Statistical Service Assurances for Applications in Utility Grid Environments

Jerry Rolia, Xiaoyun Zhu, Martin Arlitt, Artur Andrzejak
Internet Systems and Storage Laboratory
HP Laboratories Palo Alto
HPL-2002-155
June 13th , 2002*

E-mail: {jerry_rolia, xiaoyun_zhu, martin_arlitt, artur_andrzejak}@hp.com

resource management, grid computing, utility computing, business applications

Utility grid environments offer programmatic access to information technology (IT) resources for applications. There is a substantial literature regarding admission control, resource reservation, and scheduling in support of engineering and scientific jobs for grids. Typically each job is associated with a duration and peak requirements for resources. In this paper we focus on techniques to support admission control and advance resource reservation for applications with statistical demand characterizations. We consider business applications which require resources continuously but that have resource demands that change regularly based on factors such as time of day and day of week. We present an approach for statistically characterizing the demand profiles of business applications for utility grids. The profiles provide a grid with time varying expectations and bounds on application resource requirements. We present a method that provides statistical assurances regarding the number of resources needed to satisfy the combined requirements of many applications over time. We illustrate the feasibility of our approach with a case study that uses resource utilization information from 48 data center servers. Simulation experiments explore the sensitivity of the assurances to correlations between application resource demands and the precision of the demand profiles.

Statistical Service Assurances for Applications in Utility Grid Environments

Jerry Rolia Xiaoyun Zhu Martin Arlitt Artur Andrzejak

Internet Systems and Storage Laboratory

Hewlett Packard Laboratories

Palo Alto, CA 94304

{jerry_rolia, xiaoyun_zhu, martin_arlitt, artur_andrzejak}@hp.com

Abstract

Utility grid environments offer programmatic access to information technology (IT) resources for applications. There is a substantial literature regarding admission control, resource reservation, and scheduling in support of engineering and scientific jobs for grids. Typically each job is associated with a duration and peak requirements for resources. In this paper we focus on techniques to support admission control and advance resource reservation for applications with statistical demand characterizations. We consider business applications which require resources continuously but that have resource demands that change regularly based on factors such as time of day and day of week. We present an approach for statistically characterizing the demand profiles of business applications for utility grids. The profiles provide a grid with time varying expectations and bounds on application resource requirements. We present a method that provides statistical assurances regarding the number of resources needed to satisfy the combined requirements of many applications over time. We illustrate the feasibility of our approach with a case study that uses resource utilization information from 48 data center servers. Simulation experiments explore the sensitivity of the assurances to correlations between application resource demands and the precision of the demand profiles.

Keywords: Resource management, Grid computing, Utility computing, Business applications

1 Introduction

Grid environments offer programmatic access to information technology (IT) resources for applications. These environments have emerged to support the needs of the engineering and scientific communities. For example grid environments [16] [7] may harness the unused compute capacity of engineering workstations within an organization or provide access to specialized resources such as supercomputer clusters that are shared by many scientific researchers. These grid environments provide: languages to specify resources required by jobs and services for brokering, resource discovery and resource management.

To date grid environments have focused on support for scientific and engineering jobs. There is a substantial literature regarding admission control, resource reservation, and scheduling for the support of these kinds of jobs for grids. Jobs are typically given a start-time/end-time window and a maximum job duration within the window. Peak requirements for resources such as cpus, memory, network bandwidth, and storage capacity are identified and reserved for the duration of the job. This has proven to be effective for the batch job style typically associated with engineering and science workloads. However in the future grids are likely to support a more diverse set of applications [10].

In this paper we present an approach for grid resource management systems to support business applications. Many of these applications are likely to rely on grids with tightly coupled resources that are realized as data centers and offer IT resources on demand. We refer to these as *utility grids.* We loosely define *business applications* as those requiring resources on a continuous basis but with resource requirements that vary based on factors such as time of day and day of week. For these applications the peak and mean number of resources required can differ by a factor of 1.5 to 20 [19][2]. We expect such applications to exploit grid environments by acquiring and releasing resources as needed based on their own current workload conditions. For such applications workload patterns are often repeatable but future workload conditions, and corresponding resource demands, are only known statistically. The approach we present is also appropriate for scientific and engineering applications with processing, communication, and input-output stages that have significantly different resource requirements.

Substantial differences between peak and mean resource requirements motivate the development of admission control mechanisms that exploit statistical multiplexing. In other words there is an opportunity to overbook a utility grid's resources yet provide high statistical assurance, i.e., high probability, that resources will be made available to applications when they need them.

Our contributions in this paper are as follows. We present an approach for statistically characterizing the demand profiles of business applications for the grid and a method that uses these profiles to provide statistical assurance regarding the number of resources needed to satisfy aggregate demand. We illustrate the feasibility of our approach with a case study that uses resource utilization information from 48 data center servers. Simulation experiments explore the sensitivity of the assurances to correlations between application resource demands and the precision of the demand profiles.

Our demand characterization approach is straightforward yet provides much of the information needed to support admission control with advance resource reservation. Each customer application is characterized with respect to its pattern of repeatable behaviour. As an example a week-day profile would be appropriate for a business application if its demands are repeatable by time of day on weekdays. The profiles are based on historical observations and/or projections. We define an application's *statistical demand profile* (SDP) as a set of sequences of probability mass functions (pmfs) with one sequence per resource type (for example:

server type, networking bandwidth, storage capacity). Each sequence has one pmf per time slot where *time slots* correspond to reservation and measurement/monitoring periods that are expected to be on the order of minutes to tens of minutes.

SDPs bound the expected resource demand of customers. We present a technique that uses the SDPs along with the central limit theorem to estimate the number of resources needed to satisfy the aggregate demand of applications. A correlation factor for application loads is introduced to augment the estimate. Its magnitude is the remaining information we use to support admission control. We treat this factor as a calibration parameter for our approach.

For our system under study we found that application demands were not highly correlated at the one hour time scale and that there were clear gains possible from statistical multiplexing. Our simulation experiments show the technique did a good job at estimating the maximum aggregate number of resources needed by applications even with high (simulated) correlations in application demands. The technique was more sensitive to an accurate characterization of demand profiles.

The results presented in this paper are limited in the following ways. We consider only one resource type, namely a shared pool of identically configured servers. Applications may only attempt to acquire or release servers from the pool at the end of each time slot, i.e., measurement interval. All time slots have the same duration. All profiles have the same duration. Our focus is on whether sufficient resources are available over the duration of a profile; the allocation of specific resources is beyond the scope of this paper. We do not consider long term, i.e., monthly or longer, trends (capacity planning) for increasing or decreasing resource usage by applications. Also, we do not consider techniques for characterizing the time scales over which application loads are repeatable.

Related work is discussed in Section 2. Section 3 formally defines statistical demand profiles and presents techniques for estimating the number of resources needed to support the aggregate demand of many applications. Section 4 illustrates the feasibility of the techniques by applying them to a subset of servers from a data center. We explore the robustness of the technique using a sensitivity analysis. Section 5 gives summary and concluding remarks.

