Randomized routing of virtual machines in IaaS data centers

Hadi Khani [Corresp., 1] , **Hamed Khanmirza** [2]

1 Department of Engineering, Islamic Azad University Garmsar Branch, Garmsar, Semnan, Iran

2 Department of Computer Engineering, K. N. Toosi University of Technology, Tehran, Tehran, Iran

Corresponding Author: Hadi Khani
Email address: hkhani@ut.ac.ir

The cloud computing technology has been a change player in recent years. The cloud computing providers promise cost-effective and on demand resource computing for their users. The cloud computing providers are running the workloads of users as virtual machines in a large scale data center consisting a few thousands physical servers. The cloud data centers face highly dynamic, varying over time and many short tasks that demand quick resource management decisions. These data centers are large scale and the behavior of workload is unpredictable. The incoming virtual machine must be assigned onto the proper physical machine in order to keep a balance between power consumption and quality of service. The scale and agility of cloud computing data centers are unprecedented so the previous approaches are fruitless. We suggest an analytical model for cloud computing data center when the number of physical machines in the data center is large. In particular, we focus on the assignment of virtual machine onto physical machines regardless of their current load. For exponential virtual machine arrival with general distribution sojourn time, the mean power consumption is calculated. Then we show the minimum power consumption under quality of service constraint will be achieved with randomize assignment of incoming virtual machines onto physical machines. Extensive simulation supports the validity of our analytical model.

Randomized Routing of Virtual Machines in IaaS Data Centers

Hadi Khani[1], Hamed Khanmirza[2]

[1] CE Department, Islamic Azad University- Garmsar Branch, Garmsar, Semnan, Iran

[2] CE Department, K. N. Toosi University of Technology, Tehran, Tehran, Iran

Corresponding Author:

Hadi Khani[1]

Entefaze Street, Garmsar, Semnan, 3581631167, Iran

Email address: hkhani@ut.ac.ir

Randomized Routing of Virtual Machines in IaaS Data Centers

Hadi Khani[1] **and Hamed Khanmirza**[2]

[1]**Faculty of Engineering,Islamic Azad University- Garmsar Branch, Garmsar, Iran**
[2]**Faculty of Computer Engineering, K. N. Toosi University of Technolog, Tehran, Iran**

Corresponding author:
Hadi Khani[1]

Email address: hkhani@ut.ac.ir

ABSTRACT

The cloud computing technology has been a change player in recent years. The cloud computing providers promise cost-effective and on demand resource computing for their users. The cloud computing providers are running the workloads of users as virtual machines in a large scale data center consisting a few thousands physical servers. The cloud data centers face highly dynamic, varying over time and many short tasks that demand quick resource management decisions. These data centers are large scale and the behavior of workload is unpredictable. The incoming virtual machine must be assigned onto the proper physical machine in order to keep a balance between power consumption and quality of service. The scale and agility of cloud computing data centers are unprecedented so the previous approaches are fruitless. We suggest an analytical model for cloud computing data center when the number of physical machines in the data center is large. In particular, we focus on the assignment of virtual machine onto physical machines regardless of their current load. For exponential virtual machine arrival with general distribution sojourn time, the mean power consumption is calculated. Then we show the minimum power consumption under quality of service constraint will be achieved with randomize assignment of incoming virtual machines onto physical machines. Extensive simulation supports the validity of our analytical model.

INTRODUCTION

Infrastructure-as-a-Service (IaaS) Cloud Providers (CP), such as Amazon, Google and Microsoft, have huge datacenters to provide on demand Virtual Machines (VM) to their customers. An important issue in such data centers is to determine the server to which an incoming VM should be placed in order to optimize a given performance criterion. The CP has a variety of challenges, such as higher resource utilization, less cooling expenses and lower operation expenses. Fortunately, all of these efficiency metrics are positively correlated. The less power consumption means less operational expense, less cooling bills and higher utilization in the data center. This let us choose the power consumption as the key metric representing others. On the other hand cloud users who run their application on VMs have their own concerns, Quality of Service. The resource management of CP has the chance to revise the initial placement of VMs onto PMs by live migrating techniques or dynamic consolidation. Considering live migration, the problem of VM placement can be divided in two parts as pictured in figure 1.

1. The routing of arriving VMs onto PMs
2. The optimization of the current VM placement by VM migration

According to this scenario, a VM request transmitted by a user to the data center, is routed to the proper PM in the first step and then its placement can be optimized later.

