VMatch: A Matching Theory Based VDC Reconfiguration Strategy

Anurag Satpathy, Manmath Narayan Sahoo, Lucky Behera, Chittaranjan Swain, Ashutosh Mishra
Department of Computer Science and Engineering
National Institute of Technology, Rourkela, India.
{anurag.satpathy, vickybehera332, chittaiit42, ashutosh11mishra1}@gmail.com, sahoom@nitrkl.ac.in

***Abstract*—A virtual data center (VDC) mostly encapsulates multiple virtual machines (VMs) with communication dependencies. These VDC requests are dynamic in nature and often experience fluctuating demands across different resources. In this paper, we propose a dynamic resource reconfiguration strategy called *VMatch* that generates an efficient relocation/remapping plan for already assigned virtual links (VLs) facing bandwidth expansion. The overall problem is formulated as a one-to-one matching game that aims to minimize the relocation cost from the users perspective and at the same time improves resource utilization from a service providers (SPs) perspective. By using the concept of preferences in the matching game, different stakeholders, i.e., end-users and SPs express their priorities. Thorough simulation analysis of the proposed approach shows that the model on an average can reduce the remapping cost by 19% and improve server utilization by 21% in comparison with the baselines.**

***Index Terms*—Cloud; Virtual Data Center; Reconfiguration Cost; Matching Game; Gale Shapley.**

I. Introduction

Data center (DC) virtualization technologies allow the service providers (SPs) to logically partition its resources into independent and isolated entities termed as virtual data centers (VDCs) [1]. A VDC request as depicted in Figure 1 consists of multiple virtual components, i.e., virtual machines (VMs) and virtual links (VLs) having disparate resource demands. Referring to Figure 1, numbers inside the rectangle correspond to CPU and memory requirements of a VM, whereas values next to the VLs correspond to minimum bandwidth demand for two communicating VMs. Most of the existing research [1]–[3] focused on resource management of VDCs have considered a static deployment scenario viz., resource demands for VDC components do not change over time. However, Sun *et al.* [4] pointed out that for some practical use cases, such as distributed computing and online gaming, users often experience time varying demands for substrate[1] resources. Further, Dab *et al.* [5] emphasized on the fact that such applications frequently experience the issue of bandwidth expansion for already assigned VLs of VDC requests.

In this paper, we propose a model called *VMatch* based on matching theory aimed at building a dynamic remapping plan for already assigned VLs facing bandwidth expansion. Bandwidth expansion often leads to relocation[2] of virtual components, and in some cases encompasses reconfiguring both VMs and its attached VLs. We specifically focus on such a use case where a remapping involves relocating a VM and its associated VLs, collectively referred to as a solution component (SC). Remapping in such scenario introduces some complex challenges as the involved stakeholders often have contrasting goals. For instance, an end-user would ideally want to minimizes its remapping cost whereas a SP would prefer to maximize the utilization of its substrate resources. Therefore, it is essential to build a solution that considers the interest of different stakeholders while generating a remapping. In this context, matching theory is an elegant solution concept that captures the design objectives of different stakeholders by using the concept of *preferences* [6]. Further, the concept of *stability* depicts the efficiency and fairness of allocation. Although some solution approaches [5], [7] have addressed similar problems using optimization techniques but it has its own pitfalls. Firstly, optimization techniques are incapable of considering contrasting objectives of stakeholders that do not align well with the system-wide objectives [8]. Secondly, the optimization solvers are computationally intensive and are not scalable.

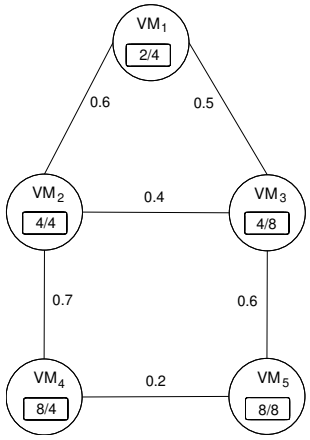

Fig. 1. A VDC request with five VMs and six VLs.

The primary agenda of *VMatch* is to reduce the remapping cost for end-users and maximize utilization of resources for SPs without demeaning its overall revenue. *VMatch* proceeds as follows: On receiving a bandwidth expansion request,

[1]The term DC and substrate refer to the same thing.

[2]We interchangeably use the terms relocation, reassignment, reconfiguration and remapping.

2159-6190/20/$31.00 ©2020 IEEE
DOI 10.1109/CLOUD49709.2020.00031

VMatch generates set of star topologies [5] also referred to as SCs. Each SC comprises a central VM and VLs that are directly attached to it with at least one link facing bandwidth expansion. Next, *VMatch* constructs the preferences of the stakeholders, i.e., users and SPs adhering to the objectives as discussed earlier. After establishing the preferences *VMatch* performs a one-to-one stable matching based on a modified version of the classical deferred acceptance algorithm [9]. In a nutshell the contributions of the paper can be highlighted as follows:

1) We propose a model called *VMatch* to address the problem of bandwidth expansion of VLs for already assigned VDC requests. The overall agenda of *VMatch* is to minimize the remapping cost from an end-users perspective while maximize the utilization of substrate resources with minimum impact on overall revenue from a SPs standpoint.
2) The overall problem is formulated as a one-to-one matching game. Preferences are generated keeping both the stakeholders, i.e., users and SPs into consideration. As multiple criteria are involved in decision making we use analytical hierarchy process (AHP) to prioritize the alternatives while generating preferences.
3) To perform stable allocation via matching we propose a modified version of the classical deferred acceptance algorithm.
4) To evaluate the performance of *VMatch*, we perform extensive simulation and compare the results obtained with two baselines: VNR-GA [5] and EVPF [10]. Simulation results show that on an average *VMatch* is able to reduce the remapping cost by 19% and improve the server utilization by 21% in comparison with the baselines. Further, results also confirm the fact that overall hosting cost of VDCs and corresponding bandwidth utilization of the substrate network is almost at par with the baselines.

