Egalitarian Stable Matching for VM Migration in Cloud Computing

Hong Xu, Baochun Li
Department of Electrical and Computer Engineering
University of Toronto
{*henryxu, bli*}@*eecg.toronto.edu*

***Abstract*—Virtual machine migration represents a new challenge to design efficient and practical migration algorithms that work well with hundreds or even thousands of VMs and servers. In our previous work, we advocated the use of a general stable matching framework from economics to solve networking problems, and showed that it offers unique advantages compared to utility-based optimization, using a simple procedure of deferred acceptance. In this paper, we apply this framework to the VM migration problem, and propose an *egalitarian* approach that finds a stable matching fair to both VMs and servers, instead of favoring either side as a result of the deferred acceptance procedure. Such an egalitarian stable matching that minimizes the total rank sum of the outcome is shown to be a sensible notion of fairness under mild conditions, and through simulations is demonstrated to offer superior results.**

I. Introduction

Cloud computing is touted over recent years to provide ubiquitous IT services, ranging from search and online social networks to high performance computing and infrastructure outsourcing, in a cost-efficient and flexible way. These cloud services are usually packaged and run in the form of virtual machines (VM) in the server farm, so that the statistical multiplexing effect improves the resource utilization of the infrastructure.

Live migration of virtual machines (VMs), the process of dynamically transferring a virtual machine across different physical servers at runtime, has attracted significant attention in both industry and academia. It represents a new opportunity to enable agile and dynamic resource management in modern data centers [1], which is barely touched in existing studies. This is especially important since data center networks are fraught with scalability and efficiency issues, which have already become subjects of concerns among practitioners and researchers [2], [3].

The problem is more complicated than it seems if we consider the practical aspects of data center networks. To a large extent, current data center networks follow a three-tier architecture put forth by [4] as shown in Fig. 1. It is thus the interest of both VMs and servers to migrate VMs to topologically nearby servers, because doing so minimizes the imposed traffic footprint to the data center network and the application downtime to the VMs.

Conflicts of interest abound, however: servers differ in their traffic loads, affecting the performance of migration. VMs also differ in the volume of disk images to be migrated, incurring different amounts of transmission overhead to the network — a critical factor that servers need to consider.

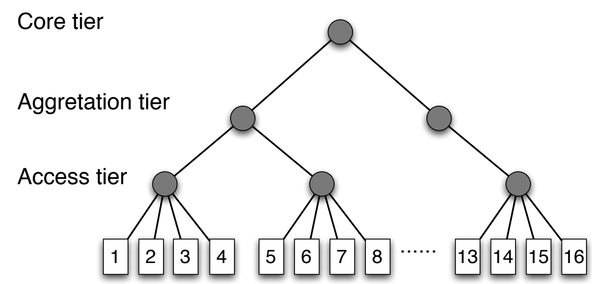

Fig. 1. A three-tier datacenter network architecture.

One may be tempted to resort to optimization to tackle this problem. However, optimization dictates an arbitrary way of resolving the conflicts of interest between different stakeholders in order to achieve a global notion of performance optimality. It ignores the individual rationality of participants of the market, which is critical since the cloud provider and consumers are autonomously and selfishly seeking to maximize their own benefits. Moreover, a concrete utility function is often needed for optimization, which may not be possible to define since different stakeholders have different considerations that they deem fit.

In our previous work [5], we advocated the use of *stable matching* as a general framework to tackle networking problems, where *preferences* are used to model each agent's interest, and *stability* serves as the solution concept instead of optimality. An ISP peering case study was considered in [5]. In this paper we apply the same framework to VM migration problem. Each agent, be it a VM or a server, ranks the agents on the other side of the market based on some criteria, which is its preference list. An algorithm then computes a matching that cannot be improved by any pair of VM and server, i.e. it is stable.

The use of the stable matching framework marks a substantial shift from utility-based optimization or game-theoretic solution methods. The merit of the framework, as we have shown in [5], is its overall practicality. The generic preference abstraction embraces many heterogeneous and complex considerations that network operators and cloud applications may have. The classical deferred acceptance algorithm can be applied in a centralized manner with little complexity. The performance

978-1-4244-9920-5/11/$26.00 ©2011 IEEE

of stable matching is competitive to that of an optimization approach, despite its use of ordinal information only.