The optimization of the current VM placement in a data center is analogous to the NP-hard "Bin Packing Problem". In this problem, a given set of items of variable size should assigned to the minimum number of bins taken from a given set. The VMs experience dynamic workloads, which means that the resource usage by a VM arbitrarily varies over time. In fact, the data center resource manager does not have the complete knowledge of the future resource usage (size) of VMs. The placement of VMs is
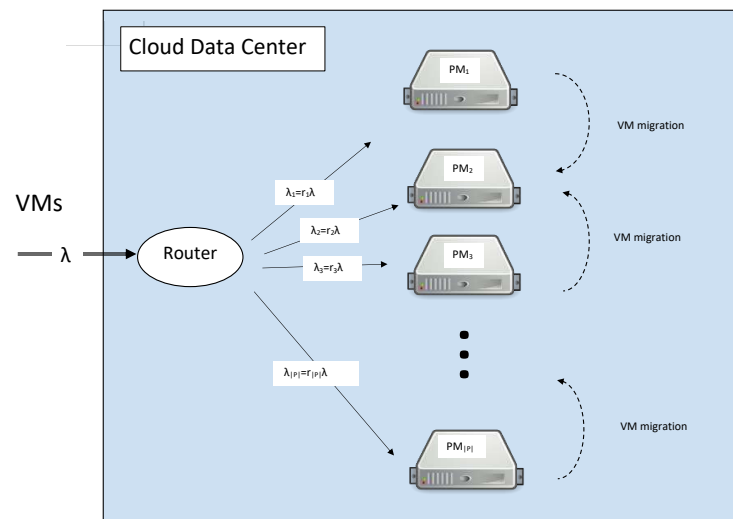
Figure 1. Randomized Router

monitored continuously and is tuned through the migration procedure. The virtualization technology let VM migrates (moves) between PMs on the fly. The migration of VM can be advantageous either when the resources utilization is too low, meaning that the PM is highly underutilized, or when it is too high, possibly causing overload situations and service level agreement violations. The optimization problem of VM placement problem is so complex that centralized and deterministic solutions are practically useless in large data centers with hundreds or thousands of servers as shown in several researches like (Wang and Gelenbe, 2015; Shojafar et al., 2016; Wang et al., 2016). The Centralized and deterministic algorithms may be appropriate in data centers with a limited number of servers, but may become inefficient in large and very large data centers, due to the complexity of the problem and the need for the simultaneous migrations of a large number of VMs.

These decentralized approaches have some side effect on routing procedure in the first part. The router does not have complete knowledge of the current placement of cloud data center. It is noteworthy that the router neither does not know about the size, the exact arriving time and sojourn time of the future VMs. These facts justify the stochastic modeling and analyzing of the router performance.

In this paper, we focus on the first problem- the problem of routing arriving VM to the host. We calculate probability of Service Level Agreement Violation (SLAVs) as well as total power consumption in a cloud data center using tools of queueing theory. A cloud data center differs from traditional queueing systems. First, a cloud data can have a large number of PMs; traditional queueing analysis rarely consider system of this size. Second, VM sojourn time must be modeled by general distribution instead of convenient exponential distribution. These differences pose significant challenges to the analysis. We use a novel approach to respond to these challenges. Our contributions in this paper are:

1. We model the cloud data centers as a group of $M/G/n/n$ queuing systems with single task arrivals and a task buffer of finite capacity
2. We define a novel optimization problem to minimize the power consumption under an explicit QoS goal for any VM consolidation system
3. We find the optimal routing policy using numerical methods

Analytical results are validated through discrete event simulation. Then we compare our result with some benchmark algorithm for Google workload. The remainder of the paper is organized as follows. Section 2 gives an overview of existing work on cloud performance evaluation and performance characterization of $M/G/n/n+r$ queuing systems. It also introduces some heuristic algorithms for VM

consolidation that we use for comparison. Section 3 discusses our analytical model in detail. We solve our optimization problem in order to obtain desired performance metrics in section 4. In section 5, we present and discuss analytical as well as simulation results. We conclude the paper with Section 6 discussing the results and future research directions.

RELATED WORK

Prior approaches to VM placement in the literature can be broadly divided into two categories: rigorous analytical approach and heuristic algorithms. One of the first works on analysis of performance issues in VM placement has been performed by Yang et al (Yang et al., 2009). They obtained the distribution of response time for a cloud data center modeled as an $M/M/n+r$ queueing system. They assumed both interarrival and service times are exponentially distributed, and the system has finite buffer of size n+r. The response time was broken down into waiting, service, and execution periods, assuming that all three periods are independent which is unrealistic.

By relaxing the assumption that the service times are not exponential, one can construct an accurate and close-to-reality model at the expense of greater complexity in the analysis. Most theoretical analyses have relied on extensive research in performance evaluation of M/G/n queuing systems (Ma and Mark, 1995; Miyazawa, 1986; Yao, 1985). However, the probability distributions of response time and queue length in $M/G/n$ and $M/G/n+r$ cannot be obtained in closed form, which necessitates the search for a suitable approximation. An approximate solution for steady-state queue length distribution in an $M/G/n$ system with finite waiting space is described in (Kimura, 1996). The proposed approach is exact for $M/G/n/n+r$ when $r=0$.

