An Architecture to Stimulate Behavioral Development of Academic Cloud Users☆

Gabor Kecskemeti[a,b,*], Simon Ostermann[a], Radu Prodan[a]

[a] *Distributed and Parallel Systems group of the Institute of Computer Science at the University of Innsbruck, Technikerstraße 21a, Innsbruck 6020, Austria*
[b] *Laboratory of Parallel and Distributed Systems of the Institute of Computer Science and Control of the Hungarian Academy of Sciences, Kende u. 13-17, Budapest 1111, Hungary*

Abstract

Academic cloud infrastructures are constructed and maintained so they minimally constrain their users. Since they are free and do not limit usage patterns, academics developed such behavior that jeopardizes fair and flexible resource provisioning. For efficiency, related work either explicitly limits user access to resources, or introduce automatic rationing techniques. Surprisingly, the root cause (i.e., the user behavior) is disregarded by these approaches. This article compares academic cloud user behavior to its commercial equivalent. We deduce, that academics should behave like commercial cloud users to relieve resource provisioning. To encourage commercial like behavior, we propose an architectural extension to existing academic infrastructure clouds. First, every user's energy consumption and efficiency is monitored. Then, energy efficiency based leader boards are used to ignite competition between academics and reveal their worst practices. Leader boards are not sufficient to completely change user behavior. Thus, we introduce engaging options that encourage academics to delay resource requests and prefer resources more suitable for the infrastructure's internal provisioning. Finally, we evaluate our extensions via a simulation using real life academic resource request traces. We show a potential resource utilization reduction (by the factor of at most 2.6) while maintaining the unlimited nature of academic clouds.

Keywords: Cloud Computing, Pricing, Infrastructure as a Service, Energy Awareness, Academic Clouds

1. Introduction

Academic computing infrastructures are built and maintained in order to support scientific users in their research endeavors. Introducing limitations on the hardware usage in any ways would defeat the very reason for the existence of these infrastructures. However, the more limitless a system is the more responsibility it requires from the scientific users. For example, they must learn to eliminate their impact on other user's workings. Therefore, maintainers of such systems traditionally make the compromise of introducing such limitations for the users that stop unintentional obstructions on the work of other users [1]. Meanwhile, for future systems, computer science tries to reduce the amount of limitations and their impact on the scientific users.

Infrastructure as a service (IaaS) cloud computing systems [2] are amongst the most recent developments in this field. These systems offer on demand resource access with such flexibility in software configurations [3] that the users could even utilize highly customized operating systems and support environments for their tasks. This flexibility is achieved through the application of virtualized data centers. Although, the cloud computing concept has been proposed by commercial companies (e.g., Amazon[1], Rackspace[2]), academic solutions (like Eucalyptus [4], Nimbus [5] or OpenNebula [6]) started to arise first by imitating the behavior of the commercial solutions then by advancing towards specific academic needs.

Pricing is one of the essential aspects of commercial IaaS systems [7] that academic solutions did not copy. Thus academic providers who apply such academic solutions will appear as offering unlimited resources for free to academic users. This promise is tempting for the users as it lifts one of their last remaining limitations. Unfortunately, this setting leads to an unprecedented demand of resources that is often latent (e.g., users maintaining demand for resources similarly to pilot jobs in grids [8]).

Academic providers have to fulfill these demands with the limited physical resources they are operating on. To meet the demands with the infrastructure's real capabilities they usually apply two solutions: (*i*) access rationing, (*ii*) under provisioning (N to 1 mapping of virtual to physical resources). Both approaches were utilized in academic infrastructures even before the cloud era, but they both

☆The research leading to these results has received funding from the Austrian Science Fund projects TRP 237-N23 and ICT COST Action IC1305.

*Corresponding author. Tel: +36 1 279 6065
Email addresses: gabor@dps.uibk.ac.at (Gabor Kecskemeti), simon@dps.uibk.ac.at (Simon Ostermann), radu@dps.uibk.ac.at (Radu Prodan)

[1] http://aws.amazon.com/ec2
[2] http://www.rackspace.com/

have serious downsides for academic uses. First, access rationing directly intrudes the freedom of researchers access to the infrastructure [9]. For example, when a credit system is applied for rationing, then the research of users with no credits could be postponed for indefinite time periods (i.e., until they acquire some new credits for computation). Second, providers with under provisioning policies promise resources that are heavily shared amongst users [10], therefore these shared resources could vary in performance over time (the unintentional effects of others who introduce background load to the shared resource).

Instead of the previously applied solutions, we propose to direct users towards self-rationing. We derived the rationing problem from the missing pricing in academic clouds and argue that it is possible to construct academic systems that feature similar behavior to commercial clouds (where the rationing is imposed by the cost of further leasing resources) but still promises unlimited resources and unprecedented software configurability. We achieve this behavior with an architecture that exposes energy efficiency metrics to the users. First, our architecture provides the foundations for various leader boards where academic users can compete with each other on how energy efficient they use the acquired computing resources. Second, to ignite the rivalry on the leader boards, we recommend providers to allow the specification of energy related constraints on resource requests. Finally, we introduce the concept of engaging options (electronic representations of underused capacities) that allow users to attract others for particular resources. These options offer a chance to users to utilize resources from hosts that are already used and thus these hosts could operate more energy efficiently.

The proposed architecture is built on three fundamental assumptions: (*i*) availability of energy readings, (*ii*) application of energy aware virtual machine placement, and (*iii*) leader board publicity. First, we require the energy readings because we propose to publicize the accountable user consumption either directly or on a transformed way through leader boards. Second, users should be able to influence their leader board position, thus particular resource requests should have deterministic energy behavior. This behavior should be guaranteed by an energy efficient virtual machine placement policy at the provider. Finally, the expected effects of the leader boards and engaging options are really dependent on their publicity, thus it is expected that they are soon adopted by a significant percentage of the academic community. For example, the adoption could be forced by the providers by automatically enlisting their users in local leader boards.

To test the feasibility of our architecture and its positive effects on the academic cloud communities, we have analyzed the behavior of typical academic users. First we have classified the users by behavior and identified the ways users could be transformed to behave more self-constraining while still performing their tasks. Second, we simulated the possible behavior of the academic users found in the Grid Workload Archives. Based on our simulations, we have concluded that there is a high chance of increasing energy efficiency and reducing resource demand on the provider side while still performing all user tasks. Our findings show, that the effect of our architecture could decrease the energy footprint of the provider's computing infrastructure by a factor of 2.6 at most. We have also revealed that a few users could particularly reduce the demands of the infrastructure, thus we introduced the concept of the hall of shame (for the list of least efficient users). This list should be presented alongside the leader boards in order to put immediate tension on the misbehaving users by their fellows.

The rest of the paper is structured as follows. First, we provide an overview on related research topics and the research issues in Section 2. Next, in Section 3, we reveal our architecture that could answer the identified research issues. Afterwards, in Sections 4 and 5, we discuss the inner workings of the architecture, first starting with the characteristics of leader boards, their contents and their relations with the users. Then, we continue with engaging options and we show their life cycle from the time they are issued to the time they are used or become invalid. Later, in Section 6, we analyze the effects of the architecture in a simulated environment. Finally, Section 7 concludes our findings and summarizes the architecture's properties.

2. Related work

Cloud computing is interesting for the scientific community from the beginning of the transformation of Amazon Web Services towards the Amazon Elastic Compute Cloud. Early evaluations investigated how scientists can benefit from this new technological infrastructure compared to Grids [11], especially with respect to storage [12] and computation [13] costs.

Maximizing the revenue from infrastructure operation is an important objective of Cloud providers, analyzed in [14] based on SLA relationships. This work does not take into account the possible energy savings that may further help in reducing the costs.

In [15], it is shown that saving power can increase the revenue of Cloud providers with only slight impact in the overall performance. Academic clouds are not adapting such power optimizations promptly, because the persons responsible for the resource usage are not responsible for the incurred electricity costs.

Rigid allocation mechanisms such as accounting based limitations can restrict many cloud and grid computing use cases and [9] reduce the overall scientific productiveness. Contrary to credit or accounting based limitations, our approach motivates the users to optimize their resource usage through reduced power consumption.

Existing resource management and scheduling systems such as SLURM [16] and Maui [17] incorporate fairshare mechanisms enforcing user quotas or data access within ownership domains which constrain scientists in using such

systems. These approaches however do not want to educate their users on expected user behavior. Therefore, even if academic clouds apply these resource management systems, users could still behave on a way that these schedulers could not resolve with their fairshare mechanisms. For example, incoming virtual machine and job requests could be so unnecessary large and long running that these algorithms cannot manage them energy efficiently on their own without external assistance (e.g., someone/something suggesting users for more energy efficient request timing and size that could still fulfill user tasks).

The work in [18] demonstrates that user awareness of power consumption results in possible savings. In this paper, we also aim at increasing user awareness but propose to do so by using leader boards. The leader board is a heavily utilized concept in computing to increase user involvement and awareness regarding particular topics (through user ranking and competition). Furthermore, several leader boards not only offer rankings for individuals, but also groups. Group rankings within a leader board system can be used to start additional competition between countries, research institutes or other self-forming user groups (e.g., multinational research groups). Leader boards became widely known through projects like SETI@home [19] or the TOP500 supercomputer ranking [20]. Similarly to these projects we also use the leader board concept, where we introduce a new ranking based on energy consumption and efficiency.

[21] evaluated and quantified the impact of user feedback on power savings as 5% - 15%. To allow feedback on the power consumption, there is a need for a model that maps the physical power consumption to the virtual machines. Such models are provided in [22, 23] and show sufficient accuracy to be employed in our proposed architecture. Measuring the individual power consumption of VMs executed on a physical machine results in a higher power consumption per VM if the machine is under utilized and the static idle power is shared [24, 25]. We apply this knowledge in our approach so we can present academics valuable information on how they can improve their energy efficiency measures.

There is also an interest in saving power in Cloud infrastructures by turning off unused virtual machines to optimize resource use [26, 27]. This case is relevant for commercial providers, as users are not willing to pay for the idle times of their virtual machines. Optimized VM placement can save up to 55% of energy using the approach in [28]. Combining the savings possible through awareness of the energy consumption and optimized VM placement can significantly reduce the power consumption and further increase the productivity. Unfortunately, these techniques are only achieving these significant savings if rational users are utilizing the infrastructure, which is not the case with current academic users. Thus in order to fully utilize the effects of these techniques this paper aims at transforming academic users to behave more rational (i.e. like the price constrained commercial cloud users).

