Probabilistic Virtual Machine Assignment

David Wilcox
CS Deptartment, BYU
Provo, USA
davidw@cs.byu.edu

Andrew McNabb
CS Deptartment, BYU
Provo, USA
a@cs.byu.edu

Kevin Seppi
CS Deptartment, BYU
Provo, USA
k@cs.byu.edu

Kelly Flanagan
CS Deptartment, BYU
Provo, USA
kelly_flanagan@byu.edu

***Abstract*—We cast the assignment of virtual machines (VMs) to physical servers as a variant of the classic bin packing problem. We then develop a model of VM load that can be used to produce assignments of VMs to servers. Using this problem formulation and model, we evaluate heuristic solutions to this problem. We evaluate the performance of these solutions in stochastic load environments. We verify the model proposed and show that it can be adapted to respond well to varying VM loads.**

***Keywords*-Energy optimization; Probabilistic Models**

I. Introduction

One of the major causes of energy inefficiency in data centers is the idle power wasted when servers run at low utilization [17]. In 2005, data centers accounted for 0.8% of all world energy consumption, costing $7.2 billion (US) [9]. Part of the problem is that most servers and desktops are in use only 5-15% of the time they are powered on, yet most x86 hardware consumes 60-90% of normal workload power even when idle [20], [4], [5].

Data center costs can be reduced by utilizing virtual machines (VMs). Using virtualization, multiple operating system instances can run on the same physical machine, exploiting hardware capabilities more fully, allowing administrators to save money on hardware and energy costs. To maximize the savings, administrators should assign as many VMs as possible to servers given performance requirements. We refer to this problem as the *Virtual Machine Assignment Problem.*

The Virtual Machine Assignment Problem (VMAP) is the problem of, given probabilistic distributions over the VM load, find an initial assignment which distributes the load on the VMs such that all have access to adequate resources and the number of servers used is minimized.

This type of problem might need to be solved at an e-commerce web site, which generally have times of lower hardware utilization. During these times of lower utilization, the web site acquires very few sales and therefore has more liberty to move VMs around. Once the day begins however, traffic will pick up and administrators will be less able to move VMs around. A good initial assignment means moving fewer VMs during peak hours of production. VMAP is not the problem of reassigning load after an initial assignment has already been made. We address this issue separately.

VMAP can be seen as a type of Bin Packing Problem, the problem of assigning a set of items into a set of bins, minimizing the number of bins in use [8]. However, VMAP is not as simple as the conventional Bin Packing Problem. VMAP is different in two important ways.

1) Each server has multiple types of constrained resources which the VMs consume. Each VM adds some amount of load to each resource type provided by the server, such as memory, disk space and CPU.
2) Loads that VMs exert on servers are probabilistic and not completely known ahead of time.

Prior research has partially addressed VMAP [15], [18]. One thing that prior research has not discussed is what happens when loads are probabilistically distributed. Specifically, we wish to investigate how load distributions can be incorporated into packing virtual machines onto servers. Prior research has not focused on this specific sub-aspect of VMAP.

This paper presents a novel way of taking expectations on loads of virtual machines. If system administrators have knowledge about the loads of virtual machines, expectations can be taken at different points in the probabilistic load distribution. The purpose of this paper is to present a way that this process can be done and to investigate some consequences of treating loads probabilistically.

We outline the paper here for reference. In Section II, we describe our background research. In Section III, we describe the model that we propose in this paper. In Section IV, we describe the assignment algorithms that we will compare. In Section V, we discuss the metrics to determine success in our results. In Section VI, we detail our experimental setup. Finally, in Section VII, we discuss the results of our experiments.

II. Background Research

Different aspects of server consolidation have been studied and modeled [16], [14], [19], [3]. Other literature has focused on decreasing power use in virtualized environments [12]. In this section we review background research in three parts. First, prior research on modeling server load will be discussed. Second, We will discuss the Bin Packing problem. Third, we will briefly describe Genetic Algorithms as this is the foundation for one solution technique.

A. Virtual Machine Assignment

The problem that we identify in this paper as VMAP takes other names in literature. Stillwell et al. [18] define ResAlloc, which is a Mixed Integer Linear Program formulation of

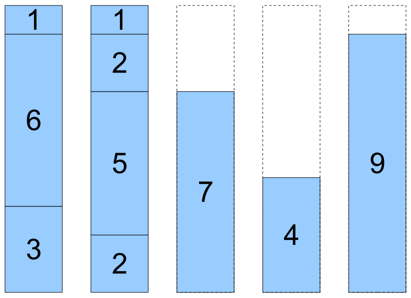

Fig. 1. An inefficient Bin Packing solution.

