lease for each user VM request. In resource sharing environments resource providers serve requests from external users along with their own local users. The problem arises when there is not sufficient resources for local users, who have higher priority than external ones, and need resources urgently.

This problem could be solved by pre-empting leases from external users and allocating them to the local ones. However, pre-empting leases entails side-effects in terms of overhead time as well as increasing make span of external requests.

2. Related Work

In this context we have discussed some related work, which introduces and analyzes various Lease Pre-emption algorithms based on various parameters.

a) Sotomayer et al[3]
   The overhead time imposed for suspending and resuming a VM-based lease is estimated. The proposed model is based on the amount of memory that should be de-allocated. Nevertheless, they have not considered situation where there is communication between VMs of a lease.
b) Haizea[4]
   Haizea operates based on the duration of the pre-emptable leases. In other words, it pre-empts leases that require more time to be completed. It cannot determine the optimal candidate set for pre-emption without any prior knowledge or any assumption about leases' durations.
c) Walters et al[2]
   Its policy of weighted summation of several factors such as the time spent in the queue
d) Snell et al[5]
   They do not emphasize more on VM based leases. They concentrate on the impact of pre-emption on current .requests but not advance- reservation requests waiting in the queue. Also they kill pre-empted requests to make the overhead zero but computational power is wasted in that case.

3. Existing Pre-Emption Policies

In this paper various Pre-emption policies are analyzed and explained. In which different policies use different parameters for the basis of their algorithm. One algorithm uses minimum number of lease as a parameter, other uses minimum overhead as a criteria another uses both. They are compared for knowing advantages and disadvantages of them

Parameters/ Algorithm	*Technique*	*Advantages*	*Limitations*
Minimum Overhead Policy(MOV)[6]	Based on Minimum time Overhead	Maximum resource utilization	More Resource contention and optimal set of leases is not obtained
Minimum Leases Involved Policy(MOV)[6]	Based on Minimum number of leases	Minimum Resource contention and optimal set of leases is obtained	Maximum resource utilization is not obtained
Minimum Overhead Minimum Leases Policy(MOML) [6]	Based on both Minimum time Overhead and Minimum number of leases	Maximum resource utilization and Minimum Resource contention	Does not consider inter-dependable leases and cancellable , suspendable, migratable types of local requests

4. Problems with Existing System

Many improvements to Pre-emption algorithm are done so far. But ,We require an efficient provisioning algorithm that calculates the minimum time overhead to increase the resource utilization and finds out the optimal set of leases to minimize contention of resources. The existing MOML algorithm provides a trade - off between resource utilization and minimum resource contention. However along with increasing the ratio of local requests being served we need to also focus on minimizing the external rejection ratio. The present algorithm considers only one type of local request deadline-constraint non – preemptable. But practically, local request can be of other types as cancellable, migratable etc. Killing of external requests to minimize the makespan or starving situation means wastage of computational power.

5. Proposed System

The proposed algorithm works on all four types of local requests and external requests i.e cancellable, suspendable, migratable, and non- migratable . It takes into account both the type of requests before finding an optimal situation for the pre-emption. In our work, we consider various scenarios on the basis of both the local request as well as external request and analyse which request to be pre-empted by allocating priority to each request and comparing the priorities to determine the low – priority request to be pre-empted. The proposed algorithm is more intelligent and scenario based i.e. selects an optimal solution based on different scenarios and different types of request

In this work, instead of using a single queue for waiting external requests we maintain separate prioritized queues, one for each type of waiting external requests and for each type of local waiting local requests. Even though the local request queue gets a higher priority every time a resource is free for utilization, it does provide efficient solution to starvation problem of external requests by applying them to priority based scheduling. Also by maintaining fixed length queues and transferring priority to the queue that is full, we prevent monopolizing resources.

Eight separate queues are created one for each type of requests and given a priority as shown below(1 being the highest and 8 being the lowest).

