Characterization of Workload Suites for Performance Evaluation

Carlos Manuel Gomes Jardim

Dep. Informática, Universidade do Minho
4710 – 057 Braga, Portugal
Carlos.jardim@aeiou.pt

Abstract. The main goal of this communication is to present "what to measure" and "how to measure" to evaluate the system performance: its characteristics and the impact of the execution environment. Measurements are of fundamental importance for performance evaluation since they provide a "picture" of the behaviour of the system and of their load. Experimental approaches to workload characterization are based on the application of techniques for the interpretation and the analysis of measurements. A survey of methodologies is presented to model workloads in distributed and parallel systems.

1 Introduction

The characteristics of hardware and software components influence not only the performance of a system but also the load it has to process and we can not determine it without knowing the requests that are being processed – the workload. The analysis of the workload plays a key role in all the studies where the performance marks of a system are to be determined.

Measurements represent the basis for workload characterization [1]. As the workload is one of the major components which determine system performance, the measurements' accuracy has a particular importance. With a characterisation of the system under study and a characterisation of the load, a performance model is built and performance results are obtained by applying performance evaluation techniques (including analytical, numerical, or simulation techniques).

Today, it is more and more difficult to evaluate performance, due to the progress and complexity of modern software systems, together with the wide range of performance improvement techniques employed by hardware designers. For different types of applications, different performance metrics may be appropriate and different aspects of a computer system may be the most significant in evaluating the overall performance.

The characteristics of the workloads may vary both from a functional viewpoint and for their impact on the physical resources of the systems, depending on the system architecture and on the application environments. Anyway, a few commonalities for the construction of workload models of different types of systems are identified. This means that the same analysis techniques can be applied in different frameworks and to different parameters. A correct interpretation of the obtained results has then to be performed.

The communication addresses the measurement based approach to workload characterization where two types of systems are analysed, distributed and parallel systems; a survey of the adopted methodologies to describe their loads is also presented.

2 Approach

Before initiating the analysis of methodologies and metrics used to modulate the workload in distributed and parallel systems, the most important characteristics are presented, related to the systems under study – the distributed and the centralized model.

A distributed system is a collection of heterogeneous computers and processors connected via network, working together to achieve a common goal. The computation is distributed among several physical processors, each processor having its own local memory, and communicating with another through various communication lines. Its main advances are resource sharing (files, printers, etc), computation speeding up, reliability (redundancy) and communication (e-mail).

A parallel system consists of multiple processors interacting together, often located within the same computer. There are multi-computers where each processor has it own local memory and others where processors share memory and clock. Their main advances are increased throughput and reliability, speed up and economical.

The analysis of workload is experimental and consists of several phases, being the input represented by the data obtained from measurements taken on real systems, a few intermediate results, which give insights into the workload behaviour, and final outputs for systems modelling activities are produced by applying various analysis techniques.

However, this experimental analysis became a great challenge due to the increasing complexity of the systems and their workload, making it necessary to develop the appropriate instrumentation in the order to promote quality in the measures. It is important to refer that the influence of this instrumentation's system must be the lowest possible not to disturb the system's behaviour and its workload, thus increasing its credibility and precision. So we can say that the measures are fundamental to evaluate the performance, but these must give an "image" of the behaviour of the system and its workload. When we refer to performance's analysis it must be taken into account the metrics' analysis that characterise it.

Generally, metrics can be divided into three separate classes: *application metrics* that define the resource requirements of the application, independent of any architecture; *machine metrics* that represent specific capabilities and resources offered by a candidate computer system; and *performance metrics* that integrate application with machine metrics to give an overall view of the performance of a specific application on a particular computer system. Examples of performance evaluation metrics are: *execution time* – the elapsed time between your request and the system's response and that depends on main factors like the algorithm, the data structure, input data, the platform and the programming language; *processing speed* – measured, for example, by the executed number of floating point operations per second; *system throughput* – the number of jobs processed per time unit; and *utilization* – the ratio of the achieved speed to the peak speed of a given computer.