The problem of the deferred acceptance algorithm is that it can only produce two extreme outcomes, one that is VM-optimal and one server-optimal [5]. This is known as *polarization* of stable matchings [6]. In many cases, the network operator looks for a "fair" stable matching that does not favor either side as the operating point of the system, with which the VMs' migration performance and the data center network's traffic footprint are better balanced. It is therefore important to answer such practical needs with efficient implementations.

In this paper, we apply the *egalitarian* stable matching concept to solve this issue [7]. The intuition of egalitarian stable matching is simple: it tries to find the matching that minimizes the total rank sum of the outcome among all stable matchings [8], [9]. We will first show that the total rank sum, i.e. the total sum of the ranks of the VMs and servers in their matched servers' and VMs' preferences, respectively, can be unambiguously used to compare stable matchings under mild conditions. We then apply a polynomial-time algorithm developed in [7] to find such egalitarian stable matching. We also conduct simulations to demonstrate the effectiveness and practicality of our approach.

II. THE STABLE MATCHING FRAMEWORK

A. Background

We start by introducing the basic theory of stable matching in the one-to-one marriage model as necessary background for this paper. In this model, there are two disjoint sets of agents $\mathcal{M} = \{m_1, m_2, \ldots, m_n\}$ and $\mathcal{W} = \{w_1, w_2, \ldots, w_p\}$, i.e. men and women. Each agent has a complete and transitive preference over individuals on the other side, and the possibility of being unmatched [10]. Preferences are rank order lists of the form $p_{m_1} = w_4, w_2, \ldots, \emptyset$, meaning that man m_1 favors w_4 the most as its partner, w_2 the next and so on, until at some point he prefers to be unmatched (i.e. matched to the void set). We use $>_i$ to denote the ordering relationship of i. If i prefers to remain unmatched than being matched to agent j, i.e. $\emptyset >_i j$, then j is said to be *unacceptable* to i, and preferences can be represented just by the list of acceptable partners. Preferences are *strict* if each agent is not indifferent between any two acceptable partners.

Definition 1: An outcome of the market is a *matching* $\mu : \mathcal{M} \times \mathcal{W} \to \mathcal{M} \times \mathcal{W}$ such that $w = \mu(m)$ if and only if $\mu(w) = m$, and $\mu(m) \in \mathcal{W} \cup \emptyset$, $\mu(w) \in \mathcal{M} \cup \emptyset$, $\forall m, w$.

This implies that the outcome matches agents on one side to those on the other side, or to the empty set. Agents' preferences over outcomes are determined solely by their preferences for their own partners in the matching.

It is clear that we need further criteria to distill a "good" set of matchings from all the possible outcomes. One natural criterion is that a *blocking set* defined as follows should not occur:

Definition 2: A matching μ is *blocked* by a pair of agents (m, w) if they each prefer each other to the partner they receive at μ. That is, $w >_m \mu(m)$ and $m >_w \mu(w)$. Such a pair is called a *blocking set* in general.
If there is a blocking set in the matching, the agents in the set have an incentive to break up and form a new marriage. Therefore such an "unstable" matching is not desirable.

Definition 3: A matching μ is *stable* if and only if it is individual rational, and is not blocked by any pair of agents.

In the marriage model, a stable matching is efficient, and the set of stable matchings equals the *core* of the game whose rules are that agents from opposite sides of the market can match if and only if they both agree [11].

The following important theorem establishes the existence of stable matching:

Theorem 1: A stable matching exists for every marriage market.

This can be readily proved by the classic *deferred acceptance algorithm*, or the *Gale-Shapley algorithm* proposed in [12] (with men proposing). In the first round, each men proposes to his first choice if he has any acceptable ones. Each woman rejects any unacceptable proposals and, if more than one acceptable proposals are received, holds the most preferred and rejects all others. In each round that follows, any man rejected at the previous round makes a new proposal to his most preferred acceptable partner who has not yet rejected him, or makes no proposal if no acceptable choices remain. Each woman holds her most preferred offer *up to this round*, and rejects all the rest. When no further proposals are made, the algorithm stops and matches each woman to the man (if any) whose proposal she is holding. The women-proposing version works in the same way by swapping the roles of men and women.

