Selective resource characterization for evaluation of system dynamics

Sara Casolari, Michele Colajanni, Stefania Tosi
Department of Information Engineering, University of Modena and Reggio Emilia
{sara.casolari, michele.colajanni, stefania.tosi}@unimore.it

ABSTRACT

Management decisions to achieve peak performance operations, scalability and availability in distributed systems require a continuous statistical characterization of data sets coming from server and network monitors. Due to the increasing sizes of data centers and their continuous dynamic changes, the traditional approaches that work on all data sets in a centralized way are impractical. We propose a strategy for data processing that is able to limit the analysis of the large sets of collected measures to a smaller subset of significant information for a twofold purpose: to classify the collected data sets in few classes characterized by similar statistical behaviors, to evaluate the dynamics of the overall system and its most relevant changes. The proposed strategy works at the level of server resources and of significant aggregation of servers of the overall distributed system. Several experimental results demonstrate the feasibility of the proposed strategy that is validated in real contexts.

1. INTRODUCTION

Internet data centers are becoming quite complex infrastructures consisting of several distributed servers enhancing virtualization capabilities and supporting several heterogeneous services that are characterized by varying demands, and different performance and availability objectives. In similar contexts, resource management systems have to apply multiple models for system dynamics evaluation including anomaly detection, performance quality control, prediction of future trends, estimation of expected capacity needs (e.g., [27, 17]). Most management decisions are taken on the basis of continuous analysis and characterization of data sets coming from system monitors and referring to different resources (e.g., CPU, processors, memories, storage elements, virtual machines, network) and various metrics (e.g., utilization, response time, throughput). Even in a medium-sized data center, several thousands of resource data sets may reach the management system that should analyze, model and treat them at different temporal scales in order to guarantee reliability, scalability, and to avoid performance degradation and overloads. It would be possible to pass all resource data collected by the system monitors to the management system only in the case of short data sets consisting of few monitored samples. As this becomes impracticable when the data sets are collected over long periods, it is necessary to characterize all (and only) the relevant data carrying on useful information. This paper gives a contribution in the direction of simplify the complexity of managing huge amounts of heterogeneous and highly variable data sets. The idea is to extract a concise representation of how the most important components and sets of components are acting and to signaling main changes to the management system of a modern data center consisting of distributed servers.

Data collected from resource monitors generate highly heterogeneous sets (e.g., [38, 2, 3, 7]): some of them are characterized by periodic or linear behaviors; others present spikes and noises so high that any immediate application of models for trend, anomaly detection, and forecast analysis is impracticable. The first goal is to characterize resource heterogeneity in some statistical classes that can guide management decisions. In particular, models and algorithms for performance management, prediction, capacity planning, anomaly detection can work if they can distinguish random from deterministic patterns in data sets [10, 3, 38, 6]. The algorithms for resource characterization that are oriented to analyze the entire data set of each server (e.g., [36, 25, 34, 13]) are impracticable. Other approaches that aim to reduce the dimension of the problem, such as [37, 28, 2], are specific for data mining contexts or are oriented to problems different from ours, such as correlation analysis and anomaly detection. We aim to reduce the analysis of the huge amount of monitored data to focus the investigation on a subset of relevant information by *selecting* just the most important data sets. In this study, the importance is a measure of the impact that each resource has on the overall system behavior. We use the results of data set reduction as a mean to pass just relevant statistical information to a management system that is facilitated in the application of the most suitable decision strategy.

- We take advantage of data heterogeneity by *distinguishing* deterministic from random behaviors of servers. The idea is that former data sets can be used for taking reliable decisions and applying adequate models for forecasting and anomaly detection to the specific server; the latter sets can be used for detections only after the application of some filtering model.
- We adopt a similar analysis at the level of subsystems, that is, significant subsets of servers, where the context defines the right partition.
- We evaluate the *dynamics* of the system by signaling to the management framework just the most significant changes in system dynamics.

The remainder of this paper is structured as follows. Section 2 presents the foundations of the proposed strategy. Section 3 describes how we select the most relevant from the least important information. Section 4 presents the method for the statistical characterization of the resources. Section 5 analyzes system dynamics as a basis for signaling just the most relevant changes to the management system. Section 6 evaluates the computational complexity, while in Section 7 we analyze related work. Concluding remarks and future work are presented in Section 8.

2. PROPOSED STRATEGY

The proposed strategy is oriented to the selection and the statistical characterization of the resources of data center's servers. The measures related to system resources are typically huge in size because they refer to long monitoring periods, variable and heterogeneous. Hence, the first goal for management purposes is to discriminate between relevant and less relevant information. We consider variability as the main statistical property that quantifies the degree of activity of a process and its importance in overall system behavior as in [5]. The index of variation of the data sets related to each monitored process is denoted by the *energy*. We use this measure as a means to distinguish between data sets carrying relevant and not-relevant information. In order to reduce the dimensionality of a data set analysis and to consider just the most relevant data sets in terms of energy, we refer to the *Principal Component Analysis* (PCA) model [1] that, unlike other models for dimension reduction [16, 26, 24, 19], is able to express the intrinsic structure of a data set in terms of variance and requires no prior knowledge about statistical properties of data sets.

By considering only the relevant sources of information, we characterize the server statistical behavior emerging from the monitored data. Only relevant information is passed to the statistical analyzer that aims to classify data sets in one of the following classes:

- *deterministic*, presenting systematic trends and periodic patterns that are predictable and possible to model;
- *random*, manifesting isolated spikes and/or stochastic noises;
- *negligible*, giving a minor contribution to the overall system activity.

This classification allows a management system to apply the right model to the right data sets, as well as reducing the complexity of system management with null or minimum loss of information. The negligible category does not carry meaningful information for system management. Its contribution on system activity has a low impact and therefore a management framework can ignore data sets belonging to this class. When a data set is characterized by a prevalently deterministic behavior, a management system can adopt several models for prediction, anomaly detection, event identification. On the other hand, servers characterized by data sets with spiky and noisy behaviors complicate management. These data sets require some preliminary aggregation and filtering treatment before their utilization. In the most difficult cases they result useless or unfeasible for management even after the application of some data treatments.

Unlike existing solutions for server characterization that analyze each collected data set [36, 25, 34, 13], the proposed solution classifies server behaviors by working on a subset of information. Our approach, that is based on PCA [1], aims (1) to extract from data sets the most relevant information, where the importance is measured as a function of the impact of each resource on the overall system behavior, and (2) to focus the classification analysis on this smaller subset of information. In this way, we are able to limit the computational complexity of the analysis, and to allow the integration of server characterization results with system management frameworks operating in large data centers.