Next it will be presented the methodologies and metric's analysis used to model a workload in distributed and parallels systems.

3 Workloads in Distributed Systems

Due to the large collection of hardware and software components and of the synchronization and communication constraints, distributed systems are ill-suited to monitoring activities. As said before, measurements are fundamental for workload

characterization, thus the measurements collected on each local node, which in the case of distributed systems becomes quite complex because of the lack of adequate measurement facilities, must be assembled and matched together so that global information on the system performance may be obtained. For instance, the request types and request patterns from the clients to a file server or the packets flow on a local area network are accurately described by measurements, in this way, appropriate performance management facilities and integrated measurement techniques, such as systematic monitoring approach for distributed, client-server environments and computer networks has to be defined and applied for performance evaluation.

Now it is going to be presented a DMS's (Distributed Measurement System) analysis, proposed in [2], that is software architecture for performance monitoring of distributed systems. How does it work? DMS measures the distributed user workload and extended services with macros such as resource consumptions across network nodes. This software architecture has to collect data from such nodes with minimum overhead, and define a set of performance metrics and instrumentation. The figure 1, presented below, represents the architecture of a Distributed Measurement System.

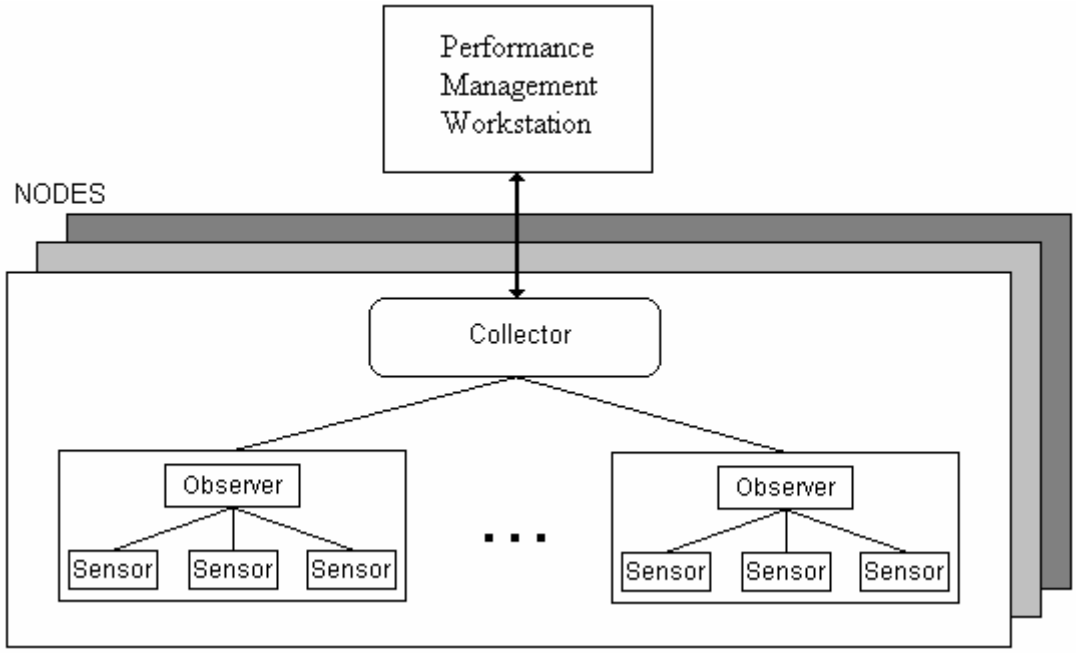


Figure1 – Architecture of Distributed Measurement System [2]

As it can be observed in the diagram, this architecture is hierarchically divided in to different levels to be analysed in the following study. The bottom level consists of sensors that implement the actual instrumentation function by computing and storing specialized data (residence times at processor) and primitive statistical quantities (counts, sums). The first level of communication is between sensors and observer, which represents the next level. The observers collect data for all individual sensors in the local servers. There are several servers (or clients) residing on each node that have associated custom sensors which record specific behaviour, like cache operations, queue lengths, etc. There is a single Collector in each network node that works as storage units and performs management operations. At the end, the accumulated measurements from all node observers are transmitted to the centralized component of DMS - Performance Management Workstation (PMW). Statistical techniques for the analysis of measured data such as an interface for controlling the system parameters from a performance perspective are provided with PWM.

