Chapter 6: Conclusions and Further Work

This Chapter presents the main conclusions of this thesis, drawn from the previous chapters, with a special emphasis on the identification of problems and solutions to obtain the maximum performance with parallel computing in computers local networks. The initial context of the parallel processing hardware is given by the computers local area networks already installed and which can be used up in order to solve problems in parallel. A brief summary of the contributions made by this thesis is also presented together with its relation to the work published during this thesis development.

This chapter also presents some considerations as regards the continuation of the research in this area. Even though it is difficult to properly estimate extensions, it is indeed possible to identify quite clearly some of the immediate problems that can be solved within this parallel computing context, and some hardware-use alternatives beyond a computer local area network.

6.1 Conclusions

Parallel computer evolution has clearly advanced in several directions, one of which is the use of standard computing hardware. In this sense, massive-use microprocessors in low-cost computers, such as workstations and PCs, are used in those parallel computers that count with the highest, absolute computing capacity [86]. Local computer networks can be placed at the lowest level in terms of parallel computing cost; these networks have the same base processors type but they have not been originally oriented to parallel computation. In fact, the best parallel hardware cost/benefit relation is that of the already installed local networks since they do not entail any installation nor maintenance cost because its presence is independent of the parallel computation. However, other additional costs inherent to these networks cannot be left aside, such as that of parallel computing specific software's installation and maintenance and that of the low availability of those computers that, as previously mentioned, do not have parallel program execution as priority.

Both the already installed computer local area networks used for parallel computation and Beowulf-type installations evolving towards the replacement or addition of computers often have heterogeneous computing hardware and homogeneous communications hardware. The heterogeneity of the installed local network computers is more or less "natural", both taking into account setup time and its subsequent evolution and the different functions or types of problems towards which each local network computer is oriented. Heterogeneity of Beowulf-type installation computers is a non-expected consequence of low cost hardware evolution using PCs as basis. Availability time in the market of basic components, such as a type of processor, memory, and PCs' hard disks, is really short. Thus, whenever more computing power (more PCs) is needed or it is necessary to replace a computer that stops working, the likelihood of having an heterogeneous hardware is rather high in comparison to that of a "traditional" parallel computer under the same circumstances. And the likelihood increases as time passes and components are not readily available in the market.

Communications hardware homogeneity is also given by the low cost, in this case of Ethernet communications' interface cards (NIC: Network Interface Cards). The standard defined as Ethernet, in its various versions, has been installed as the one with lowest cost, and will seemingly keep this trend in all its versions defined up to the present: 10 Mb/s, 10/100 Mb/s, 1 Gb/s, and 10 Gb/s. In fact, most of the local networks use Ethernet of 10Mb/s since it has proved to be extremely useful for most of office applications, and Beowulf installations are recommended to be used with 100 Mb/S communications hardware and with 100 Mb/s and/ or 1 Gb/s Ethernet communications switch-based wiring. The low cost of these networks not only includes the very computer's interconnection hardware (NICs), but also all the technical staff already trained and qualified - lacked by other types of the networks used.

Both the computing heterogeneity and Ethernet processor interconnection networks (from the point of view of a parallel machine) have very well-defined characteristics which are not specifically apt for parallel computation. Computing heterogeneity presents a problem that should rarely (if not *ever*) be faced in traditional parallel computers, i.e. unbalance

given by the different computing capacities of processors. Ethernet interconnection hardware presents more serious problems:

- Ethernet is not *a priori* oriented to parallel computation and thus the performance indexes, such as latency and bandwidth, are quite higher than those of parallel computers' interconnection networks. In other words, Ethernet communication networks' performance is not *balanced* according to the computers' processing capacity.
- The access method to the "only" communication means defined by the standard, CSMA/CD (Carrier Sense-Multiple Access/Collision Detect), makes the interconnection network performance highly dependent on the traffic and wiring (with the use of switches, as example).

Consequently, it is necessary to revise quite exhaustively parallel algorithms in order to identify problems and solutions in the context of this *new* parallel hardware provided by the heterogeneous computer network. In no case should we overlook that parallel processing's *raison d'etre* is the performance increase with respect to that provided by sequential processing. The revision of parallel algorithms tends to be made case by case, at least in terms of application or problem areas to be solved using parallel processing.