The seminal paper [12] has thus spurred the research of stable matching in both economics and computer science. Many models have been developed that consider other variants of different markets. More importantly, the theory of stable matching has also been extensively tested in real world. Many labor markets have adopted and extended the deferred acceptance procedure to match employers with employees. Prominent examples include the National Residency Program in the U.S. and many medical labor markets in Britain and Canada [10].

B. VM Migration as A College Admissions Problem

First let us state our assumptions. We focus on a server maintenance scenario where VM migration is triggered mainly by periodic upgrades and maintenances, as well as by failures of hardware components [13]. We assume that each VM enjoys a uniform provision of resources that it can possibly use in terms of CPU, memory and disk, though their actual resource usage may differ widely. This is usually the case in practice. In addition, we assume techniques for monitoring the traffic load on servers are available, which is widely provided by vendors [14], [15].

We study VM migration as a college admissions problem [12], a variant of the stable matching problems. The "market" consists of a set of VMs $\mathcal{V}$ with cardinality V on one side, and a set of servers $\mathcal{S}$ with cardinality S on the other side. Each VM ("student") seeks to be migrated to one server ("college"), while each server may have vacant capacity to hold multiple VMs. The capacity of a server is the maximum number of instances q_s it can hold, up to the resource provisioning limit per VM. We assume $\sum_v q_s \geq V$, so that every VM will have one match.

To model the common and conflicting interest, the concept of preferences is used in the stable matching literature. In our VM migration problem, the derivation of preferences can be based on a wide spectrum of practical considerations, possibly including the hop distance, storage image size of VMs, and traffic load of servers. We are not specifically concerned with the construction of preferences here in this paper.

Given $P_V = (p_{v_1}, \ldots)$ and $P_S = (p_{s_1}, \ldots)$, the vectors of preferences from both parties, we can define stability of matchings in our college admissions model:

Definition 4: A matching μ is *stable* if there is no incentive for any pair (v, s) to deviate from μ. That is, there is no pair $(v, s) \in \mathcal{V} \times \mathcal{S}$ such that (i) VM v prefers server s to its matched one $u(v)$; and (ii) s prefers to add v to its set, possibly at the expense of another less-preferred VM according to p_s.

The existence of stable outcomes is then immediate from [6], which may be proven by a simple extension of the centralized deferred acceptance algorithm [6]. Suppose we let VMs be the proposing side as in [5]. First, VMs propose to their first choices. A server with quota q_s then places on its waiting list q_s VMs who rank highest, or all VMs if there are fewer than q_s proposals, and rejects the rest. Rejected VMs then apply to their second choices, and servers again accept q_s highest ranked VMs from all the proposals it has received up to this round. The algorithm continues until every VM is placed on one waiting list. Each server then admits every VM on its waiting list, and a stable matching has been produced. This centralized algorithm can be implemented as a service module in the control plane.

It can be readily proven that in the VM migration problem, the set of stable matchings has a specific lattice structure as in the classical one-to-one marriage problem [5], [16]. This structure implies that there is a best stable matching for one side of the market, which is at the same time the worst for the other side, i.e. the *polarization* of stable matchings. Specifically,

Theorem 2: The matching produced by the VM-proposing algorithm is the VM-optimal stable matching, while the one produced by the server-proposing algorithm is server-optimal.

Therefore, which side proposes has a direct impact on the outcomes. In our context, this implies that the VM-proposing algorithm offers best performance for VMs in terms of minimizing their individual downtime, while having the worst performance for servers in that the overall overhead of transmission is not optimized. In this work, our objective is to find a plausible fairness criterion that strikes a balance between the benefits of VMs and servers.