2 Related work

The grid community provides infrastructure for the support of scientific batch jobs in utility environments [16]. Jobs are typically described by their peak resource requirements, maximum job duration, start-time, and end-time. A job description is submitted to a resource manager. The resource manager uses resource availability information to decide whether it has sufficient resources to support the job. Current grid technologies rely on resource management systems such as LSF [25][17][18].

Advance reservation is appropriate for jobs that require access to large numbers of resources or access to popular resources that are hard to obtain. With advance reservation time is partitioned into slots. The slots form a calendar. Reservations are typically made in the first available slot where all required resources are available. Hollingsworth *et al.* [13] describe a method named "Imprecise Calendars" for scheduling resources on computational grids. They consider reservation calendars that operate over multiple time scales to better organize batch schedules without committing specific resources too early. They demonstrate increased effectiveness in utilizing resources with respect to previous batch approaches. Some systems make use of the concept of backfilling to reorganize schedules to make more effective use of resources. Smith *et al.* [21] present a study showing the impact of job stop/restart capabilities, backfilling, and accurate runtime predictions for jobs on the effective use of resources and job wait times for jobs in a supercomputing center. Berman *et al.* [4] present a framework to orchestrate a job's phases of execution in grid environments. At each phase the job's resource requirements may change. Their system could orchestrate a corresponding sequence of sub-jobs and their interactions with resource management systems. Their work is at the application level.

We note that the above systems do not attempt to exploit the resource savings offered by statistical multiplexing while providing statistical assurances regarding resource availability. Statistical multiplexing is important for business applications because of their continuous operation and potential for large peak-to-mean ratios in resource demands. Assurances are essential; business applications must have confidence they will have access to resources when needed.

Hechmann *et al.* [11] consider several methods for deciding a bandwidth allocation schedule for wide area network access. The methods include deterministic substitution (mixed integer programming), chance constrained stochastic optimization methods based on scenarios expressing historical demands, and recourse (penalty based) strategies. In general the techniques find a reservation schedule that minimizes overall costs by deciding times for resource re-allocation. The fewer the number of re-allocations the lower the re-allocation overhead costs but the lower the flexibility for exploiting multiplexing. Their scenario based approach could offer a similar characterization as our pmfs. However to provide statistical assurance their techniques require as constraints an enumeration of all possible scenarios for application resource requirements. Furthermore scenario modelling would have to take into account correlation. The scenario based techniques do not appear to scale for large systems.

There are resource management systems that aim to support business applications. We refer to these as *utility computing environments*. Broadly they fall into two categories. The first is a *shared server utility* model where server resources are exploited by multiple customer applications at the same time. The second is a more recent approach we define as a *full server utility* model where applications programmatically acquire and release entire servers as needed. We note that a shared server utility can act as an application to a full server utility in that it can operate as a service to its customers and acquire and release servers as needed from the full server utility data center. In general shared server utilities rely on scheduling mechanisms that operate over times scales of seconds to tens of seconds – the time needed to reconfigure server interconnections. Full server utilities operate on times scales of minutes to tens of minutes – the time needed to migrate (boot/configure) servers. Inevitably both rely on network fabrics and possibly storage systems that are shared by multiple applications.

An example of a shared utility environment is MUSE [6] in which hosted Web sites are treated as services. All services run concurrently on all servers in a cluster. A pricing/optimization model is used to determine the fraction of cpu resources allocated to each service on each server. A special dispatcher, a level 4 load balancer, routes Web page requests for all services only to those servers where the service has a cpu allocation. The optimization model shifts load to use as few servers as possible while satisfying application level Service Level Agreements (SLA) for the services. A major goal of the work is to maximize the number of unused servers so that they can be powered down to reduce energy consumption. The over-subscription of resources is dealt with via the pricing/optimization model. When resources are scarce costs increase thereby limiting demand. Commercial implementations of such goal driven technology are emerging [22][9].

In a sense our SDP approach is also goal driven. However we use historical and/or anticipated load information to specify an application's expected resource requirements. This enables support for statistical multiplexing and corresponding statistical assurances regarding resource availability. This separates concerns. Each application is solely responsible for delivering an appropriate quality of service to its customers. It must translate a quality of service to a quantity of resources required to achieve that level of service [23]. The utility is responsible for providing resources on demand with a particular level of assurance to its applications. We believe this separation of concerns is practical for many kinds of business applications. We note that if an application exceeds its expected requirements then other goal and policy mechanisms may be appropriate.

A full server utility named the Adaptive Internet Data Center is described in reference [20]. Its infrastructure concepts have been realized as a product [12]. It exploits the use of virtual LANs and SANs for partitioning resources into secure domains called virtual application environments. These environments support multi-tier as well as single-tier applications and can also contain systems that internally implement a shared server utility model. A second example of a full server utility approach is Oceano [1], an architecture for an e-business utility.

Ranjan *et al.* [19] consider QoS driven server migration within full server utility environments. They provide application level workload characterizations and explore the effectiveness of an online algorithm, Quality of Infrastructure on Demand (QuID), that decides when Web based applications should acquire and release resources from a full server utility. It is an example of an approach that could by used by an application to navigate within its SDP, i.e., to decide how many resources it needs in its next time slot.

There is a rich literature on statistical multiplexing with regard to Network Quality of Service [24] [15] [5]. Statistical multiplexing provides the potential for supporting more work with the same number of resources. The techniques provide a statistical assurance, i.e., a probability, that routers will be able to support large numbers of independent flows subject to delay and/or loss constraints. In general the techniques rely on either the central limit theorem or large deviation theory for estimating the number of flows that can be supported by a router with a given number resources and scheduling policy. The results of large deviation theory appear to provide better estimates for cases with smaller numbers of flows but offer the same results as the central limit theorem as the number of flows becomes asymptotically large [5], i.e., the shape of the aggregate traffic approaches that of the Normal distribution.

Our use of the central limit theorem is similar to the above. However our application resource demands are not expected to be independent. We explore the impact of correlations among application demands on the accuracy of our statistical assurance. Note that our approach for admission control with advance resource reservation applies to both styles of utility computing though over the time scales of minutes. For shared server models we can treat fractions of resources in the way we treat servers in server pools.

3 Statistical Demand Profiles and Assurances

This section presents notation for SDPs along with the statistical framework for estimating the number of resources needed to satisfy joint application resource demands.

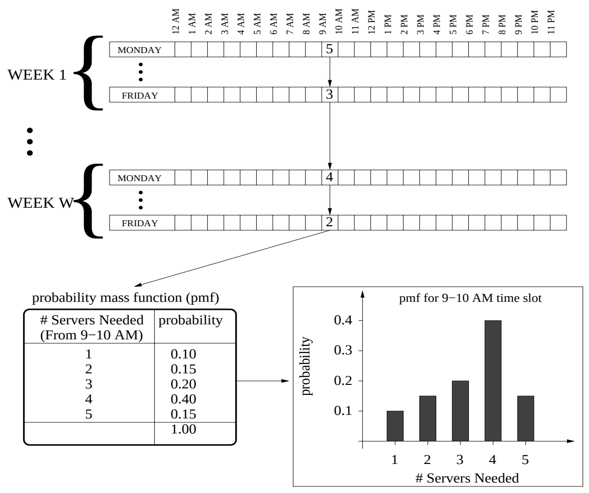
3.1 Statistical Demand Profiles

SDPs represent historical and/or anticipated resource requirements for applications. Suppose there is a particular type of resource used by an application. We model its corresponding required number of resources as a sequence of random variables, $\{\mathbf{X}_t,\ t = 1, ..., T\}$. Here each t indicates a particular time slot, and T is the total number of slots used in this profile. For example, if each t corresponds to a 60-minute time slot, and $T = 24$, then this profile represents resource requirements by hour of day.

