Bargaining Towards Maximized Resource Utilization in Video Streaming Datacenters

Yuan Feng, Baochun Li
Department of Electrical and Computer Engineering
University of Toronto

Bo Li
Department of Computer Science
Hong Kong University of Science and Technology

***Abstract*—Datacenters can be used to host large-scale video streaming services with better operational efficiency, as the multiplexing achieved by virtualization technologies allows different videos to share resources at the same physical server. Live migration of videos from servers that are overloaded to those that are under-utilized may be a solution to handle a flash crowd of requests, in the form of virtual machines (VMs). However, such migration has to be performed with a well-designed mechanism to fully utilize available resources in all three resource dimensions: storage, bandwidth and CPU cycles. In this paper, we show why the challenge of maximizing resource utilization in a video streaming datacenter is equivalent to maximizing the joint profit in the context of *Nash bargaining solutions*, by defining utility functions properly. Having servers participating as players to bargain with each other, and VMs as commodities in the game, trades conducted after bargaining govern VM migration decisions in each server. With extensive simulations driven by real-world traces from UUSee Inc., we show that our new VM migration algorithm based on such Nash bargaining solutions increases both the resource utilization ratio and the number of video streaming requests handled by the datacenter, yet achievable in a lightweight fashion.**

I. Introduction

As an efficient means of providing computing resources in a form of utility, cloud computing has recently attracted a substantial amount of attention from both industry and academia. The paradigm shift to cloud computing is driven by strong demand, especially from enterprises, to improve the overall efficiency of using and managing computing resources. Cloud service providers use *datacenters* to provision a shared pool of computation, storage and bandwidth resources, to be used by applications when the need arises. Since resources at datacenters are shared by using virtualization, applications are allowed to statistically multiplex such resources in the form of virtual machines.

A large-scale video streaming service requires both computation and bandwidth. Due to its highly varied demand from users, it is one of the prime examples that migration to datacenters in the cloud becomes more economical. Take the video streaming service offered by NetFlix Inc. as an example, with widely varying user demand for bandwidth and the fact that Internet Service Providers (ISPs) bill for 95% of the peak bandwidth usage, it would be much more economical to use cloud services rather than deploying privately owned media servers.

This research was supported in part by a grant from NSFC/RGC under the contract N_HKUST610/11, by grants from HKUST under contracts RPC11EG29, SRFI11EG17-C and SBI09/10.EG01-C, by a grant from Huawei Technologies Co. Ltd. under the contract HUAW18-15L0181011/PN, and by a grant from Guangdong Bureau of Science and Technology under the contract GDST11EG06.

We have also observed from our first-hand real-world experiences that demand for video streaming services may peak at different times. Fig. 1 shows the normalized population of two videos in three days by analyzing 200 Gigabytes worth of operational traces that we collected at UUSee Inc. [2], one of the leading peer-assisted video streaming providers in China. As we can see, Video 1 (an on-demand stream) and Video 2 (a live stream) have peaked on August 12 and August 13, 2008, respectively. If privately owned media servers are used by provisioning for peak bandwidth usage, bandwidth will remain severely under-utilized during off-peak times. It is an economically sound decision to migrate video streaming services to datacenters in the cloud.

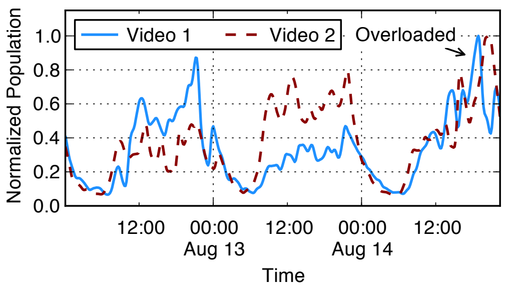


Fig. 1. The normalized population of two videos from August 12 to 14, 2008.

Once the decision is made to host video streaming services with datacenters in the cloud, the question becomes how datacenter resources can be better utilized. A datacenter consists of tens of thousands of physical servers. Since videos may reach their respective peak demand at different times, there is an opportunity to increase server utilization ratios by letting multiple videos share physical resources on the same server. With virtualization ubiquitously used in Infrastructure-as-a-Service cloud platforms, several VMs, each of which packaging one video, are able to be placed on the same physical server and benefit from "on-demand" resource provisioning. Specifically, resources at a physical server are shared among all VMs during off-peak seasons to achieve a higher utilization ratio; once one of the videos encounters its peak in demand, VMs packaging this video can use up all available resources at servers where they are placed, in order to satisfy as many requests as possible.

One possible challenge we may encounter is that servers may become overloaded when a video encounters highly bursty requests, or several videos placed on the same server reach their peak period in demand at the same time. Shown in Fig. 1, both Video 1 and Video 2 reached their peaks on August 14. A naive solution will be to move VMs away from overloaded servers to under-utilized ones. Thanks to the support of live migration with virtual machines [12], migrating from one server to another with minimal downtime to the streaming service is feasible in datacenters. With live VM migration, the number of requests being handled at the same time may be effectively increased by migrating VMs with additional resource needed from resource-deficient to resource-rich servers.

Nevertheless, we argue that such migration should be planned with care, by fully exploring all possibilities of utilizing currently available resources to handle the increased requests. Since servers in the datacenter provide resources in three main dimensions of storage, bandwidth and CPU in a tightly-coupled manner, utilization of resources may gradually become severely unbalanced across different dimensions, which implies unnecessary idling of available resources. Since datacenters are expensive to build and to operate, it is a waste of both investment and energy when resources are under-utilized [7].

In this paper, we seek to design an efficient and practical algorithm to maximize resource utilization, with all three dimensions considered, by migrating VMs across servers in a video streaming datacenter. We first formally formulate the challenge of maximizing system-wide resource utilization as a centralized optimization problem, taking into account practical constraints of storage, bandwidth, and CPU cycles. The optimal solution dictates the destination server to which VMs should be migrated. We find that the optimization problem is in the form of a *Generalized Assignment Problem* (GAP) in integer programming, which is NP-hard and even APX-hard to approximate. Obtaining a near-optimal solution to this problem requires to decouple it into several 0-1 knapsack problems and iterate hundreds of times.