Let us take a look at an example to understand the polarization problem in context. Table. I and II show the preference lists of 5 VMs and 4 servers, together with servers' quotas. We can readily obtain the VM-optimal stable matching:

$$\text{VM-optimal: } (s_1, v_4), (s_2, v_2), (s_3, v_1, v_3), (s_4, v_5), \quad (1)$$

TABLE I
PREFERENCE LISTS FOR VMS.

v_1	s_3, s_2, s_1, s_4
v_2	s_2, s_1, s_4, s_3
v_3	s_3, s_1, s_4, s_2
v_4	s_1, s_4, s_2, s_3
v_5	s_4, s_2, s_3, s_1

TABLE II
PREFERENCE LISTS FOR SERVERS.

quota		
1	s_1	v_5, v_2, v_3, v_1, v_4
1	s_2	v_3, v_5, v_4, v_1, v_2
2	s_3	v_2, v_4, v_5, v_3, v_1
1	s_4	v_1, v_4, v_2, v_3, v_5

by running the VM-proposing deferred acceptance algorithm for one round, since all VMs are accepted to their first choices. This outcome, however, is the worst for servers since all servers are assigned their worst choices of VMs.

The server-optimal stable matching can be similarly obtained

$$\text{Server-optimal: } (s_1, v_5), (s_2, v_3), (s_3, v_2, v_4), (s_4, v_1), \quad (2)$$

where servers are all matched to their first choices. It is clearly distinct from the VM-optimal stable matching, and one can readily verify that it is the worst for VMs.

III. EGALITARIAN STABLE MATCHING

We now present our egalitarian stable matching framework in this section.

A. Cases of Complications

There are many possible fairness criteria to distill a good stable matching. One natural choice is to minimize the total rank sum of partners of all agents in the matching, as first stipulated in [17] for one-to-one matching problems. This means that the average "happiness" of the agents involved are maximized. However, since in our problem a server can admit multiple VMs, a server's ranking over individual VMs alone may not be sufficient to determine its preferences over combinations of them.

Specifically, there are several cases one needs to take care of. First, a server with quota 2 may not be indifferent between the combinations of VMs (1, 4) and (2, 3), where the numbers indicate the ranks of VMs in the server's preference. Second, it may prefer the combination of (1, 5) over (2, 3) due to the significance of the first VM in its list, though the rank sum of the former is greater (worse) than the latter. Third, it may also prefer the combination of (2, 4) over (2, 3), if the second and fourth VMs are complementary to each other and they have to be migrated to the exact same machine.

The first two complications can be readily resolved by using weighted preferences, which provide not only ordinal but also quantitative information. However, this might not be possible when policy or other subjective configurations are involved, which is quite common in production data centers. Luckily, we note that some properties of the general many-to-many stable matchings due to [18] also hold for our many-to-one matching problem, and can help us eliminate the first complication. Further, a mild assumption on the structure of preferences eliminates the remaining complications and establishes the total rank sum as an unambiguous notion of fairness we can use.

B. Analysis

We first introduce some notations. Given a set of VMs $\mu(s)$ paired with server s in a stable matching μ, we define the *dissatisfaction score* $DS(\mu(s))$ of s to be the sum of ranks over $\mu(s)$ in its preference p_s, as in [18].

$$DS(\mu(s)) = \sum_{v\in\mu(s)} R_s(v), \tag{3}$$

where $R_s(v)$ denotes the rank given by s to v. The dissatisfaction score of VM v is simply $DS(\mu(v)) = R_v(\mu(v))$. The dissatisfaction score of the stable matching μ is then the sum of the scores of all agents involved.

$$DS(\mu) = \sum_{v\in\mathcal{V}} DS(\mu(v)) + \sum_{s\in\mathcal{S}} DS(\mu(s)). \tag{4}$$

The following results from [18] are stated for servers. They are also true for VMs by symmetry.

Proposition 1: A server $s \in \mathcal{S}$ is assigned the same number of VMs, n_s, in all stable matchings. Further, if $n_s < q_s$, then s has the same set of VMs in all stable matchings [18].

Proposition 2: Suppose μ and μ^* are different stable matchings that assign different sets of VMs to a server s. Then there is one matching (say μ) such that if $(v, s) \in \mu$ and $(v^*, s) \in \mu^*\backslash\mu$, $R_s(v) < R_s(v^*)$ [18].

A useful corollary of Proposition 2 is that if a server is assigned different sets of VMs in different stable matchings, then its least preferred VM in each of them must be different.

