A peer review of the parallel and distributed toolboxes of matrix programming languages for mathematical programming

Glavelis Themistoklis
University of Macedonia
Egnatia Street 156, Thessaloniki, 54006
mai086@uom.gr

Ploskas Nikolaos
University of Macedonia
Egnatia Street 156, Thessaloniki, 54006
mai084@uom.gr

Samaras Nikolaos
University of Macedonia
Egnatia Street 156, Thessaloniki, 54006
samaras@uom.gr

Abstract

Matrix programming languages are widely used by scientists due to theirs' specific orientation towards the construction and manipulation of mathematical and statistical models. Mathematical programming is closely connected with these software packages. Although the capabilities that these programming languages offer for mathematical programming, there are sub cases where their effectiveness is not satisfactory with regard to the computational time. These problems are solved with the use of parallel programming. It is well-known that parallelization can lead to important reductions in computational time. More recently, clusters of workstation computers have become widely used. Nowadays, this solution is not very expensive in terms of hardware costs.

In this paper we examine the parallel toolboxes that matrix programming languages provide. There are several advantages and opportunities that scientific community can gain from the use of these libraries and toolboxes. However, these toolboxes have some limitations and cannot be used to solve any kind of problem. We review the main features of the most recent versions of the parallel toolboxes of the more common mathematical software including MATHEMATICA, MATLAB and Ox. Despite their strong similarities there are substantial differences between the parallel toolboxes of these matrix languages.

Index terms: Matrix programming languages, Parallel and distributed programming, MATHEMATICA, MATLAB, OX.

1. Introduction

The requirement for faster and more effective computations in scientific applications has been increased considerably in the last years. Natural restrictions and high costs render impossible the increase of speed of processors beyond concrete limits. In order to overcome these difficulties new architectures have been developed importing the parallel processing [13], [14], [15], [21]. In our days different types of machines with high capabilities are available and supported in the concentration of many processors together.

Networked computers have become a common infrastructure in most organisations, especially large ones. Additionally, the speed of the network has also improved significantly. Nowadays, it is very common for local area networks to carry various kinds of data such as voice and video. High performance computing has also benefitted and today, distributed computing is quickly gaining popularity [19].
Parallel programming is a good practice for solving computationally intensive problems in various fields. In operations research, for instance, solving maximization problems with simplex method is an area where parallel algorithms are being developed [16], [17], [18], [20]. A popular approach to implement parallel algorithms is to configure a cluster or a network of personal computers. With the advances made in computer hardware and software, it is now quite a simple matter to configure a computer network.
The primary reasons for using parallel computing are: i) reduction of execution time, ii) solution of larger problems, iii) provide concurrency, iv) taking advantage of non-local resources - using available compute resources on a wide area network and finally v) overcoming memory constraints.
The parallelism leads to the need for new algorithms, because many times the most optimal parallel solutions are not performed with simple modifications of serial programs. In addition to the serial programs, the parallel algorithms depend dynamically on the architecture of system for which they have been created. The available support for communication and timing between the parts of a parallel program are certain from the important scripts of parallel programming.
The easiness and the time for the development of a program are many times exceptionally critical. The tools can help the programmer to develop and improve parallel programs. One of the difficulties faced by grid and distributed computing users is the user-friendliness of the software and middleware used to run a job across the network. This is compounded by the fact that not all computers in the network are exactly the same, even though they may be running the same operating system. As a result, many software developers are beginning to realise the need to make parallel computing more user-friendly.
MATHEMATICA [1], MATLAB [2] and Ox [3] have attempted to solve this problem by offering an environment for creating parallel algorithms. MATHEMATICA, MATLAB and Ox are popular mathematical software used by many students and academic staff. Using these programming environments to implement parallel algorithms, users will take advantage both from the wide range of functions that these software offer and from the advantages of the parallel processing.
An outline of the rest of the paper is as follows. In section 2, general description of the systems is given, and in section 3, we present the installation requirements in order to use the parallel toolboxes of the three mathematical packages. In section 4, we present the capabilities of the parallel toolboxes and in section 5 the structure of parallel programming of each mathematical environment. In section 6, there is an analysis for the optimization role in parallel toolboxes. Finally, in section 7 we give our conclusions.

