Automatic Parameter Tuning for Class-Based Virtual Machine Placement in Cloud Infrastructures

Claudia Canali, Riccardo Lancellotti
Department of Engineering "Enzo Ferrari",
University of Modena and Reggio Emilia,
Via Vivarelli, 10, Modena, Italy
Email: {claudia.canali, riccardo.lancellotti}@unimore.it

***Abstract*—A critical task in the management of Infrastructure as a Service cloud data centers is the placement of Virtual Machines (VMs) over the infrastructure of physical nodes. However, as the size of data centers grows, finding optimal VM placement solutions becomes challenging. The typical approach is to rely on heuristics that improve VM placement scalability by (partially) discarding information about the VM behavior. An alternative approach providing encouraging results, namely *Class-Based Placement* (CBP), has been proposed recently. CBP considers VMs divided in classes with similar behavior in terms of resource usage. This technique can obtain high quality placement because it considers a detailed model of VM behavior on a per-class base. At the same time, scalability is achieved by considering a small-scale VM placement problem that is replicated as a building block for the whole data center. However, a critical parameter of CBP technique is the number (and size) of building blocks to consider. Many small building blocks may reduce the overall VM placement solution quality due to fragmentation of the physical node resources over blocks. On the other hand, few large building blocks may become computationally expensive to handle and may be unsolvable due to the problem complexity. This paper addresses this problem analyzing the impact of block size on the performance of the VM class-based placement. Furthermore, we propose an algorithm to estimate the best number of blocks. Our proposal is validated through experimental results based on a real cloud computing data center.**

I. Introduction

The success of cloud computing is testified by the constant growth of cloud-based infrastructures. Demands in terms of storage and processing power for the cloud are expected to increase by two orders of magnitude in fifteen years [1]. Indeed, more and more applications are deployed over Infrastructure as a Service (IaaS) cloud infrastructures to cope with highly variable workloads following a on-demand, pay-as-you-go philosophy. However, the success of cloud computing, resulting in more and larger data-centers, creates new challenges at the level of infrastructure monitoring and management. An ever-increasing number of Virtual Machines (VMs) with variable demands in term of system resources must be placed over a large number of physical nodes. Monitoring VMs to understand the dynamics of resource demands represents a challenging problem when thousands of VMs are involved. The VM placement problem is an even more critical issue because it involves the solution of a bin-packing problem encompassing the whole data center. Ensuring a scalable and effective solution for the VM placement problem is currently a major challenge for the cloud computing industry.

State of the art approaches for the VMs placement problem typically fail to consider the actual behavior of VMs in terms of resource demands (i.e., they consider all VMs of the same *nominal size* equal to each other [2], [3]). When the actual behavior of each VM is taken into account, the placement problem becomes so complex that it must be solved using highly simplified heuristics, such as the First Fit Decreasing (FFD) algorithm [4]. In both cases the result is a low quality solutions of the VM placement problem that lead to a waste of cloud data center resources. Only recently, the authors proposed a novel approach, namely *Class-Based Placement* (CBP), that exploits similar behavior of classes of VMs (i.e., VMs hosting the same software component of the same application) to increase the scalability of the VM placement [5]. Instead of considering a single bin-packing problem for VM placement, the CBP approach splits the problem into small building blocks that are easy to solve and can be composed to reach a global solution. However, while the initial proposal of CBP can reduce the computational demand and achieve higher quality in the VM placement solution compared to existing alternatives, the study in [5] does not provide a complete analysis of the parameters that may affect its performance. In particular, the study does not take into account the impact of the building block size on the quality of the final solution. The trade-off should be clear: on one hand, a large number of small building blocks can obtain a benefit in terms of scalability at the expenses of a less efficient placement due to unused spare capacity within each block (capacity fragmentation); on the other hand, larger blocks tend to avoid fragmentation effects, but the underlying placement problem is much more demanding from a computational point of view.

The contribution of this paper is twofold: 1) we perform an in-depth analysis of the impact of the number of building blocks on the quality of the VM placement solutions; 2) we propose an algorithm to automatically determine the best number of blocks in order to improve the quality of the VMs placement.

