Resource and Virtualization Costs up in the Cloud: Models and Design Choices

Daniel Gmach
HP Labs
Palo Alto, CA, USA
e-mail: daniel.gmach@hp.com

Jerry Rolia
HP Labs
Bristol, UK
e-mail: jerry.rolia@hp.com

Ludmila Cherkasova
HP Labs
Palo Alto, CA, USA
e-mail: lucy.cherkasova@hp.com

***Abstract*—Virtualization offers the potential for cost-effective service provisioning. For service providers who make significant investments in new virtualized data centers in support of private or public clouds, one of the serious challenges is the problem of recovering costs for new server hardware, software, network, storage, management, etc. Gaining visibility and accurately determining the cost of shared resources used by collocated services is essential for implementing a proper chargeback approach in cloud environments. We introduce and compare three different models for apportioning cost and champion the one that is least sensitive to workload placement decisions and provides the most robust and repeatable cost estimates. A detailed study involving 312 workloads from an HP customer environment demonstrates the result. Finally, we employ the cost model in a case study that evaluates the impact on the cost of exploiting different virtualization platform alternatives for the 312 workloads. For example, some workloads may cost more to host using certain virtualization platforms than on others or on standalone hosts. We demonstrate different decision points with potential cost savings of nearly 20% by "right-virtualizing" the workloads.**

Keywords-component; Resource Sharing, Workload Placement, Virtualization, Burstiness, Cost models

I. INTRODUCTION

Virtualization technologies promise great opportunities for reducing energy and hardware costs through server consolidation. Moreover, virtualization can optimize resource sharing among applications hosted in different virtual machines to better meet their resource needs. As a result more and more computing can be conducted in shared resource pools that act as private and public clouds. A new hot topic in cloud computing and the virtualized world is how to account for shared infrastructure usage and to chargeback the costs of running services on top of the underlying physical infrastructure. In the recent past, before the virtualization era, the accounting model was relatively simple and straightforward: the server hardware, its power usage, and software costs were directly associated with the deployed application using these resources, while the storage and networking costs were typically apportioned on a usage basis. When multiple virtual machines with different resource requirements are deployed to a resource pool and when the virtual machines may be frequently reassigned to different physical servers, the question becomes more complex: "who is responsible for the incurred costs?" and "how to attribute the cost recovery"? The focus of this paper is on the notion of cost recovery or chargeback, as opposed to pricing or what customers are willing to bid/pay for resources.

A common sense approach for establishing the cost of providing a service is to extend the usage-based model, i.e., from virtualization layer monitoring information one can derive average resource usage per application for a costing interval, e.g., three weeks, and then the physical server costs can be split up respectively. Currently, many service providers employ such simplified usage–based accounting models [1─4]. However, the relationship between workloads and costs is actually more complex. Some workloads may have a large peak to mean ratio for demands upon server resources. We refer to such workloads as *bursty*. For example, a workload may have a peak CPU demand of 5 CPU cores but a mean demand of 0.5 of a CPU core. Such ratios may have an impact on shared resource pools. A pool that aims to consistently satisfy the demands of bursty workloads will have to limit the number of workloads assigned to each server. This affects the number of servers needed for a resource pool. Thus, burstiness affects costs. Further, server resources are rarely fully utilized even when workloads are tightly consolidated and all servers are needed Even though many services can be assigned to a server, some portion of the resources remain unused over time. The amount of unused resources may depend on workload placement/consolidation choices and these choices may change frequently. The costs of such unallocated resources must be apportioned across workloads, but it should be done in a fair and predictable way. Even traditional cloud service provider pay-per-use models factor in such unusable capacity into their pay-per-use pricing.

In this paper, we discuss these issues, consider three models for apportioning server costs among workloads that share servers in such environments, and consider the implications of these different choices in a study with 312 workloads from an HP customer environment. We then employ our choice of cost model in a case study that evaluates the impact on the cost of exploiting different virtualization platform alternatives for the 312 workloads. Each alternative has its advantages and disadvantages; a key differentiator is cost. We demonstrate different decision points with potential cost savings of nearly 20% by "right-virtualizing" the workloads.

This paper is organized as follows. Section II presents the background on the workload consolidation approach and tools we employ. Section III formally introduces the notion of costs

and three models for apportioning costs. Section IV presents a workload characterization for a server consolidation exercise considered in the paper. Section V presents a study that compares the three proposed cost models. Section VI demonstrates the usefulness of the proposed cost model by evaluating design choices for a virtualized environment. Finally, we present related work and offer a summary, conclusions, and a description of our next steps.

II. BACKGROUND: WORKLOAD CONSOLIDATION ENGINE

This section briefly describes the workload consolidation engine employed in the costing method and right-virtualization studies. Its main functionality is to find an appropriate workload placement while minimizing the number of servers used for hosting these workloads. The workload consolidation engine has two components [6].

- *A simulator component* that emulates the assignment of several application workloads on a single server. It traverses per-workload historical time varying traces of demand to determine the peak of the aggregate demand for the combined workloads. If for each capacity attribute, e.g., CPU and memory, the peak demand is less than the capacity of the attribute for the server then the workloads fit on the server.
- *An optimizing search component* that examines many alternative placements of workloads on servers and reports the best solution found. The optimizing search is based on a genetic algorithm [5].