2. General description of matrix programming languages

MATHEMATICA is a specialized computer program used mainly in scientific and mathematical fields, developed and distributed by Wolfram Research Europe Ltd. MATHEMATICA covers mainly subjects of mathematics, including number theory, linear algebra and mathematical analysis by dealing with them from both the symbolic and the numerical points of view. Apart from that, there is a large range of libraries and toolboxes specialized in different sectors. Consequently, this is the reason for being wide spread among the scientific community. Furthermore, it is recognized as one of the world's most

powerful mathematical software systems. Moreover, MATHEMATICA consists of two applications, the "kernel" which is responsible for all the computations and the "front end" which is the user interface. MATHEMATICA is available for Windows Vista/XP (32-bit, 64-bit), Apple Macintosh Mac OS X 10.4/10.5 (32-bit, 64-bit), Linux 2.4 or later (32-bit, 64-bit), Sun Solaris (64-bit), HP-UX 11.11 (64-bit), IBM POWER AIX 5.1, 5.2, 5.3 (64-bit).

MATLAB is a matrix language developed and distributed by The MathWorks Inc. MATLAB is wide spread due to its capability of easy programming tasks like plotting, implementation of algorithms and interacting with programs which have been developed in other languages (like Fortran and C/C++). MATLAB (MATrix LABoratory) as the name suggests, is especially designed for matrix computations like, solving systems of linear equations or factoring matrices. Its utilities and functions are organized in various toolboxes. These toolboxes are comprehensive collections of MATLAB functions (M-files) that extend the MATLAB environment in order to solve particular classes of problems. In addition, some toolboxes are designed to connect MATLAB to other software tools. For example, MATLAB itself handles only numerical computations, but the Symbolic Math toolbox adds symbolic computation capabilities through an interface with the Maple engine. As of 2004, MathWorks claimed that MATLAB was used by more than one million people in industry and scientific community. Furthermore, it is remarkable the fact that MATLAB has been widely adopted in the academic community. More than 3,500 universities around the world use MATLAB for teaching and research in a broad range of technical disciplines. MATLAB is available for Windows Vista/XP (32-bit, 64-bit), Apple Macintosh Mac OS X 10.4/10.5 (32-bit), Linux 2.4 or later (32-bit, 64-bit), Sun Solaris (64-bit).

Ox has been developed by Jurgen Doornik and is distributed by International Thomson Press. Ox is a matrix programming language with built-in capabilities of using the object-oriented approach. This capability is its main advantage comparing to the other two software packages. Apart from that, Ox is relatively younger language in comparison with MATHEMATICA and MATLAB. Ox was created with the help of C language and this is the reason for the significant similarities between their languages. Moreover, Ox includes a comprehensive range of commands for quick matrix operations. Its wide spread among the statisticians and econometricians implies its large and useful range of statistical and econometrical libraries. Among the special features of Ox, are its speed, the ease of programming, the well designed syntax and editor, and some graphical facilities. Most of these advantages are due to its origin of C language. OxMetrics is an econometrical software including the Ox programming language. Ox is available for Windows Vista/XP (32-bit, 64-bit), Apple Macintosh Mac OS X 10.4/10.5 (32-bit), Linux 2.4 or later (32-bit,64-bit), Sun Solaris (64-bit), HP-UX 11.11 (64-bit), IBM POWER AIX 5.1, 5.2, 5.3 (64-bit).

3. Installation Requirements for using the parallel toolboxes

3.1 MATHEMATICA

In order to perform computations in parallel with the help of MATHEMATICA, the only thing which is needed is the 'Parallel Computing Toolkit'. This toolkit is available under these operating systems: Linux, Macintosh and Microsoft Windows. With this toolbox users are able to: i) start processes and connect them in order to interact, ii) organize processes and computations on available processors and iii) share data between processes and synchronize access to common resources.