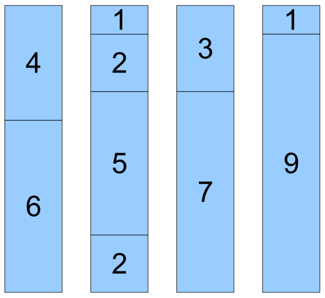

Fig. 2. An efficient Bin Packing solution.

VMAP. In their problem formulation, they consider maximizing the yield, which represents the faction of a job's achievable compute rate that is achieved, on the server with the minimum yield. They then identify different solutions and evaluate the solutions on ResAlloc.

Song et al. [15] created RAINBOW, a prototype to evaluate a multi-tiered resource scheduling scheme on a workload scenario reflecting resource demands of services in a real enterprise environment. They first define resource flowing as the process in which resources released by some VMs/services are allocated to others. Their main contributions were a multi-tiered resource scheduling scheme for a VM-based data center, a model for resource flowing using optimization theory, and a global resource flowing algorithm.

Something that has not been investigated well in prior research is the fact that virtual machines are probabilistically distributed. We investigate what happens when we know something about the probabilistic nature of loads on virtual machines. This paper will give system administrators knowledge on what they should do, given that they have prior knowledge of how their virtual machines are distributed.

B. Conventional Bin Packing

In this paper we build a model that, given a set of VMs, each with a distribution of resource utilizations, decides how VMs should be assigned to servers. We present our model as a variant of the Conventional Bin Packing Problem. The Bin Packing Problem is the problem of finding the assignment of items to bins under which the number of bins is minimized.

Definition 1. The *Bin Packing Problem* is formulated as follows. Given a finite set of n items $I = \{1, 2, \ldots, n\}$ with corresponding weights $W = \{w_1, w_2, \ldots, w_n\}$ and a set of identical bins each with capacity $\mathcal{C}$, find the minimum number of bins into which the items can be placed without exceeding the bin capacity $\mathcal{C}$ of any bin. A solution to the Bin Packing Problem is of the form $B = \{b_1, b_2, \ldots, b_m\}$, where each b_i is the set of items assigned to bin i, and is subject to the following constraints:

1) $\forall i \, \exists! \, j$ such that $i \in b_j$ (Every item belongs to one unique bin.)
2) $\forall j \sum_{n \in b_j} w_n \leq \mathcal{C}$ (The sum of the weights of items inside any bin cannot be greater than the bin capacity.)

In the Bin Packing Problem, the objective is to assign the set of items into a set of bins, minimizing the number of bins used. This idea is illustrated in Figures 1 and 2. Figure 2 shows an assignment that uses the same items as the assignment shown in Figure 1, but packs them in fewer bins.

Doing an initial placement of VMs onto servers can be seen as a type of Bin Packing Problem. In the Bin Packing Problem, a set of items are placed into bins, minimizing the number of bins. In the problem of placing VMs onto servers, VMs are assigned to servers, minimizing the number of servers, while assuring that some performance criteria is met. Because these two problems are similar in this way, we will model the problem of making an initial assignment of items to servers as a type of Bin Packing Problem.

The Bin Packing Problem has been shown to be NP Hard [2]. We solve the problem that Bin Packing is inherently intractable by using approximation algorithms to solve the problem in our experiments.

One reason why the conventional Bin Packing Problem itself cannot be used to model the VM Assignment Problem is that there is no way in the conventional Bin Packing Problem to model multiple resources on one server. For that reason, we defer to prior research and model this problem using the Multi-Capacity Bin Packing Problem [21]. We will describe this problem in more detail in Section II-C.

C. Multi-Capacity Bin Packing Problem

Definition 2. The *Multi-Capacity Bin Packing Problem* or *Vector Packing Problem* is similar to the conventional Bin Packing Problem that was given in Definition 1, but not identical. In the Multi-Capacity Bin Packing Problem, the capacity is a d-dimensional vector $\mathcal{C} = \langle C_1, C_2, \ldots, C_d \rangle$ where d is the number of resource types. The weights are redefined so that the weight of item i is a d-dimensional vector $\vec{w}_i = \langle w_{i_1}, w_{i_2}, \ldots, w_{i_d} \rangle$ [18], [10].

1) $\forall i \, \exists! \, j$ such that $i \in b_j$ (Every item has to belong to some unique bin.)
2) $\forall j \, \forall k \sum_{n_k \in b_j} w_{n_k} \leq C_k$ (The sum of all the weights for any capacity for any bin must be less than the corresponding capacity for that bin.)