The consolidation engine supports both consolidation and load leveling exercises. Load leveling balances workloads across a set of resources to reduce the likelihood of service level violations. The engine offers the controlled overbooking of capacity and is capable of supporting a different quality of service for each workload [7]. Without loss of generality, this paper considers the highest quality of service, which corresponds to a required capacity for workloads on a server that is the peak of their aggregate demand. The engine can be used to support studies on the advantages of consolidation and for operational management [16].

III. COSTS AND APPORTIOINING COSTS

The total costs of a resource pool include the acquisition costs for facilities, physical IT equipment and software, power costs for operating the physical machines and facilities, and administration costs. Acquisition costs are often considered with respect to a three year time horizon and reclaimed according to an assumed rate for each costing interval. Without loss of generality, this paper focuses on server and virtualization software licensing costs only.

Below, we define three categories of resource usage that can be tracked separately for each server resource, e.g., CPU, memory, for each costing interval. To simplify the notation, the equations that we present consider only one server resource at a time, e.g., CPU or memory for one costing interval. Then the corresponding costs over all resources are summed up to give a total cost for all server resources for each costing interval. Final costs are the sum of costs over all costing intervals. The three categories of resource usage are:

- **Direct resource consumption by a workload**: the notation $d_{s,w}$ represents the average physical server utilization of a server s by a workload w. The values of $d_{s,w}$ are in *[0,100]*. Note, that $d_{s,w}$ is *0* if a workload w does not use a server s.
- **Burstiness for a workload and for a server**: the notation $b_{s,w}$ represents the difference between peak utilization of a server s by workload w and its average utilization represented by $d_{s,w}$. The values of $b_{s,w}$ are in *[0,100]*. Additionally, b_s represents the difference between the peak utilization of a server s and its average utilization. The values of b_s are in *[0,100]*.
- **Unallocated resource for a server**: the notation a_s represents unallocated (unused) server capacity; it is defined as the difference between 100 and the peak utilization of server s. The values of a_s are in *[0,100*]. The notation a refers to unallocated resource.

Next, we present 3 different models for apportioning cost. We refer to these as *server-usage*, *server-burst*, and *pool-burst models*.

First, we consider a **server-usage model** that takes into account only the direct resource consumption by W workloads. This a traditional usage-based approach applied by many service providers due to its simple definition and straightforward resource accounting schema. Suppose a server s has a cost C_s. The server costs include CAPEX, e.g., fraction of acquisition costs based on the length of the considered interval, as well as OPEX, e.g., costs for power associated with the server. We define a workload's server-usage share of a server as $\prod^{\text{server-usage}}_{s,w}$:

$$\prod_{s,w}^{server-usage} = C_s \frac{d_{s,w}}{\sum_{w'=1}^{W} d_{s,w'}} \quad (1)$$

We will demonstrate the outcome of the introduced models by considering and comparing the costs of two specially selected workloads in the set of 312 workloads from an HP customer environment (this workload set will be described in more detail in the next Section IV). We consolidate these workloads using the consolidation engine described in the previous Section II, and then compute their cost in the produced consolidation scenario c.

Figure 1 shows CPU demands of two workloads for three weeks (100 shares correspond to one 1GHz CPU) and demonstrates the impact of load burstiness on costs. Both workloads exhibit similar average CPU demands: 162 CPU shares for Workload A and 170 for Workload B. Using Eq. (1) for the consolidation scenario c, the CPU cost for hosting Workload A is \$36 whereas for Workload B \$39. However, this cost model does not reflect the real hosting costs for the two considered workloads. Workload A has much higher variability and much higher peaks than Workload B, 1200 CPU shares compared to 645 CPU shares, i.e., Workload A

has two times higher peaks than Workload B. Burstiness of Workload A actually causes a less dense workload placement possible on the server, and hence a lower average server utilization, and the need for more servers. The *server-usage* approach does not take into account the impact of workload burstiness on costs.

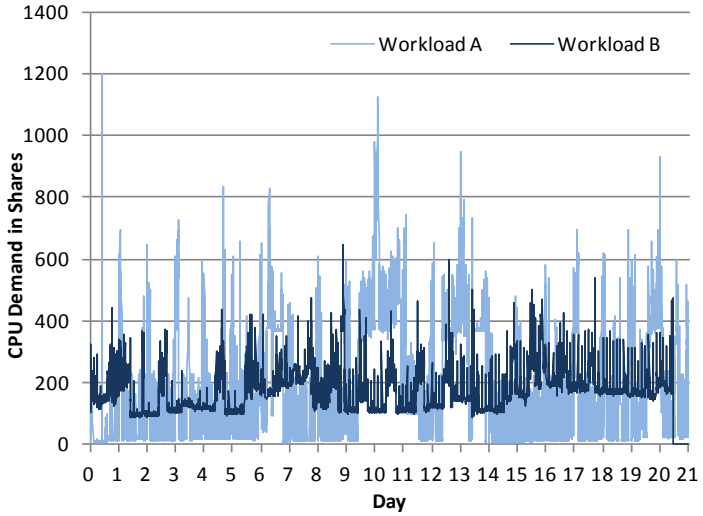

Figure 1 Example: CPU demands of two workloads

To take into account burstiness and unallocated resources we partition server cost C_s based on utilization to get C^d_s, C^b_s, C^a_s, respectively, where C^d_s corresponds to costs associated with the average utilization of the server s, and C^b_s and C^a_s correspond to the difference between peak and average utilization of the resource, and difference between 100% and the peak utilization of the resource, respectively.

$$C^d_s = C_s u^d_s, \quad C^b_s = C_s(u^b_s - u^d_s), \quad C^d_s = C_s(1 - u^b_s),$$

with the average server utilization $0 \le u^d_s \le 1$ and the peak utilization $0 \le u^b_s \le 1$.