Linear algebra applications constitute one of the biggest problem areas that have traditionally been solved taking advantage of the performance provided by available parallel computing architectures. Within linear algebra applications, a set of operations or, more directly, computing routines have been identified; these are considered as basic and of extensive use in most of the problems included within this area. Such routines have been called BLAS (Basic Linear Algebra Subroutines) and, both for their classification and the identification of their memory and computing requirements, they have been divided in three levels: level 1, level 2, and level 3 (Level 1 or L1 BLAS, Level 2 or L2 BLAS, and Level 3 or L3 BLAS). From the performance point of view, level 3 routines (L3 BLAS) are those to be optimized in order to obtain a near optimal performance of each machine and, in fact, many standard microprocessors' companies provide BLAS libraries with a strong emphasis in the optimization and the resultant routines performance included in level 3 BLAS.

Matrix multiplications can be considered as the mainstay or the routine from which the remaining routines included in level 3 BLAS can be defined. For this reason, and/or due to its simplicity, most of the research reports on this parallel processing area begin with the "problem" of matrix multiplication in parallel. In other words, by optimizing matrix multiplications, the complete 3 level BLAS is optimized in some way, and thus most of linear algebra-based applications - which also depend on routines optimization carried out by operations arising from linear algebra - would be also optimized. Although this optimization is not necessary direct, it can indeed be asserted that the processing type that should be applied to solve a matrix multiplication is quite similar to that of the rest of the routines defined as level 3 BLAS, and even similar to the more specific problems solved by linear algebra operations. In this sense, it very likely that what is used to optimize the matrix multiplication (in parallel or not) will be usable and/or utilizable in other operations. For this reason, this thesis is oriented to matrix multiplication in parallel with some comments on generalization.

Specifically focusing on matrix multiplications in parallel, and analyzing the proposed

algorithms, we reached the conclusion that they are quite oriented to traditional parallel computers. In fact, each parallel algorithm of the matrix multiplication can be identified as specially appropriate for shared-memory parallel computers (called multiprocessors) or for distributed-memory or message-passing parallel computers (called multicomputers).

Parallel algorithms oriented to multiprocessors are not appropriate for distributed-memory parallel computing systems. What is more, computers networks in general (Beowulf installations, heterogeneous systems, etc.) are specially inappropriate due to the low coupling level or, more specifically, the distribution or division of the available hardware for parallel computation.

Parallel algorithms oriented to multicomputers still have a base closely linked to traditional parallel computers hardware. More specifically, when these algorithms were proposed, several underlying parallel hardware characteristics were assumed, namely:

- Processor interconnection via a mesh or two-dimensional torus, tree arrays or hypercubes. This means that the interconnection network has the possibility of multiple point-to-point connections and multiple optional ways of transferring data among processors.
- Homogeneous processing elements. This implies that the load balance is trivial and is directly given by the distribution of the same data quantity of the involved matrix to all processors.

None of the previous characteristics can be kept in heterogeneous computers local networks. Consequently, it is necessary to develop algorithms that make an efficient use of the characteristics of these *new* parallel architectures. On the one hand, these algorithms must be ready for the differences in computing capacities of those machines interconnected by local networks; and on the other, they must use to the maximum the performance and characteristics of Ethernet interconnection networks. And these are the two supporting basis of the matrix multiplication parallel algorithms proposed in this thesis (in fact, they can be considered as two variants of a same parallel algorithm):

- Load balance given by the data distribution which, in turn, is made according to the each computer's computing relative capacity.
- Only broadcast type communications, so that Ethernet networks' capacity is used to the maximum.

As the experimentation chapter shows, the mere fact of proposing an "appropriate" algorithm does not assure an acceptable or scalable performance. In fact, such as the experiments results show, when using the PVM communication library, and when machines are added to carry out parallel computation (increase of computing capacity) and to solve the same problem, the performance is reduced. What is more, depending on the computers, this performance reduction can be radical to the point of obtaining a worse performance than with a single computer. In this case, the parallel computing time ends up being dominated by the time necessary for broadcast messages.

Since local computing performance is satisfactory, it is necessary to upgrade the performance of broadcast messages in order to obtain an acceptable performance for this problem. As previously discussed, it is really difficult to assure *a priori* that the broadcast messages implementation proposed by "general purpose" libraries, such as PVM, and MPI

implementations are specifically made in order to make use of Ethernet networks' broadcast capacity. Consequently, a *new* broadcast message routine is proposed among processes directly based on UDP protocol, which is as used as, or more used than, the very Ethernet networks. Even though this broadcast message routine can be extended to a whole collective communications library, the purpose is not, in principle, to define a *new* library or replace existing libraries. Ethernet networks are indeed to be used to their maximum and, thus, the implementation of the most used routines and/or those on which the performance depends should be adapted to the characteristics and capacities of these computer interconnection networks.