Engaging options proposed in Section 5 show similar characteristics as referral programs studied in [29]. By helping the Cloud provider to better utilize its virtualized hardware, the energy consumption accounted to the users accounts can be reduced. Research shows that such programs can achieve up to 16% better results [29]. Referrals are often more personal than normal advertisements or SLAs and, therefore, show a high impact on user behavior as showed in [30]. Similar impact is possible with our approach because engaging options from coworkers are more trustworthy than SLAs from Cloud providers.

3. The architecture

3.1. Behavioral differences amongst cloud users

In response to the unprecedented latent demand from users, state of the art research focuses on changing the operation of academic cloud providers so they no longer appear to provide unlimited resources to their users. Unfortunately, this approach does not provide the cure for the root cause: the misbehavior of the users. This misbehavior is caused by the provider's promise about unlimited resources, and can be characterized as follows:

- Unnaturally and unnecessarily long infrastructure leases: E.g., academics frequently maintain their virtual machines even if they don't use their resources just to avoid the often long times they have to wait for the resource and its preparation for their particular need.

- Academic users also tend to prefer resources with the highest performance without giving too much consideration on other properties of the acquired resources (e.g., availability, energy efficiency, effect on other users).

In contrast, users of commercial cloud systems behave more rationally, since the prices imposed by commercial clouds ensure that users will not maintain economically unsustainable virtual infrastructures. These are the relevant characteristics of commercial users:

- They *delay the instantiation* of their virtual machines. For example, until these machines are an absolute necessity for the further progression of user tasks.

- They try to *ensure continuous use* of the acquired VMs (doing as much work as possible during the time the VM is available).

- They *terminate VMs early* on (considering the billing periods – e.g., on Amazon there is no use to terminate a VM before one hour). Thus they immediately terminate a VM that has no further tasks or it is not expected to have a task for it in the foreseeable future.

Table 1: The effects of the various pricing models in commercial clouds

Note: this table is discussed throughout Sections 3–5, thus some of its definitions are put into context later on.

User characteristics	Pricing models offered by commercial providers		
	Standard	Reserved	Spot
Delay instantiation	Need triggered	Limit tasks to reserved VMs	Price & need triggered
Ensure continuous use	Use the VM regularly	Prolonged use is desired	Burst VM use until abortion
Early VM termination	Terminate if not needed	If unjustified, sell reservation and switch model	Terminate if not needed
Performance compromise	Instantiate a smaller VM	Bound to a resource type	Abrupt VM abortion

- They make a *compromise between price and performance* and allow increased task makespans. For example, a smaller priced instance could still be capable to perform the necessary tasks within the billing period of the provider, thus if the tasks are not time critical, they could take a little longer. Or they would postpone the execution of a new task until an existing VM could run them (assuming that serializing the tasks is more cost effective than having a new VM for the new task).

Commercial providers (like Amazon) establish such user characteristics through the following three pricing models: (*i*) *Standard* where the resource usage is paid with a constant hourly price, (*ii*) *Reserved* where a upfront payment reduces the hourly prices and (*iii*) *Spot* where the compromise of a possible VM abortion potentially lowers the prices. In Table 1, we show how the aforementioned user characteristics are achieved by these three pricing models. The table also reveals the rational user actions one can observe when a particular pricing model is applied. Fortunately, all these user actions are possible in current academic clouds (i.e., these actions can be accomplished through the usually available IaaS interfaces). Therefore, if academics would have the incentive to take these actions then they would bear similar characteristics as commercial users. And commercial like user characteristics would allow academics clouds to maintain sufficient balance between their users and resources.

3.2. IaaS extensions to support behavioral change

To encourage such user behavior, we propose to motivate the academics through presenting them the energy impact of their operations. We propose such architectural extensions to academic cloud environments that not only collect and present the energy consumption data to academic users but also provide them information on how to increase their efficiency. Figure 1 presents these extensions to an existing IaaS software stack (shown in the bottom right corner) and shows their relations to academic users. As the extensions are aimed at academics, other actors like system administrators are not shown. In fact, these extensions do not even change the ways the users would interact with the existing cloud environment, they just add an optional functionality that – if used widely – could lead users towards self-rationing. So the users still utilize their usual command line interfaces or web portals, but by having access to our extensions they are expected to change their behavior towards these interfaces. These extensions are built on two cornerstones: (*i*) *leader boards* (for disseminating and comparing how energy efficient are the users) and (*ii*) *engaging options* (to ensure that VM requests are arriving in groups more suitable for those physical machines that will host the VMs). The following paragraphs discuss how these cornerstones are related to user behavior and show their basic properties. Later these two are further elaborated in the following two sections.

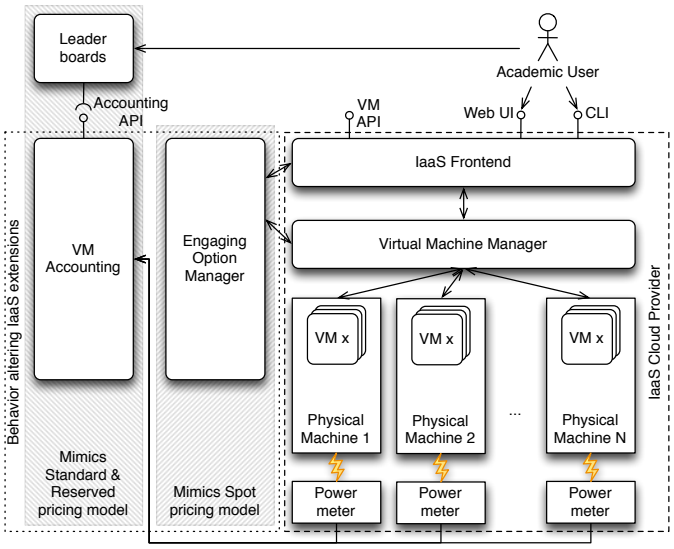


Figure 1: Overall view of the proposed architecture

First, similarly to the account statements of commercial providers, with our extensions, academic cloud providers collect the energy consumption accounted to particular actions of their users. Our architecture ensures that they publish this data through an *Accounting API*. Through this API, trusted third parties are allowed to query the energy consumption accounted to particular virtual machines (e.g., a user can query the consumption of its own VMs). Leader boards are special users of this API who collect data regarding every registered user. They aggregate energy consumption data for users. Depending on the

intended user base, each leader board can use their own method for aggregation. At the end, to each user they assign a score that is comparable with other scores in the same leader board. Every board presents a ranking list for its users (the more energy efficient a user is the higher his/her ranking is). This list is our major motivational instrument as users often compare themselves to their peers and try to improve their ranking.

Next to the list, leader boards present users with techniques that could increase their score. Recommended techniques reveal how to accomplish similar behavior to the users of commercial clouds. Table 1 also acts as a summary for the recommended techniques. So, for example, academics are expected to terminate their unused virtual machines. The details of the resulting behavior and scoring are discussed in Section 4. Unfortunately, there are some user behavior (e.g., price and need triggered VM instantiation or the use of dedicated VMs for repetitive and prolonged tasks) that the accounting API and the leader boards cannot impose on academics.

To guide users towards these behavioral patterns we also introduce the concept of engaging options. These options are such electronic documents that represent inefficiencies in the system. If one user is assigned to a physical machine that's power efficiency could be increased then he/she receives some engaging options. The received options then can be exchanged with other users. Amongst the techniques listed on the leader boards, users will be noted that if they would share/wait for such options, then they would have a chance to increase their scores. Thus users will act similarly as those in commercial clouds: they will wait for a suitable engaging option to appear before instantiating their VMs (just like commercial users would behave for spot pricing). Thus, our architecture extends current IaaS systems with the management of the entire life-cycle of engaging options (from their issuing to their dissolution). Figure 1 refers to this extension as the *Engaging Option Manager*. We discuss its properties and behavior in Section 5.

4. Towards the behavior of static pricing models

Leader boards are known to have an attractive influence on most scientists, as can be seen from TOP500 supercomputing list [20] or the various volunteer computing solutions like [19]. Especially when ranked in groups [31], ranking can have high influence on all members of the group, thus they will all try to achieve higher scores. Therefore, we propose to set up leader boards where users can compete by using cloud resources in a energy efficient way. We define leader boards as such entities that are independent from cloud providers but still able to present the accounting data (as scores) to their users so they start to challenge each other. In this section, we first analyze the relevant characteristics of leader boards. Then we show a way to motivate academic users through leader boards so they start to resemble their commercial counterparts.

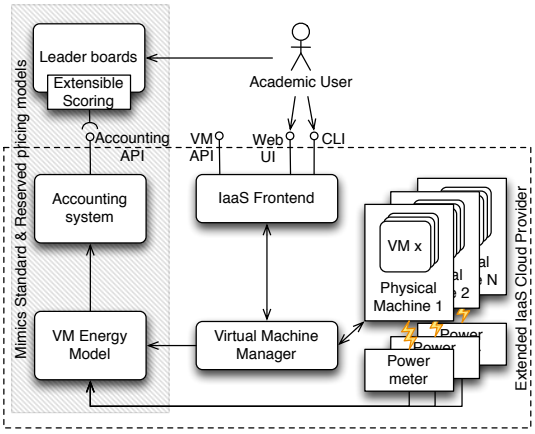


Figure 2: Leader boards and our accounting extensions

4.1. An extensible scoring system

The scoring system and its presentation on the leader board represents a core asset. With improper solutions the academics will not be motivated to submit to the behavioral patterns of commercial users. The major success of similar leader boards is based on the ability that users can heavily influence their scores and thus can ignite competition amongst each other. The cornerstone of the score based motivation is that the leader board should present the way it transforms the accounting data to the actual scores. Since users are not expected to know the internal workings of the cloud systems, the leader board should also present the behavioral patterns that positively affect user scores. Users also receive a breakdown on their scores so they can know how their particular activities affect their score. With all this information the users will have a better chance to influence their scores and they will perform better in their competitions.

Motivation can be further increased with the formation of user groups within a leader board [31]. In general, groups allow group members to compete with each other. If groups also receive an overall score based on their member's scores, then group members are encouraged to pursue higher scores together by enabling the competition of groups based on overall group scores. To reach higher overall scores, enthusiastic users will try to convince more resistant users to revise their resource usage patterns. Since group interaction is essential to strengthen the overall motivation in the system, our leader boards automatically form groups based on user affiliation and interest (e.g., groups are created for departments or computer scientists in general). These groups help building up the initial momentum towards widespread user behavioral changes, but users are not restricted to them. New groups can be formed by enthusiasts also.