For the **server-burst model**, we divide the burst portion of costs for a server in a manner that is weighted by the burstiness of each workload on the server. In a second step, the server's unallocated resources are apportioned based on the bursty costs. Server-burst $\prod^{server\text{-}burst}_{s,w}$ is defined as:

$$\prod^{server-burst-temp}_{s,w} = C^d_s \frac{d_{s,w}}{\sum_{w'=1}^{W} d_{s,w'}} + C^b_s \frac{\varepsilon + b_{s,w}}{\sum_{w'=1}^{W} (\varepsilon + b_{sw'})} \qquad (2)$$

$$\prod^{server-burst}_{s,w} = \prod^{server-burst-temp}_{s,w} + C^a_s \frac{\prod^{server-burst-temp}_{s,w}}{\sum_{w'=1}^{W} \prod^{server-burst-temp}_{s,w'}}$$

The ε value, a small value, in the numerator and denominator of the 2nd term of the first equation ensures that the denominator does not evaluate to zero for cases where there is no difference between peak and mean resource usage. Using Eq. (2) with consolidation scenario *c*, the total CPU costs are $54.7 for Workload A and $21.7 for Workload B. The difference stems from the fact that Workload A is much burstier than Workload B.

The proposed server-burst model incorporates more accurately the workload resource usage patterns over time in the cost structure. However, note that dividing costs in this way may lead to a lack of robustness for workload costs. The computed costs are sensitive to the placement of workloads on servers. In particular, the amount of unused resources at the server depends on the workloads assigned to the server, and hence may differ under different placement decisions. This might correspond to a significant portion of the cost. It may change based on placement decisions and therefore introduces variability for a workload's reported share of cost. Intuitively, a cost for a given workload should be mostly defined by the amount of resources used and the resource usage pattern and should be independent of workload placement decisions: the customer should not be charged different costs for the same workload under different workload placement scenarios.

To provide a more robust cost estimate, we introduce the following **pool-burst model** that attributes burstiness cost and unallocated resources using measures for the *S* servers in the resource pool instead of the individual servers.

$$\prod^{pool-burst-temp}_{s,w} = C^d_s \frac{d_{s,w}}{\sum_{w'=1}^{W} d_{s,w'}} + \left(\sum_{s'=1}^{S} C^b_{s'}\right) \frac{\varepsilon + b_{s,w}}{\sum_{s'=1,w'=1}^{S,W} (\varepsilon + b_{s',w'})} \qquad (3)$$

$$\prod^{pool-burst}_{s,w} = \prod^{pool-burst-temp}_{s,w} + \left(\sum_{s'=1}^{S} C^a_{s'}\right) \frac{\prod^{pool-burst-temp}_{s,w}}{\sum_{s'=1,w'=1}^{S,W} \prod^{pool-burst-temp}_{s',w'}}$$

Using Eq. (3) with consolidation scenario *c*, the total CPU costs are $62.2 for Workload A and $23.0 for Workload B. This approach attributes all the unallocated resources in the server pool in a fair way among all the workloads and makes it less dependent on the workload placement decisions.

In Section V, we present a study that compares cost apportioning results for different models defined by Eq. (1), (2) and (3). The introduced formulas are applied separately to various resources such as CPU and memory. The sum of the resulting costs represents the total costs for a workload.

IV. Workload Characterization

To evaluate the effectiveness of different cost apportioning models, we obtained three months of workload trace data for 312 workloads from one HP customer data center. Each workload was hosted on its own server, so we use resource demand measurements for the server to characterize its workload's demand trace. Each trace describes resource usage, e.g., processor and memory demands, as measured every 5 minutes.

We define CPU capacity and CPU demand in units of CPU shares. A CPU share denotes one percentage of utilization of a processor with a clock rate of 1 GHz. A scale factor adjusts for the capacity between nodes with different processor speeds or architectures. For example, the nodes with 2.2 GHz CPUs in

our case study were assigned 220 shares. We note that the scaling factors are only approximate; the calculation of more precise scale factors is beyond the scope of this paper. The memory usage is measured in GB.

Figure 2 and 3 summarize the memory and CPU usage for the workloads under study. Figure 2 shows the average and maximum memory usage for each workload. Note, that we order workloads by their average memory usage for presentation purposes. Figure 3 shows the average and maximum CPU usage of corresponding workloads. There are a few interesting observations:

- For 80% of the workloads, the memory usage is less than 2 GB. While the maximum and average memory usage are small and very close in absolute terms the peak to mean ratios are still high.
- For 10% of the workloads the memory usage is much higher, 10─70 GB; the maximum memory usage can be very large in absolute terms but the peak to mean ratios are less than 3.
- There are strong correlations: workloads with a high memory usage (both peak and average) have higher average CPU usage. Figure 3 shows that the first 30 workloads have high memory usage and higher average CPU usage than the remaining workloads.
- Most workloads have very bursty CPU demands: while most of the time these workloads have low CPU usage (80% of the workloads use on average less than 220 CPU shares, which corresponds to one physical CPU) their maximum CPU demand is rather high (42% of the workloads have a peak usage of more than 1000 CPU shares).
- The average peak to mean ratio for CPU usage was 52.6, with some workloads having a peak to mean ratio above 1000.