The rest of the paper is organized as follows: In Section II, work reported in the literature are discussed. In Section III, we discuss the system model in detail. Section IV talks about the proposed one-to-one matching game. Performance analysis of *VMatch* is discussed in Section V and conclusions are drawn in Section VI.

II. Related Work

Resource allocation of VDC components with constraints is polynomial time reducible to the *NP-Hard* multi-way separator problem [11]. For static deployment scenarios, Sun *et al.* [1] proposed a online live migration strategy for relocating VDC requests across Geo-distributed DCs. Their primary agenda is to reduce the remapping cost, blocking ratio and migration overheads. On the other hand, Metwally *et al.* [12] discussed two distributed resource provisioning schemes for VDC requests based on auction-theory. To maximize the revenue of SPs in an eco-friendly setting, Amokrane *et al.* [13] presented a resource management framework called *Greenhead* for provisioning resources across Geo-distributed DCs interconnected via a backbone network. Chowdhury *et al.* [2] highlighted the fact that a better coordination between VM and VL mapping can help in reducing resource mapping cost for static virtual networks (VNs). Although the above solution concepts were able to address the issue of resource allocation for VDCs but they were limited to static deployment scenarios.

Majority of the literature that we have reviewed are focused on static VDC deployment [1] [2] [3] [12] [13] [14]. However, Dab *et al.* [5] pointed out that VDC components especially VLs often face bandwidth expansion. This excessive bandwidth demand can be handled by relocating VDC components. This relocation is a challenging problem and is also proven to be NP-Hard [11]. Some works focused on handling dynamic VDC requests are discussed in [4] [5]. Specifically, authors in [4] have solved the dynamic remapping solution using a greedy approach to minimize remapping cost. A major pitfall of this approach is that greedy approach may not always lead to an efficient solution. On the other hand, authors in [5] modelled the bandwidth expansion as an optimization problem and solved it using genetic algorithm (GA). However, this solution not only fails to consider preferences of different stakeholders but is also computationally expensive. Further GA based solutions do not scale well. To overcome such drawbacks, we in this paper, propose a matching theory based approach to remap VLs that face bandwidth expansion. The objectives of stakeholders are captured as preferences and the efficiency is expressed in the form of stability. In the following section we discuss the system model in detail.

III. System Model and Assumptions

In this section, we discuss the problem of relocating SCs across Geo-distributed DCs facing bandwidth expansion. Next, we formally define the entities involved in relocation process and subsequently formulate the overall problem.

A. VDC Request

A VDC request is modelled as a weighted undirected graph $G^V = (N^V, L^V)$, where $N^V = \{v_1, v_2, \cdots, v_n\}$ represents the set of VMs, and n denotes the total number of VMs. The set of VLs are designated by $L^V = \{e_1, e_2, \cdots, e_m\}$, where m is the total number of VLs. A typical VDC request graph as depicted in Figure 1, consists of multiple virtual entities, viz., VMs and VLs having different resource demands across different dimensions. A VM often demands CPU and memory resources on the server hosting it. For simplicity, in this paper, we assume that a VM resource demand is expressed in terms of computational resource blocks (CRBs). A CRB is considered as a basic unit of computation [8]. For instance, if a VM demands one CRB its request essentially corresponds to 1 CPU core and 512 MB of memory, and similarly a VM requesting a CRB of size 2 implies 2 CPU cores and 1 GB memory. In fact, such sizing policies are widely adopted by large scale SPs [15]. Therefore, for a given VM v_i, $v_i \in N^V$; $i \in \{1, 2, \cdots, n\}$, its CRB demand can be represented as $d(v_i)$. The initial bandwidth demand for a VL e_j, $e_j \in L^V$; $j \in \{1, 2, \cdots, m\}$, is expressed as $b(e_j)$.

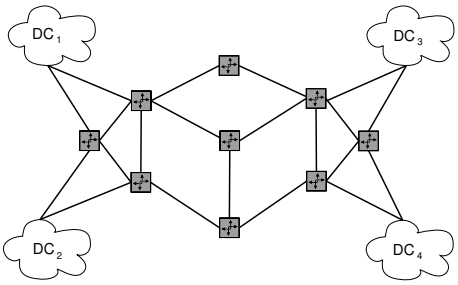

Fig. 2. Substrate Network.

B. Substrate Network

The substrate network is also modelled as an undirected weighted graph $G^S = (N^S, L^S)$. The set of nodes $N^S = N^R \cup N^H$, where N^R and N^H denote the set of routers/switches and hosts respectively. The set of substrate links are represented by L^S. Each host $h_k \in N^H$ has a maximum capacity $cap(h_k)$ and cost per CRB $c(h_k)$. Note that to add heterogeneity to our model, the maximum capacity and cost of using a unit CRB is dissimilar across servers. Similarly, each physical link $e_s \in L^S$ is characterised by its capacity $cap(e_s)$ and unit bandwidth cost $c(e_s)$. A typical illustration of a Geo-distributed infrastructure interconnected via a backbone network is shown in Figure 2.