A similar approach for $M/G/m$ queues is proposed in (Kimura, 1983). In this work, analysis is extended to compute the blocking probability and, thus, determines the smallest buffer capacity such that the rate of lost tasks remains below a predefined level. In (Nozaki and Ross, 1978),], another approximation for the average queuing delay in an $M/G/n/n+r$ queue was proposed. The approximation is based on the relationship of joint distribution of remaining service time to the equilibrium service distribution. Another approximation for the blocking probability is based on the exact solution for finite capacity $M/M/n/n+r$ queues (Smith, 2003). Again, the estimate of the blocking probability is used to guide the allocation of buffers so that the blocking probability remains below a specific threshold.

Most of above findings rely on some approximations. Approximations are reasonably accurate only when the number of servers is comparatively small, typically below 10 or so. In addition, approximation errors are high when the traffic intensity is small as stated in (Boxma et al., 1979; Kimura, 1983; Tijms et al., 1981). As a result, we cannot apply the above results directly for performance analysis of CP data center when one or more of the following holds: the number of servers is very large or the distribution of service times is unknown and does not follow any of the "well-behaved" probability distributions such as exponential distribution.

As we use $M/G/n/n$ queueing system to model a physical machine not the whole data center, our analysis is suitable for performance analysis of cloud scale data centers. In addition, we study $M/G/n/n$ in steady station setting. Kimura has proposed an exact closed form for queue length distribution in an $M/G/n/n$. In this paper, we use this closed form in defining optimization problem (Kimura, 1996) which let us apply numerical computation for analyzing the performance of the whole data center in next step.

In order to compare the performance of randomized router in practice, we have chosen two algorithms from heuristic algorithms as benchmark: Power Aware Best Fit Decreasing (PABFD) (Beloglazov and Buyya, 2012) and Modified Throttled (MT) (Domanal and Reddy, 2013).

As mentioned before, the VM placement can be seen as a bin packing problem with variable bin sizes and prices, where bins represent the physical machines (PMs); items are the VMs that have to be allocated; bin sizes are the available CPU capacities of the PMs; and prices correspond to the power consumption by the nodes. As the bin packing problem is NP-hard, to solve it Beloglazov and Buya apply a modification of the Best Fit Decreasing (BFD) algorithm that is shown to use no more than $\frac{11}{9}OPT + 1$ bins (where OPT is the number of bins provided by the optimal solution) [yue1991simple].In PABFD, they sort all the VMs in the decreasing order of their current CPU utilizations and allocate each VM to a host that provides the least increase of the power consumption caused by the allocation. In each round of PABFD all VMs are placed again. The number of VM migrations skyrockets in PABFD and it is not practical in a real large scale data center.

129   Modified throttled (MT) algorithm maintains an index table of PMs and also the state of PMs (Domanal
130 and Reddy, 2013). There has been an attempt made to improve the response time and achieve efficient
131 usage of available PMs. Proposed algorithm employs a method for selecting a machine for hosting
132 arriving VM of user where, machine at first index is initially selected depending upon the state of the
133 machine. If the machine is available, it is assigned with the request and id of machine is returned to data
134 center resource manager, else -1 is returned. When the next request arrives, the machine at the index
135 next to already assigned machine is chosen depending on the state of machine and follows the above
136 step. This method needs to keep an updated index table of state of machines. In large data centers this
137 task is not trivial, in particular when you taking into account the decentralized consolidation of VMs.
138 It is important that both MT and PABFD is not practical in real scenario, and here we use them just as
139 idealistic benchmark algorithms.