Since the scoring is such an important motivational factor for both individuals and groups, our IaaS extensions provide a foundation for scoring systems. As seen in Figure 2, this foundation is built around two vital elements:

(*i*) an accounting API, (*ii*) an extensible scoring system. Our accounting API is responsible to offer details (like energy, storage usage) about current and past virtual machines in the academic cloud. While, the extensible scoring system offers ways to transform and aggregate the data from accounting to stable and motivating scores presentable through leader boards.

4.1.1. Accounting

The accounting API is the major information source for the scoring system. Behind the API an accounting database is set up on which the API offers a simplified view. This database contains static and dynamic data from various sources. The static data entries represent the general properties of the infrastructure that are not offered by other information systems (like the energy characteristics of the available physical machines in the infrastructure). While the dynamic entries represent virtual machine related data recordings (e.g., energy consumed, storage used, network utilization, physical machine allocation). Most of the data aggregated in the accounting database is already available or could be collected with some means from the IaaS provider. For example, when the IaaS runs its virtual machine placement algorithm, it can notify interested parties about the VM placement decisions. Alternatively, the virtual machine mappings can be determined by agents deployed on the physical machines of the IaaS. Although these agents could report not only the placement information but other VM related information, their impact on energy metering results is undesirable. Thus our extension offers hooks to placement algorithms and only falls back to the agents when these hooks are unusable by the original IaaS system. Unfortunately, the energy consumption details are unlikely to be available on the level of the virtual machines. Thus this subsection is focusing on how these details are collected and calculated.

In order to collect the necessary data for VM energy accounting, we first shortly review the life cycle of a virtual machine. There are four major phases in the VM's life: (*i*) prelude (from the request for a VM until the user can actually do some processing with the VM), (*ii*) runtime (while the VM is capable to do the tasks of the user), (*iii*) migrating (while the VM is down because it is moving between physical machines) and (*iv*) post runtime (from the user termination request, until there are no further tasks performed by the IaaS regarding the VM). All phases, except runtime, represent operations done by the IaaS because of the VM. To determine the energy consumed because of the existence of the VM, one should account for all the IaaS components that are involved in the VM's life cycle (e.g., IaaS frontend, VM Manager, or virtual appliance storage subsystem). Thus overall VM energy consumption could be described as follows:

$$E(VM) = E_{PL}(VM) + E_R(VM) + E_M(VM) + E_{PR}(VM) \quad (1)$$

Where $E(VM)$ defines the total energy consumed by the VM, which is a composition of the energy consumed during the lifetime of the VM. Thus, $E_{PL}(VM)$, $E_R(VM)$, $E_M(VM)$ and $E_{PR}(VM)$ represents the VM's consumption during the prelude, runtime, migration and post runtime phases respectively. This is the function that our *VM Energy Model* component in Figure 2 estimates. The rest of this subsection is focusing on the behavior of this component and how it estimates the energy consumption of a particular VM.

There has been significant research on the runtime behavior of VMs and their energy characteristics [23, 25]. So, these energy models can be utilized to determine $E_R(VM)$. Unfortunately, these works do not consider the energy consumption of the other phases in the VM's life cycle. The reason they are not considered is because the other energy consumption values are often not comparable to the runtime consumption of the VM (e.g., if the VM runs for several days, then the few minute long VM instantiation and termination does not increase $E(VM)$ significantly). On the other hand, in this article, we are trying to ensure that users run their VMs for as short time as possible. This behavior results in a smaller gap between the runtime consumption and the other consumption values. Therefore, we argue that these values should also be represented in the accounting.

Per VM consumption of the IaaS. Since $E_{PL}(VM)$ and $E_{PR}(VM)$ are mostly dependent on the IaaS behavior, we propose to check the consumption of those machines that are not hosting virtual machines. These machines are there to support the instantiation, the termination and other VM management tasks. Although it is not an easy task to separate consumption dedicated to virtual machines, we recommend to share the energy consumption of these machines amongst the VMs that they handled in the period when the consumption was recorded. If there are no significant differences between the user VM requests (e.g., no significant size difference on their appliances, or no unusual SLA requirements present that need extra operations on the IaaS side), then we can assume this share is equal. If there are significant differences then, we recommend to first take into account the appliance size as the base of the share:

$$E_{PL}(VM) + E_{PR}(VM) = \frac{E_{IaaS} \cdot size(VM)}{\sum_{vm \in VMS} size(vm)} \quad (2)$$

In this equation, we mark the overall energy consumption of the non hosting machines as E_{IaaS}. VMS defines the set of all VMs that coincide with the VM in question. We refer to the size of the virtual appliance of a virtual machine as $size(VM)$. The more virtual machines an IaaS handles in a given time period, the more constant this part of the energy consumption becomes.

Next, we also need to define the energy consumption of VM migration. For the sake of simplicity, we decompose this consumption in two parts: (*i*) $N_M(VM)$ the number

of migrations that take place for a given VM, (ii) $E_m(VM)$ the estimated consumption of a single migration. We assume the cost of a single migration for a particular VM is constant, and thus we will use the following equation for the overall cost of migration:

$$E_M(VM) = N_M(VM) \cdot E_m(VM) \tag{3}$$

In most cases, when there is no migration during the VM's lifetime, this consumption can be neglected. On the other hand, the longer the runtime of the VM the more likely it will be accounted for migration energy also.

From the individual user point of view, it is impossible to influence the consumption of the IaaS, thus users could see it as a constant consumption that is accounted to them. In case of infrastructure initiated migrations, the user has also no chance to directly reduce the consumption $E_M(VM)$. But the migration related consumption can be mitigated if the *users terminate their VMs* as soon as possible, reducing the chances to be automatically migrated to some other host. This shows us that with this energy metric exposed to the academics, they would be tempted to terminate their VMs when the migration energy cost is still marginal (this behavior is similar to commercial users who time VM terminations according to billing periods). Finally, users only have direct influence on their runtime consumption. Therefore the following paragraphs are focusing on what influences their VM's runtime consumption and how they can change it.

Runtime VM consumption. Finally, we shortly discuss the model of runtime energy consumption. Virtual machine consumption is derived from the physical machine's consumption based on two components: (i) $E_{idle}(VM)$ the idle or static consumption, (ii) $E_{use}(VM)$ the usage related consumption:

$$E_R(VM) = E_{idle}(VM) + E_{use}(VM) \tag{4}$$

The first component is the result of the physical machine's base power draw that is consumed even without any particular load on the machine. The second component is dependent on the actual use of the resources of the physical machine. In the followings we discuss how these components are considered in our accounting database.

Albeit, related works (e.g., [23, 25]) vary on how they map the idle consumption of the host to a particular VM, they all agree that this component is heavily dependent on the number and kind of virtual machines that share the physical host. Therefore, the virtual machine placement heavily impacts how the idle component of the VM is evaluated: by opting for multi-tenancy or by choosing machines with smaller idle consumption academics could reduce the idle component of their VMs. In the first case, the users willingly share resources and thus reduce the minimum size of the infrastructure that could serve them (directly increases flexibility). In the second case, academics could put economical considerations before their performance requirements, thus making a performance compromise similar to commercial cloud users (see Table 1 about user applied bounds to specific resources or smaller priced resources because of pricing advantages).

To support the usage related consumption calculations, we collect the instantaneous power draw of the physical hosts alongside with the usage patterns of the various VMs on the host. We collect the basic and widely available information on when a particular VM arrives to the host and when it leaves from it (e.g., via termination or migration). We collect the VM placement information via either already existing IaaS interfaces, or if they are not available we offer interface extensions on the applied VM Manager. Unfortunately, some IaaS toolkits do not provide detailed VM level usage information necessary to evaluate existing runtime energy models. Therefore, we inject utilization collector agents into the physical machines that could host the VMs. This approach notably raises the idle consumption of the machine (that will be accounted to the VMs indirectly), however it provides the fine grained VM consumption metrics needed. So it is important to select the agents that can provide the necessary input for the runtime energy models with the smallest possible energy impact.

4.1.2. Scoring

Requirements. Scoring is the quintessential part of leader boards. To determine the features and properties of the necessary scoring system for our extensions, we first identified the minimum requirements that this scoring system should adhere: (i) it should highlight the behavioral differences between academics and commercial cloud users, (ii) it should provide higher scores to academics who behave more closely to commercial users (see the expected behavior in Table 1), (iii) scores should be independent from the time the users joined the leader board, (iv) good user behavior should always provide high scores independent from the use of the underlying infrastructure, (v) users should be able to compare their scores to their past selves (whether they improve or worsen compared to the expected behavior), (vi) one's contribution to a group score should not diminish because of diminishing resource use, and (vii) scores should be independent from how many virtual machines for how long the users use.

Next, we investigate the expected scoring behavior in the scope of the user characteristics presented in Table 1:

Need triggered If the VM is started too early, the VM's utilization is minimal before the first tasks arrive. Scoring should punish this with smaller scores. Consequently, users only instantiate their VMs when there are no external dependencies that could defer their computational tasks planned for the VM.

Use the VM regularly If a VM has usage gaps it should have smaller score than a fully utilized one in the same situation.

Terminate if not needed The VM's score will decrease if it is not executing tasks for a considerable amount of time, thus users are inclined to stop their VMs when there are no tasks in the foreseeable future for the VM.

Instantiate a smaller VM The achievable score should be smaller if a not so energy efficient physical machine hosts the VM. Also, the score should be smaller if the user would request a VM with such resources that he/she cannot utilize fully later on. Thus users would limit the resource requests to the bare minimum that is sufficient for their tasks so they can maximize the use of the acquired resources.

Limit tasks to reserved VMs The score should be better for those who almost continuously use efficient resources than those who terminate and request VMs with little time left in between. There are two major user scenarios for this behavior: (*i*) guaranteed high efficiency VMs for longer tasks, (*ii*) reducing the impact of IaaS energy share.

In the first scenario we assume that the user already has a more energy efficient VM than the VMs he/she can have in the close future. In such situation, the user could wait until his/her existing VM finishes its tasks and then assign new tasks for it. This behavior would render a higher score because of the other possible VM's less efficient operation.

In the second scenario, the user has such a short task that it is not beneficial to create a dedicated VM for the task (e.g., the energy accounted for the VM would be dominated by the IaaS's operations – see Equations 1 and 2). Instead it is more beneficial to wait until an already existing VM becomes available.

Prolonged use is desired When the user expects to run tasks for an extensive time period, and these tasks mostly fill this time period (i.e., there are no big gaps in between), then acquiring the most energy efficient VM possible for these tasks should result in better scores than asking for VMs on demand.

