Jornada de Seguimiento de Proyectos, 2004
Programa Nacional de Tecnologías Informáticas

PELICAN: Parallelization and Extension of the GNU Scientific Library TIC2002-04400-C03

Enrique S. Quintana-Ortí[*]
Depto. de Ingeniería y Ciencia de Computadores
Universidad Jaume I
12.071–Castellón

Francisco Almeida Rodríguez[†]
Depto. de Estadística, Investigación Operativa y Computación
Universidad de La Laguna
38.071–La Laguna

Abstract

As a natural challenge and evolution of our research in the area of parallel and distributed computing, and as a service to the scientific community, we intend to extend and parallelize the free–distribution "GNU Scientific Library" (GSL).

We will study five classes of target architectures: Sequential, Shared Memory, Distributed Memory, Hybrid Systems combination of the latter two, and the GRID. We also aim to satisfy the requirements of two classes of users: one with skills in sequential C programming, and another with a certain knowledge in parallel programming (in particular, one of the two standard parallel application programming interfaces, OpenMP and MPI).

Furthermore, the proposed library extensions cover three key topics which are highly interesting from the point of view of engineers: Discrete Mathematics, Optimization, and Matrix Algebra.

The achievement of these goals implies the search and validation of solutions for a formidable set of challenges, including: how to exploit nested parallelism, how to accomplish efficient parallelization of irregular problems, and how to find efficient and simple schemes for the use of hybrid and GRID systems. One of the strengths of this project is to provide efficient solutions to these challenges.

As a basic benefit from the project, we will develop a parallel scientific library which will be used by researchers from several national and international research groups who collaborate with us.

Keywords: GNU scientific library, discrete mathematics, optimization, matrix algebra, parallel algorithms and architectures.

[*]Email: `quintana@icc.uji.es`
[†]Email: `falmeida@ull.es`

1 Introduction

The GNU Scientific Library [1] is composed of hundreds of numerical routines, written in C from scratch, as part of a project that was started by M. Galassi and J. Theiler at *Los Alamos National Laboratory.* GSL employs an object-oriented methodology and includes mathematical functions for complex numbers, complex roots, matrices and arrays, linear algebra, polynomials, and Fourier transforms, to name only but a few. The library is nowadays used by thousands of scientists and engineers.

For many researchers, the main reason for using a numerical library is the increasing need towards international cooperation. Here, the use of non-public libraries becomes an important obstacle because of their economic cost and the limitations these libraries impose on the spreadability of results of a project. Both these types of problems can be avoided by employing instead a free-distribution library such as GSL.

There is currently no parallel version of GSL, though much of its functionality is partially covered by other (in many cases non-public) libraries. This can only be justified by the lack of standards for parallelization when the GSL project started. However, we believe that with the introduction of OpenMP [2] and MPI [3], the situation has changed substantially. OpenMP is now considered the standard *de facto* for exploiting loop-level parallelism on shared-memory architectures, while MPI plays a similar role for message-passing architectures.

2 Objectives

The main goal for this project is the extension and parallelization of GSL. The extension is aimed to complete certain basic-functionality routines that are not available in GSL. The parallelization targets a wide range of the parallel architectures that engineers and scientists have usually access to such as shared-memory multiprocessors (SMM), distributed-memory multiprocessors (multicomputers), clusters, and hybrid systems.

A second goal for this project is to hide the intricacies of using parallel codes and architectures from a certain type of our "customers", who have little experience with parallel computing or none at all, providing those with the image (interface) of a traditional serial machine, but with the benefits of a parallel execution.