Matrix multiplication performance in parallel is acceptable in heterogeneous local networks when the algorithm specifically proposed for these is implemented using a broadcast routine that makes use of Ethernet networks' capacities. Consequently and as expected, from the performance point of view, at least two aspects are combined to obtain the maximum of the installed local networks that can be used for parallel computation: algorithm and implementation in general, and broadcast messages implementation in particular. In other words, without a suitable algorithm, a good performance cannot be obtained, and even with a suitable algorithm the performance is not acceptable if the implementation is unsuitable. In this case, the most problematic part of the implementation has been that of broadcast messages. Since none of the message-passing libraries properly implements broadcast messages - or, at least, we cannot ascertain *a priori* that it really does – a specific routine has been developed to solve the performance problem. Once more, we should recall that the performance is the *raison d'etre* of parallel processing, in general, or at least of the parallel processing used to solve numerical problems, in general, and linear algebra operations, in particular.

In particular, LIDI network shows that the proposed algorithms are also suitable for Beowulf-type installations, and/or with homogeneous processing hardware and an interconnection network with better performance than that of the installed local networks. In the case of *traditional* parallel computers, the use of this algorithm is not so immediate or unconditional. In multiprocessors, it seems unnecessary using *a priori* a "new" parallel algorithm since those proposed are really suitable or, at least, more suitable than any proposed for multicomputers. In the case of multicomputers, special attention should be paid to broadcast message implementation and performance. In this sense, static interconnection networks (with limited and predefined point-to-point links) often impose certain limits to scalability and the subsequent performance of broadcast messages. In any case, multiple research efforts oriented to the upgrading of collective communications and broadcast messages performance in these particular computers can be used by the algorithms proposed to multiply matrices in parallel. From the point of view of the solved problem, two important issues should be taken into account:

- The problem is not significant in itself, since it is not frequent to solve a problem which only implies a matrix multiplication. Usually, a matrix multiplication is part of or is used, among other operations, to solve a general problem.
- As previously explained, matrix multiplication is representative as regards the processing of the entire level 3 BLAS and, thus, what is obtained in the matrices multiplication can be used in all the routines included in level 3 BLAS. Since that, in general, the most important points regarding performance are related to these routines (L3 BLAS), matrix multiplication optimization becomes a significant contribution to all

or most of linear algebra applications. This has been the general tendency, be it in sequential processors, parallel computers in general, multicomputers and/or computer local networks in particular.

Since that
- communication among processes is always solved via broadcast messages, and
- these messages implementation has been carried out taking advantage of Ethernet networks capacities,

the performance is scalable at least to the limit given by the minimum granularity. In any case, we should not forget that this minimum granularity is quite high in the case of local networks and depends on the communications hardware (10 Mb/s, 100 Mb/s, 1 Gb/s, etc.).

From another point of view, the same broadcast message routine based on UDP already implemented shows that local networks computing hardware heterogeneity should not necessarily be translated to communications performance. More specifically,
- Communications latency time depends on the computing capacity of the machines involved in a data transference.
- The asymptotic bandwidth and/or the transference time or relatively big messages is independent of the involved computers and depends on the communications network capacity.

As shown both in the very experimentation with matrix multiplications – and, specifically, in the Appendix C for point-to-point messages -, general purpose communication libraries such as PVM and free-use MPI implementations tend to make the communications performance dependent on computers' heterogeneity. This is due to the software layers that have to be added to solve multiple communications routines among processes that generally implement and imply an significant processing overhead.