If unjustified sell the reservation and switch model When the users badly estimate the amount (or length) of tasks in the previous scenario, they should immediately terminate the acquired VMs and reconsider the VM usage scenario. Otherwise they should end up with smaller scores.

Bound to the resource type If the user acquired one of the most energy efficient VMs possible for prolonged use, then it is beneficial to mostly utilize this VM for its computations, unless there is some urgency. The scores will be higher for the user even if he/she delays some of his/her computations because if those computations would run on other VMs they would receive less points caused by their inferior energy efficiency.

Burst VM use until abortion VMs are not aborted automatically in academic clouds, but otherwise this case can be derived from to the behavior titled "Use VM regularly".

The behavioral patterns "Price & need triggered" and "Abrupt VM abortion" are not possible to reach with scoring only, thus we propose the use of engaging options (detailed in Section 5) to achieve this behavior.

The remainder of this section shows the basic properties of our extensible scoring system (seen in Figure 2). This scoring will already aim at supporting the expected behavioral changes of academics. However, in Section 4.2, we show how various leader boards could already extend this basic functionality for advanced leader board setups.

The score of an individual virtual machine. Based on the requirements above, we aim at first defining a score for an individual virtual machine. Then we will provide a way to aggregate these scores so they allow the comparison of users.

For virtual machine level scoring, we borrow the concept of energy conversion efficiency (η) from physics. This concept shows the relation of the energy invested in to a transformation process and the actual useful energy after the process. In our application, we would like to convert academics to behave more like commercial users. Thus, the energy invested will be represented with the actual energy consumed by a virtual machine ($E(VM)$ – see Equation 1 for details) of an academic user. While the useful energy will be represented by the energy that would have been consumed by a virtual machine ($E^*(VM)$) utilized by a price constrained rational user. In both cases, we assume that the same tasks are executed in both virtual machines. Also, we assume that the price constrained (or commercial like user) would utilize a fully equivalent hypothetical infrastructure (in terms of physical machines, actual running other virtual machines, and other IaaS components). Thus the only difference between the academic infrastructure and the hypothetical one is that the hypothetical would offer functionalities that are needed to accomplish commercial pricing models. This definition renders our base – or virtual machine level – score ($S(VM)$) as the following:

$$S(VM) = \frac{E^*(VM)}{E(VM)} \tag{5}$$

The calculation of the rational user's energy consumption should be set up according to the needs of the particular leader board. When a new leader board is set up, the way this estimate is made should be given by the leader board's owner. Although, we offer a basic definition with the leader boards, this definition cannot take into consideration all circumstances in the particular situation (e.g., it is not known in advance what information is going to be

available through the accounting API as collecting some of the data might be tedious work for the actual provider). Our scoring is extensible as it allows the leader board owners to set up their own base score (and only utilizing the score aggregating facilities of the leader boards). Therefore either the following definition has to be adopted for the use of the new leader board (with an option to choose only the energy model for the VM), or a brand new formula has to be developed. We define the rational user's energy consumption as the minimum consumption possible of a virtual machine with the same tasks on the same physical host (assuming that the physical machine's every other resource is fully utilized).

For every user, his/her own virtual machines' individual scores are presented. This means that a general leader board has an overall view (to compare users with each other), and a user specific one, that reveals the competition between the virtual machines of a particular user. On this user specific leader board (which is only offered to the user who owns the VMs), the best and the worst scored virtual machines are prominently highlighted. To allow the users to learn from their past bad practices, this leader board gives a reasoning for the worst scores, e.g.: (*i*) VM was on a not so efficient machine, (*ii*) the VM was left running on a machine that was not fully utilized by other users – shows the need for multi-tenancy –, (*iii*) the VM was placed on a machine with further resources (i.e., the VM was not occupying all the remaining resources of a physical machine) but other users did not arrive to utilize them early enough (the VM was terminated before other users come).

User level scoring. Although, this virtual machine level score fulfills most of our requirements already, it is not the individual virtual machines that we would like to enter into a competition but their users. Thus there is a need for an overall score. The overall score – $S(U)$ – should keep the properties of our virtual machine level scoring, while it should present the overall conversion of an academic. For this score, we propose to weight virtual machine level scores with the execution time – $t(VM)$ – of the particular virtual machines:

$$S(U) = \frac{\sum_{\forall VM \text{ of U}} S(VM) \cdot t(VM)}{\sum_{\forall VM \text{ of U}} t(VM)} \tag{6}$$

4.2. Infrastructure independent scoring

Our overall score is already offering a great detail for the user, however it suffers from issues with comparability when it would be applied across cloud infrastructures. Therefore, in the following section we detail a second level of scoring (building on top of our extensible scoring seen in Figure 2). This second level builds on our overall score, but improves its comparability by transforming the score to a new one that is more suitable for a particular situation. Below we list the various cases when there is a need to adjust scoring.

4.2.1. Provider leader boards

If an academic cloud provider frequently faces resource provisioning issues (e.g., unexpected jumps in demand, orphaned VMs, VMs with little or no utilization of extensive periods of time), then the adoption of our extensions could offer remedy without investments in newer hardware. In such case leader boards are maintained internally by the cloud provider (one provider offers a single leader board for all its users) as shown in Figure 3a. In this leader board the users are automatically registered. Then, the provider should prominently advertise its existence so users will not miss the fact that they are ranked. This advertisement allows the early adoption of the system and ensures that scoring conscious users build up a critical mass. As a result, users at the provider start to have a chance to compete with each other and become more and more power aware. As an advantage, this not only helps the provider to better serve its user community but also increases user satisfaction (as seemingly more resources are available when more users adopt our self-rationing scheme). This approach however needs a substantial amount of users who are willing to join the competition early. Therefore, these provider level leader boards are unusable when there are only a few users served by the provider.

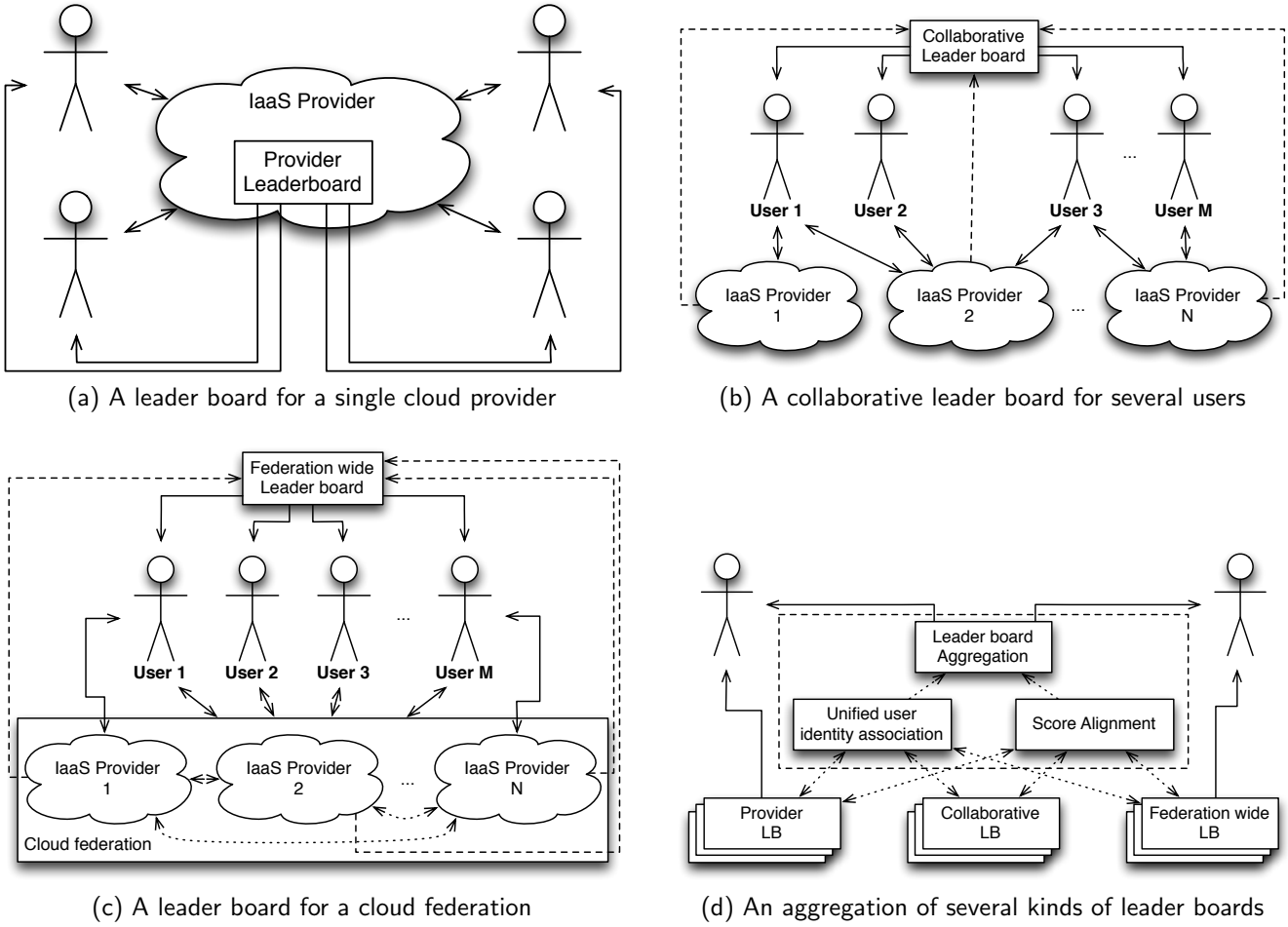
The rest of the leader boards are external to the providers (they have no influence neither on their scoring nor on their maintenance).

4.2.2. Collaborative leader boards

If there are some enthusiasts who would like to compete despite they are served by different cloud providers, then they should set up a collaborative leader board on their own (see Figure 3b). However, the different properties of the various clouds (e.g., different machines with different power consumption profiles) would not allow direct comparison of the participant's scores. Therefore, a collaborative leader board should apply a relative scoring system, e.g.: (*i*) scores could be calculated based on ranks at the different clouds if there are provider leader boards, or (*ii*) users should be ranked based on how close they are to the maximally achievable energy efficiency at their provider. These kind of leader boards even allow the inclusion of providers with few users because the participants on the collaborative leader boards are enthusiasts and known to be more active than regular users would be. Unfortunately, these leader boards are of limited use because of the expertise needed to set up such a leader board.