C. Relocation of Solution Components

In this paper, we assume a scenario where VDC components experience dynamic resource demands. Precisely, we deal with increasing bandwidth demands of already assigned VLs of VDCs [5]. To deal with such scenarios we are often faced with the challenge of relocating solution components (SCs). A SC is defined as a star topology with a central VM and all its directly connected VLs having at least one VL facing bandwidth expansion. For instance, corresponding to a VDC request shown in Figure 3, a SC s_3 centred at v_3 is depicted as per Figure 4. The SC s_3 comprises a central VM v_3 and its VLs connecting v_1, v_2, v_4 and v_5 respectively. The VLs coloured red are the one's experiencing bandwidth expansion and requesting additional resources. Relocating SCs is a more sensible option

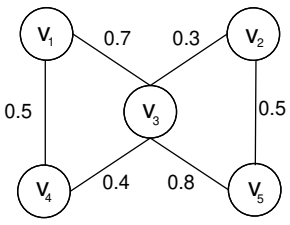

Fig. 3. An example of a VDC Request graph.

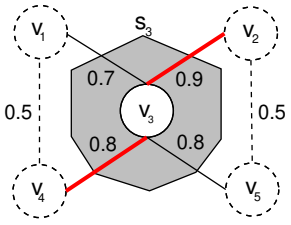

Fig. 4. Solution Component s_3 centred at v_3.

to handle the excess bandwidth demands in comparison to relocating an entire VDC request [5]. For a VDC request G^V, VLs requesting excess bandwidth demands are denoted by a $\langle VL, bandwidth \rangle$ pair; $E_{G^V} = \{\langle e_j, b'(e_j)\rangle\}$, where $e_j \in L^V$ and $b'(e_j) > b(e_j)$ represents its updated bandwidth demand. As discussed earlier, the overall agenda of *VMatch* is to reduce remapping cost of SCs and maximize utilization of substrate resources without compromising much with the overall revenue. Next, we discuss the problem formulation in detail.

D. Problem Formulation

Given a VDC request G^V, we assume E_{G^V} to be a set of VLs experiencing bandwidth expansion and needing relocation. Based on the above information, we construct $R_{G^V} = \{s_1, s_2, \cdots, s_l\}$ to be a set of SCs of G^V requiring reconfiguration. Therefore an efficient remapping plan is one where relocation of SCs is achieved at minimum cost from a users perspective and higher utilization of resources is attained from a SPs standpoint. The remapping of s_i encompasses the remapping of *(a.)* a central VM v_i and *(b.)* set of VLs attached with v_i represented by the set $L_i = \{\langle e_j, b(e_j)\rangle\}_{\forall e_j \in s_i}$. Note that L_i includes all VLs attached to v_i irrespective of its condition, i.e., facing or not facing expansion.

To find a feasible remapping of s_i, the first step involves finding a remap for the VM v_i centred at s_i. The remapping cost of assigning v_i to a destination host h_k can be computed as per Equation (1).

$$c(v_i \rightarrow h_k) = d(v_i) * c(h_k) \tag{1}$$

The VM remapping is possible if the destination server has enough resources in the form of CRBs, and this constraint is captured by Equation (2). Here, $A(h_k)$ denotes the current availability of server h_k in terms of CRBs.

$$A(h_k) \geq d(v_i) \tag{2}$$

The second step involves relocating all the VLs associated with s_i. Referring to Figure 4, a complete remapping of s_3 involves remapping four VLs, two of which have excess bandwidth demands (coloured red) and remaining two VLs (coloured black) that do not require extra resources but are remapped anyway due to the relocation of v_3. For instance Figure 5 shows that relocating v_3 from a server $h_{k'}$ to h_k implies finding a new substrate path (coloured blue) to map the VL connecting v_3 and v_2. Therefore, remapping a SC implies remapping two kinds of VLs, viz., VLs with excess demand and VLs without any excess demand. Equation (3) captures the VL remapping cost corresponding to s_i. The first term of equation portrays the remapping cost of VLs facing excess demand. The second term, however deals with the remapping cost of VLs that do not have excess demands but require a remap due to relocation of v_i. $p(h_k, h_z)$ depicts the substrate path corresponding to a VL between v_i and $v_{i'}$ such that $v_{i'} \in Adj(v_i)$ and v_i and $v_{i'}$ are mapped to hosts h_k and h_z respectively. The cost of using a unit bandwidth resource of substrate link e_t is denoted by $c(e_t)$.

$$\begin{aligned} LC(s_i) = & \sum_{e_j \in (L_i \cap E_{G^V})} \sum_{e_t \in p(h_k, h_z)} b'(e_j) * c(e_t) \\ & + \sum_{e_j \in (L_i \setminus E_{G^V})} \sum_{e_t \in p(h_k, h_z)} b(e_j) * c(e_t) \end{aligned} \tag{3}$$

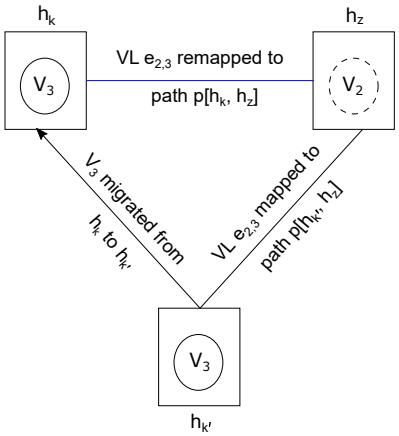

Fig. 5. Remapping v_3 from $h_{k'}$ to h_k implies remapping the VL $e_{2,3}$.

