Elasticity-Aware Virtual Machine Placement in K-ary Cloud Data Centers

Kangkang Li*1, Jie Wu2, Adam Blaisse3

1, 2, 3 Department of Computer and Information Sciences, Temple University, Philadelphia, USA

*1 kang.kang.li@temple.edu , 2 jiewu@temple.edu, 3 adam.blaisse@temple.edu

Abstract- **With the increasing popularity of cloud computing, the cloud data center suffers from both limited resources and the variation of users' requests. One important feature of cloud computing is on-demand scaling, enabling the fluctuation of one user's resource demand. However, amongst previous work concerning the virtual machine (VM) placement in data centers, satisfying requested resources of VMs is the primary objective, which neglects the future demand variation. In this paper, we propose the concept of *elasticity*, referring to how well the data center can satisfy the growth of the input VMs' resource demands under both the limitations of physical machines (PMs) capacities and links capacities. In order to consider both dimensions of the machine and bandwidth resources simultaneously, we propose our hierarchical VM placement algorithm. Besides that, we also prove the optimality of our algorithm in a frequently used *semi-homogeneous* data center configuration. Furthermore, we study the heterogeneous data center configuration, favoring the characteristics of multi-tenant data centers. Evaluation results validate the efficiency of our algorithm.**

Key words- Elasticity-aware; VM Placement; Data Centers; K-ary Topology

I. INTRODUCTION

Nowadays, cloud computing is an emerging technology that greatly shapes our lives. Several key advantages are brought with the rise of this new paradigm of computing, such as pay as-you-go metered service. With the large pools of computing and storage resources provided by cloud providers, many companies can rent these resources over the Internet, thereby saving a great amount of investment in upfront infrastructures.

Driven by technology advances, data centers are becoming the mainstream hosting platform for a variety of cloud services. Today's data center usually adopts a topology of K-ary multi-layer tree with K PMs connecting to each access switch at the bottom. However, due to the frequently exhibition of high link utilization and even congestion at data center's aggregation or core layers [1], most data centers usually reserve more bandwidth for the upper layer links to ease higher-layer links congestion. With the resource limitation of both physical machines (PMs) and links, many previous works have focused on the efficient resource management of data centers. One basic issue of resource management in cloud data centers is the VM placement problem, which is a complicated task involving various constraints, including performance [2], availability [3], network [4], and cost [5]. However, among the existing literatures which address the VM placement in data centers, most of them only focus on satisfying the resource demands of VM requests, neglecting the future variation of VMs' resource demands. In fact, on-demand scaling is one important advantage of cloud computing. Ignoring the potential future growth of resource demands is not an efficient way to manage the limited resources.

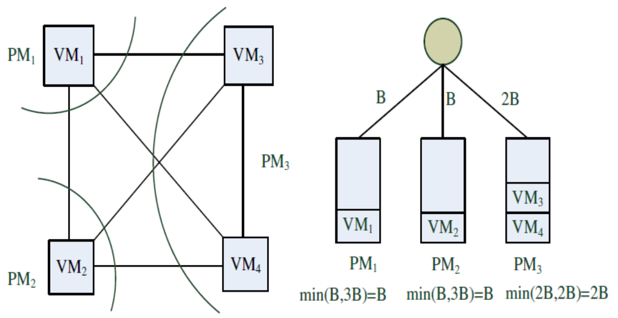


Fig.1. Communication model

In this paper, we assume that VMs have identical machine resource demands (i.e. CPU) of R and bandwidth demands of B. Due to various reasons (e.g. incremental tasks from users), the resource demands may fluctuate in the future. For instance, if R and B increase to R' and B', then the growth ratios of $\frac{R'-R}{R}$ and $\frac{B'-B}{B}$ describe, respectively, to what extent the growth of machine and bandwidth demands can be satisfied. So we define the *machine/bandwidth elasticity* as the largest ratio that the machine/bandwidth demand of each VM can increase. Due to the difference between bandwidth and machine resources, for one VM, the elasticities of machine and bandwidth are not the same. Therefore, we choose the smaller elasticity to be the *combinational elasticity* of the corresponding VM. Furthermore, the VMs belonging to the same user usually require the same growth ratio, while these VMs scattered across the data center are not likely to have the same elasticity. Therefore, we pay attention to the worst-case. That is, we select the minimal elasticity among all VMs in the data center as the objective to be maximized.

A hose model is used in [6] to calculate the bandwidth demands of PMs. An example is shown in Fig. 1, where 4 VMs are placed into 3 PMs (each cut of the graph means the corresponding VMs are placed into a PM). As aforementioned, each VM desires an identical bandwidth, B, to communicate with the other VMs, however, the bandwidth allocated to pairs of VMs are *unknown*. For example, the bandwidth assigned between VM_1 and VM_2 is unknown, while the total bandwidth token by VM_1 is B. Therefore, PM_1 has a communication demand of *at most* B. This is because: (1) VM_1 , which is placed in PM_1 can use at most B bandwidth; (2) VMs placed outside PM_1 (three VMs) can use at most 3B bandwidth; (3) the bandwidth desired by PM_1 is limited to both the VMs located in PM_1 and the VMs located outside PM_1, i.e., min{B,3B}=B. In other words, the bandwidth desired by a PM is the minimum bandwidth demands of the VMs located in it and the VMs located outside it.

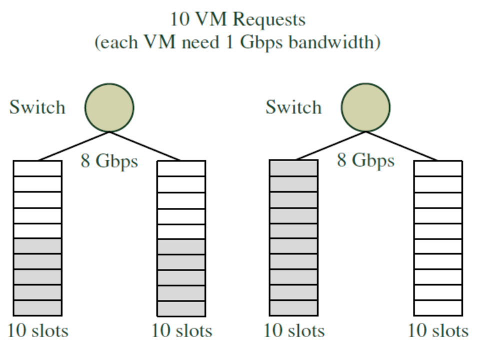


Fig.2. Illurstration of VM placement

In order to maximize the combinational elasticity, we need to consider both machine elasticity and bandwidth elasticity. An example is shown in Fig. 2 (each PM has 10 VM slots, each link bandwidth is 8 Gbps). Now we want to place 10 VMs, each of which needs one VM slot in the PM with 1 Gbps bandwidth. In order to optimize the machine elasticity of each VM, we should adopt load balancing placement, and each PM is assigned 5 VMs. In that case, the maximal machine resource of each VM can increase to $\frac{10}{5}=2$ slots (therefore, the machine elasticity is $\frac{2-1}{1}=100\%$). But, according to the hose communication model, the bandwidth usage on the links connecting two PMs is 5 Gbps, leading to less reserved links resources. The maximal bandwidth resource of each VM can only increase to $\frac{8}{5}=1.6$ Gbps (the bandwidth elasticity is only $\frac{1.6-1}{1}=60\%$). On the other hand, if we want to maximize the bandwidth elasticity, we should take a load-unbalancing placement, which puts all 10 VMs on one PM. In that case, there is no bandwidth load on the link connecting two PMs. However, the machine resources are sacrificed; since the machine elasticity is only 0% (each VM cannot have any growth in machine resources). From here, we can see the conflict on the optimization of machine and bandwidth elasticity.

