Cloud Overbooking Through Stochastic Admission Controller

Merve Unuvar, Yurdaer N. Doganata, Asser N. Tantawi and Malgorzata Steinder

IBM T. J. Watson Research Center
Yorktown Heights, NY 10598
{munuvar,yurdaer,tantawi,steinder@us.ibm.com}

***Abstract*—Cloud providers apply overbooking to increase utilization of data center resources, which is associated with the risk of overloading cloud resources. In this paper, we study an admission control technique that permits cloud overbooking with bounded probability of resource over-utilization. The objective is to achieve a specified quality-of-service related to the probability of resource over-utilization in an uncertain loading condition, while maintaining high resource utilizations. Our method relies on estimating the probability of over-utilization based on approximating the probability distribution of the total resource demand on hosts as Beta. We perform a qualitative study to investigate the efficiency of using our method on Google Compute Cluster where we disclose some empirical observations on how well the resource utilization can be estimated as Beta Distribution. We also report results on the performance of the stochastic admission controller by estimating the mean and the standard deviation of the usage data.**

***Keywords*—admission control, cloud management, dynamic demand, performance comparison, policies, virtual machines**

I. Introduction

In a Cloud system, managing the utilization of physical resources with effective admission control policies is essential. Admission control policies ensure that sufficient resources are available in a cluster to provide fail-over protection and to ensure that virtual machine resource reservations are respected [1].

The most commonly used admission policy admits a virtual machine (VM) into the Cloud if its full resource request can be reserved on some physical machine (PM) in the Cloud. As VM resource requests often greatly exceed the actual VM time-varying demand, such policy leads to Cloud resources being under-utilized. Cloud providers want to take advantage of that unused capacity by resource overbooking. Overbooking allows the sum of virtual machine resource requests to exceed the capacity of the Cloud. The choice of placing a VM on a particular PM is a decision of the Placement Controller, which is not the subject of this paper. The placement decision usually involves many factors, including load balancing, reliability, availability, and networking. Here we are concerned with the decision to admit a VM once its suggested host PM is made.

Overbooking carries with it the risk that the actual demand of VMs placed on a PM will exceed that PM's usable capacity leading to VM interruption or poor performance. Admission control policies that permit overbooking must therefore be selected such that the risk of PM overload is minimized.

Each resource is described by the stochastic properties of its utilization. The probability density function (pdf) of the utilization of a resource is the convolution of all resource demands of the accepted requests that utilize that resource. In such aggregation of independent resource demands, the probability that the aggregate utilization will reach the sum of the peak demand is infinitesimally small. Using the pdf of the aggregated resource utilization in admission criteria provides for probabilistic guarantees. In mathematical terms, resource k is stable if its utilization, U_k, satisfies the following constraint,

$$P(U_k > U_k^o) \leq \epsilon^o \tag{1}$$

where U^o is the over-utilization threshold and ϵ^o is the probabilistic bound on over-utilization.

In this study, we approximate the pdf of the aggregated resource utilization using the first and second moments as a Beta distribution. Then, we employ (1) as the admission criterion to decide if the statistical properties of an arriving *VM* request will likely to drive the physical machine into over-utilization. Thus, we enforce an admission criterion that guarantees a bound on the probability of over-utilization. We evaluate our technique using Google Compute Cluster usage data. The Google Cluster data discloses resource usage by millions of tasks running on a set of machines hosted in racks and connected by a high bandwidth network. Our aim is to verify the Beta Distribution assumption by aggregating usage of different number of tasks and check if we can fit a Beta Distribution to the actual resource utilization distribution. We also compare the over-utilization probability that we estimate using the Beta assumption with the actual resource over-utilization in Google Cluster data.

The paper is organized as follows. We introduce a mathematical representation for the arriving resource requests and the associated distribution for the resource demand in section II. The details of stochastic admission controller is presented in section III. Our numerical results for Google Compute Cluster are presented in section IV. We summarize the conclusion and future work in section V.