Furthermore, the very broadcast message routine shows that it is possible to obtain a broadcast message among user processes that fulfill the following:
- Near optimal, absolute performance provided by communications hardware. Even though the routine is meant for broadcast messages, point-to-point communications performance (between two computers) is also highly satisfactory.
- Scalable performance, i.e. broadcast time is *almost* dependable on the quantity of machines involved. Evidently, synchronization and the way to confirm the reception of messages by each computer implies the existence of a cost per computer involved in a broadcast message, but this cost is much lesser in running time than the complete replication of the whole message to each receptor (machine) process.
- Portability, because the only requirements for using use this routine are IP (TCP and UDP) connectivity and a C compiler. In fact, by using standard protocols exhaustively, a communications hardware independence can be even attained. Though originally oriented to the utilization of Ethernet networks, the routine used to carry out broadcast messages is portable to any environment with IP connectivity. While no specific tests have been carried out, it is rather likely that, in computers interconnection networks whose hardware is not capable of data broadcast (such as ATM networks), this routine performance will be satisfactory at any rate.
- No extra requirement from the user's point of view of computer local networks used for parallel computation. In particular, it is not necessary to impose alterations in the operating system or priorities or processes with priorities beyond the available for user's

processes.
- Easy of use/Simplicity. In fact, in the programs over which the experimentation was carried out, the changes at source code level did not exceed the replacement of PVM routine used for broadcast messages. The rest of communications (which do not influence the performance, or its influence is minimum) were also carried out with routines provided by PVM.
- Heterogeneity management in the representation of data of those computers typically interconnected in a local network. In the same experimentation, different types of machines were used with different processors and their own numeric data types representations.
- Common use interface to that of the remaining general-purpose message-passing libraries. In fact, the implementation of this routine leads us to suppose that the rest of the routines commonly included in collective communications are relatively simple enough to have a complete collective communications library.

The algorithm that solves matrix multiplications in parallel with sequential periods of local processing and broadcast messages is simple and reliable as regards performance estimates. In this sense, we obtain a model of a parallel machine with the following characteristics:
- Capable of running simultaneously in each processor, just like any other belonging to the distributed memory MIMD type. There are no interferences among different processors (machines) to solve this local computation.
- Capable of carrying out broadcast messages relatively independently of the quantity of involved machines.
- Message interference over local computing performance is rather low.
- There is no interference of the local computation over communications performance.

And, on the other hand, we count with a parallel computing algorithm that, apart from taking advantage of these characteristics, involves a highly regular type of processing. Even though we cannot assure that every numeric problem will have such a regular processing, we can indeed ascertain that it is a characteristic similar to most of the routines and applications coming from linear algebra. The combination of this model of machine and this type of parallel algorithms makes the obtainable performance estimate very easy and relatively reliable. As a consequence, it is also possible to identify quite clearly when performance problems begin due to the granularity of the problems that are being solved. Thus, the very program of matrix multiplication in parallel with sequentially run or organized computing and communications periods can be used as a benchmark for the identification of the minimum granularity of a set of networked computers in a local network.

Beyond obtaining the best performance, the matrix multiplication algorithm in parallel –designed to overlap computation with communications - is particularly useful in order to identify performance problems. More specifically, implementing this algorithm makes it possible to clearly identify those computers capable of efficiently overlapping computation with communications and even which the penalization in terms of performance is. In this sense, in heterogeneous environments, there can be various penalizations in different machines, and, with the quantification of this penalization, the load balance can be upgraded in order to compensate the differences. A tool of this type becomes valuable when it works in multiple computers and provides information that is really hard to obtain by other means.

The computers involved in a local network may be those with lesser capacity both in terms of processing and main memory installed among all the available in the market. In this sense, the obtained gain by the use of a local network processing in parallel can be really big. The experimentation made by involving problems that went beyond the storage capacity of the best computer of each local network tries to quantify this gain. The basic underlying idea is: even if the best computer of a local network is used, there can be performance problems since it is not enough to solve a given problem, mainly due to the quantity of available memory in that computer. Even if it is possible to store a large quantity of data beyond the installed memory through the management of swap memory, the performance can be subject to a great penalization. Consequently, the use of the rest of the computers of the local network not only provides memory to store data but also allows *all* the computers to carry out their processing at their maximum speed. That is, in all the computers, the available resources can be used optimally or in an optimized manner.

Specifically, in terms of speedup values as performance metrics, we have shown something that is relatively simple but rarely frequent as regards research contributions: performance in heterogeneous environments is not directly related to the *quantity* of computers (or processors) that are being used. In this sense, traditional parallel machines, with their homogeneous processing hardware, have established that the maximum, possible speedup value is equal to the quantity of processors used. In heterogeneous environments, this cannot be maintained since processors do not necessarily have the same computing capacity. In fact, the line $y = x$, which the maximum speedup value has traditionally been related to, has allowed the interpolation of intermediate values, and this interpolation of intermediate values cannot be maintained in parallel processing environments with heterogeneous processors either.