Note that in the Multi-Capacity Bin Packing Problem the resources consumed by each VM accumulate, up to some maximum. Items are placed on each corner to show that

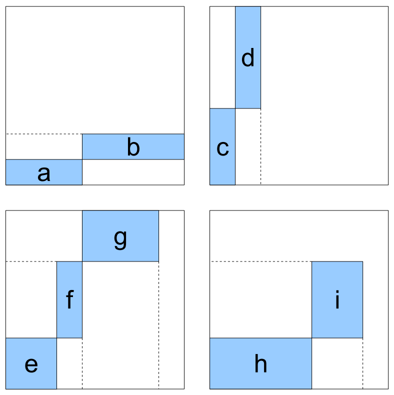

Fig. 3. An inefficient Multi-Capacity Bin Packing solution.

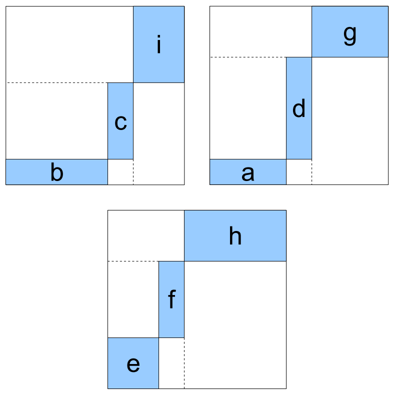

Fig. 4. An efficient Multi-Capacity Bin Packing solution.

the sum of the weights for any one type has to be less than the corresponding bin capacity. The weights on items in the same bin are additive. For all resources, the sum of all weights on items of that particular resource must be less than the corresponding resource capacity on the server.In VMAP, resources used by one VM can not be used by any VM on the same physical hardware.

As with the conventional Bin Packing Problem, the objective is to minimize the number of bins. Figures 3 and 4 illustrate this principle. Figure 4 packs the exact same items as are found in Figure 3 in three bins instead of four. This means one fewer server drawing power.

The Multi-Capacity Bin Packing Problem lends itself better to the problem of assigning VMs onto servers than the conventional Bin Packing Problem. In the conventional Bin Packing Problem, there is no way to model the multiple different resources that are available on a server, such as CPU, RAM and disk bandwidth. Because the Multi-Capacity Bin Packing Problem lends itself well to modeling the many different resources of a server, we use this problem formulation in our optimization techniques.

D. Genetic Algorithms

Bin Packing Problems have been solved with Genetic Algorithms (GAs). There is no rigorous definition for GAs [11]; they derive much of their inspiration from Darwinian biological processes. In GAs, individuals represent candidate solutions to the problem. These candidate solutions explore the solution space by undergoing processes similar to those of biological organisms. The simplest form of genetic algorithm involves three types of operators:

- **Selection–**Individuals in the population are selected for crossover with other individuals. Usually, selection is based on elitism, where the more fit individuals are selected more often than less fit individuals.
- **Crossover–**Two individuals in the population exchange subsections of their candidate solution with each other to create new offspring.
- **Mutation–**After crossover, each individual has a probability of having their candidate solution modified slightly.

III. Description of Model

As described in Section II, we model VMAP by using the Bin Packing Problem. However, there are a few differences between the Bin Packing Problem and VMAP which were identified in Section I. The model that we will use for VMAP is based on the Multi-Capacity Bin Packing Problem, as discussed in Section II-C, and it also uses probabilistic estimates of loads, which we will discuss here.

A. Probabilistic Estimates

The second way that VMAP is different from the conventional Bin Packing Problem is that the loads VMs exhibit on servers are not known completely when initial assignments are made. Even though these loads are not completely known ahead of time, probabilistic estimates can be made for loads on VMs. Therefore, we treat loads as probabilistic. There are at least two ways in which system administrators can derive probabilistic estimates for loads on VMs.

1) If the system administrators have reason to believe that VM loads can be characterized as a type of known probabilistic distribution, this problem becomes the problem of parameter estimation. Using data, it is possible to estimate the parameters of a parametric probabilistic model using known methods such at methods of moment or maximum likelihood estimation.
2) If system administrators do not know the probabilistic distribution which describes the load on VMs, a *nonparametric model* can be used. We will describe nonparametric distributions in more detail in Section VI.

We assume that the probabilistic distribution on the future load for each resource for each VM is to the algorithm ahead of time. Recall as well that in the Multi-Capacity Bin Packing Problem, weights are deterministic and known instead of probabilistic and unknown. This means that if the Multi-Capacity Bin Packing Problem is to be used as a model, some estimate or expectation of the probabilistic distribution must be given to the algorithm solving VMAP. In this section, we will explore how to make this estimate from probabilistic distributions of the load.