Corollary 1: Suppose μ and μ^* are stable matchings of the same problem instance $(\mathcal{V}\times\mathcal{S}, P_V, P_S)$. Then for every server $s \in \mathcal{S}$, either $\mu(s) = \mu^*(s)$, or $\min(\mu(s)) \neq \min(\mu^*(s))$.
Here $\min(\mu(s))$ denotes the least preferred VM among those matched to s in μ. Note that this holds trivially for VMs, since in different matchings a VM is matched to either the same or different servers.

With the two propositions and Corollary 1, we can prove the following theorem that essentially eliminates the first case of complication.

Theorem 3: Suppose μ and μ^* are different stable matchings that assign distinct sets of VMs to a server $s \in \mathcal{S}$. Then $DS(\mu(s)) \neq DS(\mu^*(s))$.

Proof: By Proposition 1, s must be matched to the same number of VMs in μ and μ^*. By Proposition 2, we can assume that $R_s(\min(\mu(s))) < R_s(\min(\mu^*(s)))$ without loss of generality. Each VM in the preference list p_s that ranks below $R_s(\min(\mu(s)))$ corresponds to at most one set of VMs for s (amongst which it is the least preferred VM).

Consider the first VM that ranks below $R_s(\min(\mu(s)))$ in p_s, which is the least preferred VM of s in some stable matching, say μ^{**}. By Proposition 2, any VM $v \in \mu^{**}\backslash\mu$ must rank below any VM $v' \in \mu$ in p_s. Moreover, since $|\mu^{**}(s)| = |\mu(s)|$, each such v is a replacement of some other $v' \in \mu$ that ranks above v in p_s. Each replacement of v' with v leads to an increase of the dissatisfaction score of s so that $DS(\mu(s)) < DS(\mu^{**}(s))$. ∎

Thus, each server has distinct total rank sum of its matched VMs in different stable matchings. This theorem implies that the first case of complication with equal total rank sum for distinct stable matchings will not happen. However, this alone cannot resolve the other two cases of complications that largely concern the relationship between VMs. It can be readily seen that we need further assumptions on the complementariness among VMs in order to do so.

Thus, we impose an additional *no-complementarities* assumption on the preference orderings of servers over combinations of VMs in the outcome as in [7]:

Definition 5: Given two sets of VMs A_1 and A_2, if a server s prefers A_1 at least as much as A_2, and $v_1 >_s v_2$, then s strictly prefers $A_1 \cup v_1$ to $A_2 \cup v_2$.

No-complementarities is a special case of preferences in which VMs are substitutes rather than complements to servers [10]. This essentially means that a server always prefers adding an acceptable VM before reaching the quota and it always prefers replacing a VM with a better one when the quota is met. Clearly this is reasonable to assume for servers.

Finally, the following theorem proved in [7] essentially asserts that with the no-complementarities assumption, we can obtain a strict preference ordering over all possible stable matchings for any server by specifying only the preferences on individual VMs.

Theorem 4: Suppose $\mu(s)$ and $\mu^*(s)$ are two distinct sets of VMs of server s under stable matchings μ and μ^*, respectively. Then, (i) $DS(\mu(s)) \neq DS(\mu^*(s))$, and (ii) if $DS(\mu(s)) < DS(\mu^*(s))$, then s prefers μ over μ^* and vice versa [7].

This obviates the need for preferences that specify the orderings over all possible combinations of VMs *a priori* which would be of exponential size. We can now unambiguously compare any two sets of stable outcomes by comparing the dissatisfaction scores. An egalitarian measure of fairness that minimizes the total rank sum across all agents thus makes sense.

As a simple illustration, let us reuse the example as shown in Table. I, II. The total rank sums of the VM-optimal and server-optimal stable matchings as shown in (1) and (2) are 29 and 26 respectively. Now consider the following egalitarian stable matching for the same problem instance:

$$\text{egalitarian: } (s_1, v_2), (s_2, v_5), (s_3, v_1, v_3), (s_4, v_4), \tag{5}$$

We can see that the total rank sum is 23, which is smaller than that of both VM-optimal and server-optimal matching. Most of the servers (except s_3) are matched to their second choices, and most of the VMs are also assigned their second choices (except v_1, v_3). This matching clearly represents a fair balance between the two parties.