Our assumption here is that, for each fixed t, the behaviour of $\mathbf{X}_t$ is predictable statistically given a sufficiently large number of observations from historical data. This means we can use statistical inference to predict how frequently a particular number of resources may be needed. We use a probability mass function (pmf) to represent this information. Suppose $\mathbf{X}_t$ can take a value from $\{1, \ldots, m\}$, where m is the observed maximum of the required number of resources of a particular type, then the pmf consists of a set of probabilities, $\{p_k,\ k = 1, \ldots, m\}$, where $p_k = Pr[\mathbf{X}_t = k]$. Note that although m and p_k don't have a subscript t for simplicity of the notation, they are defined within each time slot. A demand profile is composed of sequences of pmfs, each characterizing the resource requirement $\mathbf{X}_t$ for a particular time slot t and resource type.

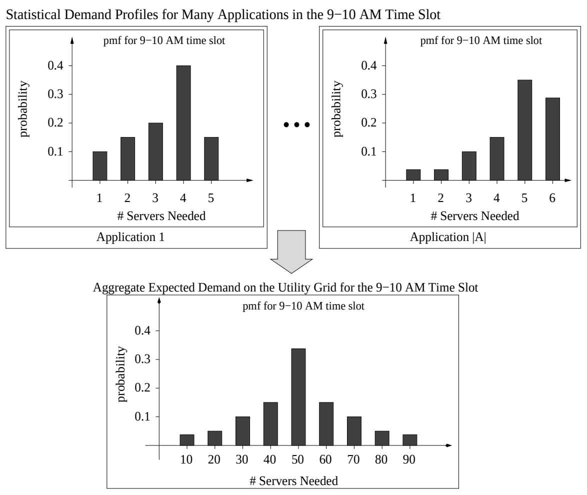
Figure 1(a) shows the construction of a pmf for the 9-10 am time slot for an SDP of an application. The application required between 1 and 5 servers over W weeks of observation. Since there are 5 observations per week there are a total of 5 W observations contributing to each application pmf. Figure 1(b) illustrates how the pmfs of many applications contribute to a pmf for the utility as a whole. As with applications, the SDP for the utility has one sequence of pmfs per resource type.

The advantages of the pmf approach are mainly two fold. First, it does not rely on any apriori knowledge of the underlying distribution for $\mathbf{X}_t$. It can be directly estimated using a sample of independent observations,



# Servers Needed (From 9−10 AM)	probability
1	0.10
2	0.15
3	0.20
4	0.40
5	0.15
	1.00

(a) A pmf of a Statistical Demand Profile



(b) A pmf of a Utility Profile

Figure 1: Pmfs of Application and Utility Profiles

which can be obtained from historical measurement of the application demand. Second, compared to a mean-variance characterization alone, it offers more flexability in terms of how it can be used. If the mean and variance of the demand are needed, they can be easily computed from the pmf. Furthermore the observed minimum and peak demands are also provided. Moreover, we will see later in the paper how the pmfs of individual applications can be used to bound the potential correlations between applications.

The pmfs in the SDP are estimated in the following way. Suppose $\{p_k,\ k = 1, \ldots, m\}$ is the "true" pmf for $\mathbf{X}_t$, which is unknown to us. What we have is a sample of N observations for $\mathbf{X}_t$. Let $\mathbf{Z}_k$ denote the number of observations that have a value k, and let $\hat{p}_k = \frac{\mathbf{Z}_k}{N}$. Then $\hat{p}_k$ gives us an estimate of the true probability p_k.

Inherent uncertainty exists in these estimated probabilities, especially when the sample size N is relatively small. This uncertainty can be bounded by computing confidence intervals for each estimated probability $\hat{p}_k$. Assume that the N observations for $\mathbf{X}_t$ are independent. Then each $\mathbf{Z}_k$ has a binomial distribution with parameters N and p_k [8]. Because p_k is unknown, we approximate its distribution using $\hat{p}_k$ as the substitute for p_k. For a given level of confidence $1-\delta$, where $\delta \in (0,1)$, suppose the confidence interval for p_k is $[p_l, p_u]$. Then p_l and p_u can be found by estimating the corresponding lower and upper percentiles for $\mathbf{Z}_k$, n_l and n_u, respectively.

$$n_l = \max_n\{n : B(N, \hat{p}_k, n) \le \delta/2\}, \qquad n_u = \min_n\{n : B(N, \hat{p}_k, n) \ge 1 - \delta/2\},$$

where $B(N,p,n) = Pr[\mathbf{Z}_k \le n]$. Now let $p_l = \frac{n_l}{N}$, and $p_u = \frac{n_u}{N}$. Then we have

$$Pr[p_l \le p_k \le p_u] \ge 1 - \delta.$$

Confidence intervals for the probabilities can help a utility assess the quality of the profile. Such information could be used to affect an admission control decision. For a given confidence level, tighter confidence intervals are achieved by increasing the number of observations N.

3.2 Statistical Assurance

The previous subsection introduced statistical demand profiles for applications. This subsection considers the number of resources needed by a utility to support many applications with a specific statistical assurance.

The utility has shared pools of resources that support its applications. The required size of a shared pool for a particular resource type is modeled as a sequence of random variables, denoted as $\{\mathbf{Y}_t,\ t = 1, \ldots, T\}$. Similar to the application demand profile, the *utility profile* consists of sequences of pmfs, with one sequence per resource type, that describe the statistical behaviour of its aggregate resource requirements for each time slot t. The construction of a pmf for a utility profile is illustrated in Figure 1(b).

Let $A = \{1, 2, \ldots, |A|\}$ denote the set of indicies for applications that require resources of the same type from the utility, where $|A|$ is the cardinality of the set A. Then at any particular time, the number of resources needed by the utility is the total number of resources needed by all the applications in A. Let $\mathbf{X}_t^a$ be the random variable for the number of resources needed by application a in time slot t. Then $\mathbf{Y}_t = \sum_{a=1}^{|A|} \mathbf{X}_t^a,\ t = 1, \ldots, T$. Based on this relationship, there are two ways to compute the utility profile using the demand profiles for individual applications.

The pmf Approach:

If the demand of each application in each time slot, $\mathbf{X}_t^a$, is independent of the demands of other applications, we can compute the pmf of the aggregate demand $\mathbf{Y}_t$ in the following way.