After define measurement methodologies and monitoring tools are available, the data collection can start. It will still be necessary to apply appropriate techniques for the identification of workload parameters and its characterization.

As said before, the workload characterization of distributed system can be hierarchically decomposed. Structure using techniques and parameters typical of each level have to be taking into account by workload models. In [3] and [4] it is proposed a hierarchical approach to workload characterization, which we will analyse.

In the first one [3], the layered structure of the workload is determined by taking into account the logical sub-division in three groups of the hardware components of distributed systems: user terminals, processing nodes and communication subsystem.

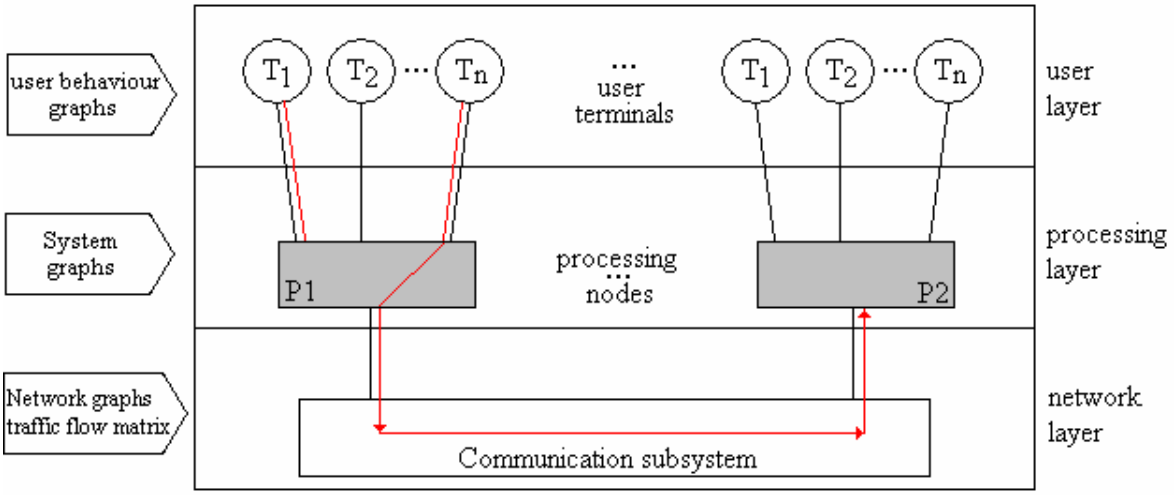


Fig2 – Layered structure for workload characterization of distributed systems [3]

The requests (representing the load) submitted by the user can be executed by the local processing node or may require services from remote nodes which generate messages towards the network. Where we analyse figure 2 we can see three layers and two paths followed by local and remote requests (red line). The dynamic behaviour of the several types of requests is reproduced using probabilistic graphs and the messages flowing on the communication lines are characterized by their traffic flow matrix. Physical resources are involved and parameters such as message length and message type are identified. The consumption of hardware and software resources is used in order to characterize the load of processing nodes.

In the second one [4], we will be analyse a methodology and metrics that describe the load generated by all the clients of a network - Networkload. It is hierarchically decomposed in three levels: Session level, Command level and Request level, as we can see in layered structure presented in figure3. This approach is different from the previous one because it is closer to a client/server perspective.

