INNODOCT 2018
Valencia, 14th-16th November 2018
DOI: http://dx.doi.org/10.4995/INN2018.2018.8908

Implement of a high-performance computing system for parallel processing of scientific applications and the teaching of multicore and parallel programming[1]

Ph. D. Apolinar Velarde Martinez
Instituto Tecnológico el Llano Aguascalientes, Aguascalientes México

Abstract

Increasingly complex algorithms for the modeling and resolution of different problems, which are currently facing humanity, has made it necessary the advent of new data processing requirements and the consequent implementation of high performance computing systems; but due to the high economic cost of this type of equipment and considering that an education institution cannot acquire, it is necessary to develop and implement computable architectures that are economical and scalable in their construction, such as heterogeneous distributed computing systems, constituted by several clustering of multicore processing elements with shared and distributed memory systems. This paper presents the analysis, design and implementation of a high-performance computing system called Liebres InTELigentes, whose purpose is the design and execution of intrinsically parallel algorithms, which require high amounts of storage and excessive processing times. The proposed computer system is constituted by conventional computing equipment (desktop computers, lap top equipment and servers), linked by a high-speed network. The main objective of this research is to build technology for the purposes of scientific and educational research.

Keywords: *Single Instruction Multiple Data,Multiple Instruction Multiple Data, Grid Systems, Cluster, Cloud Computing, MPI, Threads, High Speed Computer Network.*

[1] This project is sponsored by Tecnologico Nacional de México TecNM. 2018-2 110

1. Introduction

Without loss of generality, and according to the definitions proposed in the literature, the systems of parallel and distributed computation or high-performance computing systems are defined, such as systems that agglutinate a certain number of processing elements, also called processors or nodes. that are physically separated; these processing elements, work together for the solution of tasks or jobs that require large amounts of computing time (Ragsdale, 1992) (Flynn, 1966) (Pacheco, 2011).

The implementation of different parallel computing systems such as multiprocessor systems, multicomputer systems (Ragsdale, 1992) (Velarde, 2016) and currently cluster systems (clusters), grid systems (grids) and cloud systems (Hameed, 2013), have been motivated by two important aspects: first, the current information processing requirements for the solution of scientific applications, which require computer systems with data processing speeds, greater than those offered by conventional computer systems, such as personal computers and servers with a single processor; Some of such scientific applications are: such as the analysis of satellite data (Briceño, 2013), natural language processing, recognition and digital processing of images, data analysis, data mining, among others. Second, because of the limitations currently imposed by processor speeds, which are caused by problems with energy consumption and heat dissipation, produced by the integrated circuits inside the devices (Pacheco, 2011) (Qamhieh, 2013) (Flynn, 1966).

Parallel computing systems have been classified by two parameters, which distinguish them: by software and by hardware. For example, in (Flynn, 1966), the classification is based on the number of the instruction flow and the number of the data flow, which the computer system uses in the processing of the algorithms; thus, in (Flynn, 1966) there are single-instruction systems that process multiple data (SIMD, for its acronym in English Single Instruction Multiple Data) and multiple-instruction systems, which process multiple data (MIMD, for its acronym in English Multiple Instruction , Multiple Data) this type of systems in turn are classified into shared memory systems and distributed memory systems. In (Hameed, 2013), the current high performance computing systems are classified into three groups: clusters, grids and clouds, this classification is based on the type of architecture that the hardware has, and the type of software that each system handles.

In this work, we present the implementation of a high-performance computing system for the parallel processing of scientific applications and the teaching of multicore and parallel programming, based on the distributed memory scheme of the MIMD systems. The implementation of a system with this type of architecture is due to the fact that these systems have three important advantages in relation to other parallel computing systems, as explained in (Pacheco, 2011):

1. they offer a higher absolute performance compared to shared memory schemes, providing uniform and faster memory access times,
2. they are designed to be scaled to hundreds or thousands of processors, adding more hardware with different and complex processing potentials, allowing a great data storage, and
3. reduce or eliminate central and global resources that produce bottlenecks that increase complexity in the system, when the number of processors in the system also increases.