Priority	*Priority Queue*
1	Local Non-Migratable
2	Local Migratable
3	External Non-Migratable
4	External Migratable
5	Local Suspendable
6	Local Cancellable
7	External Suspendable
8	External Cancellable

6. Proposed Algorithm

Input: Local Request/ External Request
Output: Request to be pre-empted and added to the queue

Step1: Check the priority of the incoming Request based on its request type.
Step2: Check the priority of the executing Request. .(In case of no executing request assign i.e. the Vm being idle, skip step 2-6).
Step3: Compare the priority of the Incoming Request and the Executing Request.
Step4: Pre-empt the request with lower priority and add the request to the respective queue.
Step5: If the queue is full, the lower priority request is not pre-empted but keeps on executing and the higher priority request is added to the queue.
Step6: Calculate number of leases per Vm.
Step7: Assign the higher priority request to the Vm with the minimum number of leases.

7. Experimental Results

In this section we provide the detailed results of the experiments carried out using the proposed framework..The following experimental analysis is carriedout using CloudSim. Before execution the requests arrive at a respective Vm randomly, the priorities are assigned from the type of given requestand assigned to its respective queue as shown below leases are calculated by no of requests executing on single Vm.

Cid	Sid	P	Did	n	L/E	Request Type
1	5	1	5	1	L	Non-Migratable
2	5	7	4	1	E	Migratable
3	1	3	3	1	E	Non-Migratable
4	3	6	4	2	E	Cancellable
5	3	8	3	2	L	Cancellable

Where, Cid – cloudlet id,
Sid – sourceVmid,
Did - DestinationVmid,
P - Priority,
n – no of leases
L – Local Request
E – External Request

The cloudlets are executed based on the priority assigned to each Request. In table below, as we can see the order of execution differs from the order in which requests arrive.The

cloudlet- 2 is executed first and cloudlet-4 is executed later on Vm- 4, as cloudlet- 2 is an External Migratable type of Request and has higher priority while Cloudlet-4 is External cancellable type of Request

Cid	Sid	P	Did	n	L/E	Request Types
1	5	1	5	1	L	Non-Migratable
3	1	3	3	1	E	Non- Migratable
2	5	7	4	1	E	Migratable
5	3	8	3	2	L	Cancellable
4	3	6	4	2	E	Cancellable

8. Conclusion

After understanding various Pre-emption algorithms, we understood that algorithms consider only one type of external request, i.e. **deadline – constraint non- migratable**. But we need to consider other types of local requests i.e **suspendable, cancellable, migratable** for a more practical and intelligent approach. Also the introduction of 8 separate priority based queues simplifies the scheduling process. Also monopolization of resources is prevented by creating fixed length queues. So we have performed experimental analysis on various scenarios regarding different types of local and external request and reach to the conclusion that provides more intelligent resource utilization and minimum resource contention efficiently.

References

[1] F. Hermenier, A. L`ebre, J. Menaud, Cluster-wide context switch of virtualized jobs, in: Proceedings of the 19thACM International Symposium on High Performance Distributed Computing (HPDC ’10), USA, 2010.

[2] J. Walters, B. Bantwal, V. Chaudhary, Enabling interactive jobs in virtualized data centers, Cloud Computing and Applications 1 (2008) 21–26.

[3] B. Sotomayor, R. S. Montero, I. M. Llorente, I. Foster, Resource leasing and the art of suspending virtual machines, in: Proceedings of the 11th IEEE International Conference on High Performance Computing and Communications,USA, 2009

[4] B. Sotomayor, K. Keahey, I. Foster, Combining batch execution and leasing using virtual machines, in: Proceedings of the 17th International Symposium on High Performance Distributed Computing, ACM, USA, 2008.

[5] Q. Snell, M. J. Clement, D. B. Jackson, Preemption based backfill, in: Job Scheduling Strategies for Parallel Processing (JSSPP ’02), Springer, 2002.

[6] Rajkumar Buyya, Mohsen Amini Salehi, Bahman Javadi, Concurrency and Computation: Practice and Experience,ISSN: 1532-0626, Wiley Press, New York, USA (in press, accepted on Jan. 5, 2013).

[7] M. Amini Salehi, B. Javadi, R. Buyya, Resource provisioning based on leases preemption in InterGrid, in: Proceeding of the 34th Australasian Computer Science Conference (ACSC’11), Australia, 2011

[8] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, M. Zaharia, A view of cloud computing, Communications of the ACM 53 (4) (2010) .

[9] B. Sharma, R. K. Thulasiram, P. Thulasiraman, S. K. Garg, R. Buyya, Pricing cloud compute commodities: A novel financial economic model, in: Proceedings of the 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, CCGRID ’12, IEEE Computer Society, 2012.