For each $a \in A$ and each time slot t, let $\{p_k^a,\ k = 1, \ldots, m_t^a\}$ be the pmf for $\mathbf{X}_t^a$, i.e., $p_k^a = Pr[\mathbf{X}_t^a = k]$, where m_t^a is the observed maximum of the number of resources needed by application a in time slot t. We define the joint pmf for the resource requirements by all the applications as:

$$p^A_{(k_1, k_2, \ldots, k_{|A|})} = Pr[\mathbf{X}_t^1 = k_1,\ \mathbf{X}_t^2 = k_2, \ldots, \mathbf{X}_t^{|A|} = k_{|A|}].$$

Assuming application demands are independent, we have:

$$p^A_{(k_1, k_2, \ldots, k_{|A|})} = p^1_{k_1} \times p^2_{k_2} \times \ldots \times p^{|A|}_{k_{|A|}}.$$

Denote $M_t = \sum_{a=1}^{|A|} m_t^a$ as the peak demand for $\mathbf{Y}_t$. Then the pmf for the utility is, for $k = 1, \ldots, M_t$,

$$\begin{aligned} p_k = Pr[\mathbf{Y}_t = k] &= \sum_{k_1 + k_2 + \ldots + k_{|A|} = k} p^A_{(k_1, k_2, \ldots, k_{|A|})} \\ &= \sum_{k_1 + k_2 + \ldots + k_{|A|} = k} \left(p^1_{k_1} \times p^2_{k_2} \times \ldots \times p^{|A|}_{k_{|A|}} \right). \end{aligned}$$

Unfortunately this approach does not offer a convenient method to determine the impact of correlations in application demands on aggregate demand. This motivates our second approach.

The Central Limit Theorem (CLT) Approach:

The second approach is based on the central limit theorem (CLT) which states that the sum of many independent random variables with finite variances tends to have a Normal distribution. Therefore, when the number of applications is large and their individual demands are independent, we can characterize the aggregate demand $\mathbf{Y}_t$ for each time slot t by its mean and variance, μ_t and σ_t^2. They are estimated in the following way.

Suppose the demand of application a for time slot t, $\mathbf{X}_t^a$, has a mean μ_t^a and a variance $(\sigma_t^a)^2$, which can be derived from its pmf. Then the mean and the variance of the aggregate demand for time slot t, $\mathbf{Y}_t$, can be computed as:

$$\mu_t = \sum_a \mu_t^a, \qquad \sigma_t^2 = \sum_a (\sigma_t^a)^2. \tag{1}$$

Hence, the distribution of $\mathbf{Y}_t$ is approximated by the continuous distribution $Pr[\mathbf{Y}_t \le k] = \int_{-\infty}^{k} p_t(x)$, where

$$p_t(x) = \frac{1}{\sqrt{2\pi\sigma_t^2}} e^{-\frac{(x-\mu_t)^2}{2\sigma_t^2}}. \tag{2}$$

When the demands of individual applications are correlated we can still approximate the aggregate demand by a Normal distribution, but the quality of the approximation may be poorer due to the deviation of the real distribution from the Normal distribution. In addition, the variance of $\mathbf{Y}_t$ needs to be revised as follows:

$$\sigma_t^2 = \sum_a (\sigma_t^a)^2 + 2\sum_{a\neq b} Cov(\mathbf{X}_t^a, \mathbf{X}_t^b) = \sum_a (\sigma_t^a)^2 + 2\sum_{a\neq b} \rho_t^{ab}\sigma_t^a\sigma_t^b, \tag{3}$$

where $Cov(\mathbf{X}_t^a, \mathbf{X}_t^b)$ is the covariance between the demand of application a, $\mathbf{X}_t^a$, and the demand of application b, $\mathbf{X}_t^b$, in time slot t, and ρ_t^{ab} is the corresponding correlation coefficient. For any given pair of applications, ρ_t^{ab} cannot be computed solely from the pmfs in the SDPs. Instead, its evaluation requires access to the raw observations for $\mathbf{X}_t^a$ and $\mathbf{X}_t^b$, which typically will not be available to a utility grid resource management system when an admission control decision must be made. We propose the following two schemes for bounding and estimating correlations between applications.

Worst-case Bound:

Based on the observation from Equation (3) that the variance of the aggregate demand $\mathbf{Y}_t$ increases when the correlation between each pair of applications increases, a technique called *correlation bounding* is designed to find the highest possible positive correlation between two discrete random variables with known marginal distributions. The idea is to solve a linear programming problem on the joint pmf using the marginal pmfs as the constraints and the maximization of the covariance as the objective. Using this technique, we can compute the exact upper bound on ρ_t^{ab}, $\hat{\rho}_t^{ab}$, for any pair of $\mathbf{X}_t^a$ and $\mathbf{X}_t^b$ in each time slot t. Then an upper bound on σ_t, $\hat{\sigma}_t$, can be computed as:

$$\hat{\sigma_t}^2 = \sum_a (\sigma_t^a)^2 + 2\sum_{a\neq b} \hat{\rho}_t^{ab}\sigma_t^a\sigma_t^b. \tag{4}$$

This calculation provides an estimate of the variability of the aggregate demand in the worst case.

Measurement Approach:

In practice the probability that every pair of applications are correlated to the maximum extent possible ought to be fairly low, so decisions based on the above estimate tend to be pessimistic. An alternative in an online environment is to use an approach which monitors the pairwise application correlations, and computes a single parameter, ρ_t, that captures the impact of correlations on σ_t. ρ_t can be computed using raw observations of application demands as follows:

$$\rho_t = \frac{\sum_{a \neq b} Cov(\mathbf{X}_t^a, \mathbf{X}_t^b)}{\sum_{a \neq b} \sigma_t^a \sigma_t^b}. \tag{5}$$

Note that ρ_t expresses the magnitude of the aggregate correlation as it impacts the aggregate demand. To reduce the number of parameters that characterize application correlation, by substitution, we can rewrite Equation (3) with all pairs of applications have correlation coefficient ρ_t:

$$\sigma_t^2 = \sum_a (\sigma_t^a)^2 + 2\rho_t \sum_{a \neq b} \sigma_t^a \sigma_t^b. \tag{6}$$

Let ρ be the maximum of ρ_t over all time slots t. Then the parameter ρ can be used as a calibration factor for the impact of correlations on the aggregate demand.

For admission control tests for utility grids in operation we expect to use a combination of the above two approaches. When an application is first submitted to the utility, correlations between the new application and applications already within the utility can be estimated using the pessimistic correlation bounding approach. As the new application executes, its correlation with other applications can be observed and gradually used to replace the pessimistic bound. We expect to explore this issue and whether ρ can be considered as a utility grid invariant for specific kinds of workloads in our future research.