We apply our proposal to a problem based on a real data center to evaluate the impact of the parameters of the CBP technique on the overall performance. Specifically we show how the size of the block affects the quality of the VM placement and we compare the solution quality achieved by the early proposal of CPB [5] with the new algorithm for block size estimation.

Our results demonstrate that the new proposal for block size estimation outperforms the previous algorithm, achieving

a solution quality very close to the lower bound of the VM placement problem.

The remainder of this paper is organized as follows. Section II describes the Class-Based Placement and provides a model to determine an appropriate number of blocks. Section III describes the results of the technique evaluation. Finally, Section IV concludes the paper with some final remarks and outlines open research problems.

II. VM Placement problem

We now provide a formal model for the VM placement problem. Specifically, we first outline the Class-based placement technique proposed in [5], that is reference scenario for our proposal. Next, we discuss a parameter that affects the performance of the CBP, that is the number of the small-scale placement problems that are the building blocks of the CBP technique. In particular, we outline the pros and cons of having few large problems *vs.* having many small problems. Finally, we propose an algorithm to determine the best number of building blocks for the placement problem.

A. Class-Based Placement

The application of our proposal to a cloud data center is based on the following two assumptions. First, we consider that the VMs placement is a periodic task that aims at mapping VMs over the physical nodes of the infrastructure, with the goal of minimizing the number of used nodes while satisfying the requirements of each VM in terms of resource demand. Second, we assume to be able to group VMs into classes with similar behavior, where VMs belonging to the same class exhibit similar resource requirements. The presence of classes of VMs with similar behavior represents a common condition that occurs every time an application is replicated over a distributed architecture for scalability and availability [6]. Even if the knowledge of replicated application deployment is not directly available to IaaS cloud providers, we can exploit proposals in literature that allow to cluster together VMs with similar behavior [7], [8], [9].

The basic idea is to reduce the global bin packing problem for VM placement, that operates on the whole data center, to a smaller problem involving only few VMs for each class. The reduced size of the problem allows us to solve to optimality the VM placement considering a multi dimensional formulation with a number of time intervals that would not be possible to consider for the global problem; then, the obtained solution can be replicated as a building block to determine the solution for the global VM placement problem.

Figure 1 depicts the periodic VMs placement in a cloud data center that adopts the proposed approach. We consider as the input of our process a prediction of the future resource demands for the next planning period. Resource demands are expressed for each class of VMs (we present them as "F_1", ..., "F_C" in Figure 1). We also consider to have a description of the infrastructure (e.g, the nodes on which the VMs are to be placed, marked with the letter "I") and we expect as the output a decision (letter "D") indicating the placement of the VMs over the physical nodes.

Let us consider a set $\mathbf{M}$ of VMs that have to be deployed on a set $\mathbf{N}$ of physical nodes. We assume that the VMs are divided

