Virtual Instance Resource Usage Modeling: A Method for Efficient Resource Provisioning in the Cloud

SeyedAli Jokar Jandaghi
Computer Science Dept., University of Toronto
Toronto, ON M5S3G4, Canada
Email: saj@cs.toronto.edu

Kaveh Mahdaviani
ECE Dept., University of Toronto
Toronto, ON M5S3G4, Canada
Email: kaveh@comm.utoronto.ca

Cristiana Amza
ECE Dept., University of Toronto
Toronto, ON M5S3G4, Canada
Email: amza@ece.utoronto.ca

***Abstract*—Cloud computing is a promising framework providing a variety of solutions, ranging from software services to infrastructure services through the mechanism of customizable virtual instances. The cloud manager is responsible for resource provisioning for these instances to provide guaranteed performance but at the same time avoiding underutilization of the platform. In this paper, we introduce a novel method for modeling the resource usage of VIs which allows for better VI placement with more efficient resource usage in the physical infrastructure. Our proposed framework uses the mixture of Gaussians to model each virtual instance resource usage. Then for placement, a modified probabilistic bin packing method is been proposed to take advantage of modeling for placing virtual instances. We compared our scheme with other bin packing methods that use rigid statistical models, and the results support the efficiency and accuracy of our method which leads to more than 50% resource saving while preserving the given performance guarantee.**

I. Introduction

Cloud computing has shown a tremendous potential to be a utilitarian solution in the past decade. The paradigm is based on creating an isolated environment for users over a pool of resources that meet user's demand. This inception is fulfilled by computation components, namely virtual instances. The virtual instances are assigned a certain amount of hardware resources based on what user specified. This designation happens through the cloud placement algorithm on the virtual instance(s) to address performance isolation between co-located virtual instances.

Despite the recent evolution in cloud computing management techniques, still the big players like Google [1] and Twitter [2] suffers from the lack of efficient resource utilization. Performance isolation on one hand and efficient resource utilization, on the other hand, are the main trade-off for a placement algorithm in the cloud [3]. Mostly users overestimate their indispensable resources which means the host server would be underutilized due to this mistaken assessment. In addition, servers are among the most significant cost of a cloud platform [4] [5] and they deliver less performance capability in case of being underutilized, thus it is crucial to employ them efficiently. To address the under utilization problem, we are proposing a packing method aware of the past resource usage patterns of virtual instances. We believe monitoring the past behavior of virtual instances in terms of resource usage model provides useful insight on future consumption of that virtual instance.

Fig. 1. Overall architecture of Proposed solution for packing virtual instances with probabilistic sizing

Our suggested solution is based on monitoring of resource usage over time and modeling the utilization pattern with mixture of Gaussian. With the assumption that this modeling is a deterministic profile of a virtual instance for future, we propose a bin packing method to use this model for more efficient packing. In other words, rather than assuming a rigid size for virtual instances, which could be overestimation or underestimation based on model type, we use a probabilistic model of the resource usage. Our approach is divided into four main parts as described in Figure 1. Each part will be explained in the next sections. The main contribution is in the packing algorithm that places virtual instances using modeling to provide an efficient resource provisioning at the same time with an assurance on the quality of service.

Studying the characteristics of virtual instances in the light of their resource usage footprint over a significant amount of time demonstrates their real resource usage versus the requested resources. A statistical analysis of this behavior over time, for example the expected value, expresses the key feature of the resource usage which can be used in placement algorithm towards efficient resource provisioning.

Statistical modeling aims at providing a set of probability distributions that describe the sampled data in an accurate way. The feature embodied by a statistical model can be as simple as the first moment function which builds a loose model for the sample points. We propose using an advanced modeling method, namely Mixture of Gaussian, to extract more accurate probability distributions of a resource usage pattern.

This modeling provides a prediction of the resources usage

978-3-901882-89-0 @2017 IFIP

for each virtual instance. Consequently, the resource usage of a server is the aggregated models of its guest instances. In this regard, our simulation shows that using this technique gives a stringent model of the resource usage. This modeling can improve resource allocation in compared to both no modeling and light modeling techniques. we mapped the placement strategy to bin packing algorithm and explained the integration of modeling with this technique to make the most of it. In nutshell, the proposed method is about applying the model on real data, for example from Google, and on integration of our model to bin packing algorithm.