SYSTEM MODEL

141 Consider a IaaS data center consisting of a set of $|P|$ PMs. Let $P = \{1, 2, ..., |P|\}$ denote index set of the
142 set of PMs. Users request VMs to the provider. New request is either admitted or rejected in the admission
143 control phase. An admitted request moves to the placement phase, where it will be assigned to one of
144 PMs ((Carvalho et al., 2015)). We suggest a randomized router after the admission. The router sends
145 incoming VMs to PM i with probability r_i, for all $i \in P$. The vector $\vec{r} = \{r_1, r_2, ..., r_{|P|}\}$ is a probability
146 vector and satisfies $\sum_{i \in P} r_i = 1$. VMs have independent and identically distributed service (sojourn) time
147 with mean $1/\mu$. In addition, these sojourn times are independent of the host load. Assume that the the
148 utilization demand of all VMs is equal to one. Extensive analysis of huge data centers shows that majority
149 of the VMs have approximately the same utilization ((Reiss et al., 2011)). This observation supports our
150 assumption. Assume that each PM only hosts at most n VMs. In the case a VM is assigned to an already
151 full PM, the PM is overloaded. An overloaded PM degrades the QoS of the end user and we can assume
152 this event as an SLAV ((Domanal and Reddy, 2013) and (Beloglazov and Buyya, 2012)). Remind that
153 the router is after admission control and admitted VMs could not be queued. All VMs arrive at the data
154 center according to Poisson process with rate λ, thereby VMs arrive at PM i according to Poisson process
155 with rate $\lambda_i = \lambda r_i$, for all $i \in P$ ((Wang et al., 2015)). These processes are independent (see Sec. 6.4 in
156 (Trivedi, 2001)). The whole data center can be modeled as a group of independent $M/G/n/n$ (also known
157 as Generalized Erlang Loss System) systems that work in parallel.

158 **Generalized Erlang Loss System**

It is well known that for an $M/G/n/n$ system, the steady-state distribution exists and is given by the
'product form' as below ((Kelly, 1991) and (Kimura, 1983))

$$\pi_k = \begin{cases} \pi_0 \left(\frac{\lambda}{\mu}\right)^k \frac{1}{k!}, & \text{for } k \leq n \\ 0, & \text{for } k > n \end{cases} \tag{1}$$

where π_k denotes the steady-state probability that there are k VMs in an $M/G/n/n$ system, that is
$\pi_k = \lim_{t \to \infty} \pi_k(t)$, and the steady-state probability that $M/G/n/n$ is empty (π_0) is given by

$$\pi_0 = \left[\sum_{k=0}^{n} (\frac{\lambda}{\mu})^k \frac{1}{k!}\right]^{-1} \tag{2}$$

Specifically, π_n describes the fraction of time that the PM is fully utilized. We call this probability
SLAV and it is given by

$$\pi_n = \frac{\frac{(\lambda/\mu)^n}{n!}}{\sum_{k=0}^{n} \frac{(\lambda/\mu)^k}{k!}} \tag{3}$$
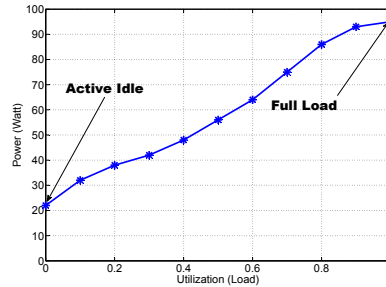
Figure 2. Linear Power Consumption

Power Consumption

We are interested in minimizing the total power consumption of the data center. According to the results of the standard experiments stated in (spec.org, 2018) (Figure2) , the instantaneous power consumption of PM i is a function of utilization level of that PM (k) as below

$$W_i = \begin{cases} a + bk, & \text{for } k > 0 \\ 0, & \text{for } k = 0 \end{cases} \tag{4}\tag{5}$$

where $0 \leq k \leq 100$ is integer variable and both a and b are fixed values. In our model, k can be seen as the number of VMs in PM i. Then the expected steady state of power consumption will be

$$Exp[W_i] \quad = \sum_{k=1}^{100} W_{i,k}\pi_{i,k} \tag{6}$$

$$= a(1 - \pi_{i,0}) + b\sum_{k=1}^{100} k\pi_{i,k} \tag{7}$$

where $\pi_{i,k}$ denotes the steady-state probability that the utilization of PM i is k and can be calculated by (1) for $n = 100$. Note that $Exp[W_i] = 0$ if and only if $\lambda_i = 0$ ($\pi_{i,0} = 1$). Our objective is to determine the vector $\vec{r}$ that minimizes the total expected steady-state power consumption of the data center, that is,

$$\min_{\vec{r}} \quad \sum_{i \in P} Exp[W_i] \tag{8}$$

SLA Constraint

We are interested in keeping the probability of SLAV below a given value ε. SLAV happens when a PM i does not have sufficient capacity for a new arriving VM (when M/G/100/100 is in state 100). The SLAV constraint is

$$\Pr(SLAV) = \pi_{i,100} \leq \varepsilon \qquad \forall i \in P \tag{9}$$

where $\pi_{i,100}$ denotes the steady-state blocking probability for PM i.