First of all, in order to be able to take full advantage of the opportunities that Parallel Computing Toolkit offers, users should have access to a number of remote computers capable of running MATHEMATICA or use of a multiprocessor local machine. Apart from that, a suitable network connection between a local computer and the remote machines are necessary. Although, a network is set up, there may be some security restrictions which limit the ability to execute MATHEMATICA programs on remote computers in parallel, like firewall restrictions.
Furthermore, the remote computer must run a rsh (for Windows platforms) or ssh daemon (for Linux and Macintosh platforms) or other remote login service in order to start MATHEMATICA on the remote computer through a local computer. On the other hand, if there is not a rsh or ssh daemon on the remote computer, the connection can be achieved with the help of a TCP/IP network. Before that, the users must start manually the kernels on the remote computers and then connect them with the local computer.

3.2 MATLAB

Distributed computing with MATLAB on a cluster requires two products: i) Parallel Computing Toolbox and ii) MATLAB Distributed Computing Server. Parallel Computing Toolbox (formerly Distributed Computing Toolbox) should be installed on the computer where the applications are written. MATLAB Distributed Computing Server (formerly MATLAB Distributed Computing Engine) should be installed on each computer of the cluster that performs the computation. These packages are available for the above operating systems: Linux, Macintosh and Microsoft Windows.
In all three platforms, to set up a cluster for use with Parallel Computing Toolbox and MATLAB Distributed Computing Server are required four stages. The first stage involves the installation of the MATLAB Distributed Computing Server. This installation can be accomplished with two patterns: either i) installation in a shared directory and then map the installation directory to all nodes (this option increases network overhead) or ii) installation individually on each node (this option involves no additional network overhead). The second stage concerns the configuration of the MATLAB Distributed Computing Server for use with the job manager. MATLAB Distributed Computing Server requires a scheduler to queue and manage jobs submitted to the cluster. MATLAB offers the possibility either to use the Job Manager (the scheduler included with MATLAB Distributed Computing Server), or integrate with third-party schedulers, such as Platform LSF [4], Microsoft Windows Compute Cluster Server [5], PBS [6], TORQUE [7], Sun Grid Engine [8], Grid MP [9], etc. In the third stage takes place the installation of the Parallel Computing Toolbox and finally in the last stage is verified that the parallel computing products are installed and configured correctly.
Moreover, MATLAB offers the possibility to setup these parallel products on a single machine without a cluster. A local scheduler is included with Parallel Computing Toolbox that allows programmers to schedule jobs and run up to four workers or labs on a single MATLAB client machine. This local scheduler and its workers do not require a job manager, a third-party scheduler, or MATLAB Distributed Computing Server.

3.3 Ox

OxMPI is an Ox package enabling the development of distributed Ox programs. The resulting Ox program can run on all processors of a multicore workstation, or on a cluster of machines. The installation requirements to develop and run distributed Ox programs are: i) a working MPI installation and ii) Ox Console 4 or 5, or Ox Professional 4 or 5.

OxMPI has been tested by the creator of Ox (Jurgen Doornik) both in Windows and Linux. OxMPI for Windows has been tested with DeinoMPI 1.1.0 [10] using 32-bit Windows XP and Vista, as well as 64-bit XP and also with MPICH2 1.0.6p1 [11]. OxMPI for Linux has been tested with OpenMPI [12] under Fedora 7.