The VL relocation is subject to certain constraints as depicted in Equation (4). The first and second constraints make sure that physical links to which VLs are mapped must have enough bandwidth resources for hosting both categorises of VLs of s_i. The third and fourth constraints enforce the fact that adjacent VMs to v_i must have a valid mapping. $S(v_i)$ returns the server onto which v_i is mapped. A SC is considered to be successfully remapped iff all its virtual entities are assigned resources.

$$\begin{gathered} b'(e_j) \geq b(e_t) \\ b(e_j) \geq b(e_t) \\ \forall\, e_t \in p(h_k,\ h_z);\ (h_k,\ h_z) \in N^H \\ \forall\, v_{i'} \in Adj(v_i),\ S(v_{i'}) = h_z \end{gathered} \tag{4}$$

The aggregate cost of remapping s_i can be calculated by combining Equations (1) and (3).

$$c(s_i,\ h_k) = c(v_i \rightarrow h_k) + LC(s_i) \tag{5}$$

From a SPs perspective it would ideally like to maximize its resource utilization without compromising much with the overall revenue obtained by hosting VDC requests. In this paper, we specifically focus on maximizing the server utilization however, the model can be extended to take other substrate resources such as bandwidth into consideration. Since, we consider variable costs of using server resources, maximizing server utilization may not always lead to an improved revenue. Hence, there is a need to develop an ingenious strategy that is able to balance both server usage and deployment costs while constructing a remapping plan for SCs. The number of available CRBs of a server h_k can be calculated as Equation (6). $y_{i,k} \in \{0, 1\}$ is a binary variable and is set to 1 if a VM v_i is assigned to host h_k, otherwise 0.

$$A(h_k) = cap(h_k) - \sum_{G^V} \sum_{\forall\, v_i \in N^V} y_{i,k} * d(v_i)\,;\quad \forall k \in \{1, 2, ..., |N^H|\} \tag{6}$$

The overall utility of a host h_k obtained by assigning v_i corresponding to s_i is computed from Equations (1) and (6) and is expressed as Equation (7). The weights w_1 and w_2 in Equation (7) are obtained using analytical hierarchy process (AHP) which is discussed in detail in Section IV. To improve the overall utilization of hosts, allocation of v_i corresponding to s_i should be performed keeping in view its current utilization level. Therefore we assign a higher values to hosts that are under-utilized, i.e., have higher available resources thereby improving overall utilization levels. We also consider the VM hosting cost while calculating the utility of a host which impacts the overall revenue obtained. Hence, Equation (7) accurately captures the design objectives from a SPs standpoint.

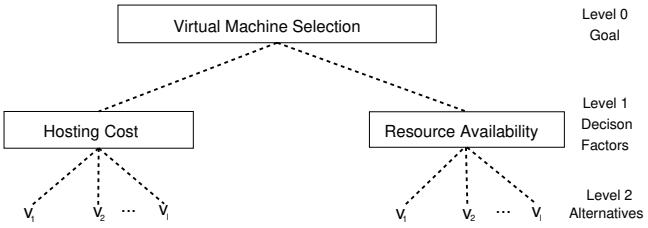

Fig. 6. AHP hierarchy for VM selection.

$$U_{i,k} = w_1.A(h_k) + w_2.(c(v_i \rightarrow h_k)) \tag{7}$$

The overall objective of *VMatch* can be represented as per Equation (8). The first constraint makes sure that a VM v_i of a SC s_i can be assigned to a server h_k if it has sufficient resources in terms of CRBs. The following two constraints state that the remapped VLs should have enough bandwidth resources on each physical links on its remapped path. The fourth constraint asserts the fact that every VM of a VDC must be mapped to a substrate server. Moreover, the next constraint states that no two VMs of the same VDC can be mapped onto the same server. The final two constraints ascertain the valid set of values the decision variables can take. In the next section, we discuss the matching theory based solution approach in detail.

$$\min\left(\textstyle\sum_{\forall\, s_i \in R_{G^V}} c(s_i, h_k)\ and \sum_{\substack{\forall\, v_i \in s_i \\ \forall\, h_k \in N^H}} 1/U_{i,k}\right)$$

$$s.t. \begin{cases} A(h_k) \geq d(v_i) \\ b'(e_j) \geq b(e_t) \\ b(e_j) \geq b(e_t) \\ \sum_{k=1}^{|N^H|} y_{i,k} = 1 \\ y_{i,k} \neq y_{i',k} \\ \forall\,(i,\, i') \in \{1,\, 2,\, \cdots,\, n\};\ \forall k, z \in \{1,\, 2,\, \ldots,\, |N^H|\} \\ \forall j \in \{1,\, 2,\, \cdots,\, m\};\ \forall\, e_j \in s_i;\ \forall\, e_t \in p(h_k, h_z) \end{cases} \tag{8}$$

IV. Solution Approach

Matching theory has recently emerged as an appropriate framework to perform association between two distinct sets

TABLE I
PAIRWISE COMPARISON SCALE [17]

Relative Importance	Description
1	Equally preferred.
3	Moderately preferred.
5	Strongly preferred.
7	Very strongly preferred.
9	Absolutely preferred.
2, 4, 6, 8	Intermediate preference values between adjacent scales.
Reciprocals of above scales	if $p_{i,j} = x$ then $p_{j,i} = 1/x$.

taking individual preferences of each element into consideration [16]. The preferences of each individual expresses the level of satisfaction obtained in the matching process. Further, to address the issue of fairness of allocation, the notion of *stability* is also defined [8]. *VMatch* aims to address the problem of SC relocation in a way that not only reduces the remapping cost for end-users but also improves the utilization of substrate resources (specifically hosts) with minimum impact on revenue. *VMatch* utilizes a matching theory based framework to elegantly capture the diverse agendas of stakeholders (users and SPs) participating in the matching process. Through the concept of preferences stakeholders express their policies using a simple ranking ordered list. As SPs have multiple contrasting criteria involved in the decision making process we utilize an analytical hierarchical process (AHP) based technique to assign preferences. Next, we discuss the working of AHP in detail.

