ISSN:2348-2079

Volume-5 Issue-2

International Journal of Intellectual Advancements and Research in Engineering Computations

Burstiness of virtual machine resource reservation in Cling firefly optimizationoud us

JasminePersi T[1] Department of CSE Sasurie college of engineering jasminepersijas@gmail.com
[2]
Rajasekaran P
Assistan professor Department of CSE rajasekaranp91@gmail.com

ABSTRACT

In computing clouds, burstiness of a virtual machine (VM) workload widely exists in real applications, where spikes usually occur aperiodically with low frequency and short duration. This could be effectively handled through dynamically scaling up/down in a virtualization-based computing cloud; however, to minimize energy consumption, VMs are often highly consolidated with the minimum number of physical machines (PMs) used. In this case, to meet the dynamic runtime resource demands of VMs in a PM, some VMs haveto be migrated to some other PMs, which may cause potential performance degradation. In this paper, we investigate the burstinessaware server consolidation problem from the perspective of resource reservation.

1. INTRODUCTION

CLOUD computing has been gaining more and more traction in the past few years, and it is changing the way we access and retrieve information [1]. The recent emergence of virtual desktop [2] has further elevated the importance of computing clouds. As a crucial technique in modern

clouds, virtualization enables one physical machine (PM) to host many performance isolated virtual machines (VMs). It greatly benefits a computing cloud where VMs running various applications are aggregated together to improve resource utilization. It has been shown in previous work [3] that, the cost of energy consumption, *e.g.*, power supply,making optimal energy consumption in clouds, appropriately placed, especially in a highly consolidated cloud.Live migration moves some VM(s)to a relatively idle PM,when local resizing is not able to allocate enough resources.

and cooling, occupies a significant fraction of the total oper
server consolidation is proposed to tightly pack VMs to reduce the numbe

2. RELATED WORK

Most of prior studies [3], [15], [16] on server consolidation focused on minimizing the number of active PMs from the perspective of bin packing. A heterogeneityaware resource management system for dynamic capacity provisioning in clouds was developed in [17]. Stable resource allocation in geographically-distributed clouds was considered in [18].

Network-aware virtual machine placement was considered in [19]. Scalable virtual network models were designed in [8], [20] to allow cloud tenants to explicitly specify computing and networking requirements to achieve predictable performance.In a computing cloud, burstiness of workload widely exists in real applications, which becomes an inevitable characteristic in server consolidation [1], [4], [6], [7], [21].Some recent works [22], [23] used stochastic bin-packing (SBP) techniques to deal with variable workloads, where workload is modeled as random variable. Some other research [10], [24], [25] studied the SBP problem assuming VM workload follows normal distribution. Several other studies [26], [27] focused on workload prediction while the application runs. SDifferent from them, in our model a lower limit of provisioning is set at the normal workload level which effectively prevents VM interference caused by unpredictable behaviors from co-located VMs.Markov chain was used to inject burstiness into a traditional benchmark in [7]. Several works [5], [28], [29] studied modeling and dynamic provisioning of bursty workload in cloud computing. A previous study [30] proposed to reserve a constant level of hardware resource on each PM to tolerate workload fluctuation; but how much resource should be reserved was not given.To the best of our knowledge, we are the first to quantify the amount of reserved resources with consideration on various, but distinct, workload burstiness.

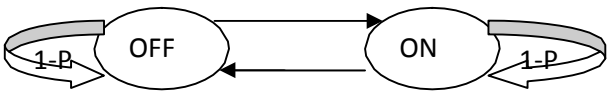

Fig. 1. Two-state Markov chain. The “ON” state represents peak workload (*Rp*) while the “OFF” state represents normal workload (*Rb*). *pon* and *poff* are the state switch probabilities.

3 MODELING VIRTUAL MACHINE WORKLOAD

Two-state Markov Chain

It has been well recognized in previous studies [4], [6],[7] that VM workload is time-varying with bursty

