Ergodic Hidden Markov Models for Workload Characterization Problems

Alfredo Cuzzocrea
DIA Dept., University of Trieste and ICAR-CNR, Italy
alfredo.cuzzocrea@dia.units.it

Enzo Mumolo
DIA Dept., University of Trieste, Italy
mumolo@units.it

Gianni Vercelli
DIBRIS Dept., University of Genova, Italy
gianni.vercelli@unige.it

June 3, 2017

Abstract

We present a novel approach for accurate characterization of workloads. Workloads are generally described with statistical models and are based on the analysis of resource requests measurements of a running program. In this paper we propose to consider the sequence of virtual memory references generated from a program during its execution as a temporal series, and to use spectral analysis principles to process the sequence. However, the sequence is time-varying, so we employed processing approaches based on Ergodic Continuous Hidden Markov Models (ECHMMs) which extend conventional stationary spectral analysis approaches to the analysis of time-varying sequences.

In this work, we describe two applications of the proposed approach: the on-line classification of a running process and the generation of synthetic traces of a given workload. The first step was to show that ECHMMs accurately describe virtual memory sequences; to this goal a different ECHMM was trained for each sequence and the related run-time average process classification accuracy, evaluated using trace driven simulations over a wide range of traces of SPEC2000, was about 82%. Then, a single ECHMM was trained using all the sequences obtained from a given running application; again, the classification accuracy has been evaluated using the same traces and it resulted about 76%. As regards the synthetic trace generation, a single ECHMM characterizing a given application has been used as a stochastic generator to produce benchmarks for spanning a large application space.

1 Introduction

Performance evaluation of computer systems requires to test different alternatives under identical conditions. However, a real computing environment is generally not repeatable, and for this reason it is necessary to characterize the workload by developing a workload model that can be used repeatedly. Once a workload model is available, changes in the workload and in the system can be studied under controlled conditions.

As pointed out in [1], workload characterization using a model plays a fundamental role in many areas, namely to understand the key resource usage of applications, to tune computer architectures, to validate trace reduction mechanisms, to guide the selection of programs for obtaining benchmark sets, to generate synthetic traces to span application spaces, and to create abstract program behavior models for perfor-

DOI reference number: 10.18293/DMSVLSS2017-020

mance studies of computer systems.

Workloads are typically modeled as stochastic processes and analyzed with statistical techniques [3] [4]. This is because different benchmarks are obtained from a single application for different inputs, and the only way to describe all the potential application space is through the extraction from the running application of suitable parameters which describes the main features of the workload.

A running application thus produces a huge amount of data; the only way to analyze such data is by means of statistical techniques. In this paper we propose to use ergodic Hidden Markov Models as statistical models of workloads. Our approach is based on the idea to treat the sequences of memory page references produced by a running application as time-varying discrete-time series of data and to analyze them with statistical techniques using spectral parameters. The proposed methodology operates as follows: the page references sequences obtained from a running application is divided into segments of some hundreds of page numbers, and each piece is then described with a vector of spectral parameters. Chunks of references are formed by some hundreds of such vectors; the chunks are then used to estimate the parameters of a Hidden Markov Model. Repeating this operation for each running application, we compute a HMM model of the application. The accuracy of such models has been estimated as quite good.

By considering a number of workloads obtained from the same type of application, and re-estimating the parameters of a single Hidden Markov Model, a statistical model of that type of application can be computed. In this way, we have obtained models for several application types, as described in 1.1. In this paper, models have been used in two ways: to determine to which application type belongs a running application and to generate synthetic traces. Both these points are very important from a computer architecture perspective. As regards the benchmark classification, it is important to note that using our approach the classification is possible in run-time, i.e. during the application execution, since the computational complexity is quite low. As regards the synthetic traces generation, HMMs can indeed be viewed as generators of observations, in our case allowing to cover a large application space for computer architecture studies and designs [5].

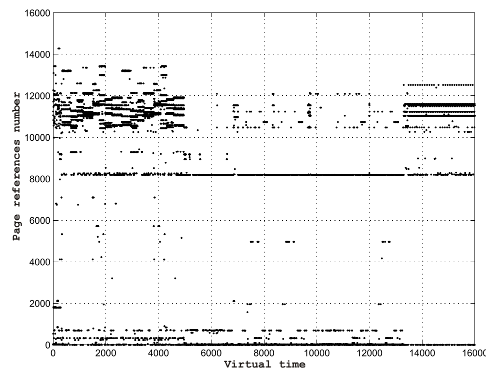

Figure 1: Graphical view of a portion of a sequence of page references.