The rest of the paper is organized as follows. A review of state of the art research on resource usage modeling and addressing the underutilization problem is provided in Section II. Section III discusses the method description and states a big picture of proposed solution. While IV overviews some preliminaries, Section V puts forward the modified packing algorithms we used to reflect the probabilistic sizing of virtual instances. Section VI describes the proposed approach as a concrete method to resolve under utilization problem. Our experiment results and evaluations are presented in Section VII. At the end, we presented conclusion and future work in Section VIII followed by references.

II. Related Work

The main efforts on tackling the under utilization problem is in the area of virtual instance management by taking into account the quality of service alongside the efficiency. Younge et al. [6] provided a hierarchy of methods and techniques used to improve the data center efficiency including virtual instance placement, scheduling, and data center design. These techniques are availed in server consolidation techniques and have been surveyed by Ahmad et al [7].

A taxonomy of server consolidation techniques is presented in [8] considering a range of parameters including hardware utilization and performance impact. Dynamic resource assignment policy and (re)allocation of resources to servers are the techniques of our interest which consider both performance and utilization. This assignment needs close monitoring of resources being used by virtual instance and analyzing them as stochastic process, time series model [9], or to be modeled by a probability distribution.

In [10], a consolidation method was suggested based on dynamic utilization threshold by assuming a normal CPU resource utilization for any individual host. The paper also provides a framework to perform placement regarding the migration, energy, and quality of service cost. A number of these mathematical modeling techniques in cloud are surveyed in [11]. As for experimental analysis, [12] criticizes some of the dynamic resource allocation techniques and highlighted the overhead of migration cost in dynamic resource provisioning. An other group of works examine virtual instance behavior on real data. [13] studied Google cluster data set and show the underutilization of servers in the cluster.

III. Method Description

In this section we briefly describe the method proposed in this work to confront with the underutilization problem in cloud computing.

The first step in our method is to derive a probabilistic model for a virtual instance resource utilization in a predefined time slot, based on the observation of its resource usage in the past. We require the probabilistic model to be a scalar, independent and identically distributed over all time slots in the future. In order to handle the temporal correlation in the sampled resource utilization, we propose an averaging window and set the window length appropriately to mitigate the dependency among the averaged samples. Finally, for the probabilistic modeling we suggest to use the mixture of Gaussians method.

The second step in the proposed method is to perform a probabilistic placement approach for virtual instances. To this end, we introduce a probabilistic bin packing algorithm based on a modification of the well known *first fit bin packing* algorithm proposed in [14]. The key feature in this probabilistic approach in placement is to provide a collective quality of service guarantee for the aggregate resource usage of all virtual instances in a server, rather than providing an individual guarantee for each VI separately. This is possible due to the fact that we have developed simple and independent probabilistic models for virtual instances.

This idea turns out to provide a significant gain in terms of reduction in the underutilization, or equivalently reduces the required allocated resources for a fixed number of virtual instances.

IV. Preliminaries

We will provide a brief review of mixture models, more specifically Gaussian mixture model,followed by the methods used for learning the parameters of the model. Since this modeling needs data points to be i.i.d, we will examine a statistical test method to assure i.i.d condition before explaining the mixture of Gaussians.

A. Chi-Square Test of Independence

We assume two random variables, each representing a sample of consecutive pair. Chi-square test [15] can be used for these two random variables to show independence of successive sample points. more detail about Chi-Square method can be found in [15].

B. Mixture of Gaussians

Among the many ways to approximate the true distribution of i.i.d observed data points, mixture model represents the mixture distribution of the overall population in the format of component densities. Although Mixture Model provides information on boundaries of sub-population but it has no assumption on the probability distribution of each sub-population.

If we add the assumption that each sub-population can be represented by a normal distribution the mixture model would be called Mixture of Gaussians. Having said that, Mixture

of Gaussians is a probability density function composed of Gaussian Components, each with a corresponding weight. We will provide a formal definition of the mixture of Gaussians followed by a brief introduction to the commonly used algorithm for estimating its parameter.

