Elastic Scaling of Virtual Clusters in Cloud Data Center Networks

Shuaibing Lu*†, Zhiyi Fang*, Jie Wu†, and Guannan Qu*
*College of Computer Science and Technology, Jilin University Changchun,130012, China
†Center for Networked Computing, Temple University, USA
Email: lushuaibing11@163.com, fangzy@jlu.edu.cn, jiewu@temple.edu

***Abstract*—Data Center Networks (DCNs) become more extensively applied on the cloud computing in recent years. One important mission for DCNs is to satisfy the fluctuation of on-demand resources for tenants. Existing works fail to fully consider the scaling virtual clusters (VCs) placement techniques and the elasticity of the physical resource in the DCN at the same time. To address this mission, we use the concept of elasticity to measures the scaling potential of VCs in terms of both computation and communication resources. In this paper, we consider the scaling placement for the existing VCs, to maximize the elasticity with the constrain of communication cost in the DCN. We first achieve this through a resource allocation scheme VCS which comes with provable optimality guarantees for single VC scaling. After that, we extend it into the multiple VCs scaling, and prove that multiple VCs scaling for over-time elasticity maximization problem is NP-hard. We propose heuristic algorithms MVCS and OMVCS for both offline and online conditions on the multiple VCs scaling. We only consider the condition which scaling on both computing and communication resources, which can also be adapted to each individual resource. Extensive simulations demonstrate that, our elastic VCs scaling placement schemes outperform various existing state-of-the-art methods in terms of flexibility in the DCN.**

***Index Terms*—Data center networks (DCNs), elastic scaling, virtual cluster, optimization, virtual machine (VM) placement.**

I. Introduction

Data Center Networks (DCNs) become more extensively applied on the cloud computing in recent years. The applications which are based on the cloud generate a significant amount of network traffic and a considerable fraction of their runtime is due to network activity [1]. As reported in [2], the available resource to the tenants vary a good deal over time in EC2. One major problem for tenants to the cloud computing is the lack of performance guarantee, which includes both resource limitation and unpredictable application demands.

To address these problems, we propose an flexible placement strategy to dealing with the resource scaling for the existing Virtual Clusters (VCs) in the DCN. A set of Virtual Machines (VMs) which connect on one virtual switch is defined as a VC. Each VC not only has computing requirements, it also requires communications among themselves to complete the specified tasks for the applications. We use the concept of elasticity [3] to measures the growth potential of a VC placement in both computation and communication, which denotes the physical machine (PM) elasticity and the physical link (PL) elasticity. Our objective is to maximize the elasticity during the VCs placement in the DCN under the resource and communication cost constraints. This paper is based on existing DCN architecture which is a Fat-tree, and the capacities of PMs are slotted, and each slot can only host one VM, as shown in Fig. 1. The communication between VMs occurs through the PLs for each VC. The corresponding communication demands are determined by VM communication models. This paper uses hose model, which is a communication model used to calculate bandwidth demand for VMs.

To maximize the elasticity in DCN under the hose communication model, we face one important challenge is balancing the trade-off between communication cost and the elasticity during the VC placement, and guaranteeing both computing and communication resource scaling for one VC under the DCN. Take Fig. 1 as example, VC1 has 7 existing VMs in the DCN. We assume the upper bound of subtree root is in the aggregation switch level. The scaling request for VC1 is from 5 VMs. One extreme assignment for the scaling request is to concentrated place all the scaling VMs into the PMs M_{00} and M_{01} under the switch S_{11}, as shown in the Fig. 1(a). This solution can save the communication cost between existing VMs and the incoming ones. However, the elasticity of VC1 will decrease to 0, which contains two bottleneck PMs M_{00} and M_{01} with no scaling potential. Another extreme assignment is to place the VMs dispersedly as shown in Fig. 1(b). In this case, the elasticity of VC1 will be $\min\{\frac{2}{5}, \frac{2}{5}, \frac{3}{5}, \frac{3}{5}, \frac{4}{5}, \frac{4}{5}\} = \frac{2}{5}$. However, the communication cost under this assignment has already beyond the upper bound of subtree root, which can not guarantee the QoS for the users. In order to balance the trade-off between communication cost and the elasticity, we try to find a solution between the two extreme assignments. In this paper, we propose an optimal solution that doing the placement based on the proportion of the remaining available capacities for the scaling request under the limitation of communication cost, as shown in Fig. 1(c). Then we extent it into online multiple VCs scaling placement problem, which solved by a heuristic method with prediction. Our algorithm can improve the overtime elasticity through using history knowledge and distribution to place the current scaling request.

In this paper, we jointly consider the placement and elasticity adjustment problems for the scaling VCs, to maximize the elasticity with the constrain of communication cost in the DCN. Our contributions can be summarized as follows:

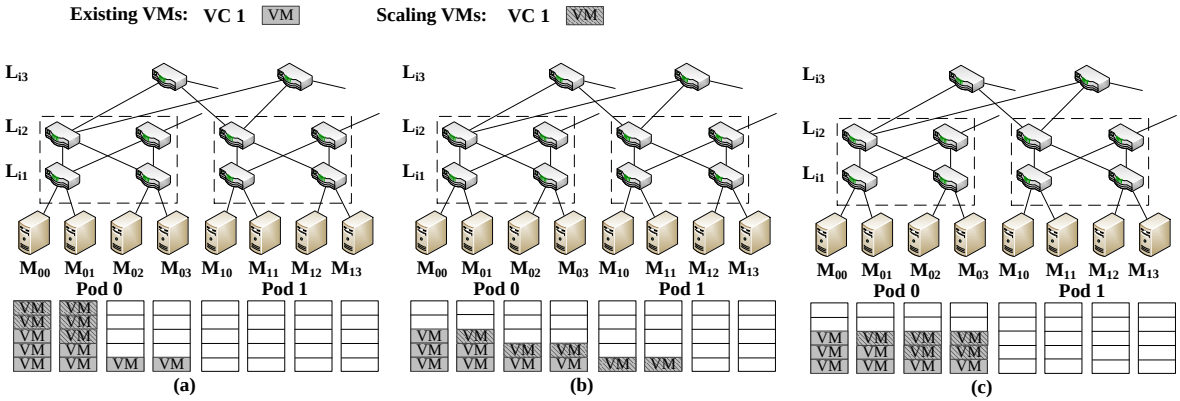


Fig. 1. An example of different placements for single virtual cluster scaling.