4.2.3. Federation wide leader boards

In case a cloud federation aims at reducing its overall energy footprint it could operate a federation level leader board (as revealed in Figure 3c). These boards could present the absolute scores similar to provider leader boards (absolute scores will not hide the differences between the energy efficient behavior of the various providers in the federation). As a result, they could influence users



(a) A leader board for a single cloud provider

(b) A collaborative leader board for several users

(c) A leader board for a cloud federation

(d) An aggregation of several kinds of leader boards

Figure 3: Leader board variations

to choose more energy efficient cloud providers participating in the federation. It is important to note that these leader boards lose their importance if there is no chance for the users to move between providers as they prefer. Advantages of the federation wide leader boards: (*i*) allow the individual providers to concentrate on their infrastructure (they don't need to set up a provider leader board for their own users) and (*ii*) a larger set of users are aggregated on the federation level thus there are more chances to increase multi-tenancy on the physical machines. Regrettably, such an installation could result in over usage of the most efficient provider in a federation as it drives users away from the less efficient offerings. On the positive side, this case could highlight the omitted provider the time to update its infrastructure to a more efficient solution. However, the complete resolution of this possible unbalance remains future work.

4.2.4. Leader board aggregations

It is also possible that third parties set up aggregating leader boards based on various provider or federation wide leader boards (see Figure 3d). These kind of leader boards introduce new issues like: (*i*) user merging (e.g., there should be an automatic or semi-automatic way to identify identical users using different accounts on multiple providers), (*ii*) score alignment (the various leader boards could use different scoring systems thus they should be transformed and presented on a unified way). This could allow global leader boards covering even unrelated providers (i.e., those that are not part of the same federation). Consequently, it would be possible to set up a global ranking resulting in the maximum level of competition possible. The operational and management issues of this kind of leader boards are out of scope of this paper and will be addressed in future work.

5. Approaching spot pricing like behavior

Although leader boards already achieve significant behavioral changes, there are several user characteristics that they cannot support. Their support is impossible, because leader boards cannot reveal the temporal behavior

of the infrastructure (e.g., letting users know that there is a chance of multi-tenancy at a particular moment). Therefore, first, we analyzed the behavioral patterns of Table 1, then we identified those cases where academics could have a chance to achieve higher scores if they would have some insight on the behavior of others or the inner workings of the infrastructure. In the following, we list these cases and also provide discussion on how academics could reach higher scores in these cases:

Instantiate a smaller VM If an academic has a computational task then he/she has to formulate the resource requirements for it. Very few academics know their resource demands precisely, thus showing them the possible scoring changes a resource requirement would result in could really help their decisions. E.g., if the academic would know that utilizing lesser resources would result in bigger scores, then the expected future score could push the academic to a compromise on resource requests.

Limit tasks to reserved VMs In this case, an academic already has several virtual machines that yield high scores. When a new computational task arises for such academic, he/she should be tempted to maintain his/her high scores. Thus the academic has to decide to either postpone the task or utilize a new VM for it. This decision is however dependent on what kind of resources can be acquired for the task. The user thus decides whether acquiring a resource right now would hinder his/her score or would strengthen it. If the new VM would strengthen the score then the task will be executed in this new VM, otherwise the task will be postponed until the already available VMs of the user can process it.

Price & need triggered If the user's task is not urgent, then one could limit the execution of the task only on resources which would generate higher scores. However, one would need to know when such higher scoring resources are available, otherwise its VM requests would end up on resources that negatively affect the user's scores.

Abrupt VM abortion Because of shared physical machines, the score of the user's VM will be heavily dependent on those who share the same machine. E.g., if a VM is terminated then the remaining VMs on the same host will start to receive lesser scores. Thus these users have three basic options: (*i*) ensure the utilization of the currently not utilized resources (e.g., by requesting a VM on their own or attracting others), (*ii*) checkpoint and terminate their own VM then use the previous – price & need triggered – case to determine when to resume, or (*iii*) migrate their VM to another machine that could result higher scores.

Based on these cases, we identified the need for a mechanism that allows users to know the state of the underlying infrastructure. The remainder of this section introduces the concept of engaging options that acts as the foundation for such mechanism.

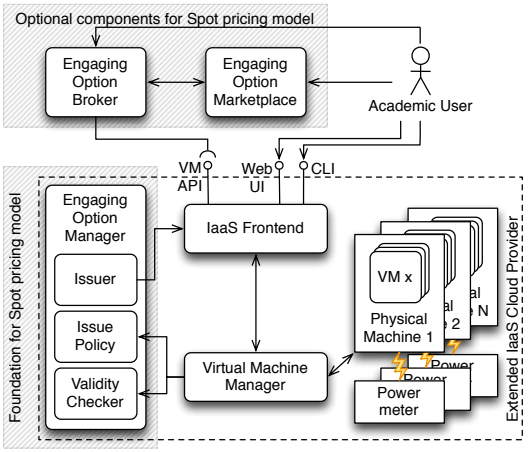


Figure 4: Architectural extensions towards engaging options

5.1. *The definition of engaging options*

We define an engaging option as an electronic document that represents a non binding, non exclusive, on the other hand time limited offer to utilize a particular set of resources on a physical machine. This document (e.g., an XML file) contains the following items: (*i*) the descriptor of the resources represented by the offer (e.g., 1 CPU, 500 MB of memory or a single micro instance etc.), (*ii*) the unique identifier of the physical host where the resources are available, (*iii*) expected maximum energy footprint of the resource, (*iv*) the issue date, (*v*) the maximum length of validity (e.g., until two hours from the issue date), (*vi*) access details to the provider that offers the resources (e.g., a URL to the provider's IaaS frontend or a textual description on how to get access) and (*vii*) a signature of the issuer (i.e., to show that the document is authentic).

Engaging options allow users to know when and in which infrastructure there is an underutilized physical machine available. Thus users are no longer deemed to request VMs at arbitrary moments. Instead, they can utilize engaging options (if available and valid) to approach their providers and receive their VMs on a shared machine. In other words, the engaging option is a mechanism to attract users at the right time to the right machine (so the virtual machine manager of the IaaS system will be able to utilize the underlying physical machines more efficiently – while their users will end up with higher scores on the leader boards).

Next, according to Figure 4, we detail (*i*) how an IaaS system would operate with engaging options, and (*ii*) how academics are expected to behave after the introduction of the concept, especially in the scope of the behavioral patterns identified in Table 1.

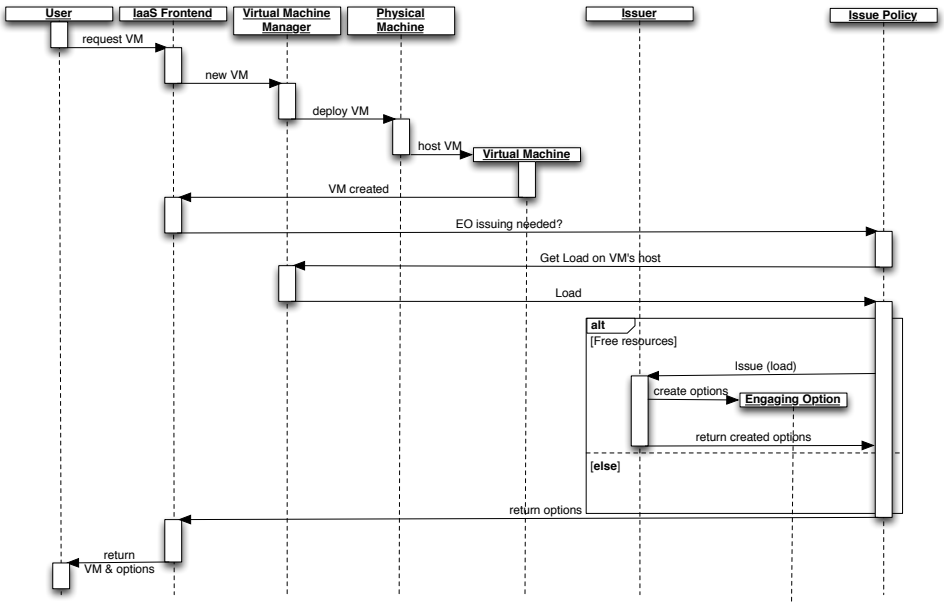


Figure 5: Sequence of emitting an engaging option during VM creation

5.2. Issuing engaging options

Within the engaging option manager (see Figure 4), the engaging option issue policy is the main entity that is responsible to decide when, to whom, how many and with what length of validity should the system emit engaging options. The issue policy is customizable by the provider who applies it (e.g., it can define a new function that determines the length of validity). When the issue policy decides on all aspects of a new engaging option, then it passes this information to the engaging option issuer. The issuer then prepares the electronic document of the option in the user desired output format and hands it over to the user. This issuing process is exemplified in Figure 5, in which we present the IaaS behavior during the creation of a new virtual machine. Since it is the issue policy that is responsible for the decisions, we are going to detail its behavior in the following paragraphs.

First of all, the policy decides when to issue an option. Whenever a virtual machine arrives to or leaves from a physical machine the policy will try to emit new engaging options for the abundant capacities. There are three stop conditions that can halt the issuing process: (*i*) the physical machine is fully utilized – the last VM that has arrived occupied all previously available resources –, (*ii*) the physical machine hosts no VMs – the latest VM that has left was the last one –, and (*iii*) the expected length of validity for the option is too short – i.e., under the useful runtime of a VM.

Next, the policy identifies those users who should receive options. If the options are issued because of a new VM, then the requestor of the VM will receive the options. Otherwise, the options will be sent to those who still have virtual machines running on aforementioned physical machine. In both cases, the system ensures that one particular user would never have more engaging options at his/her possession than the currently available resources would necessitate. This last requirement significantly limits the number of possible engaging options in circulation, but avoids flooding the users with unnecessary options.

Afterwards, the policy decides how many options should be issued. Basically, the policy will try to issue engaging options for all available resources of a physical machine. To avoid facing a bin packing problem (because of mismatches in abundant capacities and various sized engaging options) and to ensure that the issued options fit every user's need, the issued options should be of the smallest possible granularity. Therefore, the issuer is instructed to create an option per processor core or per smallest VM instance type possible on the physical machine. Figure 5 presents this operation as a request to the function `Issue(load)`, where the `load` represents the amount of smallest granularity options needed.

When there are multiple users for a physical machine, the issue policy often emits as many engaging options for a particular resource as many users are. This approach not only allows wider visibility of the issued engaging options, but as a consequence it also increases the chances to fill the not so energy efficiently used resources. The visibility improves because it is not guaranteed that any particular user shares the engaging options he/she receives, but the more users receive them the more likely that the options are going to be shared. This is particularly useful when a new VM from a new user arrives to a host for which the system already issued some engaging options. If this new user was not attracted by a previously issued engaging option then we can assume the previous options did

not reach a wide enough community. As a result, the new user's community can be also targeted by issuing new engaging options (in parallel to the already issued ones) for the remaining capacities of the host.

