# Stable Matching with Ties for Cloud-assisted Smart TV Services

Gyuyeong Kim, *Student Member, IEEE,* Wonjun Lee, *Senior Member, IEEE,*
Department of Computer Science and Engineering, Korea University, Seoul, Korea
wlee@korea.ac.kr

***Abstract*—Cloud-assisted services have become into the limelight owing to the recent growth of intelligent consumer electronics such as smart TVs. Virtual machine (VM) migration has substantial leverage to quality of cloud-assisted services. Proposals in the literature, which shift the issue as a stable matching problem miss a crucial scenario when preferences are not given in a strict order. We show that the our VM migration scheme that adopts the Hospital/Residents problem with Ties can solve the issue efficiently.**

## I. Introduction

Smart TVs have emerged as one of promising consumer electronics for its interactive features. Users can enjoy and share various contents of smart TVs together. Recent smart TVs utilize clouds to deliver various on-demand services. Unfortunately, intermittent failures (e.g. high latency and oversubscription) decrease the quality of the contents.

VM migration algorithm is a key factor to address poor cloud-assisted smart TV service performance issue. Xu and Li [1] show that they can formulate VM migration problem as a stable matching problem. The authors extend their general framework by applying egalitarian stable matching to prevent one-side optimal result [2]. Kim *et al.* [3] have applied the classical stable matching algorithm to address initial VM migration (i.e. VM placement) problem. Guo *et al.* [4] adopt a shadow routing to address large-scale heterogeneous servers. One of precursors [5] delivers optimal upper bounds for online migration algorithm through anayltical approach.

Both efforts [2], [3], stable matching based schemes, are prone to a crucial scenario where making a preference list in strict order is difficult. The situation arises when benefits and costs of migration to servers are indifferent. Let us consider two servers under a rack. Two servers run a partition/aggregation job and hence workloads are very similar. If the availability of resource capacity has marginal difference, it is difficult to order the preference of them strictly. Especially, partition/aggregation is a major workload pattern of MapReduce-based services (e.g. social networking services). Since smart TVs have their strength on sharing contents with friends via social networking services, addressing the issue has substantial leverages for cloud-assisted smart TV services.

This research was jointly sponsored by MEST, Korea under WCU (R33-2008-000-10044-0), MSIP, Korea under grant (No. 2013R1A2A2A01014000), MEST, Korea under Basic Science Research Program (2011-0012216), MKE/KEIT, Korea under the IT R&D program [KI001810041244, SmartTV 2.0 Software Platform], and MSIP, Korea under ITRC NIPA-2013-(H0301-13-3001), and NRF of Korea under the project of Global Ph.D. Fellowship conducted from 2012. Wonjun Lee is the corresponding author.

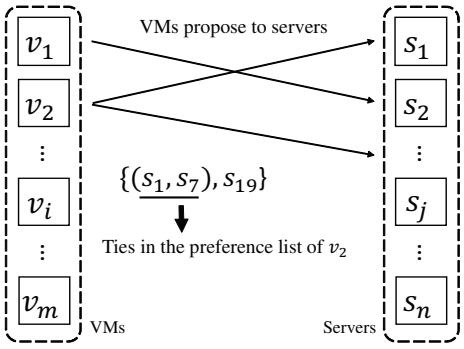


Fig. 1. A formulated stable matching between VMs and servers

In this paper, we address the problem by extending existing solutions to the Hospital/Residents problem with Ties (HRT), which allows binding identically preferred entities in a tie. We also present that our model not only embraces heterogeneity of the precursor [4], but also present stability. Since our adoption stems from moderate theoretical work, we note that discussions about optimality like Schenider [5] are available in the literature, which is omitted in this paper due to the scope. Our solution outperforms the alternative in terms of average transmission cost.

## II. Stable Matching for VM migration

In this section, we present the formulation and solution to address VM migration problem.

### A. Problem Formulation

A famous variant of the stable matching, the Hospital/Residents problem (HR) is a candidate to address VM migration problem owing to the aspect that a hospital can accept a number of residents. Figure 1 navigates how matching performs between VMs and servers.

Let $\mathcal{V} = \{v_1, v_2, ..., v_m\}$ be the set for VMs. A set $\mathcal{S} = \{s_1, s_2, ..., s_n\}$ depicts servers in clouds. In contrast with the classical one, each server $s_j$ in the HR problem has a capacity $c_j \in \mathbb{Z}^+$, which indicates the maximum number of VMs that could be migrated to $s_j$. If a VM $v$ and a server $s$ are matched, we express a matching $\mathcal{M}$ as $\mathcal{M}(v) = s$ or $\mathcal{M}(s) = v$. Since the numbers of VMs and servers are not the same, each entity has an incomplete preference list.