OPTIMIZATION PROBLEM

In this section we consider the optimization problem, in which the router decides where each incoming VM will be sent, so as to minimize the total expected power consumption subject to the SLA constraints. The optimization problem can be formulated as follows:

$$\min_{\vec{r}} \quad \sum_{i \in P} Exp[W_i] \tag{10}$$

$$\text{s.t.} \quad \sum_{i \in P} r_i = 1 \tag{11}$$

$$\pi_{i,100} \leq \varepsilon \quad \forall i \in P \tag{12}$$

$$\lambda_i = \lambda r_i \quad \forall i \in P \tag{13}$$

Let us rewrite the optimization problem by changing our *optimization variable* from $\vec{r} = \{r_1, r_2, ..., r_{|P|}\}$ to $\vec{x} = \{x_1, x_2, ..., x_{|P|}\}$ which is defined below

$$x_i = \frac{\lambda_i}{\mu} = \frac{\lambda r_i}{\mu} \tag{14}$$

Using (14) and putting (1) for $k = 100$ in (12) we get

$$\frac{x_i^{100}}{100! \sum_{k=0}^{100} \frac{x_i^k}{k!}} \leq \varepsilon \tag{15}$$

If we can solve this inequality constraint for x_i then we have a simple inequality constraint in the form of $x_i < f(\varepsilon)$. Fortunately, numerical methods can be used to solve it. We show numerical results β for some practical values of ε in Table 1. For example in the first row, we have $\beta = 100$ for $\varepsilon = 0.1$. It means that if we send VMs with a rate below 100 ($x_i < \beta = 100$) to PM i then we will guarantee that the probability of SLAV is below 0.1 ($Pr(SLAV) < \varepsilon = 0.1$) for that PM. The equivalent optimization problem will be

$$\min_{\vec{x}} \quad f_o(\vec{x}) = \sum_{i \in P} \gamma(x_i) \tag{16}$$

$$\text{s.t.} \quad \sum_{i \in P} x_i = \frac{\lambda}{\mu} \tag{17}$$

$$x_i \leq \beta \quad \forall i \in P \tag{18}$$

SLAV probability (ε)	maximum rate (β)
0.1	100
0.05	95
10^{-3}	80
10^{-4}	73
10^{-5}	69

Table 1. Numerical results for maximum incoming rate ($x_i < \beta$) based on acceptable SLAV probability (ε)

in which

$$\gamma(x) = a + \frac{1}{\sum_{k=0}^{100} \frac{x^k}{k!}} \left(-a + b \sum_{k=1}^{100} k \frac{x^k}{k!} \right) \tag{19}$$

The $\gamma(x)$ can be obtained using (1), (2) and (7) with ease. We set $a = 22$ and $b = 0.73$ for a PM equipped with Intel Xeon E3 processor((spec.org, 2018)). Then, we can show that the first order derivative of $\gamma(x)$ is positive ($\gamma'(x) > 0$) and the second order derivative of $\gamma(x)$ is negative ($\gamma''(x) < 0$) for ($0 \leq x \leq 100$) with numerical methods.

178   **Theorem 1.** *For any x and y ($0 < y \leq x \leq \beta$), there exists δ ($0 < \delta \leq y$ and $0 < \delta \leq \beta - x$), so that*

$$\gamma(x + \delta) + \gamma(y - \delta) \leq \gamma(x) + \gamma(y) \tag{20}$$

Proof. $\gamma''(x) < 0$ means that the derivative ($\gamma'(x)$) is nonincreasing. The condition $y \leq x$ implies that the $\gamma'(x) \leq \gamma'(y)$. Using definition of derivative and $0 < \gamma'(x)$, we obtain

$$0 < \frac{\gamma(x + \delta) - \gamma(x)}{\delta} \leq \frac{\gamma(y) - \gamma(y - \delta)}{\delta} \tag{21}$$

179   Multiplying the above inequality by δ and adding $\gamma(y - \delta) + \gamma(x)$ yields (20).   □

180   Let X denotes the set of elements of $\vec{x}$. We define $\Phi(\vec{x}) = \{x \in X | 0 < x < \beta\}$.

181   **Theorem 2.** *The size of the subset Φ for minimal vector is at most one.*

182   *Proof.* Consider $\vec{x}$ which satisfies (17)- (18) and $|\Phi(\vec{x})| > 1$. In our proof, we define a method to
183   transform $\vec{x}$ to $\vec{x'}$. Then we show the following properties for the transformation.

1. The value of f_o for the transformed $\vec{x}$ is not greater than the original $\vec{x}$

$$f_o(\vec{x'}) \leq f_o(\vec{x}) \tag{22}$$

184 2. The transformation has convergence property. If we repeat the transformation we reach $\vec{x^*}$ where $\Phi(\vec{x^*}) \leq 1$
185   and we could not apply transformation anymore.