Inspired by the power of markets in arbitrating decisions of both buyers and sellers in a decentralized fashion, we relate the entire datacenter to a bargaining market. VMs are considered as *"commodities"* in this market. Every server makes its decision by participating in this market and bargaining for its desired commodities. We model this market as a *Nash bargaining game*, and prove that the problem of maximizing resource utilization in a datacenter is equivalent to that of maximizing the joint profit in the Nash bargaining solution. The VM migration decisions are governed by the individually made bargaining strategies in each server in a *laissez-faire* manner, which avoids additional CPU consumption at a centralized decision maker and reduces the bandwidth cost of VM migration.

The remainder of this paper is organized as follows. In Sec. II, we formulate the challenge of maximizing resource utilization in virtualized datacenters and motivate to use the Nash bargaining solution to this problem. Sec. III shows that the formulation is equivalent to the joint profit maximization problem in a Nash bargaining solution, and describes the VM migration algorithm. In Sec. IV, we show the effectiveness of our VM migration algorithm by presenting an in-depth analysis, driven by real-world traces from UUSee Inc. We discuss related work and conclude the paper in Sec. V and Sec. VI, respectively.

II. Maximizing Resource Utilization in Video Streaming Datacenters

We first present an example to show that VM migration may help to fully utilize resources in all three dimensions to handle more requests, and then formulate the problem of maximizing resource utilization in video streaming datacenters.

A. Benefits of VM Migration

Satisfying one request for streaming a video, at High-Definition (HD) or Standard-Definition (SD) quality levels, requires different amounts of bandwidth. The variety of existing media formats requires on-demand transcoding, which results in distinct requirements on CPU cycles when processing a video streaming request. As the number of requests for each video fluctuates over time, the required amount of resources for each video in dimensions of storage, bandwidth and CPU cycles may increase in an unbalanced manner. With live migration of virtual machines (VMs), we may migrate VMs across the boundary of servers in a datacenter, in order to fully explore possibilities of utilizing available resources, which leads to more concurrent requests being handled.

The following conceptual and proof-of-concept example illustrates the potential benefits with such migration. Consider a video streaming datacenter with two servers and three videos. Each of the videos is served by a corresponding VM: VM_1, VM_2, and VM_3 respectively. In this example, Video 1 represents a live video streaming service, *e.g.*, with the standard-definition (SD) quality level, and with network coding adopted in transmission, which require more CPU but less bandwidth resources. Video 2 represents an on-demand streaming (VoD) service, *e.g.*, in HD quality without using network coding during transmission, which requires more bandwidth, but with relatively low CPU demand. Video 3, agin, represents a live streaming service, but no network coding is involved in its transmission. Resources required to handle one request for each video are summarized in Table I, along with resource capacities at servers. We note that there are no additional storage resources required for handling each request in video streaming services. The size of each video is assumed to be 4 GB is this example.

TABLE I
REQUIRED RESOURCES TO ACCOMMODATE ONE REQUEST IN EACH VM, AS WELL AS THE RESOURCE CAPACITY IN EACH SERVER.

Resources	Server 1/2	VM_1	VM_2	VM_3
Storage Space (GB)	8	–	–	–
Bandwidth (Mbps)	9	1	3	1
CPU (MIPS)	10	3	1	2

Suppose initially Video 1 is popular and Video 2 and 3 are less popular, as there are five requests for Video 1 and one request for Video 2 and 3, respectively. It is not difficult to find out that the optimal resource utilization strategy is to have the popular video provided by two servers, each of which is concurrently multiplexed by one unpopular video. That is, placing $(\mathrm{VM}_1, \mathrm{VM}_2)$ in one server and $(\mathrm{VM}_1, \mathrm{VM}_3)$ in another server, shown in Fig. 2. In this fit, a total of seven requests can be handled at the same time.

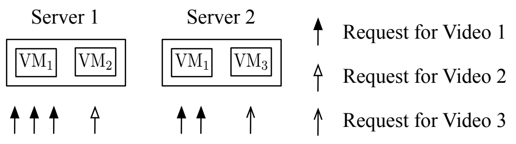


Fig. 2. The initial fit of three videos on two servers.

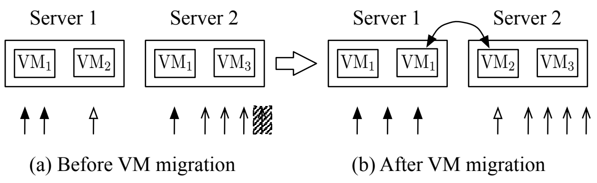


Fig. 3. Improved video fit after VM migration.

Due to the variation of requests over time, the popularity of videos changes as time elapses, with, say, three requests for Video 1, one request for Video 2 and four requests for Video 3 at this time. No possible migration plan exists if we use the naive solution of moving VMs away from overloaded to under-utilized servers, which means one of the eight requests will not be satisfied at the same time due to the unbalanced increase of resource demand in bandwidth and CPU cycles, no matter how the three requests for Video 1 is scheduled. Fig. 3(a) indicates one possible case, where one request for Video 1 is directed to Server 2. The heavily utilized CPU in Server 2 results in a drop of one request for Video 3, which also requires CPU. As a consequence, one of the requests can not be satisfied immediately, *even as resources are still available on the two servers.*

However, if VM migration is conducted in the datacenter, we can swap VM_2 in Server 1 and VM_1 in Server 2, which is shown in Fig. 3(b). In such a scenario, the VM, which requires more resource in one dimension, is fit into the server with another VM which requires less resource in that dimension in a complementary manner. The eight requests for three videos can be satisfied by the two servers at the same time, which leads to a higher level of resource utilization.

From this example, we can see that by migrating VMs across the boundary of servers, a datacenter is able to better utilize available resources and handle more requests at the same time.

B. Maximizing Resource Utilization in a Generalized Form

Our primary objective in this paper is to find out how VM migration strategies should be designed so that resource utilization in datacenters is maximized. We first present the context of our discussion and a model of a datacenter.

Instead of restricting ourselves to video streaming datacenters, we discuss this problem in a generalized form so that the designed algorithm is also applicable to datacenters holding general application instances. We consider a datacenter constituting a set of heterogeneous servers, denoted by $\mathcal{N}$. For every server $i \in \mathcal{N}$, the amount of storage space capacity is C_i, in Gigabytes; the amount of bandwidth capacity is U_i, in Mbps; and the amount of CPU computing capability is P_i, in MIPS.