A server $s_j$ is *undersubscribed* with a matching $\mathcal{M}$ if $\|\mathcal{M}(s_j)\| < c_j$. We call a matched pair $(v_i, s_j)$ as a *blocking*

**Algorithm 1** An algorithm for a super-stable matching

```
assign each VM and server to be free;
    full(S) := false;
while some VM v is free and a list is not exhausted do
    for all server s at the head of v's list do
    v proposes to s
        if s is oversubscribed then
            for all VM w' at the tail of s's list do
                if w' is provisionally assigned to s then
    break the assignment;
                end if
    delete the pair (w',s);
                /*delete each from the list vice versa */
            end for
        end if
        if s is full then
    full(s):= true;
    w := worst VM provisionally assigned to s;
            for all strict successor w' of w on s' list do
    delete the pair (w',s);
            end for
        end if
    end for
end while
```

*pair* if $v_i$ and $s_j$ discover each other acceptable; a VM $v_i$ is single or strictly prefers a server $s_j$ to its current matched server in $\mathcal{M}$; a server $s_j$ is undersubscribed or strictly prefers a VM $v_i$ to the worst VM, which is matched to $s_j$ in $\mathcal{M}$. A matching $\mathcal{M}$ is *stable* if it is not blocked by any individual or blocking pair. Every problem instance admits at least one stable matching $\mathcal{M}$ in the formulation.

### B. The Extension with Ties

We now extend the model with ties. If a system permits ties, then each VM/server can situate more than two opposite entities, which have the identical preference in a tie. In Figure 1, we can observe that $s_2$ and $s_7$ are in a tie of $v_2$.

We apopt a matching algorithm, which find a stable matching $\mathcal{M}$ in Algorithm 1. By allowing ties, a VM proposes all servers in a tie of the preference list simultaneously. With termination of the algorithm, the number of assigned servers for a VM $v$ is decreased to the only one server. This straightforward property produces notable results on VM transmission cost, and we examine the result in the next section.

In HRT, there exist a stability, which is named super-stability.

**Super stability**: We say a matching $\mathcal{M}$ is *super-stable* if $\mathcal{M}$ admits no blocking pair, where a blocking pair is a pair $(v_i, s_j)$, which meets four conditions as follows. $(v_i, s_j) \notin \mathcal{M}$; $v_i$ and $s_j$ discover each other acceptable; a VM $v_i$ is single or strictly prefers a server $s_j$ to its current matched server in $\mathcal{M}$ or is indifferent between them; a server $s_j$ is undersubscribed or strictly prefers a VM $v_i$ to the worst VM, which is matched to $s_j$ in $\mathcal{M}$ or is indifferent between them.

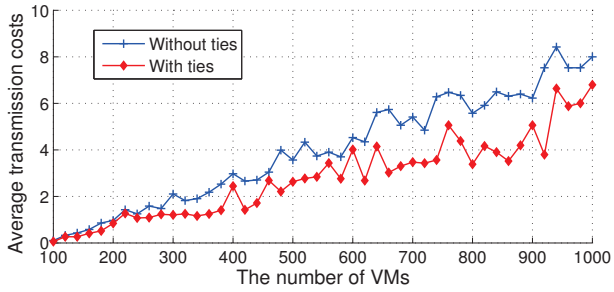


Fig. 2. Comparison against the alternative

## III. Simulation

We compare our proposal with the alternative, a stable matching algorithm without ties. We gracefully adopt a metric, average VM transmission cost [2], which is denoted as $\mathcal{D}_{s(v),s}/\theta_{s(v),s}$, where $\mathcal{D}_{s(v),s}$ is the hop distance to a target server $s$ from the residing server $s(v)$ of a VM $v$, and $\theta_{s(v),s}$ denotes the available bandwidth between $s(v)$ and $s$. The average bandwidth is set to 1 Gbps. Commonly, the capacity of servers is uniformed and hence we set the capacity of servers 4. In the alternative, a VM sorts servers in a tie randomly. We run tests 100 times to obtain averages with an environment where the numbers of VMs are diverse.

The result in Figure 2 denotes that our proposal outperforms the alternative in terms of transmission cost. The low transmission cost of our scheme implies that VMs are assigned to the server where they preferred more. In other words, VMs have matched to the server, which is minimizing hop distances. The difference stems from that, the tie-missed algorithm orders indifferent entities compulsorily. As a result, the system computes inefficient matchings.

The tendency shows that our proposal results noticeable difference as the number of VM increases. When the number of VM is around 920, our scheme outperforms the alternative about two times. By considering common clouds are consisted of a huge number of servers, we can derive that our solution can be a qualified candidate to improve the performance.

## IV. Conclusion

We discussed a VM live migration problem when preferences are identical for cloud-assisted smart TV services. We formulated the problem as the Hospital/Residents problem with Ties. The simulation result shows that our proposal outperms the alternative in terms of average VM transmission cost.

## References


[1] H. Xu and B. Li, "Seen as stable marriages," in *Proc. of IEEE INFOCOM*, 2011.

[2] ——, "Egalitarian stable matching for vm migration in cloud computing," in *Proc. of IEEE INFOCOM Workshop on Cloud Computing*, 2011.

[3] G. Kim, H. Park, J. Yu, and W. Lee, "Virtual machines placement for network isolation in clouds," in *Proc. of ACM RACS*, 2012.

[4] Y. Guo, A. L. Stolyar, and A. Walid, "Shadow-routing based dynamic algorithms for virtual machine placement in a network cloud," in *Proc. of IEEE INFOCOM*, 2013.

[5] J. Schneider and S. Schmid, "Optimal bounds for online page migration with generalized migration costs," in *Proc. of IEEE INFOCOM*, 2013.