A Trace-Based Service Level Planning Framework for Enterprise Application Clouds

Anas Youssef and Diwakar Krishnamurthy
Department of Electrical & Computer Engineering
University of Calgary
Calgary, Alberta, Canada
Email: {ayoussef, dkrishna}@ucalgary.ca

Abstract—**As enterprises increasingly adopt the cloud paradigm, there is a need for tools that can help cloud service providers (SPs) decide on service level agreements with customers. Existing approaches have not considered several important challenges such as workload burstiness, workload uncertainty, and scalability that need to be addressed to realize such tools. This paper describes a trace-based framework developed to consider these issues. We present simulation experiments that show that our framework is scalable and provides more accurate planning insights to SPs than those that ignore complex workload behaviour such as burstiness. Furthermore, the results also illustrate how SPs can exploit our framework to assess the risks of workload uncertainty.**

Keywords- ***cloud management; service level planning; workload burstiness***

I. Introduction

As enterprises increasingly adopt the cloud paradigm, cloud service providers (SPs) need tools that can help them decide on Service Level Agreements (SLAs) with a customer *prior* to deploying the customer's applications. For example, consider a customer that stipulates a certain mean response time threshold for their application's transactions. A SP should consider the application's workload as well as the workload of other existing applications on the cloud to determine whether there is adequate capacity to satisfy this requirement. If the requirement cannot be satisfied, the SP may suggest a less stringent requirement. Alternatively, the SP may determine the additional capacity needed to satisfy the request and use that information to present a revised cost estimate to the customer. Our work focuses on tools that can help SPs undertake such exercises.

Implementing service level planning (SLP) tools for enterprise application clouds requires addressing several difficult challenges. Firstly, such tools need to take into account the complex nature of enterprise application workloads which are often bursty in nature [1]. Performance models that are traditionally used for SLP exercises, e.g., product-form Queuing Network Models (QNMs) [2], are known to be inadequate for bursty workloads [3]. Furthermore, resource allocation policies evaluated as part of SLP exercises that ignore workload behaviour such as burstiness, e.g., serial correlations in request arrivals, inherent in application workloads can provide inaccurate estimates of the capacity and cost involved in delivering a target service level. Such inaccuracies can in turn impact service pricing exercises and diminish the competitive advantage of the SP. Secondly, SPs need methods to assess the risks that arise due to workload uncertainty. The workload experienced by many applications may deviate significantly from the workload projections estimated by application owners. Such applications may encounter many possible workload scenarios (WSs) each with some probability of occurrence and each placing significantly different capacity requirements on the cloud. SPs need systematic techniques to take into account such uncertainties during planning exercises. Finally, the tools should scale to support a large number of applications and a large number of WSs per application. To the best of our knowledge, we are not aware of existing methodologies which address these issues simultaneously.

This paper describes a trace-based framework that considers the challenges outlined previously. It allows a SP to consider a set of applications whose workload behaviour is characterized by traces and Service Level Objectives (SLOs) for these applications. Workload uncertainty is accommodated by allowing SPs to consider a set of probable WSs for each application. To facilitate scalability, a hierarchical clustering algorithm is used to group applications with similar WSs to form a small set of application classes for SLP exercises. A resource allocation module that employs a performance model is then invoked to estimate a probability distribution of the capacity needed in the cloud to satisfy application SLOs or, alternatively, the degree of service level violations that the SP can incur with a given cloud capacity. In particular the framework uses a resource allocation module that mimics the elastic resource allocation of clouds and the trace-based Weighted Average Method (WAM) [3] performance modeling technique that is more accurate for bursty workloads than the performance models typically used by others. Results show that these two features when combined with the clustering algorithm can permit SPs to more accurately determine the capacity required for delivering specified SLOs in large scale clouds when compared to other similar techniques. This can in turn allow SPs to accurately estimate the cost involved in delivering a target service level thereby affording them a competitive advantage.

The paper is organized as follows. Section II discusses related work. Section III provides an overview of the framework. Section IV describes our experimental setup. Section V presents experiments to evaluate the framework. Section VI concludes the paper and outlines future research.