spikes,as shown in Fig. 1. Several works [9], [10], [22], [23],[24], [25] modeled the workload of a VM as a random variable, which follows the Bernoulli distribution in [9] or normal distribution in [10], [24], [25]. Different from these works, we model the workload of a VM as a twostate Markov chain, which takes the additional dimension of time into consideration, and thus describes the characteristics of spikes more precisely.Fig. 1 shows an example. We denote the resource requirements of peak workload, normal workload, and workload spike by *Rp*, *Rb*, and *Re*, respectively, where *Re* = *Rp* - *Rb* as demonstrated in Fig. 1. The "ON"state represents peak workload while the "OFF" staterepresents normal workload. We use *pon* and *poff* to denote the state switch probabilities. More specifically,if a VM is in the ON state, then the probability of it switching to OFF at the next time is *poff* , and remaining ON is 1 - *poff* . Similarly if a VM is in the OFF state,then the probability of it switching to ON at next time is *pon* and remaining ON is 1-*pon*. We emphasize that this model is able to describe the characteristics of spikesprecisely—intuitively, *Re* denotes the size of a spike, and *pon* denotes the frequency of spike occurrence. Thus,each VM can be described by a four-tuple.

Vi = (*pi on; pi off ;Rib;Rie*); *1<i<n,*

where *n* is the number of VMs.

Learning Model Parameters

This subsection provides a simple strategy for cloud tenants to generate model parameters for their VM workload. It consists of two phases.First, a cloud tenant must have the workload traces and guarantees that they will be consistent with the realistic deployment in computing clouds. This could be achieved by tentatively deploying VMs in a cloud for a relatively short period; the cloud system collects the resource usage traces and feeds them back to tenants.Second, given a VM workload trace, a cloud tenant generates a four-tuple. where the solid black curve represents the workload over time. Given the predetermined *Rb* and *Re*, we conservatively round the solid black curve up to the dashed red curve. Denote by *W*(*t*) the workload at time *t* in the dashed red curve.

Potential Benefits

The 2-state Markov chain model allows cloud tenants to flexibly control the tradeoff between VM performance and deployment cost through adjusting *Rb* and *Re.When a tenant wants to maximize VM*

performance, the tenant should choose a large Rb and a small Re. As we will show later in this paper, there may be multiple workload spikes that share some common physical resources.Thus, when the aggregated amount of workload spikes that simultaneously occur is larger than the amount of the shared common resources, capacity overflow happens and VM performance is probably affected.

When a tenant wants to minimize deployment cost, the tenant should choose a small Rb and a large Re. By ”deployment cost”, we mean the fee which is paid by a cloud tenant to a cloud provider. Since physical resources are opportunistically shared among multiple workload spikes, the charge for workload spike should be smaller than that for normal workload [9]. Therefore, decreasing *Rb* helps tenants to reduce the deployment cost.Our model is also a tradeoff between modeling complexity and precision. We could model time-varying workload by 3-state or even more states of Markov chain, which should capture the workload bustiness more precisely; however, the complexity in learning model parameters and allocating physical resources increases as well, which may complicate the interaction between cloud providers and tenants.

4 PROBLEM FORMULATION

We consider a computing cloud with 1-dimensional resource;for scenarios with multi-dimensional resources,we provide a few remarks in Section 8. There are *m* physical machines in the computing cloud, and each PM is described by its physical capacity

$$Hj = (Cj), 1 < j < m.$$

We use a binary matrix $\mathbf{X} = [xij\ ]n\times m$ to represent the results of placing *n* VMs on *m* PMs: $xij = 1$, if *Vi* is placed on *Hj* , and 0 otherwise. We assume that the workloads of VMs are mutually independent. Let $Wi(t)$ be the resource requirements of *Vi* at time *t*. According the Markov chain model, we have

$W_i(t) = Ri\ b$ if vi is in the “OFF” state at time t.
$W_i(t) = Ri\ p$ if vi is in the “ON” state at time t.

We now can define our metric for probabilistic performance guarantee—*capacity overflow ratio* (COR), which is the fraction of time that the aggregated workloads of a PM exceed its physical capacity.

5 A PROPOSED SYSTEM FOR BURSTINESS-AWARE RESOURCE RESERVATION

In this section, we first present the main idea of our solution to the BASC

problem, then we design a resource reservation strategy for a single PM, based on which we develop QUEUE, a complete server consolidation algorithm. In the end, we provide a concrete example to help readers better understand our algorithm.

Overview of QUEUE

We propose reserving a certain amount of physical resources
on each PM to accommodate workload spikes.The main idea is to abstract the reserved spaces as blocks (*i.e.*, serving windows in queueing theory). We give an informal illustration of the evolution process of our queueing system.