The proposed strategy achieves the above mentioned goals through the steps outlined below, and detailed in the next three sections of this paper.

1. **Initial analysis.** First, we evaluate the impact of each data set collected by system monitors on the overall system activity, so to distinguish *relevant* from *not-relevant* sources of information (Section 3).

2. **Server characterization.** Considering only the relevant sources of information, we then characterize the server statistical behavior emerging from the monitored data. We choose to classify the statistical behavior in three main classes. The idea is to focus the analysis only to those classes that can facilitate system management (Section 4).

3. **Evaluation of system dynamics.** The evaluation of the system dynamics looks at the overall behavior of the system. A significant variation in the behavior of relevant sources of information denotes a significant change in the system that is useful to signal for better management (Section 5).

It is important to observe that a similar strategy can be applied to evaluate the system dynamics of resources belonging to groups of servers up to the whole system. By looking at the overall energy of the system, we can dynamically discriminate whether the system has a prevalent deterministic behavior and therefore it is manageable, or if the system is mainly driven by a random behavior that makes the system hard to treat and to model. A significant variation in the amounts of deterministic and random energies of the system denotes a significant change in system activity that is useful to signal for adapting management decision to changing environments.

A final remark is in order. Since the proposed strategy does not make any a-priori assumption on the statistical properties of the data sets, as it is required by other approaches [6, 2], it can characterize any type of resource and metrics, such as CPU utilization, memory and swap usage, disk usage, network packet rate. In the present version, it operates on homogeneous resource metrics belonging to distributed servers belonging to the same data center. Extensions to heterogeneous resources are possible, but out of the scope of this paper.

3. INITIAL ANALYSIS

In large data centers, the data sets collected from the monitored system resources form a multivariate structure characterized by multiple dimensions each consisting of several samples. The proposed approach reduces the computational complexity of the analysis when the number of samples is huge with respect to the number of monitored resources (that is, dimensions). In these large structures, a reliable management decision requires a preliminary identification of the most relevant information. A common approach is to find a new coordinate space consisting of a lower dimensionality that is representative of the original space [18]. When a structure can be approximated through a smaller number of dimensions in a way that minimizes the error and the loss of information, we can refer to the smaller number of dimensions as the *structure intrinsic dimensionality.*

In literature, there are many reduction models to extract the intrinsic dimensionality of system resource measures, such as the Principal Component Analysis (PCA) [1], the Correspondence Analysis [16], the Factor Analysis [26], the Non-Negative Matrix-Factorization [24], the Independent Component Analysis [19]. Among these algorithms, we choose the PCA model because it is able to express the intrinsic structure of a data set in terms of variance without requiring any prior knowledge about the statistical characteristics of data sets.

At time t, the sets of sampled measures originate a matrix X_t corresponding to the highly dimensional multivariate structure. It is a n x p matrix, where n is the number of considered samples (for example, $n = 2016$ if we consider a five minutes sampling in a one week period), and p is the number of considered system resource measures. Our approach is more efficient than existing solutions when we consider long monitoring periods where $n > p$. As resource measures are statistically heterogeneous, it is useful to normalize them as data sets characterized by zero mean and unit variance, as suggested in [4]. The PCA is a coordinate transformation method that maps X_t on a new set of axes [18] called components. Calculating the components is equivalent to solve the symmetric eigenvalue problem for the matrix $\Psi_t = X_t^T X_t$ that is, a measure of the covariance of the data sets deriving from samples. In practice, each component v_i is the i-th eigenvector computed from the spectral decomposition of Ψ_t [4]:

$$\Psi_t v_i \quad = \quad \lambda_i v_i \qquad i = 1, \ldots, p \tag{1}$$

where λ_i is the eigenvalue corresponding to the eigenvector v_i and represents the magnitude of the variation along each component v_i. Previous work uses the term *energy* obtained as a sum of quadratic terms to quantify how well the eigenvectors describe the original data set [31, 28]. On the other hand, we compute the energy by considering the output of the PCA and evaluate the energy σ_i associated to the component v_i through the percentage of variation related to its eigenvalue λ_i, that is, $\sigma_i = \frac{\lambda_i}{\sum_{j=1}^{p} \lambda_j} * 100$. As Ψ_t is symmetric positive definite, its eigenvectors are orthogonal and the corresponding eigenvalues are non-negative real numbers. By convention, the eigenvectors are unit norm and the eigenvalues are arranged from large to small, so that $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_p$.

In this new dimensional space, a *dimension* u_i $(i = 1, \ldots, p)$ is defined as a vector of size n obtained by the contribution of the data set matrix X_t and of the component v_i. As suggested in [4], this vector is normalized to unit length by dividing it by $\sqrt{\lambda_i}$. Hence, for each principal component v_i we have:

$$u_i \quad = \quad \frac{X_t v_i}{\sqrt{\lambda_i}} \qquad i = 1, \ldots, p \tag{2}$$

Through the above equation all server behaviors weighted by v_i produce one dimension of the transformed data. Hence, the vector u_i can capture the temporal variation common to all server measures along the component v_i. Since the components are ordered with respect to their contribution to the overall energy, u_1 captures the strongest temporal trend that is common to all server measures, u_2 captures the second strongest trend, and so on.

The energy characterizing each component is used to reduce the data dimensionality: we exclude the least *energetic* components of the structure, that is, we ignore the less variable components. There are several methods [21] to choose how many components it is useful to retain and to exclude. They include graphical approaches, such as the *scree plot* [8], the quantitative tests based on PCA singular values [22], the percentage of variability expressed by the principal components [20]. In this paper, we pursue this last criterion by determining in advance the amount of residual variability that we want to tolerate and by excluding residual components. A typical threshold [23] establishes that the cumulated percent on variation expressed by the retained dimensions should be higher than the 90-percentile. Our method does not depend on the choice of this threshold, hence in this paper we consider not-relevant the dimensions contributing to the overall variance of the data set for less than 10%.

When a smaller set of r dimensions are relevant, we can have interesting implications: X_t can be mapped on an r-dimensional subspace of $\mathbb{R}^p$, and $r \ll p$ is the intrinsic dimension of X_t. This result allows the method to distinguish between the *relevant* dimensions $U_t = \{u_1, \ldots, u_r\}$ that are contributions shared by all resource measures on the r principal components, and the *not-relevant* dimensions $\overline{U_t} = \{u_{r+1}, \ldots, u_p\}$ corresponding to the least important components in terms of system energy.