Let the set of application instances hosted by the datacenter be denoted by $\mathcal{M}$. For any $k \in \mathcal{M}$, VM_k represents the corresponding virtual machine serving that application. Operational datacenters typically provide customers the flexibility to choose from a number of different VM instances equipped with different amounts of resources in each dimension. For example, Amazon Elastic Compute Cloud (EC2) provides *high-memory*, *micro* and *high-CPU* instances [1], in order to serve applications with different resource demands. To better indicate the amount of dedicated resources used by each VM, let s_k be the amount of storage space, b_k be the amount of bandwidth, and cl_k be the amount of computing capability devoted by VM_k to handle one request when serving its corresponding application.

We are aware that it is possible that some applications may span over a set of VMs, such as Map-Reduce computational jobs and multi-tier Web services. These applications may place further restrictions on the *locality* of their own set of VMs, since they may require a large amount of inter-VM bandwidth. To serve these applications well, a set of VMs serving the same application should be located in the same server. In this case, our model considers the set of VMs serving one application as *a special* VM, in a sense that they are considered as an single entity. To be exact, for an application $k' \in \mathcal{M}$, $\mathrm{VM}_{k'}$ represents the special VM if k' spans over a set of VMs, and each $\mathrm{VM}_{k'}$ requires storage space $s_{k'}$, bandwidth $b_{k'}$, and computing capability $cl_{k'}$ to handle one request. Since descriptions of special VMs are in essence the same as regular VMs in our model, we do not distinguish k and k' in subsequent analysis.

Let $I_i^k(t)$ be the binary variable that indicates whether VM_k is stored in server i at time t.

$$I_i^k(t) = \begin{cases} 1 & \mathrm{VM}_k \text{ is stored in server } i \text{ at time } t \\ 0 & \text{otherwise} \end{cases}$$

Let $D_i^k(t)$ denote the number of requests for VM_k to which server i handles at time t.

Under the assumption that all servers in the datacenter possess global knowledge of which set of VMs other servers have and how many requests each server handles with respect to each of its VMs, at time t, the centralized resource utilization maximization problem in a datacenter can be formulated as a binary integer programming problem:

$$\max_{\hat{I}_i^k(t)} \quad \frac{1}{|\mathcal{N}|} \sum_{i \in \mathcal{N}} \hat{R}_i(t) \tag{1}$$

$$\text{s.t.} \quad \sum_{k \in \mathcal{M}} \hat{I}_i^k(t) s_k \hat{D}_i^k(t) \leq C_i, \; \forall i \tag{2}$$

$$\sum_{k \in \mathcal{M}} \hat{I}_i^k(t) b_k \hat{D}_i^k(t) \leq U_i, \; \forall i \tag{3}$$

$$\sum_{k\in\mathcal{M}} \hat{I}_i^k(t) cl_k \hat{D}_i^k(t) \leq P_i, \, \forall i \quad (4)$$

$$\sum_{i\in\mathcal{N}} \hat{I}_i^k(t) = 1, \, \forall k, \quad (5)$$

where $\hat{R}_i(t)$ is the estimated resource utilization ratio at server i after time t, based on the current number of requests and information about resource capacities, and $\hat{D}_i^k(t)$ is the estimated number of requests for VM$_k$ to which server i handles after time t, which has the following form:

$$\hat{D}_i^k(t) = \sum_{i\in\mathcal{N}} D_i^k(t) I_i^k(t).$$

Note that $\forall i$ at time t, there is only one $I_i^k(t) = 1$, since each VM$_k$ can be possessed by only one server at one time.

To capture three integrated dimensions of resource usage at each server, we define the estimated resource utilization ratio at each server to be the weighted sum of estimated resource utilization ratios in dimensions of storage, bandwidth and CPU:

$$\begin{aligned} \hat{R}_i(t) &= \omega_i^s(t)\hat{r}_i^s(t) + \omega_i^b(t)\hat{r}_i^b(t) + \omega_i^c(t)\hat{r}_i^c(t) \\ &= \omega_i^s(t)\frac{\sum_k \hat{I}_i^k(t) s_k \hat{D}_i^k(t)}{C_i} + \omega_i^b(t)\frac{\sum_k \hat{I}_i^k(t) b_k \hat{D}_i^k(t)}{U_i} \\ &\quad + \omega_i^c(t)\frac{\sum_k \hat{I}_i^k(t) cl_k \hat{D}_i^k(t)}{P_i} \end{aligned} \quad (6)$$

where $\omega_i^s(t), \omega_i^b(t)$ and $\omega_i^c(t)$ are the weights given to resources in different dimensions according to server i's current resource usage states, constrained by $\omega_i^s(t) + \omega_i^b(t) + \omega_i^c(t) = 1$.

In this formulation, $\hat{I}_i^k(t)$ is the optimization variable, which is the binary indicator to denote the placement of each VM after time t based on the information at present. Inequality (2) stands for the storage space constraint; Inequality (3) represents the bandwidth capacity constraint; and Inequality (4) denotes the CPU computing capability constraint. The rationale behind this is that resources consumed by every server can not exceed the resource capacity in each dimension. Eq. (5) ensures that each VM can only be possessed by one server at a time. In a centralized manner, the current VM placement indicator $I_i^k(t)$, the number of requests handled $D_i^k(t)$, and resource capacities of each server C_i, U_i, P_i are supposed to be known at time t.

In our subsequent analysis, we focus on decisions at a specific time. Therefore, the time indices t in the expressions are dropped, as we obtain the following optimization problem equivalent to the original one (1).

$$\max_{\hat{I}_i^k} \sum_i \sum_k \left(\frac{\omega_i^s s_k \hat{D}_i^k}{C_i} + \frac{\omega_i^b b_k \hat{D}_i^k}{U_i} + \frac{\omega_i^c cl_k \hat{D}_i^k}{P_i} \right) \hat{I}_i^k \quad (7)$$

To this end, we have formally described the problem of maximizing resource utilization in a datacenter in a centralized manner by optimization problem (7). Based on existing knowledge, we seek to find out what is the best placement strategy of each VM under the constraint of resource capacities in dimensions of storage, bandwidth and CPU computing capability, so that the resource utilization is maximized.

C. Lagrangian Heuristic vs. Nash Bargaining Solution

The formulation is a comprehensive integer optimization problem, which appears to be in the form of a multi-dimensional Generalized Assignment Problem (GAP). The GAP is NP-hard, and it is even APX-hard to approximation [8]. Though we may approach the optimal solution through the Lagrangian relaxation heuristic: decoupling it into several 0-1 knapsack problems, it incurs high computational complexity. Since our objective is to design a practical VM migration algorithm that can be implemented in real-world settings, we seek to design alternative feasible heuristics.