In a multi-layer cluster with M machines and N VM requests, traversing all the possibilities to partition the N VMs into M machines could find an optimal solution; however, it would be extremely time-consuming. In that case, we propose our hierarchical scheme that recursively places VMs step by step from the top layer to the bottom layer.

The remainder of the paper is organized as follows: In Section II, we formulate the maximal-elasticity VM placement problem. In Section III, we study a one-layer cluster with its optimal solution. Section IV focuses on the multi-layer cluster, and gives the hierarchical VM placement algorithm. In Section V, we study the heterogeneous data center configuration. Section VI conducts the simulations to validate the efficiency of our algorithm. In Section VII, we introduce some previous work. Finally, conclusions are in Section VIII.

II. PROBLEM FORMULATION

In this section, we formulate the maximal-elasticity VM placement problem in a multi-layer K-ary tree data center. The data center configuration is *semi-homogeneous*. That is, each link of the same layer has the same bandwidth capacity: L_k (the k^{th} layer link capacity). However, the upper layer links usually have larger bandwidth capacities than the lower layer links, i.e., $L_1 \geq L_2 \geq L_3$. The links capacities only differ between layers. Also, each PM has the same capacity of C. Therefore, we refer to this as the *semi-homogeneous* configuration, which is widely used to ease upper-layer link congestion. Fig. 3 shows a binary-tree data center topology with semi-homogeneous configuration.

In this paper, we only study the scenario of VM requests with homogeneous resource demands. Each VM has an identical machine resource demand of R and bandwidth demand of B. The machine and bandwidth elasticities of a VM_i are denoted as E_i^r and E_i^l. The combinational elasticity of VM is $E_i = \min\{E_i^r, E_i^l\}$. Therefore, we have:

$$R_i' = R_i * (1+E_i) \quad and \quad B_i' = B_i * (1+E_i) \tag{1}$$

R_i' and B_i' are, respectively, the potential machine and bandwidth resource demands of a VM_i might request in the future. Our objective is to maximize the achievable E_i among all the VMs

$$Maximize \ \min_i\{E_i\} \tag{2}$$

However, each VM has both dimensions of bandwidth and machine resource demand, they could have different growth ratios, thereby, different elasticities. Hence, let a linear constant coefficient c show the elasticity relationship between bandwidth and machine resource. In that case, the combinational elasticity of VM_i is $E_i = \min\{E_i^r, cE_i^l\}$. When c=1, each VM has the same elasticity for its bandwidth and machine resources.

E_i is limited by both the PM capacities and the link capacities. First, let us consider the machine elasticity. Suppose there are m_r VMs allocated into PM_r, then the maximal potential resource is $R' = \frac{C}{m_r}$, with the machine elasticity to be $\frac{R'-R}{R}$. Ignoring the constant 1, to maximize the machine elasticity, we have:

$$Maximize \ E^r = \min_r\{\frac{C}{R * m_r}\} \tag{3}$$

On the other hand, the bandwidth elasticity is constrained by the links. Suppose that, the sub-tree of link l with bandwidth capacity of L contains m_l VMs. According to the hose model, the bandwidth requirement for link l is min B*{m_l, N-m_l}. To maximize the bandwidth elasticity, we have:

$$Maximize\ E^l = \min_l \left\{ \frac{L}{B * \min\{m_l, N - m_l\}} \right\} \quad (4)$$

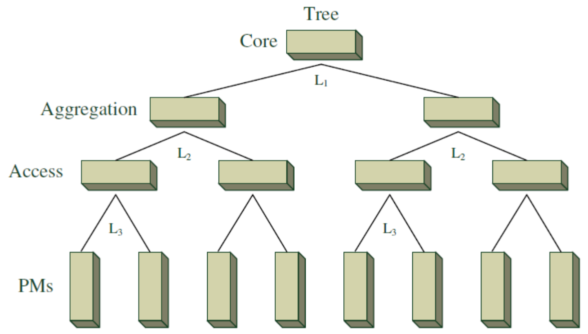


Fig.3. Tree-based Topology

According to Eqs. 3 and 4, the combinational elasticity is:

$$E = \min\left\{\min_r\left\{\frac{C}{R * m_r}\right\}, c * \min_l\left\{\frac{L}{B * \min\{m_l, N - m_l\}}\right\}\right\} \quad (5)$$

The combinational elasticity in Eq. 5 is our objective to maximize. Notably, maximizing the elasticity can be viewed as minimizing the utilization of the PM and link resources. So we transfer the objective functions Eqs. 3 and 4 into:

$$Minimize\ \max_r \left\{m_r \frac{R}{C}\right\} \quad (6)$$

$$Minimize\ \max_l \left\{\min\{m_l, N - m_l\} \frac{B}{L}\right\} \quad (7)$$

According to Eqs. 6 and 7, the combinational utilization is:

$$U = \max\left\{\max_r\left\{m_r \frac{R}{C}\right\}, \max_l\left\{\min\{m_l, N - m_l\} \frac{B}{c * L}\right\}\right\} \quad (8)$$

Again, maximizing the elasticity is equivalent to minimizing the utilization. In the next section, we focus on minimizing the utilization in Eq. 8. For simplicity, let each VM's machine resource demand as one VM slot [7]. That is, Let < N, B> denote that there are N VM requests, and each VM's bandwidth is B.

III. A ONE-LAYER CLUSTER STUDY

We start from a simple case of a one-layer cluster. As shown in Fig. 4, we have N VMs requests, each with a demand of B. Each of the K PMs has a machine capacity of C, and each of the K links' capacity is L. At first, we need to determine how many VM requests can be accepted into this one-layer cluster.

First of all, we consider the machine resources of the cluster. For each PM, the sum of the VMs' machine resources should not exceed the capacity limit. Therefore, we have N ≤ C. Second of all, we will take care of the link capacity, which will play another role in limiting the total number of supported VMs.

On the other hand, constrained by link capacity, each PM under that link can support $\frac{L}{B}$ VM slots. If L ≥ BC, one PM can support C VMs. However, if L≤BC, one PM can support, at most, $\frac{L}{B}$ VMs. Therefore, the number of VMs that one PM can support is:

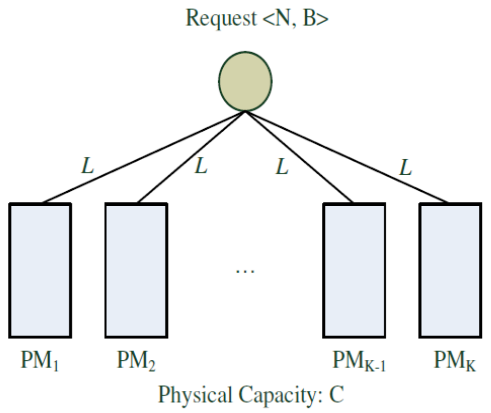


Fig.4. One-layer cluster

$$\min\left\{\frac{L}{B}, C\right\} \quad (9)$$

Given K identical PMs, K $\min\{\frac{L}{B}, C\}$ in Eq. 9 is the maximal number of VMs that a one-level cluster can support. Hence, given N VM requests, the number of VMs that can be accepted is:

$$N^* = \min\left\{N, K \min\left\{\frac{L}{B}, C\right\}\right\} \quad (10)$$

Elasticity Maximization in One-level Clusters

There are two main factors that determine the elasticity. The first one is the input size. With limited resources, large inputs will consume more resources, leading to worse elasticity. The second is the VM placement, which is our major concern. Given N VM requests, there are N^* VMs accepted into the cluster (Eq. 10).