We validate the proposed approach by taking into account long data traces collected from a data center consisting of distributed servers that support different types of applications, including Web sites, databases, access controls, CMS, mail server, management software. In this data center, system monitors collect resource measures every five minutes referring to CPU utilization, primary and secondary memory-related metrics, network activities. For the sake of simplifying a complex procedure, we limit our presentation to a week of data sets referring to the CPU utilization of 50 servers hosting Web sites providing static, dynamic and interactive content. These data sets originate a matrix X_t with $n = 2016$ rows and $p = 50$ columns.

We initially evaluate the energy σ_i related to each dimension u_i in order to identify how many dimensions bring energy information to the data set. Figure 1 shows the cumulative distribution of the energy provided by each dimension [11] by reporting for each dimension u_i the proportion of the energy related to the previous dimensions that is, $\sum_{j \leq i} \sigma_j$.

Thanks to the rapid growth of the distribution curve in Figure 1, the 90-percentile of the energy of the overall system is captured just by the first nine dimensions that contribute to most server variability. In this example, our approach evaluates the first 9 dimensions as *relevant*, while the other 41 dimensions are marked as *not-relevant* from the point of view of the energy. This is an important result of the proposed methodology applied to a real data center because it confirms that, in terms of energy, the resource measures all together form a structure with an intrinsic dimensionality of $r = 9$, that is much lower than the original number (50)

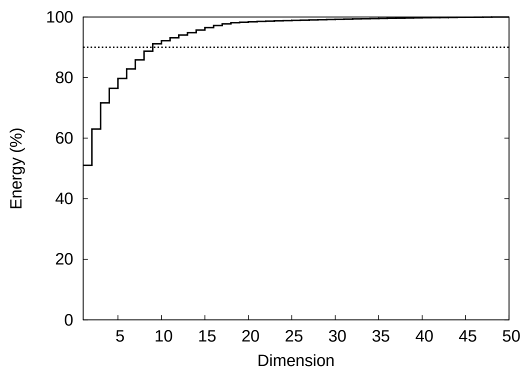


Figure 1: Cumulative distribution of energy.

of the considered set of data.

As a confirmation of this result, we evaluate the degree of correlation between the original resource traces sampled from the servers and the reconstructed data sets as a function of the number k of dimensions. We use the following equation for reconstruction:

$$X'_t \quad \approx \quad \sum_{i=1}^{k} \sqrt{\lambda_i} u_i v_i^T, \quad k = 1, \ldots, p \tag{3}$$

Figure 2 reports an example of the high correlation existing between original and reconstructed CPU utilizations of three servers as a function of the considered dimensions (k spans from 1 to 20). This figure evidences that the first 9 dimensions achieve a perfect positive linear correlation between original and reconstructed data sets for servers 7 and 11, and a very high correlation for server 3. These results confirm that a PCA-based methodology can be applied to identify the most relevant information for management decisions in large data centers.

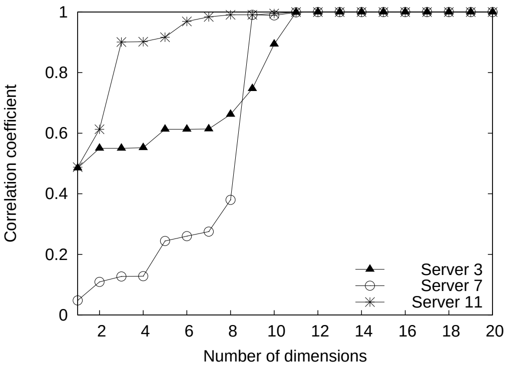


Figure 2: Correlation coefficient between original and reconstructed data sets as a function of the number of considered dimensions.

4. SERVER CHARACTERIZATION

The first application of the proposed methodology is to provide a statistical characterization of the server behaviors as *relevant* and *deterministic*, *relevant* and *random*, or *negligible*. For each data set corresponding to a server, we propose an algorithm activated periodically and based on the following steps. We consider the activation at a generic time t. The contributions in terms of energy allow the algorithm to distinguish between relevant dimensions U_t and not-relevant $\overline{U_t}$ dimensions.

Then, we distinguish the relevant dimensions between *correlated* U_t^C and *low-correlated* U_t^L by evaluating the autocorrelation function (ACF) of each relevant dimension U_t as in [9] and in many other contexts (e.g., [3, 6]). A quick decrease of the ACF means that the observed dimension values exhibit low (or null) autocorrelation. This is the case of dimensions capturing the random perturbations and/or spikes varying in time and intensity in the system behavior. On the other hand, a slow decay of the autocorrelation function indicates that the dimension shows a dependency among its values. The dimension typically exhibits trends, periodicity and seasonal fluctuations due to the diurnal activity, as well as the difference between weekday and weekend activity of the system [9].

We use the classification of the dimensions to distinguish three behavioral values for each data set referring to a server resource. The idea is to add all the contributions for each type of dimensions and then to evaluate the maximum of them. This method is applied to each server resource k as following.

- $D_t^{Ser}[k] = \sum_j v_j[k], \ \forall j \mid u_j \in U_t^C$ denotes the contributions provided by the relevant correlated dimensions;
- $R_t^{Ser}[k] = \sum_j v_j[k], \ \forall j \mid u_j \in U_t^L$ denotes the contributions provided by the relevant low-correlated dimensions;
- $N_t^{Ser}[k] = \sum_j v_j[k], \ \forall j \mid u_j \in \overline{U_t}$ accumulates the contributions of the not-relevant dimensions.

For each server resource k, we evaluate the highest term that is, $M_t^{Ser}[k] = max\{D_t^{Ser}[k], R_t^{Ser}[k], N_t^{Ser}[k]\}$. In such a way, we can characterize each server resource on the basis of its main behavioral value:

$$Server\ k = \begin{cases} Deterministic & \text{if } M_t^{Ser}[k] = D_t^{Ser}[k] \\ Random & \text{if } M_t^{Ser}[k] = R_t^{Ser}[k] \\ Negligible & \text{if } M_t^{Ser}[k] = N_t^{Ser}[k] \end{cases} \tag{4}$$

It is important to compare the proposed strategy against those obtained through other approaches that analyze each monitored data set, such as [36, 25, 34, 13]. Existing models require computationally expensive analyses to evaluate correlation [9], probability distributions [14], and mean loads [12]. The complexity becomes excessive especially when each data set consists of several entries, and it is even increased by the necessity of applying some pre-filtering technique when the measures are highly variable, as it is typical in the considered contexts.

