Live Consolidation for Data Centers in Cloud Environment

Chuan Pham
Department of Computer Engineering, KyungHee University, Korea
pchuan@khu.ac.kr

Nguyen H.Tran
Department of Computer Engineering, KyungHee University, Korea
nguyenth@khu.ac.kr

Cuong T. Do
Department of Computer Engineering, KyungHee University, Korea
dtcuong@khu.ac.kr

Choong Seon Hong*
Department of Computer Engineering, KyungHee University, Korea
cshong@khu.ac.kr

ABSTRACT

In this paper, we consider two critical issues in data centers: load balancing and server consolidation. We propose a live consolidation method, which identifies an optimal strategy of hosting a new virtual machine (VM). Our algorithm finds the optimal allocation scheme to host a VM in term of load balancing, reducing the migration cost, and mitigating resource redundancy. Based on queueing and optimization theory, we model the allocation resource in data centers and analyze how to choose the location for each VM, and also minimize the number of active servers. The live consolidate algorithm exploits the distribution VM process and determines the optimal active server set. The numerical analysis demonstrates that our algorithm is able to consolidate the system lively. By using a real dataset, our simulation results show that the optimal distribution probability vector can allocate a new VM into a appropriate server to improve the resource utilization and the energy efficiency.

Categories and Subject Descriptors

C.2.3 [**Network Operations**]: Resoure management.

General Terms

Algorithms, Management.

Keywords

Consolidation, live migration, queueing theory, optimization theory, VM placement problem.

1. INTRODUCTION

*Corresponding author.

IMCOM (ICUIMC)'15, January 08 - 10 2015, BALI, Indonesia

In cloud computing environment, power consumption is a critical factor for data centers. Effective power consumption control can improve significantly benefit for providers. Many data centers and cloud providers are facing up to high energy costs, and a discouraging low server utilization. In the survey of Barroso et al. [1], most of the time during six months, the CPU utilization of 5000 Google servers is from 10-15%. Keeping underutilized servers is still consume about 70% of their peak power [2]. It means that they are most inefficient in terms of energy.

And in consequences, using virtualization technology to improve the energy efficiency is one of the ways that many researchers are focusing on. The virtualization technology allows cloud providers to create multiple Virtual Machines (VMs) on a single physical machine (called PM or server) and use live migration to consolidate system. By this way, data centers can improve the utilization of resource and reduce energy consumption by switching nodes to lower modes (i.e., sleep, hibernation) [3]. However, efficient resource management in clouds is not trivial and this is still a potential research area.

In cloud computing literature, consolidation system plays an important role in reducing the power consumption and stabilize whole system. However, implementing consolidation techniques are still far from standard in data centers due to a number of challenges. One of challenges is live migration problem. The live migration function is applied in moving VMs from a server to others for providing additional resources or releasing cold spot (i.e., an underloaded host) or hot spot (i.e., an overloaded host) [4, 5, 6, 7]. However, the copy process in the migration requires more time, resources on the source and destination servers (such as: CPU, memory, storage, bandwidth, etc.) to perform. In some special conditions, a live migration causes a dramatic reduction in the amount of available resources that can easily lead to rejection of new user requests due to lack of available physical slots [8]. Usually, in cloud computing, many researches use a slot to represent one basic resource unit, including CPU, memory, disk, etc., each slot can host one VM and each server has finite slots depend on the server configuration. Tenants submit their resource requirements, in term of number of VM (slots), to cloud system, and the cloud decides the destination physical servers to host these slots. When all slots of a server are full, this server cannot host

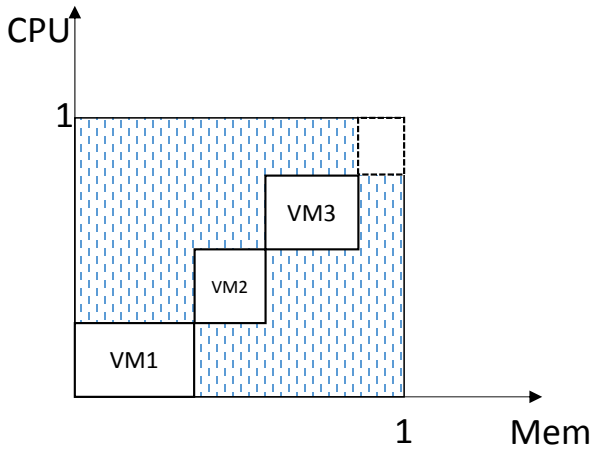

Figure 1: The VM location in VM Placement problem.

any VM.

