Workload Characterization Methodologies in Parallel Environments [1]

L. Massari and P. Rossaro

Dipartimento di Informatica e Sistemistica, Università di Pavia, I - 27100 Pavia, Italy

Abstract

In this paper, new approaches for workload characterization of parallel systems are presented. Parameters able to give insigths into the workload behavior are described and their possible applications to performance studies are also discussed.

1. INTRODUCTION

Workload modeling, a fundamental phase of performance evaluation, is influenced by the new features of parallel systems (see e.g., [1], [2]).

The methodologies usually adopted in sequential environments do not always provide a correct description of the load of parallel architectures. Unlike uniprocessor systems, the characterization of a parallel algorithm is more difficult due to the multiplicity of factors that affect its performance. Hence, the choice of suitable parameters and techniques able to describe the parallel aspects typical of these environments (see e.g., [3]), represents one of the most critical phases of workload analysis.

The aim of this paper is to propose some methodologies for workload characterization of parallel environments. In Section 2, an overview of the new parameters and techniques for describing the behavior of the workload of parallel systems is given. Section 3 presents a few applications of the proposed methodology. Comments and conclusions are summarized in Section 4.

2. WORKLOAD PARAMETERS AND TECHNIQUES

In traditional uniprocessor architectures, the workload consists of a set of programs processed by the system in a given time interval. In the case of multiprocessor systems, the definition of workload is more complex; it may consist of a set of different programs or algorithms or it may be represented by the same algorithm executed under different configurations (e.g., number of processors, interconnection topologies, mapping strategies).

Workload analysis consists of several phases. The input is represented by the data obtained from measurements taken on real systems; a few intermediate results, which give insigths into the workload behavior, and final outputs for system modeling activities are produced by applying various analysis techniques.

The preliminary phase deals with the choice of the parameters to be used in the following steps. This is one of the most critical aspects in workload characterization. These parameters make the difference between parallel and sequential environments, in that they have to reflect the hardware as well as the software characteristics of the system itself.

[1] Partially supported by "Progetto Finalizzato CNR Sistemi Informatici e Calcolo Parallelo" under the Grant N. 91.00887.PF69 and by the Italian M.U.R.S.T. under the 40% Project

Two groups of parameters can be identified: static and dynamic indices. "Static" indices provide an architectural independent description of the resource requirements of the algorithm and can be derived, for example, from its precedence graph [1].
"Dynamic" indices take into account both time and data dependencies and can be derived by monitoring the execution of the algorithm. From a single execution, we can obtain simple parameters, which can give insigths into the behavior of the system and of the algorithm itself. Total computation, communication and execution times, number of I/O operations and of messages exchanged between processors, mean and standard deviation of the number of busy processors, are examples.

Additional parameters, expressed in terms of curves, describe the dynamic behavior of the algorithm as a function of the execution time and of the number of available processors. From the execution signature, which expresses the execution time as a function of the number of processors, speedup, efficacy and efficiency can be derived. They represent the improvement achieved in the performance of the algorithm, and the cost in terms of system resources, obtained by increasing the number of processors. Efficacy allows us to derive the processor working set (pws) [4], that is, the number of processors which maximizes the efficacy, and can be used in allocating processors in a multiprogrammed parallel system.
Parallelism profiles express the number of active processors as a function of execution time, when the number of available processors is unlimited. Thus, they represent the inherent parallelism in the algorithm under ideal conditions. The average parallelism [5], derived from the parallelism profile, then denotes the average number of busy processors for each algorithm, and can also be used for allocation strategies.

Once the parameters have been selected, they can be further processed in the following workload characterization phases. Since most of these indices are provided by monitoring tools, they are expressed as discrete values, even if the variations of the workload can be better represented by analytical functions. Starting from the values initially provided in tabular form, fitting techniques can be applied for deriving the best function that fits the measured data. This method provides also inputs for analytical and simulation models and for the classification techniques, which represent the core for the construction of workload models.

Functional analysis provides a qualitative classification, by grouping the data concerning the programs submitted to the system, according to their type. The types are related to the events executed by the system, and vary according to the level of the analysis. This method can be particularly helpful for load balancing and scheduling policies. Functional analysis can also be considered as a preliminary phase for clustering. Cluster analysis provides a subdivision of the workload data in classes, according to some resource-oriented criteria. Generally, the similarities among the components are based on a predefined metric. From each class, representative components can be extracted and used for further analyses of the system as well as for defining the input parameters of queueing network models, where classes correspond to the workload types in the system.

3. APPLICATION EXAMPLES

An application of the techniques presented is described in what follows. The data considered, measured on a Meiko system with 16 transputers, consists of various types of programs, such as, matrix multiplication and sorting algorithms. The measurements provided by a monitoring tool are structured as a set of records, each of them representing

a particular activity. Information, such as, type of activity (e.g., communication, computation, I/O), duration, message length, sender and receiver identifiers, are provided for each record.
Functional analysis has identified five different types of activity, referring to program starting and completion, data transmission and reception and I/Os on disks, respectively. By processing the times related to each activity, "parallel" parameters, such as, mean communication and computation times (referred to as t_comm and t_comp, respectively) for each algorithm have been derived. A third parameter, referred to as n_proc, represents the mean number of processors used, and has been directly computed from the measurements. Before applying cluster analysis, the distribution of each parameter and statistical information, such as, mean and standard deviation have been obtained. As an example, the distributions of t_comp and t_comm are depicted in Fig. 1. The cluster analysis led to an optimal partition of three classes, whose centroids are described in Table 1. As can be seen, very distinct values characterize these clusters, identifying a light, medium and heavy workload, respectively.