A centralized polynomial time algorithm to find such egalitarian stable matching is developed in [7], and can be readily applied to our problem here. The complexity is $O(n^6)$, where $n = \min\{V, S\}$. Note that it is worse than the simple deferred acceptance algorithm which is only $O(n^2)$.

C. Discussions

It should be noted that our framework is generally applicable to any sensible preference derivation that VMs and servers have.

Readers may be interested in how preferences can be defined in different scenarios, and how the different definitions affect the performance of the migration algorithm. Indeed we are exploring this direction as one of our future work. However, this paper is mainly positioned at provoking the use of stable matching theory as a new theoretical tool, and thus is concerned with the theoretical development as the first step. We use a simple preference derivation in our evaluation in this paper, which is shared among agents on the same side of the market. However, it should be noted that stable matching allows the preference derivation to be heterogeneous across agents. That is, one VM may have an entirely different preference definition than another. This is also one of the key merits of the framework compared with optimization.

IV. Evaluation

We are now ready to resort to simulations to study the performance of egalitarian stable matchings. As no previous work has been done using the theory of stable matching, we rely on the VM-proposing deferred acceptance algorithm in our previous work [5] as the performance benchmark, which produces the VM-optimal stable matching.

As discussed above we adopt a simple preference derivation. We assume that VMs rank servers in an ascending order of the *transmission cost* $\Phi_{v,s}$ defined as follows:

$$\Phi_{v,s} = d_{s(v),s}/\theta_{s(v),s}, \forall v \in \mathcal{V}, s \in \mathcal{S}, \tag{6}$$

where $s(v)$ denotes v's residing server, $d_{s(v),s}$ denotes its hop distance to server s, and $\theta_{s(v),s}$ denotes the available end-to-end bandwidth between $s(v)$ and s. This derivation takes a joint consideration of hop distance and server traffic load. Servers, on the other hand, rank VMs in an ascending order of the *migration overhead* defined as

$$\Omega_{s,v} = d_{s(v),s} \cdot \zeta_v, \forall v \in \mathcal{V}, s \in \mathcal{S}, \tag{7}$$

where ζ_v is the size of VM disk image in bytes. It considers both the transmission distance and volume, and captures the total amount of bytes processed by routers in the network for migrating v to s. Both servers' and VMs' preferences are assumed to be complete and strict.

In the simulations, we set the maximum capacity of each server to be 4 VMs. The quota of each server is uniformly distributed in $[1, 4]$. The hop distance between servers is generated by assuming the fan-out of access routers to be 4, and the fan-out of aggregation routers to be 8. The initial placement of VMs on servers is random. The size of VM's disk image ζ_v is uniformly distributed in $[1, 100]$ Gb, and the available bandwidth between servers $\theta_{s(v),s}$ is uniformly distributed in $[0.5, 1.5]$ Gbps.

A. Overall Performance

Fig. 2, 3 show the overall performance averaged over 100 runs with 200 servers and increasing number of VMs. We can clearly see that egalitarian stable matching achieves a different tradeoff point between the benefits of two parties. In terms of average transmission cost, the VM-optimal stable matching

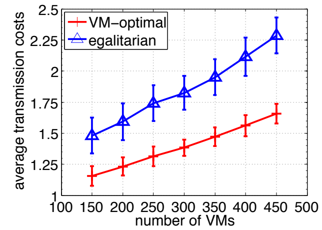

Fig. 2. Transmission costs comparison under general conditions ($S = 200$).

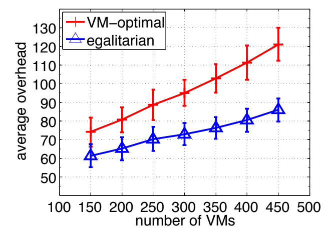

Fig. 3. Migration overhead comparison under general condition ($S = 200$).

produced by deferred acceptance algorithm outperforms the egalitarian stable matching by over $10 - 15\%$. This is because the VM-optimal stable matching provides the best performance for VMs among all stable matchings of the problem instance, i.e. the least transmission cost. The egalitarian approach tries to strike a balance between these conflicting interest while ensuring that the matching is still stable. Thus the performance of VMs is inevitably scarified.