These general goals can be further decomposed into the following three specific objectives:

- **Extension of GSL.** We detect some important functionality missing in GSL in the areas of minimization and sparse matrix algebra that should be fixed.
- **Analysis, design, and implementation of efficient parallel versions of GSL for different parallel architectures.** In particular, we intend to study the parallelization of sparse matrix algebra operations and also discrete algorithms arising in operations research, operations on strings, and geometry.
- **Design and development of a middleware that delivers a serial image of the parallel system.**

In order to fulfill these objectives, the coordinated project (PELICAN) was divided into two[1] subprojects: PELICANUM was to deal with the numerical part of the project, including linear algebra algorithms for sparse computations, and was to be developed by the *Grupo de Computación Científica y Paralela* (GCCP) from the University Jaume I (UJI) at Castellón. On the other hand, PELICANTO would include those algorithms arising in discrete mathematics such as graphs, computational geometry, and string processing. This second subproject would be carried out by the *Parallel Computing Group of the University of La Laguna* (PCGULL).

2.1 Workplan

The project is being developed in three different stages. The first stage corresponds to the first specific objective stated above (that is, the extension of the functionality of the serial GSL) and includes the analysis of the parallelization of these algorithms as well. In the second stage, we planed to design and implement several parallel versions of GSL for SMM and multicomputers using, respectively, OpenMP and MPI. In this stage, we also had to develop the corresponding serial interfaces for the "sequential" users of the message-passing versions. In the third stage, we intend to investigate the parallelization on hybrid systems: multicomputers consisting of the nodes which are SMM and heterogeneous systems. Here, we also want to study the design and use of interfaces for web and GRID computing.

In order to execute the project, the following tasks were identified:

Tasks 1. Extension of GSL and analysis of the parallelization.

Tasks 2. Parallelization of numerical algorithms.

Tasks 3. Parallelization of discrete algorithms.

Tasks 4. Analysis of hybrid architectures.

Tasks 5. Coordination and spread of results.

Figure 1 shows the schedule of the tasks of the project. It is divided into three different tables, one per year. Each row corresponds to a different task, and each column designates a month of that year. Tasks 5 includes the coordination activities and is therefore active during the three years of the project.

[1]The original proposal included a third subproject which had to study the parallelization of minimization algorithms. That subproject was not approved, though.

			First year											
Task	University	People	1	2	3	4	5	6	7	8	9	10	11	12
1	UJI, ULL	All	1	1	1	1	1	1	1	1	1	1	1	1
5	UJI, ULL	All	5	5	5	5	5	5	5	5	5	5	5	5

			Second year											
Task	University	People	1	2	3	4	5	6	7	8	9	10	11	12
1	UJI, ULL	All	1	1	1	1	1	1	1	1	1	1	1	1
2	UJI	GCCP	2	2	2	2	2	2	2	2	2	2	2	2
3	ULL	PCGULL	3	3	3	3	3	3	3	3	3	3	3	3
5	UJI, ULL	All	5	5	5	5	5	5	5	5	5	5	5	5

			Third year											
Task	University	People	1	2	3	4	5	6	7	8	9	10	11	12
1	UJI, ULL	All	1	1	1	1	1							
2	UJI	GCCP	2	2	2	2	2							
3	ULL	PCGULL	3	3	3	3	3							
4	UJI, ULL	All						4	4	4	4	4	4	4
5	UJI, ULL	All	5	5	5	5	5	5	5	5	5	5	5	5

Figure 1: Schedule of the tasks of the project.

3 Current status of the project

3.1 Subproject PELICANUM

The tasks that were completed during the first year of this subproject included the extension of GSL with a module for sparse linear algebra operations. One of the difficulties that was encountered here, and possibly the reason this module was not already available in GSL, was the lack of a well-defined standard interface and functionality for sparse linear algebra operations that was globally accepted by the scientific community. This is due to the impossibility of defining an optimal storage scheme for sparse matrices, as the performance of using a certain scheme strongly depends on the sparsity pattern of the matrix, which is completely dependent on the application the data arises in. Furthermore, different applications surely deliver widely different sparsity patterns.

This problem was recently solved (in part) by the *BLAS Technical Forum* (BLAS-TF), which declared the interface and functionality for such routines, while defining the actual storage scheme as an implementation issue. This is possible by using *handlers* to refer to the sparse data structures.