Therefore, in this paper, we propose a novel manner "live consolidation". It means that whenever creating a new VM, we try to consolidate each server to reduce the cost of migration and resource fragmentation. We analyze the allocation resource in data centers by queueing model and propose an optimal algorithm that runs iteratively to calculate an optimal scheme in term of reducing number of active servers.

In recent solutions of server consolidation, many papers consider this problem is the same as VM placement problem. They often formulate their model as a bin packing problem [9], which was solved by using the First fit algorithm, Best fit algorithm, Exact fit algorithm, etc. However, the VM placement problem is exactly difference from bin packing problem [10]. While packing objects into a given container, bin packing algorithm does not consider about the server load balancing issue and the server consolidation problem in the system. It only tries to minimize or maximize the objective function (e.g., minimize number of container, maximize the total profit, maximize number of items, etc). The Figure 1 shows the allocation scheme in VM Placement problem. Resource on physical servers are allocated along multiple dimensions according to the resource demands of VM requests.

In this paper, we propose an algorithm to determine the VM allocation scheme to minimize the total utility of active servers and dynamically control number of active servers. Based on queueing theory, an optimal distribution probability vector is calculated to obtain the most efficiency number of active servers. By this way, our proposed algorithm can allocate VMs into the best hosts in data center in term of consolidation. Our main contributions are summarized as follows:

1. We formulate the VM allocation scheme in data center by M/M/k/k model [11]. From the queuing model, we illustrate the transition state of each server in data centers. This analysis shows the probability of idle and busy status of a server when hosting VM. Based on the blocking model Erlang B [11], we calculate the blocking probability of each server. This is an important variable in our algorithm to determine when a new active server should be turned on. Moreover, we define two thresholds of the overloaded probability and the underloaded utility. These thresholds are used in the algorithm to adjust the efficiency of controlling number of active servers to serve all requests in data centers.

2. Adapting to circumstance of user demands at each time, we propose a dynamic allocation algorithm based on calculating an optimal distribution probability vector. The proposed algorithm can consolidate directly when creating a new VM in term of reducing the migration cost and the power consumption.

3. Finally, we validate our live consolidation algorithm using the load trace from the web log file to evaluate the efficiency of saving energy consumption. We show that significant energy saving is possible by controlling the optimal distribution probability vector. In the simulation, we illustrate that the combination helps the system more stable and efficient in allocating resource. Furthermore, the simulation results show that the proposed algorithm can reduce the power consumption in data centers compare to the fixed power allocation.

The rest of the paper is organized as follows. The related works are reviewed in Section II. Section III presents the system model and problem statement. In section IV, V, we describe the theoretical analysis and solve the optimization problem. The live consolidation algorithm is represented in Section VI. Finally, we show performance results in Section VII and present conclusion in Section VIII.

2. RELATED WORKS

This paper is not alone in approaching the task of developing algorithms for consolidation system. Interest in consolidation data centers has been growing since [12, 13] appeared at the start of this area research. Chebiyyam et al. [14] discussed the motivation and benefits of server consolidation and research trends. But most of current works whether mitigates VM placement problem or only consider about optimal power consumption. Server consolidation is a VM placement problem, which is a NP-Hard [15]. Therefore, it is too hard to find a new allocation scheme not only guarantees Quality of Services (QoS) of whole system but also optimizes power consumption. In [4, 16], based on the dynamic workload, the authors defined an algorithm to find an optimal number of active servers. However, they did not consider about how to place VM in these active servers.

Related function in consolidation, live migration is a great approach of recent researches. In [17], the authors consider live migration as an effective tool to enable data center management in a non-disruptive manner. However, we should take into account two costs of migration: service downtime and total migrations time. These factors also make the consolidation cost increase. In our propose, we try to consolidate lively. It means that when creating a new VM, we consolidate the hosting server, and also balance resource usage.

