Parallel Linear Algebra on Clusters

Fernando G. Tinetti

Investigador Asistente Comisión de Investigaciones Científicas Prov. Bs. As.[1]

III-LIDI, Facultad de Informática, UNLP

50 y 115, 1er. Piso, 1900 La Plata

Argentina

fernando@info.unlp.edu.ar

Abstract

Parallel performance optimization is being applied and further improvements are studied for parallel linear algebra on clusters. Several parallelization guidelines have been defined and are being used on single clusters and local area networks used for parallel computing. In this context, some linear algebra parallel algorithms have been implemented following the parallelization guidelines, and experimentation has shown very good performance. Also, the parallel algorithms outperform the corresponding parallel algorithms implemented on ScaLAPACK (Scalable LAPACK), which is considered to have highly optimized parallel algorithms for distributed memory parallel computers. Also, using more than a single cluster or local area network for parallel linear algebra computing seems to be a natural approach, taking into account the high availability of such computing platforms in academic/research environments. In this context of multiple clusters, there are many interesting challenges, and many of them are still to be exactly defined and/or characterized. Intercluster communication performance characterization seems to be the first factor to be precisely quantified and it is expected that communication performance quantification will give a starting point from which analyze current and future approaches for parallel performance using more than one cluster or local area network for parallel cooperating processing.

1.- Introduction

Cluster computing is already established as a low-cost high-performance way for parallel computing. Computation intensive applications take advantage of the growing processing power of standard desktop computers, along with their low cost and the relatively easy way in which they can be available for parallel processing. Usually, computation intensive areas have been referred to as scientific processing, such as linear algebra applications, where a great effort has been made in order to optimize solution methods for serial as well as parallel computing [1] [2].

In the context of parallel computing hardware, installed local area networks that can be used for parallel processing provide a “hardware zero cost parallel computer”. Hardware installation as well as maintenance cost is “zero”, because LANs are already installed and each computer has its own application programs, user/s, etc., independently of parallel computing. However, parallel computing on these platforms is not “zero cost”. Even if the minimum installation of libraries for developing and running parallel programs -such as implementations of MPI (Message Passing Interface) [3]- are discarded, there are other costs involved, such as applications parallelization and computers availability.

Since some years ago, an algorithmic base for parallel linear algebra computing on clusters has been

1 Director: Armando A. De Giusti

developed [4] [5] [6] [7] [8]. Algorithms for specific linear algebra methods/problems have been implemented following a few guidelines, resulting in simple algorithms with optimized performance on Ethernet-based clusters. Also, heterogeneity has been taken into account on these algorithms for balance computing workload and a low-performance, local area oriented interconnection network is always used. Using more than one cluster could lead to experiment with algorithms (e.g. to verify their quality on a greater number of hardware platforms) as well as define specific guidelines for parallel computing on more than one interconnected cluster. Also, it would be necessary to modify the already defined guidelines if they lead to performance penalties or if they are not useful on this *new* parallel computing platforms.

Given the complexity of heterogeneous hardware for parallel computing as well as an interconnection network with strong contention and/or high latencies make necessary having at least a quantification mechanism or tool to identify bottlenecks and parallel performance penalties. From the interconnection of computers point of view, this leads to aid the parallel programmer to face varying message performance depending on external and unknown factors.

2.- Current Research

Parallelization guidelines on clusters have been defined taking into account clusters characteristics from the point of view of parallel computing and, more specifically, the differences of clusters and traditional parallel computers. The underlying objective is performance optimization; parallelization guidelines are basic but really important since they are specifically defined in the context of the clusters interconnected by Ethernet networks. Summarizing, parallel applications to be run over Ethernet-interconnected clusters should:

- Follow the message-passing programming model. *A priori*, the message-passing library with the highest performance should be used.
- Follow the SPMD (Single Program, Multiple Data) execution model. This simple execution model is favored in the context of linear algebra, where most operations have predictable computing and communication patterns.
- Use (if possible) only broadcast messages. Broadcast messages can directly make use of Ethernet networks (*physical*) broadcast. Also, whenever a single type of messages is used, there are more possibilities of optimization (a single type of message is optimized, instead of every MPI routine, for instance).
- Arrange computing and communication phases in the algorithm so that computers can take advantage of communication overlapped with local (numerical) processing, where facilities are available to do so.
- Have one-dimensional processors interconnection and data distribution, thus making easier the use of broadcast messages as well as the workload balance in clusters with heterogeneous computers. With one-dimensional data distribution, all processors are assumed to be connected to a single bus, such as in the definition of the Ethernet logical bus.

Parallel programming following the message passing model is complex and complexities such as those derived from intercluster communication should be avoided. These interconnections are exposed to potential problems not found on the classical message passing context, with dedicated interconnection networks. Some of problems to be solved in this context are: channel failures, channel recovery, and communication time-outs depending on channel contention. Intercluster communication complexities should be hidden as much as possible to the application programmer.

Given the great number of factors affecting Internet traffic and Internet performance, it is necessary

to define at least a characterization of Internet network performance available for inter-cluster communication. Parallel applications at least should be aware of performance penalties of intercluster communication performance. It is expected to provide a tool and/or a methodology for automatic identification of communication performance profile for intercluster communications.

Computers processing performance and heterogeneity quantification are strongly needed for parallel computing in general and for intercluster parallel computing. Important issues such as parallel performance and computing workload are not possible to quantify without characterization of sequential performance and processing heterogeneity of computers used on clusters. Even when optimized parallel performance algorithms are already available for a single cluster it is expected these algorithms are not optimal for intercluster parallel computing. Thus, it is possible new algorithms and/or specific modifications to the existing algorithms should be defined. This task could be made easier taking into account specific clusters for experimentation, which can be used for identifying specific bottlenecks difficult to estimate and/or derive from other source/s.