A. Ranking based on AHP

As we already discussed in Section III, relocating SCs should be focused at improving the host utilization with minimum impact on revenue from a SPs standpoint. We have specifically focused our attention on substrate hosts as they are expensive and are a primary component of revenue generation in comparison to other resources. However, *VMatch* can easily be extended to incorporate parameters concerning other resources, i.e., bandwidth in the decision making process. The overall selection problem is decomposed into hierarchical sub-problems as illustrated in Figure 6. In level 0, the goal of AHP is selection of a candidate VM corresponding to a SC. Level 1 consists of decision factors or criteria and finally level 2 assess the alternatives based on the evaluation of decision factors at level 1. AHP works as per the following steps: Firstly, we construct an evaluation matrix $E \in R^{|R_{G^V}| \times |C|}$ where an entry $e_{l,c}$ represents the value of v_l corresponding to s_l for a criteria $c \in C$, where C denotes the set of all criteria. Next, we create a pairwise comparison matrix $P \in R^{|C| \times |C|}$ based on the relative importance of the criteria. Each entry $p_{q,s} \in P$ depicts the relative importance of criteria q against s. The relative importance of criteria are set as per the subjective judgement scale expressed in Table I. This judgement is based on the level of importance of criteria enforced by the stakeholders. Although different variants of the judgement scales are available we have considered the linear scale due to its superiority in comparison to other scales [17]. Next, we perform normalization of the pairwise comparison matrix P to obtain the weights of criteria as per Algorithm 1. The pairwise comparison matrix P is column-wise normalized as depicted by lines 2-6 (Algorithm 1). The normalized matrix represented as $\bar{P}$ is subsequently averaged row-wise to obtain the weights of criteria (Refer to lines 7-9 of Algorithm 1). The weight vector $W = \{w_1, w_2, ..., w_{|C|}\}^T$ is the principal Eigen vector that represents weights at level 1 of AHP process. As we deal with two criteria the weights obtained via AHP are considered to be consistent and verification via *consistency index* is not necessary [18]. Next, we determine the global weight vector/rank $R_g = \{r_1, r_2, ..., r_l\}^T$ that depicts the preferences list of hosts corresponding to every VM of a SC and is obtained via Algorithm 2. The local weight vector W obtained from Algorithm 1 is multiplied with the corresponding entries in $E \in R^{l \times c}$ matrix to obtain the global weights that are used to rank the VMs.

Algorithm 1: Pairwise Comparison Matrix Normalization.

Input: $P \in R^{|C| \times |C|}$
Result: W
1 **Initialize** : $\bar{P} \in R^{|C| \times |C|}$, $c = 1$
2 **while** $c \neq |C|$ **do**
3 $\bar{a}[c] \leftarrow \sum_{b=1}^{|C|} P[b][c]$
4 **for** *every* b *in the* c^{th} *column of* P **do**
5 $\bar{P}[b][c] = \frac{P[b][c]}{\bar{a}[c]}$
6 $c = c + 1$
7 **while** $c \neq |C|$ **do**
8 $w_c \leftarrow \frac{1}{|C|} \sum_{b=1}^{|C|} \bar{P}[c][b]$
9 $W = W \cup w_c$

Algorithm 2: Rank Determination.

Input: W, $E \in R^{|R_{G^V}| \times |C|}$
Result: R_g
1 **for** $s_i \in R_{G_V}$ **do**
2 **for** *each* c^{th} *criterion of the* i^{th} *candidate* **do**
3 $X[i][c] = E[i][c] * w_c$
4 $r_g \leftarrow \sum_{b=1}^{|C|} X[i][b]$
5 $R_g = R_g \cup r_g$

B. Resource Allocation using a Matching Game

In order to allocate resources to SCs facing excess resource demand, we propose a stable matching based association between two sets $V = \{v_1, v_2, \cdots, v_l\}$ and $N^H = \{h_1, h_2, \cdots, h_{|N^H|}\}$, where an entry $v_i \in V$ corresponds to the central VM of s_i and $h_k \in N^H$ corresponds to a host in the substrate network. The association is modelled as a one-to-one matching game considering the individual preferences of each player. As multiple VMs of a VDC cannot be matched to the same server [4], the selection game is modelled as a one-to-one matching game. Formally, the matching game is interpreted as per Definition 1.

Definition 1. *Let V and N^H be two sets of players. A matching game defined over (V, N^H) has two preference*

Algorithm 3: Modified Gale-Shapley.

Input: P_V, P_{N^H}
Result: $\mu : V \rightarrow N^H$
1 **Initialize**: $v_i \in V$ and $h_k \in N^H$ as free.
2 **while** $\exists\, v_i$ *such that* v_i *is unassigned and* $P_{v_i} \neq \phi$ **do**
3 h_{k^*} = highest ranked server h_k such that v_i has not proposed to h_k.
4 Send proposal to h_{k^*}.
5 Set Flag = false;
6 **if** $|\mu(h_{k^*})| \neq 1$ **then**
7 Flag = true;
8 **else if** h_{k^*} *is matched to* $v_{i'}$ **then**
9 **if** h_{k^*} *prefers* $v_{i'}$ *to* v_i **then**
10 Flag = false;
11 **else**
12 Flag = true;
13 Set $v_{i'}$ as free and release resources.
14 **if** *flag!= false* **then**
15 Set X=0
16 **while** $\exists\ e_j \in L_{i^*}$ *that is unassigned and has a feasible path* **do**
17 X++;
18 **if** $X == |Adj(v_{i*})|$ & $d(v_i) \leq A\ (h_{k^*})$ **then**
19 Match v_i and h_{k^*}.
20 **end while**

relations $\succ_i$ *and* $\succ_k$ *that allows each player, i.e., a VM* $v_i \in V$ *to indicate preference over players in the opposite set, i.e.,* $\forall k \in N^H$ *and vice-versa.*