II. Formulation

Consider p homogenous *PMs* subjected to a stream of homogenous requests with a Poisson arrival process with rate λ and a generally distributed lifetime with mean τ. A *PM* has K different resource types, each having capacity C_k, $k = 1, 2, \cdots, K$. A request has a demand D_k for resource k that is generally distributed with distribution function

$F_{D_k}(d_k) = Pr[D_k \leq d_k]$, where $d_k \in [D_k^{min}, D_k^{max}]$. Without loss of generality we assume that $D_k^{min} = 0$ and $D_k^{max} > 0$. We denote the mean and standard deviation of the demand for resource k by μ_{D_k} and σ_{D_k}, respectively. Hence, the mean offered load for the k^{th} resource is given by

$$\rho_k = \lambda\, \tau\, \mu_{D_k}. \tag{2}$$

Let Z_k^n denote the sum of n independent k^{th} resource demands, i.e. $Z_k^n = nD_k$. Thus, the mean of Z_k^n is $E[Z_k^n] = n\, \mu_{D_k}$, the variance is $V[Z_k^n] = n\, {\sigma_{D_k}}^2$, and the probability distribution, denoted by $F_{Z_k^n}(z_k)$, is the n-fold convolution.

III. Stochastic Admission Controller

An admission controller admits a request into the Cloud based on some policy $\mathbb{P}(\overline{\phi})$, with a set of parameters $\overline{\phi}$ used in admission criteria.

A request is admitted if the admission policy allows it given the current state of the system, otherwise it is rejected. The resulting request rejection probability δ, and the mean utilization of k^{th} resource $\overline{U_k}$ are related as

$$\overline{U_k} = \frac{(1-\delta)\, \rho_k}{p\, C_k} \tag{3}$$

where ρ_k is the offered load for the resource k and C_k is the capacity for the resource k. Admission based on a probabilistic bound over-utilization can be denoted by $\mathbb{P}(U_k^*, \epsilon, \mu_k, \sigma_k)$, where U_k^* is the utilization threshold, ϵ_k is the probabilistic bound on the over-utilization probability for resource k such that the probability of over-utilization being above U_k^* is limited to ϵ_k, and μ_k and σ_k are the mean and standard deviation of the utilization for resource k, respectively.

In this probabilistic admission policy, the dynamic nature of resource demand is represented by its mean, μ_{D_k}, and standard deviation, σ_{D_k}. The utilization of resource k, U_k, is a random variable between [0,1]. It is characterized by its mean and standard deviation, μ_k and σ_k, respectively.

The admission criterion is given by

$$F_{Z_k^n}(U_k^*) \geq (1-\epsilon_k), \tag{4}$$

otherwise the request is rejected.

The key to implementing the stochastic admission controller is thus the knowledge of $F_{Z_k^n}(U_k^*)$. We approximate the probability distribution function (pdf) of U_k as a Beta distribution. The Beta Distribution is a family of continuous probability distributions defined on the interval [0,1] by two positive shape parameters, denoted by α and β. Hence, we characterize the utilization U_k with two parameters, α_k and β_k, associated with the first and second moments of U_k. The values of α_k and β_k are computed from the estimated mean and variance values of the utilization of resource k in the *PM* as:

$$\alpha_k = \bar{R}_k \left(\frac{\bar{R}_k(1-\bar{R}_k)}{\bar{S}_k^2} - 1 \right), \tag{5}$$

$$\beta_k = (1-\bar{R}_k) \left(\frac{\bar{R}_k(1-\bar{R}_k)}{\bar{S}_k^2} - 1 \right), \tag{6}$$

where $\bar{R}$ and $\bar{S}$ are estimates of μ_k and σ_k, respectively. Hence the cumulative distribution function $F_{Z_k^n}(U_k)$ for the utilization of resource k is expressed as:

$$F_{Z_k^n}(U_k, \alpha_k, \beta_k) = B(U_k, \alpha_k, \beta_k)/B(\alpha_k, \beta_k), \tag{7}$$

where B is the *Beta* function. $\mathbb{P}$ does not need to be adjusted as the workload characteristics change, since it is dynamically adjusted with the measured statistical properties of resource utilization. The predefined parameters U_k^* and ϵ_k are set by the Cloud manager depending on the specifications of the physical machine. More detailed theoretical explanation can be found in [2].

IV. Numerical Results

We use Google Compute Cluster data, which is available online[1], to support our assumption on representing resource utilization by Beta Distribution, and to estimate the mean and the standard deviation of real workloads to experiment the performance of stochastic admission controller. The detailed explanation on Google Compute Cluster data can be found in [3].