In this paper, we propose to use the *Nash bargaining solution* to solve the utilization maximization problem. The Nash bargaining game discusses the situation in which two or more players reach an agreement regarding how commodities are to be distributed among them, so that the social utility gains are maximized and commodities owned by each player do not exceed its capacity. This is exactly the same as the GAP, which aims to assign a set of objects to agents so that the total profit of the assignment is maximized and all agents do not exceed their budget. With the same objective, we believe that the mechanism of the Nash bargaining solution is a suitable alternative.

III. VM Migration Algorithm Based on Nash Bargaining Solution

In this section, we prove that the Nash bargaining solution in the bargaining game can be used to solve the resource utilization maximization problem in virtualized datacenters. The VM migration algorithm based on the Nash bargaining solution is presented in detail.

A. The Nash Bargaining Solution

Bargaining problems are known as non-zero-sum games that participating players try to achieve a win-win situation. In the Nash bargaining game, there is always a solution for the optimal strategy at each player, which guarantees that their average payoff is maximized under the assumption that opposing players also use the optimal strategy.

In Nash bargaining games, each player has a different anticipation to each commodity, which represents a state of expectation that may involve the certainty of some contingencies and various probabilities of other contingencies [11]. For example, if Bill prefers apple to banana, then he may have a higher anticipation of apple than that of banana. The utility of each player is a function of his anticipations to commodities he has. The Nash bargaining solution is a Pareto efficient solution to a Nash bargaining game so that the joint profit, which is the product of utility gains of all players, is maximized.

Let $\mathcal{F}_i$ be the utility function for player i. Rational players will seek to maximize the *Nash product* $\prod G_i$, where $G_i = |\mathcal{F}_i(x) - \mathcal{F}_i(d)|$. $\mathcal{F}_i(d)$ is the *status quo* utilities (*i.e.*, the utility obtained if one decides not to bargain with other players). Nash has shown that obtaining the maximum of the Nash product will attain the Pareto-optimal solution for the bargaining situation.

***Theorem** 1:* The problem of maximizing resource utilization in a virtualized datacenter is equivalent to the joint profit maximization problem in the Nash bargaining game.

Proof: Envision a market, in which servers are treated as players and VMs are considered as commodities. The Nash bargaining game here is to exchange VMs among the set of servers so that their joint profit is maximized. Let A_i^k be player i's anticipation to commodity k. Define the utility function of player i to be $\mathcal{F}_i$, which is represented as follows:

$$\mathcal{F}_i(x) = \mathcal{F}_i(d) + \exp(\sum_{k\in\mathcal{M}} A_i^k(\hat{I}_i^k - I_i^k)).$$

The definition can be explained as the utility of player i after bargaining equals its status quo utility plus a function of added anticipations during bargaining. That is to say, the utility gain of player i can be represented as:

$$\begin{aligned} G_i &= |\mathcal{F}_i(x) - \mathcal{F}_i(d)| \\ &= \exp(\sum_{k\in\mathcal{M}} A_i^k(\hat{I}_i^k - I_i^k)). \end{aligned} \tag{8}$$

In practice, the Nash bargaining solution aims to maximize the Nash Product $\prod G_i$, which can be interpreted as follows:

$$\max \quad \prod_{i\in\mathcal{N}} G_i \tag{9}$$

$$\text{s.t.} \quad \sum_{k\in\mathcal{M}} \hat{I}_i^k s_k \hat{D}_i^k \le C_i,\ \forall i \tag{10}$$

$$\sum_{k\in\mathcal{M}} \hat{I}_i^k b_k \hat{D}_i^k \le U_i,\ \forall i \tag{11}$$

$$\sum_{k\in\mathcal{M}} \hat{I}_i^k cl_k \hat{D}_i^k \le P_i,\ \forall i \tag{12}$$

$$\sum_{i\in\mathcal{N}} \hat{I}_i^k = 1,\ \forall k. \tag{13}$$

Constraints (10), (11) and (12) represent the fact that commodities each player owns can not exceed its capacity. Eq. (13) confirms that each commodity can only be possessed by one player at a time.

Substitute (8) into (9), the maximization problem in the Nash bargaining game is equivalent to the following ones.

$$\begin{aligned} \max \prod_{i\in\mathcal{N}} G_i &\iff \max \exp\log \prod_{i\in\mathcal{N}} G_i \\ &\iff \max \exp \sum_{i\in\mathcal{N}} \log G_i \\ &\iff \max \sum_{i\in\mathcal{N}} \sum_{k\in\mathcal{M}} A_i^k \hat{I}_i^k. \end{aligned}$$

If we define player i's anticipation to commodity k, *i.e.*, A_i^k in the following form:

$$A_i^k = \frac{\omega_i^s s_k \hat{D}_i^k}{C_i} + \frac{\omega_i^b b_k \hat{D}_i^k}{U_i} + \frac{\omega_i^c cl_k \hat{D}_i^k}{P_i},$$

the optimization problem (9) in the Nash bargaining game can be rewritten as:

$$\max \sum_{i\in\mathcal{N}} \sum_{k\in\mathcal{M}} \left(\frac{\omega_i^s s_k \hat{D}_i^k}{C_i} + \frac{\omega_i^b b_k \hat{D}_i^k}{U_i} + \frac{\omega_i^c cl_k \hat{D}_i^k}{P_i} \right) \hat{I}_i^k,$$

which is equivalent to the resource utilization maximization problem (7) in virtualized datacenters. It then follows that the joint profit maximization problem in the Nash bargaining solution is equivalent to the resource utilization maximization problem in virtualized datacenters.

■

Based on the established connection between the two problems, the optimization variable $\hat{I}_i^k$ can be viewed as an indicator of the strategy adopted by each player in the bargaining game. The estimated resource utilization ratio of server i, $\hat{R}_i$, can be treated as the profit gains of player i by adopting strategy $\hat{I}_i^k$. This implies that the resource utilization maximization problem can be solved using the mechanism of Nash bargaining solution.