A random variable X is capturing the monitored results of resource usage. We define our Mixture Model probability $f_X(X = x|\vartheta)$ to be a weighted compound of k Gaussian Distributions. Each Gaussian distribution $\mathcal{N}$ is parameterized in terms of its mean μ_i and variance σ_i^2 with a corresponding weight ω_i representing the probability of an observation belonging to sub-population i. The distribution of the overall population is given by equation (1). All the parameter of the model including weights, Means, and variances are in tuple ϑ.

$$f_X(X = x|\vartheta) = \sum_{i=1}^{k} \omega_i \mathcal{N}_i(X = x|\mu_i, \sigma_i^2) \tag{1}$$

$$\vartheta = \{(\omega_i, \mu_i, \sigma_i) \quad | \quad \forall i \in \{1 \dots K\}\} \tag{2}$$

Each Gaussian component $\mathcal{N}_i$ is a univariate Gaussian distribution given by the equation (3).

$$\mathcal{N}_i(X = x|\mu_i, {\sigma_i}^2) = \frac{1}{\sigma_i\sqrt{2\pi}}\, e^{-\frac{(x-\mu_i)^2}{2\sigma_i^2}} \tag{3}$$

C. Mixture of Gaussians Parameter Estimation

Likelihood function is the function of ϑ denoted by $\mathcal{L}(\vartheta|x_1, x_2, ..., x_D)$ given sample set of size d. A technique called the Expectation-Maximization (EM) algorithm [16] is being used to learn the values of ϑ to maximize the log likelihood function denoted by $\ln \mathcal{L}(\vartheta|x_1, x_2, ..., x_d)$ as specified in equation (4). An iterative solution is suggested by EM algorithm to refine a random initial parameter since log-likelihood maximization does not have close form solution. Details of EM algorithm is beyond the scope of this paper and more details can be found in [17].

$$\ln \mathcal{L}(\vartheta|x_1, x_2, ..., x_d) = \sum_{j=1}^{d} \ln\{\sum_{i=1}^{k} \omega_i \mathcal{N}_i(X = x_j|\mu_i, \sigma_i)\} \tag{4}$$

Last but not least is choosing the number of components, denoted by k, to have an accurate but not over-fitted model. We used *Bayesian Information Criterion (BIC)* [18] , as shown in equation (5), to learn the desired number of components. In this equation, a penalty term is added to the negative log likelihood. The penalty term is proportional to the number of components denoted by k and size of sample set specified by d. The most appropriate k returns the minimum possible value for *BIC*.

$$\text{BIC} = -2 \cdot \ln max(\mathcal{L}) + k \cdot \ln(d) \tag{5}$$

Mixture of Gaussians model will be a set of known Gaussian distributions with their corresponding weight. The Model gives an accurate estimation of the real distribution that the sample points were generated from. We will use this mode in the proposed algorithm in Section VI as a critical component.

V. Bin Packing with Probabilistic Item Sizing

Bin Packing problem is a well defined NP-hard problem, where we are required to pack d items into as few bins as possible. Although the items can have multiple different dimensions [19] and the bins are having limited capacity in each dimension. Since our emphasis is on the role of modeling in propounded packing method, for now, we take one dimension into account. Bin packing problem has been extensively studied in combinatorial optimization with a rich literature on theoretical constructs [14].

A set of d items is given in a one dimensional Bin Packing problem where item i represented by I_i has a corresponding variable size denoted by a random variable S_i. We assume a fix capacity c for all the bins and the maximum size of an item is less than the bin's capacity. A valid packing is selection of m bins to accommodate all the items without violating any bin's capacity. The objective is to find a valid packing with with minimum number of non-empty bins.

We call fully packed bins *crammed* bins while the other available bins are called *underused* bins. Content of a bin is measured by *level* parameter ι_{bin}, as specified in equation (6), which is the expected summation of co-located items' sizes. An empty bin's level is zero whereas a crammed bin's level is equal to its capacity.

$$\iota_{bin} = \sum_{j \in bin} \mathrm{E}[S_j] \tag{6}$$

As this problem is an NP-hard problem, so the optimal solution may not admit any polynomial-time algorithm. Therefore, many heuristics have been proposed to solve different variant of this problem. First Fit Decreasing (FFD) algorithm is a commonly used heuristic solution, which involves sorting all the items in a descending order before assigning bins to them in a greedy fashion. The Pseudo code is provided in Algorithm 1.