Assume that the i^{th} PM is allocated to x_i VMs. Due to the symmetry of this K-ary tree, without of loss of generality, we assume that $x_1 < x_2 <,...,< x_K$. Therefore, for the machine utilization, Eq. 6 can be transferred as $\frac{x_K}{C}$. For the bandwidth utilization, Eq. 7 can also be transferred as: $\min\{x_K, N- x_K\}*\frac{B}{L}$. Then, considering the linear elasticity relationship between bandwidth and machine resource, the combinational utilization for PMs and links in Eq. 8 is:

$$U(x_K) = \max\left\{\frac{x_K}{C}, \min\{x_K, N - x_K\} * \frac{B}{cL}\right\} \quad (11)$$

Since we focus on worst-case, we therefore minimize the worst-case combinational utilization in Eq. 11. In that case, we only pay attention to the allotment of x_K. We can obtain the optimal result to minimize Eq. 11, as shown in Appendix A. We also give the detailed analysis for binary data center in Appendix B.

A. Discussion

Suppose that c=1, we can see from the result in Appendix A, when BC≤ L, the optimal solution for x is load-balancing.

The insight is that, the machine resource is not opulent compared to the link resources. We refer this case as the machine resources hungry, i.e., the machine resources dominate. When BC ≥ L, the link resources are comparatively scarce. Load balancing will cause large consumption on the links. To favor the link, we have to use load-unbalancing. At this time, the optimal solution is no longer evenly divided. If all the VMs are allocated into one machine, the link capacity is not used at all. The more scarce the link capacity is, the more load-unbalancing is needed.

IV. MULTI-LAYER CLUSTER STUDY

Given N VMs input and M machines at the bottom in a data center, we could traverse all the possibilities dividing the N inputs into M machines to obtain the optimal solution. However, it is time-consuming to exhaustively search for the optimal solution. However, based on the optimal results of the one-layer cluster, we can generalize this solution to multi-layer clusters, which has a considerably low time complexity.

A. K-ary Abstraction

For each switch in each layer, we can view its sub-trees as abstraction nodes. Then, the multi-layer cluster can be abstracted as a one-level binary cluster, which is easier to study. However, the obstacle is to figure out how to abstract the left or right sub-trees into a single abstraction node. We need to determine the accumulative capacity of the abstraction node. Since a sub-tree consists of constraints of both links and bottom-layer machines, we cannot just add up all the machine resources of the machines as the accumulative capacity, especially when the cluster is link resource hungry. The accumulative capacity needs to reflect resource constraints of both the links and PMs.

In order to combine the bandwidth and machine resources, we must first unify the measuring unit. For the machine resources, the measurement unit is the VM slot. Each PM has several slots, reflecting how many VMs it can support. For the bandwidth resources of links, we also want to convert them into VM slots. For instance, if a VM has a bandwidth demand of B, and the link capacity is 2B. Based on the hose model, this link could support the communication of two VM slots, i.e., the measuring unit can be converted into VM slots. Therefore, we can use VM slots as the measuring unit to represent the accumulative capacity of an abstraction node.

We can view each switch as the root of a one-layer binary cluster, and try to abstract it into a single node. Starting from the bottom-layer, each access connects K identical PMs with capacities of C. On the one hand, each PM can support at most C VM slots. That is, N ≤ C. On the other hand, constrained by link capacity, each PM under that link can support $\frac{L}{B}$ VM slots. If L≥ BC, then, one PM can support C VMs. However, if L≤ BC, one PM can support, at most, $\frac{L}{B}$ VMs. Therefore, the number of VMs that one PM can support is still min{ $\frac{L}{B}$, C}.

Algorithm 1 Hierarchical VM Placement Algorithm

Input: The links and PMs capacity; VM requests $\langle N, B\rangle$

```
1: for layer $i$=1 to $K$ do
2:   for all switches in layer $i$ do
3:     Calculate the accumulative capacity for each switch
       connecting $K$ sub-trees in the layer
4: if input VMs could be accepted then
5:   for layer $j$=$K$ to 1 do
6:     for all switches in layer $j$ do
7:       Optimally allocate the VMs to this subtree
```

Due to the symmetry of the one-layer cluster, this result also applies to all PMs connecting to the same switch. Therefore, adding these K PMs together, for each switch connecting K PMs, we can conclude that the maximum VMs this switch connecting K nodes can support is: K min{$\frac{L}{B}$,C}. We would view this maximal number of VMs as the accumulative capacity of the abstraction node, as shown in Fig. 5.

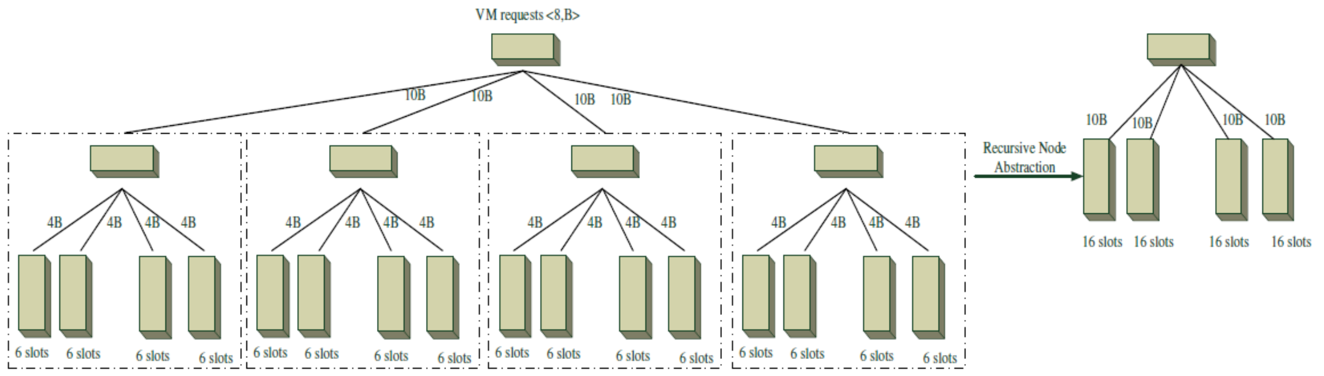


Fig. 5. The process of recursive node abstraction and recursive VM placement

Based on this abstraction of a bottom-layer switch, we are able to abstract the entire multi-layer cluster to a one-layer cluster. For each switch connecting K sub-trees at each layer from bottom to top, each one-layer cluster can be recursively abstracted into a single node. Upon reaching the root switch at the top, the whole multi-layer cluster is abstracted into a one-layer cluster. In the semi-homogeneous configuration, all the links of the same layer share the same capacity, and all the

PMs have the same machine capacity, therefore, for abstraction nodes with the same accumulative capacity, the inner structure of the original sub-trees are the same. We can see in Fig.5, a cluster consisting of four machines with two lower-layer links are abstracted into a single node of accumulative capacity of 16. Both the abstraction nodes with a capacity of 16 share the same structure of the original abstracted one-layer sub-tree.