B. VM Migration in the Bargaining Game

VM-based market organization. We envision the existence of an online market place, where all servers in a datacenter behave as *players*; application VMs are treated as *commodities*. Every VM is associated with an anticipation A_i^k from each player, which is the evaluation of VM$_k$ from server i's perspective based on its own information. According to the proof of Theorem 1, A_i^k should be of the following form:

$$A_i^k = \omega_i^s \frac{s_k \hat{D}_i^k}{C_i} + \omega_i^b \frac{b_k \hat{D}_i^k}{U_i} + \omega_i^c \frac{cl_k \hat{D}_i^k}{P_i}. \tag{14}$$

In our bargaining market, the anticipation of player i to each commodity is defined as the consumed fraction of resources in the corresponding application VM in server i, with respect to dimensions of storage, bandwidth and CPU computing resources. Since our objective is to fully utilize resources in every server, application VMs requiring more resources will be more valuable. However, fractions in different dimensions should be treated differently, which are weighted by ω_i^s, ω_i^b and ω_i^c according to Player i's current resource usage states. The rationale is that resources with a high utilization ratio will become the "bottleneck" towards fully utilizing resources in other dimensions, hence should be less desirable by that server. For simplicity, we define the weights to be inversely proportional to the current resource utilization ratio in the corresponding dimension. That is,

$$\omega_i^s = \frac{\frac{1}{r_i^s}}{\sum \frac{1}{r}},\ \omega_i^b = \frac{\frac{1}{r_i^b}}{\sum \frac{1}{r}},\ \omega_i^c = \frac{\frac{1}{r_i^c}}{\sum \frac{1}{r}},$$

where $\sum \frac{1}{r} = \frac{1}{r_i^s} + \frac{1}{r_i^b} + \frac{1}{r_i^c}$. Recall that r_i^s, r_i^b and r_i^c are defined as the current resource utilization ratio of server i in the dimension of storage, bandwidth and CPU computing, respectively, *i.e.*,

$$\begin{aligned} r_i^s &= \frac{\sum_{k\in\mathcal{M}} I_i^k s_k D_i^k}{C_i} \\ r_i^b &= \frac{\sum_{k\in\mathcal{M}} I_i^k b_k D_i^k}{U_i} \\ r_i^c &= \frac{\sum_{k\in\mathcal{M}} I_i^k cl_k D_i^k}{P_i}. \end{aligned}$$

The bargaining strategy based on spacial representation. It is proved that the problem of determining whether there exists a Nash equilibrium in which each player has a specific minimum payoff is NP-complete as a function of the number of players [4], so that the Nash bargaining solution in a bargaining game is usually approximated by numerical methods such as the Newton's method or the bisection method [5], which require numerous iterations. Since our objective is a practical VM migration algorithm that can be implemented in real-world datacenters and executed in a lightweight fashion, we propose to adopt a bargaining strategy based on the *spacial representation* of Nash bargaining games [16].

This bargaining strategy based on the spacial representation fits our design objective in virtualized datacenters, since it reduces the computational overhead significantly by eliminating the requirement of generating the utility gain for every single possible set of strategy execution. By introducing the *utility-distance product* ϕ_i^k, a function of the anticipation A_i^k, it is proved that the utility-distance product of a commodity is analogous to the moment of force by weights based on a lever system. By suitably locating a pivot location such that the distribution of the utility-distance product is uniformly positioned about a pivot, equilibrium can be achieved.

From a spacial perspective, outcomes of games have been assumed to lie in some low-dimensional Euclidean space, such that anticipations to the players are defined in terms of distances from them [13]. In such spacial games, anticipation of a player to a commodity is assumed to be an inverse function of the distance that commodity lies from the player [4], such that commodities of *higher* anticipation values have a *closer* spatial proximity (*i.e.,* a shorter spatial distance).

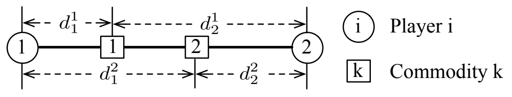


Fig. 4. Spacial representation of a 2-player game.

For example, Fig. 4 shows the 2-dimensional spatial representation of a 2-player game with player-to-player distances defined to be constants. Anticipations of the two players to commodities 1 and 2 are clearly reflected by spacial distances d_i^1 and d_i^2, where $i \in \{1, 2\}$. In the example, commodities are represented as points based on their spatial proximities, which lie within the boundary enclosed by all participating players. The relative distances of a commodity k to player i is defined as d_i^k, which is in the form of:

$$d_i^k = f(A_i^k) = \frac{1/A_i^k}{\sum_i (1/A_i^k)} \ \forall i \in \mathcal{N},\, k \in \mathcal{M}. \qquad (15)$$

The distances of each commodity to all players are normalized so that they add up to a unitary value, *i.e.*, $\sum_i d_i^k = 1$, $\forall k$. In our example, player 1 has higher anticipation to commodity 1 and player 2 prefers to commodity 2.

In this bargaining strategy, each player possesses commodities sorted by their relative distance d_i^k, such that commodities with higher anticipations will be given higher priority. The

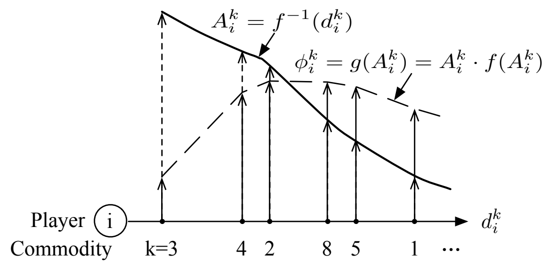
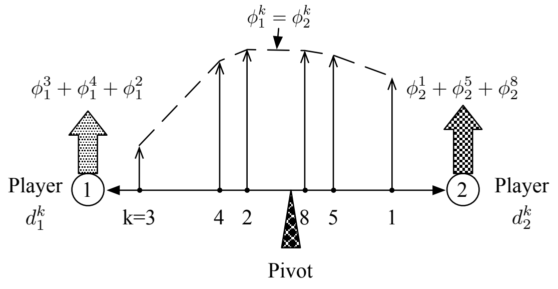


Fig. 5. The utility-distance product of commodities of Player i.

utility-distance product of a player to a commodity is defined as $\phi_i^k = d_i^k \cdot A_i^k$. Fig. 5 shows how commodities are sorted by player i and the associated utility-distance product ϕ_i^k of each commodity.

From a mechanical perspective, weights on a lever are aligned along the same direction such that weights on the left hand side generate a collective moment that opposes the moment caused by weights on the right. Similarly, to maintain an equilibrium in a bargaining game, the sum of utility-distance products of all commodities should be equally divided among all participating players, as shown in Fig. 6. Players 1 and 2 are considered to be lying at the end points of the lever, where forces can be applied. The utility-distance products of commodities are regarded as weights.