3. SYSTEM MODEL AND PROBLEM STATEMENT

We consider a cloud scenario that a user requests a certain service, and the cloud provider creates a VM for that user. The data center consists of a set of physical machines

(PMs) that can host multiple VMs through a hypervisor. We assume that the number of physical machines is fixed and there is a set of active servers that is serving the requests from users. A server will not receive the VM request from the dispatcher, if it is not in the active set. It can switch off or turn to the sleep mode to save power consumption. Every active server has a finite number of slots to host VMs. A slot represents one basic resource unit, including CPU, memory, disk, etc., and each slot can host one VM. Tenants submit their resource requirements, in term of number of VM (slots), to cloud system, and the cloud decides the destination physical servers to host these slots. When all slots of a server are full, this server cannot host any VM. This capacity depends on the amount of resource in each server. The server cannot serve any request if all slots are allocated. We call that it is the blocking status on each server. Therefore, we suppose the behavior of each server as same as a loss system M/M/k/k.

When having a request to create a new VM, a dispatcher will choose a candidate server to host this VM. We assume that VM requests arrive to the data center according to a Poisson process. Therefore, the request arrival in each server is also the Poisson process. We consider a probability-based selection scheme that can evenly distribute the request to multiple servers. If all active servers are overloaded, a new PM will be turned on and added into the active set. If the average utilization of all active servers is lower than the under loaded threshold, the system will randomly remove one of active servers out of the active set. The removed server will return all remaining VMs to the dispatcher or continuously host them until they release. Thereby, we can improve the average utility of active servers. Without loss of generality, we consider that each PM has a finite capacity slots to host VMs. We assume that all VMs can dynamically distribute to their hosting servers with optimal probability that can balance the resource usage in multiple servers. The system model can be illustrated by Figure 2.

We now formalize in more detail the inner working of decision modules which dispatches VMs to servers. Let M is the number of PMs in data center and m ($m \leq M$) is the number of active PMs which are running to host VMs. In server i, we denote k_i is the finite capacity slots to host VMs. When all active servers are overloaded, a new PM will be turn on and add to the active set. Each VM can select one of M candidate servers for its operating server. The distribution probability vector $p = (p_1, p_2, ..., p_m)$ represents the set of probabilities of selecting all the candidate servers, in which p_i denotes the probability of selecting server i. We assume that VM requests arrive to the data centers according to a Poisson process as modeled in [8] with rate λ_s. Thus, the effective arrival rate of VM request at server i is $\lambda_i = p_i \lambda_s$. When a VM request arrive at the system, it can be directly connected to the selected server based on the predetermined distribution probability vector p. Then, we aim to find the optimal distribution probability vector p^* to minimize the utility of whole active servers. Hence, we have Problem I as follows:

$$\begin{aligned} p^* &= \arg\min_{\forall p>0} \sum_{i=1}^{m} p_i U_i \\ 0 &\leq p_i \leq 1, \forall i \\ \sum_{i=1}^{m} p_i &= 1 \end{aligned} \tag{1}$$

where U_i is the utility at server i. The expression of U_i will be discussed in the next section based on queuing model.

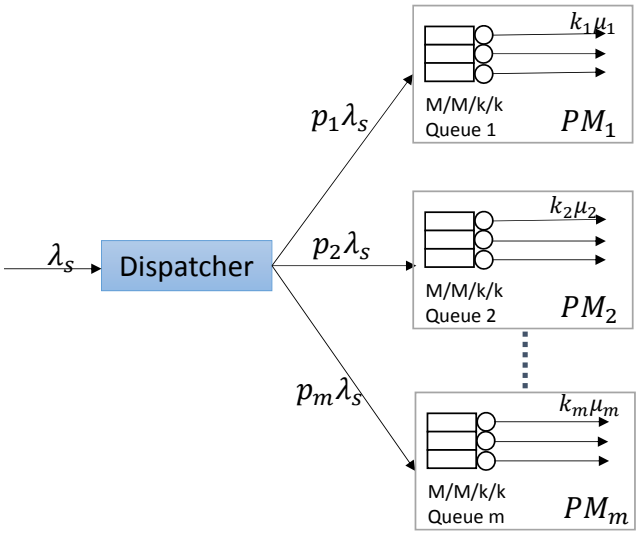

Figure 2: The probability-based server selection scheme.