- We show that there is a trade-off between the elasticity and the communication cost for one VC scaling request. Given one scaling request of VC, the decreasing placement of the elasticity may lead the increasing of the communication cost. We prove the bound for the extra cost, and discuss the existence of optimal solution during the elasticity adjustment.
- We propose an algorithm VCS for the scaling request of existing VC under the constrains of resource and communication cost, and prove that it is an optimal solution.
- We extend single VC scaling placement problem into multiple ones. We also prove that it is NP-hard, and propose a heuristic algorithm MVCS for the scaling resource during one stable time period to maximize the over time elasticity of VCs.
- We conduct various simulations to compare our joint optimization method with several state-of-art ones. The results are shown from different perspectives to provide conclusions.

The remainder of this paper is organized as follows. Section II surveys related works. Section III describes the model and then formulates the problem. Section IV investigates scaling placement problem under the single VC condition, and proposes an optimal solution. Section V extend the problem into multiple VCs scaling, and proposes two heuristic algorithms for both offline and online conditions. Section VI includes the experiments. Finally, Section VII concludes the paper.

II. Related work

There have been tremendous existing works on resource allocation for VC scaling. It is a technique of crucial importance, which means that researchers must find appropriate embedding for virtual clusters in DCNs. This section provides a brief overview of the relevant methodologies proposed to address this problem. Since most researches focus on dynamically adjusting of the cluster size without considering any bandwidth guarantee targeted by current network abstractions, several methods have been proposed. [4] propose scaling up a virtual network abstraction with bandwidth guarantee. Efficient algorithms are proposed to find the valid allocation for the scaled cluster abstraction with optimization on the VM locality of the cluster. [5] propose a virtual cluster abstraction called Stochastic Virtual Cluster (SVC) to realize the bandwidth guarantee during the resource allocation. The framework and algorithms can ensure that the bandwidth demands of tenants on any link are satisfied with a high probability, while minimizing the bandwidth occupancy cost on links.

Elastic resource scaling has attracted a considerable amount of attention in cloud computing, recently [3**?**]. [6] proposes a lightweight approach to enable cost-effective elasticity for cloud applications, which realized by designing an automatic system. There are also a number of works have been proposed to scale the resources through using prediction-driven method. [7–9] employ resource demand prediction to achieve the elastic resource allocation without assuming any prior knowledge about the applications in the cloud. There is a little difference on [9], which uses VM replication to reduce application start up times.

Quite a few works consider coordinated VC scaling on both optimization of the VCs' localities and the elastic resource allocation. [10] proposes a system that allows tenants to dynamically request and update minimum guarantees for both network bandwidth and compute resources at runtime, which realized by using resource reservation method. [11] studies survivable and bandwidth-guaranteed embedding of virtual clusters, and proposes novel algorithm to jointly optimize primary and backup embeddings of the virtual clusters. These existing works on VC resource allocation fail to fully consider both localities and elasticities for the scaling requests in one determined time period. In this paper, we jointly consider the VC placement on localities and elasticities with scaling fluctuation, to maximize the over time elasticity during one time period with minimal extra cost in DCNs.

III. Model and Problem Formulation

A. Platform Model

This paper focuses on the elastic VC scaling placement problem for hose communication model on Fat-tree. We jointly consider the localities and elasticities during the resource allocation for the VCs scaling, and use elasticity to measure the

growth potential for VCs, which is also an important factor for weighting the flexibility during the placement. Our objective is to maximize the elasticity for VCs with communication cost constrain in the DCN.

B. Data Center Model

In this paper, we consider the Fat-tree as our data center network topology model. Fat-tree is an extended tree topology which has been applied to DCNs by several researchers [12]. In Fat-tree, each θ-port switch in the edge layer is connected to $\frac{\theta}{2}$ PMs [12]. Each PM in the Fat-tree is denoted as M_i, and divided into multiple slots where can be placed VMs. The capacity of each PM is denoted by C_i, and the PMs in the DCN is homogeneous. The PLs are denoted by $L = \{L_{ij}\}$, and the capacity of PL is denoted by B_{ij}. $T_{S_{ij}}$ denotes the subtree under the root (physical switch) S_{ij} which contains a set of PMs and PLs. This paper, we set call the root (physical switch) S_{ij} as the $locality$, which uses to denote the position of the virtual cluster V_i. There are two properties of the $locality$, private and public. When the property is private, the resource of the subtree $T_{S'_{ij}}$ only can be used by V_i. Otherwise, the resource under the subtree $T_{S'_{ij}}$ can be used by any other VCs when the property is public.

C. Virtual Cluster (VC)

For each tenant, the VC is an abstraction which allows each tenant to specify both the virtual machines (VMs) and per-VM bandwidth demand of its service [13]. Let V_i denote the i^{th} existing VC in the DCN, and each VC is consisted by a set of VMs and one virtual switch, where $V_i =< N_i, B_i >$. N_i is the number of VMs in the i^{th} VC, and B_i is the bandwidth demand between VMs and virtual switch. This paper, we consider the hose model based the VC abstraction. In hose model, each customer specifies a set of endpoints to be connected with common endpoint-to-endpoint performance guarantee [14].