For validation purposes, we consider again the data set referring to the CPU utilization of the 50 servers introduced in Section 3. We compare the classification results of servers achieved by the proposed strategy and other approaches that analyze each data set. The results of these models are reported in the second column of Table 1. The results obtained through the proposed approach are shown in the three *Proposed strategy* columns. If we compare the classes with bold indexes to the server characterization arising from the baseline characterization reported in the second column, we see a perfect correspondence of results. The proposed strategy is able to identify the main statistical behavior of 50 servers through a classification and a statistical analysis of only the most relevant 9 dimensions of the data set. On this small set of dimensions, we do not need to apply any time consuming filtering technique since PCA itself allows us to isolate perturbations from data. Thanks to the dimensionality reduction, we can integrate the server characterization with any decision support for system management in large distributed systems.

Let us detail the steps of this algorithm applied to $r = 9$ relevant dimensions and 41 not-relevant dimensions. We apply the ACF to the $r = 9$ relevant dimensions. For server characterization purposes, we need to be selective in the choice of the correlated dimensions and we set a high cut-off value of 0.8. The ACF gives that 5 dimensions are *correlated* and 4 are *low-correlated* (Figure 3). The goal is now to determine which dimensions actually bring information to which CPU traces of the data set. By evaluating the D_t^{Ser}, R_t^{Ser} and N_t^{Ser} values, we compute the extent to which correlated U_t^C, low-correlated U_t^L and not-relevant $\overline{U_t}$ dimensions are present in the data set.

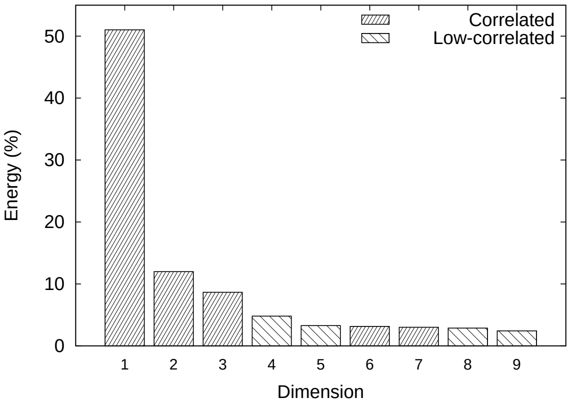


Figure 3: Percentage of energy and classification of the relevant dimensions.

The proposed strategy classifies the characteristics of the server k through the three behavioral values $D_t^{Ser}[k]$, $R_t^{Ser}[k]$ and $N_t^{Ser}[k]$ without the need of applying multiple statistical tests to all data sets. In the results reported in the *Proposed strategy* columns of Table 1, each column highlights in bold text the $M_t^{Ser}[k]$ value for each server. This value denotes the behavioral characterization that is assigned by the proposed strategy. For example, the baseline analysis of the server 3 computes a mean load utilization of filtered data of 6.39% that classifies it as a negligible server. Our strategy reaches the same characterization through the evaluation of only three behavioral values: a predominant $N_t^{Ser}[3]$ value of 0.1866 characterizes the server as under-loaded, thus avoiding the analysis and treatment of the entire data set.

In this paper we do not report all possible exploitations of the proposed strategy. Just to give another example, we observe that considering the amount of servers characterized by negligible activities offers a valuable information for management optimizations. In our case study, we have that 20 over 50 servers are prevalently negligible, that is, mainly underloaded during two weeks of observation. In a similar context, a resource management system that logically pools all server resources would increase the overall system utilization and diminish operational costs by turning off some unnecessary servers.

5. EVALUATION OF SYSTEM DYNAMICS

Several important problems related to system design and management require a continuous evaluation of the dynamics of the overall system or of significant parts of it when the data center is so huge that global measures are unfeasible or useless. In this paper, we use the term *system* to refer to the overall set of servers or to a suitable subset of homogeneous server resources. (The extension to heterogeneous resources is left to future work).

The motivation of a system's view comes from the observation that performance is affected not only by each server, but also by the interactions of components of multiple servers interacting and competing for the same resources. As a consequence, the system dynamics are the result of the superimposition of several contributions that often are not independent.

The main goal of our system dynamics evaluation is to define whether the behavior of the energy of the system is mainly driven by deterministic or by random behaviors, and to continuously check how this energy evolves at different evaluation steps. (We exclude the case of a data center that has a predominant component of negligible information in terms of energy because it would represent an unrealistic case of a poorly managed set of resources). In large data centers, where continuous changes and migrations of virtual machines are at the norm, we want to detect only relevant changes in system dynamics and not minor instabilities, thus preventing the activation of management procedures when there are changes that affect the behavior of few servers, but do not have an impact in overall system operations.

Our model evaluates periodically the proportion of the total system energy by focusing on the r most relevant dimensions (U_t) emerging from Section 3 instead of considering each component of the data set. For each time interval t, the proposed algorithm is based on the following steps.

1. We consider the data set of resource measures X_t collected over the time period $[t-1, t]$.

2. As for server characterization, we distinguish the relevant dimensions U_t between *correlated* U_t^C and *low-correlated* U_t^L dimensions.

3. We compute two values, D_t^{Sys} and R_t^{Sys}, that sum up the energy σ associated to the correlated and low-correlated dimensions, respectively:

$$D_t^{Sys} \quad = \textstyle\sum_j \sigma_j, \quad \forall j \mid u_j \in U_t^C \tag{5}$$

$$R_t^{Sys} \quad = \textstyle\sum_j \sigma_j, \quad \forall j \mid u_j \in U_t^L \tag{6}$$

D_t^{Sys} and R_t^{Sys} values capture the quantity of variation that resource measures X_t distribute between the deterministic and the random dimensions, respectively.