Definition 2. *The outcome of a matching game is a matching function* $\mu : V \times N^H \rightarrow V \times N^H$ *such that:*

(a.) $\mu(v_i) \in N^H \ \&\ |\mu(v_i)| = 1$
(b.) $\mu(h_k) \in V \ \&|\mu(h_k)| = 1$
(c.) $\mu(v_i) = h_k \Leftrightarrow \mu(h_k) = v_i$
(d.) $b'(e_j) \geq b(e_k)\,;\ \forall e_k \in p(\mu(v_i), \mu(v_{i'}))\,;\ \forall\, e_j \in L_i;\ (v_i, v_{i'}) \in V;\ \forall v_{i'} \in Adj(v_i),\ i \neq i',\ |\mu(v_{i'})| = 1$

The first condition highlights the fact that a VM is matched to only one host. The second condition states that a host is matched to only one VM of a VDC. The third condition states that a VM is matched onto a host iff the host is matched to that VM. Remapping SC centered at a VM involves remapping all its VLs satisfying their respective bandwidth demands and this is captured in the final condition.

Definition 3. *A matching* μ *is said to be individually rational for all players iff there is no player* f *that prefers to remain unmatched than being matched to* $\mu(f)$.

This implies that the matched partner in the matching process should not be unacceptable for the player, i.e., a player should not prefer being unmatched than its matched partner. Next, we discuss the generation of preference profile for the players in the matching game.

C. Preference Profile of Players

The matching game as per Definition 1 is between a set of VMs V and a set of hosts N^V. Each player has a strict and transitive preference over players of the other set. The preference relation of the players $P_{v_i}, \forall v_i \in V$ and $P_{h_k}, \forall h_k \in N^H$ are elucidated as per Definition 4.

Definition 4. *A matching game is defined over two sets* V *and* N^H*, where* $\succ_i$ *and* $\succ_k$ *for each player* $v_i \in V$ *and* $h_k \in N^H$ *respectively is used to denote the preferences over the entries in the other set.*

In this matching game, the preference profile for $v_i \in V$ corresponding to s_i is denoted by P_{v_i}. This preference profile of v_i is incomplete and does not include two types of hosts, viz., (*a.*) the current matching of v_i, and (*b.*) matching of all $v_{i'}$ not part of V. It makes sense not to include the current matched host of v_i in P_{v_i} as v_i will never be relocated to the same host. Further, the second category of hosts are the ones hosting VMs of a VDC that are not involved in the remapping process. As no two VMs of a VDC can be matched to the same host they are no longer feasible options. Therefore the ranked preference $\succ_i$ over the feasible hosts is expressed as per Equation (9).

$$h_{k'} \succ_i h_k \Leftrightarrow c(s_i, h_k) > c(s_i, h_{k'}); \quad k \neq k' \tag{9}$$

Likewise each host $h_k \in N^H$ generates a preference profile over VMs corresponding to SCs excluding a VM $v_{i'}$ not in V and matched to h_k. Therefore the preference profile is represented as per Equation (7):

$$v_i \succ_k v_{i'} \Leftrightarrow U_{i,k} > U_{i',k}; \quad i \neq i' \tag{10}$$

The overall working of matching algorithm is depicted in Algorithm 3. The input to the algorithm is the set of preferences profile of players, i.e., VMs and hosts. The output of the algorithm is a stable allocation that not only assigns VMs to hosts but also assigns resources to VLs attached to the relocated VMs, thereby relocating SCs. The preference profile of the players is expressed in the form of a ranked list as discussed earlier. We also use a Flag to keep track of a VM successful assignment. Steps 2 - 19 denote the overall matching process. The while loop of statement 2 continues to execute till there is an unassigned VM with servers left to propose. A VM v_i selects its most preferred host h_{k^*} to which it has not proposed yet and send out a proposal. Once the server receives a proposal one of the two cases may arise: (*a.*) the server has no VM matched to it (Steps 6 - 7) or (*b.*) the server already has a VM matched to it (Steps 8 - 13). The latter has two cases: (b.1) where the server h_{k^*} prefers the already assigned $v_{i'}$ over the proposing v_i, hence, h_{k^*} turns down the proposal and the flag is set to false, (b.2) where h_{k^*} prefers v_i over the already assigned $v_{i'}$, hence, the match between h_{k^*} and $v_{i'}$ is removed from the matching and $v_{i'}$ is set to free. Steps 14 - 19 perform the match between v_i and h_{k^*} if and only if the server has enough resources in terms of CRBs and the relocation of VM can satisfy the bandwidth demands for all VLs in L_i. If all such demands are met, then resources are allocated and consequently s_i is successfully remapped. This process is repeated till all the VMs in V are assigned resources. Next, we discuss the time complexity of *VMatch*.

D. Time Complexity Analysis

The preference lists of the players, i.e., VMs and servers are sorted lists. Although we consider an incomplete preference

list for VMs, the number of entries in the lists are close to $|N^H|$. Therefore the overall time complexity to build the preference list P_V of all VMs is $O(l * |N^H| log |N^H|)$. Similarly the preference list P_{N^H} corresponding to the hosts can be constructed in $O(|N^H| l \ log \ l)$ time in its worst case. The overall time complexity of the matching algorithm given the preference profile of players in its worst case results to be $O(l * |N^H|)$.

V. Performance Evaluation

We have performed simulation using CloudSim[3] simulator with certain modifications. The environmental setup and analysis of the simulation results are discussed elaborately in this section.