1) Communication Cost: Since a good locality is desired for the allocation of a virtual cluster is to reduce the communication latency among VMs, we define a new function to measure the communication cost. The standard metric to evaluate the communication cost is to measure the embedding footprint [1], [15], [16]. During the VM placement, we try to minimize the communication cost. For each virtual request V_i, we define

$$m(V_i) := \sum_{j=1}^{3} \left|T_{S_{ij}}\right| \cdot H_j \cdot \gamma \tag{1}$$

$\left|T_{S_{ij}}\right|$ denotes the total amount of VMs under the subtree L_{ij} of virtual cluster V_i. H_j is the hops between PMs that VMs are located. Since the architecture of DCN is Fat-tree in this paper, the value of H_j will be $H_1 = 2$, $H_2 = 4$, $H_3 = 6$. γ is a constant value which denotes the communication cost between each pair of VMs in V_i. The communication cost of a virtual request can be calculated via the following case distinction: (1) If all VMs of V_i place into one PM, the communication cost $m(V_i) = 0$. (2) If V_i place under ToR switches or aggregation switches of a pod, the communication cost $m(V_i) = 2 * \left|S_{L^0_{i1}}\right| + 4 * \left|S_{L^1_{i1}}\right|$. (3) If V_i place under core switches of different pods, the communication cost $m(V_i) = 6 * |S_{L_{i3}}|$.

TABLE I
NOTATIONS

M_i	PM in the DCN
C_i	Capacity of the i^{th} PM in the DCN
L_{ij}	PL in the DCN
B_{ij}	Capacity of PL in the DCN
V_i	The i^{th} existing VC in the DCN
N_i	Existing VMs of V_i
N'_i	Scaling VMs of V_i
B_i	Existing bandwidth demand of V_i
δB	Scaling bandwidth demand of V_i
$T_{S_{ij}}$	Subtree of V_i under the locality S_{ij}
$R^M_{S_{ij}}$	Available computing resource in the subtree $T_{S_{ij}}$ for V_i
$R^L_{S_{ij}}$	Available communication resource in the subtree $T_{S_{ij}}$ for V_i
$m(\cdot)$	Communication cost of V_i
Φ	Upper bound for communication cost
ρ_i	Scaling ratio for V_i
ζ_i	Adjust factor of the elasticity for V_i

2) Elasticity: Let E_i denotes the elasticity of the V_i, which measures the growth potential of the V_i under the communication cost constrain. We use this factor to weight the flexibility of the placement of the VCs. The combinational elasticity is defined as $E_i = \min\{E^M_i, E^L_i\}$, where E^M_i is defined as the minimum percentage of available VM slots among PMs under the subtree $T_{S_{ij}}$ of V_i.

$$E^M_i = \min_{C_i \in T_{S_{ij}}} \{1 - \max_i \frac{C^*_i + C'_i}{C_i}\} \tag{2}$$

Similarly, E^L_i is defined as the minimum percentage of available bandwidth resource among PLs under the subtree $T_{S_{ij}}$ of V_i.

$$E^L_i = 1 - \max_{ij} \frac{f(\sum_{C_i \in T_{S_{ij}}} (C^*_i + C'_i))}{B_{ij}} \tag{3}$$

C^*_i and C'_i denote the number of existing and incoming VMs of V_i which belong to the PMs under the subtree $T_{S_{ij}}$. Similarly, $f(\sum_{C_i \in T_{S_{ij}}} (C^*_i + C'_i))$ denotes the communication demand between of the V_i under the subtree $T_{S_{ij}}$, where $f(\cdot)$ denotes the bandwidth demand function of VM communications.

D. Basic Problem Formulation

1) Definition of VC Scaling Placement: In this paper, we consider the VC placement based on hose model and Fat-tree. Let $V_i =< N_i, B_i >$ denote the i^{th} VC, which are several types of problem instances for scaling up. The basic ones are either to increase the cluster size on computing resource from N_i to $N_i + N_i$, or to increase the communication resource from B_i to δB_i. The difficult one is to increase both computing and communication resource, which ranges from N_i to $N_i + N_i$ and B_i to δB_i, i.e., $V_i =< N_i, B_i >$ to $V_i =< N_i + N'_i, \delta B_i >$. We mainly focus on the both scaling case, and the

algorithms proposed for it can also efficiently solve other two types of scaling problems.

2) Objective Function: Our objective is to maximize the elasticity for V_i with the constrain of the communication cost. We use a constant Φ to denote the upper bound of the communication cost, which initialized by the uses based on the demands of the applications [15]. Our problem can be formally formulated as follows:

$$\text{maximize } E_i \tag{4}$$

$$\text{subject to } 0 \leq m(V_i) \leq \Phi_i \tag{5}$$

$$C_i^* + C_i' \leq C_i \tag{6}$$

$$f(\sum_{C_i \in T_{S_{ij}}} (C_i^* + C_i')) \leq B_{ij} \tag{7}$$

Variables are C_i' and B_{ij}', and E_i is derived. Others are given, i.e., C_i^*, B_{ij}^* and Φ_i. Equation (5) shows the constraints on the communication cost $m(V_i)$, where its value should greater than or equal to 0 with the upper bound Φ_i. Equation (6) shows the constraints on capacities of PMs, which the total number of existing and incoming VMs C^* and C_i' on the i^{th} PM cannot exceed its capacity C_i. Equation (5) is the link capacity constraint, which shows the bandwidth usage of existing and incoming VMs on L_{ij} under the subtree $T_{S_{ij}}$ can not exceed the link capacity B_{ij}. The major notations used in this paper are listed in Table I.

IV. Single Virtual Cluster Scaling

This section proposes one optimal solution, VCS, which can be applied to deal with the scaling of single virtual cluster.