A. Fitting sample usage data to beta distribution

In this section, we explain the method of calculating the CPU utilization from the sample usage data and using this utilization data to fit a Beta Distribution. In order to calculate the utilization of a resource, the machine resource capacities need to be known exactly. Since Google did not publish the exact machine capacities but published the normalized values, we cannot use that information directly to calculate the resource utilization. Rather, we take the maximum resource usage throughout the 29 days of time span and set that level of utilization to be 90% for the CPU utilization. Google reports that not all of the resource capacity is available to the tasks therefore 10% of the machine is assumed to be reserved for the cluster scheduler and operating system [4]. For example, for the same sample (average of 17 jobs involving 18 tasks running on machine ID= "351618647"), the maximum CPU usage over 29 days is 3.97 core-hours. We set the usage of 3.97 core-hours to be 90% utilization and normalize the rest of the usage data by dividing to $3.97/0.9$. Since actual resource consumption is done by the tasks, we ony use the number of tasks rather than number of jobs through the rest of the paper.

Next, we calculate the shape parameters for Beta Distribution from the mean and the standard deviation of the sample as described in section III. The mean CPU utilization and the standard deviation for the 500 sample are 0.49 and 0.14 respectively. By substituting the calculated mean and the standard deviation to Equation 5 and 6, we estimate Beta Distribution shape parameters as follows: $\alpha = 5.76, \beta = 5.93$. Figure 1 depicts the Q-Q plot showing how well this estimation is, based on our sample. Q-Q plot is a probability graph that graphically compares two probability distributions by plotting their quantiles against each other, [5]. If the two distributions being compared are similar, the plotted points in the Q-Q plot will approximately lie on $y = x$ line. As Figure 1 shows, the estimation of the parameters are fitting to $y = x$ line indicating that the estimated parameters are well chosen.

[1] http://code.google.com/p/googleclusterdata/

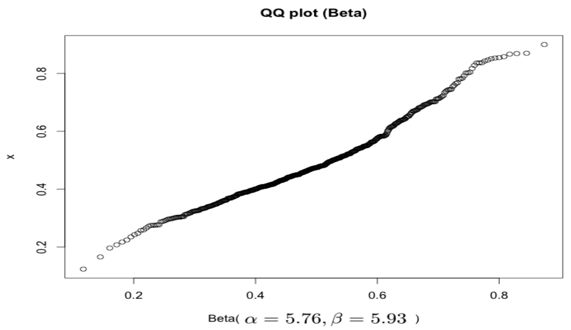


Fig. 1: CPU Utilization for average of 18 tasks fit to Beta Distribution parameters

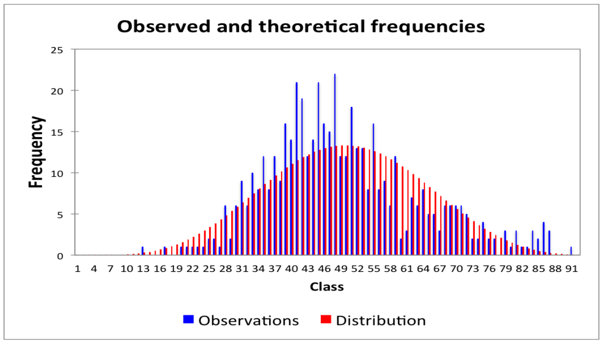


Fig. 2: Histogram of actual CPU Utilization for average of 18 tasks and fitted Beta Distribution

We use Kolmogrov-Smirnov test to compare the fit of actual CPU utilization sample to estimated Beta Distribution sample. The K-S test is one of the most useful and general nonparametric methods for comparing the empirical distribution functions, [7]. K-S test measures a distance between these empirical distributions by drawing samples from the same null distribution of the desired statistics, defined in null hypothesis. We state our null and alternative hypothesis as:

$$H_0 \quad : \quad \text{The sample follows a Beta Distribution}$$
$$H_a \quad : \quad \text{The sample does not follow a Beta Distribution.}$$

K-S test result is as follows: D value is 0.084, $p-value$ is 0.002 and the significance level, α is 0.05. As the computed $p-value$ is lower than the significance level $\alpha = 0.05$, one should reject the null hypothesis H_0, and accept the alternative hypothesis H_a for this particular test. The risk to reject the null hypothesis H_0 while it is true is lower than 0.05%. This result indicates that Beta Distribution is not a good fit for the average of 18 tasks worth of usage data. This may be caused by the lack of representative usage sample. In other words, the convolution of the usage distribution of 18 tasks did not fit to a Beta Distribution. In order to minimize the uncertainty caused by the smaller subsets, we increase our task sample size to 49, 88 and 154. We show that as we sample the usage data from more tasks uniformly, the resource utilization fits to Beta Distribution better. The results of larger samples are described in the next section.