The characterization of the utility profile enables the utility to provide statistical assurances to hosted applications so that they have a high probability of receiving a server when it is needed. In any time slot t, suppose the number of resources in a particular resource pool is Γ_t. We define θ to be the expected probability that a server is available to an application that needs it as:

$$\theta(\Gamma_t) = E[min(\frac{\Gamma_t}{\hat{\mathbf{Y}}_t}, 1)],$$

where $\hat{\mathbf{Y}}_t = min(\mathbf{Y}_t, M_t)$, and M_t is the peak demand of $\mathbf{Y}_t$ computed as the sum of peak demands of individual applications. Our motivation is as follows. Recall that $\mathbf{Y}_t$ represents the aggregate demand on a resource by all the applications. If $\mathbf{Y}_t > \Gamma_t$, among the $\mathbf{Y}_t$ resources that are needed, only Γ_t of them can be served by the utility. So the satisfaction rate is $\frac{\Gamma_t}{\mathbf{Y}_t}$. If, on the other hand, $\mathbf{Y}_t \leq \Gamma_t$, then the satisfaction rate is 100%. Note that $\mathbf{Y}_t$ is cut off at M_t because the aggregate demand should not exceed M_t based on the pmfs in the application SDPs.

Using the CLT approach, for any given value of Γ_t, θ can be computed as

$$\theta(\Gamma_t) = \int_{-\infty}^{\Gamma_t} p_t(x)dx + \int_{\Gamma_t}^{M_t} \frac{\Gamma_t}{x} p_t(x)dx + \frac{\Gamma_t}{M_t} \int_{M_t}^{\infty} p_t(x)dx, \tag{7}$$

where $p_t(x)$ is defined in Equation (2).

Conversely, given any desired value for θ, we can determine the corresponding required number of resources Γ_t for each time slot t. Let $\Gamma = \max_t \Gamma_t$. Then Γ is the required size of the resource pool in the utility so that the targeted assurance level θ can be achieved at all times.

4 Case Study

This section presents a case study involving cpu utilization from 48 servers in a data center. The purpose of the study is to demonstrate our techniques.

4.1 Measurement Data

For the purpose of our study we were able to obtain cpu utilization information for a collection of 48 servers. The servers have between 2 and 8 cpus each, with the majority having either 4 or 6 cpus. The data was collected between September 2, 2001 and October 24, 2001. For each server, the average cpu utilization across all processors in the server was reported for each five minute measurement interval. This information was collected using MeasureWare (Openview Performance) Agent [1].

We *interpret* the load of each server as an application for a full server utility grid environment. Whereas the groups' servers have multiple cpus in our study we assume the utility has many servers with one cpu each. If a server only required one cpu in an interval then its corresponding application requires one server in the utility. If the actual server required four cpus in an interval then its corresponding application requires four servers in the utility. We exploit the fact that changes in server utilization reflect real changes in required activity for its applications. Our applications have the same changing behaviour. However since our purpose is simply to validate our techniques we are not concerned with whether it is feasible for the loads on the multi-cpu servers to be hosted on many single cpu servers. Similarly we do not scale the loads with respect to processor speed and/or memory capacity.

We define the number of cpus required by a server based on a per-cpu target utilization $\hat{u}$. We consider a target $\hat{u} = 0.5$ as it is likely to be appropriate for interactive work. A target of $\hat{u} = 0.8$ may be acceptable for latency insensitive loads.

The required number of cpus k of a server for a measurement interval is computed as follows:

$$k = max(\lceil \frac{U\,\hat{k}}{\hat{u}} \rceil, 1),$$

[1]http://www.openview.hp.com/products/performance/

where U is the average utilization of the server's $\hat{k}$ cpus. We assume that at least one cpu must be available at all times. For the corresponding application, k is therefore the number of single cpu servers required for the interval.

4.2 Statistical Demand Profiles

For this case study we chose to characterize the application SDPs by weekday and by hour of day. We consider all weekdays to be equivalent, and we omit data from weekends. As a result, our profile consists of 24 one hour time slots. Since we have utilization data for 35 days, there are 35 data points that contribute to each pmf in each application's SDP.

The original measurement data was gathered at 5 minute intervals. When constructing pmfs for time slots with duration greater than 5 minutes we require one extra processing step. Consider an interval of duration b with 5 minute sub-intervals $a_1 \cdots a_j$, where j is an integer such that $j = \frac{b}{a}$. The number of servers required for b is chosen as the maximum of the number of servers required over sub-intervals $a_1 \cdots a_j$. The number of servers required for b is used to construct the pmf for its corresponding time slot. For the purpose of our study we consider time slots that are one hour in duration. They are constructed from measurement intervals of duration $b = 60$ minutes.

Table 1 lists three consecutive pmfs for a randomly selected application. In time slot t, the probability of requiring one server is 0.14; in time slot t + 1 the probability of requiring one server is 0.11. For each time slot, the sum of probabilities adds to one.

Time Slot	1 server ($\hat{p}_1$)	2 servers ($\hat{p}_2$)	3 servers ($\hat{p}_3$)	4 servers ($\hat{p}_4$)
t	0.14	0	0.66	0.20
t + 1	0.11	0.03	0.75	0.11
t + 2	0.14	0	0.83	0.03

Table 1: Subsequence of pmfs for a Randomly Selected Application, $\hat{u} = 0.5$, with 60 minute time slots

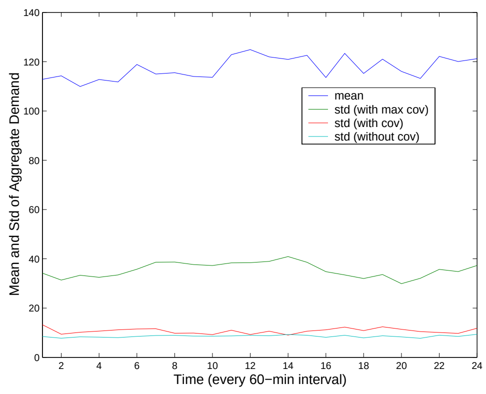
Confidence intervals for the estimated probabilities of pmfs describe uncertainty with respect to the unknown true values for the probabilities. To illustrate the use of confidence intervals, consider Table 1 time slot t. A 90% confidence level ($\delta = 0.1$) for $\hat{p}_3 = 0.66$ yields a confidence interval of $[0.51, 0.80]$. The same confidence level for $\hat{p}_1 = 0.14$ produces a narrower confidence interval of $[0.03, 0.23]$. Improving these requires more observations of each resource requirement. For instance, a sample size of 100 that leads to the same values for estimated probabilities would tighten the confidence interval for $\hat{p}_1 = 0.14$ to $[0.08, 0.20]$, but would require 13 more weeks of data.

4.3 Utility Profile

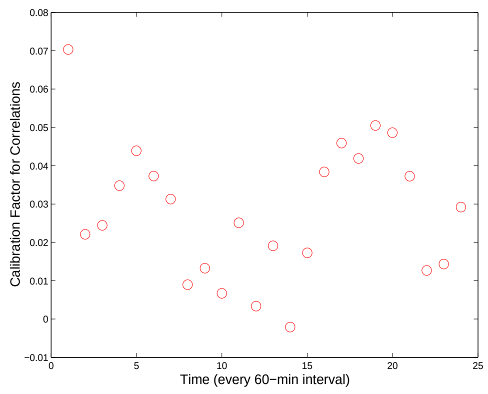
In this section we verify that the aggregate number of resources required per time slot t appears to be Normally distributed. Once verified we are able to take the SDPs of applications and use the CLT approach to compute the numbers of servers needed (Γ) to provide resources at specific levels of assurance (θ).

