A Correlation-based Methodology to Infer Communication Patterns between Cloud Virtual Machines

Claudia Canali
Department of Engineering “Enzo Ferrari”
University of Modena and Reggio Emilia
Via P. Vivarelli 10
41125, Modena, Italy
claudia.canali@unimore.it

Riccardo Lancellotti
Department of Engineering “Enzo Ferrari”
University of Modena and Reggio Emilia
Via P. Vivarelli 10
41125, Modena, Italy
riccardo.lancellotti@unimore.it

ABSTRACT

The VMs allocation over the servers of a cloud data center is becoming a critical task to guarantee energy savings and high performance. Only recently network-aware techniques for VMs allocation have been proposed. However, a network-aware placement requires the knowledge of data transfer patterns between VMs, so that VMs exchanging significant amount of information can be placed on low cost communication paths (e.g. on the same server). The knowledge of this information is not easy to obtain unless a specialized monitoring function is deployed over the data center infrastructure. In this paper, we propose a correlation-based methodology that aims to infer communication patterns starting from the network traffic time series of each VM without relaying on a special purpose monitoring. Our study focuses on the case where a data center hosts a multi-tier application deployed using horizontal replication. This typical case of application deployment makes particularly challenging the identification of VMs communications because the traffic patterns are similar in every VM belonging to the same application tier. In the evaluation of the proposed methodology, we compare different correlation indexes and we consider different time granularities for the monitoring of network traffic. Our study demonstrates the feasibility of the proposed approach, that can identify which VMs are interacting among themselves even in the challenging scenario considered in our experiments.

1. INTRODUCTION

The problem of energy-efficient VMs allocation in cloud data centers is becoming a popular research field. Most research studies merely consider VMs allocation as a way to reduce the number of active servers [2, 5, 3], while more recent proposals also take into account the energy related to the data transfered between VMs, attempting to locate on the same server the VMs that tend to exchange more data among themselves [4, 8]. However, the adoption of a network-aware VMs allocation strategy requires the knowledge about the communication patterns between VMs; unfortunately, it is not possible to obtain a data transfer matrix between the VMs unless a specialized monitoring infrastructure is developed and deployed over the cloud data center, as in [6, 9]. On the other hand, the most popular monitoring services in industry[1] and literature [1] just limit their output to the inbound/outbound aggregate traffic of each VM, without a per-source/per-destination breakdown.

In order to address this issue, we propose a different approach explicitly aiming to identify pairs of VMs communicating among themselves by identifying co-occurring similarities in the communications patterns of different VMs. Our approach is explicitly focused on the identification of communicating VMs in the case where multi-tier applications are horizontally replicated over a cloud infrastructure. In this scenario, each VM typically hosts a different software tier of the application, and a request dispatcher distributes the workload over each vertical stack composed by the multiple application tiers. It is important to note that the presence of a load-balancing request dispatcher leads to very similar utilization and communication patterns in the VMs corresponding to the same tier of the application. This similarity represents one of the most challenging aspects for the problem of inferring the communication patterns among different VMs.

In this paper we introduce a methodology that exploits correlation to identify groups of VMs that communicate among themselves, starting from the time series of the global network utilization of each VM. Furthermore, we compare different indexes to measure the level of correlation among VMs network utilization time series and evaluate the capability of the proposed methodology to identify communicating VMs. Our experiments, based on the deployment of a benchmark over a cloud infrastructure, show that the analysis of the correlation among VMs network utilization time series is a promising way to identify VMs that exchange large amount of data among themselves.

The remainder of this paper is organized as follows. Section 2 models the problem and outlines the proposed methodology. Section 3 describes the experimental results used to validate our proposal. Finally, Section 4 concludes the paper with some final remarks and outlines open research problems.

2. METHODOLOGY

Our reference scenario is a cloud data center hosting multi-tier applications, where the tiers of an application are organized in vertical stacks, which are horizontally replicated as in Figure 1.