One of the basis for obtaining a satisfactory and *predictable* performance with parallel processing is the use of the best sequential code for local computation. Also, if non-optimized local computation code is used, the performance estimate given by the speedup factor loses almost all of its meaning because the parallel performance is obtained as a combination of

- each computer's local performance,
- the quantity of operations that can be carried out simultaneously, and
- the performance of communications.

Appendix B shows in detail that when a non-optimized code is used, each computer's performance is highly dependant on the size of the problem, basically due to the relation existing between the quantity of data to be processed and the processor's cache memory capacity (specifically, of the first level cache memory). In general, in all distributed memory parallel architectures, and in the particular case of computer local networks used for parallel computation, when the quantity of processors is increased, each processor has problems every time smaller in terms of the quantity of data to process. This smaller quantity of data can take a better advantage of the cache memory space and, thus, the computing performance is significantly upgraded. Consequently, when local computing routines are not optimized, parallel performance does not necessarily improve by using more computers but because each computer solves a problem with less quantity of data and, thus, the local computing performance is significantly higher. On the other hand, the

completely optimized computing code makes the performance relatively independent of the size of problem being solved, and thus all gain obtained by parallel computation is

- "Real" since there is no other way of obtaining a better performance from the sequential computers being used because the sequential performance with which it is compared to is the optimum.
- Only due to the use of the highest quantity of computers or processors, since the size of problem does not influence significantly on the local performance of each machine.

6.2 Summary of the Contributions and Publications related to this Thesis

Contributions of this thesis in relation to the work published can be briefly enumerated. Initially, the basic problem of parallel performance in computers and local are networks should be identified quite accurately, and propose some type of solution. These two initial contributions can be summarized as follows:

1. **Analysis of the matrix multiplication algorithms in parallel for their use in computers local networks that can be used up for parallel computing.**
2. **Proposal of the parallelization guidelines used to design the algorithms proposed in this thesis.**

And these contributions are directly related to the publications:

- [135] Tinetti F., A. Quijano, A. De Giusti, "Heterogeneous Networks of Workstations and SPMD Scientific Computing", 1999 International Conference on Parallel Processing, The University of Aizu, Aizu-Wakamatsu, Fukushima, Japan, September 21 - 24, 1999, pp. 338-342.
- [137] Tinetti F., Sager G., Rexachs D., Luque E., "Cómputo Paralelo en Estaciones de Trabajo no Dedicadas", VI Congreso Argentino de Ciencias de la Computación, Ushuaia, Argentina, Octubre de 2000, Tomo II, pp. 1121-1132.

In which the experimentation specially oriented towards demonstrating that traditional algorithms are not necessarily useful in computers local networks is presented. On the other hand, the publications (in chronological order):

- [116] Tinetti F., "Aplicaciones Paralelas de Cómputo Intensivo en NOW Heterogéneas", Workshop de Investigadores en Ciencias de la Computación (WICC 99), San Juan, Argentina, 27 y 28 de Mayo de 1999, pp. 17-20.
- [117] Tinetti F., "Performance of Scientific Processing in Networks of Workstations", Workshop de Investigadores en Ciencias de la Computación (WICC 2000), La Plata, Argentina, 22 y 23 de Mayo de 2000, pp. 10-12.
- [124] Tinetti F., Barbieri A., Denham M., "Algoritmos Paralelos para Aprovechar Redes Locales Instaladas", Workshop de Investigadores en Ciencias de la Computación (WICC 2002), Bahía Blanca, Argentina, 17-18 de Mayo de 2002, pp. 399-401.
- [128] Tinetti F., Denham M., "Algebra Lineal en Clusters Basados en Redes Ethernet", Workshop de Investigadores en Ciencias de la Computación (WICC 2003), Tandil, Argentina, 22-23 de Mayo de 2003, pp. 575-579.
- [134] Tinetti F., Quijano A., "Costos del Cómputo Paralelo en Clusters Heterogéneos",

Workshop de Investigadores en Ciencias de la Computación (WICC 2003), Tandil, Argentina, 22-23 de Mayo de 2003, pp. 580-584.

Are more oriented to presenting the ideas as open and / or under-development research lines. It should be noticed that in each of the years in which we have participated in this congress, we have presented advances closely related to the research lines of the previous years.