1.1 Methodology

We used traces-driven simulations to test the proposed approach. The traces were a subset of the SPEC2000 benchmark suite [2], as reported in Tab. 1.

Benchmark	Category	Total number of page references
Bzip2	compression	519960950
Crafty	chess game	322625985
Eon	ray traces	526065045
Gcc	C compiler	646344471
Gzip	compression	477528457
Perl	Perl interpreter	351047065
Twolf	place and route simulator	5246007019
Vpr	FPGA placement and routing	2240811177

Table 1: The traces used in this work.

CPU address traces have been obtained by running the applications of Tab. 1 with different input data; several executions of each application have been considered. The applications of Tab. 1 run on a Pentium 2 processor at 450 MHz under Windows NT operating system. The benchmarks were downloaded from *www.byu.com*. In Fig. 1 a part of a page references trace (16000 virtual time instants) is shown. This figure illustrates the time-varying characteristic of the trace.

The rest of this paper is organized as follows. In Section 2 the properties of HMMs are described together with the considered workload parameters. In Section 3 the workload classification methodologies based on HMM are described while in Section 4 the generation of synthetic traces with HMMs is briefly reported. Finally, in Section 5 some final remarks are reported.

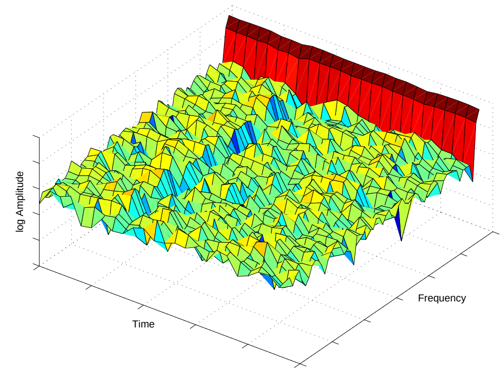

Figure 2: Log-spectral data of the portion of the page references sequence shown in Fig. 1.

2 Hidden Markov Models for Workload Classification

2.1 Parameters

The page references are produced at a CPU instruction clock rate, because each virtual memory address is translated to a virtual page reference. This information rate is too high to make reasonable workload evaluations, and consequently the number of page references is too large. Therefore, some feature extraction must be performed for getting rid of the redundant information and for reducing the data rate. According to the idea of considering the page references sequence as a signal, we use a spectral description of the page references sequences. Characteristics in the sequences, such as for examples loops or sequential program behaviors, are indeed described in the spectrum. For instance, loops introduce peaks in the spectrum while a sequential address sequence produces a DC component. For example, representing the sequence of Fig. 1 in the log spectral domain, we obtain the data shown in Fig. 2.

Since the page references sequence is time varying, as suggested in Fig. 1, the result of Fig. 2 is obtained with short-time spectral analysis. In particular, the sequence of virtual memory pages is divided into short sections – 120 references long – and analyzed by means of a discrete Fourier transform.

It is worth noting that Fig. 2 reports a log-spectral view of the page references trace shown in Fig. 1. In Fig. 2 it is possible to see how the change of behavior in the trace of Fig. 1 at about 5000 virtual time instants reflects in the spectral domain. As in the proposed approach a fundamental issue is related to the comparison between log-spectral data, it is important to define a log-spectral distance between two spectra. To show how to define the log-spectral distance, let us start with the Euclidean distance definition between the log spectra of two sequences, x_n and y_n:

$$\begin{aligned} e(\omega) &= log\,|X(\omega)|^2 - log\,|Y(\omega)|^2 = \\ &= 2(log\,|X(\omega)| - log\,|Y(\omega)|) = \\ &= 2Re\left[log\,(X(\omega)) - log\,(Y(\omega))\right] \end{aligned}$$