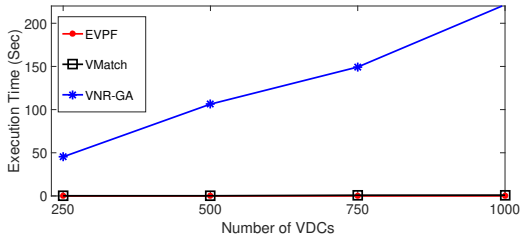

Fig. 7. Execution Time Vs. Number of VDCs.

A. Environmental Setup

For the purpose of simulation, we have considered 4 Geo-distributed DCs connected via an interconnection network as depicted in Figure 2. Each DC follows a 3-tier fat-tree topology [1] generated using BRITE[4] topology generator. The Top of the Rack (ToR) links (connecting the servers and switches) have a capacity of 10Gbps, switch to switch links (both aggregate and core) and interconnecting DC links are 40Gbps links. The computational capacity of servers expressed in terms of CRBs is generated using a uniform distribution in the range of U[512, 1024] (where U[m, n] signifies uniform distribution between m and n). The cost (in dollars) of using a CRB for a server is a variable and is expressed as a uniform process between U[1, 6]. However, cost (in dollar) of using a unit bandwidth resource is fixed to 1. The arrival of the VDC requests follow a Poisson Process. The number of VMs in a VDC requests are generated following a uniform process in the range of U[2, 10]. The CRB demands of each VM corresponding to a VDC request is generated in the range of U[2, 6]. The VLs connecting the VMs are generated randomly with each pair of VMs having 0.5 probability of being connected. The bandwidth demands of such VLs are generated in the range of U[1, 40] Mbps. The rate of VDCs requesting bandwidth upgrade follows a uniform process in the range of U[25%, 75%]. Note that bandwidth upgrade is for VLs of VDCs that have already been allocated resources. For VDCs experiencing excess demand for bandwidth, number of VLs requesting bandwidth upgrade are also distributed uniformly in the range of U[1, $|L^V|$]. Further, the amount of bandwidth upgrade requested for each VL also real numbers in the range of U[10%, 100%]. Further to assign importance to criteria in AHP, we use a linear judgment scale as per Table I.

[3]http://www.cloudbus.org/cloudsim
[4]https://www.cs.bu.edu/brite/

B. The Baseline Algorithms

To analyse the performance of VMatch, we compare its performance with two different baselines inspired by Dab *et al.* [5] (referred to as VNR-GA) and Sun *et al.* [10] (referred to as EVPF). VNR-GA based on genetic-algorithm aims to minimize number of migrations and remapping cost while maximizing SPs revenue for remapping SCs of multiple VDCs. On the other hand, EVPF is focused on reducing the embedding cost and energy consumption of VDCs requesting additional resources. The simulation parameters of VNR-GA are set in accordance with [5].

C. Simulation Results

Figure 7 reports the total execution time to build a remapping plan for migrating 250-1000 VDCs at an equal interval of 250 per observation. It can be observed from the figure that *VMatch* and EVPF are able to construct a solution in reasonably quick time in comparison to VNR-GA. This is attributed to the fact that VNR-GA is based on genetic meta-heuristic that iterates over multiple runs over a large sized population adding to the overall execution time. Figure 8 shows the remapping cost for different number of VDC requests. It can be observed from the figure that *VMatch* is able to reduce the remapping cost in comparison to both baselines. On an average *VMatch* is able to reduce the remapping cost by 19 % across baselines. This is because cost is considered as a evaluation parameter while ranking the alternatives (Refer to Equation (9)). Further, it is inferred from Figure 9 that the overall revenue (total cost) earned by SP is at par in all approaches. Hence, we can safely conclude that *VMatch* is able to reduce the remapping cost without compromising much with the total cost. Figures 10 and 11 illustrates the average host and bandwidth utilization of substrate network for provisioning multiple VDCs. Note that the utilization of resources takes into consideration the entire VDC request, i.e., it includes the remapped as well as already mapped components. It can be observed from Figure 10 that on an average *VMatch* is able to improve the host utilization by 21% across baselines. This is attributed to the fact that *VMatch* takes into consideration the individual host usage during relocation (Refer to Equation (7)). However, VNR-GA and EVPF are primarily focused at reducing the remapping costs, hence, the utilization of hosts is compromised. Further, it can also be inferred from Figure 11 that although *VMatch* improves the host utilization it does not degrade the average bandwidth utilization in comparison with the baselines.

VI. Conclusion

In this paper, we have proposed a model called *VMatch* that is aimed at building efficient remapping plan to deal

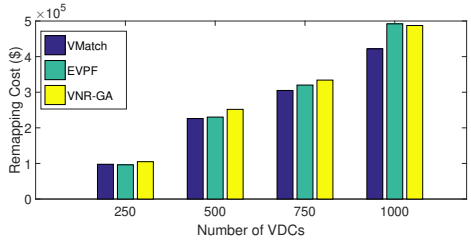

Fig. 8. Remapping Cost ($) Vs. Number of VDCs.

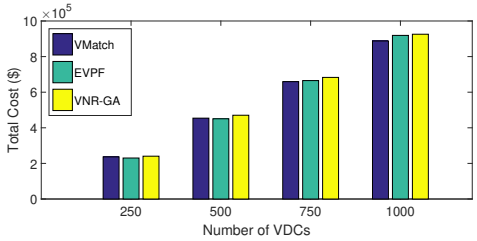

Fig. 9. Total Cost ($) Vs. Number of VDCs.

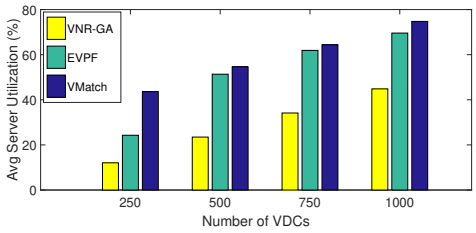

Fig. 10. Server Utilization (%) Vs. Number of VDCs.