Resource Reservation Strategy for a Single PM

In this subsection, We focus on resource reservation for a single PM. For the sake of convenience, we set the size of each block as the size of the maximum spike of all co-located VMs on a PM..We also assume that all VMs have the same state switch probabilities, *i.e.*, $p_{i\,on} = p_{on}$ and $p_{i\,off} = p_{off}$, for all $1<i< n$.we will show how to cluster VMs when they have different state switch probabilities.Suppose there are k VMs on the PM of interest and initially each VM V_i occupies R_{ib} resources. We initialize the number of blocks reserved on this PM as k, and our objective is to reduce the number of blocks to K ($K < k$),while the capacity overflow ratio does not exceed the threshold (*i.e.*, VMs that leave the queueing system).

QUEUE

In this subsection, we present the complete server consolidation algorithm, QUEUE, which is to place a set of n VMs onto a set of m PMs.As we mentioned before, weconservatively set thesize of a single block as the size of the maximum spike of all the VMs on each PM, which may result in low utilization if the workload spikes of the co-located VMs differ too much. Therefore, in QUEUE, we tend to place VMs with similar R_e 's on the same PM in an effort to reduce the average size of a single block.

LMM Algorithm

This new method called Local Minimum Migration in Cloud (LMM).This method monitors all the virtual machines in all cloud locations centre. It identifies the state virtual machines whether it is sleep, idle, or running (less or over) state. And also checks number of tasks running and number of tasks can able to run. When a virtual

machine is considered to be overloaded requiring live migration to one or more VMs from the nearby location cloud centres. When a virtual machine is considered to be under loaded (minimum task running) requiring live migration to one or more VMs (which are already running to perform some jobs) from the nearby location cloud centres. Location based VM selection policy (algorithm) has to be applied to carry out the selection process. The main contributions are shown below, Migration of Virtual machine tasks where less number of tasks are running, which occupies extra energy and power consumption. Migration of Virtual machine tasks where more number tasks are running, which gives overhead to the system performance. Less number of migrations should be selected, to avoid more processing cost for migration process. Migration of Virtual machine tasks to reduce the user provisioning cost, by finishing the user's tasks in their deadline.

5 Server Consolidation

In this subsection, we present the complete server consolidation algorithm, QUEUE, which is to place a set of n VMs onto a set of m PMs. As we mentioned before, we conservatively set the size of a single block as the size of the maximum spike of all the VMs on each PM, which may result in low utilization if the workload spikes of the co-located VMs differ too much. Therefore, in QUEUE, we tend to place VMs with similar Re′s on the same PM in an effort to reduce the average size of a single block. Algorithm of QUEUE, which consists of three phases. In the preprocessing phase , we introduce an array named MinN, which stores the information about the minimum number of blocks that need to be reserved on a PM, given k, pon, pof f , and ρ. That is, if there are k VMs placed on a PM, then we need to reserve MinN[k] blocks to bound the capacity overflow ratio. Without loss of generality, we assume that a single PM can host up to d VMs, thus we can calculate the array MinN[k] for all possible $k \in [1, d]$ before VM placement. We also let MinN[0] = 0 for compatibility. We notice that, for the same k, when the capacity overflow ratio threshold ρ increases (from green circles to red triangles), MinN[k] decreases; when pon increases (from green circles to blue squares), MinN[k] increases. These observations are consistent with our intuitions. During the sorting phase, we first cluster all VMs so that VMs with similar Re′s are in the same cluster1, and then sort

these clusters in the descending order of Re. In each cluster, we sort VMs in the descending order of Rb. We also sorted PMs in the descending order of the physical capacity C. This is a cluster-level heuristic to let co-located VMs have similar Re′s, thus to minimize the average size of blocks on all PMs. In the third phase, we adopt the First FitDecrease (FFD) heuristic to place VMs on PMs.

6 DISCUSSIONS

The bioluminescence processes is responsible for flashing light of fireflies. There are many theories regarding reason and importance of flashing light in firefly's life cycle but many of them converge to mating phase. The basic objective of flashing light is to attract mating partner The pattern of these rhythmic flashes is unique based upon rhythm of flashes, rate of flashing and amount of time for which flashes are observed. This pattern attract both the males and females to each other and female of a species respond to individual pattern of male of same species . According to the inverse square law, intensity of the light I, keeps on decreasing as the distance r increases in terms of I α 1/r2. Air also acts as absorbent and light gets weaker with increasing distance. Combining these two factors reduce the visibility of fireflies to a limited distance normally few hundred meters at night which is sufficient for fireflies to communicate with each other.