II. Related Work

Several researchers [4, 5, 6, 7] have proposed techniques to dynamically allocate resources to applications post deployment to maintain SLAs in response to workload fluctuations. While such techniques are crucial, they need to be complemented by off-line techniques that provide pre-deployment insights to SPs [8].

A number of offline approaches have been introduced in the literature [9, 10, 11]. Rolia *et al.* [9] proposed a capacity manager service, *Quartermaster*, for enterprise applications sharing a pool of resources. *Quartermaster* relies on historical traces of observed demands, e.g., CPU, disk, and network demands, of applications and a genetic algorithm to predict resources required to satisfy future demands of applications. *Quartermaster* uses resource utilization thresholds as an indirect mechanism for achieving adequate application response times. Our work in contrast allows response time thresholds to be specified directly as part of an application's SLO.

Jung *et al.* [10] proposed an approach which combined a Layered Queuing Model (LQM) [12] with a heuristic optimization algorithm to generate optimal server configurations. These configurations were encoded in the form of rules and policies to be used while the system is running. In contrast to our work, the configuration generation process did not consider fine timescale resource allocation strategies that exploit workload burstiness. Furthermore, LQMs are not capable of reflecting the impact of burstiness unless they are integrated with a trace-based technique such as WAM [3] used in this paper.

Mylavarapu *et al.* [11] proposed a Monte Carlo technique in conjunction with a genetic algorithm to obtain an optimized virtual machine (VM) assignment scheme that ensures peak application demands are satisfied while avoiding over-provisioning of resources. In contrast to our study, this work only considers SLOs based on CPU utilization targets and does not explicitly handle workload uncertainty.

III. Framework Description

A. Overview

The SLP framework integrates a set of interacting modules as shown in Fig. 1. Each application is characterized by a set of possible WSs and each WS is characterized by a probability of occurrence to accommodate workload uncertainties. A WS is essentially a trace of sessions that describes an application's request arrival patterns and system resource usage patterns over a time epoch, e.g., 24 hours. These traces can either be historical traces obtained from production environments or be synthetically generated based on observed application workload characteristics. It is also assumed that each application has a SLO defined based on thresholds for metrics such as mean response time. Finally, the cloud is assumed to have a maximum limit C_{max} of VMs or *capacity units* (CUs).

The SLP process starts by invoking a Workload Uncertainty Module (WUM). The WUM employs a Monte Carlo simulation to generate a set of highly probable composite workload scenarios (CWSs) with total probability of occurrence equaling a specified certainty threshold. Each CWS is a combination of different WSs one for each application. The certainty threshold determines the trade-off between coverage of all possible CWSs that can arise due to workload uncertainties and SLP analysis time. Each CWS obtained is passed to a Workload Clustering Module (WCM) which groups application WSs into clusters with similar workload characteristics. Using a trace aggregation technique, the WCM produces a compact CWS composed of a WS for each application cluster.

A Resource Allocation Planning (RAP) module is used to mimic the elastic operation of clouds. It operates on the compact CWS representing WSs for application clusters to allocate resources in the cloud to applications in each cluster. This allocation can be performed over a specified planning horizon, e.g. 24 hours, with a specified resource allocation granularity, e.g. 1 hour. Specifically, the RAP module divides each cluster WS into a number of time intervals. In each interval cluster WSs are allocated up to C_{max} CUs to satisfy individual application SLOs. The RAP module uses a heuristic algorithm in conjunction with a Performance Model (PM) module based on WAM [3] to drive the CU allocation. Specifically, the PM module predicts SLO metrics of each cluster WS in each time interval. The RAP and PM modules interact to allocate CUs to applications in a manner that reflects the burstiness characteristics exhibited by the applications. The process of invoking WCM followed by RAP and PM modules is repeated for every CWS generated by the WUM. The final outcome of this process is an estimate of the distribution of the number of CUs required in each resource allocation interval for each cluster and an estimate of the number of SLO violations (if any) for each cluster. Based on these results, SPs may need to modify CU limits C_{max} or application SLOs and restart the process.