Recall that the *probability density function (pdf)* $f(X)$ of a random variable X describes the relative likelihood of X

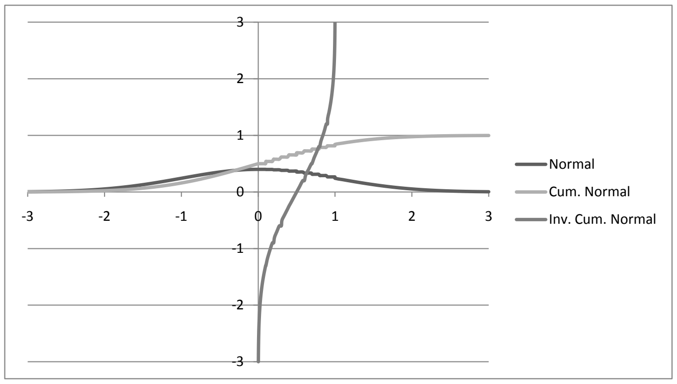

Fig. 5. The pdf, cdf and icdf for the normal distribution with mean $\mu = 0$ and variance $\sigma^2 = 1$.

to occur at a given point in the observation space. Recall, also, that the *cumulative distribution function (cdf)* $F(X)$ of a random variable X is defined for a number χ by:

$$F(\chi) = P(X \leq \chi) = \int_{-\infty}^{\chi} f(s)ds \tag{1}$$

where $f(s)$ is the likelihood associated with the random variable X obtained from the pdf f at s. The pdf for the normal distribution with mean $\mu = 0$ and variance $\sigma^2 = 1$ is shown in Figure 5.

The cdf of the normal distribution the same distribution with mean $\mu = 0$ and variance $\sigma^2 = 1$ is also shown in Figure 5. The cdf $P(X \leq \chi)$ is the probability that X is less than or equal to χ. It answers the question of "What is the likelihood of getting any load less than a specified load for a particular resource for a VM?" However, even though this metric may be useful, a better question to ask may be in the opposite order. "What is the maximum load y for a given likelihood z, such that $F(y) \leq z$?" This question can be answered by using the inverse cumulative distribution function.

The *inverse cumulative distribution function* (icdf) or *quantile function* returns the value below which random draws from the given cumulative distribution function would fall, $p * 100$ percent of the time. That is, it returns the value of χ such that

$$F(\chi) = Pr(X \leq \chi) = p \tag{2}$$

for a given probability $0 < p < 1$.

Using the icdf, we can specify a percentile value and obtain a corresponding load which can be passed to the assignment algorithm. Using the value from a high percentile will result in a high load being passed to the assignment algorithm, and tend to make the assignment more robust to random variation. Lower percentiles result in assignments more likely to become over loaded. Using the quantile function to decide which load to use means that the algorithm designer can incorporate any level of robustness or aggressiveness into the algorithm. Figure 5 shows the inverse cumulative distribution function for the normal distribution with mean $\mu = 0$ and variance $\sigma^2 = 1$. We call the load derived from the icdf value the *derived load.* The derived load is the load used by the bin packing algrorithm when a concrete value must be used.

IV. Initial Assignment Algorithms

Many companies and organizations do not use a structured approach to the initial placement of VMs onto servers. Even though a system administrator may view a summarization of the load for each VM, and place VMs onto servers using that summarization, we have not found any formalization of a model for VMAP, nor any algorithm meant to make an initial placement of VMs to servers. Because this is a new model, and the model derives its roots in the Bin Packing Problem, we present some well-known algorithms that solve the Bin Packing Problem for consideration. We will describe and compare four assignment algorithms in this paper–Worst Fit, First Fit, Permutation Pack [10] and Reordering Grouping Genetic Algorithm [21].

A. Worst Fit

The worst fit algorithm for bin packing considers each item in order. For each item, first, it considers only bins that already have at least one item placed in them. It places the item in the bin into which it fits which has the most amount of free space. If the item does not fit in any of the bins considered, it is placed into a new bin.

The worst fit algorithm is considered in this paper because of its tendency to leave a bit of space in every bin that it uses. This extra space may be helpful when observed loads on VMs exceed what the assignment algorithm expected.

Finding the emptiest bin is easy to do in the conventional Bin Packing Problem as each item only has one weight. The sum of the weights of all the items in any one bin can be used to give a number describing the fullness of a bin. However, finding the emptiest server in VMAP is not as obvious because each VM has multiple loads which it exerts on the server. In order to compare the emptiness of one server to another's, there must be a way to combine the different loads. We use a Euclidean distance metric to compare the amount of free space in two different bins.