B. *Hierarchical VM Placement for Multi-layer Clusters*

With the abstraction of a K layer multi-layer cluster into a one-layer cluster, we can use the optimal solution for the discussion in Section III. Based on that result, we propose our hierarchical VM placement algorithm for a multi-layer cluster.

THEOREM

Under the semi-homogeneous data center configuration, if BC ≤ L_K, the proposed hierarchical VM placement algorithm is optimal to minimize the combinational utilization.

The proof of optimality is shown in Appendix C. As a matter of fact, the semi-homogeneous configuration with BC ≤ L_K is widely adopted in most data centers.

Our algorithm can be divided into two steps. Firstly, for each switch from bottom to top, the accumulative capacity of the abstraction node rooted at that switch is calculated. Upon reaching the top-layer root switch, we obtain an abstracted one-layer cluster. Secondly, for the input, provided that they can be accepted, for each switch connecting K sub-trees at each layer from top to bottom, recursively allocate the input VMs into its K sub-trees according to our optimal result. Upon finishing the bottom-layer switch (access switch), all the VMs are allocated into the PMs, as shown in Fig. 5. We summarize our algorithm in Algorithm 1.

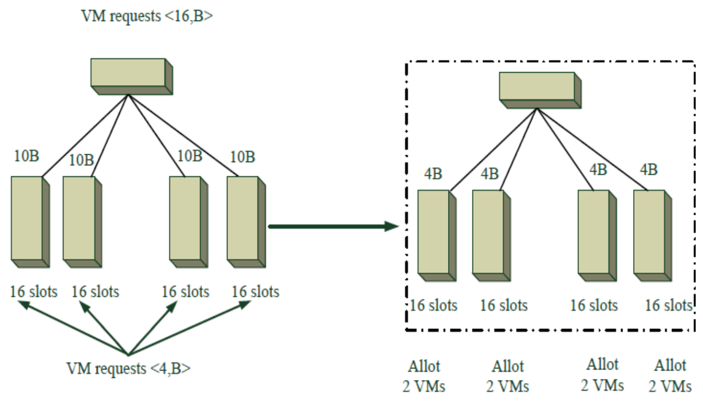


Fig.6. The process of VM placement

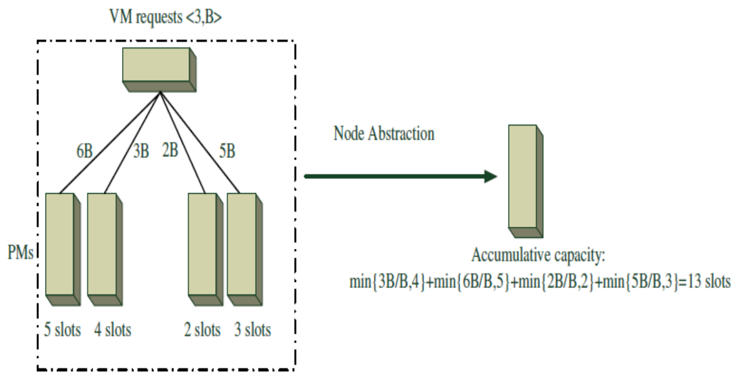


Fig.7. Conservative schedulability

For each switch at each layer, our algorithm takes a constant time to calculate the optimal solution, For a K layers cluster, layer k has 2^{k-1} switches. Therefore, the total time of calculation is $\sum_{k=1}^{K-1} k$. Suppose we have M machines at the bottom, then M=2^k. Our algorithm takes two loops, one is the abstraction from bottom to top, and the other is the placement from top to bottom. One loop takes a time of O (M). Two loops is still O (M). Therefore, the total time complexity of our algorithm is O (M), which is very efficient.

C. *Discussion*

In a link hungry condition, the abstraction process from bottom to top can be conservative. Therefore, the accumulative capacity of the abstraction node can be smaller than the actual capacity. This will lead to some situations where our algorithm cannot schedule a VM request that should be able to be scheduled. Since our abstraction process is layer-by-layer from bottom to top, each step can be conservative calculated. The boundary situation is that:

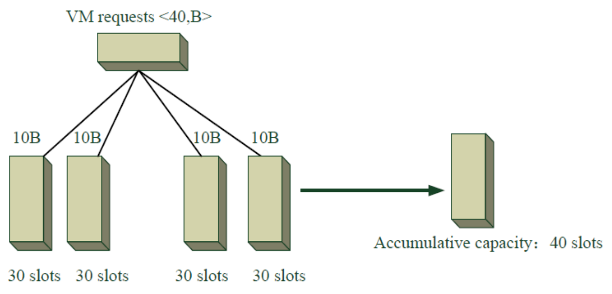


Fig.8. One-layer heterogeneous cluster

OBSERVATION

In layer k, if one abstraction node capacity is larger than the sum of layer k's links capacities minus one layer k's link capacity, then, the accumulative capacity is conservatively calculated for layer k.

The boundary situation is in Fig. 7. During the abstraction process from the bottom layer to the top, granted that one such situation happened, the whole calculation will be conservative. However, in real data centers, each PM usually supports 4~VM or 8~VM slots, usually below 10. However, for the link capacity, the bottom layer link is usually 1~Gbps, and the upper layer link capacities are usually more than 10Gbps. On the other hand, one VM usually requests for 100~Mbps. We can see that each link can support more than 10~VMs. The links are usually not hungry; therefore, the conservative calculation is *NOT* a common case.

V. A HETEROGENEOUS CASE STUDY

Now we study the scenario of heterogeneous clusters, where all the capacities of PMs are heterogeneous, along with the link capacities, as shown in Fig. 8. The motivation for studying this scenario is that today's data centers can support multiple tenants' requests. The VM requests of different tenants may come at different times. After one tenant's VMs are placed into the data center, all the links and PMs capacities will change, making the data center a heterogeneous configuration, which will make this maximal-elasticity problem an NP-hard problem. However, our hierarchical algorithm is still useful, and can provide a great approximation to the optimal results.

Similarly, we firstly study a one-layer cluster under heterogeneous configuration. However, the calculation of accumulative capacity is different, as shown in Fig. 8. We would view this maximal number of VMs as the accumulative capacity of the abstraction node.

After we recursively do the abstraction process from bottom to top, we can use the optimal result in Eq. 26 and Algorithm 1 to recursively place the input VMs into each switch. Upon finishing the bottom-layer switch, all VMs are placed in the PMs.

We give an exquisite result for binary heterogeneous data center in Appendix D.

VI. EVALUATION

In this section, we conduct two sets of simulations for both semi-homogeneous and heterogeneous configurations of data center. The topology we use is a three-layer binary tree structure, as in Fig. 3. Our VM placement algorithms are compared with the optimal solution. We produce the optimal solution by programs that traverse all the possibilities dividing the N inputs into M machines.