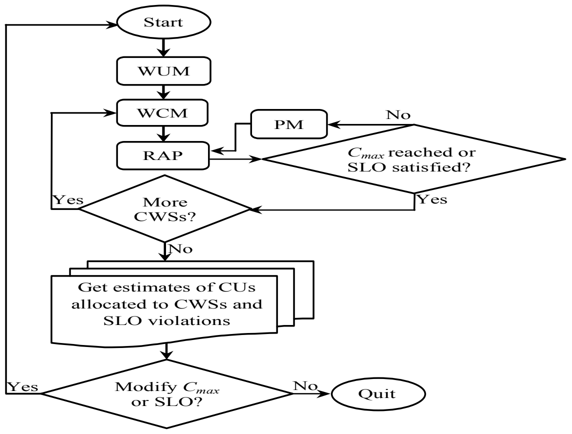
Start
WUM
WCM
PM
No
RAP
C_{max} reached or SLO satisfied?
Yes
Yes
More CWSs?
No
Get estimates of CUs allocated to CWSs and SLO violations
Yes
Modify C_{max} or SLO?
No
Quit

Figure 1. Overview of the SLP framework

B. Application Workload Characterization

Each application $a \in \{1,2,\ldots A\}$ is characterized by a WS k which is represented by a trace of sessions. The application WS describes its session and request arrival patterns, and system resource usage patterns over some time period, e.g. 24 hours. The application's session and request arrival patterns are specified by the application arrival process model. The application's system resource usage patterns are specified by the application's service process model.

The arrival process model $AM_{a,k}$ of the probable WS k of application a is characterized by the set $\{S_{a,k}, F_{a,k}, Z_{a,k}, \lambda_{a,k}, SCV_{a,k}, I_{a,k}\}$. This characterization is based on the approach described by Casale *et al.* [13]. $S_{a,k}$ denotes the number of representative sessions constituting the WS. $F_{a,k}$ and $Z_{a,k}$ specify the distribution of number of requests per session and the distribution of think time between session requests, respectively. $\lambda_{a,k}$ denotes the mean session arrival rate. The variability of the session inter-arrival time distribution is captured by $SCV_{a,k}$ which is the squared coefficient of variation (SCV) of session inter-arrival time. Finally the burstiness, i.e., correlations between successive session arrivals, is captured by the parameter $I_{a,k}$ which is the index of dispersion of session inter-arrival times [13]. Workloads with bursty session arrivals have very large values for $I_{a,k}$.

The application's service process model $SM_{a,k}$ captures the demands placed on application resources, e.g., Web server CPU and database server CPU. Specifically, each application a has a number of application tiers N_a. The application's service process model $SM_{a,k}$ is characterized by the set $\{D_{a,k,n}, SCV_{a,k,n}, I_{a,k,n}\}$ where $D_{a,k,n}$, $SCV_{a,k,n}$ and $I_{a,k,n}$ are the mean, the SCV and the index of dispersion, of service demand at tier $n \in \{1,2,\ldots N_a\}$, respectively.

IV. Experimental Setup

To evaluate the SLP framework, we used discrete event simulation models for applications. Each application a is represented by a queuing network model (QNM) consisting of two resources representing the typical web and database tiers of an enterprise application. The QNM is subjected to an input WS k consisting of a trace of customer sessions. As mentioned earlier in section III.B the WS trace of sessions is governed by the parameters of the arrival process model $AM_{a,k}$ and the service demands at the two tiers of the application are controlled by the application's service process model $SM_{a,k}$. We note that this QNM violates assumptions underlying product form models [2] solved using Approximate Mean Value Analysis (AMVA) when the input WS is characterized by arrival or service process burstiness. All the experiments in this paper define an application SLO as a target mean response time over the planning horizon to be twice the demand at the application's bottleneck resource. The resource allocation granularity is selected to be 1 hour.