$\phi_1^k = \phi_2^k$
$\phi_1^3 + \phi_1^4 + \phi_1^2$
$\phi_2^1 + \phi_2^5 + \phi_2^8$
Player
1
d_1^k
k=3
4
2
8
5
1
2
Player
d_2^k
Pivot

Fig. 6. Finding the pivot point in a lever system.

It is then not difficult to find out that the pivot point of the lever in our example should be lying between commodities 2 and 8, where the collective moment generated by weights $\phi_1^3 + \phi_1^4 + \phi_1^2$ on Player 1's side equals to that of $\phi_2^1 + \phi_2^5 + \phi_2^8$ on Player 2's side. To be precise, the pivot point in a lever system is determined by balancing weights between two end points, which is:

$$\mu = \frac{1}{2} \sum_{k \in \mathcal{M}} \phi_i^k.$$

After determining the pivot point, it is natural that the bargaining solution is to assign commodities lying on the left hand side of the pivot point to player 1, and commodities on the right hand side to player 2.

This bargaining strategy can be easily generalized to multi-player bargaining games. For the ideal condition whereby all commodities lie in a space between vertices representing all players, the determination of a pivot location can be based on balancing the utility-distance product with respect to all players, which is given by:

$$\mu_i = \frac{1}{|\mathcal{N}|} \sum_{k \in \mathcal{M}} \phi_i^k, \forall i \in \mathcal{N}. \quad (16)$$

Fig. 7 shows an example of a 3-player game. The vertical arrows on each commodity represents the cumulative utility-distance product p_i^k, which is defined as the sum of utility-distance products of commodities whose relative distances are not greater than that of k, *i.e.*, $p_i^k = \sum_{\forall j \in \mathcal{M}, d_i^j \le d_i^k} \phi_i^j$. By finding the pivot point where $p_1^{k_1} = p_2^{k_2} = p_3^{k_3}$, the bargaining solution to the game is found, which is, in our example, $\mathcal{C}_1 = \{1, 3, 5, 7\}$, $\mathcal{C}_2 = \{2, 8, 9, 12\}$, and $\mathcal{C}_3 = \{4, 6, 10, 11\}$.

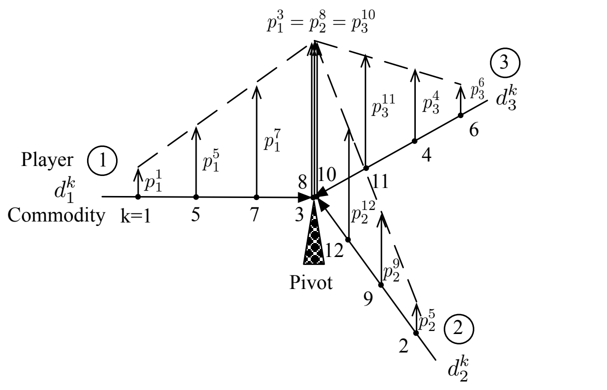


Fig. 7. Finding the pivot point in a 3-player game according to the utility-distance products.

After each player determines its own pivot point μ_i, commodities inside the market will be assigned to each player accordingly. To be precise, each player will be assigned a set of commodities $\mathcal{C}_i$, so that the sum of utility-distance products of $\mathcal{C}_i$ is maximized but not larger than μ_i. If one commodity k is assigned to more than one players at one time, the player who has the smallest relative distance d_i^k will obtain this commodity. To conclude, our VM migration algorithm based on the Nash bargaining solution is summarized in **Algorithm 1**.

Practicality. Though migrating VMs have potential benefits to improve the resource utilization ratio, it does not come without substantial upfront costs of bandwidth. An example orchestration of live VM and storage migration on the testbed through HARMONY shows that the transaction throughput is reduced by 12% during VM migration [14]. Since the application performance may be negatively affected by live VM migration, it should be avoided as much as possible.

As a consequence, our VM migration algorithm is triggered in a *laissez-faire* manner. Whenever the resources provided by one server can not sustain requests for applications placed on that server, the migration algorithm is triggered. The idea is, if requests can be satisfied under the current provision, we'll maintain the same even if the resource utilization is not the optimal, *e.g.*, when the number of requests for one application decreases dramatically at some time. Note that it might be the case that even after VM migration, the application fit in the datacenter still can not handle all requests at the same time, *i.e.*, the total available resources is less than the sum of required resources for all requests. In this scenario, the only solution for the datacenter is to add new servers. To avoid triggering the VM migration algorithm constantly in this scenario, we restrict the minimum interval between two trigger points to be T, where $T = 20$ min in our simulation.

Algorithm 1 *The VM Migration Algorithm.*

```
1: Each player $i$ computes its anticipation to each commodity $k$ in the market, i.e., $A_i^k$ given by (14). Then, it derives the relative distance $d_i^k$ using (15), and the corresponding utility-distance product $\phi_i^k$, given by $\phi_i^k = d_i^k \cdot A_i^k$.
2: Each player $i$ computes the cumulative utility-distance product $p_i^k$ for all commodities, according to their relative distances $d_i^k$.
3: Each player $i$ determines his pivot point $\mu_i$ according to (16).
4: Each player $i$ finds a subset of commodities $\mathcal{C}_i$, whose cumulative utility-distance products are not larger than $\mu_i$, i.e., $\mathcal{C}_i = \{k : p_i^k \le \mu_i, \forall k \in \mathcal{M}\}$.
5: if Commodity $k$ belongs to more than one $\mathcal{C}_i, \forall i \in \mathcal{N}$ then
6:     The player $i'$ ($k \in \mathcal{C}_{i'}$) who has the smallest $d_i^k$ to commodity $k$ will obtain this commodity.
7: else
8:     The player $i'$ ($k \in \mathcal{C}_{i'}$) will obtain this commodity.
9: end if
10: All players migrate their commodities according to the obtained assignment results.
```

IV. Experimental Evaluation

In this section, we investigate how the proposed VM migration algorithm performs in practical scenarios. Our real-world trace-driven experimental results validate that our VM migration algorithm increases resource utilization ratios with fluctuating requests in video streaming datacenters effectively, yet in a lightweight fashion.