Let us consider a set $\mathcal{N}$ of VMs. We define $P_{j_1}^{out}$ as the time series describing the output packet rate of VM $j_1 \in \mathcal{N}$. In a similar way, we can define $P_{j_2}^{in}$ as the time series of input packet rate for a generic VM j_2. We denote as τ the time interval between samples. Starting from these time series, we aim to infer which VMs communicate among themselves by considering the correlation in

[1] https://aws.amazon.com/cloudwatch/

InfQ '16 October 24–25, 2016, Taormina, Italy

DOI: 10.1145/1235

VALUETOOLS 2016, October 25-28, Taormina, Italy

DOI 10.4108/eai.25-10-2016.2268731

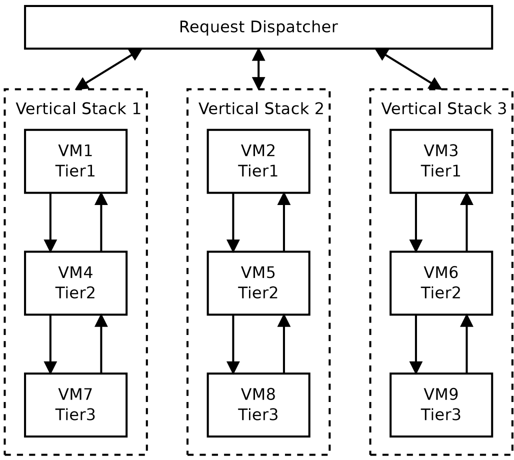


Figure 1: Multi-tier application deployed over a cloud data center

their network activity. The proposed methodology is based on the following steps: 1) Interpolation and synchronization of the traces; 2) Computation of a correlation matrix; 3) Identification of group of interacting VMs.

2.1 Traces interpolation and synchronization

The cloud monitor that collects information about the VMs network utilization is based on the prototype described in [1]. The data collector produces a sequence of tuples in the form `<timestamp, pkt_in, pkt_out>`, where the last two values contain the number of packets received and transmitted since the last data sampling. However, the agents monitoring each VM are not explicitly synchronized. As a result, traces collected from different VMs may be not synchronized.

Preliminary tests with artificially generated traces demonstrate the detrimental effect of not synchronizing traces before applying the correlation analysis of network traffic patterns belonging to different VMs. For this reason, we introduce in our system a trace synchronization step: to achieve this result we rely on data interpolation. Our approach can be summarized as follows:

- we remove samples at the beginning from each time series such that each time series starts in the time interval $[t_0, t_0+\tau]$ and we make sure that each time series contains T samples;
- we compute a synchronization time t_0^* that is the average of the starting times of each trace; from this time we have the interpolated time series timestamps that we define as $t_0^*, t_0^* + \tau, \ldots, t_0^* + i\tau, \ldots, t_0^* + T\tau$;
- for each time series $P_{j_1}^{out}$, we define a synchronized time series $P_{j_1}^{*out}$ using cubic interpolation. In our prototype the implementation of the cubic interpolation is provided by the python *Pandas*[2] framework. A similar procedure is carried out also for the time series of inbound packets $P_{j_2}^{in}$.

As a simplified notation, in the following of the paper we will use $P_{j_1}^{*out}(i)$ and $P_{j_2}^{*in}(i)$ for the i-th sample of a synchronized time series.

2.2 Correlation matrix

Starting from the synchronized time series we can compute the correlation matrix $\mathbf{C}$ between input and output packet rates of each pair of VMs. Specifically, we define the generic element c_{j_i,j_2} of the matrix as:

$$c_{j_i,j_2} = \mathrm{Cor}(P_{j_1}^{*out}, P_{j_2}^{*in}) \quad (1)$$

where j_1 and j_2 are two generic VMs and $\mathrm{Cor}(\cdot)$ is a function that computes the correlation among two different time series. In literature multiple functions have been proposed to measure the correlation between data series. In our analysis we consider two different alternatives, that are the Pearson and the Spearman correlation functions [7]. The first is the most widely used solution to compare time series, while the second emerged in preliminary results as one of the most interesting alternatives in terms of ability to discriminate groups of time series characterized by similar long-term trends, as is the case of our experiments.

We recall that the Pearson correlation index ρ is defined as:

$$\rho(P_{j_1}^{*out}, P_{j_2}^{*in}) = \frac{E[(P_{j_1}^{*out} - \mu(P_{j_1}^{*out}))(P_{j_2}^{in} - \mu(P_{j_2}^{*in}))]}{\sigma(P_{j_1}^{*out})\sigma(P_{j_2}^{*in})} \quad (2)$$

where $\mu(\cdot)$ is the mean value of a time series, $\sigma(\cdot)$ is the standard deviation, and $E[\cdot]$ is a shorthand for the average function (in equation 2 the average is used to compute the covariance of the two time series).