Once drawbacks are identified and some general solution is proposed, it is necessary to test the proposal. The alternative chosen is to carry out this in a specific manner within the area of lineal algebra applications and of basic operations. Within this context, this thesis contributes with:

3. **A proposal of specific matrix multiplication algorithms in clusters, designed following the parallelization principles previously mentioned.**
4. **The use of the parallel matrix multiplication algorithm designed to overlap computing with communications in order to identify performance problems (as a benchmark, in a certain way).**

These algorithms have been presented together with the experimentation backing up its validity in the publications:

- [118] Tinetti F., “Performance of Scientific Processing in NOW: Matrix Multiplication Example”, JCS&T, Journal of Computer Science & Technology, Special Issue on Computer Science Research, Vol. 1 No. 4, March 2001, pp. 78-87.
- [131] Tinetti F., Luque E., “Parallel Matrix Multiplication on Heterogeneous Networks of Workstations”, Proceedings VIII Congreso Argentino de Ciencias de la Computación (CACIC), Fac. de Ciencias Exactas y Naturales, Universidad de Buenos Aires, Buenos Aires, Argentina, 15 al 18 de Octubre de 2002, p. 122.
- [132] Tinetti F., Luque E., “Efficient Broadcasts and Simple Algorithms for Parallel Linear Algebra Computing in Clusters”, Workshop on Communication Architecture for Clusters, International Parallel and Distributed Processing Symposium (IPDPS '03), Nice Acropolis Convention Center, Nice, France April 22-26, 2003.
- [136] Tinetti F., A. Quijano, A. De Giusti, E. Luque, “Heterogeneous Networks of Workstations and the Parallel Matrix Multiplication”, Recent Advances in Parallel Virtual Machine and Message Passing Interface, 8th European PVM/MPI Users' Group Meeting, Santorini/Thera, Greece, September 23-26, 2001, Proceedings, Yannis Cotronis, Jack Dongarra (Eds.), Lecture Notes in Computer Science 2131 Springer 2001, ISBN 3-540-42609-4, pp. 296-303.

Several of the previous publications also present another contribution of this thesis, specifically oriented to taking advantage of Ethernet networks, contribution that can be summed up as:

5. **A proposal of a UDP protocol-based broadcast message routine to optimize the use of Ethernet networks.**

In this case, the publications more specifically related are:

- [120] Tinetti F., Barbieri A., “Collective Communications for Parallel Processing in

Networks of Workstations", Proceedings SCI 2001, Volume XIV, Computer Science and Engineering: Part II, Nagib Callaos, Fernando G. Tinetti, Jean Marc Champarnaud, Jong Kun Lee, Editors, International Institute of Informatics and Systemics, Orlando, Florida, USA, ISBN 980-07-7554-4, July 2001, pp. 285-289.

- [123] Tinetti F., Barbieri A., "An Efficient Implementation for Broadcasting Data in Parallel Applications over Ethernet Clusters", Proceedings of the 17th International Conference on Advanced Information Networking and Applications (AINA 2003), IEEE Press, ISBN 0-7695-1906-7, March 2003.

This thesis also deals with some aspects of parallel computing performance in homogeneous clusters, which can be summarized as:

6. **A proposal of specific matrix multiplication algorithms and LU factorization of matrices in homogeneous clusters, designed following the previously mentioned parallelization principles. In fact, the matrix multiplication algorithm is the same as the presented for heterogeneous clusters, thus showing its direct utilization in homogenous clusters.**

In the context of homogeneous clusters, these algorithms are presented with the specific experimentation and/or in comparison with ScaLAPACK in some of the previous publications and in:

- [127] Tinetti F., Denham M., "Paralelización de la Factorización de Matrices en Clusters", Proceedings VIII Congreso Argentino de Ciencias de la Computación (CACIC), Fac. de Ciencias Exactas y Naturales, Universidad de Buenos Aires, Buenos Aires, Argentina, 15 al 18 de Octubre de 2002, p. 121.
- [130] Tinetti F., Denham M., De Giusti A., "Parallel Matrix Multiplication and LU Factorization on Ethernet-based Clusters", High Performance Computing. 5th International Symposium, ISHPC 2003, Tokyo-Odaiba, Japan, October 20-22, 2003, Proceedings. Series: Lecture Notes in Computer Science, Vol. 2858. Veidenbaum, A.; Joe, K.; Amano, H.; Aiso, H. (Eds.), 2003, XV, 566 p. ISBN: 3-540-20359-1
- [129] Tinetti F., Denham M., "Paralelización de la Factorización LU de Matrices en Clusters Heterogéneos", Proceedings IX Congreso Argentino de Ciencias de la Computación (CACIC), Fac. de Informática, Universidad Nacional de La Plata, La Plata, Argentina, 6 al 10 de Octubre de 2003, p. 385-396.