Finally, the issue policy defines the length of validity of the options before they are sent to the users. Similarly to the virtual machines on the host, issued engaging options should also have a limited length of validity (i.e., it is not desired that one is attracted to a host when others already terminated their VMs). The length of validity of the engaging option is an important factor to the liveliness of the community. If it is too short, then fellow users might not be able to notice it. If it is too long then the VM placement algorithm of the provider might already have taken the resources that the engaging option represents. Conceptually, the engaging options should be valid as long as there is still one running VM on the host. Unfortunately, during issue time, the future execution time of the original VMs are unknown, thus the engaging options should be issued according to provider policies (e.g., the length of validity could be set as the median run time of its past VMs).

5.3. Attracting academic users

In this subsection, we assume that a user has just received one or more engaging options. With the options, the user is now informed about the level of inefficiencies on the physical machine its VM is scheduled on. Although the options are issued so the user would have the incentive to refer others to the same physical machine, in two cases it is not beneficial for the user to advertise the received options. These are the following cases: (*i*) plans for additional VMs in the near future, (*ii*) expected short VM lifetime on the host.

In the first case, the user already has plans to execute additional VMs, and therefore it is recommended to use the received engaging options for his/her next virtual machine requests. This strategy allows immediate energy efficiency increases for the user because the next VMs are provisioned on the same physical machine. In the second case, the user either has a task that will run on its new VM for a relatively short amount of time or expects to find a physical machine where the VM can be migrated soon. Therefore, if such engaging options would be advertised, their benefits would be negligible or even nonexistent for other users (one would attract users to a resource which he/she does not plan to use in parallel to the attracted users).

In any other cases, the received engaging options would better serve their original owners if they are shared. The user has two basic options: (*i*) sharing manually and (*ii*) sharing on a marketplace. Manual sharing involves direct interaction between academics (i.e., one sends the options to his/her fellows) and thus strengthening the communities formed around leader boards. On the other hand, manual engaging option circulation is a tedious task as it needs significant work from the sender (selecting and sending to those who might plan using a VM within a few minutes/hours) and the receiver also (filtering the already expired options). To remove the burden of engaging option handling from users we introduce the concept of engaging option marketplaces (see Figure 4).

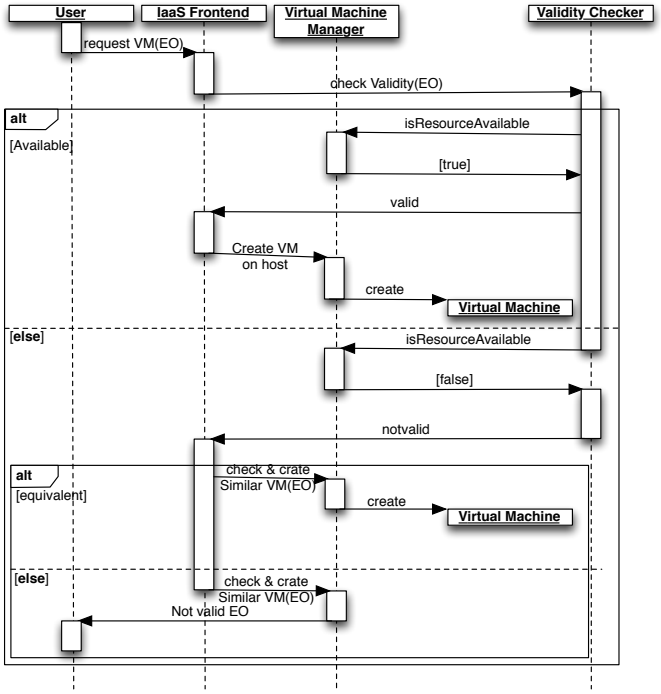


Figure 6: Using an engaging option to create a VM

An engaging option marketplace is a component that allows users to promote their engaging options to other users. Users can add new engaging options while others can query, acquire (and remove) them from a browsable index. In a marketplace, users browse the advertised engaging options so they can find resources that meet their requirements (regarding specific energy efficiency measures or resource models). The wider the community a marketplace reaches is the better options it can offer to its users. Therefore, it is expected that marketplaces are offered alongside leader boards. Thus, those users who already use leader boards can also exploit the offers on marketplaces to increase their score.

Finally, marketplaces can be operated independently from an the IaaS provider revealing the usefulness of engaging options across cloud federations. For example, we envision federation wide engaging option marketplaces. With these large marketplaces, it becomes possible to guide those users who are hesitant about their choice of infrastructure (e.g., one can decide to use a particular infrastructure for its computations when he/she sees a convincing engaging option from that infrastructure).

5.4. Utilizing the acquired options

5.4.1. Basic use of engaging options

To make sure of the highest scores possible on the leader boards, users with immediate computing needs would acquire engaging options (representing highly energy efficient resources) either from their fellow academics or from marketplaces. When requesting a virtual machine from the provider, users would also present the acquired engaging options (see Figure 6). If multiple engaging options are used for a single virtual machine request, then users must ensure that the presented options have identical identifier for the hosting machine (otherwise their VM request cannot be fulfilled). At the provider, this will result in a VM request composed from all the resources represented by the specified engaging options. As engaging options are non binding and non exclusive, it might happen that the requested resources are already taken by someone. Therefore, before creating an actual VM for the user, the IaaS must *check the validity* of the engaging options it was approached with.

There could be three reasons behind an invalid engaging option: (*i*) expiration (the option is older than its length of validity), (*ii*) independent VM placement (some user already acquired the resources in question despite not having the relevant engaging option) , and (*iii*) concurrent engaging option use (some user also acquired and used an engaging option that represent the same resource). To reduce the chances of invalid engaging options, users can approach the providers and check if an engaging option still refers to available resources. If, in spite of this check, an engaging option is invalid, the provider tries to replace it with a similar one that has the same amount of resources and the same or better energy characteristics (but might be located on other physical machine). If there is no valid engaging option after these steps, then the VM request is cancelled.

Therefore, the provider now can safely assume that there will be enough resources for the virtual machine at a host which already runs some VMs (from those users who the engaging options were originally issued for). Consequently, in its next step, the provider will create the new user's virtual machine on the host referred by the engaging options presented in the VM request. Because of the placement of this new virtual machine, the energy consumption accounted to the other VMs will be reduced in their remaining lifetime (thus engaging options positively impact the scores of those who shared them).

5.4.2. Automated use of engaging options

While engaging options are already a powerful tool for users who compete for higher scores in the leader boards, automating their usage is still a must. This enables their inclusion in such scenarios where academics use automated experiments like workflows. We have identified a semi automatic (which looks for a specific engaging option and ensures its validity but leaves the VM creation to the user) and a fully automatic (which automates the VM creation process) approach that supports these scenarios. Both of these scenarios are built on our last component from Figure 4: the engaging option broker.

In the semi automatic approach, the broker is approached by a user who would like to have a virtual machine with a particular resource set and energy properties. The broker monitors the marketplaces for the user and whenever an suitable engaging option appears, it automatically checks its validity at the provider. Next, if this option is found valid, then the broker will immediately forward it to the user. This requires the user to create the VM on its own. Despite looking like a disadvantage, this is beneficial for the user if he/she cannot automate the preparation of his experimental environment in the virtual machine. With the returned engaging option the user can start the VM whenever he/she is ready for the preparation task. Since there is a manual operation at the end of this process, the user must specify the expected length of the task in the VM and the expected time when he/she will be able to utilize the resources first. With these two details, the broker delays the marketplace monitoring process until the chance of utilizing a successfully identified and validated engaging option is high.

In the automatic approach, the broker is approached by a similar user as before. The only difference that this time the user is capable to automate the preparation of the experiment in a newly created virtual machine. Since the requested virtual machine can operate automatically (i.e., it starts processing user tasks without human intervention and terminates if there are no more tasks), the broker will start monitoring the marketplaces immediately after receiving the user request. When there is a suitable valid engaging option, the broker will even instantiate the necessary VM for the user. With this procedure, academics will exercise the same behavior as those commercial users who apply price & need triggered virtual machine instantiation strategy in case of spot pricing models (see Table 1).

6. Evaluation

In this section, we reveal our analysis on how scientific infrastructures are impacted while academic user behavior is transformed to a more commercial like behavior. For our analysis, the behavior of scientific users is really important. Thus we briefly turn our attention on the infrastructure's general properties, then we discuss the simulated user behavior on the system.

In all of our simulations, we assume an unlimited cloud infrastructure that offers a single VM type (with one core) and a single kind of physical machine with 8 cores. We applied a simple VM placement policy: new VMs are placed on the first physical machine with available cores. If there are no physical machines with free capacities then a new physical machine is switched on and the new VM is placed on that physical machine. If a physical machine does not host a single VM then it is switched off. We also assumed

Overall		Users/cluster			Tasks/cluster		
Workload	No. of clusters	Min	Max	Median	Min	Max	Median
Grid5000	11593 (42)	1 (2)	98 (=)	2 (13)	1 (1004)	5025 (=)	47 (1504)
LCG	186 (73)	4 (17)	54 (=)	24 (28)	40 (1000)	4378 (=)	815 (1510)
AuverGrid	4380 (71)	1 (12)	94 (=)	2 (47)	1 (1003)	15058 (=)	5 (2181)
DAS2	7905 (35)	1 (=)	30 (24)	4 (6)	1 (1009)	3737 (=)	108 (1210)
SHARCNET	4692 (204)	1 (=)	92 (=)	2 (30)	1 (1004)	29700 (=)	5 (2190)

Table 2: Overview of the clustering results for the different workloads – numbers in parentheses reveal the details on clusters with over 1000 jobs, later on we only use these clusters for the statistical analysis

that amongst the switched on physical machines only one has less VMs than cores. When this is not the case we assumed that the IaaS automatically migrates VMs from less crowded to more crowded physical machines to minimize the count of necessary physical machines switched on. With these infrastructural assumptions we ensure that the momentary resource utilization will not experience large variance because of the internals of the simulated IaaS system. The elimination of this variance is important because otherwise it would be impossible to tell whether resource utilization reductions are caused by internal IaaS operations or by the behavioral changes we intend to simulate for the users.