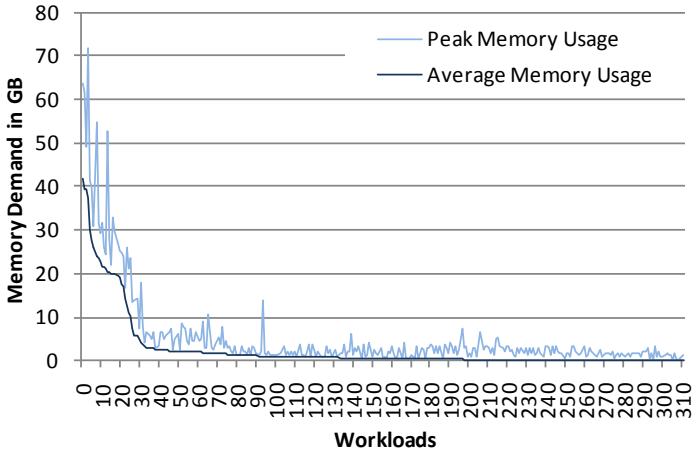

Figure 2: Workload memory usage

One of the traditional questions for any workload analysis is: how typical are the observed characteristics that are presented above? Most observations about burstiness of the CPU usage patterns were found and discussed in some other studies as well. In particular, a study presented in [8] has analyzed the CPU demands of 139 applications over a period of 5 weeks. It showed that more than half of all studied workloads are very bursty: their top 3% of CPU demand values are 2─10 times higher than the remaining CPU demands in the same workload. Furthermore, more than half of the workloads observe a mean demand less than 30% of the peak demand. These observations show the bursty nature of CPU demands for enterprise applications in different studies. Consolidating such bursty workloads onto a smaller number of more powerful servers is likely to reduce the capacity needed to support the workloads.

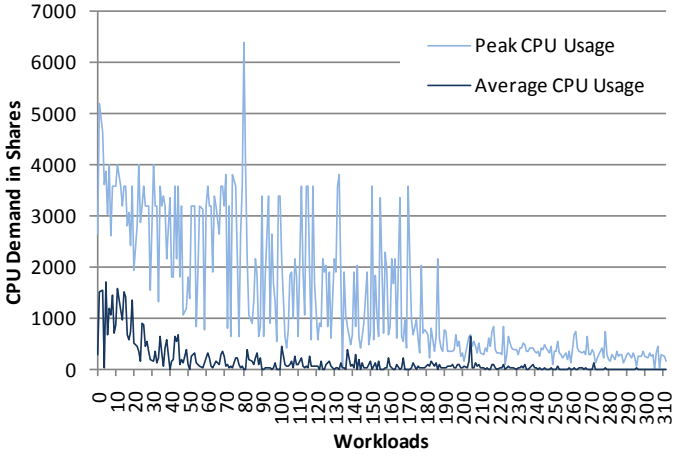

Figure 3: Workload CPU usage

V. Cost Apportioning Study

We conducted a comprehensive study using the workload data for the 312 workloads to evaluate the three introduced cost models. In the study, we consider the following shared resource pool configuration: each server consists of 24 x 2.2-GHz processor cores, 96 GB of memory, and two dual 10 Gb/s Ethernet network interface cards for network traffic and virtualization management traffic, respectively. The total acquisition cost for each of these servers was estimated as $23,000, including licensing costs. The costs were approximately $10,500 for CPU and $12,500 for memory. Using a linear depreciation and assuming a lifetime of three years the cost for three weeks is $442 per server.

For workload consolidation, we employ the consolidation engine described in Section II that minimizes the number of servers needed to host the workloads while satisfying their time varying resource demand requirements. The engine is able to offer many solutions that are near-optimal. To evaluate the robustness, i.e., repeatability, of costs assignments for our approaches, we consider 100 consolidation solutions for a three week costing interval. For the 100 solutions, the consolidation engine reported solutions that assigned the 312 workloads to between 18 and 20 physical servers in the resource pool causing the fine sharing of resources.

Figure 4, 5, and 6 show the costs for the workloads as calculated using the server-usage, Eq. (1), server-burst, Eq. (2), and pool-burst, Eq. (3), approaches, respectively.

Visual inspection shows that the server-usage and server-burst approaches have very wide ranges for the assigned costs. The average differences between min and max costs assigned to the workloads are 79% and 72%, respectively. Taking into account burstiness decreases the variability in assigned costs, but does not yet yield robust cost assignments. For example, for Workload 5 of Figure 5 cost could range from $107 to $334 for the same work. Such big differences would make it very hard to plan and charge for customers workloads.

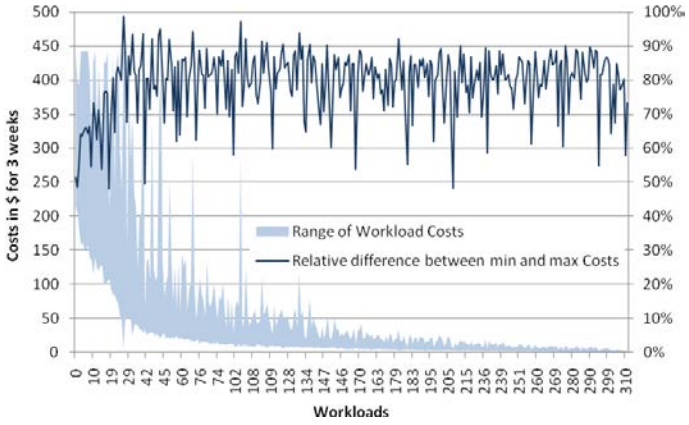

Figure 4: Server-usage model: costs with Eq. (1)