V. Evaluation Results

A. Accuracy of Performance Model

We used WAM-QNM and AMVA-QNM with the RAP module to estimate the CU requirement per interval for five different CWSs characterized by progressively higher degrees of session arrival burstiness. The CU estimates obtained using WAM-QNM and AMVA-QNM are compared with the actual values obtained using discrete event simulation to calculate the absolute capacity estimation errors in each interval for the two techniques. Fig. 2 shows the mean error in the CU estimates obtained over 8 hours planning horizon from both techniques. Results show that WAM-QNM provides very accurate CU estimates. The worst case WAM-QNM mean CU estimate error is only 3%. The AMVA-QNM provides accurate results for the non-bursty exponential workload. In contrast, it provides very inaccurate results for CWSs characterized by high application workload burstiness. The worst case AMVA-QNM mean CU estimate error is 10%. The results suggest that AMVA-QNM models used by others in SLP exercises may not be appropriate for enterprise workloads that are often bursty in nature [1].

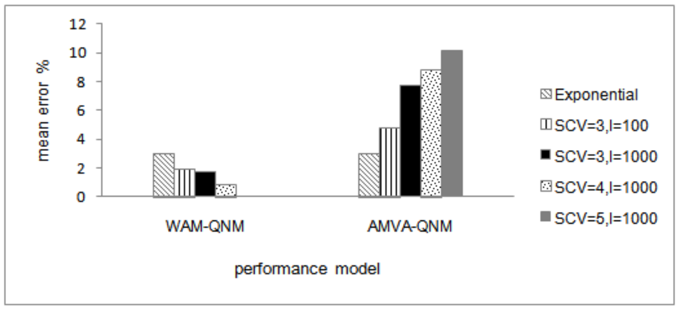

Figure 2. Mean CU allocation prediction errors over 8 hours

B. Resource Allocation Planning

To illustrate the accuracy of our burstiness-aware RAP algorithm, we compare it with two burstiness agnostic allocation approaches. The first approach, named *whole approach*, dedicates an incremental number of CUs to an application over the whole planning horizon without considering finer-grained resource allocation intervals until the application's SLO is satisfied or until all CUs have been allocated. The second approach, named *basic interval approach*, attempts resource allocation at a granularity finer than the planning horizon. CUs are allocated incrementally for each resource allocation interval for each WS in sequence starting from the first interval in the planning horizon until application SLO is satisfied or CUs in the cloud are exhausted for each interval. As described previously, the proposed RAP algorithm takes variability and burstiness into account by allocating CUs in a more sophisticated manner starting with the application having maximum violation percentage and for that application the interval with maximum mean response time is selected first.

Fig. 3 shows the effect of each of the three allocation approaches on the mean violation percentage of three different CWSs where each CWS has eight application WSs. As shown in the figure, not all applications can satisfy their SLOs because we imposed a capacity constraint of 22 CUs per interval which is less than the 24 per interval needed to avoid SLO violations for these WSs. All three allocation approaches estimate the same violation percentage for the non-bursty exponential CWSs. However, the whole and basic interval approaches provide highly pessimistic violation percentage estimates for the most bursty CWS (about 50%) while the RAP algorithm yields only a 13% violation.

C. Accuracy of Workload Clustering

To evaluate the accuracy of the WCM, a CWS consisting of 100 application WSs with various burstiness and time-of-the-hour arrival patterns is generated. Although the traces are synthetically generated, the clustering algorithm does not assume any knowledge of the trace characteristics. In this experiment, we compare the resource allocation estimates obtained using the framework with and without clustering. By design, all 100 application WSs have similar mean resource demands at two application tiers. Hence, the first level of clustering starts directly with similarities in mean session arrival rate followed by a second level of clustering based on index of dispersion of session inter-arrival time and finally the third level of clustering based on a vector of mean session arrival rates for the eight 1-hour intervals constituting the planning horizon. We evaluate three levels of clustering. The CU estimate obtained with a given clustering level for a given time interval is compared with the CU estimate obtained for that interval without clustering. The absolute difference between these values, normalized by the CU estimate obtained without clustering, is used to compute an error measure per interval.