B. First Fit

The first fit algorithm for bin packing is a way to place an initial set of VMs on servers. In the first fit packing algorithm, items are arranged in order (often decreasing order). Bins are also arranged in a list. Each item is then considered in order and placed into the first bin into which it fits.

C. Permutation Pack

Permutation Pack (PP) attempts to find items in which the largest w components are exactly ordered with respect to the ordering of the corresponding smallest elements in the current bin [10]. Let R_i denote the remaining space in the ith capacity of a particular bin. If $d = 2$ and $R_1 < R_2$, then we look for an item n such that $w_{n_1} < w_{n_2}$. If no item is found, the requirements are continually relaxed until one is found. One of the weaknesses of Permutation Pack is the running time. If all permutations are considered, it runs in $O(d!\,n^2)$ where d is the number of resource types and n is the number of items to be packed. We refer the reader to Leinberger et al. [10] for further description of the algorithm.

D. Reordering Grouping Genetic Algorithm

Reordering Grouping Genetic Algorithm (RGGA) is a genetic algorithm that was developed for the Multi-Capacity Bin Packing Problem [21]. RGGA has been shown to solve the Multi-Capacity Bin Packing Problem quickly, developing good solutions. RGGA represents an instance of the bin packing problem not only as an assignment of items to bins, but also as a list of items to be first fit packed. RGGA's crossover operator is an exon shuffling crossover as defined by Rohlfshagen et al [13]. RGGA uses a mutation operator that swaps two items in the first fit list $\frac{1}{3}$ of the time, moves an item from one spot to another in the first fit list $\frac{1}{3}$ of the time, and eliminates a bin, reinserting the contents $\frac{1}{3}$ of the time. This last idea is the mutation operator used by Faulkenauer in his Grouping Genetic Algorithm [7].

RGGA was shown to find optimal solutions to the bin packing algorithm in fewer iterations than the leading genetic algorithms in literature. As well, RGGA was shown to generate very good solutions to even large problem sizes of the Multi-Capacity Bin Packing Problem.

V. Metrics to Determine Success

In this section, we investigate two different metrics to determine the level of success of an initial VM assignment. These two metrics are total number of servers used and the proportion of servers over capacity.

A. Number of Servers Used

The total *number of servers* used is defined as the number of servers upon which VMs are placed. This metric is useful in determining the tightness of a particular assignment. An assignment which places its VMs on fewer servers will likely save energy in the long run. Aggressive assignment algorithms often maximize this metric.

B. Proportion of Server Resources Over Capacity

A server resource is over capacity if VMs on the server request more of that resource than is available on the server. For example, if the sum of the total amount of RAM requested by all the VMs on a particular server is greater than the amount available on the server, then the RAM on the server would be considered over capacity. In order to calculate the proportion of server resources over capacity, we divide the sum of all total server resources over capacity by the sum of all total server resources. The algorithm for calculating the *proportion of servers over capacity* is shown in Equation 3.

$$\frac{\sum_{b_i \in B} \sum_{j \in b_i} F(\sum_{w_{j_k} \in \vec{w}_k} w_{j_k}, C_k)}{\sum_{b_i \in B} \sum_{j \in b_i} 1} \tag{3}$$

where $F(X, Y)$ returns 1 if X is greater than Y and 0 otherwise.

Conservative VM assignment algorithms perform worse with regard to this metric, because they, on average, have less allocated resources per server. If a particular VM uses more server resources than was allotted to it, the server might not be over capacity if the algorithm chose to leave extra room. Algorithms that are more conservative when assigning VMs also yield solutions with a greater total number of servers.

VI. Experimental Setup

Because the contribution of this paper is the model proposed for VMAP, we wish to validate this model in our results by showing that the modifications we made to the conventional Bin Packing Problem are indeed helpful in modeling VMAP.

In our experiments, we did exactly what we expect real system administrators to do with our work. We deployed various VMs to a cluster of computers, gathered data on resource utilization of these VMs, generated an assignment for these VMs to servers, and carried out that assignment with virtualization software. The VMs that we created were of the form such that 8-12 of them would fit on a physical server. Because deployments in real data centers normally have 8-12 VMs per physical server [6], we expect our results to generalize well to other clusters.

We use the data itself as a nonparametric statistical distribution. The mean of this distribution can be computed by finding the mean of the data. The variance of the distribution can be found by finding the variance of the data. This distribution can be sampled by picking a data point with uniform probability. The icdf of this distribution can be found for any percentile by sorting the data, multiplying the percentile used by the number of data points, and returning that particular data point.