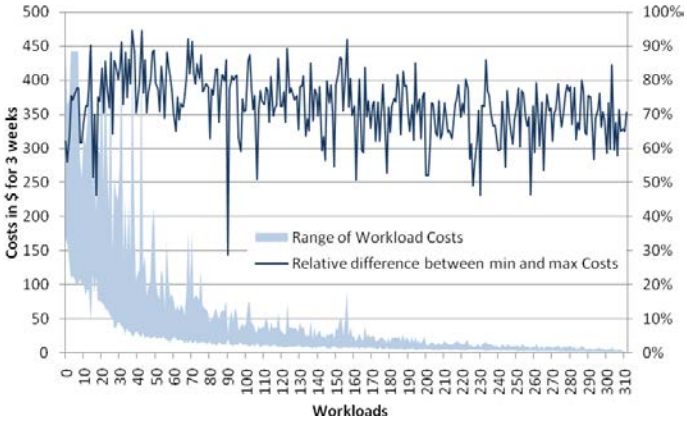

Figure 5: Server-burst model: costs with Eq. (2)

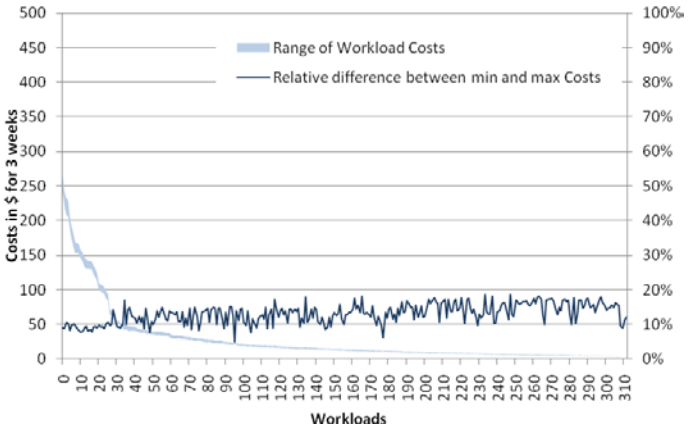

Figure 6: Pool-burst model: costs with Eq. (3)

Figure 6 shows the results for the pool-burst approach. Its cost assignment is much less sensitive to workload placement decisions and thus has a much tighter range. The average difference between min and max costs is reduced to 13%, which mostly reflects the difference that arises from consolidating to between 18 and 20 servers.

Clearly, the pool-burst model provides the most predictable cost per hosted workload. This cost is much less impacted by the workload placement outcome compared to the other two considered cost models, and it incorporates the resources usage characteristics of a given workload in the most fair way.

Our next goal is to analyze the cost breakdown with respect to its different components: i) overall cost structure with respect to direct resource usage, burstiness, and unused resources, ii) CPU cost structure with respect to direct CPU usage, CPU burstiness, and unused CPU portion, and a similar analysis for the memory cost: iii) direct memory usage, its burstiness, and unused memory resources, and finally, i) CPU vs memory cost.

Figure 7 gives a breakdown of the average sum of CPU and memory costs over the 100 consolidation scenarios as apportioned by the direct usage (according to $d_{s,w}$), bursty usage (according to $b_{s,w}$), and unallocated usage (i.e., a_s) with respect to Eq. (3). The workloads are sorted by total cost. The figure shows that for most workloads, the largest components in the costs are due to direct resource usage and usage burstiness. As defined by Eq. (3), the relative costs for unallocated resources are similar for all workloads. In this study, the unallocated costs were almost entirely due to memory costs as it is apparent from the more detailed memory cost analysis shown in Figure 9.

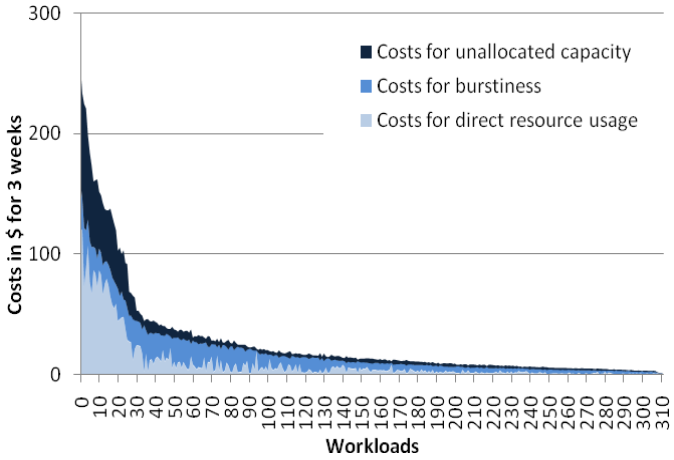

Figure 7: CPU + Memory Costs per Workload

Finally, Figure 7 shows that the ratio between costs for direct usage and for burstiness differs significantly between the workloads. This is expected as the usage burstiness is an individual characteristic of a workload, and may differ significantly across different workloads.

Figure 8 and 9 show the detailed analysis of costs distribution for direct usage, burstiness and unallocated capacity for CPU and memory, respectively. We note that in this consolidation scenario CPU was the bottleneck, so costs for unallocated CPU resources are almost negligible as shown in Figure 8, and the most of unallocated costs are due to memory costs as it is apparent from the detailed memory cost analysis shown in Figure 9. The two figures also indicate that costs for burstiness are much higher for CPU than for memory indicating that a typical memory usage pattern is less bursty compared to the CPU usage pattern.