Starting from the sparse BLAS-TF standard, we developed a complete set of routines for sparse linear algebra that is structured into three layers. (Here, the routines in layer N offer services to the ones in layer N+1 by means of the functionality of those routines in layer N-1.) Specifically, the contents of each layer, from bottom to top, are the following:

- **Layer 1. Basic routines for sparse linear algebra.** This layer includes routines to deal with the specific implementation of storage schemes for sparse matrices and cover all the computational functionality defined in the sparse BLAS-TF standard for the Level 1, 2, 3 BLAS. In particular, three specific storage schemes for sparse matrices are adopted in our library: coordinate, Harwell-Boeing (or compressed row/column) format, and sparse blocked format. The user is then given the possibility of choosing among any of those formats during the installation of the library.
- **Layer 2.** This level is composed of *wrappers* to the routines in the underlying level so that the user is presented with the interface defined in the computational part of the sparse BLAS-TF standard. Several other routines are also included in this layer in order to manage sparse objects (creation, destruction, tuning, definition of specific properties, etc.), as defined in the sparse BLAS-TF standard.
- **Layer 3.** This layer only includes wrappers to the underlying routines so that the sparse BLAS-TF standard is "adapted" to the philosophy driving GSL. For example, the names of the routines and data structures for GSL and BLAS-TF follows two completely different approaches. Also, the goal was modify the functionality and interfaces of the sparse BLAS-TF standard in order to mimic the functionality that GSL offers for (dense) vectors and matrices.

All the functions were developed using ANSI C, following basic style requirements in GSL. We only employed performance optimization techniques that were independent of the underlying computer architecture. Therefore, the codes can be migrated to any architecture provided the appropriate ANSI C compiler is available.

In this second year we have analyzed, designed, and implemented the parallel versions for SMM/OpenMP and multicomputers/MPI of the routines for sparse linear algebra developed during the previous year. Our parallelization efforts follow closely the three-layered structure defined above.

It is worth pointing out here that our parallel codes for multicomputers are specially designed to consider the existence of nodes with different capabilities in the parallel system (heterogeneity). The idea here is to distribute the workload unevenly among the nodes, so that faster processors perform a larger amount of the computations. With the appropriate distribution of the workload, the performance is thus not driven by the performance of the slowest node in the system, but by its average performance. An homogeneous multicomputer can then be considered as a special case of an heterogeneous one, and only a minor decrease of the performance was observed compared with that obtained with an *ad hoc* programming of the homogeneous case.

3.2 Subproject PELICANTO

The tasks that were completed during the first year range from the analysis of the serial GSL discrete and optimization algorithms, to the analysis and development of their parallel versions.

The analysis of the serial algorithms in the GSL library showed the lack of exact algorithms to solve optimization problems. Therefore, our initial plan was to extend the sequential library by developing exact algorithms from scratch. However, a general inconvenient for the parallelization of GSL is that, being a sequential library, the serial routines were not designed with parallelism in mind. The parallel design is therefore conditioned by this fact. For that reason, we postponed the development of sequential routines for optimization problems until our parallel design was defined.

Several difficulties have been found when parallelizing discrete algorithms. Most of these algorithms share a lack of structure and also present irregular loop patterns and complex data dependencies. During the analysis of the parallelization for distributed-memory platforms we studied two different alternatives: the *master-slave* paradigm and a *Simple Program Multiple Data* (SPMD) programming strategy. After the analysis of two approaches, we decided to adopt the SPMD strategy due to its simplicity. However, no scheme arises as an optimal solution in general, and every algorithm should be analyzed independently. In order to develop our codes, we followed an incremental method based in prototypes. Thus, with each new prototype (release) of the parallel library, we enlarge the range of functionalities and types of parallel platforms supported while maintaining the sequential interface.