	Analysis of each data set	Proposed strategy				Analysis of each data set	Proposed strategy		
k	Behavior	D_t^{Ser}	R_t^{Ser}	N_t^{Ser}	k	Behavior	D_t^{Ser}	R_t^{Ser}	N_t^{Ser}
1	Random	0.2211	**0.2379**	0.0181	26	Negligible	0.0569	0.0402	**0.1912**
2	Negligible	0.0006	0.0009	**0.1297**	27	Negligible	0.0071	0.0186	**0.2028**
3	Negligible	0.0443	0.0334	**0.1866**	28	Negligible	0.0220	0.0268	**0.2112**
4	Random	0.0391	**0.3012**	0.1185	29	Random	0.0432	**0.1261**	0.1127
5	Negligible	0.0186	0.0116	**0.1955**	30	Random	0.0207	**0.2498**	0.0137
6	Negligible	0.0162	0.0128	**0.1938**	31	Deterministic	**0.3955**	0.0846	0.0927
7	Random	0.0615	**0.2379**	0.0181	32	Deterministic	**0.4183**	0.2324	0.0277
8	Negligible	0.0492	0.0370	**0.2030**	33	Deterministic	**0.3037**	0.2695	0.0293
9	Random	0.1160	**0.2206**	0.1236	34	Random	0.2114	**0.2388**	0.0166
10	Negligible	0.0005	0.0005	**0.1309**	35	Negligible	0.0478	0.0410	**0.1858**
11	Deterministic	**0.3539**	0.1612	0.0540	36	Negligible	0.0387	0.0353	**0.2114**
12	Negligible	0.0099	0.0049	**0.1960**	37	Random	0.0275	**0.3535**	0.0616
13	Negligible	0.0139	0.0122	**0.2052**	38	Negligible	0.0236	0.0842	**0.1214**
14	Deterministic	**0.3158**	0.2091	0.0150	39	Random	0.0611	**0.2059**	0.0287
15	Deterministic	**0.2302**	0.0792	0.0860	40	Deterministic	**0.3810**	0.2160	0.0421
16	Deterministic	**0.3057**	0.2396	0.0176	41	Deterministic	**0.2462**	0.0784	0.0815
17	Deterministic	**0.3368**	0.1016	0.0402	42	Deterministic	**0.3902**	0.0892	0.0960
18	Random	0.0357	**0.3350**	0.0712	43	Negligible	0.0003	0.0008	**0.1094**
19	Random	0.0685	**0.2085**	0.0071	44	Negligible	0.0039	0.0100	**0.1755**
20	Deterministic	**0.2755**	0.1612	0.0251	45	Deterministic	**0.3879**	0.1695	0.0458
21	Random	0.1170	**0.2858**	0.0239	46	Deterministic	**0.3002**	0.0716	0.0866
22	Random	0.0837	**0.3412**	0.0213	47	Negligible	0.0090	0.0198	**0.1659**
23	Deterministic	**0.3955**	0.0846	0.0927	48	Negligible	0.0221	0.0237	**0.2081**
24	Deterministic	**0.3163**	0.3154	0.0374	49	Negligible	0.0006	0.0003	**0.1388**
25	Negligible	0.0002	0.0006	**0.1037**	50	Deterministic	**0.2032**	0.0898	0.0844

Table 1: Characterization of servers behavior.

4. We compare the two values, so that we can characterize the system as *deterministic* if $D_t^{Sys} > R_t^{Sys}$, as *random* otherwise.

5. We compare the system dynamics at time t and $t-1$. A change is detected whenever the two evaluations differ significantly. For example, a change is signaled if $D_{t-1}^{Sys} > R_{t-1}^{Sys}$ and $D_t^{Sys} < R_t^{Sys}$. This means that the energy that characterizes the system dynamics passed from a deterministic behavior at time $t-1$ to a random behavior at time t.

The system characterization is fundamental for management purposes. For example, if at time t the contribution of correlated dimensions prevails, the system can be considered more manageable, in the sense that its system behavior can be predicted through forecasting models. On the other hand, if the system has a predominant random characterization, we have to adopt specific filtering techniques to reduce its randomness. This information about the prevalent randomness of the system is important for the management system, that is aware that it is impossible to extrapolate and forecast global behaviors.

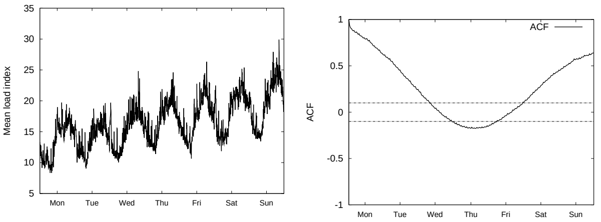
On the testbed presented in Section 3, the results shown in Figure 3 reveal that the most important sources of variation in CPU utilization are the correlated dimensions. Low-correlated dimensions are next in importance. This conclusion is confirmed by the results in Table 2 showing the percentage of total energy that can be assigned to each of the three dimension classes, including also the not-relevant class.

Correlated dimensions provide almost six times the contribution of the low-correlated class, and almost nine times the contribution of the not-relevant class. As $D_t^{Sys} > R_t^{Sys}$, we can consider that the distributed system has a prevalently deterministic behavior. This conclusion has important consequences for management. For example, we can assume that it is possible to represent the overall system behavior through a global index as a linear combination of homogeneous indexes (e.g., the CPU), similarly to the mean load index of all servers used in [38]. Such a global index of homogeneous and prevalently deterministic resources would be an acceptably accurate approximation of the system characterization that could be used for several management decisions in terms of trend analysis, forecasting and capacity planning models.

Correlated D_t^{Sys}	**Low-correlated** R_t^{Sys}	Not-relevant
77.79%	**13.36%**	8.85%

Table 2: Contributions of energy provided by the classes of dimensions (system's view).

If we evaluate the mean CPU utilization computed over the 50 servers of our testbed as a global index, then we obtain in Figure 4(a) that the system activity is dominated by periodic trends where increases during diurnal activity are followed by decreases during the night. These correlated fluctuations reflect the prevalently deterministic nature of the system. This visual interpretation is confirmed by the corresponding autocorrelation values reported in Figure 4(b). The slow decay of the ACF curve means that the global index exhibits high temporal dependence among its samples and that a strong deterministic component influences the mean CPU utilization behavior.



(a) Global index of system behavior

(b) Autocorrelation function

Figure 4: Example of global index in a prevalently deterministic system.

Besides a static characterization, the most important contribution is to identify major changes in system dynamics. To this purpose, we continuously evaluate how the deterministic and random system components evolve at different check points. When only minor instabilities with no impact on the overall system operations affect the system, the two percentages D_t^{Sys} and R_t^{Sys} of energy should remain almost unchanged. When the two percentages change significantly, we can assume that the system experienced some relevant change(s) that must be signaled to the management support. In this way, we selectively signal to system management only those changes having repercussions on the overall system, instead of activating management procedures for any variations in the server behaviors that in a similar context are continuously changing.

Figure 5 shows the evolution of the deterministic (white bars) and random (black bars) percentages of energy during 22 business days. Both white and black bars show a stable nature with small differences from day to day. When at $t = 19$ a new business application connected to a database was installed, the two percentages experience a sudden increase/decrease thus evidencing a significant change in system dynamics.

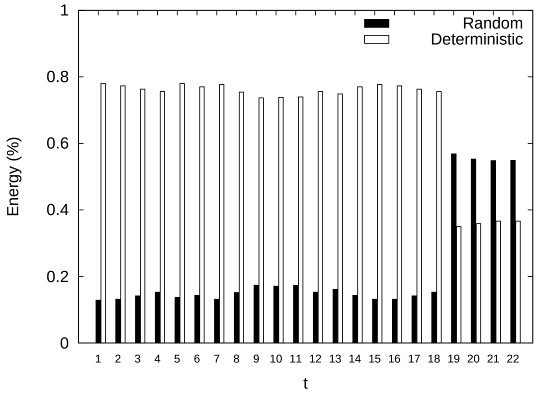


Figure 5: Evolution in time of deterministic and random percentages of energy.