The Spearman correlation index ρ_s corresponds to the correlation between two time series where the original values have been substituted by their ranks in the time series:

$$\rho_s(P_{j_1}^{*out}, P_{j_2}^{*in}) = 1 - \frac{6\sum_{i=0}^{T} r(P_{j_1}^{*out}(i)) - r(P_{j_2}^{*in}(i))}{T(T^2-1)} \quad (3)$$

where T is the number of samples in the time series and $r(\cdot)$ is the rank of a sample among the other samples of the same time series. Switching from the values to the ranks in a time series increases significantly the capacity of this function to discriminate the small fluctuations in the packet rate that may place apart two time series with the same long-term behavior. This is a promising solution to cope with the characteristics of our scenario, where we assume to have a similar workload on every element of the replicated vertical stacks of the considered application.

2.3 Identification of interacting VMs

The final step of our proposal is the definition of groups of VMs; this step exploits the correlation matrix as an input to identify the groups of communicating VMs, that are VMs belonging to the same vertical stack. To this aim, we use a threshold to filter out *correlated* and *uncorrelated* time series of network resource utilization. Basically, we consider as correlated every couple of time series with a correlation value higher or equal than a defined threshold, while if the correlation value is lower than the threshold we assume the corresponding time series to be uncorrelated.

After applying the threshold, VMs belonging to the same vertical stack can be identified looking for the connected components in the correlation matrix.

3. EXPERIMENTAL RESULTS

3.1 Experimental setup

We test our infrastructure using an experimental setup where multiple copies of the TPC-W benchmark[3] are executed in parallel over a cloud data center hosted by the Amazon EC2 infrastructure[4]. The monitoring is based on the architecture described

[2] http://pandas.pydata.org/

[3] www.tpc.org/tpcw/

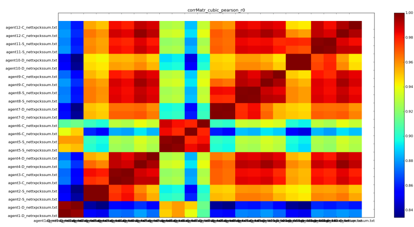
[4] https://aws.amazon.com/ec2/

in [1]. Our traces refer to a TPC-W run with 12 VMs divided into 4 vertical stacks for a period of 12 hours, with samples collected by default every 30 seconds (but we present also experiments with traces where data collection has a granularity of 1 and 2 minutes).

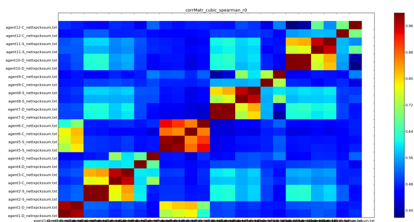
As metrics for the comparison of the different correlation functions, we consider the classic measures of classification problems, that are *precision*, *recall*, and *accuracy*. *Precision* is defined as the fraction of identified couples of VMs that are actually communicating, while *recall* is the fraction of commucating couples of VMs that are correctly identified. In terms of true/false positives/negatives, we can define $Precision = \frac{TP}{TP+FP}$, $Recall = \frac{TP}{TP+FN}$, and $Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$

3.2 Comparison of correlation functions

We first present the correlation matrices computed over every couple of time series (including correlation between couples of input and couples of output packet rate time series) using both Pearson and Spearman correlation indexes.



(a) Pearson correlation index

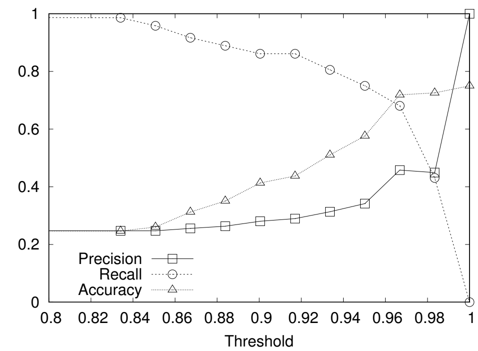


(b) Spearman correlation index

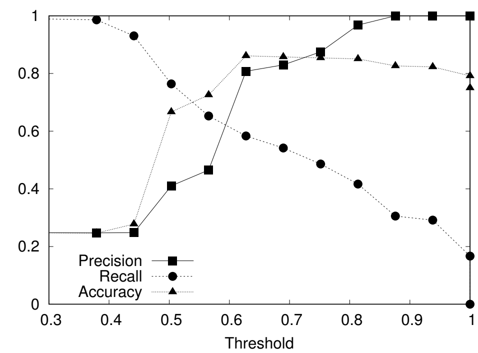
Figure 2: Heatmap of the correlation matrices