where $X(\omega) = \sum_{n=-\infty}^{+\infty} x_n e^{j\omega n}$ is the spectrum of the x_n sequence. On the other hand, $\log(X(\omega)) = \sum_{n=-\infty}^{+\infty} c_n e^{j\omega n}$ where c_n is the cepstrum sequence [7] which is obtained applying an inverse Fourier transform to the log spectrum of the input page references sequence. Hence, calling c_n^x and c_n^y the cepstrum of the x_n and y_n sequences respectively,

$$e(\omega) = 2Re\left[\sum_{n=-\infty}^{+\infty} (c_n^x - c_n^y)\, e^{j\omega n}\right] = \sum_{n=-\infty}^{+\infty} (c_n^x - c_n^y)\, e^{j\omega n}$$

because the c_n sequences are symmetrical since the input reference page sequence is real. Finally, the spectral distance between two sequences x_n and y_n is

$$d(X,Y) = \frac{1}{2\pi}\int_{-\pi}^{\pi} e^2(\omega)d\omega = \sum_{n=-\infty}^{+\infty} (c_n^x - c_n^y)^2 .$$

In conclusion, the spectral distance between the log spectra is simply the Euclidean distance between the cepstal sequences.

On the basis of this consideration, we described the page references sequences with cepstral coefficients. In Fig. 3 the cepstral representation of the page references sequence of Fig. 1 is reported. As shown in Fig. 2 the change of trace behavior at about 5000 time instants is reflected in the cepstral domain. In fact, the slow spectral characteristics are seen in the part around zero in the cepstral domain, in Fig. 3 we can see that the initial part of the cepstrum is more spiky around zero reflecting in this way the change of trace behavior seen in Fig. 1. On the basis of that, it is useless to consider all the cepstral coefficient to represent traces; for this reason we used only the first 10 cepstral coefficients.

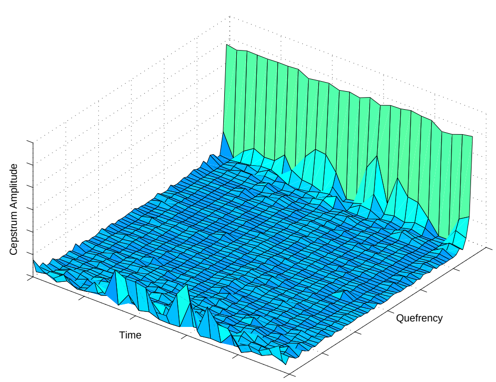

Figure 3: Cepstral description of the portion of the page references sequence shown in Fig. 1.

2.2 Hidden Markov Modeling

Markov models are stochastic interpretations of time series. The basic Markov model is the Markov chain, which is represented with a graph composed by a set of N states; the graph describes the fact that the probability of the next event depends on the previous event. The current state is temporally linked to k states in the past via a set of N^k transition probabilities. Let us denote the generic state of the system with S_t, $S_t \in \{1, 2, ..., N\}$ and by $a(S_t|S_{t-1}, S_{t-2}, \ldots, S_{t-k})$ the probability that the system is currently in state S_t given the previously sequence of states $S_{t-1}, S_{t-2}, \ldots, S_{t-k}$; $a()$ is called the transition probability for a model of order k. In homogeneous Markov chains, the transition probability depend on the previous state only; in such case the transition probabilities can be represented by a transition matrix. If the Markov chain is fully connected, or **ergodic**, each state of the model can be reached from every other state in a single virtual time step. As regards the macroscopic capabilities of such models, we can say that the self loops describe a locality in the process.

Other types of HMMs could better describe the statistical properties of the observed process. For example, the left-to-right models have the property that, as virtual time increases, the state index also increases; they can therefore model sequences whose properties change over time in a successive manner.

In general, an homogeneous Markov chain has the following properties:

1. limited horizon: $Prob(S_{t+1}|S_t, S_{t-1}, \ldots, S_1) = Prob(S_{t+1}|S_t)$;
2. stationarity: $Prob(S_{t+1}|S_t) = Prob(S_2|S_1)$.