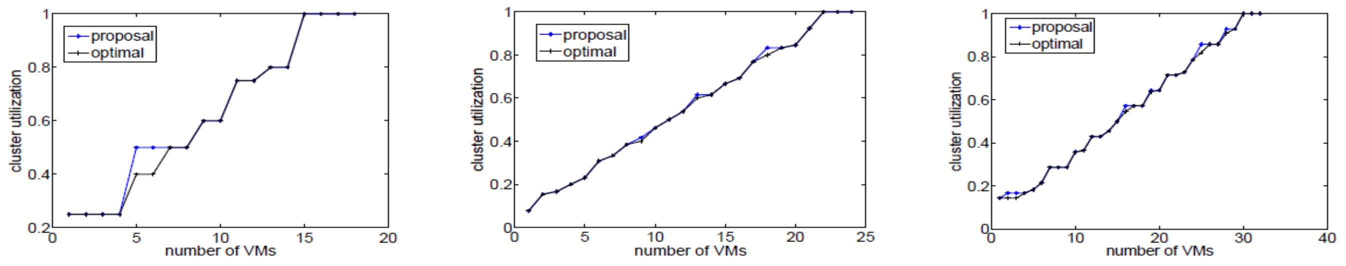
We conduct one evaluation for the semi-homogeneous configuration. Since we have proven the optimality of our algorithm for the machine resource hungry scenario, we only evaluate under the link hungry scenario. For the link hungry semi-homogeneous scenario, we vary the capacities of the bottom-layer links as 2~Gbps, 4~Gbps, and 6~Gbps. All the links between switches are identical: 10~Gbps. Each VM's bandwidth demand is 1~Gbps. Each PM has 10 VM slots. Another group of evaluation is conducted for the heterogeneous configuration. We vary link capacity and keep the PM capacity stable. Each link capacity range is [5,10] Gbps, [5,15] Gbps, [5,20] Gbps. Each VM's bandwidth demand is still 1~Gpbs.

From Fig. 9 and Fig. 10, we can see that when the number of VMs increases, the utilization increases. This is because more VMs will consume more resources, leading to the increase in the combinational utilization of the clusters, which will lower the elasticity of VMs. From Fig. 9, when the bottom-layer links capacities increase, the situation of a link hungry cluster is alleviated. Then, the data center can support more VMs, as shown in the comparison in the sub-figures. Besides, we can observe that our algorithm is very close to the optimal value. From Fig. 10, under the heterogeneous scenario, we can see that, as the links capacities increase, more VMs can be supported by the data center. Again, we can see that our algorithm is very close to the optimal solution. In sum, under both link hungry semi-homogeneous and heterogeneous scenarios, our algorithm has a high approximation to the optimal solution, which is likely to lead to a good performance for a multi-tenant data center.

Notably, according to our observation, for a 3-ary tree data center, the time to obtain the optimal result is more than ten thousand times than our algorithm, which prove the high time-saving of our algorithm.

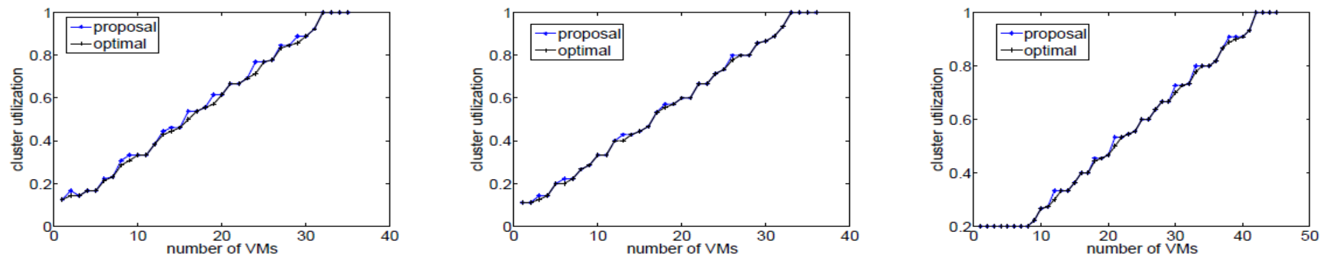
VII. RELATED WORK

Nowadays, cloud computing is an emerging technology that greatly shapes our lives. Through management by virtual machine monitor (VMM) [8-11], the physical resources of one PM can be sliced into multiple VMs. Such resource multiplexing largely reduces the total cost of ownership, and significantly improves the resource utilization. With the large pools of computing and storage resources provided by cloud providers, many companies can rent these resources and run their jobs on virtual machines (VMs), saving the investment in upfront infrastructures. Much work has been done regarding the topic of resource provisioning and VM placement in the cloud computing environment.



(a) bottom-layer links capacities: 2 Gbps (b) bottom-layer links capacities: 4 Gbps (c) bottom-layer links capacities: 6 Gbps

Fig. 9. Performance evaluation: semi-homogeneous configuration



(a) link capacity range: [5,10] Gbps (b) link capacity range: [5,15] Gbps (c) link capacity range: [5,20] Gbps

Fig. 10. Performance evaluation: heterogeneous configuration

Economic interests are one popular topic that shows up in research literature [12, 13]. They tried to find an optimal VM placement that could either maximize the revenue for the cloud provider, or minimize the costs for the customers. In [13], for maximizing the total economic benefit of the cloud provider, the author introduced an SLA-based dynamic resource allocation. The pricing mechanisms are related to the performance of QoS that the cloud provider could guarantee. The better performance the cloud provider could offer, the more revenue the cloud provider could obtain. Since different service requests have different pricing, higher pricing service could get more resource provisioning from the cloud provider. The author also used a convex optimization to present the optimal resource allocation. On the contrary, the author in [12] proposed an optimal VM placement with the objective of minimizing the total renting of the users. In this paper, the author gave another pricing mechanism, referring to two payment plans: reservation plan and on-demand plan. Since the total resource demand is uncertain and reservation plan has to be decided in advance, it may not meet the future demand of the user. For that reason, the author used the optimal solution of stochastic integer programming to give an optimal VM placement that could minimize the total renting. However, the VM placement problem in [12] and [13] could achieve an optimal solution, which is not a common case. Many VM placement issues are NP-hard, thus we need to find a good heuristic algorithm to solve the problem, such as the first-fit and best-fit greedy algorithm used in [14].

Besides the above VM placement aiming to get an optimal economic interest, network is one important concern in the VM placement problem. The author in proposed a network-aware VM placement. In [15], when performing VM placement, not only the physical resources (like CPU and memory) are considered, but also the traffic demand between different VMs was taken into account. The author gave a heuristic algorithm to allocate placement to satisfy both the communication demands and physical resource restrictions. In [16], the author proposes minimizing the traffic cost through VM placement. Their objective is to place VMs that have large communication requirements close to each other, so as to reduce network capacity needs in the data center. Oktopus [7] uses the hose model to abstract the tenant's bandwidth request, including both virtual cluster and oversubscribed virtual clusters. They propose a VM allocation algorithm to deal with homogeneous bandwidth demands, which is also what we are using in this paper. The virtual cluster provides tenants with guarantees on the network bandwidth they demand, which, according to [17], can be interpreted as the min-guarantee requirements. However, this min-guarantee fails to consider the potential growth of a tenant's network demand. In that case, the virtual cluster allocation based on this min-guarantee policy will not have enough resources to accommodate tenants' future growth demands, which can lead to the loss of customers. In order to alleviate this problem, in our previous paper [18], we propose the concept of elasticity, which considers existing customers' potential growing bandwidth demand in the future. However, that paper only considers the binary tree data center topology. To obtain a more general result, we study the K-ary tree topology data center in this paper.