The general structure of a MIMD system or multicomputer system, is a locally concentrated set of autonomous processing nodes, weakly coupled, with an identical structure in which each node has its own private memory (Tannenbaum, 2000). Each node itself can consist of a strongly coupled multiprocessor system, as shown in Figure 1, and referenced in (Nehmer, 1987). When the nodes are presented as a strongly coupled system, then a MIMD system can also be considered, a SIMD system (Flynn, 1966).

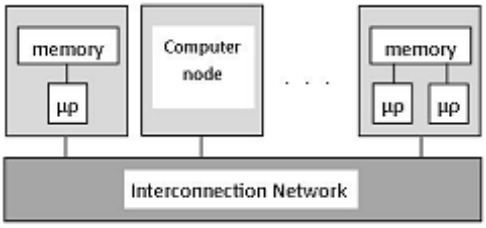

Figure 1. General structure of a multicomputer system

According to the previous paragraph, the expansion of the application area for parallel computing will lead to an enormous need for software developers with parallel programming skills; some chip manufacturers already demand to include parallel programming as a standard course in computer science curricula (Rauber, 2010).

The objective of this research is to present the theoretical foundation, the justification for the development, and the hardware implementation of the project: Liebres InTELigentes, a distributed memory MIMD system, for the processing of parallel algorithms. The technologies used for the development of the proposed algorithms are: MPI, OpenMP and threads. The proposed algorithms implement scientific applications for educational and research purposes, and are programmed with the specified tools.

The educational purposes that are pursued with the development of the system are:

1. allow engineering students in information technology to have access to the programming of parallel computing systems and multicore systems with different programming languages.

2. allow students of engineering in agronomy, business management and others, to use software for mathematical modeling,
3. strengthen the preparation of students in distributed environments for the current needs of the industry, and
4. establish comparisons of speeds and performances between single-user systems and multi-user systems

The organization of this work is as follows: in the classification section of parallel computing systems, the best known classification of this type of systems is presented; in the basic concepts section, the formal definitions of the parts that make up a high-performance computing system are presented; In the following section, we describe some examples of Multicomputer systems that have been developed for educational purposes, for research purposes or for commercial purposes; in the section justification of the design, development and implementation of the parallel computing system, the causes that have led to the development of this type of architecture are explained; The system implementation section lists the hardware characteristics of the computing equipment that make up the Liebre InTELigentes system. Finally, the projects that are intended to be developed with the installed equipment are mentioned in the future works section. The conclusions that we have reached with the development of this work are described at the end of this research work.

2. Classification of parallel computing systems

Over the years, different classifications of parallel computing systems have been exposed; In our work we address the classifications made in (Flynn, 1966) and (Hameed, 2013); These classifications allow us to carry out the theoretical foundation and the justification of why, a parallel system was implemented with the characteristics described here.

The classification proposed in (Flynn, 1966), called Flynn's taxonomy, is the classification that is frequently used to classify parallel computing architectures. This taxonomy classifies the systems according to the number of instructions flow, and the number of data flow that the system can handle simultaneously (Flynn, 1966) (Ragsdale, 1992) (Pacheco, 2011). This classification is the following:

Systems of a single instruction, multiple data (SIMD). This type of system operates on a multiple data flow by applying the same instruction to multiple data elements; thus, an abstract SIMD system has a single control unit and multiple arithmetic-logic units. Within this type of systems, Vector Processing Systems and Graphic Processing Units are also considered (Ragsdale, 1992).

Multi-Instruction, Multiple Data Systems (MIMD). This type of systems supports multiple instructions simultaneously operating on multiple data streams. MIMD systems consist of a collection of independent processing units or cores, each of which has its own control unit and its own arithmetic-logic units (ALU). MIMD systems are asynchronous, that is, processors can operate at their own pace 2. In many MIMD systems, there is no global clock and there can be no relationship between system times on two different processors. Unless the programmer imposes some synchronization, the processors will execute exactly the same sequence of instructions in a given time or they may be executing different instructions at the same time.