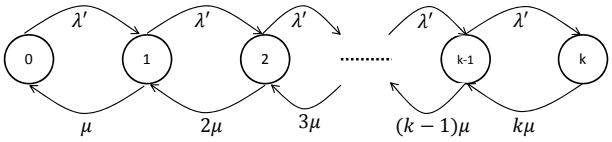

Figure 3: Transition diagram for each server.

4. THE ALLOCATION STATUS ANALYSIS

In this section, we apply M/M/k/k queueing model [11] to analyze the allocation scheme in each PM. Each PM can serve k VMs (i.e., k is the maximum number of VMs that can host at the same time in a server). The required time to process of a VM (i.e., the duration time from creating a VM until it shuts down) is modeled by an exponential distribution, with mean service time of $1/\mu$. The transition diagram, used to calculate the blocking probability, is given in Figure 3. In this figure, each state is represented the total number of occupied slots in the PM. When k slots are located, this is rejected state in this server. It means that this server cannot allocate resource for VM anymore. In this case, the request must be forwarded to another available server.

The Markov process in Figure 3 shows the process corresponding to the system evolution. The initial state (0) means that the server is free. And, the end state (k), when the server is block. In general, if the server is in state (j) and one VM leaves, the system will change to state (j-1) with rate μ. Otherwise, when a new VM arrives, the system state from (j) will change to (j+1) with rate λ'. Therefore, the steady-state condition of the system is $\lambda' - \mu < 0$. Let define $a = \frac{\lambda'}{\mu}$ and the utility of each server $U = \rho = \frac{a}{k}$. We

have a set of global balance equations as:

$$\begin{cases} \lambda' P_0 = \mu P_1 \\ (\lambda' + \mu) P_1 = \lambda' P_0 + 2\mu P_2 \\ ... \\ (\lambda' + (k-1)\mu) P_{k-1} = \lambda' P_{k-2} + k\mu P_k \\ k\mu P_k = \lambda' P_{k-1} \end{cases} \quad (2)$$

From the theoretical analysis of [11], we have the probability that an arrival will find all k slots busy and therefore will be forwarded to another available server is:

$$P_k = \frac{a^k}{k! \sum_{j=0}^{k} \frac{a^k}{k!}} \quad (3)$$

The blocking probability will be used by the live consolidation algorithm in Section VI. Now, based on the utility calculation of each server, we solve the optimal solution of the active set in the next section.

5. THE OPTIMAL UTILITY SOLUTION OF ACTIVE SERVERS

We return to solve the Problem I in the section III. The arrival rate of a VM and the utility at server i are:

$$\lambda_i = p_i \lambda_s \quad (4)$$

$$U_i = \frac{a_i}{k} = \frac{p_i \lambda_s}{k \mu_i} \quad (5)$$

Then, the objective function of Problem I can be rewritten as follows:

$$p* = \arg\min_{\forall p>0} \sum_{i=1}^{m} p_i U_i = \arg\min_{\forall \lambda>0} \frac{1}{\lambda_s} \sum_{i=1}^{m} \frac{(\lambda_i)^2}{k_i \mu_i} \quad (6)$$

Problem I can be expressed as Problem II:

$$\begin{aligned} &\min \sum_{i=1}^{m} \frac{(\lambda_i)^2}{k_i \mu_i} \\ &s.t. \sum_{i=1}^{m} \lambda_i = \lambda_s \\ &0 \le \lambda_i \le \mu_i, i = 1, 2, ..., m. \end{aligned} \quad (7)$$

By relaxing the final constraint of λ_i, the Lagrangian function of Problem II

$$L(\lambda, \alpha) = \sum_{i=1}^{m} \frac{(\lambda_i)^2}{k_i \mu_i} - \alpha(\sum_{i=1}^{m} \lambda_i - \lambda_s) \quad (8)$$

where α is Lagrangian multipliers and arrival rate vector λ $(\lambda_i, \lambda_2, ..., \lambda_m)$. We apply KKT condition [18] for the optimal solution (λ^*, α^*) as follows

$$\begin{aligned} &\lambda^* \ge 0 \\ &\sum_{i=1}^{m} \lambda_i^* = \lambda_s \\ &\tfrac{\partial L(\lambda^*, \alpha^*)}{\partial \lambda_i} = \tfrac{2\lambda_i^*}{k_i \mu_i} - \alpha^* = 0 \end{aligned} \quad (9)$$