To simulate user behavior, we have turned our attention towards workload traces offered by the scientific community. We have looked for traces that fulfill the following three criteria: (*i*) represent extensive durations because the usage patterns for particular users become more clear in the long term, (*ii*) focus on a large user base so we will have a chance to see the behavior of a multitude of users and (*iii*) offer a description which clearly associates activities and users. After employing these criteria, we have selected the grid workload archive (GWA – [32]) as a good representative for the behavior of our expected users (unfortunately our last criteria eliminates the possibility of using some of the well known traces like PlanetLab). Although, this archive is not cloud oriented, we assumed that the frequency of the user tasks and the user's overall behavior regarding the use of the infrastructure is independent from the underlying technologies. In the following subsections we first present how we processed the scientific workload in GWA for our analysis. Then we present an in depth overview on the expected impacts of our architecture on the resource utilization of academic clouds.

6.1. Trace processing methods

Before our analysis, we have processed the traces so we can simulate both commercial and academic user characteristics (in terms on how and when they would instantiate virtual machines for their tasks). Throughout the simulation, we have focused our attention to three of the main user characteristics listed in Table 1: (*i*) delay instantiation, (*ii*) early VM termination and (*iii*) ensure continuous use. In our simulations, we did not analyze the performance compromise characteristic because of our assumptions on the simulated infrastructure (i.e., there is only one VM type usable), and because of the exact resource utilization pattern of the user tasks are not specified in the traces (thus one cannot tell if a smaller VM would be sufficient for the user).

As our first trace processing step, we have analyzed the tasks and identified those that have the highest chance to be delayed and grouped together in a virtual machine (so that the tasks can be grouped like a commercial user would group them). We assumed that no particular user would delay its tasks for an indefinite amount of time. Therefore, after some time, even the most scoring conscious users would run their tasks on resources (or new VMs) that would negatively impact their score. The more commercial like the user is the more he/she is willing to delay his/her tasks hoping for a more energy efficient (thus better scoring) resource opening. To identify those tasks that together could lead to the minimum amount of VM instantiations, we have processed the traces with a data mining algorithm called K-Means. Based on submission time proximity, this algorithm clusters those tasks together that are the least likely to cause impatience in users (we assumed that a user becomes impatient when he/she would have to wait for more than two hours for a more energy efficient resource). Table 2 shows details about the five selected workload traces and their clustered tasks. From the formed clusters, we have selected those that could statistically represent an entire trace and we only kept those for further analysis (the properties of these clusters are revealed in parentheses). These clusters all contain over 1000 tasks. In the rest of the paper, we only focused our analysis on these clusters.

As the next processing step, for each cluster we have run a user simulator which assigned the user's tasks in the cluster to virtual machines (ensuring that users can only assign tasks to their own VMs). Based on the commercial user characteristics, we have identified three assignment approaches: (*i*) act as a non-transformed academic user, (*ii*) employ early VM termination if possible, (*iii*) delay tasks until they would form a continuous block and create a VM for them. In the next paragraphs, we detail these three options.

With the first approach, we simulated the academic users before their transformation: we assumed users would create a VM as soon as there is a task to be executed and

the user does not have a VM that could execute it immediately. This could lead to several VMs in parallel depending on the number of tasks that must be executed in a particular instance of time. When a task finishes in a particular VM, the user could choose to terminate the VM. This is only done when the remaining tasks of the cluster have a parallelism less than the amount of virtual machines the user currently has. For example, when the trace has three simultaneous tasks for a while, the user would have three VMs, but immediately when there is no more chance to have three simultaneous tasks within the the cluster, the user would terminate one of his/her VMs.

The second approach partially simulates commercial like behavior: the users terminate their VMs as soon as feasible. The simulation of this approach only differs marginally from the previous solution. We set up an initial set of virtual machines utilizing that approach. Then we split these VMs by repeating the following procedure until there are no new VMs formed: (*i*) we identify the longest gap between the tasks of the original VM, (*ii*) if the length of this gap is over a threshold then we remove the original VM from the set of necessary VMs and add two new ones (one that includes all tasks before the gap, one for the rest). We define this threshold as the shortest time period for which two new VMs would render bigger overall score for the user than the original one (because of the impact of Equation 2 on the score of VMs with too short lifetimes).

Then, our last approach provides our most sophisticated approximation of the commercial like behavior. Here we assume users would postpone their tasks so they form a continuous block (one task immediately starts after the previous has finished). To ensure that we can postpone tasks indefinitely within the clusters, we assume that there are no dependencies between user tasks (this information is not included in the available traces). Thus we first delay the tasks as much as possible, then we apply the second approach to define the shortest lifetime VMs. This combination of task delays and early VM termination will result in VMs that are in continuous use for their entire lifetime.

6.2. Analysis

To prove our architecture's effectiveness, we have set up a simulation environment where every selected cluster from the GWA traces can be evaluated. A single experiment was done in the following phases: first we allowed the specification of user behavior (e.g., non-transformed academic, commercial like), then based on this behavior we determined the necessary VMs and their properties (request and termination time), next we run the virtual machine placement algorithm for these VMs in our simulated cloud infrastructure, finally we have estimated the energy use and the total CPU hours the VMs spent in the simulated environment.

In our first experiments, we assumed uniform likeliness that our architecture transforms a selected user. Thus we have specified a random order of users in which they are

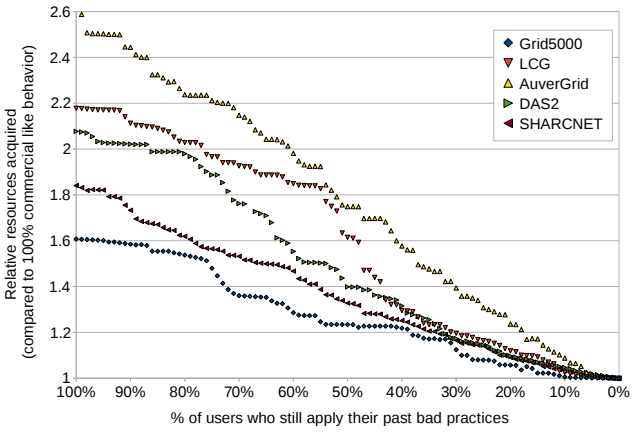


Figure 7: The effect of transforming randomly chosen users to commercial like behavior

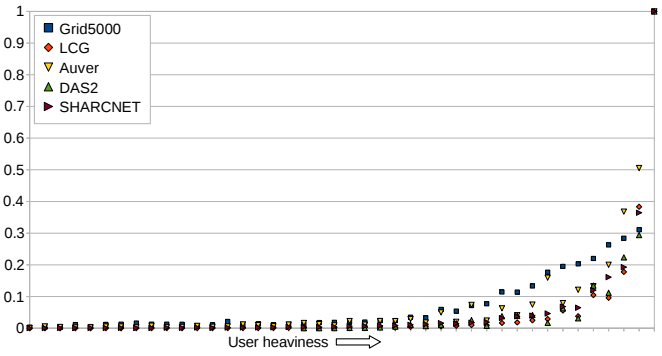


Figure 8: Chance of energy saving based on user heaviness

transformed to commercial like. At first, we have set all users to behave like commercial ones. We run an initial experiment and collected the total CPU hours spent in the system. This formed our baseline. During our first experiments, we have collected the total CPU hour figures from the experimental runs and divided by the previously acquired baseline. Figure 7 shows the possible extra resource demands (compared to our baseline) for the five analyzed traces depending on the percentage of the users who behave as non-transformed academics. Although we have evaluated the trends for all selected clusters, we presented them as an average per analyzed trace. Based on the results, we see that if only non-transformed users are using the system, then the energy and resource consumption is at its maximum. AuverGrid is impacted the most by user behavior. About 2.6 times more resources are used when all users apply the bad practices they are used to. The figure also shows, that users can be served by significantly (23%–33%) smaller infrastructure even if just half of them are converted to behave more like a commercial user.

When we looked at the non averaged trends, it clearly showed that some users have heavy impact on the resource

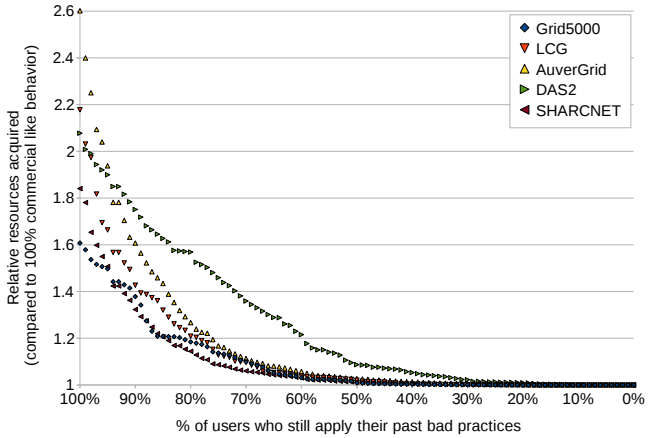


Figure 9: Prioritizing transformation of the heaviest users

utilization in the system. Therefore, we have analyzed the traces from a different point of view: if a given user transforms its behavior what is its impact on the overall resource demands? We have constructed an experiment where only one user is changed to non-transformed, the rest already behaves like commercial users. With this experiment we evaluated all the users in all the selected clusters of the five traces, then we checked the difference in energy consumption between the case when the user was non-transformed and when he/she was already acting as a commercial one. The bigger the difference the heavier the user is for the system. In Figure 8, we show how the heaviest users relate in their energy consumption to the other users in the traces. On its x axis the individual users are shown (ordered by their heaviness). To allow a less compressed figure, we omitted those users who have less than 1% of the heaviest user's energy impact. According to the figure, only a few users are responsible for the vast majority of the extra energy consumption. Thus providers should put extra effort in the advertisement of the leader boards to those users. Also, it might be beneficial to set up a hall of shame – an inverse ranking leader board – that prominently highlights those users who misbehave.

Following the analysis of the impact of the heavier users on the general saving, we evaluated a scenario where heavier users are the first to adopt commercial like behavior. We expect that targeted leader board advertisements and the hall of shame will increase the likeliness of such a scenario. With this new adoption order, we have re-evaluated our first experiments and presented the new results in Figure 9. The figure not only highlights the impact of heavy users, but also reveals that converting just half of the users result in an even more dramatic decrease in the required infrastructure size as before. The reductions are now between 37% and 61%.

Our evaluation has shown the possible transformations of academic user towards more commercial like behavior and their impact on the overall energy and resource usage. In real life systems, we expect the introduction of the leader boards will immediately turn some users towards the behavior of commercial users. Thus even though our experiments start from 100% non-transformed users, it is only a hypothetical case. The saving curve that can be expected in realistic scenarios is expected to be close to the one shown in Figure 7.