A. Algorithm and Description

1) Initialization: We take the incoming scaling request for V_i with $\langle N', \delta B\rangle$ at time slot t_i as the input, and the output is the occupation state for V_i in DCN. The initialization in line 1 is to find the locality S_{ij} of V_i, and calculate the available physical resource under the subtree $T_{S_{ij}}$, which contains computing resource $R_{S_{ij}}^M = \sum_{M_i \in T_{S_{ij}}} C_i'$ and communication resource $R_{S_{ij}}^L = \sum_{L_{ij} \in T_{S_{ij}}} B_{ij}'$, respectively. We also initialize the upper bound of the communication cost Φ_i for V_i, which usually sets by users for the QoS guarantee.

2) Virtual Cluster Scaling (VCS): For each scaling request, we try to find the appropriate subtree to obtain enough physical resources. In line 2 and 3, we first compare the incoming scaling request $\langle N', \delta B\rangle$ with the total available physical resource $R_{S_{ij}}^M$ and $R_{S_{ij}}^L$ under $T_{S_{ij}}$. If the total amount of available physical resources cannot satisfy the scaling request, the current subtree root will be positively adjusted by the step $T_{S_{ij}}$=$T_{S_{L_{i,j+1}}}$ in line 5. This process will be ended until the locality move to the upper bound S_{ij}'. We start to place scaling request in line 6 by using the function $VMP(N', \delta B)$, which describes in Algorithm 2.

Algorithm 1 Virtual Cluster Scaling (VCS)

Input: Scaling request V_i with $\langle N', \delta B\rangle$;
Output: DCN occupation state for V_i;

1: Initialize the initial locality S_{ij} and communication cost Φ_i for V_i, and calculate the available physical resources under the subtree $T_{S_{ij}}$;
2: Calculate the highest locality S_{ij}' based on the communication cost Φ_i;
3: **for** S_{ij} to S_{ij}' **do**
4: **while** $R_{S_{ij}}^M < N' \vee R_{S_{ij}}^L < N^* * \delta B$ **do**
5: $T_{S_{ij}}$=$T_{S_{L_{i,j+1}}}$;
6: $T_{S_{ij}} \leftarrow VMP(N', \delta B)$;

3) VM Placement (VMP): In this section, we propose an efficient algorithm to find the valid allocation for the scaled virtual cluster with optimization on the VMs' localities. The initialization in line 1 is to calculate the partial elasticity under the subtree $T_{S_{ij}}$, which is denoted as $E_{T_{S_{ij}}}$. In order to adjust the partial elasticity under $T_{S_{ij}}$, we define a factor ζ. Before allocating the computing resource, we check the scaling condition of communication resource in line 2, which can ensure the bandwidth B for each VM through appropriately reserving communication resource on physical links. If δ is not equal to 1, which means the communication resource has scaling or releasing, we will update the bandwidth capacities for existing VMs of V_i under the communication vary δB in line 3. After that, we compute the available computing and communication capacities under the subtree $T_{S_{ij}}$ with the limitation of $E_{T_{S_{ij}}}$ in line 4. In line 5 to line 9, we start to allocate the computing resource under the subtree $T_{S_{ij}}$, and place N' VMs into PMs based on the remaining available capacities proportionally. If the total number of available physical resources cannot satisfy the scaling request, the current partial elasticity will be negatively adjusted by the step factor ζ, as shown in lines 8 and 9. This process will be ended until the partial elasticity $E_{T_{S_{ij}}}$ is 0. In line 10, communication demands of placed VMs are evenly split into paths connecting them under the subtree $T_{S_{ij}}$.

B. Optimality Analysis

***Theorem** 1:* VCS is an optimal solution for the V_i placement under the communication cost constrain Φ.

Proof: Since the maximum number of servers in a Fat-tree is $\frac{\theta^3}{4}$, we start to proof that from $\theta = 2$, which contains 2 PMs. For the virtual cluster V_i, we first suppose that VCS is not an optimal solution, which mean there will exist another solution O be the optimal one. Since the bandwidth resource is not oversubscribed, which the bottleneck of the elasticity will existing on the computing resource. Let $\hat{C}_a$ and $\hat{C}_b$ denote the remaining available slots of PMs a and b under the subtree $T_{S_{ij}}$ of V_i. In order to place the scaling N' VMs, we assume the optimal solution O is $\{x, y\}$ $(x + y = N')$, where x VMs place on a and y VMs place on b. Similarly, we suppose the solution calculated by VCS is $\{u, v\}$ $(u + v = N')$, where u

Algorithm 2 VM Placement (VMP)

Input: Scaling request V_i with $\langle N', \delta B \rangle$;
Output: DCN occupation state for V_i;

1: Initialize E_i and adjust factor ζ_i for V_i;
2: **if** $\delta \neq 1$ **then**
3: Update the capacities of PLs for existing VMs of V_i according to the scaling $B \rightarrow \delta B$;
4: Compute $R'^{M}_{L_{ij}}$ and $R'^{L}_{L_{ij}}$ according to E_i under $T_{S_{ij}}$;
5: **while** $E_i > 0$ **do**
6: **if** $N' \leq R'^{M}_{L_{ij}}$ **then**
7: Place N' VMs into PMs in $T_{S_{ij}}$ (proportion based on the remaining available capacities of PMs);
8: **else if** $N' > R'^{M}_{L_{ij}}$ **then**
9: Update $E_i = E_i - \zeta$;
10: Communication demands of placed VMs are evenly split into paths connecting them;