4. Parallel toolboxes' strengths and weaknesses

4.1 MATHEMATICA

The parallel programming with the help of MATHEMATICA is not a complicated procedure. MATHEMATICA have many built-in functions which make the lives of programmers much easier. The first step is to load the 'Parallel Computing Toolkit' and then start a number of remote kernels. Starting and connecting a number of remote machines is not a very easy job. Programmers must insert some special commands, a task which may be difficult for beginner users.
Apart from the stage of cluster's configuration, parallel programming with MATHEMATICA can be an ideal choice for beginners and experienced programmers too. There are commands which can send data and codes from the local machine to remote systems without the programmers to concern about details. On the other hand, there are commands which permit to users to decide how they want to separate their codes and data among the remote machines, a utility which refers to advanced parallel programming. It is well known that all the programs and line of codes can not be parallelized. In this case, MATHEMATICA does not face any problem and propose a very useful utility. The particular block of code which can not be parallelized, is executed locally at the master kernel and then the results can be distributed to remote kernels and continue their jobs.
It is well known that there are two ways of handling memory, the shared memory and distributed memory. In a shared-memory machine, all processors have access to a common main memory. Moreover, a processor can simply write a value into a particular memory location, and all other processors can read this value. In a distributed-memory machine, each processor has its own main memory, and the processors are connected through a network to exchange values of variables. This involves explicit communication over the network. MATHEMATICA through 'Parallel Computing Toolkit' uses independent remote kernels as parallel processors. It is obvious that these machines do not share a common memory, even if they happen to be on the same machine. In contrast, the package Parallel `VirtualShared`, which is part of the 'Parallel Computing Toolkit', implements virtual shared memory for these processors.

4.2 MATLAB

The Parallel Computing Toolkit is fully integrated in MATLAB by Mathworks itself. Its aim is to distribute jobs in many computer machines without the need of user's configuration. This is the main advantage of this toolbox from others that have been implemented to work with MATLAB. It is fairly easy to use this toolbox, although the user still has to take care of cutting the data up in pieces. User can convert its serial MATLAB programs to parallel MATLAB programs without making significant changes to existing code or learning a low-level parallel language. Furthermore, Parallel Computing Toolbox offers the possibility to execute programs on a single multicore or multiprocessor workstation. Finally, both interactive and batch execution modes are supported from this toolbox.

On the other hand, Parallel Computing Toolbox has also some disadvantages. First of all, this toolbox comes at a cost per node. Furthermore, workers have to be started on all computer nodes and if there is a crash, they have to be restarted manually. One other significant disadvantage is that every file used in the program should be identified and stored in a fileDependencies array. These files should be accessible from every node. If there is a shared filesystem, this is not a problem. But if there isn't a shared filesystem, users have to pass these files to every node and it should be stored in the same location in every node. Obviously, this option increases network overhead.

4.3 Ox

OxMPI has adopted the master/slave model written in MPI: the same program is running on each node, with if statements selecting the appropriate code section. This means that user have to learn first the Message Passing Interface language and then convert its' serial programs to parallel. The aim of this utility is to take a popular matrix programming language and implement a message-passing interface using MPI. OxMPI isn't a full implementation of MPI for Ox. Instead it has what is sufficient to run a program, and then extend this for parallel computations. Finally, OxMPI lucks the user-friendliness that the parallel toolboxes of MATLAB and MATHEMATICA offers.

5. Structure of parallel programming in mathematical software

5.1 MATHEMATICA

First of all, debugging is not an easy utility in parallel programming. In this sector MATHEMATICA provides the programmers with the opportunity of monitoring remote machines. Through the 'Debug view' that shows which of the remote kernels are running, users can find easily and very fast the origin of many problems which may be produced during the execution of a parallel program. The 'Debug window' can present some useful information when it combines with commands like ParallelEvaluate[] which separate the initial calculation to smaller calculations and each of them is computed in different remote kernels. With 'Debug window' users can see which remote machine can crash and begin the calculation from the beginning again.

Apart from that, MATHEMATICA provides the programmers with the ability to control their programs step by step with the help of the breakpoints. With this technique, programmers have the ability to see which part of their code is causing problems and take measures in order to avoid them. Furthermore, it is possible to have many breakpoints in more than one remote machine. In this case, programmers are able to monitor each machine separately without any problem.

It is well known that all the high level programming languages except for the opportunities which offer through debugging for enhancement in codes, includes the profilers. In this direction, MATHEMATICA includes functions for parallel profiling. As well as in sequential programming, users have the opportunity to run a profile report on the command line and they will have as a response a report for each kernel in the network. In this report there is a reference to the number of calls for a particular computation and the average time of the execution of this calculation for each remote machine.