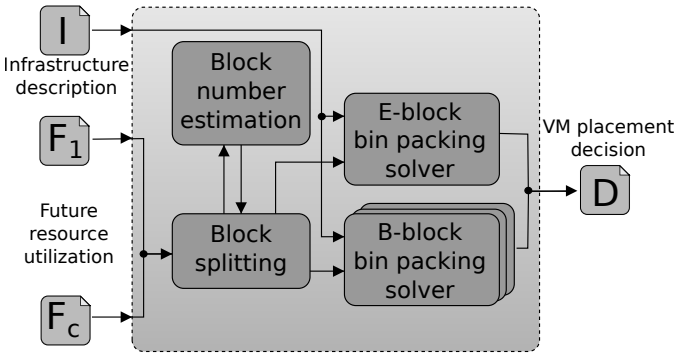

Fig. 1: Class-based VM placement technique

into a set $\mathbf{C}$ of classes, where all the VMs of a same class present similar resource requirements. Let $\mathbf{M_c}$ be the set of VMs belonging to class $c \in \mathbf{C}$. For the sake of the bin packing problem, we divide the next planning period into a set of time intervals $\mathbf{T}$ that are considered for the VM placement. The matrix R represents the resource requirements of the VMs over multiple observation time intervals. Although the most general version of the problem involves multiple resources of the VMs, such as CPU, memory, network, and disk [10], [11], [12], we limit our model to a single resource (CPU utilization), that is a typical bottleneck resource for cloud infrastructures [13]. However, it is worth to note that the inclusion of multiple resources in our model represents a straightforward extension. Since VMs belonging to the same class are characterized by similar resource demand, we can define the resource demand of a generic VM belonging to class c ($c \in \mathbf{C}$) for the time interval t ($t \in \mathbf{T}$) as $R_{c,t}$. Furthermore, V_n represents the available CPU capacity on node n ($n \in \mathbf{N}$).

The traditional approach to address the VM placement problem is to solve a multi-dimensional bin packing problem (MBP). However, solving the global problem is likely to become computationally infeasible for medium-large data centers. To improve the scalability of the VM placement, a possible solution is to simplify the bin packing problem, for example by increasing the length of the time intervals considered for resource demand estimation: this means reducing the cardinality of the set $\mathbf{T}$ and, consequently, the number of constraints of the optimization problem. Unfortunately this solution tends to reduce the quality of the solution, leading to the use of more physical nodes with respect to the optimum [5]. An alternative solution is to exploit the Class-based placement approach proposed in [5].

In Class-based placement, the global set of VMs is divided in $\bar{b}$ B-blocks composed by the same number of VMs for each class, while the remaining VMs form the E-block. A block number estimator (shown in Figure 1) determines the number of the B-blocks. For each class $c \in \mathbf{C}$, each B-block contains a set $\mathbf{B_c} \subset \mathbf{M_c}$ of VMs belonging to class c. The remaining set of VMs $\mathbf{E_c}$, that are not assigned to any B- block, is assigned to the E-block.

Since all the VMs of a same class present similar resource requirements, the placement solution computed for a single B-block can be replicated for all the B-blocks. We can thus formulate the optimization problem for the generic B-block as:

$$\min \sum_{n \in \mathbf{N}} O_n \tag{1}$$

subject to:

$$\sum_{n \in \mathbf{N}} I_{n,m} = 1 \qquad \forall m \in \bigcup_{c \in \mathbf{C}} \mathbf{B_c} \tag{2}$$

$$\sum_{c \in \mathbf{C}} \sum_{m \in \mathbf{B_c}} R_{c,t} \cdot I_{n,m} \leq V_n \cdot O_n \qquad \forall n \in \mathbf{N}, \forall t \in \mathbf{T} \tag{3}$$

$$I_{n,m} = \{0,1\} \qquad \forall n \in \mathbf{N}, \forall m \in \bigcup_{c \in \mathbf{C}} \mathbf{B_c} \tag{4}$$

$$O_n = \{0,1\} \qquad \forall n \in \mathbf{N} \tag{5}$$

Where O_n is a binary decision variable that discriminates if a physical node n in the data center is on or off, $I_{n,m}$ is a binary decision variable that decides if VM m is allocated on node n. Expression 1 is the objective function of the optimization problem that aims to minimize the number of used nodes. Due to the set of constraints 2, every VM is allocated exactly on one physical node. The set of constraints 3 expresses the bound that on each node the allocated VMs must not exceed the overall capacity of the node for every considered time interval. Finally, the sets of constraints 4 and 5 model the boolean nature of the decision variables.

A similar optimization problem applies to the E-block problem.

B. Block size estimation

We now focus on how VMs are assigned to the B-blocks and to the E-block. To this aim, the parameter $\overline{b}$ plays a major role, hence the selection of the right value for $\overline{b}$ is a critical factor for the performance of the proposed CBP technique. The impact of $\overline{b}$ over the consolidation process is twofold. On one hand, as $\overline{b}$ is reduced, the size of the problem in the B- and E-blocks increases. This may have a detrimental effect on the solvability of the VM placement problem due to the computational cost of the large optimization problems for the B-blocks. On the other hand, as $\overline{b}$ grows, we tend to have very small problems, where the amount of unused resources of the nodes in each B-block becomes relevant. In this case we observe a fragmentation effect that may reduce the quality of the solution (the number of physical nodes used is much higher than the optimum). The identification of the best value of $\overline{b}$ must solve a trade-off between computational cost and solution quality, ensuring that the splitting of the VM placement problem is feasible.