VMs place on a and v VMs place on b. We suppose the $x > u$, then we will have $y < v$. The elasticity of V_i under the optimal solution O is $\min\{\frac{\hat{C}_a - x}{C}, \frac{\hat{C}_b - y}{C}\}$, if $x > y$, then the value of elasticity will be $\frac{\hat{C}_a - x}{C}$. If the elasticity under the VCS solution is $\frac{\hat{C}_a - u}{C}$, there will be $\frac{\hat{C}_a - x}{C} > \frac{\hat{C}_a - u}{C}$. However we will have $x < u$, which obeys our assumption $x > u$. Therefore, we can prove that VCS is an optimal solution when $\theta = 2$. Then we assume that VCS is optimal when $\theta = k$, we proof that it is also optimal when $\theta = k+2$. Each $k+2$ ports switches in the edge layer is connected to $\frac{k+2}{2}$ PMs. We assume that vector X is the optimal solution which $X = \{x_1, x_2, ..., x_{\frac{k}{2}}, x_{\frac{k}{2}+1}\}$. The placement for the items are different from VCS. Since X is the optimal solution, each part in this set will be optimal, which is contrary to the assumption that VCS is optimal when $\theta = k$. Therefore, we can prove that VCS is an optimal solution for the V_i placement under the communication cost constrain Φ. ■

V. Multiple Virtual Cluster Scaling

This section, we extend our work into multiple VCs scaling problem. We assume there are already existing a set of VCs V in the DCN, and each VC uses the notation V_i to denote, where $V = \{V_1, V_2, ..., V_\varpi\}$. Multiple VCs may request to scale at the same time, however, each time slot can only deal with one VC scaling request. We consider the performance of VCs in the time period $[0, T]$ by using over-time elasticity. For the offline multiple virtual cluster scaling, the value of T is the amount of the VCs ϖ. The over-time elasticity is the summation of combinational elasticities of V_i under the time slots during the whole time period $[0, T]$, i.e., $\sum_{i=0}^{T} E_i$. Since the initial distribution of the VCs are different, the processing order for the multiple VCs may lead to different result. Take Fig. 2 for example, there are existing three VCs in the DCN, which the existing amount of VMs for these users are 5, 2 and 7, respectively. The communication cost for there three VCs have already changed into the position of subtree root, which mark by cycles with different corresponding colors in

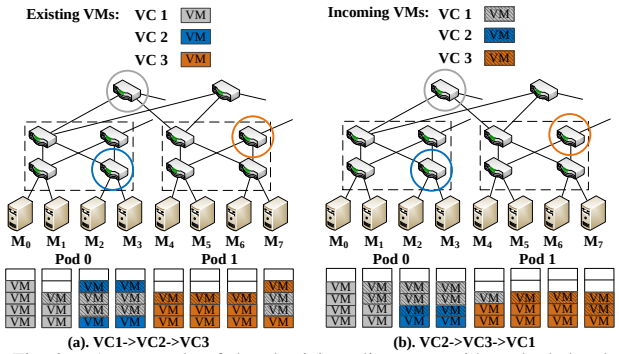


Fig. 2. An example of the elasticity adjustment without lookahead.

Fig. 2. When all these three VCs send their scaling request, we schedule them as the order VC1→VC2→VC3 as shown in Fig. 2(a). Then we will have the elasticities for VC1=$\frac{1}{5}$, VC2=$\frac{1}{5}$ and VC3=$\frac{1}{5}$, respectively. The over-time elasticity for all three VCs $\frac{1}{5} + \frac{1}{5} + \frac{1}{5} = \frac{3}{5}$. When we change the schedule order into VC2→VC3→VC1 as shown in Fig. 2(b), the elasticities for VC1=$\frac{1}{5}$, VC2=$\frac{1}{5}$ and VC3=$\frac{2}{5}$, respectively. The over-time elasticity for all three VCs under this schedule strategy will be $\frac{1}{5} + \frac{1}{5} + \frac{2}{5} = \frac{4}{5}$.

***Theorem** 2:* The MVCS placement for over-time elasticity maximization problem is NP-hard.

Proof: Given a set of scaling VCs $V = \{V_1, V_2, ..., V_\varpi\}$ Let the amount of existing VCs is ϖ, and they request to scale at the same time t. We assume the rest available resource of the DCN at t is R. The communication cost of each VC is related to the locality of its placement, which has a limitation Φ defined by the uses. The goal is to place all ϖ VCs with fewest physical resource with determined capacities under the communication cost Φ. So we reduce the original problem to the so-called variable sized bin-packing problem [17], an NP-hard problem that find an assignment that using the fewest bins. So, the MVCS placement for over-time elasticity maximization problem is NP-hard. ■

A. Algorithm and Description

Since MVCS placement is an NP-complete problem, we propose a heuristic algorithm to find a consistent scaling scheduling order that improve the over-time elasticity as higher as possible. We take the incoming scaling request set $V = \{V_1, V_2, ..., V_\varpi\}$ as the input, and the output is the occupation state for V in DCN.

The initialization in line 1 is to find the localities S_{ij} for VCs, and calculate the available physical resource under the subtree $T_{S_{ij}}$. Base on that, we initialize the scaling ratio ρ_i, which is the ratio between the scaling amount of V_i and the available physical resource under the subtree $T_{S_{ij}}$, i.e. $0 < \rho_i < 1$. After that, we initialize the upper bound of the root position S'_{ij} based on the communication cost Φ_i for VCs in the set V. In line 2, we first sort VCs in the set V to V' by localities $i = \arg\min_i S'_{ij}$ based on the communication cost Φ_i. Let V' is the sorting result of the scaling VCs. If the level of S'_{ij} are the same for the VCs, let the lower scaling ratio

Algorithm 3 Multiple Virtual Cluster Scaling (MVCS)

Input: Scaling set $V = \{V_1, V_2, ..., V_\varpi\}$;
Output: DCN occupation state for V;

1: Initialize the localities S_{ij} and S'_{ij}, ρ_i and Φ_i for VCs;
2: Sort VCs in the set V to V' by localities $i = \arg\min_i S'_{ij}$;
3: For VCs with the same localities, prioritize by scaling ratio $i = \arg\min_i \rho_i$;
4: **for** $i = 1$ to $i = \varpi$ in V' **do**
5: Place V_i into the DCN;
6: Same as Algorithm 1 form line 2 to line 5;

ρ_i one has the higher priority in line 3. From line 4 to 6, we start to place the VCs into DCN through prioritizing VCs in the set V' with the lower locality and scaling ratio ρ_i. The placement process for each VC is the same with Algorithm 1 in line 6.