First we consider whether the aggregate load for each interval t appears to be Normally distributed. The Bera-Jarque test [14] [3] evaluates the null hypothesis that a given series of data, in our case $\mathbf{Y}$, has a Normal distribution with unspecified mean and variance against the alternative that $\mathbf{Y}$ does not have a Normal distribution. We found that the aggregate numbers of servers required per interval does indeed appear to behave in a similar manner as randomly chosen data from the Normal distribution. These results are not shown in this paper.

Next, we estimate the parameters of the Normal distribution (μ_t and σ_t) for each time slot t. Figure 2(a) uses raw data from the system under study to illustrate the mean and standard deviation for the utility profile. The top curve in Figure 2(a) represents the mean (μ_t) of the total demand over time, and the bottom two curves are the standard deviation (σ_t) with and without the covariance term in the computation. The middle curve represents $\hat{\sigma}_t$, the maximum value for the standard deviation computed using the correlation bounding method. Figure 2(b) shows the value for ρ_t obtained directly from the raw data. In this case ρ, the maximum of ρ_t over all time slots, is 0.07, which happens at $t = 1$. This indicates a relatively low level of correlation between applications. Accordingly, the increase in σ_t after adding the covariance is not large relative to the value of the mean. Figure 2(a) also shows that assuming the worst case for correlation (std with max cov) can be very pessimistic.



(a) μ_t and σ_t



(b) ρ_t

Figure 2: Parameters for the utility profile

The above parameters are used with Equation (7) to compute Γ for several values of θ. The results are summarized in Table 2. The first column lists the different values for θ, and the rest of the columns show the corresponding values for Γ, the number of servers required by the utility to provide a service assurance of level θ. The second column uses the standard deviation σ_t without considering correlations, while the third column is similar but σ_t incorporates the covariance term computed using raw data. The results of these two columns are similar, showing that the amount of correlation between applications and its impact on the aggregate demand for the utility is small for the system under study. We do note that correlation has a greater impact on the higher levels of assurance θ.

To better assess the advantages of statistical multiplexing for the system under study we choose to add a migration overhead to the demand profiles of applications and evaluate its impact on the utility profile. Given that applications acquire and release server resources, the original traces of server requirements from the raw data are augmented in the following way. If an additional server is required in a time slot t, then the request is made for the server in time slot $t-1$. This provides time to migrate the server into the application. Similarly if a server is no longer needed in time slot t, it does not become available to other applications until time slot $t+1$. This provides time to migrate the server from the application. We note that with 60 minute time slots this is very pessimistic. However this is still optimal from the application's point of view and as such represents a best case scenario for statistical multiplexing. The augmented traces contribute to the pmfs in the SDPs, which in turn contribute to the utility profile that is used to compute Γ.

The numbers in the last column of Table 2 take into account both application correlations and the migration overhead. The comparison between the last two columns show that the extra number of servers required due to overhead is roughly 15-20%. For a more fair comparison with the static allocation of resources, the statistical demand profiles used in the next subsection include the migration overhead unless stated otherwise.

θ	Γ (no correlation)	Γ (with correlation)	Γ (with correlation and overhead)
0.80	101	101	117
0.90	113	113	133
0.95	120	120	143
0.98	127	129	152
0.99	131	134	158
0.999	141	147	172

Table 2: Comparison of computations of Γ

4.4 Sensitivity Analysis

This section considers the sensitivity of aggregate demand to correlations in application demands. It then uses a simulation environment to validate the analytic approach and explore the accuracy of the statistical assurance to correlations in application demands and the precision of the pmfs in SDPs. A summary of the results is given.

4.4.1 Sensitivity of Γ to correlations between application demands

In this subsection we use our analytic technique to investigate the relationship between statistical assurance θ, correlations between application demands ρ, and the required number of resources per time slot Γ_t.

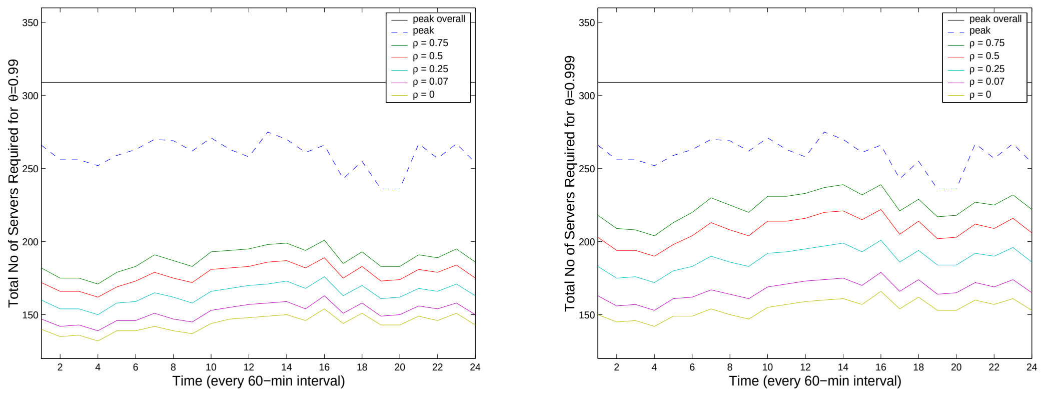
Figure 3 shows the expected value for Γ_t for $\theta = 0.99$ and $\theta = 0.999$. The top curve corresponds to the scenario where each application is always allocated its maximum required number of servers. This is the only case where the migration overhead is not included because it represents the static allocation of resources. In this case the number of servers needed by the utility is 309 servers. The second curve (dashed line) illustrates the scenario where the peak number of resources m_t^a are allocated to each application a for each time slot t. For this scenario the utility as a whole needs 275 servers to satisfy its aggregate demand for all the time slots. This demonstrates that for this system under study the potential savings due to time sharing but without statistical multiplexing is about 10% compared to a static resource allocation.

The remaining curves correspond to the scenario where each application acquires only its *required* number of servers. For these cases statistical multiplexing comes into play. Its benefit varies with the degree of correlation between applications demands. Using the correlation coefficient ρ defined in Section 3, we computed the values for Γ_t as the level of correlation decreases from 0.75 down to 0. For this data, $\rho = 0.75$ corresponds to the case where the correlation between each pair of applications is at its highest level for our system under study. In this case, the number of servers required by the utility to provide an assurance level of θ is 201 and 239 for $\theta = 0.99$ and $\theta = 0.999$, respectively. We note that even in this worst-case scenario for correlations, statistical multiplexing offers advantages over peak allocation (27% and 13%, respectively) and static allocation schemes (35% and 23%, respectively) for the system under study. As the degree of correlation goes down, the aggregate resource requirement for the utility decreases accordingly. The bottom curve represents the case when demands of all applications are mutually independent, where the benefit of statistical multiplexing is the greatest. From Figure 2(b), we see that $\rho = 0.07$. For this value the advantages of statistical multiplexing are clear for the system under study.

4.4.2 Validation by simulation

Now we describe our simulation environment. For each application, the simulator generates traces of requests for required numbers of servers in successive time slots. For each time slot the simulator draws a uniform pseudo-random number to index into the application's corresponding pmf. Correlation among the load of applications is introduced by correlating the uniform random number streams that index the application pmfs. Once the traces are prepared they are traversed on a slot by slot basis.