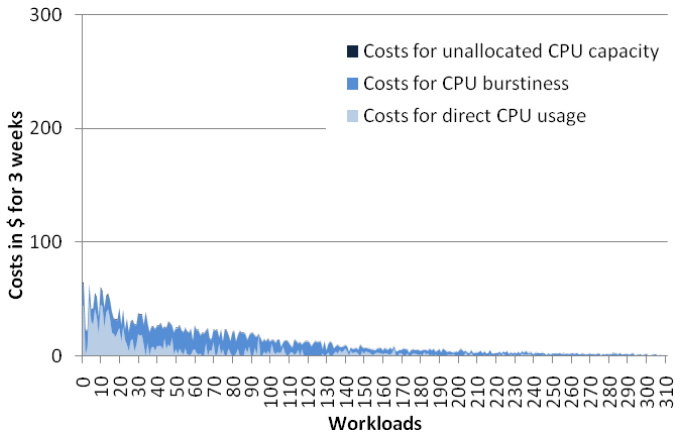

Figure 8: CPU Costs per Workload

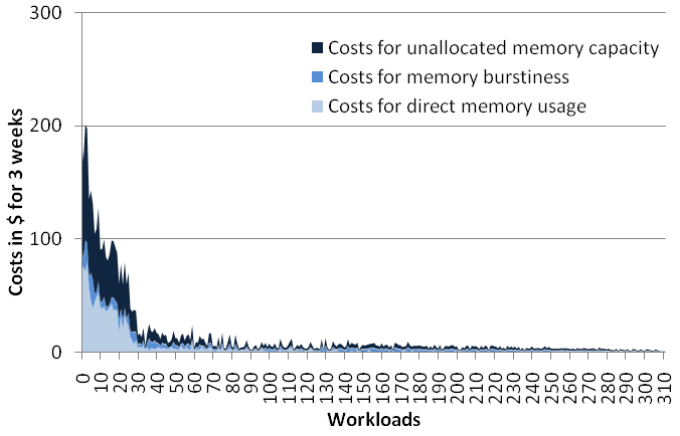

Figure 9: Memory Costs per Workload

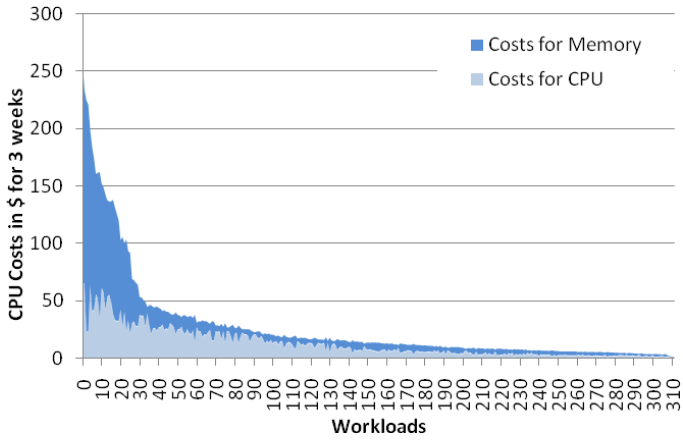

Figure 10: Sum of CPU and Mem. Costs per Workload

Finally, Figure 10 shows the sum of the costs for CPU and memory per workload in our study. Note, that the workloads are ordered in the order of decreasing total workload costs. The figure clearly shows that for most of the small workloads CPU costs dominate whereas the few really large workloads are significantly dominated by memory costs. These large cost workloads constitute less than 10% of all the workloads. Remember that the outcome of the workload consolidation engine was 18─20 servers to host all the workloads in the considered set. It means that the cost of these 20 largest workloads can dominate capacity usage on their assigned servers. The next section presents another case study that is based on the proposed cost model. It compares different alternatives for employing virtualization.

VI. TO VIRTUALIZE OR NOT

Commercial virtualization technologies offer excellent support for managing shared resource pools. Naturally, they have licensing fees. The fees per server can be comparable to the cost of stand-alone servers. As we saw in the previous sections, not all the workloads use resources in the same way in a consolidated environment. It may be that some large workloads cost more to run within a consolidated environment than to run on a dedicated server. Our goal is to design an automated approach that apportions workload cost in the shared virtualized environment to identify such workloads. Other hosting alternatives can be considered for these workloads to ensure that they are "right-virtualized." The workloads can be hosted directly on dedicated physical machines or using virtualization solutions with lower or no licensing fees. For example, a workload could be less expensively deployed to a server virtualized with Hyper-V [15] or on a server running an open-source virtualization technology such as KVM [13] or Xen [14].

Our approach takes into account the configuration of hosts and the time varying demands of workloads, i.e. resource usage traces of the application over time. The costs-per-host include the host list price, license and maintenance fees for a virtualization solution, and host power usage. Prices are obtained from the hp.com web site and power usage information from HP Power Advisor [9]. We assume a three year lifetime for the hosts. The time varying demands of workloads are customer specific.

In the ***first phase***, a desirable host configuration is chosen for the resource pool. The host has a certain capacity in terms of processing CPU cores and memory. An automated consolidation exercise packs the workloads to a small number of these hosts. A tool such as HP Capacity Advisor [10] (that is based on the CapMan [6] described earlier in Section II) can be used for this purpose. The approach takes into account the aggregate time varying (multiple) resource usage of the workloads and a given capacity of the hosts. Multiple host alternatives can be considered iteratively.