The last publication shows the first results obtained by using the parallelization principles for matrix LU factorization in heterogeneous clusters.

Even though the evaluation of communications is quite known, in this thesis it has a special relevance since it has been shown that the excessive penalization that can be imposed on parallel algorithms specifically designed to obtaining optimized performance. Appendix C also presents the complete methodology and the results obtained in terms of

7. **Assessment of communications performance from the perspective of parallel computing in heterogeneous clusters (point-to-point and collective operations).**

Which is depicted in the publications:

- [119] Tinetti F., Barbieri A., “Cómputo y Comunicación: Definición y Rendimiento en Redes de Estaciones de Trabajo”, Workshop de Investigadores en Ciencias de la Computación (WICC 2001), San Luis, Argentina, 22-24 de Mayo de 2001, pp. 45-48.
- [121] Tinetti F., Barbieri A., “Análisis del Rendimiento de las Comunicaciones sobre NOWs”, Proceedings VII Congreso Argentino de Ciencias de la Computación (CACIC), El Calafate, Santa Cruz, Argentina, 16 al 20 de Octubre de 2001, pp. 654-656.
- [122] Tinetti F., Barbieri A., “Cómputo Paralelo en Clusters: Herramienta de Evaluación de Rendimiento de las Comunicaciones”, Proceedings VIII Congreso Argentino de Ciencias de la Computación (CACIC), Fac. de Ciencias Exactas y Naturales, Universidad de Buenos Aires, Buenos Aires, Argentina, 15 al 18 de Octubre de 2002, p. 123.
- [125] Tinetti F., D' Alessandro A., Quijano A., “Communication Performance of Installed Networks of Workstations for Parallel Processing”, Proceedings SCI 2001, Volume XIV, Computer Science and Engineering: Part II, Nagib Callaos, Fernando G. Tinetti, Jean Marc Champarnaud, Jong Kun Lee, Editors, International Institute of Informatics and Systemics, Orlando, Florida, USA, ISBN 980-07-7554-4 July 2001, pp. 290-294.
- [133] Tinetti F., Quijano A., “Capacidad de Comunicaciones Disponible para Cómputo Paralelo en Redes Locales Instaladas”, Proceedings VIII Congreso Argentino de Ciencias de la Computación (CACIC), Fac. de Ciencias Exactas y Naturales, Universidad de Buenos Aires, Buenos Aires, Argentina, 15 al 18 de Octubre de 2002, p. 125.

Though not directly related to the context of the local networks installed, some researches have been carried out in relation to the performance of matrix multiplications in supercomputers or, at least, in traditional parallel computers, published in:

- [126] Tinetti F., Denham M., “Paralelización y Speedup Superlineal en Supercomputadoras. Ejemplo con Multiplicación de Matrices”, Proceedings VII Congreso Argentino de Ciencias de la Computación (CACIC), El Calafate, Santa Cruz, Argentina, 16 al 20 de Octubre de 2001, pp. 765-774.

Many of the conclusions reached at in this publication are directly related to Appendix B, which aims at showing the sequential performance of the computers used. In particular, the distorted notion of performance that can be attained when the code of the programs used are not specifically optimized for the solved application and the computing architecture used.

6.3 Further Work

As previously explained, the problem of matrix multiplication is not significant in itself, though representative of a set of data numerical processing problems. In the context of Level 3 BLAS routines, the immediate extension is:

- Using directly matrix multiplication for the implementation of all the routines included

in Level 3 BLAS.
- Using the parallelization principles used in the matrix multiplication in order to solve the rest of Level 3 BLAS routines.

The first option has the advantage of only having the cost of codifying the routines in terms of matrix multiplications. Although the second option does not have this advantage and involves the case-by-case parallelization associated cost, it has the advantage of allowing a wider range of possible gain due to parallel computation. Such as they are defined, Level 3 BLAS routines are a rather reduced quantity and, in case-by-case parallelization, the very characteristics of processing can be better utilized so as to obtain a better performance. Whatever the chosen alternative is, or even in the experimentation with both, the utilization of the ideas of parallelization of this thesis is quite direct.