Algorithm 1: First Fit Decreasing Algorithm

```
Data: list of items [I_1, I_2, ··· , I_d]
Result: Packed Bins
First_Bin = b_1  Sorted_Items =
Sort_by_expected_size( [I_1, I_2, ··· , I_d])
for items in Sorted_Items do
    Current_Bin = First_Bin
    while item not accommodated do
        if Current_Bin.CanAccomodate(item) then
            assign(Current_Bin, item)
            update(ι_{Current_Bin})
            Break
        Current_Bin = Current_Bin.Next;
```

Although the solution is very straight forward for items with fixed sizes but we are required to improve it to work

with probabilistic sizing. We use a probability bound, namely Chernoff bound [20], on each bin to give us a guarantee on marginal chance of capacity violation. Given the independence of items' size, the aggregated size of items in a bin can be represented by a random variable specifying the range and probability of the total size.

This probability bound is denoted in equation (7) for set of all items in a single bin denoted by J. The left side of the inequality is chance of capacity, denoted by c, violation and the right hand side of the inequality is the bound for this probability.

$$P(\sum_{j\in J} S_j \geq c) \leq \frac{\prod_{j\in J} E[e^{tS_j}]}{e^{tC}} \quad ; \quad \forall t > 0 \qquad (7)$$

The right hand side fraction numerator term of equation (7) has been used to refer to moment generating function of S which is the expectation of random variable e^{tS}. Expanding this inequality depends on the information from the probability distribution of sizing and capacity of each bin. While capacity is part of the problem definition, we will see in the next section how our model provides information on the probability distribution.

VI. Enhanced Packing using Mixture Models

The flavor of each virtual instance specifies how much resources should be assigned to it while in reality less amount of resources is being used by a virtual instance due to over-provisioning. To extract the real trend of resource usage, some data points can be sampled and used for modeling. if the sampling rate is very high, eventually the consecutive sample points are dependent while if the sampling rate is loose enough, statistical test can be used to show the independence. It is also to a good extent due to averaging as well.

Regarding that our proposed method is established on the independence of monitored results, we first need to show the independence of sample points. We define a window of w to aggregate over w number of sample points towards losing the dependency among successive samples. We start with a window size of one to check if the sample points are independent and if not, we keep increasing the window size by one and take the average of sample points under each windows as the representative of that window until the sample points are independent.

This independence allows us to model each virtual instance i sizing as a single random variable S_i. Accordingly, Mixture of Gaussians is used for either one of the virtual instances to model their corresponding sizes. Each model includes the number of components, weight of the components, and Gaussian parameters of each component. An examples of this modeling is provided in the next section.

The proposed bin packing method needs information about the probability distribution of the items which is included in the mixture of Gaussians modeling of each virtual instance. Given all the above, the proposed placement solution is completed and it is described in algorithm 2.

Algorithm 2: Proposed solution for packing virtual instances with varying sizes

Data: Monitoring Dataset DS = [M_1, M_2, $\cdots$, M_d]
Result: Enhanced Packing using mixture of Gaussians models
`/* Statistical Test for Independence */`
w = 1
while *Statistical_test(w,DS)* **do**
  w = w + 1
  DS = refine(DS, w)
`/* Mixture of Gaussian Modeling */`
for *item i in DS* **do**
  I_i = Gaussian_Mixture_model(i)
`/*Probabilistic First Fit Bin Packing*/`
PFFD([I_1, I_2, $\cdots$, I_d])

To address the feasibility of adding an item to a bin, *can_accomodate* function is called in bin packing algorithm. The Chernoff bound is calculated for items in the bin and check if the bound is less than the desired performance guarantee. The derivation of Chernoff bound for Mixture of Gaussians using (7) and (1) will be:

$$P(\sum_{j\in J} S_j \geq C) \leq \frac{\prod_{j\in J}\prod_{k=1}^{K(j)} E[e^{t\omega_{k,j}\mathcal{N}(X=x_{k,j}|\mu_{k,j},\sigma^2_{k,j})}]}{e^{tC}} \quad \forall t > 0 \qquad (8)$$