MIMD systems are classified into two basic types, according to the way they access the main memory of the data:

1. Shared memory systems
2. Distributed memory systems

In this paper, both systems are described briefly, for space reasons. Shared memory systems (Ragsdale, 1992), use one or more multi-core processors, which can be directly connected to a memory, or each processor can have a direct connection to a main memory block, and processors can access any block of memory through of a special hardware, built inside the processor. Figure 2 extracted from (Ragsdale, 1992), shows a shared memory system.

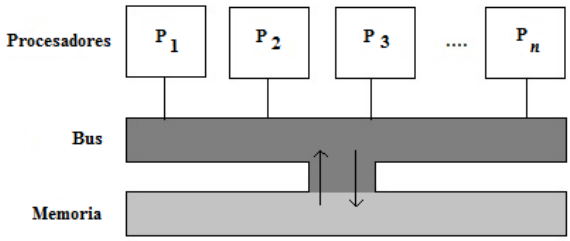

Figure 2. Shared memory system.

Distributed memory systems. In this type of systems the contents of the memory of a node can be accessed only by the processor of that node (the memory is local to the node). When the processor of one node requires information from another node, the information must be sent explicitly as a message from one node to another. For the programmer, this means that there are no shared variables, and there is no way for a processor to affect the data of another processor untimely. Figure 3 shows a distributed memory system extracted from (Ragsdale, 1992).

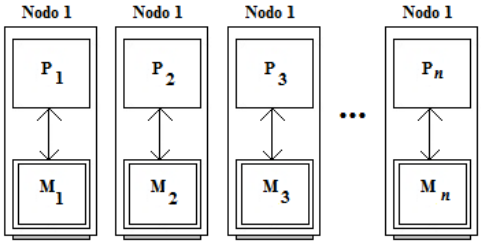

Figure 3. Distributed memory systems

These definitions offer a general idea of how the hardware is constituted, of the main types of parallel systems that have been developed over time, and shows the general context in which this research project is developed.

3. Related works

During the evolution of parallel computing systems, different architectures have been proposed for educational, research and commercial purposes. The system described in this paper is not intended to compare with other computer systems developed, in terms of speed, storage capacity, etc., but with the sole purpose of showing some examples with their main characteristics of hardware, software and type of applications that run.

INCAS Multicomputer Project (Nehmer, 1987). In its initial stage, it was built with 10 MC68000 microprocessors interconnected by a logical communication ring. The objective of the development of this system has been to develop two distributed programming languages: LADY and CSSA, the underlying philosophy of both languages was to structure the distributed software for parallel programming, developing a hardware ad oc. The structure of INCAS is made up of four logical levels: the physical network level, the LADY system support level, the level of the distributed operating system and the distributed application level. The INCAS project considers within its design the complete software spectrum of Multicomputer systems, such as: distributed operating systems, distributed programming languages, distributed applications and a methodology for distributed tests.

CM-5 Multicomputer (Z. Bozkus). It is a multiprocessor distributed memory system. The processors are interconnected using three networks: data network, control network and diagnostic network. The data network is used for communication between processors. The control network is used to execute operations that require the participation of all nodes simultaneously such as broadcast and synchronization. The communication between two nodes can be made with the data network and the control network. The diagnostic network is used for the maintenance and diagnosis of system failures. The CM-5 multicomputer system is built with SPARC microprocessors from SUN Microsystems. Each

microprocessor contains 4 unit vectors that function as memory controllers, a 33 Mhz clock, a 64 Kbyte cache used for instructions and data. The microprocessors are rated at a maximum performance of 22 million instructions per second (5 Mflops). This system, implemented in the School of Computer Science, within the Center for Science and Technology of the University of Syracuse, NY USA, has the purpose of developing scientific applications for educational purposes.

Crystal Multicomputer (J. DeWitt). It is a multicomputer system based on 64-bit INTEL processors, with a communications network with a ring topology (token ring); each node has 2 network cards that allow point-to-point communication. The objective of the development of this system is to design and implement parallel software for scientific applications within the University.