VIII. CONCLUSIONS

In this paper, we study the resource management problem on the cloud data center, which is now suffering from both limited resources and the variety of users' requests. Compared to the previous work, we focus on guaranteeing the on-demand scaling of cloud computing, and we propose the concept of elasticity of the VM requests. To maximize the elasticity of the input VMs, we transfer it into the utilization minimization problem, and propose our hierarchical VM placement algorithm. We study the scheduability of the input VMs and also prove the optimality of our algorithm under a frequently used data center configuration. Furthermore, we also conduct an extended study on the heterogeneous scenario to meet the requirements of a multi-tenant data center. The evaluation results show the high efficiency of our algorithm.

APPENDIX A

K-ARY ONE-LAYER CLUSTER OPTIMAL RESULT

The objective function is:

$$Minimize\ U(x_K) = \max\{\frac{x_K}{c}, \min\{x_K, N-x_K\} * \frac{B}{cL}\} \quad (12)$$

With link and PM capacity limits, the domain of x_K is:

$$x_K \in [\frac{N^*}{K}, \min\{\frac{L}{B}, C\}] \quad (13)$$

The minimal U (x_K) can be obtained, when x_K is:

$$x_K^* = \begin{cases} \frac{1}{K}N^*, & C \geq c\frac{L}{B} \\ \frac{C}{c\frac{L}{B}+C}N^*, & C \leq c\frac{L}{B} \end{cases} \quad (14)$$

APPENDIX B

SEMI-HOMOGENEOUS BINARY TREE ANALYSIS

Here, we assume that c=1. First of all, we consider the physical resources of the cluster. For each PM, the sum of the VMs' physical resources should not exceed the capacity limit. Therefore, we have N ≤ 2C. Secondly, we would take care of the link capacity, which plays another role in limiting the total number of supported VMs. Suppose that each PM has 5 VM slots, each VM has a bandwidth demand of B, and the capacity of each link is 2B. If we allocate one PM with 5 VMs, based on the hose communication model, the other PM could maximally hold 2 VMs, even with a capacity of 5 VM slots. Therefore, from the aspect of link capacity limit, the maximal number of VMs of this one-level cluster is C+$\frac{L}{B}$. So we have the following:

$$N \leq \min\{2C, C+\frac{L}{B}\} \quad (15)$$

min {2C, C+ $\frac{L}{B}$ } in Eq. 15 is the maximal number of VMs that a one-level cluster can support. Hence, given N VM requests, the number of VMs that can be accepted is:

$$N^* \leq \min\{N, 2C, C+\frac{L}{B}\} \quad (16)$$

With N^* VMs accepted into the cluster. Assume that we place x VMs in the left machine, and leave the N^*-x to the right node (without loss of generality, let x≤ N^*-x). As the value of x increases from 0 to $\lfloor\frac{N^*}{2}\rfloor$, due to the symmetry of binary-tree, there are $\lfloor\frac{N^*}{2}\rfloor$+1 different ways to allocate the VMs in two machines. However, the value of x may not be able to achieve a value as small as 0, due to the limitations of PM's capacity, or as large as $\lfloor\frac{N^*}{2}\rfloor$, due to the limitations of the links capacity. As the value of x increases, the utilization of the PMs decreases since the input VMs are allocated in a more balance manner. However, the utilization of the links increases, since the traffic between the VMs in two PMs are more heavily loaded.

In this one-level cluster, links have identical capacities; we can transfer the utilization in Eq. 6 of the link as $x * \frac{B}{L}$. Since we assume that, x ≤ N^*-x, the utilization of the physical resources in Eq. 6 could be transferred as max{ $\frac{N-x}{C}$, $\frac{x}{C}$ }= $\frac{N-x}{C}$. Then, the combinational utilization of PMs and links in Eq. 8 is:

$$U(x) = \max\{x\frac{B}{L}, \frac{N-x}{C}\} \quad (17)$$

The objective function is:

$$Minimize\ U(x) = \max\{x\frac{B}{L}, \frac{N-x}{C}\} \quad (18)$$

With link and PM capacity limits, the domain of x is:

$$x \in [\max\{N^*-C, 0\}, \min\{\frac{L}{B}, \lfloor\frac{N^*}{2}\rfloor\}] \quad (19)$$

The minimal point for U(x) can be obtained, when x is:

$$x^* = \frac{\frac{L}{B}}{\min\{C,\frac{L}{B}\}+C}N^* \quad (20)$$

Apparently, x^* is within the domain, which is the optimal solution to allocate the N^* input VMs into two machines. However, x^* in Eq. 20 may not be an integer. In that case, we compare the value of U(x) under both x=$\lfloor x^*\rfloor$, and x=$\lceil x^*\rceil$. So the optimal solution for maximizing the elasticity is:

$$x^* = \begin{cases} \lfloor x^*\rfloor, U(\lfloor x^*\rfloor) \leq U(\lceil x^*\rceil) \\ \lceil x^*\rceil, U(\lfloor x^*\rfloor) \geq U(\lceil x^*\rceil) \end{cases} \quad (21)$$

APPENDIX C

THEOREM PROOF

Since under machine resource hungry situation, one-step optimal result is to equally divide the input VMs. In that case, each of the K machines will get the same allotment. We give the detailed binary proof for the optimality, which could be easily extended to the K-ary tree.

A. A two-layer optimality

Here, we also assume that c=1. Suppose that the physical capacity of each machine is C in a two-layer cluster, and the bandwidth capacity is B_1 for the upper layer links and B_0 for the lower layer. We have $B_1 \geq B_0$. Here we adopt the unified measuring unit of VM slot. Given N VM requests, without loss of generality, we assume each VM requires physical resource of one unit of C, and bandwidth of one unit of B_1 or B_0. Also, we assume that N VMs requests are schedulable in this cluster. Here we study the case in which B_0 ≥ C.

Suppose that, based on our algorithm, the allotment for the four machines is [a,b,c,d]. With the symmetric characteristic of binary-tree, according to our algorithm, we always allocate the larger allotment of the given input on the right side without loss of generality. Then, we have a ≤b ≤c ≤d.

Assume our algorithm is not optimal, therefore, there is another allocation for the four machines of [a',b',c',d'], that is better than our algorithm in minimizing the combinational utilization in Eq. 30. We assume a' ≤ b' ≤ c' ≤ d'. Due to the symmetry of binary-tree, we could swap each PM's allotment, so as to make [a',b',c',d'] corresponds to [a,b,c,d].