VI. Online MVSC (OMVSC)

This section, we online condition for multiple VCs scaling problem. We assume the fluctuating of each VC is obey the same distribution, standard gaussian distribution. which means the incoming rate of the VCs are the same with the releasing rate. It is allowed that multiple VCs make scaling requests together, however, each time slot can only deal with one VC scaling request. We also consider the performance of VCs in one time period $[0, T]$ by using over-time elasticity.

A. Algorithm and Description

1) Initialization: We take the scaling requests V arriving at time slot t_i as the input, and the output is the occupation state for V in DCN. Since the incoming amount of scaling requests VCs at each time slot is uncertain, we will do preprocess for current arrival requests, include priority ranking and future prediction. The priority ranking for multiple scaling requests is same with offline part, which depends on the upper bound of the root position S'_{ij} and the scaling ratio ρ_i. The future prediction part is based on the bayesian parameter estimation as discussed below. Users need to set a flexible limitation c, where $0 < c < 1$, for the scaling ratio ρ_i^*. It means that the flexibility of resources under the subtree S'_{ij} should not below c.

2) Bayesian Parameter Estimation (BPS): Based on the current incoming scaling V_i request, we use bayesian parameter estimation to predict the future fluctuating statement. We use the historical fluctuating statement before t_i as our sample, which denotes as $\mathbb{I} = \{N'_i|_{i\in[0,t_i]}\}$. Let n denote the number of samples in $\mathbb{I}$. Before doing the prediction, we first calculate the mean for the sample $\mathbb{I}$, where $\mu' = \frac{1}{n}\sum_{i=1}^{n} N'_i$. Let the fluctuating of VCs with standard gaussian distribution as the Prior distribution which means $\mu_0 = 0$ and $\delta_0^2 = 1$. We assume the variance is constant, the mean for each VC will change with the time variable. Base on the bayesian parameter estimation [18], the prediction for the future fluctuating statement will be $\mu = \frac{n}{n+\delta^2}\mu'$.

Algorithm 4 Online MVCS (OMVCS)

Input: Scaling set $V = \{V_1, V_2, ..., V_\varpi\}$ at time slot t_i;
Output: DCN occupation state for V;

1: Initialize the localities S_{ij} and S'_{ij}, ρ_i and Φ_i for VCs in V arriving at time slot t_i;
2: Same as Algorithm 3 form line 2 to line 4;
3: **for** $i = 1$ to $i = \varpi$ **do**
4: Estimate the fluctuating mean μ_i for V_i based on bayesian parameter estimation;
5: Calculate the future scaling ratio ρ_i^* for V_i based on μ_i;
6: Set the locality property according to ρ_i^*;
7: Place V_i into the DCN;
8: Same as Algorithm 1 form line 2 to line 5;

3) Online MVSC (OMVSC): In line 2, the priority sorting for multiple scaling requests is the same with MVCS from line 2 to 4 in Algorithm 3. From line 3 to 7, we start to process each VC in V based on the sorting order one by one. In line 4, we estimate the fluctuating mean μ for V_i based on bayesian parameter estimation. Base on the predicted information of V_i, we calculate the future scaling ratio ρ^* for V_i, and modify the subtree properties based on it in line 5 and 6. If the $\rho_i^* > c$, we set the property of the locality S'_{ij} as private, which means the resource of the subtree $T_{S'_{ij}}$ only can be used by V_i. Otherwise, we set the property of the subtree S'_{ij} as pubic, which means the resource under the subtree $T_{S'_{ij}}$ can be used by any other VCs. In line 7, we start to place the V_i into DCN calculate the future scaling ratio ρ^* for V_i, and the placement process for each V_i is also the same with Algorithm 1 in line 8.

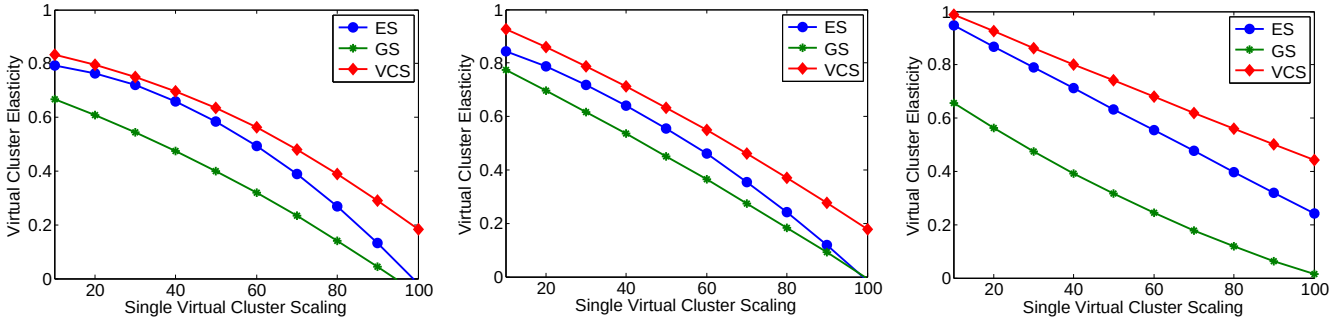
VII. Experiments

This section conducts extensive simulations to study the elastic VC scaling placement under three aspects: single VC scaling, multiple VCs scaling and online multiple VCs scaling. These experiments are conducted to evaluate the performances of the proposed algorithms. After presenting the datasets and settings, the results are shown from different perspectives to provide insightful conclusions.