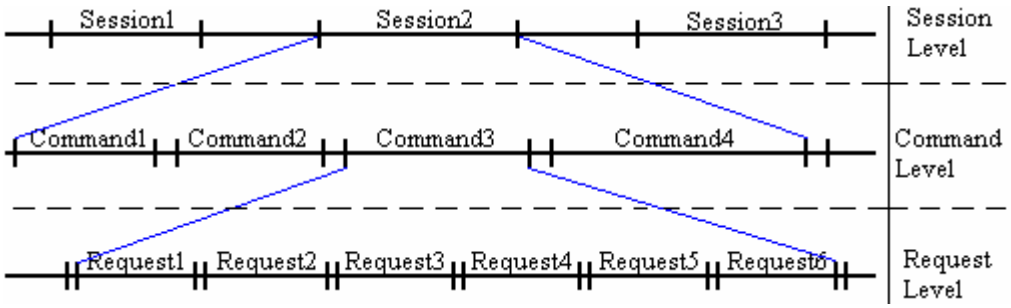


Figure3 – Layered structure for networkload models [4]

As we can see in the figure above, the highest level is session, where the workload consists of a sequence of sessions. The next level – command level, represents the commands sent out during a session and the bottom level consists of the requests that represent the client-server interactions. So that this layered structured can be better understood, we will base the analysis of the methodologies and metrics on one example. Imagine a system with fileservers and diskless client: clients generate multiple parallel activities to the servers which provide several types of services, like file services. At the session level, the workload is characterized by parameters like session duration – time between logon and logoff, representing the busy period of client and session inter-arrival time – time between the end of one session and beginning of the next one, representing the idle time of client. At the command level the sequence of events which compose a session is analysed. The workload is characterized by parameters like accessed files size and number of the file accesses. The bottom level – request level; represents the load at the server due to all clients of the network. The parameters like request generation rate of the clients, number of requests generated by the client per unit of time, and service time of the server, time taken by the server to satisfy the client request, are derived by using monitoring tools, which capture the flow packets in a given interval time.

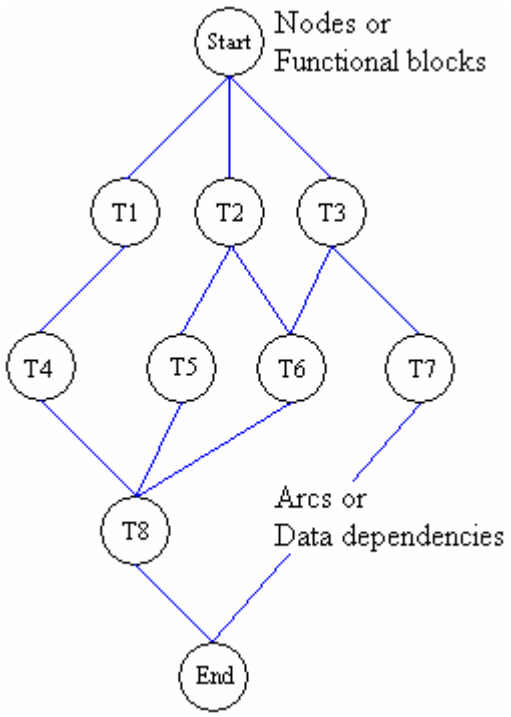
Another approach is based on accurate characterization of Client/Server workload [5], which is the basis for definition of input parameters of models. In this environment, a client generates requests that are transmitted through communication lines to the server, which processes them and sends the replies back to the client. Client/Server environments are hierarchically structured into three layers: client, network and server, and depending on the considered layer, the workload consists of the requests, as seen at the client or server layer, or of the packets owing to the network. Thus, at the Client/Server layer, it must be captured information such as arrival time of each request, size of requested files, URL, session, user, CPU and disk demands, access/modification time, etc. In the Network layer, measurements can span hours and days (see e.g.[6]) or weeks and months (see e.g. [7]) have been collected. The packets flowing on the communication lines are captured by software monitors, like *tcpdump,* which provide user level control of measurement collection, or by hardware monitors, like *sniffers,* which perform some preliminary analysis of the captured packets. Metrics like packets arrival time, port number, packet lengths, source and destination host must be captured.