7. Conclusion & Future Work

Regrettably, even if an academic cloud provider applies similar resource management approaches as commercial providers, their resource needs and energy footprint hardly decreases because of the bad practices the academic users developed in the unconstrained environments of the past. Resources are occupied longer than needed as the limiting factor from commercial cloud providers is missing: cost. We have proposed an architecture that helps to raise the resource and power usage awareness of academic cloud users by two technologies: (i) leader boards which provide direct feedback on how closely academics behave compared to commercial users and (ii) engaging options that direct users towards more economical resource usage.

Our evaluation shows potentially high savings both in terms of utilized resources and energy if users are willing to change their usage patterns. Our architecture tries to increase their willingness to change by allowing academics to compete with fellow scientists. This competition encourages academics to self-ration their resource usage for better scores on the leader boards. As a result of self rationing, academic providers will have a chance to utilize state of the art resource management approaches to reduce energy expenses just like commercial providers. Such resource provisioning algorithms can have high influence on power draw of infrastructure and in combination with energy aware users this can lead to considerable power savings. Also, self-rationing practices often increases the throughput of the infrastructure, thus academics can use the infrastructure more productively than before the application of our architectural extensions.

We plan to continue our work into multiple directions. First, we intend to evaluate different scoring schemes in detail and we will analyze their impact on the overall resource and energy utilization. We will investigate high level leader board organizational issues like proposing new schemes to reduce possible imbalances caused by leader boards between individual providers in a cloud federation. Next, we will evaluate the possible positive effects of engaging option brokers and marketplaces on user behavior and we will investigate new behavioral patterns made possible by advanced engaging option handling. Afterwards, we plan to research engaging option issue policies and how they can result in user behavior that resembles reserved and spot pricing schemes more. Finally, we expect that this article raises the awareness of academic providers so the proposed system could be implemented and tested with real users and its general usability could be demonstrated.

[1] M. Zaharia, D. Borthakur, J. Sen Sarma, K. Elmeleegy, S. Shenker, I. Stoica, Delay scheduling: a simple technique for achieving locality and fairness in cluster scheduling, in: Proceedings of the 5th European conference on Computer systems, EuroSys '10, ACM, New York, NY, USA, 2010, pp. 265–278. doi:10.1145/1755913.1755940.

[2] S. Bhardwaj, L. Jain, S. Jain, Cloud computing: A study of infrastructure as a service (iaas), International Journal of engineering and information Technology 2 (1) (2010) 60–63.

[3] J. Diaz, G. von Laszewski, F. Wang, A. J. Younge, G. Fox, Futuregrid image repository: A generic catalog and storage system for heterogeneous virtual machine images, in: IEEE Third International Conference on Cloud Computing Technology and Science (CloudCom), IEEE, 2011, pp. 560–564. doi:10.1109/CloudCom.2011.85.

[4] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, D. Zagorodnov, The eucalyptus open-source cloud-computing system, in: 9th IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID'09)., IEEE, 2009, pp. 124–131. doi:10.1109/CCGRID.2009.93.

[5] A. Harutyunyan, P. Buncic, T. Freeman, K. Keahey, Dynamic virtual alien grid sites on nimbus with cernvm, Journal of Physics: Conference Series 219 (2010) 1–8. doi:10.1088/1742-6596/219/7/072036.

[6] D. Milojičić, I. M. Llorente, R. S. Montero, Opennebula: A cloud management tool, Internet Computing, IEEE 15 (2) (2011) 11–14. doi:10.1109/MIC.2011.44.

[7] D. Durkee, Why cloud computing will never be free, Queue 8 (4) (2010) 20:20–20:29. doi:10.1145/1755884.1772130.

[8] I. Sfiligoi, D. C. Bradley, B. Holzman, P. Mhashilkar, S. Padhi, F. Wurthwein, The pilot way to grid resources using glideinWMS, in: WRI World Congress on Computer Science and Information Engineering, Vol. 2, 2009, pp. 428–432. doi:10.1109/CSIE.2009.950.

[9] P.-O. Östberg, D. Espling, E. Elmroth, Decentralized scalable fairshare scheduling, Future Generation Computer Systems 29 (1) (2013) 130–143. doi:10.1016/j.future.2012.06.001.

[10] S. Zhuravlev, S. Blagodurov, A. Fedorova, Addressing shared resource contention in multicore processors via scheduling, SIGPLAN Not. 45 (3) (2010) 129–142. doi:10.1145/1735971.1736036.

[11] I. T. Foster, Y. Zhao, I. Raicu, S. Lu, Cloud computing and grid computing 360-degree compared, in: Grid Computing Environments Workshop, 2008. GCE '08, 2009, pp. 1–10. doi:10.1109/GCE.2008.4738445.

[12] E. Deelman, G. Singh, M. Livny, G. B. Berriman, J. Good, The cost of doing science on the cloud: the montage example, in: Proceedings of the 2008 ACM/IEEE conference on Supercomputing, SC '08, IEEE Press, Piscataway, NJ, USA, 2008, pp. 50:1–50:12. doi:10.1145/1413370.1413421.

[13] D. Kondo, B. Javadi, P. Malecot, F. Cappello, D. P. Anderson, Cost-benefit analysis of cloud computing versus desktop grids, in: IEEE International Symposium on Parallel Distributed Processing (IPDPS), 2009, pp. 1–12. doi:10.1109/IPDPS.2009.5160911.

[14] G. Feng, S. Garg, R. Buyya, W. Li, Revenue maximization using adaptive resource provisioning in cloud computing environments, in: Proceedings of the 2012 ACM/IEEE 13th International Conference on Grid Computing, GRID '12, IEEE Computer Society, Washington, DC, USA, 2012, pp. 192–200. doi:10.1109/Grid.2012.16.

[15] M. Mazzucco, D. Dyachuk, R. Deters, Maximizing cloud providers revenues via energy aware allocation policies, in: IEEE 3rd International Conference on Cloud Computing (CLOUD), 2010, pp. 131–138. doi:10.1109/CLOUD.2010.68.

[16] A. B. Yoo, M. A. Jette, M. Grondona, Slurm: Simple linux utility for resource management, in: D. Feitelson, L. Rudolph, U. Schwiegelshohn (Eds.), Job Scheduling Strategies for Parallel Processing, Vol. 2862 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2003, pp. 44–60. doi:10.1007/10968987_3.

[17] D. Jackson, Q. Snell, M. Clement, Core algorithms of the maui scheduler, in: D. G. Feitelson, L. Rudolph (Eds.), Job Scheduling Strategies for Parallel Processing, Vol. 2221 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2001, pp. 87–102. doi:10.1007/3-540-45540-X_6.

[18] C. Fischer, Feedback on household electricity consumption: a tool for saving energy?, Energy Efficiency 1 (1) (2008) 79–104. doi:10.1007/s12053-008-9009-7.

[19] D. P. Anderson, J. Cobb, E. Korpela, M. Lebofsky, D. Werthimer, Seti@home: an experiment in public-resource computing, Communications of the ACM 45 (11) (2002) 56–61. doi:10.1145/581571.581573.

[20] J. J. Dongarra, H. W. Meuer, E. Strohmaier, et al., TOP500 supercomputer sites, Supercomputer 11 (2–3) (1995) 133–194.

[21] S. Darby, The effectiveness of feedback on energy consumption. a review for defra of the literature on metering, billing and direct displays., Working paper, Oxford Environmental Change Institute (April 2006).
URL http://www2.z3controls.com/doc/ECI-Effectiveness-of-Feedback.pdf

[22] W. Chengjian, L. Xiang, Y. Yang, F. Ni, Y. Mu, System power model and virtual machine power metering for cloud computing pricing, in: Third International Conference on Intelligent System Design and Engineering Applications (ISDEA), 2013, pp. 1379–1382. doi:10.1109/ISDEA.2012.327.

[23] P. Xiao, Z. Hu, D. Liu, G. Yan, X. Qu, Virtual machine power measuring technique with bounded error in cloud environments, Journal of Network and Computer Applications 36 (2) (2013) 818 – 828. doi:http://dx.doi.org/10.1016/j.jnca.2012.12.002.

[24] Y. Jin, Y. Wen, Q. Chen, Energy efficiency and server virtualization in data centers: An empirical investigation, in: IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 2012, pp. 133–138. doi:10.1109/INFCOMW.2012.6193474.

[25] A. Kansal, F. Zhao, J. Liu, N. Kothari, A. A. Bhattacharya, Virtual machine power metering and provisioning, in: Proceedings of the 1st ACM symposium on Cloud computing, SoCC '10, ACM, New York, NY, USA, 2010, pp. 39–50. doi:10.1145/1807128.1807136.

[26] D. Huang, D. Yang, H. Zhang, L. Wu, Energy-aware virtual machine placement in data centers, in: Global Communications Conference (GLOBECOM), 2012 IEEE, 2012, pp. 3243–3249. doi:10.1109/GLOCOM.2012.6503614.

[27] S. Takahashi, A. Takefusa, M. Shigeno, H. Nakada, T. Kudoh, A. Yoshise, Virtual machine packing algorithms for lower power consumption, in: IEEE 4th International Conference on Cloud Computing Technology and Science (CloudCom), 2012, pp. 161–168. doi:10.1109/CloudCom.2012.6427493.

[28] C.-C. Lin, P. Liu, J.-J. Wu, Energy-efficient virtual machine provision algorithms for cloud systems, in: Fourth IEEE International Conference on Utility and Cloud Computing (UCC), 2011, pp. 81–88. doi:10.1109/UCC.2011.21.

[29] P. Schmitt, B. Skiera, C. Van den Bulte, Referral programs and customer value, Journal of Marketing 75 (1) (2011) 46–59.

[30] G. Ryu, L. Feick, A penny for your thoughts: Referral reward programs and referral likelihood, Journal of Marketing 71 (1) (2007) 84–94.
URL http://www.jstor.org/stable/30162131

[31] M. A. Hogg, D. Abrams, Group motivation: Social psychological perspectives., Harvester Wheatsheaf, Hertfordshire, HP2 7EZ, England, 1993, Ch. Towards a single-process uncertainty-reduction model of social motivation in groups., pp. 173–190.

[32] A. Iosup, H. Li, M. Jan, S. Anoep, C. Dumitrescu, L. Wolters, D. H. J. Epema, The grid workloads archive, Future Generation Computer Systems 24 (7) (2008) 672–686. doi:10.1016/j.future.2008.02.003.