It has not escaped our notice that the moment generative function of a Gaussian is known. We also can replace the weighted normal random variable with another random variable like Z where its expected value $\mu\prime$ is $\omega\mu$ and its variance $\sigma\prime^2$ is $\omega^2\sigma^2$. The result is shown in equation (9) and 10. As for finding the tightest upper bound, BroydenFletcherGoldfarbShanno method [21] was used to find the minimum value for the bound in the inequality.

$$P(\sum_{j\in J} S_j \geq C) \leq \frac{\prod_{j\in J}\prod_{k=1}^{K(j)} E[e^{t\mathcal{N}(Z=z_{k,j}|\mu\prime_{k,j},\sigma\prime^2_{k,j})}]}{e^{tC}} \quad \forall t > 0 \qquad (9)$$

$$P(\sum_{j\in J} S_j \geq C) \leq \frac{\prod_{j\in J}\prod_{k=1}^{K(j)} e^{t\mu\prime_{k,j}+t^2\frac{1}{2}\sigma\prime^2_{k,j}}}{e^{tC}} \quad ; \quad \forall t > 0 \qquad (10)$$

VII. Evaluation

A. Data set

To show the efficiency of our approach, we used part of the Google data which is been monitored from one of its Data center's cluster over a month. Google uses its containers

as virtual instances which mostly request a small amount of resources. Another feature of a container is the abundance of events that it encounters over its life time. Failing, getting killed, and being rescheduled are some examples of events that befall a container. To provide a reliable modeling of resource usage, our method requires virtual instances with long enough lifetime. Thus, we selected a group of virtual instances that have had been alive for almost one and half a day continuously.

We also normalized all the resource usage by the requested value of that resource. Therefore, the resource usage was a value between 0 and 100 for all the virtual instances. The term *idle* virtual instance refers to instances that in average utilize a resource less than a small arbitarary value as fully explained in [22]. we removed this group of instances from our experiment set. The reason is that our previous solution can effectively consolidate *idle* virtual instances [22]. Furthermore, we wanted to lay our emphasis on modeling validity which is incontrovertible for idle instances.

Last but not least, we plateau the usage to 100 when the usage exceed the requested resources. Google lets virtual instances use more than requested resources for a while, although it has been shown [23] that these type of jobs got killed by cluster manager. All of the above, we examined millions of virtual instances in the first two days and applying aforementioned restrict filters results hundreds of virtual instances out of the huge initial set.

B. Statistical Test For Independence

The Google data set is set samples that have been taken every five minutes We used *Chi-square* test [15] to show the independence of consecutive sample points. Our results show that window of 5 minutes is loose enough for independence and there is no need to enlarge the window size for independence.

C. Modeling

We used the mixture of Gaussians explained in Section IV for each normalized monitored CPU usage vector. The results were promising and showed a very small mean square error. Figure 2 shows an example of mixture of Gaussians modeling with 8 components along with its time series in Figure 3. The BIC values associated with different number of components is showed in Figure 4 and it shows the optimal BIC is for 8 components.

Mixture of Gaussians modeling takes the maximum possible number of component as an input and obviously, it will be more accurate with a larger input. Table I shows the time and error in average for different limits on the maximum number of components values. Our examination on the number of virtual instances that changed their number of component showed that 3 is an acceptable limit for the maximum number of Gaussian components.

D. Enhanced Packing

Given the models of virtual instances, we need to pack them to servers with the capacity of 100. We assume one server