     First the definition of transformation, because $|\Phi(\vec{x})| > 1$, we can find x_i and x_j from $\Phi(\vec{x})$ where $0 < x_i < \beta$ and $0 < x_j < \beta$. Without loss of generality, assume $0 < x_i \leq x_j < \beta$. We have two cases for $x_i + x_j$: 1)$x_i + x_j \geq \beta$ 2)$x_i + x_j < \beta$. In the first case, we define $\delta = \beta - y_i$ and then change only the value x_i and x_j in $\vec{x}$ to get $\vec{x'}$ as follows:

$$x'_i = x_i + \delta = \beta \tag{23}$$

$$x'_j = x_j - \delta \tag{24}$$

186   In the second case, we define $\delta = y_j$ and then change only the value x_i and x_j in $\vec{x}$ as follows

$$x'_i = x_i + x_j \tag{25}$$

$$x'_j = 0 \tag{26}$$

187   Note that $x'_i + x'_j = x_i + x_j$ and the constraint (17) is still satisfied by $\vec{x'}$.
188   After defining transformation, we prove the first property. Using Theorem 1 with defined δ, we can
189   conclude

$$\gamma(x_i + \delta) + \gamma(x_j + \delta) = \gamma(x'_i) + \gamma(x'_j) \leq \gamma(x_i) + \gamma(x_j). \tag{27}$$

190   Adding unchanged elements of $\vec{x}$ to both sides of above inequality, yields (22).
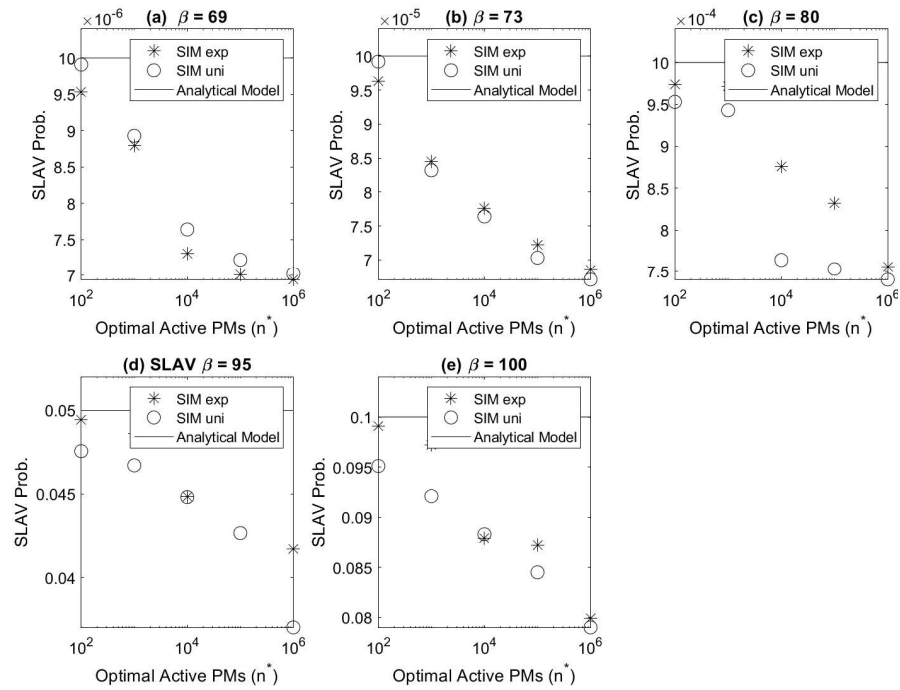
Figure 3. SLAV probability in data center at steady state

For the proof of the second property, consider (23) and (26). These imply that at least one of x_i or x_j will not be in the subset $\Phi(\vec{x'})$. Then the size of subset Φ is decreased by the transformation as follows

$$|\Phi(\vec{x})| - |\Phi(\vec{x'})| = \begin{cases} 1 & \text{for } x_i \neq x_j \\ 2 & \text{for } x_i = x_j = \dfrac{\beta}{2} \end{cases} \tag{28}$$

Because $|\Phi(\vec{x})| > 0$ it will reach 1 or 0 eventually and more transformation is not possible. At this point, due to the first property (22) we reach a minimal vector. □

Without loss of generality we assume that the elements of the minimal vector are ordered $x_1^* \geq x_2^* \geq ... \geq x_{|P|}^*$. The minimal (solution) of (17)-(18) will be

$$x_i^* = \begin{cases} \beta & \text{for } 1 \leq i \leq n^* \\ \dfrac{\lambda}{\mu} - \beta n^* & \text{for } i = n^* + 1 \\ 0 & \text{for others} \end{cases} \tag{29}$$

where $n^* = \lfloor \frac{1}{\beta \mu} \rfloor$ is the number of PMs which must be filled completely (up to β). The remaining load (if exists) must be dispatched to the next PM. For large scale data centers, we can neglect this PM and show the solution of (10)-(13) as follows.