A. Single Virtual Cluster Scaling

1) Experiment Setting: The DCN is modeled as Fat-tree, which the number of switches' ports are $\theta = 4$, $\theta = 6$, and $\theta = 8$. Let the amount of PMs in the Fat-tree are full connected with maximum number, which are 16, 54 and 128, respectively. The supplied computing and communication resources of the PMs and PLs were real numbers uniformly distributed between 50 and 100 units. For each group with different switch's port, we calculate the elasticity after doing the scaling placement process. The results are averaged 10 times for each algorithm. We compare the proposed VCS algorithm and two benchmark algorithms in a number of trace-driven settings.

- Equally Scaling (ES): the scaling request of V_i is evenly divided into several pieces, depending on the amount of



(a) $\theta = 4$ (b) $\theta = 6$ (c) $\theta = 8$

Fig. 3. The elasticity for single VC scaling under various Fat-trees.

PMs in the sub-tree. It can obtain the load-balance for each virtual request [1].

- Greedy Scaling (GS): the scaling request of V_i for the PMs are depending on the amount of rest available resource in the sub-tree, which PMs with high margin has a high priority.

2) Experiment Results: Fig. 3 present the elasticity for the single VC scaling condition, which the number of the switches' ports are $\theta = 4$, $\theta = 6$ and $\theta = 8$, respectively. For each group experiment, we use the same three algorithms: ES, GS, and VCS, and calculate averaged 10 times of the elasticity for various request scaling. Additionally, we have the following observations: (i). The elasticity of the scaling VC depends on the architectures of the Fat-tree. Since the construction of DCNs are based on the number of switches' ports. We can see that, the elasticity of the VC which scaling under the $\theta = 4$ switch is much lower than that of the fat-trees with $\theta = 6$ and $\theta = 8$. (ii). The elasticity for the scaling VC depends on the various placement algorithms. As shown in Fig. 3(a), Fig. 3(b) and Fig. 3(c), the performance of GS decreases significantly with the increasing amount of the scaling of VC. For ES, its performance depends on the existing localities of existing VMs. So, the fluctuation of FFRP is much larger than other algorithms. Compared with ES and GS, VCS has the best performance in the elasticity across the various VC scaling.

B. Multiple Virtual Cluster Scaling

1) Experiment Setting: This section evaluates the elasticity for the multiple VCs scaling, which use the same data set as the single VC scaling problem. Set the VMs of the VCs scale at one time slot are evenly distributed between 0 and 50, the bandwidth demands δ scale between 0 and 1. Let the switch's port is $\theta = 4$, $\theta = 6$, and $\theta = 8$ for each group. In addition to the proposed algorithms, three baseline algorithms are used:

- Randomly Schedule Scaling (RSS): the scheduling order for the multiple VCs is random.
- Decreasing Schedule Scaling (DSS): the scheduling order for the multiple VCs is decreasing.
- Increasing Schedule Scaling (ISS): the scheduling order for the multiple VCs is increasing.

2) Experiment Results: Since the scaling VMs from 0 to 50, we calculate the even value of the over-time elasticity under different δ between 0 and 1. We use the over-time elasticity to evaluate the performance of the proposed algorithm, and compare it with three base-line algorithms: RSS, DSS and ISS. Fig. 4 presents the over-time elasticity of the multiple VCs scaling by using different schedule strategies. For each time slot, we allow one VC to be processed. According to the simulation results, we have the following observations: (i). The volatility of the multiple scaling VCs is stable. As shown in Fig. 4, the mean value of under are marked by red lines, which are close with each other under different algorithms. (ii). The over-time elasticity for the multiple VCs depends on the scheduling order. Comparing these four algorithm, the performance of RSS is the worst one. The interval range of ISS is better than DSS, which depends on the distribution of the existing VMs of VCs. Compared with RSS, DSS and ISS, MVCS has the best performance in the over-time elasticity.

C. Online Multiple Virtual Cluster Scaling

1) Experiment Setting: This section evaluates the elasticity for the online multiple VCs scaling, which the arriving times of the VCs were discretionary, and the scaling amount of them were determined randomly by the tenants. We set the scaling frequency of the VCs to 1, which means that each time slot will have to process the scaling or releasing requests. We ran each of our simulations for 10 time slots intervals. The parameters and symbols that we varied in our simulations were over-time elasticity.

Since the incoming scaling VCs are online, the placement for the VCs at current time slot will be important for the incoming ones at future time slots. We compare our online multiple scaling algorithm with the one without prediction. Fig. 5 presents the comparison of the performances for these two solutions by calculating the over-time elasticity under the different distribution of existing VCs with various capacity ($\theta = 4$, $\theta = 6$, $\theta = 8$, $\theta = 10$and $\theta = 12$). For the online scaling, the uncertainty of VCs (the size and number) leads to the various interval sizes. The increasing of the switches' ports means the exponential scaling of the PMs in the Fat-tree, and the over-time elasticities for the VCs will have a large growth. As shown in Fig. 5(b), when the size of the Fat-tree is not very large ($\theta = 4$ and $\theta = 6$), the advantage of online scheduling with prediction is not obvious. When the size of the Fat-tree

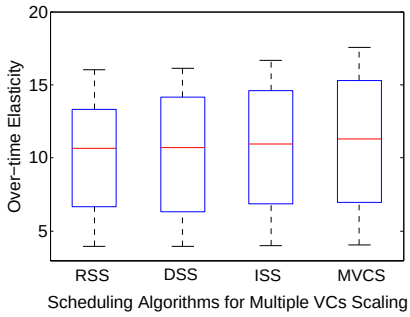
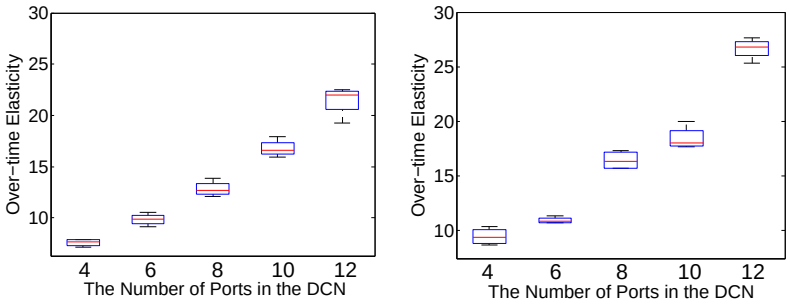


Fig. 4. The elasticity for multiple VCs scaling.