Based on our algorithm, for the first allocation step, we allocate the N requests into the two abstraction nodes of representing the two sub-trees of the root switch. By Eq. 14, the accumulative capacity of each abstraction node is C_M= min {2 B_0 , 2C} =2C. For the one-layer cluster, our algorithm optimally allocates the N VMs based on the results of Eq. 13. Since a ≤b ≤c ≤d and a' ≤ b' ≤ c' ≤d', therefore, for the first step, we have:

$$\max\{\frac{a+b}{B_1},\frac{c+d}{2C}\} \leq \max\{\frac{a'+b'}{B_1},\frac{c'+d'}{2C}\} \qquad (22)$$

Then, after finishing the allocation for the top layer, according to our algorithm, we are about to do the allocation for the two switches in the second layer. The combinational utilization for the second-layer switch is max $\{\frac{a+b}{B_1},\frac{d}{C},\frac{\min\{a+b+c,d\}}{B_0}\}$. Assuming our algorithm is not optimal, we have:

$$\max\left\{\frac{a+b}{B_1},\frac{d}{C},\frac{\min\{a+b+c,d\}}{B_0}\right\} \geq \max\left\{\frac{a'+b'}{B_1},\frac{d'}{C},\frac{\min\{a'+b'+c',d'\}}{B_0}\right\} \qquad (23)$$

Since B_0≥C, the result in Eq. 21 will evenly divide the input of the second-layer switch. Therefore, we have a=b, c=d. Then, min{a+b+c,d}=d, and $\frac{d}{C} \geq \frac{d}{B_0}$. Therefore, we have:

$$\max\left\{\frac{a+b}{B_1},\frac{d}{C},\frac{\min\{a+b+c,d\}}{B_0}\right\} = \max\left\{\frac{a+b}{B_1},\frac{d}{C}\right\} \qquad (24)$$

Since d=c, then, $\frac{d}{C}=\frac{c+d}{2C}$. Combing Eq. 24, we have:

$$\max\left\{\frac{a+b}{B_1},\frac{d}{C},\frac{\min\{a+b+c,d\}}{B_0}\right\} = \max\{\frac{a+b}{B_1},\frac{c+d}{2C}\} \qquad (25)$$

Since a' ≤ b' ≤ c' ≤ d', therefore, d' ≥ c'. Then, $\frac{d}{C} \geq \frac{c'+d'}{2C}$. Therefore, we have:

$$\max\{\frac{a'+b'}{B_1},\frac{d'}{C},\frac{\min\{a'+b'+c',d'\}}{B_0}\} \geq \max\{\frac{a'+b'}{B_1},\frac{c'+d'}{2C}\} \qquad (26)$$

Hence, combing Eqs. 25, 26

$$\max\left\{\frac{a'+b'}{B_1},\frac{d'}{C},\frac{\min\{a'+b'+c',d'\}}{B_0}\right\} \geq \max\{\frac{a'+b'}{B_1},\frac{c'+d'}{2C}\} \geq \max\{\frac{a+b}{B_1},\frac{c+d}{2C}\} = \max\left\{\frac{a+b}{B_1},\frac{d}{C},\frac{\min\{a+b+c,d\}}{B_0}\right\} \qquad (27)$$

A contradiction with the assumption in Eq. 23, therefore, the proof is complete for B_0≥C.

B. *Generalization of Optimality*

From one-layer to two-layer, the potential factor that can change the optimality of this greedy step is the change of resource capacity, since we use the accumulative capacity to complete the first step. In other words, $\frac{c+d}{C_M}=\frac{c+d}{2C}$ can be smaller than max { $\frac{\min\{a+b+c,d\}}{B_0}$, $\frac{d}{C}$ }. However, according to the allocation scheme of our algorithm, we have proven that max { $\frac{\min\{a+b+c,d\}}{B_0}$, $\frac{d}{C}$ } is equal to $\frac{c+d}{C_M}$, which guarantees the generalization of optimality for ones-step. Since all PMs' capacities are the same, and the capacity of the links at the same layer are the same, from the view of physical meaning, our abstraction process misses no information of the lower layer resources for both links and machines. Hence, we conclude that the optimality is secured for one-step generalization. Based on this observation, we can use induction method to prove the optimality of our algorithm in a k layer cluster. When k=1, it is a one-layer cluster, and the optimal solution is given. Suppose that, for the k^{th} layer, our algorithm is optimal, then, the optimality from the k^{th} to $k^{th}+1$ layer is also secured. Therefore, our algorithm is optimal.

APPENDIX D

BINARY HETEROGENEOUS CLUSTER ANALYSIS

Here, we consider the scenario that c=1. Similarly, we firstly study a one-layer cluster under heterogeneous configuration. However, the calculation of accumulative capacity is different, as shown in Fig. 8. With an input < N, B>, considering the total physical resources of the cluster, we have: N ≤ C_1+ C_2. Secondly, with the link capacity constraint, we have: N ≤max { C_1, C_2}+ min $\{\frac{L_1}{B},\frac{L_2}{B}\}$. Considering both link and physical capacity limitation, we have:

$$N \leq \min\{\max\{C_1,C_2\}+\min\left\{\frac{L_1}{B}+\frac{L_2}{B}\right\},C_1+C_2\} \qquad (28)$$

This is the maximal number of VMs that this heterogeneous one-layer cluster can support. We still use N^* as the number of VMs that can be accepted into this cluster, as follows:

$$N^* = \min\{N,\max\{C_1,C_2\}+\min\{L_1+L_2\},C_1+C_2\} \quad (29)$$

We try to obtain an optimal result for the one-layer cluster. With N^* accepted into this cluster, we allocate x VMs on the left node, leaving N^* −x on the right node. In a heterogeneous scenario, the configuration of a binary cluster is asymmetric; hence, we need to consider all N^* +1 different ways to allocate the VMs. We will discuss the problem separately in the interval of x ≤ $\left\lfloor\frac{N^*}{2}\right\rfloor$ and x ≥ $\left\lfloor\frac{N^*}{2}\right\rfloor$, and then lead to the final result.

For x ≤ $\left\lfloor\frac{N^*}{2}\right\rfloor$, the utilization of the links is max{x$\frac{B}{L_1}$, x$\frac{B}{L_2}$}, and the utilization of physical resources is max{x $\frac{R}{C_1}$, (N^*-x) $\frac{R}{C_2}$}. Then, the combinational utilization of the cluster is:

$$U(x) = \max\{x\frac{B}{L_1},x\frac{B}{L_2},x\frac{B}{C_1},(N^*-x)\frac{R}{C_2}\} \qquad (30)$$

The domain of x is:

$$x \in [\max\{\min\{N^*-C_2,C_1\},0\},\min\{\frac{L_1}{B},\frac{L_2}{B},\frac{N^*}{2}\}] \qquad (31)$$

Mathematically, there is a minimal point for U(x), when x is:

$$x_i^* = \frac{\min\{\frac{L_1}{B},\frac{L_2}{B},C_1\}}{\min\left\{\frac{L_1}{B},\frac{L_2}{B},C_1\right\}+\min\{\frac{L_1}{B},\frac{L_2}{B},C_1,C_2\}}N^* \qquad (32)$$

Obviously, x_l^* is within the domain Eq. 31. Still, the value of x^* from Eq. 32 may not be an integer. We choose both values of $\lfloor x_l^* \rfloor$ and $\lceil x_l^* \rceil$ for future comparison with another interval.