To determine the best value for the $\overline{b}$ parameter, we follow an iterative approach trying to distribute VMs between B-blocks and E-block until a set of constraint is satisfied. This is the task carried out by the block number estimation box in Figure 1. We designate with b^* the value of $\overline{b}$ identified by our algorithm.

Let x_c be the number of VMs belonging to class c ($c \in \mathbf{C}$) that are in the B-block. Referring to the problem in Section II-A, $x_c = |\mathbf{B_c}|$. In a similar way we define $y_c = |\mathbf{E_c}|$.

The values of x_c and y_c can be computed as:

$$x_c = |\mathbf{B_c}| = \left\lfloor \frac{|\mathbf{M_c}|}{\overline{b}} \right\rfloor \quad \forall c \in \mathbf{C} \tag{6}$$

$$y_c = |\mathbf{E_c}| = |\mathbf{M_c}| \% \overline{b} \quad \forall c \in \mathbf{C} \tag{7}$$

For each considered value of $\overline{b}$ we define the following constraints:

$$\sum_{c \in \mathbf{C}} x_c \geq \sum_{c \in \mathbf{C}} y_c \tag{8}$$

$$\sum_{c \in \mathbf{C}} x_c \leq S \tag{9}$$

$$x_c \geq 1 \qquad \forall c \in \mathbf{C} \tag{10}$$

Constraint 8 requires that the overall number of VMs for the E-block is less or equal to the size of a B-block. This constraint is motivated by the need to avoid a block splitting where the B-block remains small and the E-block becomes a huge and intractable problem. Constraint 9 places a maximum size on the VMs in a B-block. This bound is important because, in previous studies [5] we found that as the problem size grows, the bin-packing problem becomes unmanageable and cannot be solved. This observation motivates the upper bound on the B-block size and provides an estimate for the threshold S: in our experimental setup we will consider $S = 300$ that is the threshold value found in [5]. It is worth to note that we do not need to place a bound for the size of the E-block due to constraint 8. Constraints 10 requires the B-block to contain at least a VM for each class.

In our iterative approach, we start with a value of $\overline{b} = \left\lceil \frac{|M|}{S} \right\rceil$. This initial value descends from constraint 9: a lower value of $\overline{b}$ would automatically violate this condition. If we find a solution to the problem, then we have an acceptable block splitting for the found value of b^*. Otherwise, we increment $\overline{b}$ and we try again to solve the optimization problem until we find b^*. The maximum possible value for $\overline{b}$ is $L = \min(\{|\mathbf{M_c}|, \forall c \in \mathbf{C}\})$: any higher value of $\overline{b}$ would violate the inequality in constraint 10.

III. EXPERIMENTAL RESULTS

In this section we start describing the setup of our experiments, then we discuss the results regarding the proposed placement technique, with a detailed analysis of the impact of different values for the $\overline{b}$ parameter.

A. Experimental setup

We obtain an extensive dataset from a private cloud data center. The set contains up to 1200 VMs traces for the resource usage of Web/application/database servers and ERP applications, where the VMs belongs to 44 different classes, with each class containing from 8 to 50 VMs. We use our traces as the future resource utilization for the VM placement problem (see Figure 1). For our experiments we consider the CPU resource, that is well-know to be the bottleneck resource for this type of applications [13]. The resource usage is measured in intervals of 5 minutes, that is a setup consistent with other experiments in literature [14].