By solving KKT condition, we obtain the optimal solution λ^* such as

$$\begin{aligned} \alpha^* &= \tfrac{2\lambda_i^*}{k_i \mu_i} \\ \lambda_i^* &= \tfrac{\alpha^* k_i \mu_i}{2} \end{aligned} \quad (10)$$

Based on the constraint of (7), we obtain

$$\begin{aligned} &\sum_{i=1}^{m} \tfrac{\alpha^* k_i \mu_i}{2} = \lambda_s \\ &\Rightarrow \begin{cases} \alpha^* = \frac{2\lambda}{\sum_{i=1}^{m} k_i \mu_i} \\ \lambda_i^* = \frac{\lambda k_i \mu_i}{\sum_{i=1}^{m} k_i \mu_i} \end{cases} \end{aligned} \quad (11)$$

By using projection method [18] on the constraint (6), the optimal value of λ_i^* can be obtained as follows

$$\lambda_i{}^{*(p)} = \begin{cases} 0 & \lambda_i^* \le 0 \\ \lambda_i^* & 0 \le \lambda_i^* \le \mu_i \\ \mu_i & \lambda_i^* \ge \mu_i \end{cases} \quad (12)$$

where $\lambda_i{}^{*(p)}$ is the optimal value after projecting on the range $[0, \mu_i]$. Then, the optimal vector p_i^* can be obtained by $p_i^* = \frac{\lambda_i{}^{*(p)}}{\lambda_s}$.

Based on the analysis on Section IV and the result in section V, we propose the live consolidation algorithm to load balance the VM request and reduce the power consumption in the next section.

6. LIVE CONSOLIDATION ALGORITHM

In traditional method, consolidation is a periodic task that is invoked after a period time to consolidate whole system. This way sometimes consumes a huge of resources by rearranging whole system. In this section, we present an algorithm that dispatches requests to active server set based on the optimal result in the previous section. It was named live consolidation algorithm. When having a request to create a new VM, the controller will calculate the probability $p_i^* = \frac{\lambda_i{}^{*(p)}}{\lambda_s}$, in which the optimal vector $(\lambda_1^*, \lambda_2^*, ..., \lambda_m^*)$ is calculated by (12). From the given arrival rate, the computation tries to minimize the total utility of active servers. If the average blocking probability of all active servers is higher than the threshold $P_{blocking}^{thr}$, the controller will turn on a new PM to serve user's requests. In other words, if the average utility of all servers is lower than underloaded threshold $U_{underload}^{thr}$, the controller will remove randomly a server out of the active set. Because our model loads balance request to all active servers. When the average utility of all servers is low, it means that all active servers also have low utility. The removed server will not receive the request from the dispatcher. It will be switched off when all VMs leave out. The threshold of blocking probability and the underloaded threshold can be adjusted by an administrator to appropriate any particular system. Since, when switching off a server, the provider must take into account the reconfiguration cost to turn it on. If the underloaded threshold is high, a server is easily removed out of the active set. Thus, we set these thresholds as follows:

$$\begin{cases} 0.6 \le P_{blocking}^{thr} \le 0.7 \\ 0.2 \le U_{underload}^{thr} \le 0.3 \end{cases} \quad (13)$$

The sequence diagram in Figure 4 shows the flow to distribute VMs to servers.

7. SIMULATION AND NUMERICAL RESULTS

Simulation environment: We consider a scenario where the system has maximum 10 servers and each server can