B. *Effect of sample size on Beta fitness test*

Even though the utilization distribution for 18 task usage on average graphically looked like an "almost" good fit to Beta Distribution, the fit, however, did not pass the K-S test at 0.05 significance level. We believe this is caused by the random selection of 18 tasks, which was not representative enough for the general workload to convolute as Beta Distribution. In this section, we perform the Beta fit method described above on 3 different samples separately. We randomly, on average, first select 49 then 88 and finally 154 tasks over 29 days to see if higher sample sizes would fit utilization to Beta Distribution better. We aggregate the usage of the tasks at each day over 29 days for every sample. We again set the maximum usage level over 29 days to be 90% resource utilization to leave 10% of the capacity for cluster management & operating systems.

By using the calculated mean and standard deviation values for each samples, we estimate the shape parameters. The estimation of shape parameters for samples of 49, 88 and 154 tasks is more accurate than 18 task sample. Table I lists the estimated and actual values for mean, variance, skewness and kurtosis for 3 samples.

TABLE I: Statistics estimated on the input data and computed using the estimated parameters of the Beta Distribution

Experiment	Statistics	Data	Parameters
49 tasks	Mean	0.500	0.495
	Variance	0.029	0.028
	Skewness (Pearson)	0.202	0.012
	Kurtosis (Pearson)	-0.674	-0.543
88 tasks	Mean	0.504	0.490
	Variance	0.023	0.023
	Skewness (Pearson)	0.340	0.021
	Kurtosis (Pearson)	-0.415	-0.461
154 tasks	Mean	0.516	0.500
	Variance	0.020	0.020
	Skewness (Pearson)	0.374	0.000
	Kurtosis (Pearson)	-0.274	-0.416

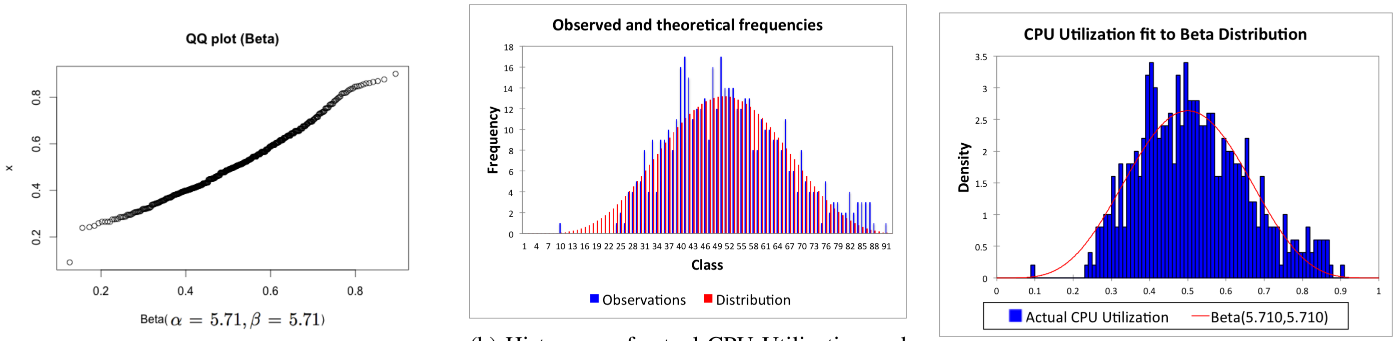
Figure 3 shows the Q-Q plot, distribution of the actual CPU utilization and the estimated Beta Distribution for 154 samples. As Figure 3 shows, the Q-Q plot is almost on $y = x$ line, indicating that the shape parameter estimations are very accurate for the largest sample size. Further, Figure 3 depicts the actual data distribution and the Beta Distribution in the form of histogram.

Even though the Figure 3 supports the Beta Distribution fit graphically, we still perform the K-S test to see how well the Beta Distribution fit to our samples. Table II shows the results for the sample of 49, 88 and 154 tasks. K-S supports that the fit gets better as the sample size increases.