3.- Obtained and Expected Results

Three parallel algorithms have been already implemented following the parallelization guidelines explained above: matrix multiplication and LU and QR matrix factorizations. Raw performance of the three algorithms has been evaluated via experimentation on several clusters. Table 1 shows the summary of the best cluster used in terms of processing power of each computer as well as number

CPU	Clock	Memory	Mflop/s	Ethernet Network
Intel P4	2.4 GHz	1 GB	≅ 3000	switched 100 Mb/s

Table 1: Cluster Characteristics.

of computers in the cluster: 20. The computers sequential performance measured as Mflop/s has been obtained by using DGEMM. Table 2 shows the performance of the three algorithms (MM: matrix multiplication, LU: LU matrix factorization, and QR: QR matrix factorization) measured as efficiency in the cluster using different number of computers. The minimum efficiency value is

Computers	MM	LU	QR
2	0.89	0.92	0.95
4	0.86	0.92	0.95
8	0.86	0.93	0.96
16	0.81	0.90	0.94
20	0.80	0.86	0.93

Table 2: Parallel Algorithms Efficiency.

obtained by the parallel matrix multiplication using 20 computing: 0.8, which means that in the worst case, the parallel algorithms obtain 80% of the available peak performance. In fact, for LU and QR factorizations, the obtained performance is over 0.85 in efficiency values, which means that the parallel algorithms obtain more than 85% of the available peak performance. This is more than satisfactory taking into account that current clusters are installed with much better interconnection

networks than Ethernet 100 Mb/s, such as Ethernet 1 and 10 Gb/s, Myrinet, Infiniband, Quadrics, etc. However, ScaLAPACK was used as to compare performance and finally confirm the quality of the previous results. Table 3 summarizes the performance comparison with ScaLAPACK in terms of percentage gain of the proposed parallel algorithms for matrix and the algorithms implemented in ScaLAPACK, which are accepted as highly optimized for distributed memory parallel computers.

Computers	MM	LU	QR
2	+31%	+61%	+6%
4	+35%	+56%	+13%
8	+57%	+92%	+21%
16	+55%	+99%	+28%
20	+53%	+95%	+27%

Table 3: Performance Comparison with ScaLAPACK.

Each column shows the percentage gain of the proposed parallel algorithms compared to the ScaLAPACK ones. The better values correspond, as expected, to LU factorization given that this factorization is highly penalized in ScaLAPACK by the bidimensional matrix distribution and the pivoting needed by numerical stability.

There are many problems to be further approached, some of them are relatively simple and short-term (sub)projects. Most of the ideas explained in the previous section related to intercluster computing are still in an analysis stage and others depend on hardware not available by the time of this writing. Summarizing expected results:

- Parallel algorithms for other computational problems included in LAPACK (Linear Algebra Package) and ScaLAPACK, using the parallelization guidelines explained above.
- Intercluster communication tool-method, hiding details of physical communication to the application programmer.
- Characterization of intercluster communication performance, which is very necessary to characterize parallel performance when the interconnection network is shared with Internet traffic, for example.

References

[1] Anderson E., Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. DuCroz, A. Greenbaum, S. Hammarling, A. McKenney, D. Sorensen, LAPACK: A Portable Linear Algebra Library for High-Performance Computers, Proceedings of Supercomputing '90, pages 1-10, IEEE Press, 1990.

[2] Bilmes J., K. Asanovic, C. Chin, J. Demmel, “Optimizing matrix multiply using phipac: a portable, high-performance, ansi c coding methodology”, Proc. Int. Conf. on Supercomputing, Vienna, Austria, July 1997, ACM SIGARC.

[3] Message Passing Interface Forum, MPI: A Message Passing Interface standard, International Journal of Supercomputer Applications, Volume 8 (3/4), 1994.

[4] Fernando G. Tinetti, Walter J. Aróztegui, Antonio A. Quijano, “Solución de Sistemas de Ecuaciones Ralas en Clusters de Computadoras”, X Congreso Argentino de Ciencias de la

Computación (CACIC 2004), Universidad Nacional de La Matanza, La Matanza, Argentina, 4 al 8 de Octubre de 2004.

[5] Fernando G. Tinetti, Mónica Denham, Andrés Barbieri, "Algebra Lineal en Clusters Basados en Redes Ethernet", V Workshop de Investigadores en Ciencias de la Computación (WICC 2003), Tandil, Argentina, 22 y 23 de Mayo de 2003, pp. 575-579.

[6] Fernando G. Tinetti, Mónica Denham, "Algebra Lineal en Paralelo: Factorizaciones en Clusters Heterogéneos", X Congreso Argentino de Ciencias de la Computación (CACIC 2004), Universidad Nacional de La Matanza, La Matanza, Argentina, 4 al 8 de Octubre de 2004.

[7] Fernando G. Tinetti, Armando De Giusti, "Parallel Linear Algebra on Clusters", 3rd International workshop on Parallel Matrix Algorithms and Applications (PMAA'04), 20-22 October, CIRM, Marseille, France, 2004.

[8] Fernando G. Tinetti, Antonio A. Quijano, "Costos del Cómputo Paralelo en Clusters Heterogéneos", V Workshop de Investigadores en Ciencias de la Computación (WICC 2003), Red de Universidades Nacionales, Tandil, Argentina, 22 y 23 de Mayo de 2003, pp. 580-584.