A step further in terms of the extension of this thesis would be dealing with a complete problem of linear algebra. For instance, the first steps towards the solution of the problem of linear equation systems are currently taken. In a more general context, we would be able to progress in the direction of the problems solved by the LAPACK library, as a way of experimenting with a relatively wide range of problems coming from linear algebra. The advantage associated to the experimentation with LAPACK is that the library in itself has been used up to the present moment and, thus, there exists a relatively large quantity of potential users. It can be asserted that up to this point, i.e. within linear algebra operations and applications, the type of processing is quite similar to the matrix multiplication processing. Even though there exist several particularities, most of the operations:
- Are rather simple in terms of codification.
- Are well known in terms of solution methods.
- Have a very well-defined scope of data dependency and also have subsets of data that can be computed independently.

This extension of the work is supported by the fact that the parallelization of operations, such as multiplication and factorization of matrices, with the relativey simple principles presented by this thesis can obtain optimized code for the local networks interconned by Ethernet. In fact, the experimentation carried out with the purpose of comparing the proposed algortihms with those implemented by ScaLAPACK backs up this future research line.

The following level of extensions - rather more complex – is represented by numerical applications, in general, and all that non-linear processing involves, in particular. It is more complex from two points of view:
- Codification of methods for solving specific problems.
- Computation dependency relationships, which are not so structured as in most of linear algebra operations.

A specific area is that of signal processing, which has multiple applications and where the solution methods to specific problems are many and, several times, really unequal. The known, relatively simple computation in this context of a FFT (Fast Fourier Transform) involves, for instance, an access pattern to data, which is, in a certain way, regular though so specific that has directly given rise to *ad hoc* data addressing modes in the processors designed to process digital signals or DSP (Digital Signal Processor). Even though parallel principles within this area are the same –since they are considered for making optimal use of local networks computing resources and not for a particular processing area-, the application of these principles is not so simple in the case of matrix multiplications or the

rest of the operations or routines related to linear algebra.

From another point of view for research, it is always possible to think about extensions or, at least, experimenting with the possibility of using more than one local network. In this sense, and for linear algebra applications with their tightly coupled processing characteristics, it is very important to know up to which point the performance gain is possible using more than one local network. More specifically, the quantification of the penalization (for instance, in terms a minimum granularity) due to data distribution in multiple local networks is useful in order to characterize *a priori* the use utility of more than one local network for solving a problem in parallel.

In the case of multiple local area networks, the contribution of other simple but effective processing methods can be really significant, such as *pipelining* (similar to a traditional assembly line), or establishing specific "servers" for tasks specially penalized by the physical layout of the computers used. In this context, special care should be taken with all *remote* communications in the sense of transferring data among two or more computers belonging to different local networks. The specific case of broadcast messages, for instance, is still useful and simple in a local network and in all the local networks where they are used, but the implementation of these messages when several computers of different local networks are involved must be considered with much care in terns of traffic, congestion (competitions) of intermediate transport links among networks, latency time, etc. The strategy to follow is not so immediate now, even though the very acceptable performance that can be obtained in each local network favors, in a certain way, this research line.

The use of the three local area networks over which all the experimentation has been carried out is still possible, and a set of experimentations could be designed in order to analyze the results and, from there, propose use alternatives of each computer. In a way, the possibility of using more than one local network considerably increases the range of problems sizes that can be solved (independently of the fact that the problem could be multiplying matrices or any other) but it also adds rather "unknown" problems, at least in this context of linear algebra applications, such as the impact on the minimum granularity and scalability, now at a level of local networks. Another of the problems in this context is that of performance vs. storage capacity in main memory: What is preferable: a local network with greater storage capacity in main memory or with greater processing capacity? It is quite likely that local networks that count with greater storage capacity (adding the capacities of each interconnected computer) are also those with greater processing capacity, though this cannot be asserted since local networks are not necessarily designed for parallel computation, not even for parallel computation with other networks.

In an extension of this thesis, which could be called "at a large scale", both types of extensions previously mentioned can be combined:

- Extension in relation to other problems to be solved;
- Extension in relation to the use of more computers involving more than one local network.

Perhaps, in both cases, the problems will be quite greater in relation to the necessary processing, such as the quantity of data to be processed, but the basic principles of matrix parallelization can still be used, at least initially. In any case, from the problems identified

through the experimentation, others more specific and appropriate for obtaining the maximum possible performance can be proposed from the available resources.