We use the simulated traces to validate our analytic results. The chosen values for θ and ρ as illustrated in Figure 3 are used with Equation (7) to give values for Γ for the simulations. When processing the traces, for each slot, we measure the fraction of total server requests that are satisfied. This gives the achieved value for θ for each experiment. For example, if applications require a total of 100 servers in an instance of a time



(a) $\theta = 0.99$ (b) $\theta = 0.999$

Figure 3: Γ_t, Aggregate Resource Requirements for Two Levels of Assurance with Several Correlation Levels

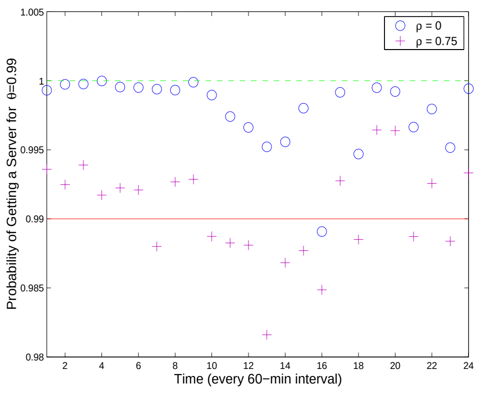
slot but only 99 are available then 0.99 contributes to the value achieved for θ as averaged over all simulated instances of that time slot.

We consider two scenarios for validation. In the first case application demands are independent – $\rho = 0$. The uniform random numbers that index different application pmfs are mutually independent. In the second case they are maximally correlated, which corresponds to $\rho = 0.75$. The same sequence of uniform random numbers is used to index all application pmfs. For each scenario we consider θ as 0.99 or 0.999. Based on our analysis, for $\theta = 0.99$, $\Gamma = 154$ and 201 for $\rho = 0$ and $\rho = 0.75$, respectively; for $\theta = 0.999$, $\Gamma = 166$ and 239 for $\rho = 0$ and $\rho = 0.75$, respectively. These numbers were used for the simulation.

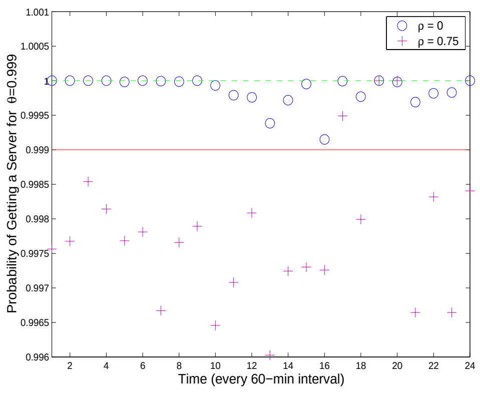
For the simulation we generated 1000 weekdays worth of observations. The results are shown in Figure 4. When application demands are independent, the estimated value for Γ is nearly always sufficient to provide the assurance level θ, with the exception of the violation for $\theta = 0.99$ at $t = 16$. However, when application demands are very highly correlated the results become poorer; this is the case despite that we take into account the correlations when estimating Γ by augmenting the variance of the aggregate demand. This is mainly due to the deviation of the aggregate distribution from the Normal distribution in the high correlation case. Nevertheless, when we specify an assurance level of $\theta = 0.99$ we actually provide an assurance level of 0.98, when we specify an assurance level of $\theta = 0.999$ we provide an assurance level of 0.996. This is encouraging as it is the worst case for correlation for our system under study.

4.4.3 Sensitivity to precision of SDPs (by simulation)

Finally we consider the sensitivity of our approach to the precision of the pmfs in the application SDPs. To illustrate we pick $\Gamma = 166$ which corresponds to $\rho = 0$ and $\theta = 0.999$. From Section 3 we know that we can approximate each probability in a pmf using a binomial distribution with parameters N and $\hat{p}_k$, where $\hat{p}_k$ is the estimated probability and N is the number of observations used in the estimation. The value of



(a) $\theta = 0.99$



(b) $\theta = 0.999$

Figure 4: Simulated Level of Assurance

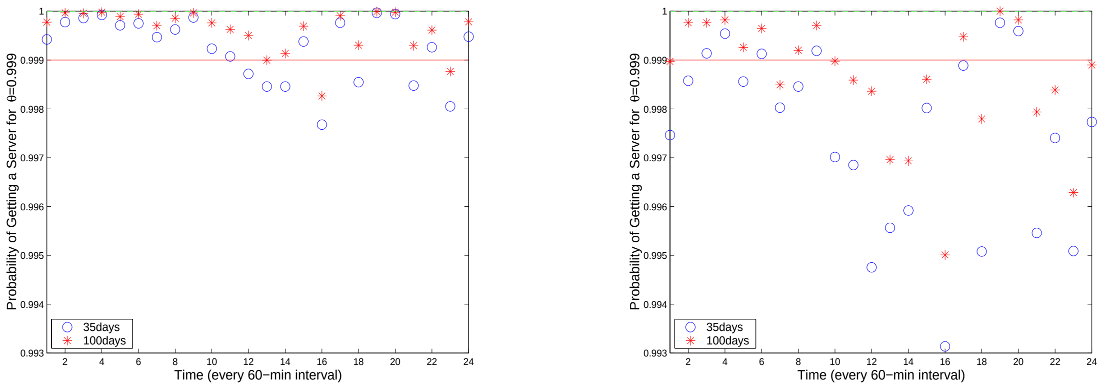
N *determines* the confidence level of the probabilities; the larger the value of N the tighter the confidence intervals. To simulate the variability of the "true" probabilities, we generated the probabilities for each pmf randomly using the binomial distribution subject to the constraint that all the probabilities in each pmf sum to one. Each *perturbed pmf* was then used to draw 100 samples for the number of required servers. This represents 100 days worth of data. This was repeated one hundred times, each offering a test case with different perturbed pmfs so that a total of 10000 days were simulated. The achieved value of θ was computed in the same way as in our validation study.

The mean and 5-percentile values for θ achieved for the 100 test cases are displayed by circles in Figure 5(a) and 5(b), respectively. The achieved values for θ are lower than the desired level due to the inherent uncertainty in the estimated pmfs. The mean of the achieved values for θ are always greater than 0.9975. Ninety-five percent of the achieved values for θ were greater than 0.993. However this is nearly an order of magnitude worse than the desired value. The lowest value achieved was 0.989. We note that the perturbed pmfs are based on the original data where $N = 35$.

To emulate an increase in confidence levels for the probabilities of the pmfs we repeat the experiments with $N = 100$, i.e., we assume nearly three times as much data was used to create the original pmfs. This provides for perturbed pmfs that are closer to the original pmfs than when $N = 35$. For comparison, the results are plotted in the same figures using stars. Ninety-five percent of the achieved values for θ were greater than 0.995. Increasing the number of observations that contribute to pmfs increases our confidence in the SDPs. In our simulation environment this leads to an increase in the statistical assurance level that is achieved. Experiments with other parameter values of θ and ρ lead to similar findings.