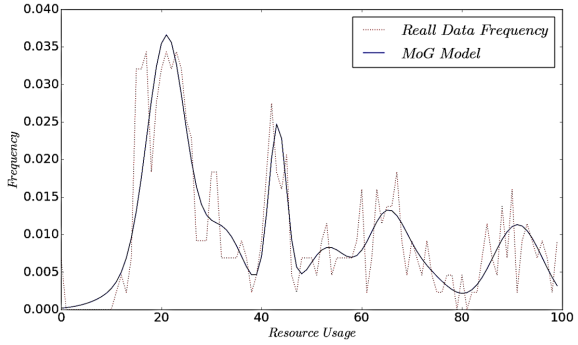

Fig. 2. Mixture of Gaussian modeling and histogram of a virtual instance

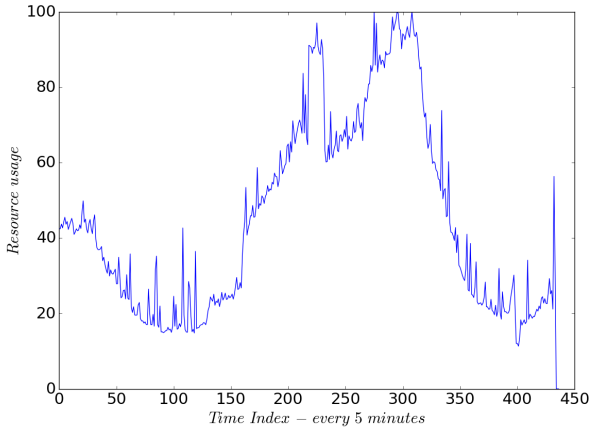

Fig. 3. Normalized CPU usage of a virtual instance over one day and half

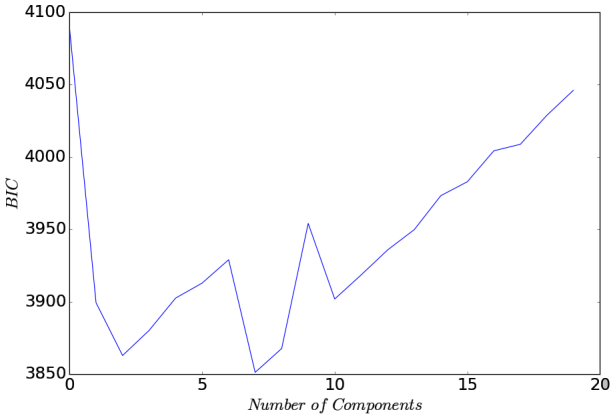

Fig. 4. BIC parameter changes as a function of number of Gaussian components. The optimum value of BIC shows the desired number of components which is 8 in this example.

TABLE I
MIXTURE OF GAUSSIAN MODELING TIME AND COMPLEXITY

Maximum Number of Components	Average MSE for 1000 Samples	Average MSE with Model	Modeling Time (seconds)
3	9.40e-5	80.85e-5	4.59
7	8.59e-5	6.32	12.13
11	8.15e-5	6.23	21.84
15	8.13e-5	6.39	34.24
19	8.16e-5	6.42	48.40

TABLE II
RESOURCE USAGE AND PERFORMANCE EVALUATION OF DIFFERENT PACKING METHODS.

Algorithm	Run Time	Number of Violations	Number of Occupied Servers
Basic	0 seconds	0	136
FFD using Mean	0.09 seconds	812	91
FFD using Mean+S.D.	0.08 seconds	41	96
Probabilistic FFD	82.51 seconds	23	48

is provisioned to each virtual instance in a naive trivial way regardless of the virtual instance resource usage.

To show the importance of modeling, We compared our packing approach with simple statistical modelings that use expected value or variance for modeling the virtual instance behavior. A similar technique for resizing virtual instance was used in our previous work [22].

We evaluated our proposed packing scheme in terms of CPU usage for different modeling. We used average usage and sum of average usage plus the standard deviation as the size of the virtual instance for two other base packing. Table II shows the number of servers that have been used for each packing method. To show the effectiveness of our modeling, we also reported the total number of violation that has happened, meaning the number of time resource usage exceeded the total capacity of any server. The results show the effectiveness of our approach due to precise modeling of virtual instances.

VIII. CONCLUSION AND FEATURE WORKS

We showed profiling the resource usage of a virtual instance can be very beneficial for saving resources. The probability model of a virtual instance gives more accurate information compare to simple statistical information. Among many methods of modeling, we chose mixture of Gaussians due to its precision and ease of use for further extensions. We plan to investigate other modeling methods for future to enhance the profiling.

As for packing, it has been shown how modeling can enhance packing by using a probabilistic bound for each server but the algorithm was one of the very simple common methods. We plan to modify and use more advanced bin packing methods, for example sum of square, for future.

Despite the astonishing result we studied in our experiments, we need to consider more complicated features of a placement algorithm for future. To name a few, considering inference, virtual instance communications, and other resources rather than CPU can be motive for future extensions.

REFERENCES

[1] Z. Liu and S. Cho, "Characterizing machines and workloads on a google cluster," in *2012 41st International Conference on Parallel Processing Workshops*. IEEE, 2012, pp. 397–403.