Fig. 4 shows that the proposed clustering technique is effective in grouping similar applications together. The figure shows the CU allocation error percentiles for three different levels of clustering. (Level 1+Level 2+Level 3) clustering gives the lowest error percentiles relative to the other two levels of clustering. Using such a clustering approach, the CU allocation error obtained does not exceed 6% in all intervals. CU allocation errors obtained from (Level 1) and (Level 1+ Level 2) clustering can reach 15% and 17%, respectively. It is noted that (Level 1+Level 2+Level 3) clustering generates 31 clusters relative to 100 applications without clustering.

D. *Workload Uncertainty Characterization*

This section illustrates the utility of the WUM and its sensitivity to the certainty threshold h. Five applications are considered each has two alternative WSs. One of these WSs is used to represent the normal operation of the application. The other WS is characterized by a higher session arrival rate and/or higher burstiness relative to the normal WS. While the normal WS captures typical behaviour over a planning horizon, the other WS represents the activity during periods of heightened customer use of the application. The probability of encountering the heavy load WS p is set to 0.1. The experiment considers values of 1.0 and 0.9 for h. For illustrative purposes, we assume that this system has a capacity constraint which limits the maximum number of CUs available per interval t to $C_{max,t}$=13.

Fig. 5 shows the probability of requiring more CUs than $C_{max,t}$ in each interval for set of five applications. The figure shows that only intervals 6 and 8 are likely to be problematic to the SP since there is a high likelihood that the cloud capacity is oversubscribed in these intervals. When considering five applications with two WSs per application a total of 32 (2^5) CWSs need to be analyzed with h=1.0. With h=0.9 only 9 CWSs need to be evaluated yielding a 72% reduction in the number of CWSs. It can be observed from Fig. 5 that using a lower h value does not impact the accuracy of the probability predictions significantly. This behaviour occurs because not all CWSs in the exhaustive list of 32 have a high likelihood of occurrence. However, this is not always the case. As p is increased, i.e., as all WSs for an application become more equally likely to occur, there would be lesser reduction in the number of CWSs generated for any given h.