4 Workloads in Parallel Systems

In parallel architectures it is very difficult to characterize the workload because of multiplicity of factors that affect its performance. The choice of suitable parameters and techniques able to describe the parallel aspects typical of these environments represents one of the most critical phases of workload analysis. In this chapter, a classification of the most relevant parallel parameters will be presented and more detailed survey can be found in [8].

The definition of workload is very complex since it may consist of a set of different programs or algorithms or it may be represented by the same algorithm executed under different configurations, for instance, number of processors, interconnection topologies, etc. The analysis of workload has several phases, the input represented by the data obtained from measurements taken on real systems, a few intermediate results, which give insights into the workload behaviour, and we produce final outputs for system modelling activities where we apply different analysis techniques.

The choice of the parameters to be used in the following steps is the preliminary step, and it is one of the most critical aspects in workload characterization. The general classification of parameters can be done in terms of their dependence or independence from system architectures: static indices and dynamic indices. The static indices provide an architectural independent description of the resource requirements of the algorithm and can be derived from task graph where each node represents a sequence of computation (tasks) and arcs represent data dependencies [9]. In the following figure, it is possible to see an example of a task graph:



Static Metrics:

N;
In-degree;
Out-degree;
Depth;
Maximum cut
Problem size;

Figure4 – Example of task graph

Thus, parameters and metrics can be gathered in order to obtain the description of the behaviour of parallel systems. The Total number of nodes N corresponds to the global number of tasks of application and is related to the granularity of the application (size of tasks); In-degree of a node corresponds to the average number of direct predecessors of all the nodes. This metrics reflects the synchronization complexity of the application, this is, a large value of the in-degree parameter means that many nodes require synchronization activities; Out-degree of a node represents the average number of direct successors of all the nodes. This metrics, as in case of the in-degree, reflects the synchronization complexity. The Depth or Critical path corresponds to the longest path between input and output nodes. This parameter represents the execution time of application, this is, the length of critical path gives the minimum completion time for the application; the Maximum cut corresponds to the maximum number of arcs taken over all possible cuts from the input node and ends the output node. This parameter reflects the maximum theoretical parallelism the application can achieve. At last, Problem size corresponds to the size of the data to be considered.

On the other hand, the Dynamic indices take into account both time and data dependencies and can be obtained by monitoring the execution of algorithm. With a single execution, it can be obtain simple parameters like 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 processors, etc which can give insights into the behaviour of the system and of the algorithm itself. Other parameters, expressed in curves describe the dynamic behaviour of the algorithm as a function of the execution time and of the number of available processors. Parameters like speedup, efficacy and efficiency represent the improvement achieved in the performance of the algorithm, and the cost due to increasing the number of processors. When the number of available processors is unlimited, parallelism profiles express the number of active processors as a function of execution time, and this represents the inherent parallelism in the algorithm under ideal conditions. In the following table is presented the Dynamic metrics – system dependent

Dynamic Metrics	Description
t_{comp}	Average computation time of the tasks
t_{comm}	Average communication time of the tasks
$n_{messages}$	Average number of messages sent_received by the tasks
$l_{messages}$	Average length of the messages
n_{I/O_op}	number of I/O operations
$T_{comp}(p)$	global computation time vs number of processors
$T_{comm}(p)$	global communication time vs number of processors
$T(p)$	global execution time vs number of processors
$S(p)$	Speedup vs number of processors ($S(p) = T(1)/T(p)$)
$E(p)$	efficiency vs number of processors ($E(p) = S(p)/p$)
$n(p)$	Efficacy vs number of processors ($n(p) = S(p)^2/p$)
n_{busy_proc}	number of busy processors vs execution time
n_{comm_proc}	number of communicating processors vs execution time
n_{comp_proc}	number of computing processors vs execution time

metrics derived from execution of the algorithm with p processors [1].