A Markov chain is therefore described by the transition matrix A whose elements are $a_{i,j} = Prob(S_{t+1} = j|S_t = i)$ and the initial probability vector π_i, $\pi_i = Prob(S_1 = i)$, $\sum_{i=1}^{N} \pi_i = 1$. However, in many cases, Markov models are too simple to describe complex real systems and signals [8]. In Hidden Markov Models (HMMs), the output for each state corresponds to an output probability distribution instead of a deterministic event. That is, if the observations are sequences of discrete symbols chosen from a finite alphabet, then for each state there is a corresponding discrete probability distribution which describes the stochastic process to be modeled. In HMMs, the state sequence is hidden and can only be observed through another set of observable stochastic processes. Thus, the state sequence can only be recovered with a suitable algorithm, on the basis of optimization criteria. It is important to note that the observation probabilities has been so far assumed discrete. In many cases, however, the observations are continuous features vectors. It is possible to convert the continuous observations into discrete ones using vector quantization, but in general some performance degradation due to the quantization process is observed. Hence, it is important, from a performance point of view, to use an overall continuous formulation of the algorithms.

Generally speaking, HMMs lead to the three basic problems:

1. the estimation problem: given the observed sequence $\mathbf{O}=O_1, O_2, \ldots, O_T$, how the model parameters λ can be adjusted to maximize $Prob(\mathbf{O}|\lambda)$? This problem concerns the estimation of the model parameters. This estimation process is performed by iteratively maximize the likelihood $Prob(\mathbf{O}|\lambda)$ using an Expectation Maximization (EM) approach [9]. The differences between discrete and continuous HMMs lead to different re-estimation algorithms for the model parameters.

2. the evaluation problem: given the observed sequence $\mathbf{O}$, the problem is to compute the probability that the observed sequence whose produced by the model. This problem can be also stated as follows: given several HMMs and a sequence of observations, how do we choose the model which best matches the observations?

3. the decoding problem: given the observation sequence $\mathbf{O}$, what is the most likely state sequence $S = S_1, S_2, \ldots, S_T$? The decoding is usually performed using the Viterbi algorithm.

3 Workload Classification

For dynamic characterization of processes, the address field of the BYU traces has been extracted. In this way we have obtained a sequence of virtual addresses generated by the processor during the execution of the processes. For converting the trace of addresses into trace of virtual pages, the sequence of addresses has been divided by the page dimension, which we set to 4096 bytes.

Once the sequence of virtual pages has been obtained from every BYU trace and thus for every process, we have tried to use discrete HMMs for their classification. Even if the sequence of pages is a discrete sequence, it can not be used for processes classification using discrete HMMs, as it contains a too high number of symbols.

In order to face this problem, the sequence of virtual pages has been turned into a sequence of few symbols, without loosing meaningful data. The sequence of virtual pages has been turned into sequence of cepstral coefficients by the short time analysis process described in Sec. 2.1.

3.1 Single Trace Classification

The sequences of cepstral coefficients are real number sequences. For analyzing cepstral sequences using a discrete HMM, vector quantization is needed. In this process some degradation is introduced and the training lacks its efficiency.

A continuous HMM can use an input sequences of 10-dimensional cepstral vectors and vector quantization is not needed. The results obtained in this way usually perform better than using the discrete model.

The multivariate Gaussian density is used for describing the cepstral observation. The 10-dimensional cepstral vector is described using a multivariate density having 10 dimensions, and it is specified by means of the mean and covariance matrixes. Using this approach it is supposed that the 10-cepstral coefficients are uncorrelated and so the covariance matrix is diagonal.

In order to choose the number of states and the topology of the HMMs, several tests have been performed. The number of states needed is lower than in the discrete HMM. Considering topology, ergodic models score better results.

In Fig. 4 a graphical representation of the mean classification of all the traces over the number of states for ergodic and left-right models is depicted.

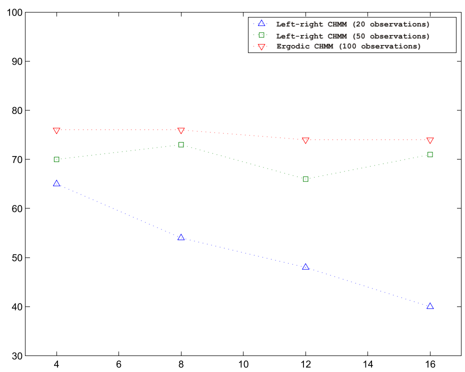

Figure 4: Average recognition rate for all the traces over the number of states of ergodic and left-right models.