In the ***second phase***, we apportion the cost of the shared hosts in the pool among the hosted workloads using the pool-burst model introduced in Section III. If the cost associated

with a workload is greater than the cost of a smaller server that could also host the workload, then the workload is a candidate for right-virtualizing. The method can be repeated for different combinations of resource pool host and smaller server host configurations.

In the ***third phase***, we evaluate the average resource usage in the pool to make sure that the selected host configuration for the resource pool is balanced and well utilized. For example, if host memory is often less than 50% utilized we may reduce the memory size for the hosts and repeat the exercise.

To evaluate the effectiveness of our approach, we again use the three month traces of monitoring data (CPU and memory) for 312 workloads from an HP customer that was described and analyzed in Section IV. For the consolidated exercise, we consider a shared resource pool configured of HP ProLiant DL385 G7 servers each with 24 x 2.2-GHz processor cores and 96 GB of memory (similar to the configuration that was considered and analyzed in Section V). We chose the hardware configuration such that after our consolidation exercise the peak utilization of CPU and memory were balanced for the servers. The acquisition cost for each servers is estimated as $23,000, including virtualization platform licensing and support costs of $9,800 for a popular commercial virtualization solution [4]. We define CPU capacity and CPU demand in units of CPU shares (100 shares correspond to one 1GHz CPU). Memory usage is measured in GB.

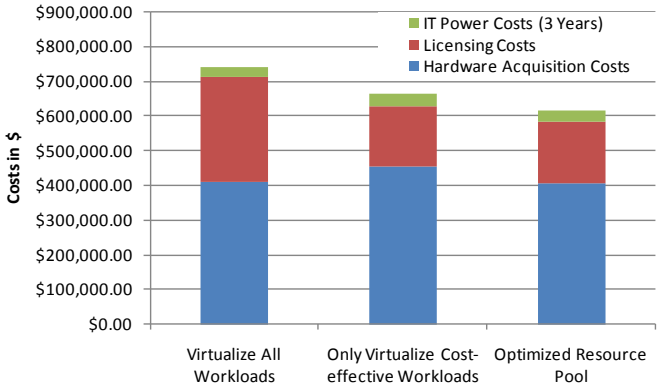

Figure 11: Costs for different scenarios

The consolidation engine minimizes the number of servers needed to host the set of workloads while satisfying their time varying resource demands [6]. Figure 11 summarizes the results. Consolidating all workloads into VMs with the popular virtualization platform requires a resource pool of 31 servers with a total cost of $741,441 for a 3 year lifetime including estimated IT power costs of $27,580 ($0.1 $/KWh).

Apportioning the costs across hosted workloads reveals that 22 workloads are the candidates for right-virtualizing. For these 22 workloads we consider DL385 G7 servers that each have two 8 core CPUs with 2.4 GHz and 72 GB of memory. We assume no additional costs for virtualization. By "right-virtualizing", the cost for the customer decreases by $77,641 (by 12%). The structure of the cost changes in the following way: hardware acquisition costs increase to $453,468 (by 10%) while virtualization costs decrease to $176,472 (by 42%).

We note that because many workloads with high memory demand are now hosted outside the pool, we are able to reduce the memory size of the resource pool nodes to 48 GB (called as the optimized resource pool) without affecting the number of workloads that can be hosted. This leads to the additional hardware savings of $49,750 for the customer and results in 18.4% of total costs savings, mostly due to lower virtualization licensing costs.

Finally, the cost of increased power demand for the optimized solution is included in our model. Power represents a small fraction of total cost for the considered servers. Large, high-end servers are often used for consolidation and are very power-efficient in this context. We note that for less power efficient and less expensive servers, power will represent a larger fraction of total cost. However, the increase in power costs for operating a few more servers is expected to be much smaller than the savings. We do not consider cooling cost and do not expect it to have much impact on total cost. Additional networking costs are also not considered.

To summarize, by considering workload costs that are based on the introduced robust cost model, a customer is offered a visibility into virtualization costs and the cost of alternative less expensive solutions. It is an important exercise that helps a customer to make an informed design choice.

VII. Related Work and Competitive Approaches

Historically, cost models in support of chargeback in shared IT systems have followed one of several strategies: no cost, fixed cost, variable cost based on resource demand, and variable cost based on volume. The choice of model depends on the desired impact of the strategy on user behavior. The pool-burst method we propose is variable cost method that can be applied to resource demands or traffic volumes. It is novel in that it addresses challenges inherent from the nature of modern shared resource pools as opposed to earlier mainframe and consolidated environments. Modern resource pools are typically made up of large numbers of servers with capacity that may be similar to a workload's demands, and where there is a great deal of flexibility regarding dynamic changes to workload placement. We have demonstrated that new cost models, such as our proposed pool-burst model, are required for these environments. These models must take into account the impact of demand burstiness that may limit the number of workloads that can be assigned to a server and resulting impact on cost for other workloads assigned to the server. The approach we introduced was shown to provide for stable chargeback results. The cost model naturally led to a "right-virtualization" case study to support decisions regarding choices for virtualization based on costs; which is a reasonable desired behavior. Pool-burst model is more similar to cost models for the electrical system that take into account the peak of power demand, or some large percentile, and total power use for cost recovery [12]. The greater the peak the more

electrical grid infrastructure must be deployed regardless of mean usage. Similarly, shared resource pool infrastructure must also be sized to handle burstiness.