On the other hand, the performance of servers is improved in the egalitarian stable matching. As seen in Fig. 3, the average migration overhead is around 15% better than that of the VM-optimal stable matching. Thus, the data center network is better served with less migration overhead, at the cost of VM migration performance. Egalitarian stable matching offers an alternative with fair tradeoff between the benefits of the two parties for the cloud operator.

We also compare the total rank sums of the two different stable matchings for the same problem instance. We observe through a set of simulation that, indeed, egalitarian stable matching minimizes the rank sum of all matched pair of VMs and servers. We omit the results due to space limit.

B. Performance under Extreme Conditions

The story is different when we evaluate the performance of egalitarian stable matching under extreme conditions, where the available total quotas of servers are barely enough to accommodate migrating VMs. We generate these problem instances by reducing the average quota of each server from 2 to 1 (it is ensured that the total quota is larger than the number of VMs).

Fig. 4, 5 show the results with increasing number of VMs averaged over 100 runs. We can see that egalitarian stable matching and VM-optimal stable matching stay close to each other in both performance metrics. We also observe that the difference in total rank sum between the two stable matchings is smaller than that under general conditions.

The reason for this indifference is that, when quota is barely enough, the total number of stable matchings for a problem instance is in general much less. As a result, the performance of egalitarian stable matching compared to the VM-optimal stable matching is not as much different as it is under general conditions. We also find that reducing the variance of server traffic load ζ_s has the same effect. The reason is similar: when ζ_s of different servers are largely clustered in a small range,

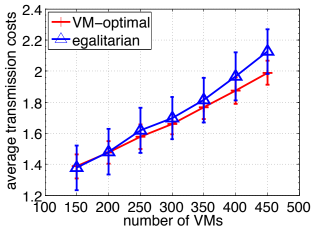

Fig. 4. Transmission costs comparison under extreme conditions ($S = 200$).

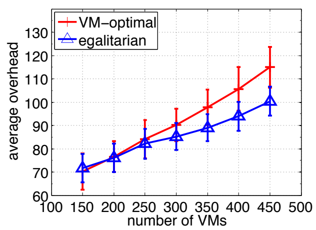

Fig. 5. Migration overhead comparison under extreme conditions ($S = 200$).

TABLE III
RUNNING TIME OF THE ALGORITHMS.

Algorithm	Average running time (s)
deferred acceptance	0.0145
egalitarian stable matching	26.69

migration overhead $\theta_{s,v}$ for different VMs is insignificant, and the final performance difference between distinct stable matchings is thus insignificant. Due to space limit we omit the figure here.

C. Discussion

Finally, in trying to be impartial, we evaluate the running time of the egalitarian stable matching algorithm as in [8], which as we discussed is worse than the simple deferred acceptance procedure. More intuitively, Table. III shows a running time comparison averaged over 100 runs on a Dual-Core Intel Xeon 3.0 Ghz machine, with a problem size of 200 servers and 350 VMs. Egalitarian stable matching has a disadvantage of being three orders of magnitude slower than deferred acceptance.

Together with the simulation results, this tells us that the egalitarian approach is applicable and effective when the problem has a moderate to large number of stable matchings. We note that this does not hurt its practicality since in general, production data center networks operate under mild workload where there are more than enough servers to host all the migrating VMs. The running time is also acceptable for practical use. Under extreme conditions the use of egalitarian stable matching is not justified due to its computational burden and little performance improvement.

V. Related Work

Live migration of virtual machines, the process of transitioning a VM across physical servers, emerges and has attracted significant attention in both industry and academia [14], [15], [19], with products such as VMotion already being shipped. Most of the existing work, however, focus on implementation issues and optimization techniques of migration. The question of which servers to migrate the VMs to is largely left untouched.

There are also some studies in networking that apply stable matching, though this line of related work is very limited. In [20], the endogenous formation of source-relay pairs between selfish nodes in cooperative wireless networks is modeled and solved as a stable roommate problem. A further variant of the stable roommate problem called stable exchange problem is proposed in [21] to model the peer selection process of peer-to-peer storage systems. Our recent work [5], to our knowledge, represents the first attempt in applying stable matching theory as the general framework for solving networking problem.