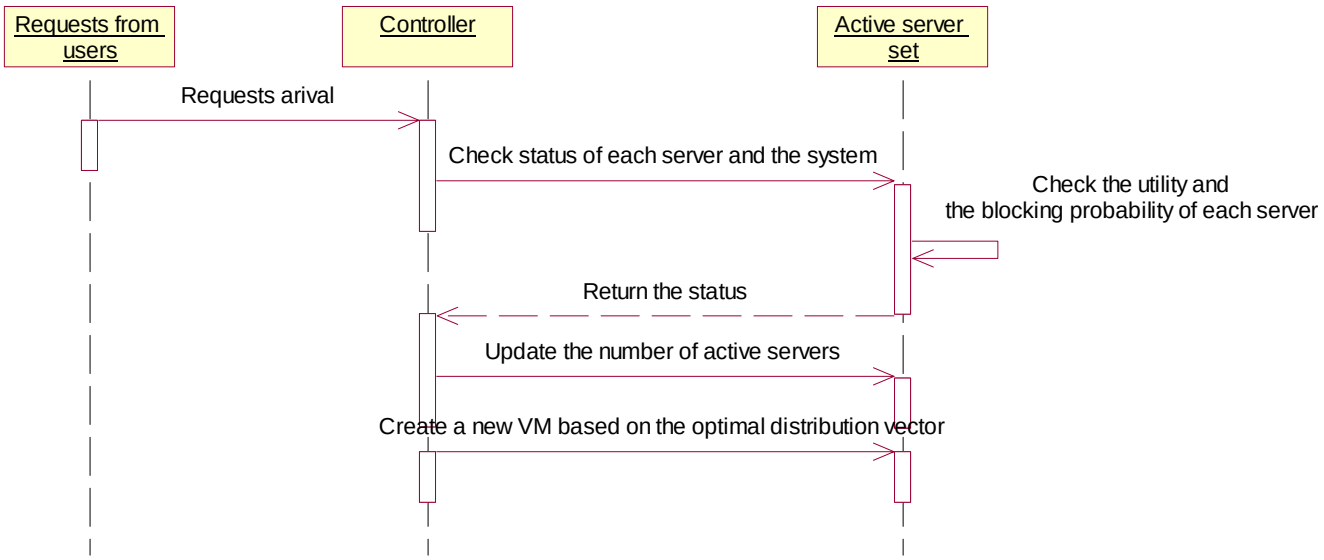

Figure 4: The sequence diagram for the live consolidation.

serve 10 VMs at the same time. At the first time, we start with 3 active servers to serve the requests from users. The parameters in Table 1 are used in our simulation.

From Figure 5, we can see that when the average arrival rate increases, the number of active servers also rises and vice versa. It also shows the capability of saving energy consumption in our model. When the arrival rate is low, the number of active servers is also low. Hence, we can use the statistic method to collect the average arrival rate in period time (e.g., using log files). For example, the average arrival rate on morning is different from noon or midnight. With difference arrival rate λ_s, we have difference probability vectors. The live consolidation algorithm can obtain more benefit from reduce power consumption.

Evaluating the power consumption: To evaluate our propose, we use log files from a web server, which traces from 8 servers and 50 VMs in average. From the log file, we obtain the average request rate and the average service time per hour in this tracking system. The log file is traced in 9 hours, starting at midnight on Monday July 1 2013. From 0 AM to 6 AM, the system has more underloaded servers than other periods. This time also is the best duration that we should consolidate the whole system. The Figure 6 shows the ability of controlling number of active servers depending on the requests of our model. We run our method with the log file in 10 hours. The number of active servers is in proportion to the requests from users.

From the simulation, we collect essential performance indicators, the percentage of released servers and effects of saving power. We assume that each server is set 200 watts at idle and 400 watts at maximum performance. To evaluate the effect in reducing power consumption, we use the formula [2]:

$$Pw = (Pw_{\max} - Pw_{idle}) * \overline{r}_i + Pw_{idle}, \qquad (14)$$

where Pw_{max} is the power consumption at maximum performance and Pw_{idle} is the power consumption at idle.

The Figure 7 shows the effect of saving energy consumption by using the proposed algorithm. Comparing with fixed

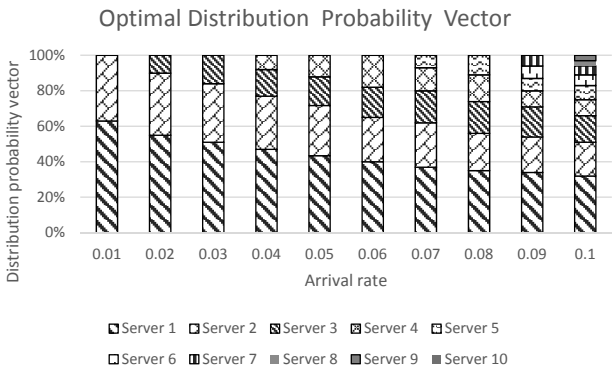

Figure 5: Probability allocation of each server.