Our evaluation of the proposed VM migration algorithm is based on a C++ implementation in an event-driven simulator, which is driven by real-world streaming requests captured by our traces. We are using 200 Gigabytes worth of operational traces, which we have collected throughout the 17-day Summer Olympic Games in August 2008, with UUSee as one of the official online broadcasting partner in China. Both VoD and live streaming videos were involved, with each of them represented by a VM in our simulation. More detailed information of our trace used in this simulation is summarized in Fig. 8.

As we can see from the table, videos in the trace vary in terms of their bitrates, which results in different bandwidth resource demands in each request. In our simulation, we assume each user asks for 10 seconds of video with every request, *i.e.*, the bandwidth resource required per request ranges from 264 KB to 879 KB. Besides, videos also differ from whether network coding is applied during transmission, which results in deviated CPU cycles required in each request. For videos without using network coding, the required CPU cycles

Fig. 8. Detailed Information About the Trace

Number of videos (live/VoD)	1625(216/1409)
Number of videos with network coding	1472
Peak number of concurrent requests (for all videos)	24757
Highest video bitrate	879 kbps
Lowest video bitrate	264 kbps

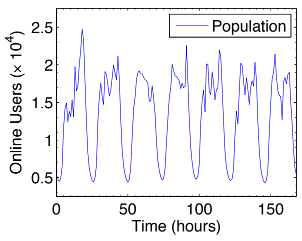


Fig. 9. The demand pattern in 200 hours.

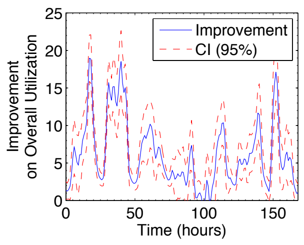


Fig. 10. Average of improvement on resource utilization ratios with the VM migration algorithm.

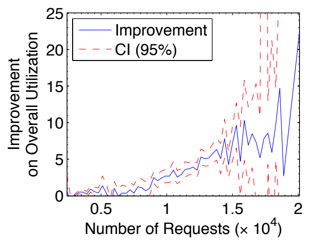


Fig. 11. Average of improvement on resource utilization ratios vs. the number of requests.

per request is assumed to follow a normal distribution of $N(2, 0.25)$ MIPS; for those with network coding, the required CPU cycles per request is assumed to be represented by $N(2, 0.25)$ + bitrate/100 × $N(1, 0.25)$, since it is measured to be proportional to the normalized bitrate of each video. We simulate a system with 25 servers, each of which is assumed to have the same amount of resources: 1000 GB storage space, 1000 Mbps bandwidth and 1000 MIPS CPU cycles.

Our objective is to increase resource utilization ratios in video streaming datacenters, so that they can satisfy as many requests as possible with their available resources. We compare the bargaining-based VM migration algorithm with the naive VM migration algorithm: only the VM that causes overload is migrated; and its destination server is greedily selected from all under-utilized servers, *i.e.*, the one with the most available resources. Main performance metrics in this simulation are, therefore, the improvement on resource utilization ratios and the number of requests the datacenter can handle successfully. In addition, we also show the bargaining overhead from an implementation point of view. We run our simulation 20 times, each lasting 100 time intervals.

We first present the improvement on resource utilization ratios by using the bargaining-based VM migration algorithm, as a percentage of the ratios using the naive algorithm. Fig. 9 shows the demand variation in 200 hours. Fig. 10 shows improvements on average resource utilization ratios at all servers over time and their 95% confidence intervals under the demand pattern shown in Fig. 9. As we can observe, the average improvement on resource utilization ratios varies with the same trend as demand patterns, with a maximum improvement of almost 20% than that of the naive VM migration algorithm.

The reason that the improvement on resource utilization is less evident and the number of requests is low is that most servers are under-utilized in this scenario. The optimal solution that maximizes the overall resource utilization by the bargaining-based VM migration algorithm is more likely to conform with the greedy results. However, when the number of requests increases, improvements on resource utilization with the bargaining-based algorithm become more evident. Fig. 11 shows the average of improvements on resource utilization ratios vs. the number of requests. It is clear that the improvement is significant when the number of requests becomes large.

In addition to resource utilization ratios, another important performance metric is the number of requests that the dat-

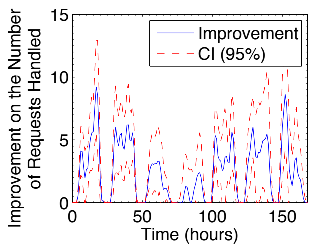


Fig. 12. Improvement on the number of requests successfully handled by the datacenter.

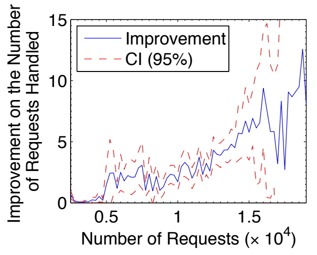


Fig. 13. Improvement on the number of request successfully handled vs. the total number of requests.

acenter is able to handle, given current available resources. As shown in Fig. 12, it is clear that the number of requests the datacenter can respond to has increased accordingly by applying the bargaining-based VM migration algorithm, with an improvement of up to 9% compared with the naive algorithm. With more than 20000 concurrent requests at the peak time, this implies an increase of more than 1800 requests being handled by the datacenter. This result confirms what we observed in the improvement on resource utilization ratios, since the more efficient available resources are being utilized, the more requests the datacenter can handle. Fig. 13 shows the improvement on the number of requests successfully handled vs. the total number of requests. We can observe a similar trend as Fig. 11, that the improvement becomes more evident as the number of requests increases.

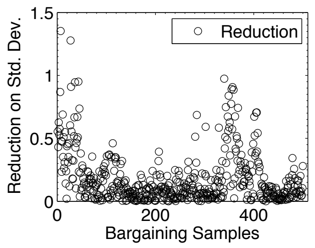


Fig. 14. Reduction in standard deviations of resource utilization ratios.

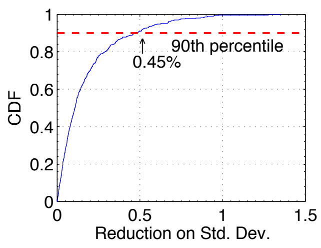


Fig. 15. CDF of reductions in standard deviations as a percentage.

To further investigate the effectiveness of our algorithm, in Fig. 14, we show reductions in the standard deviation of resource utilization ratios at each server, as a percentage across 500 bargaining samples. A reduced standard deviation of resource utilization ratios reflects a more balanced resource utilization. Fig. 15 shows the CDF of reductions on standard deviations. We can observe that the standard deviation of resource utilization ratios is successfully reduced, the 90th percentile of the ratio of such reduction is 0.45%, which shows that our VM migration algorithm is able to mitigate resource under-utilization due to imbalanced resource usage across different dimensions.