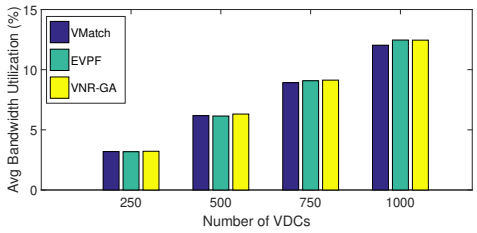

Fig. 11. Bandwidth Utilization (%) Vs. Number of VDCs.

with VLs facing bandwidth expansion. Since, relocating VLs often involves reconfiguration of both VMs and VLs, *VMatch* addresses the problem by relocating SCs. *VMatch* primarily aims to balance the priorities of different stakeholders, i.e., end-users and SPs. From the perspective of end-users *VMatch* aims to reduce the remapping cost. From a SPs standpoint, *VMatch* aims to maximize the aggregate utilization of hosts without negatively impacting the revenue. A matching theory based framework is proposed to take into consideration the contrasting agendas of stakeholders. The overall problem is formulated as a one-to-one matching game and it solved using a modified gale-shapley algorithm. Simulation results show an 19% reduction in the remapping cost and 21% improvement in the average utilization of substrate hosts across baselines.

REFERENCES

[1] G. Sun, D. Liao, D. Zhao, Z. Xu, and H. Yu, "Live migration for multiple correlated virtual machines in cloud-based data centers," *IEEE Transactions on Services Computing*, vol. 11, no. 2, pp. 279–291, 2018.

[2] N. M. M. K. Chowdhury, M. R. Rahman, and R. Boutaba, "Virtual network embedding with coordinated node and link mapping," in *IEEE INFOCOM 2009*, 2009, pp. 783–791.

[3] M. G. Rabbani, R. P. Esteves, M. Podlesny, G. Simon, L. Z. Granville, and R. Boutaba, "On tackling virtual data center embedding problem," in *2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013)*, 2013, pp. 177–184.

[4] G. Sun, H. Yu, V. Anand, and L. Li, "A cost efficient framework and algorithm for embedding dynamic virtual network requests," *Future Generation Computer Systems*, vol. 29, no. 5, pp. 1265 – 1277, 2013.

[5] B. Dab, I. Fajjari, N. Aitsaadi, and G. Pujolle, "Vnr-ga: Elastic virtual network reconfiguration algorithm based on genetic metaheuristic," in *2013 IEEE Global Communications Conference (GLOBECOM)*, 2013, pp. 2300–2306.

[6] A. E. Roth, "Deferred acceptance algorithms: History, theory, practice, and open questions," *international Journal of game Theory*, vol. 36, no. 3-4, pp. 537–569, 2008.

[7] A. Satpathy, M. N. Sahoo, L. Chottray, B. Majhi, A. Mishra, and S. Bakshi, "Vrmap: A cost and time aware remapping of virtual data centres over a geo-distributed infrastructure," in *2020 International Conference on COMmunication Systems NETworkS (COMSNETS)*, 2020, pp. 427–434.

[8] H. Xu and B. Li, "Anchor: A versatile and efficient framework for resource management in the cloud," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 6, pp. 1066–1076, 2012.

[9] A. Abouaomar, A. Kobbane, and S. Cherkaoui, "Matching-game for user-fog assignment," in *2018 IEEE Global Communications Conference (GLOBECOM)*, 2018, pp. 1–6.

[10] G. Sun, D. Liao, S. Bu, H. Yu, Z. Sun, and V. Chang, "The efficient framework and algorithm for provisioning evolving vdc in federated data centers," *Future Generation Computer Systems*, vol. 73, pp. 79 – 89, 2017.

[11] D. G. Andersen, "Theoretical approaches to node assignment," 2002.

[12] K. Metwally, A. Jarray, and A. Karmouch, "A distributed auction-based framework for scalable iaas provisioning in geo-data centers," *IEEE Transactions on Cloud Computing*, pp. 1–1, 2018.

[13] A. Amokrane, M. F. Zhani, R. Langar, R. Boutaba, and G. Pujolle, "Greenhead: Virtual data center embedding across distributed infrastructures," *IEEE Transactions on Cloud Computing*, vol. 1, no. 1, pp. 36–49, 2013.

[14] Q. Zhang, M. F. Zhani, M. Jabri, and R. Boutaba, "Venice: Reliable virtual data center embedding in clouds," in *IEEE INFOCOM 2014 - IEEE Conference on Computer Communications*, 2014, pp. 289–297.

[15] T. Wood, P. Shenoy, A. Venkataramani, and M. Yousif, "Black-box and gray-box strategies for virtual machine migration," in *Proceedings of the 4th USENIX Conference on Networked Systems Design and Implementation*, ser. NSDI'07. USA: USENIX Association, 2007, p. 17.

[16] B. Gu, Z. Zhou, S. Mumtaz, V. Frascolla, and A. Kashif Bashir, "Context-aware task offloading for multi-access edge computing: Matching with externalities," in *2018 IEEE Global Communications Conference (GLOBECOM)*, 2018, pp. 1–6.

[17] S. F. Abedin, M. G. R. Alam, S. M. A. Kazmi, N. H. Tran, D. Niyato, and C. S. Hong, "Resource allocation for ultra-reliable and enhanced mobile broadband iot applications in fog network," *IEEE Transactions on Communications*, vol. 67, no. 1, pp. 489–502, 2019.

[18] E. Triantaphyllou, "Multi-criteria decision making methods," in *Multi-criteria decision making methods: A comparative study*. Springer, 2000, pp. 5–21.