Figure 2 represents the two correlation matrices represented as heat maps. The structure of the vertical stacks should appear as four square blocks of correlated VMs on the main diagonal of the matrix. A qualitative analysis of these two figures provides some interesting insights: we observe clearly that the Pearson correlation index (Figure 2a) tends to return higher values (as testified by the general predominance of reddish colors). Particularly critical is the presence of red-orange shades far from the main diagonal, that suggest high correlation also between VMs belonging to different vertical stacks. On the other hand, the Spearman correlation index (Figure 2b) provides clearer distinction between groups of VMs in the same vertical stack (the red areas close to the diagonal) and VMs belonging to different stacks, suggesting a better capacity to distinguish communicating from not interacting VMs.

We now consider the sensitivity of the VMs classification to the threshold value: Figure 3 shows precision, recall and accuracy values as a function of the threshold. From the results, we have a confirmation of the better performance of the Spearman correlation index. Even if both correlation functions show similar traits, such as the precision increasing and the recall decreasing with the



(a) Pearson correlation index



(b) Spearman correlation index

Figure 3: Precision, Recall, and Accuracy

threshold values, Figure 3a shows that the Pearson correlation index tends to have very high values for every couple of time series (we restrict our analysis in the range [0.8–1] to better observe the impact of the threshold choice). Even worse, the Pearson index determines a low precision in the identification of correlated traces, that increases only when the threshold is very close to 1; the accuracy describes effectively this behavior, reaching the highest values just on the rightmost part of the figure. On the other hand, if we consider the Spearman correlation index (Figure 3b), we observe that the sensitivity to the threshold value is less evident, with an high precision achieved even for low threshold values (in the order of 0.6) and with an accuracy that remains rather high for threshold values in the range [0.6–1].

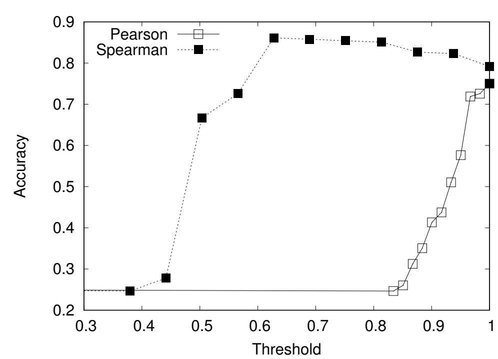


Figure 4: Accuracy comparison

Figure 4 summarizes this comparison showing the accuracy for

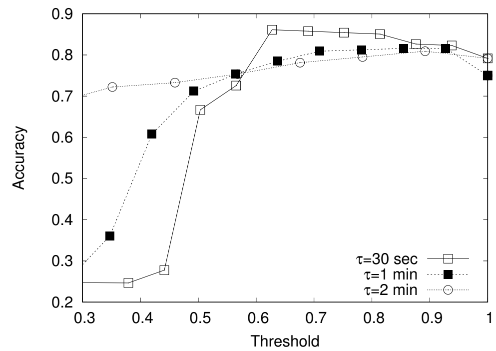


Figure 5: Sensitivity to sampling period τ

both correlation indexes as a function of the threshold value. It is clear that the Spearman correlation index provides a twofold advantage. First, it achieves better performance than the Pearson index, with higher accuracy values for every threshold. Second, it provides more stable performance with a range of threshold values returning an accuracy higher than 0.8 that spans from 0.6 to 1, while the best values for the Person function are limited in a tiny range from 0.96 to 1 (ten times smaller than the alternative).