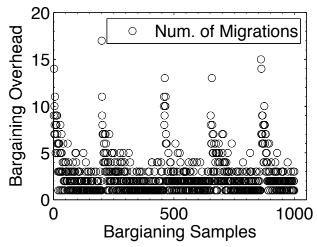


Fig. 16. The number of VM migrations after bargaining.

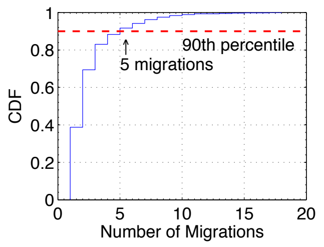


Fig. 17. CDF of the number of VM migrations after bargaining.

Finally, it is important to evaluate the VM migration overhead incurred with our bargaining-based solution. We plot the number of trades conducted after running the bargaining algorithm to balance the resource utilization across all servers in 1000 bargaining samples in Fig. 16, and the CDF of all samples in Fig. 17. As we can see, in 90% of the cases, the algorithm incurs fewer than 5 trades. It reveals that the VM migration overhead is reasonably low.

V. Related Work

Researchers are starting to find possibilities of moving video streaming services to virtualized clouds. For example, Aggarwal *et al.* investigate the potential of utilizing virtualization to deliver IPTV services, while their focus is on how the number of servers required is efficiently minimized, with only the deadline constraint considered [3]. Prior to direct video streaming, video transcoding is commercially available on cloud environments. For example, HDCloud and Encoding have provided flexible but proprietary cloud based video transcoding services integrated with Amazon EC2, S3, and CloudFront CDN services [6].

There exist research results show how VMs are to be migrated to alleviate "hotspots" in datacenters. Wang *et al.* proposed an autonomic provisioning framework, so that resources can be dynamically provisioned to different applications according to their demand on CPU capabilities [15]. Meng *et al.* presented a placement algorithm that focuses on bandwidth consumption of each VM [10]. Unlike prior works that only considers a single resource dimension, this paper addresses challenges of maximizing resource utilization across multiple resource dimensions, which is much more challenging.

To our knowledge, there exist two papers that discussed resource challenges along multiple dimensions in datacenters. Korupolu *et al.* propose a placement algorithm by applying stable matching, taking storage, bandwidth, and CPU resources into account [9]. Our work differs in that the problem they address is a static placement problem, in a sense that once VMs are placed, they are not migrated over time. Singh *et al.* [14] describes VectorDot, a load balancing algorithm for handling multi-dimensional resource constraints in datacenters. However, their objective is to alleviate an overloaded server as much as possible, while ours is to improve the utilization ratios across the board, as required resources increase in an unbalanced fashion and may negatively affect the utilization of available resources in the datacenter.

VI. Concluding Remarks

Our focus in this paper is to fully utilize resources in dimensions of storage, bandwidth and CPU computing in video streaming datacenters, by migrating VMs live among servers when they are overloaded. We argue that such migration should be conducted with careful planning, in order to fully explore possibilities of utilizing current available resources, in terms of storage, bandwidth and CPU. From time to time, conducted in a laissez-faire manner, we believe that VM migration is helpful to improve overall resource utilization, and ultimately to deliver better performance as more video requests are being handled. As a practical way to govern these VM migration decisions, we have designed an algorithm based on the Nash bargaining solution. With event-driven simulations based on real-world video streaming traces from UUSee Inc., we show that the bargaining algorithm is able to improve resource utilization over time, with a small amount of VM migration overhead.

References

[1] *Amazon EC2 Instance Types*. http://aws.amazon.com/ec2/instance-types/.
[2] *UUSee*. http://www.uusee.com.
[3] V. Aggarwal, X. Chen, V. Gopalakrishnan, R. Jana, K. Ramakrishnan, and V. Vaishampayan. Exploiting Virtualization for Delivering Cloud-based IPTV Services. In *Proc. IEEE INFOCOM Workshop on Cloud Computing*, 2011.
[4] R. Baron, J. Durieu, H. Haller, and P. Solal. Finding A Nash Equilibrium in Spatial Games Is An NP-Complete Problem. *Economic Theory*, 23(2):445–454, 2004.
[5] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.
[6] A. Garcia, H. Kalva, and B. Furht. A Study of Transcoding on Cloud Environments for Video Content Delivery. In *Proc. ACM Multimedia Workshop on Mobile Cloud Media Computing (MCMC)*, pages 13–18, 2010.
[7] A. Greenberg, J. Hamilton, D. A. Maltz, and P. Patel. The Cost of a Cloud: Research Problems in Data Center Networks. *SIGCOMM Comput. Commun. Rev.*, 39(1):68–73, 2009.
[8] J. K. Karlof, editor. *Integer Programming: Theory and Practice*. CRC Press, 2005.
[9] M. Korupolu, A. Singh, and B. Bamba. Coupled Placement in Modern Data Centers. In *Proc. IEEE International Symposium on Parallel & Distributed Processing (IPDPS)*, pages 1–12, 2009.
[10] X. Meng, V. Pappas, and L. Zhang. Improving the Scalability of Data Center Networks with Traffic-aware Virtual Machine Placement. In *Proc. IEEE INFOCOM*, 2010.
[11] J. Nash. The Bargaining Problem. *Econometrica*, 18(2):155–162, 1950.
[12] M. Nelson, B. H. Lim, and G. Hutchins. Fast Transparent Migration for Virtual Machines. In *Proc. USENIX ATC*, 2005.
[13] G. Owen. *Game Theory*. Academic Press Inc, 1995.
[14] A. Singh, M. Korupolu, and D. Mohapatra. Server-Storage Virtualization: Integration and Load Balancing in Data Centers. In *Proc. of ACM/IEEE Conf. on Supercomputing (SC)*, pages 1–12, 2008.
[15] X. Wang, D. Lan, G. Wang, X. Fang, M. Ye, Y. Chen, and Q. Wang. Appliance-Based Autonomic Provisioning Framework for Virtualized Outsourcing Data Center. In *Proc. 4th International Conf. on Autonomic Computing (ICAC)*, page 29, 2007.
[16] K. K. L. Wong. A Geometrical Perspective for the Bargaining Problem. *PLoS ONE*, 5(4):e10331, 2010.