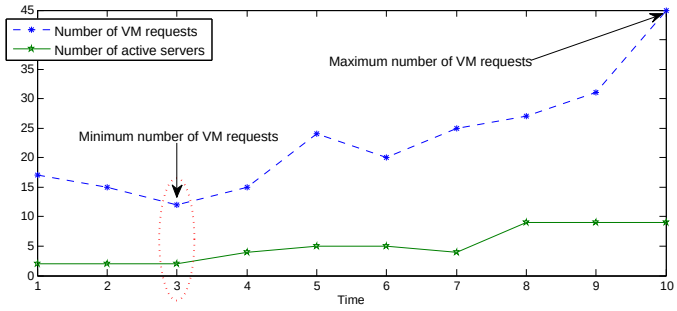

Figure 6: Controlling number of active servers.

Algorithm 1: Live consolidation algorithm

Function *allocateVMs* **is**
  Input: λ, a set of active servers $(1, 2, ..., m)$, vector of service rate $(u_1, u_2, ..., u_m)$.
  Result: Allocate VMs
  while *Having a new request* **do**
    Step 1: Computing blocking probability and utility of each server by (3) and (5).
    for $i \leftarrow 1$ **to** m **do**
      Calculate the utility of each server based on (5);
      Calculate the blocking probability of each server based on (3);
    end
    Calculate the average utility of all servers;
    Calculate the average blocking probability of all servers;
    Step 2: Update the active server set.
    if *(the average blocking probability* $> P^{thr}_{blocking}$*)* **then**
      Start a new physical machine;
      Add a new physical machine into the active server set;
      $m \leftarrow m + i$;
    end
    if *(the average utility* $< U^{thr}_{underload}$*)* **then**
      Randomly remove an active physical machine;
      $m \leftarrow m - i$;
    end
    Step 3: Distribute the request to the destination PM.
    if *(m is changed)* **then**
      Calculate the optimal distribution probability vector p^* for the new active server set;
    end
    Distribute the request into the destination server, which has the highest probability;
  end
end

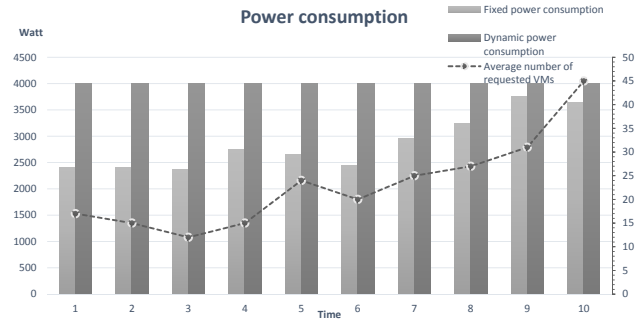

Figure 7: Comparison of the power consumption.

Table 1: System parameters

Parameter	Value
λ	0.04
$P^{thr}_{blocking}$	0.7
$U^{thr}_{underload}$	0.2
μ_1	0.03
μ_2	0.033
μ_3	0.025
μ_4	0.035
μ_5	0.035
μ_6	0.03
μ_7	0.025
μ_8	0.035
μ_9	0.033
μ_{10}	0.035

power allocation method, our model can save power consumption significantly when having low request. This reason is from reducing the number of released servers.

8. CONCLUSION

In this paper, we propose a live consolidation scheme to reduce the power consumption for servers in data centers. By using M/M/k/k queueing model, we formulate the allocation model in data center that distributes the request to servers. The analysis can help us calculate the rejection probability in each server based on transition diagram. We then represent a method to switch ON/OFF server to save power consumption. In other hands, based on the optimal distribution probability vector, we design a distribution method that eliminates underloaded servers. Thus, it can save energy consumption and reduce costs. In particular of mobile cloud computing, users move frequently and the duration of service time is short. Therefore, it is necessary for live consolidation scheme to reduce resource fragmentation and redundancy. This consideration helps us not only to reduce the power consumption, but also to load balance requests to active servers. The analysis and simulation show that our proposed system is over perform the old one and can be applied to the scheduling function of data centers.