4.5 Summary

To summarize, we find that for the system under study the CLT approach is robust with respect to correlation between application demands. The correlation term in Equation (6) is essential and appears to adequately reflect the impact of correlation on the aggregate demand. However high values of correlation can distort the distribution of aggregate demand so that it is no longer Normal, this can lead to an achieved statistical assurance θ that is lower than the desired value. The technique is less robust with respect to inaccuracies in the demand profiles. However, increasing the accuracy of the profiles has a clear and positive impact on the 5-percentile of achieved values for θ. These factors must be taken into account when assuring a level of service.



(a) $\theta = 0.999$, $\rho = 0$, mean of θ from 100 runs (b) $\theta = 0.999$, $\rho = 0$, 5-percentile of θ from 100 runs

Figure 5: Simulated Level of Assurance with Perturbed pmf Probabilities

5 Summary and Conclusions

In this paper we focus on techniques to support admission control and advance resource reservation for applications with statistical demand characterizations for utility grid environments. We present an approach for statistically characterizing the demand profiles of business applications. The profiles provide the resource management system of a utility grid with time varying expectations and bounds on application resource requirements. We exploit the central limit theorem to estimate the number of resources needed to provide statistical assurance for access to resources while exploiting the advantages of statistical multiplexing. Statistical multiplexing permits a more efficient use of utility grid resources.

Statistical demand profiles may also be appropriate for some scientific and engineering loads that are effectively described using task graphs or whose resource requirements are best characterized statistically. The profiles are appropriate when an application/job has significant variation in its resource requirements over its lifetime.

We illustrated the feasibility of our approach with a case study that uses resource utilization information from 48 data center servers. Simulation experiments explore the sensitivity of the assurances to correlations between application resource demands and the precision of the demand profiles.

Based on our sensitivity analysis, we find that for the system under study there are clear gains to be made via statistical multiplexing with respect to simple time sharing based on peak demands. The technique we present appears to be robust with respect to correlations between application demands. However high values of correlation can distort the distribution of aggregate demand so that it is no longer Normal; this can lead to an optimistic esimate for the assurance that application requests for resources will be satisfied. The technique is less robust with respect to inaccuracies in the demand profiles. However, increasing the accuracy of the profiles had a clear and positive impact on the accuracy of the statistical assurance. A Quality of Service framework is needed to deal with the situations where resources are over-committed.

Future work includes applying the techniques to additional utility grid scenarios, positioning the methods within a Quality of Service management framework, exploring the impact of correlation on sharing, and providing integrated support for the management of multiple resource types.

References

[1] K. Appleby, S. Fakhouri, L. Fong, G. Goldszmidt, and M. Kalantar. Oceano – SLA based management of a computing utility. In *Proceedings of the IFIP/IEEE International Symposium on Integrated Network Management*, May 2001.

[2] M. Arlitt and T. Jin. Workload characterization of the 1998 world cup web site. *IEEE Network*, 14(3):30–37, May/June 2000.

[3] A. Bera and C. Jarque. Model specification tests: a simultaneous approach. *Journal of Econometrics*, 20:59–82, 1982.

[4] F. Berman and R. Wolski. Scheduling from the perspective of the application. In *Proceedings of High-Performance Distributed Computing Conference*, 1996.

[5] R. Boorstyn, A. Burchard, J. Liebeherr, and C. Oottamakorn. Statistical service assurances for traffic scheduling algorithms. *IEEE Journal on Selected Areas in Communications. Special Issue on Internet QoS.*, 18(12):2651–2664, December 2000.

[6] J. Chase, D. Anderson, P. Thakar, A. Vahdat, and R. Doyle. Managing energy and server resources in hosting centers. In *Proceedings of the Eighteenth ACM Symposium on Operating Systems Principles (SOSP)*, October 2001.

[7] K. Czajkowski, I. Foster, N. Karonis, C. Kesselman, S. Martin, W. Smith, and S. Tuecke. A resource management architecture for metacomputing systems. In *JSSPP*, pages 62–82, 1998.

[8] M. Degroot. *Probability And Statistics*. Addison-Wesley, 1984.

[9] Ejasent. Utility computing white paper, November 2001. http://www.ejasent.com.

[10] I. Foster, C. Kesselman, J. Nick, and S. Tuecke. The physiology of the grid: An open grid services architecture for distributed systems integration, www.globus.org, January 2002.

[11] O. Heckmann, J. Schmitt, and R. Steinmetz. Robust bandwidth allocation strategies. In *Tenth IEEE International Workshop on Quality of Service (IWQoS)*, pages 138–147, May 2002.

[12] Hewlett-Packard. HP utility data center architecture. http://www.hp.com/solutions1/infrastructure/solutions /utilitydata/architecture/index.html.

[13] J. Hollingsworth and S. Maneewongvatana. Imprecise calendars: an approach to scheduling computational grids. In *International Conference on Distributed Computing Systems*, pages 352–359, 1999.

[14] The Mathworks Inc. Matlab statistics toolbox, 2001. http://www.mathworks.com/access/helpdesk/help/ toolbox/stats/jbtest.shtml.

[15] E. Knightly and N. Shroff. Admission control for statistical qos: Theory and practice. *IEEE Network*, 13(2):20–29, 1999.

[16] K. Krauter, R. Buyya, and M. Maheswaran. A taxonomy and survey of grid resource management systems for distributed computing. *Software-Practice and Experience*, 32(2):135–164, 2002.

[17] M. Litzkow, M. Livny, and M. Mutka. Condor - a hunter of idle workstations. In *Proceedings of the 8th International Conference on Distributed Computing Systems*, pages 104–111, June 1988.

[18] A. Natrajan, M. Humphrey, and A. Grimshaw. Grids: Harnessing geographically-separated resources in a multi-organisational context. In *Proceedings of High Performance Computing Systems*, June 2001.

[19] S. Ranjan, J. Rolia, H. Zu, and E. Knightly. Qos-driven server migration for internet data centers. In *Proceedings of IWQoS 2002*, pages 3–12, Miami, USA, May 2002.

[20] J. Rolia, S. Singhal, and R. Friedrich. Adaptive Internet Data Centers. In *Proceedings of SSGRR'00*, L'Aquila, Italy, http://www.ssgrr.it/en/ssgrr2000/papers/053.pdf, July 2000.

[21] W. Smith, I. Foster, and V. Taylor. Scheduling with advanced reservations. In *Proceedings of the 2000 International Parallel and Distributed Processing Symposium*, May 2000.

[22] Sychron. Sychron Enterprise Manager, 2001. http://www.sychron.com.

[23] D. Xu, K. Nahrstedt, and D. Wichadakul. Qos and contention-aware multi-resource reservation. *Cluster Computing*, 4(2):95–107, 2001.

[24] Z. Zhang, D. Towsley, and J. Kurose. Statistical analysis of generalized processor sharing scheduling discipline. *IEEE Journal on Selected Areas in Communications*, 13(6):1071–1080, 1995.

[25] S. Zhou. Lsf: Load sharing in large-scale heterogeneous distributed systems. In *Workshop on Cluster Computing*, 1992.