$$r_i^* = \begin{cases} \dfrac{1}{n^*} & \text{for } 1 \leq i \leq n^* \\ 0 & \text{for others} \end{cases} \tag{30}$$

For implementation, we only need a random generator. When a new VM arrives, we draw i form $[1, n^*]$ and sends this VM to PM i. We do not require any polling, therefor our implementation is simple and agile as we need it in a cloud data center.
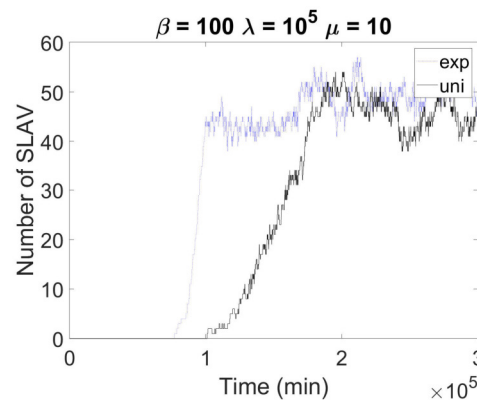
Figure 4. Effect of sojourn time distribution on the convergence time

SIMULATION RESULTS

To validate the analytical solution, we have built a discrete event simulator of a cloud provider data center using MATLAB (MATLAB, 2012). We have considered the system with two different sojourn time distribution: exponential and uniform. In both cases the mean sojourn time is fixed at $\mu = 10^1$. The mean inter arrival time of VMs was made variable from ($\lambda = 10^4$ to 10^8) to give reasonable insight into the behavior and dimensioning of CP data centers. Regarding λ and μ, traffic intensity varies from 10^4 to 10^8 which represents the mean number of VMs in data center at steady state according to the Little Formula. The number of active PMs according to ($n^* = \frac{\lambda}{\beta\mu}$) depends indirectly on ε, e.g. for $\varepsilon = 10^{-5}$ ($\beta = 69$) it varies from 145 to 145,000 servers. The values chosen may be quite applicable to small- to large-sized CPs data centers that try to keep the utilization of their servers as high as possible while guarantee a minimum QoS for the users. It is noteworthy that no cloud provider published information regarding average traffic intensity, number of servers, or the percentage of reserved.

We generate confidence intervals (95%) for steady state measurement using independent replications with deletions method. First, we run 50 independent replications of each simulation, then we remove samples of transient phase and finally we calculate the sample mean. Figure 3 depicts the SLAV probability in data center at steady state. Simulation results follows analytical model perfectly for all ($\frac{\lambda}{\mu}$) and β, values. This observation supports the validity of the analytical model findings. As can be seen, probability for exponential sojourn time is generally less than probability for uniform one. Note that the mean time is the same for both distribution, and this may relates to the variance. The variance of exponential is about 10, but the variance of uniform is about 30. As the number of active servers increases, the SLAV probability decreased steadily. This trend is due to the fact that with more active servers, an arriving VM has more places to be hosted and the chance of blocking and SLAV is lower.

Figure 4 shows the effect of sojourn time distribution on the convergence time. The data center with exponential sojourn time reaches to steady state sooner than a data center with uniform sojourn time. We only show the results for $\beta = 100, \lambda = 10^5, \mu = 10$ because the discussion aims to highlight the influences of distribution on convergence time; for larger systems and lower probability, the discussion is the same.

In order to study the power consumption of our method in practice, we use Google trace (Reiss et al., 2011) which consists of several concurrent VMs in a single $\sim 12K$ server farm where the resource demand of VMs is highly dynamic and needs quick decisions. Figure 5 compares the power consumption for our method with two benchmark algorithms in the literature: Power Aware Best Fit Decreasing (PABFD) (Beloglazov and Buyya, 2012) and Modified Throttled (MT) (Domanal and Reddy, 2013). The power consumption in our method is just about 2% more than PABFD and MT. This higher power consumption is acceptable, because the workload is highly dynamic, varying over time, and is driven by many short jobs that demand quick scheduling decisions and updates. PABFD and MT suffers from long decision process and overwhelming migrations. These shortcomings make both of them fruitless in real world cloud computing scenarios. The comparison gives us an idea about how far we are from idealistic benchmark algorithms in the literature.
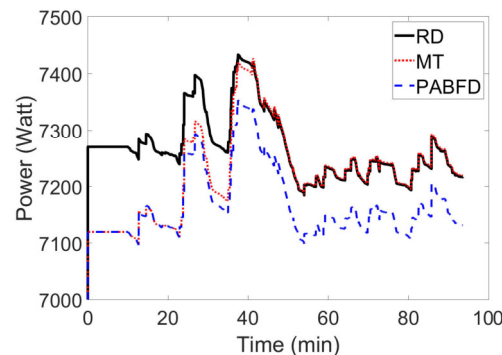
Figure 5. Power consumption by a data center managed by Random Dispatcher and benchmark algorithms