With this nonparametric distribution for the load on each VM, we were able to generate new assignments of VMs to servers. We ran each assignment algorithm for varying icdf values. We show the predicted performance for varying icdf values in our results. After simulating the assignment of VMs to servers, we predicted the number of servers and proportion of servers over capacityfor the case if we actually carried out the assignment. In order to obtain our simulated metrics, we sampled from the load distribution for each VM and assigned loads to VMs from their distributions. Using this data, we obtained predicted results for what a assignment would be like if we carried out the assigning on the servers [1].

After obtaining predicted results for assigning VMs to servers, we then reassigned VMs to servers, averaged the results and analyzed how closely our model showed what happened in real life. We show the results of these new assignments in our results. We used the Kernel-Based Virtual Machine (KVM) kernel virtualization infrastructure, the qemu processor emulator, and the libvirt virtualization management tool. In our data set, we used VMs with varying CPU and RAM loads. We kept track of the loads on the VMs and recorded the observed CPU and RAM utilization on host servers.

Because it is an option that the system administrator can tweak, we also show the predicted performance for different algorithms for varying icdf values. As mentioned in Section IV, raising the icdf value makes an assignment algorithm more conservative and lowering the icdf value makes an assignment algorithm more aggressive.

In our simulated experiments, we performed these steps:

1) The packing algorithm receives some type of distribution over the load for each virtual machine it needs to pack.
2) The packing algorithm derives a specific load using the distribution from step 1 and the icdf value used.
3) The packing algorithm develops an assignment of virtual machines to servers.
4) One specific load for each virtual machine is sampled from the load distribution for that virtual machine. This load is assigned to that virtual machine.
5) Metrics are gathered.
6) Steps 1-4 are repeated as many times as needed to achieve statistical significance for the test.

When we implemented RGGA, we used a maximum function evaluation count of 7500, a crossover rate of 0.8, and a mutation rate of 0.1. In the event that two individuals do not crossover, one of them is picked randomly and added directly to the next population. If an individual is not mutated, it is simply added as is to the next generation.

VII. Results

In our results, we wish to validate the probabilistic model proposed in Section III and the algorithms we proposed in Section IV. We will divide presentation of results in three parts. First, we will show how system administrators can use assignment algorithms with varying icdf values. We then analyze and present graphs that show in a practical standpoint number of servers used and the proportion of servers over capacity. Lastly, we analyze how closely our predictive model resembles what happens on real hardware by repacking VMs to servers.

Even though we do not directly incorporate probabilistic loads into any assignment algorithm proposed in this paper, we show results that help system administrators to indirectly incorporate probabilistic results into the assignment algorithm picked by applying the ideas presented in Section III-A. We systematically increased the icdf value and subsequently the derived load from the icdf value. Increasing this value increases the derived loads on VMs which the algorithm uses. As discussed earlier, lower values of percentile yield more conservative assignment algorithms and higher value of percentile yield more aggressive assignment algorithms.

Figure 6 shows the proportion of servers over capacity while Figure 7 shows the number of servers found by the different algorithms. The independent variable in both graphs is the icdf value used. Even though we present both figures separately, they must be interpretted together. One shows the number of servers used, while the other shows the proportion of servers over capacity. Algorithms which tend to use fewer bins will tend to look better in Figure 7, but look worse in Figure 6.

The icdf value is merely a parameter used to determine both the number of servers used and the proportion of servers over capacity. The icdf value used is really irrelevant because the icdf value picked by system administrators is a function of the number of servers used and the proportion of servers over capacity. Instead of showing this graph, we wish to combine Figures 6 and 7 so that we can see this type of interaction between the proportion of servers over capacity and the number of servers used.

In order to simplify our analysis, we joined the proportion of servers over capacity with the number of servers found using the percentile values to produce a joint graph. Using this graph, Figure 8, a system administrator can choose how many

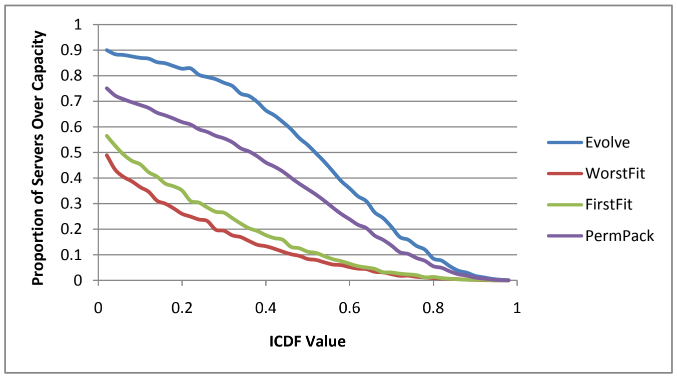

Fig. 6. A comparison of the proportion of servers over capacity.