This information can be passed to the management system because signals of relevant changes can guide operator decisions today and autonomic decisions tomorrow. The transition of the system dynamics from a deterministic to a random characterization does not provide a motivation by itself. Further investigations are necessary because several events can cause a change of system dynamics. In our approach, a change may be due to several servers changing their class of characterization (deterministic, random and negligible), and/or to relevant modifications of the index of variation (in this example, the standard deviation of the CPU utilization) of multiple servers. Hence, if the model evidences a change in system dynamics, such as that at $t = 19$, we have to evaluate the number of servers belonging to each class at time $t = 18$ and $t = 19$, and the standard deviation of the CPU utilization of each server in these two days. A system change from deterministic to random may be due to one or a combination of the following events:

1. an increase of the number of servers belonging to the random class;
2. an increase of the index of variation of random servers;
3. a decrease of the number of servers belonging to the deterministic class;
4. a decrease of the index of variation of deterministic servers.

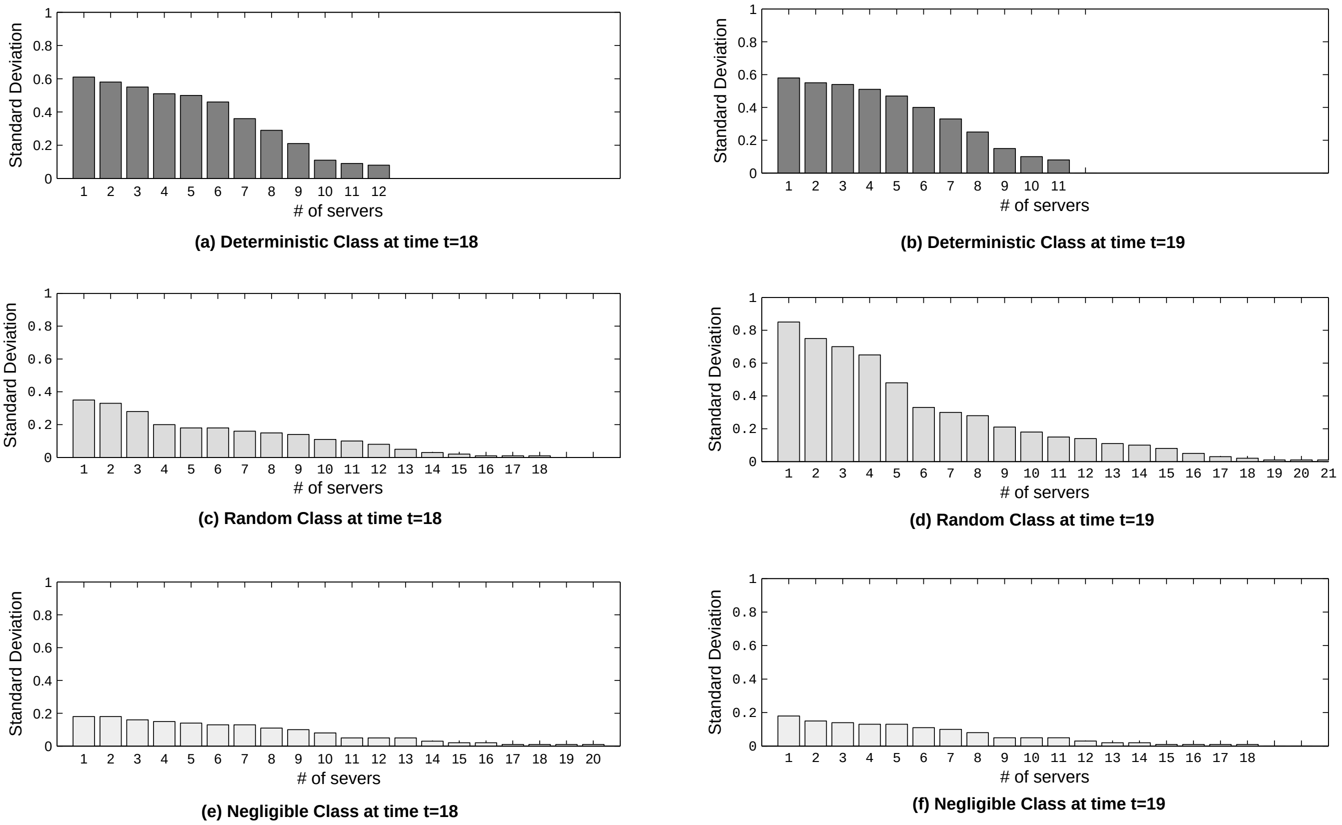
Dual events are at the basis of a system change from random to deterministic. The goal of the approach is to evaluate which event(s) caused a so relevant change of system dynamics. By taking as an example of application the testbed used for Figure 5, we evaluate the number of servers of the three classes and the CPU standard deviations of each server at time $t = 18$ and $t = 19$. Figure 6 reports the number of servers for each class on the x-axis, and the standard deviation of their CPU utilization on the y-axis.

By comparing the figures at time $t = 18$ and $t = 19$, we notice that some servers passed to the random class and that the indexes of variation of the servers of the random class increased significantly, while the analogous indexes remained almost stable for the deterministic and negligible classes. These results suggest that the main reason of the change in system dynamics at time $t = 19$ is due to a combination of factors: more servers are characterized by a random behavior and, even more important, there is a significant augment in the indexes of variation of the servers of the random class. This twofold event causes that the class of random servers achieves a major impact on the total energy of the system at the expenses of the deterministic class. This is an important conclusion for a management system that has to take some decisions as a consequence of a relevant change in system dynamics.

6. COMPUTATIONAL COMPLEXITY

In this section, we compare the computational complexity of the proposed approach against existing methods working on the entire set of monitored data. The computational complexities are summarized in Table 3, where n is the number of samples of each data set, p is the number of considered resources, and r is the number of relevant dimensions.

The PCA requires about $\mathcal{O}(p^3 + n * p^2)$ computations [15] for the eigenvalue decomposition of a covariance matrix, while the analysis of the statistical attributes of a set may



(a) Deterministic Class at time t=18

(b) Deterministic Class at time t=19

(c) Random Class at time t=18

(d) Random Class at time t=19

(e) Negligible Class at time t=18

(f) Negligible Class at time t=19

Figure 6: Evaluation of the number of servers and their standard deviation before and after the change.