CONCLUSION

Effective resource management is a major challenge for the leading cloud providers (e.g., Google, Microsoft, Amazon). Performance evaluation of data center is an important aspect of resource management which is of crucial interest for both cloud providers and cloud users. In this paper, we proposed an analytical model based on $M/G/n/n$ system for performance evaluation of a cloud computing data center. Due to the nature of the cloud computing, we assume general sojourn time for VMs as well as large number of PMs. These assumptions make our model acceptable in terms of scale and diversity. Through extensive numerical simulations, we showed our analytical model closely alignes with simulation results. Our results also indicate that our method consumes a bit more power than idealistic benchmark in the literature.

In future, we plan to extend our model for variable size VMs. Studying how "Power of Two Choice" can improve the result of randomized dispatcher will be another dimension of extension.

REFERENCES

Beloglazov, A. and Buyya, R. (2012). Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in Cloud data centers. *Concurrency and Computation: Practice and Experience*, 24(13):1397–1420.

Boxma, O. J., Cohen, J., and Huffels, N. (1979). Approximations of the mean waiting time in an m/g/s queueing system. *Operations Research*, 27(6):1115–1127.

Carvalho, M., Menasce, D., and Brasileiro, F. (2015). Prediction-Based Admission Control for IaaS Clouds with Multiple Service Classes. In *2015 IEEE 7th International Conference on Cloud Computing Technology and Science (CloudCom)*, pages 82–90. IEEE.

Domanal, S. G. and Reddy, G. R. M. (2013). Load Balancing in Cloud Computingusing Modified Throttled Algorithm. In *2013 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM)*, pages 1–5. IEEE.

Kelly, F. P. (1991). Loss networks. *The annals of applied probability*, pages 319–378.

Kimura, T. (1983). Diffusion approximation for an m/g/m queue. *Operations Research*, 31(2):304–321.

Kimura, T. (1996). A transform-free approximation for the finite capacity m/g/s queue. *Operations Research*, 44(6):984–988.

Ma, B. N. and Mark, J. W. (1995). Approximation of the mean queue length of an m/g/c queueing system. *Operations Research*, 43(1):158–165.

MATLAB (2012). *version 7.14.0 (R2010a)*. The MathWorks Inc., Natick, Massachusetts.

Miyazawa, M. (1986). Approximation of the queue-length distribution of an m/gi/s queue by the basic equations. *Journal of Applied Probability*, 23(2):443–458.

Nozaki, S. A. and Ross, S. M. (1978). Approximations in finite-capacity multi-server queues by poisson arrivals. *Journal of Applied Probability*, 15(4):826–834.

Reiss, C., Wilkes, J., and Hellerstein, J. L. (2011). {Google} cluster-usage traces: format + schema. Technical report, Google Inc., Mountain View, CA, USA.

273   Shojafar, M., Cordeschi, N., and Baccarelli, E. (2016). Energy-efficient adaptive resource management
274   for real-time vehicular cloud services. *IEEE Transactions on Cloud computing*, 7(1):196–209.

275   Smith, J. M. (2003). M/g/c/k blocking probability models and system performance. *Performance*
276   *Evaluation*, 52(4):237–267.

277   spec.org (2018). All specpower_ssj 2008 benchmark results. Accessed: Nov 2018.

278   Tijms, H. C., Van Hoorn, M. H., and Federgruen, A. (1981). Approximations for the steady-state
279   probabilities in the m/g/c queue. *Advances in Applied Probability*, 13(1):186–206.

280   Trivedi, K. S. (2001). *Probability and Statistics with Reliability, Queueing, and Computer Science*
281   *Applications*. Wiely, 2nd edition.

282   Wang, B., Chang, X., and Liu, J. (2015). Modeling Heterogeneous Virtual Machines on IaaS Data Centers.
283   *IEEE Communications Letters*, 19(4):537–540.

284   Wang, L. and Gelenbe, E. (2015). Adaptive dispatching of tasks in the cloud. *IEEE Transactions on*
285   *Cloud Computing*, 6(1):33–45.

286   Wang, W., Jiang, Y., and Wu, W. (2016). Multiagent-based resource allocation for energy minimization in
287   cloud computing systems. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 47(2):205–
288   220.

289   Yang, B., Tan, F., Dai, Y.-S., and Guo, S. (2009). Performance evaluation of cloud service considering
290   fault recovery. In *IEEE International Conference on Cloud Computing*, pages 571–576. Springer.

291   Yao, D. D. (1985). Refining the diffusion approximation for the m/g/m queue. *Operations Research*,
292   33(6):1266–1277.