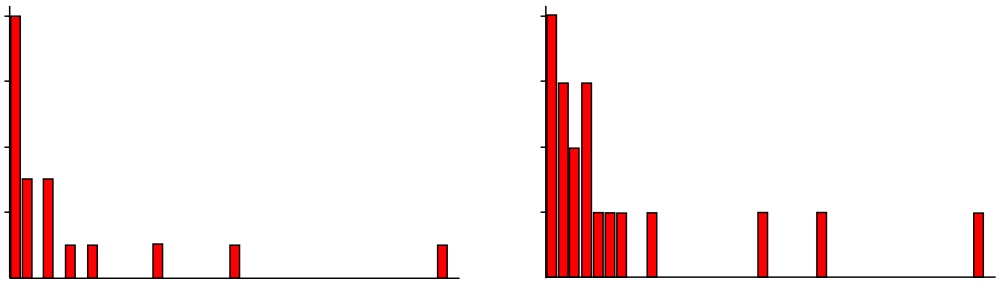

Figure 1. Distributions of the parameters computation time t_comp and communication/synchronization time t_comm.

Table 1
Centroids of the three classes obtained through clustering

Classes	n_proc	t_comp	t_comm
1	9.74	12.	13.51
2	11.9	252.3	83.4
3	14.3	2582.20	533.3

Fitting techniques have also been applied to the same measurements. For each program, parallelism profiles are computed from the initial data, and the results, expressed in tabular form, are fitted for obtaining continuous curves. The technique applied, based on the least squares method, uses linear functions. Varying the degree and the type of the function (e.g., polynomial, exponential and trigonometric) different solutions have been provided. As an example, for a matrix multiplication algorithm of size 96x96, a polynomial of degree 6, expressed as:

$$f(t) = 0.16 + 11.32t + 137.15t^2 - 114.69t^3 - 486.51t^4 + 858.78t^5 - 405.95t^6$$

is the best fitting function. The curve, compared to the original discrete values, is plotted in Fig. 2.
Cluster analysis is then applied for deriving classes of parallelism profiles, having homogeneous characteristics. The results can then be used to drive a simulator which has to exactly reproduce the dynamic behavior of the programs.

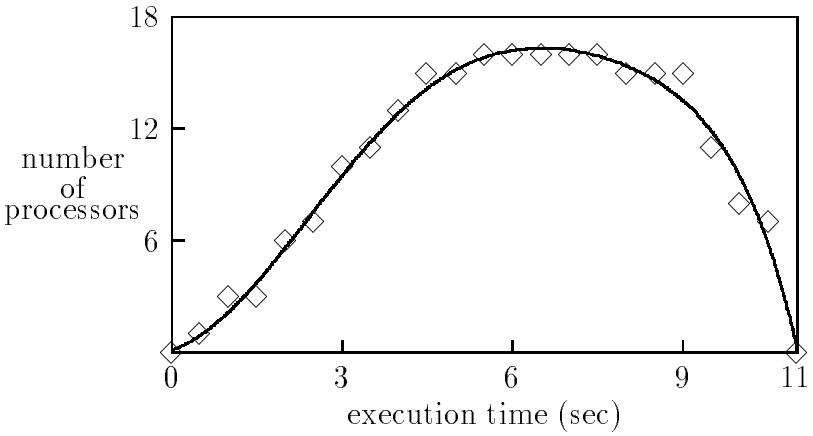

Figure 2. Fitting of a parallelism profile: the squares represent the discrete values, the continuous line the polynomial fit.

4. CONCLUSIONS

Workload characterization methodologies for parallel systems have been investigated. A description of new parameters required for parallel systems in order to give a better understanding of the dynamic behavior of the workload is given.

Future work will be devoted to the integration of these methodologies in a tool which allows the construction of various types of workload models for parallel environments.

References

[1] E. Gelenbe. *Multiprocessor Performance*. John Wiley & Sons, New York, 1989.

[2] V.W. Mak and S.F. Lundstrom. Predicting Performance of Parallel Computations. *IEEE Transactions on Parallel and Distributed Systems*, 1(3):257–270, July 1990.

[3] K.C. Sevcik. Characterization of parallelism in applications and their use in scheduling. In *Proc. 1989 ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems*, pages 171–180, 1989.

[4] D. Ghosal, G. Serazzi, and S.K. Tripathi. The processor working set and its use in scheduling multiprocessor systems. *IEEE Transactions on Software Engineering*, 17(5):443–453, May 1991.

[5] D.L. Eager, J. Zahorjan, and E.D. Lazowska. Speedup versus efficiency in parallel systems. *IEEE Transactions on Computers*, 38(3):408–423, March 1989.