As the models using 4 states provides better results using a lower number of observation, we have repeated experiments using this configuration increasing the number of observations.

Using 100 observations for every model, in the ergodic case the recognition mean of single traces is about 82%, in the left-right case this mean is 65%. In Fig. 5 and in Fig. 6 these results are depicted, gathering the traces per workload and computing for every traces group the mean recognition rate.

The ergodic continuous HMMs have been trained using 100 observations for every model. The recognition rate varying the number of states and using all the traces is reported in Fig. 6.

The results obtained using such statistical models demonstrated the effectiveness of this dynamic processes modeling approach. Cepstral coefficient obtained from the virtual pages sequences are a good parameter for describing traces of programs during execution.

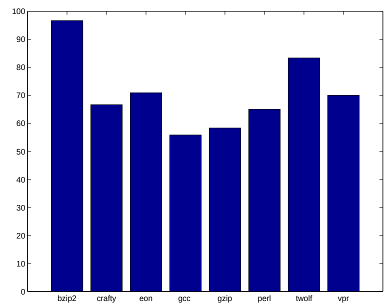

Figure 5: Average classification rate for all the traces using 16-state ergodic discrete HMMs.

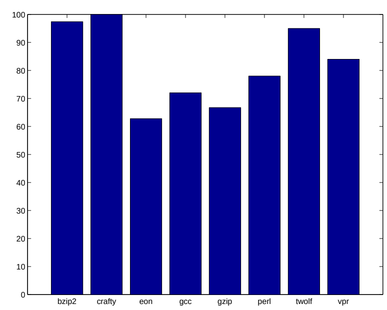

Figure 6: Average classification rate for all the traces with 4-state ergodic continuous HMMs.

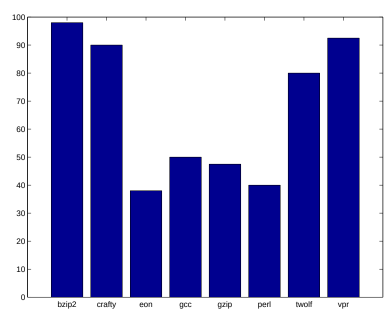

Figure 7: Average classification rate for all the traces with 4-state left-right continuous HMMs.

3.2 Program Behavior Modeling

Dynamic classification of BYU traces, taking as parameter the virtual pages, has obtained satisfactory results. As seen in 3.1, the traces of a single application have been obtained processing such application with different inputs, or processing different functions of the same program.

Then, we have classified the workloads, gathering the traces of the same workload using a single HMM trained with several traces representing the same workload.

Using several traces of the same workload for classifying program behavior using ergodic discrete and continuous HMM, we have obtained the results reported in Fig. 8 and in Fig. 9.

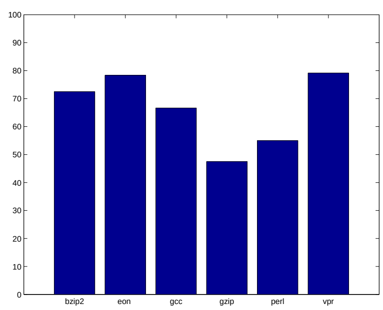

Figure 8: Workload classification using ergodic discrete HMM.

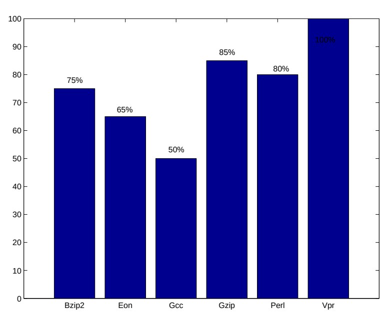

Figure 9: Workload classification using ergodic continuous HMM.

The mean results obtained in the case of ergodic discrete and ergodic continuous HMMs are reported in Tab. 2: ergodic continuous models obtain better classification accuracy than the discrete ones.

	Ergodic Discrete HMM	Ergodic Continuous HMM
Cepstral	65%	76%

Table 2: Workloads classification.

4 Synthetic Trace Generation

A Hidden Markov Model can be used as a generator of a stochastic process. The procedure is the following:

1. Choose an initial state i according to the initial distribution π.
2. Set $t = 1$.
3. Generate a N-dimensional random variable according to the characteristic of the multivariate Gaussian distribution in state i.
4. Perform a state transition according to the transition probabilities $a_{i,j}$.
5. Set $t = t + 1$. If $t < T$ go to 3, else terminate.