We consider multiple scenarios characterized by different numbers of VMs to be placed on the physical nodes of the virtualized data center. In particular, we consider a VMs set size ranging from 400 to 1200 VMs. For each VM the CPU utilization is in the range [0%-100%] with an average value of 54%. For each physical node the CPU capacity is 800%, meaning that each node can host 8 VMs with CPU utilization of 100%. For each scenario, we compare different consolidation models operating over a planning period of 24 hours. The proposed Class-Based Placement (CBP) is solved with 288 five-minutes time intervals and is evaluated for different values of the $\overline{b}$ parameter. When evaluating the traditional MBP model we consider different setups where the length of the intervals for resource requirements ranges from 5 minutes to 24 hours. We also consider a First Fit Decreasing (FFD) heuristic [4] that is used to solve very large problems [5]. The experiments are run on 2.4 GHz, 16 cores Intel Xeon with 16 GB RAM, using IBM ILOG CPLEX 12.6 as the optimizer solver[1].

As a metric for the VMs placement quality, we consider the number of physical nodes that are required for the allocation [10], [15]. The number of nodes for each solution is expressed with respect to an estimation of the optimal solution for the considered scenario. The MBP model with five minute time interval (MBP-5min) represents a lower bound for all the feasible allocations, as this consolidation model exploits all the available information to find an optimal solution. However, the number of variables and constraints for this model increases rapidly with the VMs set size, producing an optimization problem instances whose computation takes extremely long times or does not produce any feasible solution due to the huge main memory requirements, that may finally cause the solver to abort the optimization processing. For this reason, we use the objective function value of the LP relaxation of the MBP-5min consolidation model (1) as a lower bound for the optimal number of physical nodes to use. In other words, we relax the boolean nature of the decision variables, assuming that parts of a VM can be assigned to different physical nodes. This allocation is obviously not feasible from a technical point of view but can be easily computed, hence we exploit it as a convenient lower bound for any feasible allocation [10].

It is worth to note that for many problems the resolution of the MPB consolidation models may take long times, such as hours or days, even for a limited number of time intervals. For that reason, we used a time limit of 30 minutes (1800 seconds) for each problem and considered the best integer solution found as the solution of the placement problem, as commonly done in similar research studies [10], [16].

B. Experimental results

In the first experiment we evaluate the impact of different values of the $\overline{b}$ parameter, which represents the number of B-blocks, on the solution quality of the proposed Class-based placement technique. For each VMs set size, we force the $\overline{b}$ parameter to range from 2 to the maximum allowed value L, which is the cardinality of the smallest class. For each value of $\overline{b}$, we evaluate the number of VMs in B-blocks and E-blocks, and the quality of the solutions in terms of VMs placement. We achieve similar results across all the scenarios: for space reasons, we do not show all the results, but we discuss the case of 1200 VMs as an example of the most interesting and large scenario considered in our experiments.

Figure 2 shows the number of VMs in B- and E-blocks, and the solution quality as $\overline{b}$ ranges from 2 to 10, which is the cardinality of the smallest class in the 1200 VMs scenario. The graph also shows the line corresponding to the threshold $S = 300$ that we use to define the optimal value of $\overline{b}$.