Proposed strategy		**Analysis of each data set**	
Major processing steps	Computational complexity	Major processing steps	Computational complexity
PCA analysis of a nxp matrix	$\mathcal{O}(p^3 + n * p^2)$	Filtering of p sets of length n	$\mathcal{O}(p * nlog(n))$
Behavioral analysis of r sets	$\mathcal{O}(r * n^2)$	Behavioral analysis of p sets	$\mathcal{O}(p * n^2)$
Total	$\mathcal{O}(p^3 + n * p^2 + r * n^2)$	Total	$\mathcal{O}(p * n^2)$

Table 3: Computational complexity.

provide a number of operations spanning from logarithmic to exponential in the number of samples. In this evaluation, we consider the application of behavioral analyses having at most a quadratic computational cost. Hence, the total computational complexity of the proposed methodology is $\mathcal{O}(p^3 + n * p^2 + r * n^2)$.

On the other hand, the analysis of each data set needs a pre-filtering step on all monitored data. There are a lot of filtering models with different properties and complexities. In order to provide a reliable characterization, we consider the Fast Fourier Transform [29] requiring $\mathcal{O}(p * nlog(n))$ computations. Then, quadratic (or less) behavioral analyses are applied to all filtered data. Hence, the total computational complexity of the methods working on the entire data sets is about $\mathcal{O}(p * nlog(n) + p * n^2) \approx \mathcal{O}(p * n^2)$.

In Figure 7 we show the tendencies of the computational complexity for the proposed and existing approaches as a function of the number of considered resources p that are sampled every five minutes for five working days, that is, $n = 1440$. We can appreciate that by varying the dimension of p up to one thousand, the computational complexity of the proposed approach (line with filled circles in the figure) remains always lower than that of existing methods (line with squares) working on the entire data set. When the number of considered resources p increases and n remains constant, the analysis of each data set may become more efficient. However, it is important to consider that, in large data centers with tens of thousands of (virtual) machines, no management system analyzes at the same time all data sets. The typical strategy works hierarchically on subsets of components, where subsets are defined in terms of physical proximity or logical interactions. As long as we work on data sets referring to thousands of entries, the proposed strategy is much more efficient than the analysis operating on all data sets.

We should also consider that the performance of the proposed approach can be improved through the application of less expensive implementations of the Principal Component Analysis. For example, FastPCA [33] can diminish the dimensionality reduction complexity to $\mathcal{O}(p^2)$. Hence, its use in our strategy brings to a computational complexity that follows the curve denoted by star points in Figure 7.

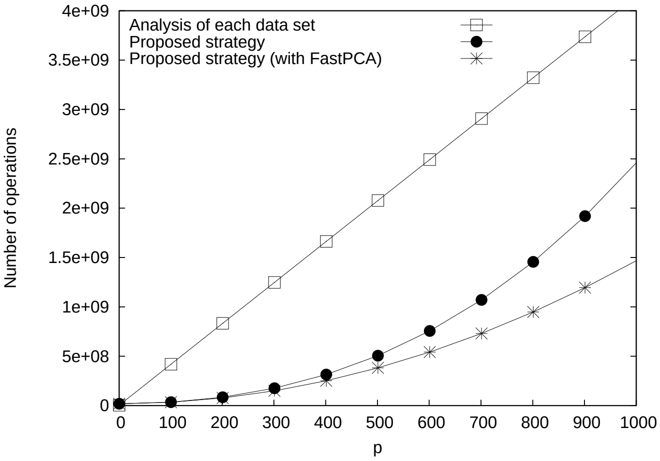


Figure 7: Analysis of the computational complexity.

7. RELATED WORK

Existing models and frameworks supporting management systems through server characterization are based on two assumptions: the number of data sets and the size of each of them allow the analysis of the entire set of data (e.g., [36, 25, 34, 13]); resource measures are characterized by statistically homogeneous behaviors and/or stable conditions (e.g., [2, 10, 35, 30, 13]). These assumptions are unrealistic in modern data centers accommodating varying demands for different services.

The number of resources that are monitored and the period of monitoring is growing [32, 38], hence the computational cost associated to the analysis of each data set is impracticable. In literature there are many reduction models oriented to limit the data that must be analyzed, such as the Principal Component Analysis (PCA) [1], the Correspondence Analysis [16], the Factor Analysis [26], the Non-Negative Matrix-Factorization [24], the Independent Component Analysis [19]. We propose an approach based on the PCA because it makes only weak assumptions on data characteristics and allows us to select the most important information on the basis of the so called energy. Thanks to these properties, PCA has been extensively used in many computer related contexts, especially for workload characterization [2] and for network traffic analysis [4]. In the context of system management, the work in [37] integrates PCA into a two-stage log processing method that is oriented to discover the statistically dominant patterns in the data set and thereby to identify possible anomalies. Another paper [28] uses PCA to identify interactions among the components of large production systems. Even if the goals of these works are different, they support the possibility of using PCA in managing large data sets and finding relevant information.

Other papers assume that the considered data sets are statistically homogeneous [2, 10, 35, 30, 13]. This assumption is invalid in the considered context because the data sets coming from different servers and system resources are quite heterogeneous in statistical terms and extremely variable. For this reason, our paper deals with the statistical heterogeneity of resource measures that must be considered as an intrinsic feature of modern data centers [38, 2] leveraging virtualization technologies and providing multiple Internet-based services subject to varying demands. We do not filter heterogeneity, but we classify resources on the basis of their deterministic, random or negligible behavior so to determine whether and which model can be better adopted by a management system.

8. CONCLUSIONS

Modern data centers require management decisions that should continuously analyze huge amount of heterogeneous data sets coming from system and network monitors. Understanding the statistical characterization of data sets of each server and the evolution of the system dynamics is crucial for system management. However, the increasing size of monitored data and their high variability limit the applicability of existing approaches working on all data sets.

We propose a novel strategy for supporting the decisions of the system management that, by taking system monitor data, is able to select the most relevant data sets, statistically characterize their behavior, and to evaluate the dynamics of the overall system. All experiments carried on real data sets demonstrate that the proposed strategy finds useful applications in system management related to modern Internet data centers. The next step is oriented to the characterization of server behavior and system dynamics through the combination of different metrics and resources.

Acknowledgements

The authors acknowledge the support of MIUR-PRIN project DOTS-LCCI Dependable Off-The-Shelf based middleware systems for Large-scale Complex Critical Infrastructures.

9. REFERENCES

[1] H. Abdi and L. Williams. Principal component analysis. *Computational Statistics*, 2010.