M-Machine Multicomputer (M. Fillo). It is a multicomputer system with 3D mesh architecture. In this system each node consists of a multi-ALU chip (for its acronym in English, Arithmetic Logic Unit) and a DRAM unit (for its acronym in English, Direct Random Access Memory). The central card of each node includes the network card that functions in turn as a router and provides a bandwidth of 800 MBytes per second. The input devices output can be connected to the controllers of each node. The M-Machine Multicomputer system was designed to solve inherently parallel problems of fixed size, rather than to achieve the maximum performance of the equipment that constitutes it, in the solution of problems; for this, the nodes are designed to handle the parallelism at the instructional level until reaching the process level.

4. Justification of the design, development and implementation of the parallel computing system

The development of this project has two main aspects: the aspect of scientific research and the educational aspect. In the first, the Liebres InTELigentes system processes scientific applications that due to the high requirement of computational processing, conventional computing equipment is insufficient, such as the genetic algorithm of islands and the evolutionary algorithms that process more than one variable. Some of the projects where these types of algorithms are applied are: the problem of the quadratic assignment in the planning of tasks in parallel computing architectures (Velarde A. , 2014). Analysis of the Opposition of the Objectives and the Pareto Front in the Planning and Assignment of Tasks in a Multicomputer System (A. Velarde, 2014), whose objective is to show the results obtained when evaluating the different opposing objectives in the planning and allocation processes of processors in a Multicomputer system. EVIA (Interactive Virtual Learning Spaces), which is constituted as a repository of contents of subjects that serve as a complement to the subjects taught in person.

In the educational aspect, in personal computer equipment such as laptops and personal desktop computers, it is becoming increasingly common to use processors that contain architectures with multiple processing cores (from the dual core to the Intel Core i7), which, through the programming of threads with languages such as JAVA (Rusty, 2005) (S. Oaks, January 1999), support the execution of multiple tasks in parallel, so it is necessary to study subjects where learners learn and interact, and take subjects that imply the management, and operation of high performance computing systems at the professional level, which will allow them to solve real problems that professional life poses, as well as develop research projects in the area of systems communication, computer networks , applicability of high performance computing, among other areas.

5. System Implementation

An additional advantage that has motivated the construction of the Multicomputer system described in this paper has been the implementation cost, which is lower in relation to the acquisition of multiprocessor shared memory systems, as well as being flexible and scalable in to the number of computing equipment that can be attached to the architecture. This characteristic shows this type of systems as a viable alternative for its implementation and, therefore, for the processing of scientific parallel applications and as platforms for teaching in higher education institutions.

The three levels that constitute the objective system developed are: the level of hardware, the level of communication and the level of software. As it was stated at the beginning, the objective of this work is to show only the hardware implementation, since in the implementation of the software, we are currently working.

The hardware level is made up of 2 servers equipped with a 4-core processor (multicore) each, which allows the programming of intrinsically parallel applications in two aspects, as established in (Pacheco, 2011) (H. Jin, 2011):

a. Through the MPI (Message Passing Interface) programming libraries that is the current design standard in MIMD systems, and
b. Using the Open Message Interface libraries, which allow programming in multi-core systems

According to the requirements of the research project that is developed, the programmer can choose any of the two aspects.

The communication system is with a Switch, which allows the communication equipment to equipment, equipment to all the equipment, and all to all the equipment, when the

programming is developed through the MPI libraries. When the programming is done for a multi-core system, the programmer decides on which node will run his application.

The software system, is the LINUX operating system with support for parallel architectures.

6. Future research work

Once the hardware design phase has been completed, a set of research projects have been proposed that will be implemented within the proposed system. For reasons of space, only three of these projects are described below:

The first of these is the continuity of the Interactive Virtual Learning Environment (EVIA) project, which is intended to be complemented using a skills approach, which will be directed to the facilitator and the student, through the website, and allow, among others things, publish more content of subjects, online exams, student-facilitator interactivity and student-student, use of electronic messaging, and any information that the facilitator wishes to publish for the good performance of the student in the subjects that he / she studies in his / her career.