The random variable generated in step 3 is a vector of cepstral coefficients. This vector must be inverted to obtain a set of page references.

A result is reported in Fig. 10, where the log-spectral data of a synthetic trace produced with the above procedure and the HMM trained with the trace of Fig. 1 is reported. Fig. 10 should be compared with Fig. 2.

5 Conclusions and Future Work

In this paper we describe an approach for workload characterization using ergodic hidden Markov mod-

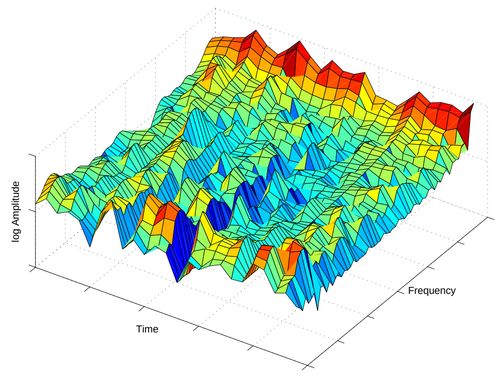

Figure 10: Example of synthetic trace generated using a continuous ergodic HMM represented in the spectral domain.

els. The page references sequences produced by a running application are divided into short virtual time segments and used to train a HMM which models the sequence and is then used for run-time classification of the application type and for synthetic traces generation. The main contribution of our approach are on one hand that a run-time classification of the running application type can be performed and on the other hand that the applications behavior are modeled in such a way that synthetic benchmarks can be generated. Using trace-driven simulation with SPEC2000 benchmarks, the mean classification rate is about 82% for each traces and about 76% using a single HMM to model a single application type. Many future developments of our approach are possible since what we propose in this paper – to use time-varying non-linear processing techniques to treat sequences produced by programs during execution – is a novel approach in computer architecture studies. In addition to this, we believe that another interesting line of research is represented by the adaption of the proposed framework to novel *big data trends* (e.g., [10, 11, 12]).

References

[1] L.K. John, P. Vasudevan, J. Sabarinathan, *Workload characterization: motivation, goals and methodology*. Workload Characterization: Methodology and Case Studies, 1998 29 Nov. 1998 Page(s):3 - 14

[2] C. Niki, J. Thornock, K. Flanagan, *Using the BACH Trace Collection Mechanism to Characterize the SPEC2000 Integer Benchmarks*. Proceedings of the Third IEEE Annual Workshop on Workload Characterization, 2000.

[3] M. Calzarossa, G. Serazzi, *Workload Characterization: A Survey*. Proceedings of the IEEE, vol. 81(8), 1993.

[4] K.J. McDonell, *Benchmark Frameworks and Tools for Modelling the Workload Profile*. Performance evaluation 22, 1995.

[5] J.P. Singh, H.S. Stone, D.F. Thiebaut, *A Model of Workloads and Its Use in Miss-Rate Prediction for Fully Associative Caches*. IEEE Transactions on Computer, vol.41(7), 1992.

[6] L.R. Rabiner, *A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition*. Proceedings of the IEEE, vol.77(2), 1989.

[7] L.R. Rabiner, B.H. Juang, *Foundamentals of Speech Recognition*. Prentice Hall Signal Processing Series, 1993.

[8] Y. Bengio, *Markovian Models for Sequential Data*. Neural Computing Surveys 2, 1999.

[9] J.A. Bilmes, *A Gentle Tutorial of the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models*. Technical Report, TR-97-021, 1997.

[10] A. Cuzzocrea, *Privacy and Security of Big Data: Current Challenges and Future Research Perspectives*. ACM PSBD 2014 Proceedings, 2014.

[11] A. Cuzzocrea, U. Matrangolo, *Analytical Synopses for Approximate Query Answering in OLAP Environments*. DEXA 2004 Proceedings, 2004.

[12] A. Cuzzocrea, G. Fortino, O.F. Rana, *Managing Data and Processes in Cloud-Enabled Large-Scale Sensor Networks: State-of-the-Art and Future Research Directions*. CCGRID 2013 Proceedings, 2013.