3.3 Comparison of time sampling intervals

As a final analysis, we evaluate how the data sampling granularity affects the quality of the classification of correlated and non-correlated time series. To this aim, Figure 5 compares the accuracy achieved by the classification based on the Spearman correlation index as the sampling period τ of the network utilization ranges from 30 seconds to 2 minutes. We observe that, even with this small change in the sampling granularity, the impact on the classification accuracy is very significant. Specifically, we observe two main effects of a coarse-grained data collection. First, the sensitivity of the Spearman correlation to the threshold value is further reduced if we consider a large sampling period. This higher robustness is effect of the smoothed time series: basically, we work with a time series where only the most evident fluctuations are taken into account. This means that the low-frequency and highly evident fluctuations, that are the only way to distinguish two vertical stacks, are easier to identify even over a wider range of threshold values, thus explaining the observed stability with respect to different theshold values. Second, if we consider the threshold range related to the higher accuracy ([0.6-0.9]), we observe that the accuracy tends to decrease as the sampling interval increases. This effect is rather intuitive, because by increasing the sampling period, we obtain a smoother time series of network resource utilization. This effect still captures the main effects of correlation, but high frequency fluctuations that allow us to distinguish two vertical stacks are lost, with a consequent reduction of the classification accuracy.

4. CONCLUSIONS AND FUTURE WORK

In this paper we analyzed the problem of identifying groups of VMs in a cloud infrastructure that exchange information without having access to a detailed model of the data transfer among them. In particular, we focus on the case of horizontally replicated vertical stacks, where each VM in a vertical stack corresponds to a tier of an application. We pointed out that this case is very common and challenging in a cloud scenario where scalability is achieved though replication.

We described a methodology to identify communicating VMs that is based on the analysis of correlation between the time series of network traffic of each VMs. We compare different correlation indexes to identify the most suitable alternative. Our experiments show that the use of ranking-based techniques (as in the Spearman correlation index) clearly outperforms traditional correlation metrics, such as the Person index. A further qualifying point of our analysis is to demonstrate that fine-grained data collection is essential to achieve high accuracy in the identification of communicating VMs.

This paper is just a preliminary study addressing the issue of inferring VMs communication patterns in a cloud data center. Further research directions are related to the investigation of a more detailed methodology (including clustering techniques to identify the groups of connected VMs) and the evaluation of our methodology over a wider experimental setup, including different applications, a higher number of VMs involved, and a real workload.

Acknowledgement

The authors acknowledge the support of the University of Modena and Reggio Emilia through the project *SAMMClouds: Secure and Adaptive Management of Multi-Clouds*. The authors also thank Davide Ferrari for his help in conducting the experiments.

5. REFERENCES

[1] M. Andreolini, M. Colajanni, and M. Pietri. A scalable architecture for real-time monitoring of large information systems. In *Proc. IEEE Symposium on Network Cloud Computing and Applications*, London, UK, Dec. 2012.

[2] A. Beloglazov, J. Abawajy, and R. Buyya. Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. *Future generation computer systems*, 28(5):755–768, 2012.

[3] 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)*, Jun. 2015.

[4] A. Marotta and S. Avallone. A Simulated Annealing Based Approach for Power Efficient Virtual Machines Consolidation. In *Proc. of 8th International Conference on Cloud Computing (CLOUD)*. IEEE, 2015.

[5] C. Mastroianni, M. Meo, and G. Papuzzo. Probabilistic Consolidation of Virtual Machines in Self-Organizing Cloud Data Centers. *IEEE Transactions on Cloud Computing*, 1(2):215–228, 2013.

[6] X. Meng, V. Pappas, and L. Zhang. Improving the Scalability of Data Center Networks with Traffic-aware Virtual Machine Placement. In *Proc. of the 29th Conference on Information Communications (INFOCOM)*, San Diego, California, USA, Mar. 2010.

[7] L. Myers and M. J. Sirois. *Spearman Correlation Coefficients, Differences between*. John Wiley & Sons, Ltd, 2014.

[8] M. Shojafar, C. Canali, R. Lancellotti, and E. Baccarelli. Minimizing computing-plus-communication energy consumptions in virtualized networked data centers. In *Proc. of The 21st IEEE Symposium on Computers and Communications (ISCC 2016)*. IEEE, 2016.

[9] J. Sonnek, J. Greensky, R. Reutiman, and A. Chandra. Starling: Minimizing Communication Overhead in Virtualized Computing Platforms Using Decentralized Affinity-Aware Migration. In *Proc. of 39th International Conference on Parallel Processing (ICPP)*, San Diego, CA, Sept 2010.