The second project, the configuration of an accessible environment for the students of the career of Engineering in Information Technology and Communications, is an environment that allows access to the Multicomputer system for the realization of the design, development and execution of parallel evolutionary algorithms, parallel genetic algorithms and algorithms developed for the solution of real problems that require high computing power developed in some of the languages described in the previous section.

The third project multicore programming, seeks to develop applications that allow to plan tasks that seek to run in cores of distant nodes, to exploit the implicit parallelism in systems, that for the execution of the processes, has more than one core of execution. Examples of this type of systems can be seen in (R. Thakur, 2009) (A. Merigot, 2008).

7. Conclusions

Current computing equipment, both desktops and laptops, have processors that execute increasingly fast tasks, due to the increase in the number of processing cores. While the processors are getting faster in the execution of tasks, it is also necessary to create parallel algorithms that exploit these processing speeds, making use of the different kernels contained within the processors. This paper presents the implementation in hardware of a parallel computing system that allows the design, programming and execution of parallel

algorithms applied to problems, which require excessive data processing and storage times, in two aspects: for educational purposes and for investigation. The educational purposes are with the intention that students at the undergraduate level, can study subjects related to the area of parallelism and they are able to develop programs in real environments. The research aims are aimed at executing algorithms that seek to solve problems that require high data processing times, which conventional equipment can not offer.

References

A. Velarde, E. P. (2014). Análisis de la Contraposición de los Objetivos y el Frente de Pareto en la Planificación y Asignación de Tareas en un Sistema de Multicomputadoras. *CONGRESO INTERNACIONAL DE INVESTIGACION DE ACADEMIA JOURNALS* (págs. 5190-5197). Celaya, Guanajuato, Mexico: Editorial Academia Journals.

Briceño, L. D. (2013). Robust static resource allocation of DAGs in a heterogeneous multicore system. *Jounal of Parallel and Distributed Computing*, 1705-1717.

Flynn, M. (1966). Very High-speed Computing Systems. *Proceedings IEEE 54*, 1901-1909.

Hameed, H. e. (2013). A survey on resource allocation in high performance distributed computing systems. *Parallel Computing 39 ELSEVIER*, 709–736.

J. DeWitt, R. F. (s.f.). The Crystal Multicomputer: Design and Implementation Experience. Computer Science Department. University of Wisconsin . Wisconsin , Madison, EUA.

M. Fillo, W. K. (s.f.). *The M-Machine Multicomputer. Massachusetts Institute of Technology Artificial Intelligence Laboratory. .* Obtenido de http://publications.ai.mit.edu

Nehmer, J. e. (1987). Key Concepts of the INCAS Multi computer Project. *IEEE Transactions on Software Engineering Vol. SE-13, NO. 8*, 913 - 923.

Pacheco, P. (2011). *An Introduction to Parallel Programming.* Burlington MA USA: ELSEVIER.

Qamhieh, M. e. (2013). Global EDF scheduling of directed acyclic graphs on multiprocessor systems. *Proceedings of the 21st International conference on Real-Time Networks and Systems*, 287-296.

Ragsdale, S. (1992). *Parallel Programming.* USA: McGraw-Hill.

Rauber, T. a. (2010). *Parallel Programming for Multicore and Cluster Systems.* Berlin Heidelberg: Springer-Verlag.

Rusty, E. (2005). *Java Network Programming.* O'REILLY.

S. Oaks, H. W. (January 1999). *JAVA Threads 2nd Edition Java 2.* O'REILLY.

Tannenbaum, A. (2000). *Distributed Systems.* Washington D. C. USA: Addison Wesley.

Velarde, A. (14 de Febrero de 2014). Planificación y Asignación de Tareas en un Sistema de Multicomputadoras. . *Tesis Doctoral. 2014.* . Aguascalientes, Aguascalientes, México: Universidad Autónoma de Aguascalientes.

Velarde, A. (2016). *Planificar y asignar tareas en un sistema de multicomputadoras utilizando algoritmos evolutivos.* España: Editorial Académica Española.

Z. Bozkus, S. R. (s.f.). *Modeling the CM-5 multicomputer.* . Obtenido de Center for Science and Technology, Syracuse University. Syracuse, NY: http://www.syr.edu