[2] B. Abrahao and A. Zhang. Characterizing application workloads on cpu utilization in utility computing. *Technical Report HPL-2004-157, Hewlet-Packard Labs*, 2004.

[3] M. Andreolini, S. Casolari, and M. Colajanni. Models and framework for supporting run-time decisions in web-based systems. *ACM Transaction on the Web*, Vol. 2, No. 3, 2008.

[4] L. Anukool, K. Papagiannaki, M. Crovella, C. Diot, E. D. Kolaczyk, and N. Taft. Structural analysis of network traffic flows. *Joint International Conference on Measurement and Modeling of Computer Systems*, 2004.

[5] M. Argollo de Menezes and A. Barabasi. Separating internal and external dynamics of complex systems. *Physical review letters*, Vol. 93, No. 6, 2004.

[6] Y. Baryshnikov, E. Coffman, G. Pierre, D. Rubenstein, M. Squillante, and T. Yimwadsana. Predictability of Web server traffic congestion. In *Proc. of the 10th International Workshop on Web Content Caching and Distribution*, Sophia Antipolis, FR, 2005.

[7] S. Casolari, F. Lo Presti, and S. Tosi. An adaptive model for online detection of relevant state changes in internet-based systems. *Performance Evaluation*, 2011.

[8] R. Cattell. *The scree test for the number of factors.* Multivariate behavioral research. Psychology Press, 1966.

[9] C. Chatfield. *The Analysis of Time Series: An Introduction.* Chapman and Hall, 1989.

[10] P. Dinda and D. O'Hallaron. Host load prediction using linear models. *Cluster Computing*, Vol. 3, No. 4, 2000.

[11] J. E. Gentle. *Computational Statistics.* Statistics and Computing. Springer, 2009.

[12] D. Gmach, J. Rolia, L. Cherkasova, G. Belrose, T. Turicchi, and A. Kemper. An integrated approach to resource pool management: Policies, efficiency and quality metrics. In *Proc. of the IEEE International Conference on Dependable Systems and Networks (DSN)*, 2008.

[13] D. Gmach, J. Rolia, L. Cherkasova, and A. Kemper. Workload analysis and demand prediction of enterprise data center applications. In *Proc. of the 2007 IEEE 10th International Symposium on Workload Characterization*, 2007.

[14] R. Gnanadesikan and M. B. Wilk. Probability plotting methods for the analysis of data. *Biometrika*, Vol. 55, No. 1, 1968.

[15] G. H. Golub and C. F. Van Loan. *Matrix Computations (Johns Hopkins Studies in Mathematical Sciences).* The Johns Hopkins University Press, Oct. 1996.

[16] M. Greenacre. *Correspondence analysis in practice.* Chapman & Hall/CRC, 2007.

[17] P. Hoogenboom and J. Lepreau. Computer system performance problem detection using time series models. In *Proc. of the USENIX Summer 1993 Technical Conference on Summer technical conference.* USENIX Association, 1993.

[18] H. Hotelling. Analysis of a complex of statistica variables into principal components. *Journal of Educational Psychology*, Vol. 24, No. 7, 1933.

[19] A. Hyvärinen and E. Oja. Independent Component Analysis: algorithms and applications. *Neural Networks*, Vol. 13, No. 4-5, 2000.

[20] J. Jackson. *A user's guide to principal components.* Wiley series in probability and mathematical statistics: Applied probability and statistics. Wiley, 1991.

[21] I. Jolliffe. *Principal Component Analysis.* Encyclopedia of Statistics in Behavioral Science, 2005.

[22] H. F. Kaiser. An index of factorial simplicity. *Psychometrica*, Vol. 39, No. 1, 1974.

[23] R. Khattree and D. Naik. *Multivariate data reduction and discrimination with SAS software.* Wiley series in probability and statistics. SAS Institute Inc., 2000.

[24] D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, Vol. 401, No. 6755, 1999.

[25] X. Liao, H. Jin, and X. Yuan. Espm: An optimized resource distribution policy in virtual user environment. In *Future Generation Computer Systems*, pages 1393–1402. Springer-Verlag, 2010.

[26] K. Mardia, J. Kent, and J. Bibby. *Multivariate Analysis.* Probability and Mathematical Statistics. Academic Press, 1995.

[27] D. A. Menascé, V. A. F. Almeida, and L. W. Dowdy. *Capacity planning and performance modeling: from mainframes to client-server systems.* Prentice-Hall, Inc., 1994.

[28] A. J. Oliner and A. Aiken. Online detection of multi-component interactions in production systems. *Dependable Systems and Networks, International Conference on*, 0:49–60, 2011.

[29] A. V. Oppenheim, R. W. Schafer, and J. R. Buck. *Discrete-time signal processing.* Prentice Hall, 1999.

[30] G. Pacifici, W. Segmuller, M. Spreitzer, and A. Tantawi. Cpu demand for web serving: Measurement analysis and dynamic estimation. *Performance Evaluation*, Vol. 65, No. 6-7, 2008.

[31] S. Papadimitriou, J. Sun, and C. Faloutsos. Streaming pattern discovery in multiple time-series. In *In VLDB*, pages 697–708, 2005.

[32] J. Rolia, L. Cherkasova, M. Arlitt, and A. Andrzejak. A capacity management service for resource pools. In *Proc. of the 5th international workshop on Software and performance*, New York, NY, USA, 2005.

[33] A. Sharma and K. K. Paliwal. Fast principal component analysis using fixed-point algorithm. *Pattern Recognition Letters*, 28(10):1151 – 1155, 2007.

[34] N. Singh and S. Rao. Energy optimization policies for server clusters. 2010.

[35] N. Tran and D. Reed. Automatic ARIMA time series modeling for adaptive I/O prefetching. *IEEE Trans. Parallel and Distributed Systems*, Vol. 15, No. 4, 2004.

[36] R. Van Der Krogt, J. Feldman, J. Little, and D. Stynes. An integrated business rules and constraints approach to data centre capacity management. In *Proceedings of the 16th international conference on Principles and practice of constraint programming*, pages 568–582. Springer-Verlag, 2010.

[37] W. Xu, L. Huang, A. Fox, D. Patterson, and M. Jordan. Online system problem detection by mining patterns of console logs. In *Proceedings of the 2009 Ninth IEEE International Conference on Data Mining*, ICDM '09, pages 588–597. IEEE Computer Society, 2009.

[38] X. e. a. Zhu. 1000 islands: an integrated approach to resource management for virtualized data centers. *Cluster Computing*, Vol. 12, No. 1, 2008.