Firefly algorithm is based upon idealizing the flashing characteristic of fireflies. The idealized three rules are:- International Journal of Computer Applications.

• All fireflies are considered as unisex and irrespective of the sex one firefly is attracted to other fireflies

•The Attractiveness is proportional to their brightness, which means for any two flashing fireflies, the movement of firefly is from less bright towards the brighter one and if no one is brighter than other it will move randomly. Furthermore they both decrease as their distance increases.

• The landscape of the objective function directly affects the brightness of the firefly. For a maximization problem, the brightness is proportional to the objective function's value. Other forms of the brightness could be defined in same way to the fitness function as in genetic algorithms.

7 CONCLUSION

Burstiness is a common pattern of VM workload in production data centers, and server consolidation also plays an important role in cloud computing. Both

topics have been studied extensively for years, but no work explicitly takes 340 workload burstiness into consideration during the consolidation process. On the other hand, in a highly consolidated cloud VM performance is prone to degradation without appropriate VM placement if distinct burstiness exists. To alleviate this problem we have to use more PMs than needed which leads to more energy consumption. To balance the performance and energy consumption with respect to bursty workload, certain amount of resources need to be reserved on the PM to accommodate burstiness, and to quantify the amount of reserved resources is not a trivial work. In this paper we propose a burstiness-aware VM consolidation scheme based on the two-state Markov chain. We formulate the performance constraint and show that our proposed algorithm is able to guarantee this performance constraint. The experiment result show that our algorithms improve the consolidation ratio by up to 45% with large spike size and around 30% with normal spike size compared to those provisioning for peak workload, and a better balance of performance and energy consumption is achieved in comparison with other commonly used consolidation schemes

REFERENCES

[1] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica *et al.*, "A view of cloud computing," *Communications of the ACM*, vol. 53, no. 4, pp. 50–58, 2010.

[2] M.-H. Oh, S.-W. Kim, D.-W. Kim, and S.-W. Kim, "Method and architecture for virtual desktop service," 2013, US Patent 20,130,007,737.

[3] M. Marzolla, O. Babaoglu, and F. Panzieri, "Server consolidation in clouds through gossiping," in *Proc. of IEEE WoWMoM 2011*, pp. 1–6.

[4] W. Vogels, "Beyond server consolidation," *ACM Queue*, vol. 6, no. 1, pp. 20–26, 2008.

[5] N. Bobroff, A. Kochut, and K. Beaty, "Dynamic placement of virtual machines for managing sla violations," in *Proc. of IFIP/IEEE IM 2007*, pp. 119–128.

[6] S. Kandula, S. Sengupta, A. Greenberg, P. Patel, and R. Chaiken, "The nature of data center traffic: measurements & analysis," in *Proc. of ACM IMC 2009*, pp. 202–208.

[7] N. Mi, G. Casale, L. Cherkasova, and E. Smirni, "Injecting realistic burstiness to a traditional client-server benchmark," in *Proc. Of IEEE ICAC 2009*, pp. 149–158.

[8] D. Xie, N. Ding, Y. C. Hu, and R. Kompella, "The only constant is change:

incorporating time-varying network reservations in data centers," in *ACM SIGCOMM 2012*, pp. 199–210.

[9] S. Zhang, Z. Qian, J. Wu, S. Lu, and L. Epstein, "Virtual network embedding with opportunistic resource sharing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 3, pp. 816–827, March 2014.

[10] M. Wang, X. Meng, and L. Zhang, "Consolidating virtual machines with dynamic bandwidth demand in data centers," in *Proc. of IEEE INFOCOM 2011*, pp. 71–75.

[11] A. Verma, G. Kumar, and R. Koller, "The cost of reconfiguration in a cloud," in *Proc. of ACM Middleware 2010*, pp. 11–16.

[12] W. Voorsluys, J. Broberg, S. Venugopal, and R. Buyya, "Cost of virtual machine live migrationin clouds: A performance evaluation," in *Proc. of IEEE CloudCom 2009*, pp. 254–265.

[13] Z. Luo and Z. Qian, "Burstiness-aware server consolidation via queuing theory approach in a computing cloud," in *Proc. of IEEE IPDPS 2013*, pp. 332–341.

[14] S. M. Ross, *Introduction to Probability Models, Tenth Edition*. Orlando, FL, USA: Academic Press, Inc., 2011.