(a) Online without Prediction (b) Online with Prediction

Fig. 5. The elasticity for online multiple VCs scaling.

is scaling, such as $\theta = 8$, $\theta = 10$and $\theta = 12$, the gap between these two solutions will increase with the scale of the Fat-tree.

VIII. Conclusion

This paper considers the elastic placement for the scaling VCs which are existing in the DCN. We proposes a maximum elasticity scheme VCS which comes with provable optimality guarantees for single VC scaling. After that, we extend it into the multiple VCs scaling, and prove that multiple VCs scaling for over-time elasticity maximization problem is NP-hard. We propose heuristic algorithms MVCS and OMVCS for both offline and online conditions for the multiple VCs scaling problem. This paper focuses on the condition that VCs scaling on both computing and communication resources, which can also be adapted to each individual resource. Extensive simulations demonstrate that, our elastic VCs scaling placement schemes outperform various existing state-of-the-art methods in terms of elasticity in the DCN.

ACKNOWLEDGEMENT

This work of the first author was done during her stay as a visitor scholar by china scholarship council at Temple University. This research was supported in part by NSF and CSC grants CNS 1629746, CNS 1564128, CNS 1449860, CNS 1461932, CNS 1460971, CNS 1439672, and CSC 20163100.

References

[1] C. Fuerst, S. Schmid, L. Suresh, and P. Costa, "Online and elastic resource reservations for multi-tenant datacenters," in *IEEE INFOCOM*. Citeseer, 2016.

[2] "EC2 Variability: The numbers revealed, Measuring EC2 system performance," http://goo.gl/V5zhEd, 2009, [Online; accessed May 13, 2009].

[3] K. Li, J. Wu, and A. Blaisse, "Elasticity-aware virtual machine placement for cloud datacenters," in *Cloud Networking (CloudNet), 2013 IEEE 2nd International Conference on*. IEEE, 2013, pp. 99–107.

[4] L. Yu and Z. Cai, "Dynamic scaling of virtual clusters with bandwidth guarantee in cloud datacenters," in *Computer Communications, IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on*. IEEE, 2016, pp. 1–9.

[5] L. Yu and H. Shen, "Bandwidth guarantee under demand uncertainty in multi-tenant clouds," in *Distributed Computing Systems (ICDCS), 2014 IEEE 34th International Conference on*. IEEE, 2014, pp. 258–267.

[6] R. Han, L. Guo, M. M. Ghanem, and Y. Guo, "Lightweight resource scaling for cloud applications," in *Cluster, Cloud and Grid Computing (CCGrid), 2012 12th IEEE/ACM International Symposium on*. IEEE, 2012, pp. 644–651.

[7] Z. Shen, S. Subbiah, X. Gu, and J. Wilkes, "Cloudscale: elastic resource scaling for multi-tenant cloud systems," in *Proceedings of the 2nd ACM Symposium on Cloud Computing*. ACM, 2011, p. 5.

[8] Z. Gong, X. Gu, and J. Wilkes, "Press: Predictive elastic resource scaling for cloud systems," in *Network and Service Management (CNSM), 2010 International Conference on*. Ieee, 2010, pp. 9–16.

[9] H. Nguyen, Z. Shen, X. Gu, S. Subbiah, and J. Wilkes, "Agile: Elastic distributed resource scaling for infrastructure-as-a-service." in *ICAC*, vol. 13, 2013, pp. 69–82.

[10] C. Fuerst, S. Schmid, L. Suresh, and P. Costa, "Kraken: Online and elastic resource reservations for multi-tenant datacenters," in *Computer Communications, IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on*. IEEE, 2016, pp. 1–9.

[11] R. Yu, G. Xue, X. Zhang, and D. Li, "Survivable and bandwidth-guaranteed embedding of virtual clusters in cloud data centers," in *IEEE INFOCOM*, 2017.

[12] M. Al-Fares, A. Loukissas, and A. Vahdat, "A scalable, commodity data center network architecture," in *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 4. ACM, 2008, pp. 63–74.

[13] R. Yu, G. Xue, X. Zhang, and D. Li, "Survivable and bandwidth-guaranteed embedding of virtual clusters in cloud data centers (extended version)," *arXiv preprint arXiv:1612.06507*, 2016.

[14] N. G. Duffield, P. Goyal, A. Greenberg, P. Mishra, K. K. Ramakrishnan, and J. E. van der Merive, "A flexible model for resource management in virtual private networks," in *ACM SIGCOMM Computer Communication Review*, vol. 29, no. 4. ACM, 1999, pp. 95–108.

[15] H. Ballani, P. Costa, T. Karagiannis, and A. Rowstron, "Towards predictable datacenter networks," in *ACM SIGCOMM Computer Communication Review*, vol. 41, no. 4. ACM, 2011, pp. 242–253.

[16] D. Xie, N. Ding, Y. C. Hu, and R. Kompella, "The only constant is change: incorporating time-varying network reservations in data centers," *ACM SIGCOMM Computer Communication Review*, vol. 42, no. 4, pp. 199–210, 2012.

[17] D. K. Friesen and M. A. Langston, "Variable sized bin packing," *SIAM journal on computing*, vol. 15, no. 1, pp. 222–230, 1986.

[18] K. Beven and A. Binley, "The future of distributed models: model calibration and uncertainty prediction," *Hydrological processes*, vol. 6, no. 3, pp. 279–298, 1992.