VI. Concluding Remarks

In this paper, we advocated an egalitarian stable matching framework, which introduces a novel perspective on solving the VM migration problem. Egalitarian stable matching aims to address the polarization issue of the deferred acceptance algorithm we used in [5], and achieve a fair balance between the benefits of VMs and servers. We presented necessary theoretical foundation for the applicability of the egalitarian approach, and through simulations demonstrated its effectiveness and practicality.

References

[1] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, "Xen and the Art of Virtualization," in *Proc. SOSP*, 2003.
[2] A. Greenberg, J. R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, P. Patel, and S. Sengupta, "VL2: A Scalable and Flexible Data Center Network," in *Proc. ACM SIGCOMM*, 2009.
[3] C. Guo, H. Wu, K. Tan, L. Shi, Y. Zhang, and S. Lu, "BCube: A High Performance, Server-Centric Network Architecture for Modular Data Centers," in *Proc. ACM SIGCOMM*, 2009.
[4] "Cisco Data Center Infrastructure 2.5 Design Guide," 2007.
[5] H. Xu and B. Li, "Seen As Stable Marriages," in *Proc. INFOCOM*, 2011.
[6] A. E. Roth, "The College Admissions Problem Is Not Equivalent to the Marriage Problem," *J. Econ. Theory*, vol. 36, pp. 277–288, 1985.
[7] V. Bansal, A. Agrawal, and V. S. Malhotra, "Polynomial Time Algorithm for An Optimal Stable Assignment with Multiple Partners," *Theoretical Computer Science*, vol. 379, pp. 317–328, 2007.
[8] R. Irving, P. Leather, and D. Gusfield, "An Efficient Algorithm for the 'Optimal' Stable Marriage," *J. ACM*, vol. 34, no. 3, pp. 532–543, 1987.
[9] D. F. Manlove, R. W. Irving, K. Iwama, S. Miyazaki, and Y. Morita, "Hard Variants of Stable Marriage," *Elsevier Theoretical Computer Science*, vol. 276, pp. 261–279, 2002.
[10] A. E. Roth, "Deferred Acceptance Algorithms: History, Theory, Practice, and Open Questions," *Int. J. Game Theory*, vol. 36, pp. 537–569, 2008.
[11] A. E. Roth and M. Sotomayor, *Two-sided Matching: A Study in Game Theoretic Modeling and Analysis*, ser. Econometric Society Monograph. Cambridge University Press, 1990, no. 18.
[12] D. Gale and L. S. Shapley, "College Admissions and the Stability of Marriage," *Amer. Math. Mon.*, vol. 69, no. 1, pp. 9–14, 1962.
[13] H. Liu, H. Jin, X. Liao, L. Hu, and C. Yu, "Live Migration of Virtual Machine Based on Full System Trace and Replay," in *Proc. HPDC*, 2009.
[14] M. Nelson, B. H. Lim, and G. Hutchins, "Fast Transparent Migration for Virtual Machines," in *Proc. USENIX*, 2005.
[15] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield, "Live Migration of Virtual Machines," in *Proc. NSDI*, 2005.
[16] A. E. Roth, "Stability and Polarization of Interest in Job Matching," *Econometrica*, vol. 53, pp. 47–57, 1984.
[17] D. McVitie and L. B. Wilson, "The Stable Marriage Problem," *Comm. ACM*, vol. 114, pp. 486–492, 1971.
[18] M. Baïou and M. Balinski, "Many-to-Many Matching: Stable Polyandrous Polygamy (or Polygamous Polyandry)," *Discrete Applied Mathematics*, vol. 101, pp. 1–12, 2000.
[19] T. Wood, P. Shenoy, A. Venkataramani, and M. Yousif, "Black-box and Gray-box Strategies for Virtual Machine Migration," in *Proc. NSDI*, 2007.
[20] F. Fazel and D. R. Brown III, "On the Endogenous Formation of Energy Efficient Cooperative Wireless Networks," in *Proc. Allerton*, 2009.
[21] L. Toka and P. Michiardi, "Analysis of User-driven Peer Selection in Peer-to-Peer Backup and Storage Systems," in *Proc. GameNets*, 2008.