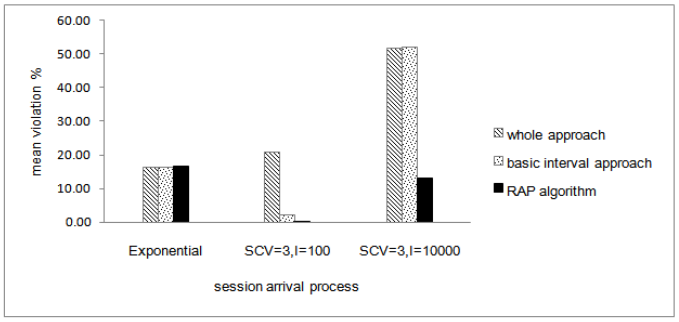

Figure 3. Mean violation percentages of three CWSs

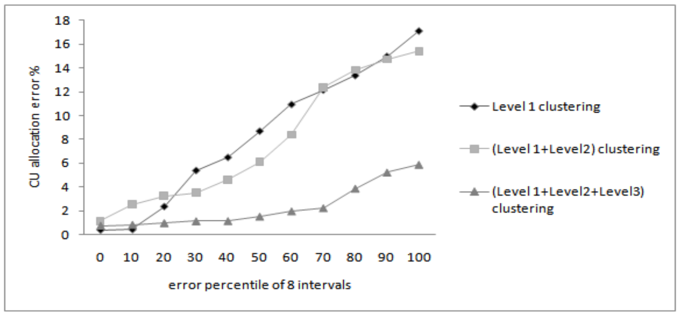

Figure 4. Percentiles of CU alloaction error per interval over 8 intervals

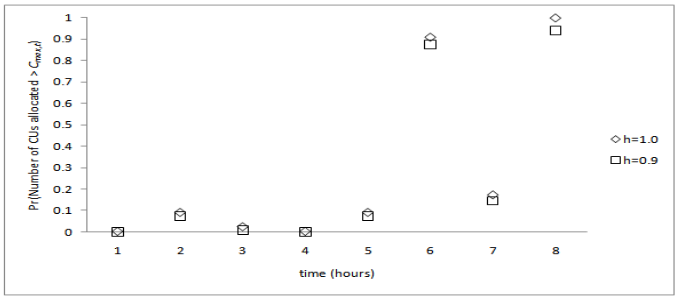

Figure 5. Probability of exceeding $C_{max,t}$ per interval for five applications with two alternative WSs, p =0.1, $C_{max,t}$ = 13

VI. CONCLUSIONS

This paper described a SLP framework for enterprise application clouds that addresses several challenges. Firstly, by considering traces it explicitly models burstiness that is known to be prevalent in enterprise applications [1] and reflects its impact on SLP through the use of the WAM modeling technique. Secondly, it explores resource allocations at fine-grained timescales for bursty workloads. Results show that this approach prevents SPs from obtaining pessimistic estimates of resource requirements and allows them to discover strategies that help limit penalties due to extreme SLO violations. Thirdly, the framework supports a clustering technique that can allow it to scale to a large number of applications. Finally, it explicitly accommodates workload uncertainty through a Monte Carlo technique. Ongoing work includes enhancing the scalability of the framework and evaluating the approach on real test beds.

Acknowledgment

This work was supported by grants from Natural Sciences and Engineering Research Council of Canada (NSERC) and Hewlett-Packard.

References

[1] U. Vallamsetty, K. Kant, P. Mohapatra, "Characterization of E-commerce traffic," WECWIS 2002, pp. 137- 144.

[2] D. Menasce, L. Dowdy, V. Almeida, "Performance by Design: Computer Capacity Planning By Example," Prentice Hall, 2004.

[3] D. Krishnamurthy, J. Rolia, M. Xu, "WAM - The Weighted Average Method for Predicting the Performance of Systems with Bursts of Customer Sessions," to appear in IEEE TSE.

[4] B. Urgaonkar, P.Shenoy, A. Chandra,P. Goyal, "Dynamic Provisioning of Multi-tier Internet Applications," ICAC 2005, pp. 217-228.

[5] J. Li, J. Chinneck, M. Woodside, Litoiu, "Fast scalable optimization to configure service systems having cost and quality of service constraints," ICAC 2009, pp. 159-168.

[6] C. Clark, K. Fraser, S. Hand, J. Hansen, E. Jul, C. Limpach, I. Pratt, A. Warfield, "Live migration of virtual machines," NSDI 2005, pp 273-286.

[7] C. Hyser, B. Mckee, R. Gardner, B. Watson, "Autonomic virtual machine placement in the data center," HP Labs, HPL-2007-189, 2007.

[8] X. Zhu *et al.*, "1000 Islands: Integrated Capacity and Workload Management for the Next Generation Data Center," ICAC 2008, pp. 172-181 .

[9] J. Rolia, L. Cherkasova, M. Arlitt, A. Andrzejak, "A capacity management service for resource pools," WOSP 2005, pp. 229-237.

[10] G. Jung, K. Joshi, M. Hiltunen , R. Schlichting, C. Pu, "Generating Adaptation Policies for Multi-tier Applications in Consolidated Server Environments," ICAC 2008, pp. 23-32.

[11] S. Mylavarapu, V. Sukthankar, P. Banerjee, "An optimized capacity planning approach for virtual infrastructure exhibiting stochastic workload", SAC 2010, pp. 386-390.

[12] J. Rolia and K. Sevcik, "The method of layers," IEEE TSE, vol. 21, no. 8, pp. 689-700, August 1995.

[13] G. Casale, N. Mi, L. Cherkasova, E. Smirni, "How to parameterize models with bursty workloads", SIGMETRICS Perform. Eval. Rev. 36, 2 (Aug. 2008), pp.38-44.