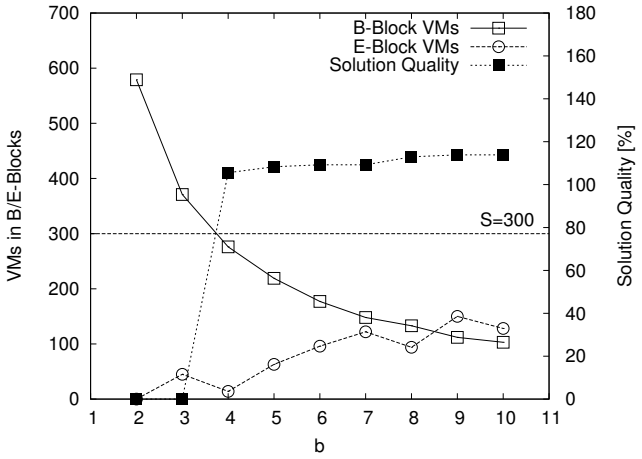

Fig. 2: Evaluation of 1200 VMs set size

In the graph we observe that the number of VMs in the B-blocks decreases as the number $\overline{b}$ of blocks increases. For $\overline{b}$ equal to 2 and 3, the solver was not able to reach a feasible integer solution within the 30 minutes limit (solution quality = 0), confirming our assumption that the problem with more than 300 VMs in a block is hardly solvable with constraints of 5-minutes time intervals. We also observe that the best solution is achieved for $\overline{b} = 4$, that corresponds to the value of b^* identified by the algorithm described in Section II-B; the number of required physical nodes slightly grows for higher values of $\overline{b}$. We recall that the solution quality is measured as the number of physical nodes required for the allocation with respect to the objective function value of the LP relaxation of MBP (5-min). In other words, a solution quality of 110% means that the allocation required 10% physical nodes more with respect to the solution of the LP relaxation problem. It is important to note that for this scenario, as for all the VMs set sizes not reported here, the value b^* corresponds to the number of B-blocks the leads to the best quality solution for the VM placement problem.

The second experiment compares the quality solutions obtained by the proposed Class-Based Placement, exploiting the automatic determination of the best $\overline{b} = b^*$, with that achieved by the same consolidation model using a number of B-blocks equal to the cardinality of the smallest VM class (L) (as proposed in [5]). We also consider as a term of comparison the solution achieved by state-of-the-art solutions based on Multiple Bin Packing (MBP) or FFD heuristic applied to the global placement problem. Table I shows the solution qualities for the considered consolidation models for each VM set size. For the CBP consolidation models (first and second columns) we report the considered $\overline{b}$ value. In the last column of the table, along with the solution quality, we report the

[1] www.ibm.com/software/commerce/optimization/cplex-optimizer/

state-of-the-art consolidation model that achieved the solution: we observe that for very large scenarios (1200 VMs) only the FFD heuristic is able to find a feasible integer solution, while feasible solutions can be achieved by MBP models with an increasing number of time intervals as the VM set size decreases.

TABLE I: Solution quality [%]

VMs Set Size	Consolidation Models: Class-Based $\overline{b} = b^*$	Class-Based $\overline{b} = L$	State of the art solution
400	103.2 ($\overline{b} = 2$)	112.9 ($\overline{b} = 8$)	125.4 (MBP-1h)
500	104.5 ($\overline{b} = 2$)	115.9 ($\overline{b} = 8$)	120.3 (MBP-1h)
600	105.5 ($\overline{b} = 2$)	111.1 ($\overline{b} = 8$)	125.0 (MBP-12h)
700	106.6 ($\overline{b} = 3$)	110.3 ($\overline{b} = 8$)	131.8 (MBP-12h)
800	107.0 ($\overline{b} = 3$)	114.1 ($\overline{b} = 10$)	128.5 (MBP-12h)
900	105.1 ($\overline{b} = 3$)	108.8 ($\overline{b} = 8$)	131.2 (MBP-12h)
1000	104.5 ($\overline{b} = 3$)	106.8 ($\overline{b} = 10$)	127.3 (MBP-12h)
1100	107.7 ($\overline{b} = 4$)	114.3 ($\overline{b} = 10$)	133.1 (MBP-1d)
1200	105.5 ($\overline{b} = 4$)	113.8 ($\overline{b} = 10$)	131.2 (FFD)

The message from Table I is twofold. First, the proposed algorithm for the determination of b^* allows to obtain significant improvements in the solution quality achieved by the CBP model. Indeed, the quality of the solutions in the first column ranges from 103.2% to 107.7%, while for $\overline{b} = L$ the quality ranges from 106.8% to 115.9%. Second, we confirm that even in the latter case, CBP significantly outperforms the state-of-the-art solutions, that use at least 20% more nodes compared to the LP relaxation of the problem.