Table1 – Dynamic Metrics derived from the execution of the algorithm with p processors [1]

After selected parameters have been chosen, we must apply statistical analysis and visualization techniques [10]. Parallel metrics, such as profiles, are obtained and used for a preliminary characterization of the different experimental runs. The statistical properties of the measurements are examined by means of cluster analysis, which provide a classification of the tasks as function of their resource usages and of functional description which provide insight into the logical composition of each cluster.

5 Conclusions

At the end of this paper, we can conclude that, although workload characterization of different types of systems benefits from a common set of well-known analysis techniques, it is necessary some sort of specialization not only in the parameter identification and in the techniques applied for taking into account new architectural features, but also new operation modes.

Some problems, which make experimental studies harder, arise. The availability of appropriate monitoring tools for collecting the information of interest is required, but this is not always the case. For instance, when applications executed in parallel or distributed systems are to be measured, special events, like start/end of transmission from a processor,

or time spent by a server to process a remote file access, are to be detected. For the purpose, specific tools must be used. From the measurements, parameters able to describe and reproduce the static characteristics of the load and its dynamic behaviour are derived. Thus, various types of techniques are then applied in order to derive a compact and manageable representation of such parameters.

I came to the conclusion that the formal study of program behaviour has become an essential ingredient in guiding the design of new computer architectures which is also another important aspect. Accurate characterization of applications leads to efficient design of high performing architectures. Quantitative and analytical characterization of workloads is important to understand and exploit their interesting features.

To end of this communication, I can complete the main idea that in different systems we will have different loads and consequently, the evaluation performance of a certain machine will have to be performed in a different way depending on its workload.

References

[1] M. Calzarossa, and G. Serazzi, Workload Characterization: A Survey. *Proceedings of the IEEE 81*, 8 (Aug. 1993), 1136-1150.

[2] R. Friedrich, J. Martinka, T. Sieknecht, M. Chelliah, and A. Ranganathan. A Distributed Performance Measurement System for Large-Scale Heterogeneous Environments. Technical Report NSA-93-031, Hewlett-Packard Company, 1994.

[3] M. Calzarossa, G. Haring, and G. Serazzi. Workload Modeling Computer Networks. In U. Kastens and F. J. Ramming, editors, *Architekture und betriebe von Rechensystemen*, pages 324-339. Springer-Verlag, 1988.

[4] S.V. Raghavan, D. Vasukiammaiyar, and G. Haring. Generative Networkload Models for a Single Server Environment. In *Proc. ACM SIGMETRICS Conf. on Measurement and Modelling of Computer Systems*, 1994.

[5] M. Calzarossa, L. Massari, and D. Tessera. Workload characterization Issues and Methodologies, 2000.

[6] V. Paxson and S. Floyd. Wide-area Traffic: The Failure of Poisson Modelling. *IEEE_ACM Trans. on Networking*, 3(3):226-244, 1995.

[7] V. Paxson. Growth Trends in Wide-area TCP Connections. *IEEE Network*, 8(4):8-17, 1994.

[8] M. Calzarossa, L. Massari, and A. Merlo. Analysis and Characterization of the Workload of Parallel Systems. Technical report CNR Progetto Finalizzato “Sistemi Informatici e Calcolo Parallelo”, 1994.

[9] M. Calzarossa, and L. Massari. Measurement-Based Approach to Workload Characterization. In *7th International Conference on Modelling Techniques and Tools for Computer Performance Evaluation* (1994), R. Marie, G. Haring, and G. Kotsis, Eds.

[10] M. Calzarossa, L. Massari, A. Merlo, M. Pântano, and D. Tessera. MEDEA: A Tool for Workload characterization of Parallel Systems, 1994.