5.2 MATLAB

First of all, parallel programs in MATLAB can be executed under interactive and batch execution modes. In interactive mode, Parallel Computing Toolbox extends the MATLAB interactive environment. The Parallel Command Window (interactive mode) is familiar with MATLAB environment and that makes easier the developing of task- and data-parallel applications. This environment helps parfor - loops (parallel for) detect the presence of workers and transfer the necessary workspace data between the MATLAB session and the workers. The Parallel Command Window sets up a data-parallel execution environment in which can be used distributed arrays and message passing functions. Commands issued at the parallel prompt are executed simultaneously on all computer nodes.
Parallel Computing Toolbox offers also a batch environment that provides an offline mode of execution. Running in batch mode the MATLAB client session can also be shut down or used for other activities while large MATLAB applications are executed in worker nodes. This utility frees the MATLAB client session for other activities and the results can be retrieved later.
A profiling utility enables users to observe performance of parallel applications that use distributed arrays and message passing functions. The profiler is collecting information about the execution of code on each lab and the communication between the labs. Such information includes: i) execution time of each function on each lab, ii) execution time of each line of code in each function, iii) amount of data transferred between each lab and iv) amount of time each lab spends waiting for communication. Moreover, using the profiler one can extract a report at the end of program's execution. This interactive report provides feedback for almost every task like the plotting of the communication scheme between labs and the comparison controls to display information for several labs simultaneously.

5.3 Ox

OxMPI offers all the well known MPI functions like send, receive, broadcast etc. The only difference is that the OxMPI functions have less input arguments than the classic MPI functions. For a beginner this is better, because it is more difficult to remember a function with a lot of arguments and also it is even more complicated to understand what the scope of these arguments is. Finally, it is possible to call external C, C++ and Fortran functions from Ox code by writing a small C wrapper around the external function.

6. Parallel Optimization in MATLAB

In the latest version of MATLAB, MathWorks has distributed the integration of the Parallel Computing Toolbox with the Optimization Toolbox. This attempt was made in order to offer a strong combination of functions for linear programming, nonlinear optimization and multi-objective optimization at the same time of taking full advantage of benefits from parallel programming.

The Optimization Toolbox is a library of functions that extends the capability of the MATLAB computing environment. The toolbox includes functions for many types of optimization including: linear programming, quadratic programming, non-linear optimization, non-linear least squares, multi-objective optimization and binary integer programming. Functions of Optimization Toolbox, like the optimization solvers, are able to take advantage of the opportunities that are available through the parallel computing toolbox. Moreover, users can benefit from the ability that MATLAB provides a friendly

environment for parallel programming. In addition, programmers are able to execute optimization problems on multicore computers or computer clusters.

The optimization toolbox of MATLAB can be used in order to enhance the performance of program codes at any field of scientific community. One of the most significant advantages is the simultaneous execution of many problems. In this case, users have the opportunity to solve more intensive problems and much faster than in a single core. Obviously, a significant reduction in the execution time is presented due to the parallel computing.
Customizable support for parallel computing involves explicitly defining the optimization problem to use parallel computing functionality. User can define either the objective function or constraint function to use parallel computing, so to decrease the time required to evaluate the objective/constraint. Finally, Built-in support for parallel computing allows users to accelerate the gradient estimation step in select solvers for constrained nonlinear optimization problems, multi-objective goal attainment and minimax problems.