During the first year we focused on the parallelization on homogeneous distributed-memory architectures using MPI. Two versions were developed, replicated and distributed. In the replicated paradigm the data structures are copied on every processor and these are manipulated there independently from other processors. At the end of the parallel routine, data are again replicated. The distributed approach scatters the data structure among the set of processors and this distribution remains so after the routine ends. The GSL sequential data structures have been extended to support the replicated and distributed paradigm.

Two typical cases of study were considered during this first year: sorting operations and the Fast Fourier Transform algorithms. Following the prototypes developed, the generation of new parallel routines can be considered in many cases an application of software reusability. Many existing parallel routines can be adapted without too much effort to use our interfaces while, in some other cases, the parallelization required a larger code re-elaboration.

Particular consideration deserves the management of data types in the parallel routines. Following the mechanism developed in the sequential GSL library, once a routine is parallelized, a family of parallel routines is provided, one for each of the C simple data types and also for generic data types.

During the second year our parallelization targeted heterogeneous distributed-memory architectures and shared-memory architectures. As a consequence of this analysis, the GSL data structure has been extended to allow a block-cyclic mapping on heterogeneous clusters, where the processors may have different processing capabilities. In particular, in an heterogeneous block-cyclic mapping data are distributed according to the processors performances. Prototypes have been successfully implemented on heterogeneous platforms.

The analysis on shared-memory architectures reveals that, for applications with regular loops, a parallelization using OpenMP directives becomes a natural and efficient choice; however, if the application presents an irregular structure, a deeper analysis is needed to adapt existing parallel routines. Several strategies may be available and no *a priori* knowledge will reveal the applicability of the strategies. Two classical examples can be mentioned here: the use of an appropriate scheduling scheme, or the use of the run time OpenMP library. We have developed a methodology to analyze both cases in the form of an OpenMP Source Code Repository (OmpSCR `http://www.pcg.ull.es/ompscr`), an open, non-proprietary and freely accessible infrastructure oriented to share OpenMP programs with the aim of promoting discussions and exchange results in the field of OpenMP research.

During this second year, we have also developed sequential versions of dynamic programming algorithms for combinatorial optimization problems. From now on, we will apply the methodology described above.

As mentioned in the workplan, for the third year we plan to extend the former ideas to hybrid architectures where the parallel platform is composed of shared-memory nodes connected through an interconnection network. Mixing the MPI/OpenMP parallel programming models will be necessary to extend current prototypes.

3.3 Coordination of the project PELICAN

In order to coordinate the two subprojects, a total of three meetings were performed during the first year and six more meetings were held this second year. These occasions were employed to define the following common aspects of the project:

- Definition of an integrated design for the library.
- Definition of data structures for parallel vectors/matrices.
- Definition of the interfaces for the parallel routines (e.g., naming schemes and parameters dealing with the parallel execution) following the GSL "philosophy".
- Definition of a common serial image or interface including initialization and termination of the parallel machine and I/O.
- Design and creation of a CVS repository to exchange information, manage the project, and control the different versions of the codes.
- Design and creation of a web portal for the project: `http://www.pelican.ull.es`.
- Development of the general strategy to spread and validate results derived from the project.
- Design and elaboration of joint papers and selection of the appropriate national and international forums for diffusion of the results of the project.

4 Milestones and publications

4.1 Ph.D. thesis

So far, the following two doctoral thesis have been successfully elaborated and defended in direct relation with the project:

- "Towards the Automatic and Efficient Parallelization of the Dynamic Programming" by Daniel González and supervised by Dr. Casiano Rodríguez and Dr. Francisco Almeida. University of La Laguna (Spain), February 2003.
- "Evaluation of models in parallel computing" by Marcela Printista and supervised by Dr. Casiano Rodríguez. National University of San Luis (Argentina), March 2004.