Similarly, for $\left\lfloor \frac{N^*}{2} \right\rfloor \leq x \leq N^*$, the objective function for the utilization of the cluster is:

$$U(x) = \max\left\{(N^* - x)\frac{B}{L_1}, (N^* - x)\frac{B}{L_2}, (N^* - x)\frac{B}{C_1}, x\frac{B}{C_2}\right\} \quad (33)$$

To minimize U(x) in Eq. 33, the domain of x is:

$$x \in [\max\left\{\min\{N^* - C_2, C_1\}, \frac{N^*}{2}\right\}, \min\{\frac{L_1}{B}, \frac{L_2}{B}, N\}] \quad (34)$$

There is also a minimal point, which is:

$$x_r^* = \frac{C_2}{\min\left\{\frac{\min\{L_1, L_2, C_1\}}{B}, C_2\right\} + C_2} N^* \quad (35)$$

We will choose the values of $\lfloor x_l^* \rfloor$, $\lceil x_l^* \rceil$, $\lfloor x_r^* \rfloor$ and $\lceil x_r^* \rceil$ for comparison. Therefore, the optimal result x^* of allocation for a one-layer heterogeneous cluster is shown below:

$$\begin{cases} \lfloor x_l^* \rfloor, U_l^h(\lfloor x_l^* \rfloor) = \min\{U_l^h(\lfloor x_l^* \rfloor), U_l^h(\lceil x_l^* \rceil), U_r^h(\lfloor x_r^* \rfloor), U_r^h(\lceil x_r^* \rceil)\} \\ \lceil x_l^* \rceil, U_l^h(\lfloor x_l^* \rfloor) = \min\{U_l^h(\lfloor x_l^* \rfloor), U_l^h(\lceil x_l^* \rceil), U_r^h(\lfloor x_r^* \rfloor), U_r^h(\lceil x_r^* \rceil)\} \\ \lfloor x_r^* \rfloor, U_l^h(\lfloor x_l^* \rfloor) = \min\{U_l^h(\lfloor x_l^* \rfloor), U_l^h(\lceil x_l^* \rceil), U_r^h(\lfloor x_r^* \rfloor), U_r^h(\lceil x_r^* \rceil)\} \\ \lceil x_r^* \rceil, U_l^h(\lfloor x_l^* \rfloor) = \min\{U_l^h(\lfloor x_l^* \rfloor), U_l^h(\lceil x_l^* \rceil), U_r^h(\lfloor x_r^* \rfloor), U_r^h(\lceil x_r^* \rceil)\} \end{cases}$$

REFERENCES

[1]. S. Kandula, S. Sengupta, A. Greenberg, P. Patel, and R. Chaiken, "The nature of data center traffic: measurements & analysis," in Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference, IMC '09, (New York, NY, USA), pp. 202-208, ACM, 2009.

[2]. D. Kusic, J. Kephart, J. Hanson, N. Kandasamy, and G. Jiang, "Power and performance management of virtualized computing environments via lookahead control," in *Proceedings of ICAC 2008*, pp. 3 –12, june 2008.

[3]. J. E. Bin, O. Biran, O. Boni, E. Hadad, E. Kolodner, Y. Moatti, and D. Lorenz, "Guaranteeing high availability goals for virtual machine placement," in *Proceedings of ICDCS 2011*, pp. 700 –709, june 2011.

[4]. E. P. W J. T. Piao and J. Yan, "A network-aware virtual machine placement and migration approach in cloud computing," in *Proceedings of GCC 2010*, pp. 87 –92, nov. 2010.

[5]. K. Li, H. Zheng, J. Wu, "Migration-based Virtual Machine Placement in Cloud Systems " in *Proc. of IEEE CloudNet 2013*.

[6]. J. Zhu, D. Li, J. Wu, H. Liu, Y. Zhang, and J. Zhang, "Towards bandwidth guarantee in multi-tenancy cloud computing networks," in Proc. of ICNP 2012, pp. 1 –10.

[7]. H. Ballani, P. Costa, T. Karagiannis, and A. Rowstron, "Towards predictable datacenter networks," in *Proc. of the ACM SIGCOMM 2011*, pp. 242–253.

[8]. J. R. Lange and P. Dinda, "Symcall: symbiotic virtualization through vmm-to-guest upcalls," in *Proceedings of the 7th ACM SIGPLAN/ SIGOPS international conference on Virtual execution environments*, VEE '11, (New York, NY, USA), pp. 193–204, ACM, 2011.

[9]. S. T. Jones, A. C. Arpaci-Dusseau, and R. H. Arpaci-Dusseau, "Vmmbased hidden process detection and identification using lycosid," in *Proceedings of the fourth ACM SIGPLAN/SIGOPS international conference on Virtual execution environments*, VEE '08, (New York, NY, USA), pp. 91–100, ACM, 2008.

[10]. M. Xu, X. Jiang, R. Sandhu, and X. Zhang, "Towards a vmm-based usage control framework for os kernel integrity protection," in *Proceedings of the 12th ACM symposium on Access control models and technologies*, SACMAT '07, (New York, NY, USA), pp. 71–80, ACM, 2007.

[11]. A. Ranadive, A. Gavrilovska, and K. Schwan, "Ibmon: monitoring vmm-bypass capable infiniband devices using memory introspection," in *Proceedings of the 3rd ACM Workshop on System-level Virtualization for High Performance Computing*, HPCVirt '09, (New York, NY, USA), pp. 25–32, ACM, 2009.

[12]. S. Chaisiri, B.-S. Lee, and D. Niyato, "Optimal virtual machine placement across multiple cloud providers," in *Proceedings of IEEE Asia-Pacific Services Computing Conference, 2009*, pp. 103 –110, dec. 2009.

[13]. G. Feng, S. Garg, R. Buyya, and W. Li, "Revenue maximization using adaptive resource provisioning in cloud computing environments," in *Proceedings of Grid Computing 2012*, pp. 192 –200, sept. 2012.

[14]. J. Xu and J. Fortes, "Multi-objective virtual machine placement in virtualized data center environments," in *Proceedings of CPSCom 2010*, pp. 179 –188, dec. 2010.

[15]. J. T. Piao and J. Yan, "A network-aware virtual machine placement and migration approach in cloud computing," in *Proceedings of GCC 2010*, pp. 87 –92, nov. 2010.

[16]. X. Meng, V. Pappas, and L. Zhang, "Improving the scalability of data center networks with traffic-aware virtual machine placement," in *Proc. of IEEE INFOCOM 2010*, pp. 1–9, 2010.

[17]. L. Popa, A. Krishnamurthy, S. Ratnasamy, and I. Stoica, "Faircloud: sharing the network in cloud computing," in *Proc. of ACM HotNets-X 2011*, pp. 22:1–22:6.

[18]. K. Li, J. Wu, A. Blaisse, "Elasticity-aware Virtual Machine Placement for Cloud Datacenters " in *Proc. of IEEE CloudNet 2013*.