Acknowledgments

This work was partially supported by the ICT R&D program of MSIP/IITP, Republic of Korea. (2014-044-011-003, Open control based on distributed mobile core network) and the MSIP (Ministry of Science, ICT&Future Planning), Korea, under the ITRC (Information Technology Research Center) support program (NIPA-2014(H0301-14-1020)) supervised by the NIPA (National IT Industry Promotion Agency). *Dr. CS Hong is the corresponding author.

9. REFERENCES

[1] Luiz André Barroso, Jimmy Clidaras, and Urs Hölzle. The datacenter as a computer: an introduction to the

design of warehouse-scale machines. *Synthesis Lectures on Computer Architecture*, 8(3):1–154, 2013.

[2] Xiaobo Fan, Wolf-Dietrich Weber, and Luiz Andre Barroso. Power provisioning for a warehouse-sized computer. 35(2):13–23, 2007.

[3] Sparsh Mittal. Power management techniques for data centers: A survey. *arXiv preprint arXiv:1404.6681*, 2014.

[4] Zhen Xiao, Weijia Song, and Qi Chen. Dynamic resource allocation using virtual machines for cloud computing environment. *Parallel and Distributed Systems, IEEE Transactions on*, 24(6):1107–1117, 2013.

[5] Carlo Mastroianni, Michela Meo, and Giuseppe Papuzzo. Probabilistic consolidation of virtual machines in self-organizing cloud data centers. 2013.

[6] Mueen Uddin and Azizah Abdul Rahman. Server consolidation: An approach to make data centers energy efficient and green. *arXiv preprint arXiv:1010.5037*, 2010.

[7] Minghong Lin, Adam Wierman, Lachlan LH Andrew, and Eno Thereska. Online dynamic capacity provisioning in data centers. In *Communication, Control, and Computing (Allerton), 2011 49th Annual Allerton Conference on*, pages 1159–1163. IEEE, 2011.

[8] Tien-Dung Nguyen, An Thuy Nguyen, Man Doan Nguyen, Mui Van Nguyen, and Eui-Nam Huh. An improvement of resource allocation for migration process in cloud environment. *The Computer Journal*, 57(2):308–318, 2014.

[9] Silvano Martello and Paolo Toth. *Knapsack problems*. Wiley New York, 1990.

[10] Mayank Mishra and Anirudha Sahoo. On theory of vm placement: Anomalies in existing methodologies and their mitigation using a novel vector based approach. In *Cloud Computing (CLOUD), 2011 IEEE International Conference on*, pages 275–282. IEEE, 2011.

[11] Dimitri P Bertsekas, Robert G Gallager, and Pierre Humblet. *Data Networks*. Prentice-hall Englewood Cliffs, 2 edition, 1992.

[12] Jeffrey S Chase, Darrell C Anderson, Prachi N Thakar, Amin M Vahdat, and Ronald P Doyle. Managing energy and server resources in hosting centers. In *ACM SIGOPS Operating Systems Review*, volume 35, pages 103–116. ACM, 2001.

[13] Eduardo Pinheiro, Ricardo Bianchini, Enrique V Carrera, and Taliver Heath. Load balancing and unbalancing for power and performance in cluster-based systems. In *Workshop on compilers and operating systems for low power*, volume 180, pages 182–195. Barcelona, Spain, 2001.

[14] C Radhakrishna, M Eshwardas, and Gokul Chebiyam. Impact of voltage sags in practical power system networks. In *Transmission and Distribution Conference and Exposition, 2001 IEEE/PES*, volume 1, pages 567–572. IEEE, 2001.

[15] Aziz Murtazaev, Sangyoon Oh, et al. Sercon: Server consolidation algorithm using live migration of virtual machines for green computing. *IETE-Technical Review*, 28(3):212, 2011.

[16] Minghong Lin, Adam Wierman, Lachlan LH Andrew, and Eno Thereska. Dynamic right-sizing for power-proportional data centers. *IEEE/ACM Transactions on Networking (TON)*, 21(5):1378–1391, 2013.

[17] Haikun Liu, Hai Jin, Xiaofei Liao, Chen Yu, and Cheng-Zhong Xu. Live virtual machine migration via asynchronous replication and state synchronization. *Parallel and Distributed Systems, IEEE Transactions on*, 22(12):1986–1999, 2011.

[18] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge university press, 2004.