TABLE II: Kolmogorov- Smirnov Test

Test Result	49 tasks	88 tasks	154 tasks
D	0.055	0.051	0.048
$p-value$	0.098	0.150	0.196
α	0.05	0.05	0.05

The above analysis showed that the quality with which the Beta distribution fits the distribution of U_k depends on the number of request random variables included in the comvolution. In practical terms, this means that in a system where



(a) CPU Utilization fit to Beta Distribution parameters for average of 154 tasks

(b) Histogram of actual CPU Utilization and fitted Beta Distribution for average of 154 tasks

(c) CPU Utilization fit to Beta Distribution for average of 154 tasks

Fig. 3: Q-Q plot, CPU utilization & Beta Distribution fit and observed & theoretical utilization for 29 days for usage of 49, 88 and 154 tasks

a physical compute node can host a large number of tasks simultaneously, the Beta distribution is a good representation of the utilization generated by the comvolution of these tasks.

C. Estimation Error for Stochastic Admission Controller

From the fitted distribution in Figure 3, one can calculate the over-utilization probability. For example, if the over-utilization threshold is set to 0.8 by the user, the probability of over-utilization -regardless of the admission controller-is estimated as $P(U_{CPU} \geq 0.8) = 0.024$. The stochastic admission controller bounds the over-utilization probability to a user selected parameter, the probabilistic bound, by rejecting the arrivals that would potentially increase the likelihood of over-utilization beyond the selected parameter.

As an example, with estimated Beta Distribution for the usage data, stochastic admission controller ensures that the probability of over-utilization (over-utilization being 0.8) as 2.4%. This can be verified by equation (4). The actual distribution of CPU, however, is higher than the over-utilization threshold 4.4% of the time. Hence, the stochastic admission controller under-estimates the over-utilization by 2.0% for the 154 task sample. Table III shows that as the beta fit gets better (higher $p - value$ in K-S test), the error in estimating over-utilization probability decreases.

TABLE III: Error in estimating $P(U_{CPU} \geq 0.8)$

Sample size	$p - value$	Actual prob.	Estimated prob.	Error
18 tasks	0.002	0.066	0.026	0.040
49 tasks	0.096	0.054	0.031	0.023
88 tasks	0.15	0.042	0.016	0.028
154 tasks	0.196	0.044	0.024	0.020

This result shows that we can expect considerable improvement in over-utilization probability, compared to a system with no controller, the the orver-utilization probability will remain higher that the theoretical goal. On the other hand, the controller is unlikely to reject workloads unnecessarily.

We implemented the stochastic admission controller on Google Compute Cluster. We detailed the advantages of using a stochastic admission controller and the percent of time when the system over-utilization could be avoided if the admission controller was used in Google Compute Cluster in [3].

V. Conclusion and Future Work

There are certain conveniences in assuming that the aggregated resource usage distribution in a Cloud machine follows a Beta Distribution. Beta is a family of distributions with two parameters where the parameters are associated with the first and the second moments of the resource usage. Thus, it covers a wide range of usage distribution possibilities.The stochastic admission control policy that we introduce in section III bounds the specified over-utilization probability to a user selected level.

Our experiment with Google Cluster data showed that the resource distribution consistently fits to Beta Distribution when the number of tasks running on a single machine is above 50 and the stochastic admission controller bounds the over-utilization probability better. Our future work will focus on statistical analysis of the Google Cluster data to determine the conditions around the number of tasks running in a single machine to justify the Beta assumption.

References

[1] Vmware vsphere ® high availability 5.0 deployment best practices. [Online]. Available: www.vmware.com/files/pdf/techpaper/vmw-vsphere-high-availability.pdf

[2] M. Unuvar, N. Y. Doganata, and A. N. Tantawi, "Configuring cloud admission policies under dynamic demand," Modeling, Analysis, Simulation of Computer and Telecommunication Systems (MASCOTS), 2013 IEEE 21st International Symposium on. IEEE, 2013.

[3] M. Unuvar, Y. N. Doganata, A. N. Tantawi, and M. Steinder, "Cloud overbooking through stochastic admission controller," IBM T. J. Watson Research Center, 1101 Kitchawan Rd. Yorktown Heights, NY, Tech. Rep. RC25469, May 2014.

[4] Google, "https://docs.google.com/file/d/0b5g07t_grdg9njznsjztzzrfbmm/edit."

[5] G. Blom, *Statistical estimates and transformed beta variables,*. John Wiley and Sons, 1958.

[6] H. Pham, *Handbook of Engineering Statistics*. Springer, 2006.

[7] N. Smirnov, "Table for estimating the goodness of fit of empirical distributions," *Annals of Mathematical Statistics*, vol. 19, pp. 279–281, 1948.