7. Conclusions

Matrix programming languages are widely used by scientists due to theirs' specific orientation towards the construction and manipulation of mathematical and statistical models. On the other hand, there are cases where their effectiveness is not satisfactory with regard to the computational time. Such problems are solved with the use of parallel programming.
In this paper we examined the parallel toolboxes that matrix programming languages provide. There are opportunities that scientific community can gain from the use of these libraries and toolboxes. However, these toolboxes have some limitations and cannot be used to solve any kind of problem. Despite their strong similarities there are substantial differences between the parallel toolboxes of these matrix languages.
MATLAB's Parallel Computing Toolbox seems to gain popularity among the other available from other software packages, due to its ease of programming and its useful profiler. Moreover, a significant advantage of MATLAB's parallel environment is that it can be combined with the Optimization Toolbox to solve large problems. MATHEMATICA's parallel environment is a user-friendly toolbox for creating parallel applications. Its advantages are the ease of programming and its powerful parallel debugger. A drawback for MATHEMATICA's parallel toolkit is its difficult configuration for not very experienced users. The developer of Ox has implemented an Ox package to enable the development of distributed Ox programs. Ox has a very useful library of statistical and econometrical functions that can be combined with the parallel processing and lead to the reduction of time. The disadvantage for OxMPI is that is just a combination of the Message Passing Interface and the Ox environment. User should first learn MPI and then use OxMPI.
In conclusion, there are several opportunities that we can gain from the combination of mathematical software with parallel processing. Even though these parallel environments have to be improved, they can lead to a reduction of time for computationally intensive problems.

References

[1] (2008) The Wolfram Research website. [Online]. Available at: http://www.wolfram.com/
[2] (2008) The MathWorks website. [Online]. Available at: http://www.mathworks.com/

[3] (2008) The Jurgen Doornik website. [Online]. Available at: http://www.doornik.com/
[4] (2008) The Platform Computing Corporation website. [Online]. Available at: http://www.platform.com/Products/Platform.LSF.Family/
[5] (2008) The Microsoft Corporation website. [Online]. Available at: http://www.microsoft.com/hpc/default.aspx
[6] (2008) The Altair Engineering website. [Online]. Available at: http://www.altair.com/Default.aspx
[7] (2008) The Cluster Resources website. [Online]. Available at: http://www.clusterresources.com/
[8] (2007) The Sun Microsystems website. [Online]. Available at: http://gridengine.sunsource.net/
[9] (2008) The Univa UD website. [Online]. Available at: http://www.univaud.com/products/grid-mp/
[10] (2008) The DeinoMPI website. [Online]. Available at: http://mpi.deino.net/
[11] (2008) The Research Laboratory Argonne website. [Online]. Available at: http://www.mcs.anl.gov/research/projects/mpich2/
[12] (2008) The Open MPI project website. [Online]. Available at: http://www.open-mpi.org/
[13] Baker M., Buyya R. and Laforenza D. (2002), "Grids and Grid technologies for wide-area distributed computing", Software-Practice and Experience, 32, 1437-1466.
[14] Cunha J.C., Rana O.F. and Medeiros P.D. (2005), "Future trends in distributed applications and problem-solving environments", Future Generation Computer Systems, 21, 843-855.
[15] Foster I., Kesselman C. and Tuecke S. (2001), "The Anatomy of the Grid: Enabling Scalable Virtual Organizations", The International Journal of High Performance Computing Applications, 15(3), 200-222.
[16] Karypis G. and Kumar V. (1994), "Performance and Scalability of the Parallel Simplex Method for Dense Linear Programming Problems an Extended Abstract", Technical Report, Computer Science Department, University of Minessota.
[17] Kilgore A. (1993), "Vary Large-scale Linear Programming: A Case Study Exploiting Both Parallelism and Distributed Memory", MSc Thesis, Center for Research on Parallel Computation, Rice University.
[18] Maros I. and Mitra G. (2000), "Investigating the sparse simplex algorithm on a distributed memory multiprocessor", Parallel Computing, 26, 151-170.
[19] Noor A.K. (1997), "New Computing Systems and Future High-Performance Computing Environment and their Impact on Structural Analysis and Design", Computers and Structures, 64, 1-30.
[20] Shu W. and Wu M. (1993), "Sparse Implementation of Revised Simplex Algorithms on Parallel Computers", Sixth SIAM Conference on Parallel Processing for Scientific Computing, March 22-24, 1993, Norfolk
[21] Skillicorn B. and Talia D. (1998), "Models and Languages for Parallel Computation", ACM Computing Surveys, 30(2), 123-169.