Amazon EC2 [1] charges per hour for fixed sized virtual machines. We consider workloads that do not all fit on their offered machine sizes. We consolidate to much larger machines and consider a much finer sharing of resources. In [11], the authors present Joulemeter, a virtual machine power metering approach. They use models to apportion power consumption of the physical machine to the hosted virtual machines. However, their models are based on actual resource utilization only. In contrast to their work, we consider the burstiness of workloads to apportion fixed costs to the workloads in a less workload placement sensitive manner.

Some CDNs charge for video delivery on a per Mbps sustained model. This means that the customer pays for the volume of traffic at any given time, and not based on the total bits transferred. Typically, in this pricing model the user is charged for excess traffic above the volume of Mbps that he is committed to. Often a 95th percentile metric is employed where the customer is allowed to burst over the committed Mbps allotment for less than 5% of the month with no penalty. This model aims to charge for burstiness when it exceeds a predefined usage budget [17].

Customers are benefiting from the advantages of virtualization and no longer hesitate to consolidate even their production workloads. Many service providers [1][2][3] use a single virtualization platform to host and manage all workloads. Such an approach provides certain advantages but may come at a price. In our case study, the alternative design scenarios, that consider different virtualization platforms as a possible design choice, could reduce infrastructure and licensing costs by up to 20%.

VIII. Conclusions and Future Work

In this paper we introduce and compare three different cost models for apportioning costs in shared resource environments. We described workload performance features that impact resource pool costs and show that these must be taken into account if the true impact of workloads on resource pool costs is to be considered. We have shown that different apportioning approaches have an impact on the robustness of cost assignments and present an approach that offers robust, i.e., predictable, cost assignments. Cost assignments based on average usage and even burstiness were not as predictable in shared resource pool environments. The proposed pool-burst cost model supports a reliable and predictable cost apportioning in the shared compute environment, and can also be useful in support of more elaborate pricing models. In particular, we demonstrated an interesting use case of the proposed model where the customer is presented with a set of alternative "right-virtualizing" solutions (and their respective costs) for the selected workloads to be hosted with different means. The customer can compare the design choices and then make an intelligent decision about them. In our case study, these different design alternatives lead to potential cost savings of nearly 20% by "right-virtualizing" the workloads.

Our future work includes: improving our cost models to better reflect the costs for non-bottleneck resources, i.e., costs for unallocated non-bottleneck resources may be better apportioned based on the resource utilization of the bottleneck resource; applying and extending the proposed method to other aspects of cost including infrastructure, power, and human operation costs; planning for resources that are not used all the time and the relationship with pricing models; considering additional dynamism where workloads are migrated at runtime; and applying the methods to more example workloads. Finally, we will also explore the impact of using other high percentiles for resource usage rather than the peak resource usage, i.e., the 100 percentile, in our apportioning formulas.

References

[1] Amazon web services. http://aws.amazon.com/

[2] IBM Tivoli Usage and Accounting Manager Virtualization Edition. http://www-01.ibm.com/software/tivoli/products/usage-accounting/index.html

[3] HP Insight Dynamics: HP Virtual Server Environment. http://h18004.www1.hp.com/products/solutions/insightdynamics/vse-overview.html

[4] VMware: Virtualize Your Business Infrastructure. http://www.vmware.com/virtualization/

[5] J. H. Holland, Adaptation in Natural and Artificial Systems. Ann Arbor: University of Michigan Press, 1975.

[6] J. Rolia, L. Cherkasova, M. Arlitt, and A. Andrzejak: "A Capacity Management Service for Resource Pools". In Proc. of the 5th Intl. Workshop on Software and Performance (WOSP). Palma, Illes Balears, Spain, pages 229─237, 2005.

[7] L. Cherkasova and J. Rolia, "R-Opus: A Composite Framework for Application Performability and QoS in Shared Resource Pools," in Proc. of the Int. Conf. on DependableSystems and Networks (DSN), Philadelphia, USA, 2006.

[8] D. Gmach, J. Rolia, L. Cherkasova, and A. Kemper: Workload Analysis and Demand Prediction of Enterprise Data Center Applications. Proc. of the 2007 IEEE International Symposium on Workload Characterization (IISWC), Boston, MA, USA, September 27─29, 2007.

[9] HP Power Advisor. http://h18000.www1.hp.com/products/solutions/power/index.html

[10] HP Capacity Advisor. https://h20392.www2.hp.com/portal/swdepot/displayProductInfo.do?productNumber=capad

[11] A. Kansal, F. Zhao, J. Liu, N. Kothari, and A. Bhattacharya: "Virtual Machine Power Metering and Provisioning". In ACM Symposium on Cloud Computing (SOCC), Indianapolis, IN, USA, June 2010.

[12] City of Ft. Collins Electric Rates. http://www.fcgov.com/utilities/residential/rates/electric

[13] http://www.linux-kvm.org/page/Main_Page

[14] http://www.xen.org/

[15] http://en.wikipedia.org/wiki/Hyper-V

[16] D. Gmach, J. Rolia, L, Cherkasova, A. Kemper. "Resource pool management: Reactive versus proactive or let's be friends", Computer Networks, pages 2905─2922, 2009.

[17] Content Delivery Pricing: Understanding CDN Overages, http://blog.streamingmedia.com/the_business_of_online_vi/2007/10/content-deliver.html