IV. Conclusions

In this paper we addressed the problem of parameter tuning in the Class-based placement technique. Specifically, we considered the parameter $\overline{b}$ that is the number of B-blocks in which the global VM placement problem is split. We pointed out a trade-off between using many small building blocks (with the risk of reducing the overall VM placement solution quality due to the fragmentation of the physical node resources over blocks) and using few large building blocks (with the risk of being unable to solve the optimization problem for the B-block). We provide an extensive evaluation of this trade-off providing the insight to understand how $\overline{b}$ impact on the quality of the CBP solution. Furthermore, we proposed a new algorithm for the estimation of a value b^* of B-blocks that can provide high quality for the global VM placement problem solution while preserving the resolvability of the problem. Our experiments confirm that our solution outperforms the initial proposal of CBP presented in [5] for every considered scenario.

References

[1] J. Gantz and D. Reinsel, "The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the far east," *IDC iView: IDC Analyze the Future*, 2012, http://www.emc.com/collateral/analyst-reports/idc-the-digital-universe-in-2020.pdf.

[2] B. Rochwerger, D. Breitgand, A. Epstein, D. Hadas *et al.*, "Reservoir - When one cloud is not enough," *IEEE computer*, vol. 44, no. 3, pp. 44–51, 2011.

[3] K. Mills, J. Filliben, and C. Dabrowski, "Comparing VM-Placement Algorithms for On-Demand Clouds," in *Proc. of IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*. IEEE, 2011, pp. 91–98.

[4] M. Kao, *Encyclopedia of Algorithms*. Springer, 2008.

[5] C. Canali and R. Lancellotti, "Exploiting Classes of Virtual Machines for Scalable IaaS Cloud Management," in *Proc. of the 4th Symposium on Network Cloud Computing and Applications (NCCA)*, Munich, Germany, Jun. 2015.

[6] M. Rabinovich and O. Spatscheck, *Web caching and replication*. Addison-Wesley Boston, USA, 2002.

[7] C. Canali and R. Lancellotti, "Improving Scalability of Cloud Monitoring through PCA-Based Clustering of Virtual Machines," *Journal of Computer Science and Technology*, vol. 29, no. 1, pp. 38–52, 2014.

[8] ——, "Exploiting ensemble techniques for automatic virtual machine clustering in cloud systems," *Automated Software Engineering*, vol. 31, no. 3, pp. 1–26, Sep. 2014.

[9] ——, "An Adaptive Technique to Model Virtual Machine Behavior for Scalable Cloud Monitoring," in *Proc. of IEEE Symposium on Computers and Communications (ISCC)*, Madeira, Portugal, Jun. 2014.

[10] T. Setzer and M. Bichler, "Using matrix approximation for high-dimensional discrete optimization problems: Server consolidation based on cyclic time-series data," *European Journal of Operational Research*, vol. 227, no. 1, pp. 62–75, 2013.

[11] W. Fang, X. Liang, S. Li, L. Chiaraviglio, and N. Xiong, "VMPlanner: Optimizing virtual machine placement and traffic flow routing to reduce network power costs in cloud data centers," *Computer Networks*, vol. 57, no. 1, pp. 179 – 196, 2013.

[12] R. Zhang, R. Routray, D. M. Eyers, D. Chambliss *et al.*, "IO Tetris: Deep storage consolidation for the cloud via fine-grained workload analysis," in *Proc. of IEEE 4th International Conference on Cloud Computing (CLOUD)*, Washington, USA, Jul. 2011.

[13] M. Andreolini, S. Casolari, and M. Colajanni, "Models and framework for supporting runtime decisions in Web-based systems," *ACM Transactions on the Web*, vol. 2, no. 3, pp. 1–43, 2008.

[14] B. Addis, D. Ardagna, B. Panicucci, M. S. Squillante, and L. Zhang, "A hierarchical approach for the resource management of very large cloud platforms," *IEEE Transactions on Dependable and Secure Computing*, vol. 10, no. 5, pp. 253–272, 2013.

[15] D. Breitgand and A. Epstein, "Improving consolidation of virtual machines with risk-aware bandwidth oversubscription in compute clouds," in *Proc. of IEEE INFOCOM*, Orlando, FL, March 2012.

[16] L. Zhang and D. Ardagna, "SLA Based Profit Optimization in Autonomic Computing Systems," in *Proc. of International Conference on Service Oriented Computing (ICSOC)*, New York, USA, Nov. 2004.