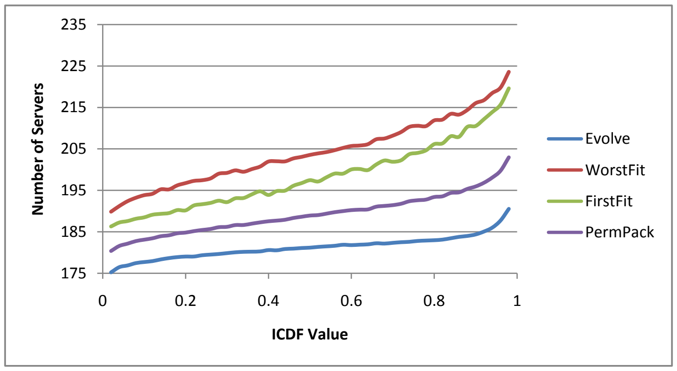

Fig. 7. A comparison of the the number of servers found.

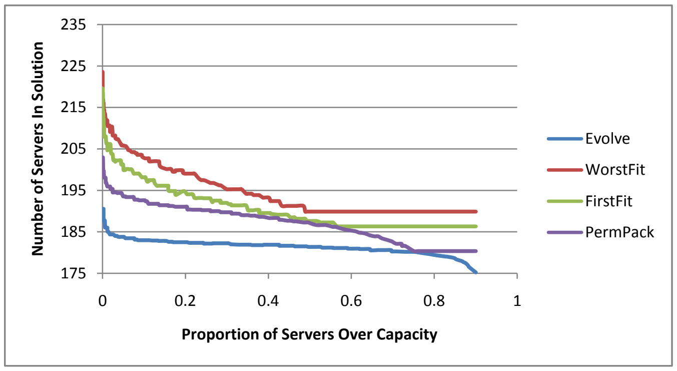

Fig. 8. A comparison of the proportion of servers over capacity with the number of servers found.

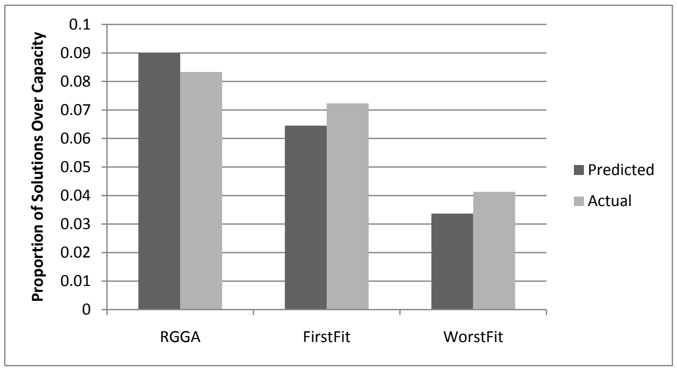

Fig. 9. A comparison of the proportion of solutions over capacity that our algorithm predicted and also the measured proportion of solutions over capacity when deploying real VMs to servers for the 80th cdf percentile.

servers on average will be in utilization or what proportion over capacity they are willing to tolerate in order to get the other parameter. This graph shows that for this particular configuration, RGGA

A. *Repacking to Servers*

Lastly, in order to validate the model we generated, we used the assignments generated to make assignments of real VMs to servers. We measured the proportion of server resources over capacity for all three assignment algorithms at the 80th cdf percentile. We show our results in Figure 9. The predicted proportion of server resources over capacity all algorithms is very close to the actual proportion of server resources over capacity measured when deploying VMs to servers. This gives validity to the model we suggest in this paper.

VIII. Conclusion

A novel model for Virtual Machine Assignment Problem is proposed. This model uses ideas from the conventional Bin Packing Problem, where servers are bins and VMs are items, with two variations. First, it allows multiple weights for each item and multiple capacities for each bin. The sum of all the weight of any one type in any bin must be less than that corresponding capacity in that bin. Second, our model proposes that VMs have probabilistic loads. The probabilistic loads should be incorporated into the assignment algorithm for best results. We, show the feasibility of using probabilistic loads with the assignment algorithm by modifying three known packing algorithms.

Adding probability theory helps system administrators to pick the correct percentile value representing the spot on the inverse cumulative distribution function representing the load of a resource. Small values for the icdf value yield more conservative assignment algorithms while larger values for the icdf value yield more aggressive assignment algorithms.

For the problems which we investigated, it seems that optimization algorithms like RGGA perform well.