[2] C. Delimitrou and C. Kozyrakis, "Quasar: resource-efficient and qos-aware cluster management," in *ACM SIGPLAN Notices*, vol. 49, no. 4. ACM, 2014, pp. 127–144.

[3] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica *et al.*, "Above the clouds: A berkeley view of cloud computing," 2009.

[4] C. Gough, I. Steiner, and W. Saunders, "Data center management," in *Energy Efficient Servers*. Springer, 2015, pp. 307–318.

[5] A. Greenberg, J. Hamilton, D. A. Maltz, and P. Patel, "The cost of a cloud: research problems in data center networks," *ACM SIGCOMM computer communication review*, vol. 39, no. 1, pp. 68–73, 2008.

[6] A. J. Younge, G. Von Laszewski, L. Wang, S. Lopez-Alarcon, and W. Carithers, "Efficient resource management for cloud computing environments," in *Green Computing Conference, 2010 International*. IEEE, 2010, pp. 357–364.

[7] R. W. Ahmad, A. Gani, S. H. A. Hamid, M. Shiraz, A. Yousafzai, and F. Xia, "A survey on virtual machine migration and server consolidation frameworks for cloud data centers," *Journal of Network and Computer Applications*, vol. 52, pp. 11–25, 2015.

[8] A. Varasteh and M. Goudarzi, "Server consolidation techniques in virtualized data centers: A survey," *IEEE Systems Journal*, 2015.

[9] S. F. Piraghaj, A. V. Dastjerdi, R. N. Calheiros, and R. Buyya, "A survey and taxonomy of energy efficient resource management techniques in platform as a service cloud," *Handbook of Research on End-to-End Cloud Computing Architecture Design*, p. 410, 2016.

[10] A. Beloglazov and R. Buyya, "Adaptive threshold-based approach for energy-efficient consolidation of virtual machines in cloud data centers," in *Proceedings of the 8th International Workshop on Middleware for Grids, Clouds and e-Science*, vol. 4. ACM, 2010.

[11] G. Sakellari and G. Loukas, "A survey of mathematical models, simulation approaches and testbeds used for research in cloud computing," *Simulation Modelling Practice and Theory*, vol. 39, pp. 92–103, 2013.

[12] A. Wolke, M. Bichler, and T. Setzer, "Planning vs. dynamic control: Resource allocation in corporate clouds," *IEEE Transactions on Cloud Computing*, vol. 4, no. 3, pp. 322–335, 2016.

[13] S. Di, D. Kondo, and F. Cappello, "Characterizing and modeling cloud applications/jobs on a google data center," *The Journal of Supercomputing*, vol. 69, no. 1, pp. 139–160, 2014.

[14] K. Bernhard and V. Jens, "Combinatorial optimization: Theory and algorithms," 2008.

[15] P. Gingrich, *Introductory statistics for the social sciences*. Department of Sociology and Social Sciences, University of Regina, 1992.

[16] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the royal statistical society. Series B (methodological)*, pp. 1–38, 1977.

[17] K. P. Murphy, *Machine learning: a probabilistic perspective*. MIT press, 2012.

[18] G. Schwarz *et al.*, "Estimating the dimension of a model," *The annals of statistics*, vol. 6, no. 2, pp. 461–464, 1978.

[19] R. Panigrahy, K. Talwar, L. Uyeda, and U. Wieder, "Heuristics for vector bin packing," *research. microsoft. com*, 2011.

[20] G. Bennett, "Probability inequalities for the sum of independent random variables," *Journal of the American Statistical Association*, vol. 57, no. 297, pp. 33–45, 1962.

[21] C. T. Kelley, *Iterative methods for optimization*. Siam, 1999, vol. 18.

[22] S. A. Jokar Jandaghi, A. Bhattacharyya, S. Sotiriadis, , and C. Amza, "Consolidation of underutilized virtual machines to reduce total power usage," in *Proceedings of the 2016 Conference of the Center for Advanced Studies on Collaborative Research, CASCON*. IBM Corp, 2016.

[23] N. El-Sayed, "Exploiting field data analysis to improve the reliability and energy-efficiency of hpc systems," Ph.D. dissertation, University of Toronto, 2016.