4.2 Publications

The PELICAN project brings together two teams of researchers and gives them the opportunity to cooperate and share results. During the two years of the project, the two groups participating in it have prepared and presented the following joint research papers:

- "Strategies for parallelizing the GNU scientific library", J. Aliaga, F. Almeida, J.M. Badía, S. Barrachina, V. Blanco, M. Castillo, U. Dorta, R. Mayo, L.M. Moreno, E.S. Quintana, G. Quintana, C. Rodríguez, F. de Sande. XIV Jornadas de Paralelismo, Leganés (Spain), 2003.
- "Parallelization of the GNU scientific library on heterogeneous systems", J. Aliaga, F. Almeida, J.M. Badía, S. Barrachina, V. Blanco, M. Castillo, U. Dorta, R. Mayo, E.S. Quintana, G. Quintana, C. Rodríguez, F. de Sande. Third Int. Workshop on Algorithms, Models, and Tools for Parallel Computing on Heterogeneous Networks (HeteroPar'04), Cork (Ireland), 2004.
- "Parallelization of GSL: Performance of case studies", J. Aliaga, F. Almeida, J.M. Badía, S. Barrachina, V. Blanco, M. Castillo, U. Dorta, R. Mayo, E.S. Quintana, G. Quintana, C. Rodríguez, F. de Sande. Workshop on State of the Art in Scientific Computing (PARA'04), Copenhaguen (Denmark), 2004. In review.
- "Parallelization of GSL: Architecture, interfaces and programming models", J. Aliaga, F. Almeida, J.M. Badía, S. Barrachina, V. Blanco, M. Castillo, U. Dorta, R. Mayo, E.S. Quintana, G. Quintana, C. Rodríguez, F. de Sande. 11th European PVM/MPI Users' Group Meeting (EuroPVM/MPI 2004), Budapest (Hungary), 2004.
- "Paralelización de la biblioteca científica de GNU para sistemas heterogéneos", J. Aliaga, F. Almeida, J.M. Badía, S. Barrachina, V. Blanco, M. Castillo, U. Dorta, R. Mayo, E.S. Quintana, G. Quintana, C. Rodríguez, F. de Sande. XV Jornadas de Paralelismo, Almería (Spain), 2004.

As part of their research tasks, the members of the GCCP at UJI have presented the following works at international conferences and journals:

- "State-space truncation methods for parallel model reduction of large-scale systems", P. Benner, E. S. Quintana, G. Quintana. Parallel Computing, 2003.
- "Parallel model reduction of large-scale unstable systems", P. Benner, M. Castillo, E.S. Quintana, G. Quintana. Parallel Computing Conference (PARCO'03), Dresden (Germany), 2003.
- "Computing passive reduced-order models for circuit simulation", P. Benner, E. S. Quintana, G. Quintana. International Conference on Parallel Computing in Electrical Engineering, PARELEC'04, Dresden (Germany), 2004.
- "Evaluación de rutinas para comunicaciones colectivas sobre clusters", A. Aranda, R. Mayo, E.S. Quintana. XV Jornadas de Paralelismo. Almería (Spain), 2004.
- "Computing optimal Hankel norm approximations of large-scale systems", P. Benner, E. S. Quintana, G. Quintana. 43rd Conference on Decision and Control, Atlantis (Bahamas), 2004.

- “Parallel model reduction of large-scale descriptor linear systems via balanced truncation”. P. Benner, E. S. Quintana, G. Quintana. 6th. Int. Meeting on High Performance Computing for Computational Science, Valencia (Spain), 2004. Springer-Verlag “Lecture Notes in Computer Sciences”.

The members of the PGULL have also presented some other results of their research in several journal and conferences:

- “llc: A parallel skeletal language”, A. Dorta, J. González, C. Rodríguez, F. de Sande, Parallel Processing Letters. 2003.
- “From Complexity Analysis to Performance Analysis”, V. Blanco, J. González, C. León, C. Rodríguez, G. Rodríguez.. Proceedings of the International Conference Euro-par 2003, Klagenfurt (Austria), 2003.
- “On the Parallel Prediction of the RNA secondary Structure”, F. Almeida, R. Andonov, L. M. Moreno, V. Poirriez, M. Pérez, C. Rodríguez. Proceedings of the International Conference Parallel Computing (PARCO) 2003, Dresden (Germany), 2003.
- “On determining an efficient vertex from an arbitrary efficient solution in multiobjective linear programming”, J. Jorge, M Carrillo, Proceedings of the 6th Interantional Conference of Operations Research, Havana (Cuba), 2003.
- “Improvements in black hole detection using Parallelism”, F. Almeida, E. Mediavilla, A. Oscoz, F. de Sande, Proceedings of the International Conference Parallel Computing (PARCO) 2003, Dresden (Alemania), 2003.
- “Pipelines on heterogeneous systems: Models and tools”. F. Almeida, D. González, L.M. Moreno, C. Rodríguez. Concurrency: Practice & Experience, 2004.
- “The master-slave paradigm on heterogeneous systems: A dynamic programming approach for the optimal mapping”. F. Almeida, D. González, L.M. Moreno. Journal of System Architecture, 2004.
- “Exploiting task and data parallelism”. J. Rodríguez-Rosa, A.J. Dorta. C. Rodríguez, F. de Sande. Fifth European Workshop on OpenMP (EWOMP 2003), Aachen (Germany), 2004.
- “OmpSCR: una infraestructura para contribuir al desarrollo de OpenMP”. A.J. Dorta, A. González-Escribano, C. Rodríguez F. de Sande. XV Jornadas de Paralelismo. Almería (Spain), 2004.
- “Applying high performace computing techniques in astrophysics”. F. Almeida, E. Mediavilla, A. Oscoz, F. de Sande. Workshop on State of the Art in Scientific Computing (PARA’04), Copenhagen (Denmark), 2004. In review.
- “Parallelization of a Monte Carlo simulation for a space cosmic particles detector”. F. Almeida, C. Delgado, R. J. García Lopez, F. de Sande. Current Trends in Numerical Simulation for Parallel Engineering Environments New Directions and Work-in-Progress. PARSIM 2004. Special Session of EuroPVM/MPI 2004, Budapest (Hungary), 2004.

4.3 Cooperation with other national and international groups

The GCCP of the UJI collaborates with the following international research groups:

- The *Numerical Mathematics Group* of the Technical University of Chemnitz (Germany) and the the *Numerical Mathematics Group* of the Technical University of Berlin (Germany). Here, we combine our knowledge of parallel techniques with both groups' expertise in numerical algorithms to solve large-scale problems arising in control theory.
- The *Parallel Computing Group* of The University of Texas at Austin (USA). The goal of this joint work is to develop new parallel algorithms for sparse and dense linear algebra.

The PCGULL group is collaborating with the following national and international research groups:

- The *Parallelism and Combinatorial Optimization Group* of the Automatics and Industrial Mechanics Laboratory (LAMIH/ROI) from the University of Valenciennes (France). The collaboration with the group is focused in the application of parallel dynamic programming techniques to Biology.
- The *Group of Computer Architecture* of the Depto. of Electrónica y Computación from the University of Santiago de Compostela (Spain). We cooperate with this group in research related with performance analysis for the GRID.

Obviously, both groups also work actively together in the development of the PELICAN project.

4.4 Other results

- In order to facilitate the spread of the results of the project we have developed the project site at `http://www.pelican.ull.es`.
- The serial and parallel codes for sparse linear algebra that are being developed as part of this project, are incorporated the parallel library *SpaRed* for model reduction of sparse linear systems, which is distributed from its own web site at `http://spine.act.uji.es/~plicmr`.

  We are aware that this parallel library is currently being used to solve problems by other groups directly cooperating with us.

References

[1] GNU scientific library. `http://www.gnu.org/software/gsl/gsl.html`.

[2] OpenMP Forum. *OpenMP Architecture Review Board: OpenMP C and C++ application program interface v. 1.0*, 1998. `http://www.openmp.org/specs/mp-documents/cspec10.ps`.

[3] MPI Forum. The MPI standard. `http://www.mpi-forum.org/`.