References

[1] Virtualization datasets. http://aml.cs.byu.edu/ davidw/datasets, July 2010.

[2] F. Glover A. C. F. Alvim, C. C. Ribeiro and D. J. Aloise. A hybrid improvment heuristic for the one-dimensional bin packing problem. *Journal of Heuristics*, 10:4–27, 2004.

[3] Sandip Agarwala, Fernando Alegre, Karsten Schwan, and Jegannathan Mehalingham. E2eprof: Automated end-to-end performance management for enterprise systems. In *DSN '07: Proceedings of the 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, pages 749–758, Washington, DC, USA, 2007. IEEE Computer Society.

[4] Michael Bailey, Timothy Rostrom, and J. Ekstrom. Operating system power dependencies. In *Int. CMG Conference*, pages 15–20, 2008.

[5] Gong Chen, Wenbo He, Jie Liu, Suman Nath, Leonidas Rigas, Lin Xiao, and Feng Zhao. Energy-aware server provisioning and load dispatching for connection-intensive internet services. In *NSDI'08: Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation*, pages 337–350, Berkeley, CA, USA, 2008. USENIX Association.

[6] IT Knowledge Exchange. Virtual machines per server: A viable metric for hardware selection? http://itknowledgeexchange.techtarget.com/server-farm/virtual-machines-per-server-a-viable-metric-for-hardware-selection/, August 2008.

[7] E. Faulkenaeur. A hybrid grouping genetic algorithm for bin packing. *Journal of Heuristics*, 2:5–30, 1996.

[8] Narendra Karmarkar and Richard M. Karp. An efficient approximation scheme for the one-dimensional bin-packing problem. *Foundations of Computer Science, 1982. SFCS '08. 23rd Annual Symposium on*, pages 312–320, November 1982.

[9] J. G. Koomey. Estimating total power consumption by servers in the u.s. and the world. *Presented at the EPA stakeholder workshop at Santa Clara Convention Center*, February 2007.

[10] William Leinberger, George Karypis, and Vipin Kumar. Multi-capacity bin packing algorithms with applications to job scheduling under multiple constraints. In *ICPP '99: Proceedings of the 1999 International Conference on Parallel Processing*, page 404, Washington, DC, USA, 1999. IEEE Computer Society.

[11] M. Mitchell. *An introduction to genetic algorithms*. MIT Press, 1998.

[12] Ripal Nathuji and Karsten Schwan. Virtualpower: coordinated power management in virtualized enterprise systems. In *SOSP '07: Proceedings of twenty-first ACM SIGOPS symposium on Operating systems principles*, pages 265–278, New York, NY, USA, 2007. ACM.

[13] P. Rohlfshagen and J. Bullinaria. A genetic algorithm with exon shuffling crossover for hard bin packing problems. *Proceedings of Genetic And Evolutionary Computation Conference*, 9:1365–1371, 2007.

[14] Louis P. Slothouber and Ph. D. A model of web server performance. In *In Proceedings of the Fifth International World Wide Web Conference*, 1996.

[15] Ying Song, Hui Wang, Yaqiong Li, Binquan Feng, and Yuzhong Sun. Multi-tiered on-demand resource scheduling for vm-based data center. In *CCGRID '09: Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid*, pages 148–155, Washington, DC, USA, 2009. IEEE Computer Society.

[16] Ying Song, Yanwei Zhang, and Yuzhong Sun. Utility analysis for internet-oriented server consolidation in vm-based data centers. *Cluster Computing, IEEE International Conference on*, 0:xvii, 2009.

[17] Shekhar Srikantaiah, Aman Kansal, and Feng Zhao. Energy aware consolidation for cloud computing. In *Proceedings of HotPower '08 Workshop on Power Aware Computing and Systems*. USENIX, December 2008.

[18] Mark Stillwell, David Schanzenbach, Frederic Vivien, and Henri Casanova. Resource allocation using virtual clusters. In *CCGRID '09: Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid*, pages 260–267, Washington, DC, USA, 2009. IEEE Computer Society.

[19] C. W. Szeto. On the modeling of www request arrivals. In *ICPP '99: Proceedings of the 1999 International Workshops on Parallel Processing*, page 248, Washington, DC, USA, 1999. IEEE Computer Society.

[20] VMware. Vmware green it energy efficiency, reduce energy costs with virtualization. http://nvd.nist.gov/nvd.cfm?cvename=CVE-2008-1368, September 2009.

[21] David Wilcox, Andrew McNabb, Kevin Seppi, and Kelly Flanagan. Virtual machine assignment with a reordering grouping genetic algorithm. *To be submitted to Congress on Evolutionary Computation*, 2011.