

High performance statistical computing with parallel R: applications to biology and climate modelling

Nagiza F. Samatova¹, Marcia Branstetter¹, Auroop R. Ganguly², Robert Hettich³,
Shiraj Khan², Guruprasad Kora¹, Jiangtian Li⁴, Xiaosong Ma^{1,4}, Chongle Pan^{1,5},
Arie Shoshani⁶ and Srikanth Yoginath¹

¹Computer Science and Mathematics Division, ²Computational Sciences and Engineering Division, and ³Chemical Sciences Division, Oak Ridge National Laboratory; ⁴Department of Computer Science, North Carolina State University; ⁵Genome Science and Technology Graduate School, ORNL-UTK; ⁶Computational Research Division, Lawrence Berkeley National Laboratory.

E-mail: samatovan@ornl.gov

Abstract. Ultrascale computing and high-throughput experimental technologies have enabled the production of scientific data about complex natural phenomena. With this opportunity, comes a new problem – the massive quantities of data so produced. Answers to fundamental questions about the nature of those phenomena remain largely hidden in the produced data. The goal of this work is to provide a scalable high performance statistical data analysis framework to help scientists perform interactive analyses of these raw data to extract knowledge. Towards this goal we have been developing an open source parallel statistical analysis package, called *Parallel R*, that lets scientists employ a wide range of statistical analysis routines on high performance shared and distributed memory architectures without having to deal with the intricacies of parallelizing these routines.

1. Introduction

The research cycle of science applications includes more than just designing an experiment (e.g. simulation model) and collecting data (e.g. simulation output). After, or while, the results are generated, the scientists perform data analysis to discover, build, or test a new view of scientific reality (scientific discovery) [1]. These discoveries feed back to the design of new experiments.

Fundamental differences in data context and access patterns exist among the data production and data analysis steps. This places some unique computation, memory, and I/O requirements on the analysis tools for massive scientific data sets. First, the data generation often proceeds from one time step to the next. In contrast, data analysis often requires the *global context* of the data, across multiple time steps, to discover not only short-range but also long-range multi-scale relationships of the phenomena under study. Thus, the full context analysis software becomes tremendously memory intensive, creating a potential bottleneck in the entire scientific discovery cycle. Second, such a data exploration process is *interactive* in nature, requiring almost real-time access and response rates. Thus, the analysis software requires high-performance I/O and computation.

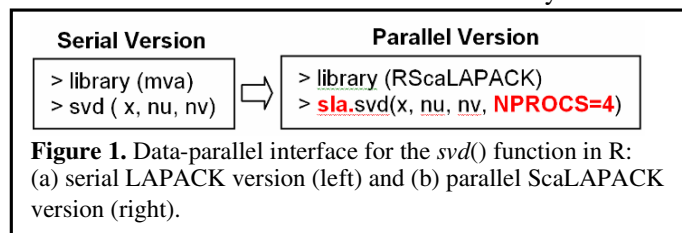
Our goal is to scale-up the existing data analysis tools, both in capability and capacity, to bridge the gap between scientific data production and data processing rates [2]. The challenge is that *many of the current data analysis routines are written in non-parallel languages such as IDL and are not scalable*

to massive data sets. To address this challenge, our tactical approach is based on parallelizing a statistical package, called R [3]. Similar to IDL, R is a non-parallel language and is not designed to handle massive data-sets. Our choice of R, however, is driven by its open source availability, extensibility, and very broad usability by a large scientific community. Our long-term strategy is to provide a middleware between a *specific high-level* data analytics language such as R, IDL, or MATLAB and a *generic, parallel, optimized, portable* computational analytics engine.

Two major approaches to enable parallel processing with R have been introduced. The first approach offers message passing capabilities between R processes. It essentially provides wrappers for MPI and PVM routines realized through R add-on libraries like *Rmpi* [4] and *rpvm* [5]. While flexible for writing almost any parallel program in R, these approaches lack efficiency due to the interpreted nature of the underlying programming language. These libraries are not transparent to the users with limited knowledge of parallel computing. The second approach, such as R *snow* library [6], offers a capability to address embarrassingly parallel statistical computations. While simple-to-use, it is limited to computations that require no coordination between R processes – all processes work on their local data, perform exactly the same function, and return the result to the parent process. In contrast, our approach addresses the above issues by providing a framework that can explore both data parallelism and task parallelism in R automatically and efficiently.

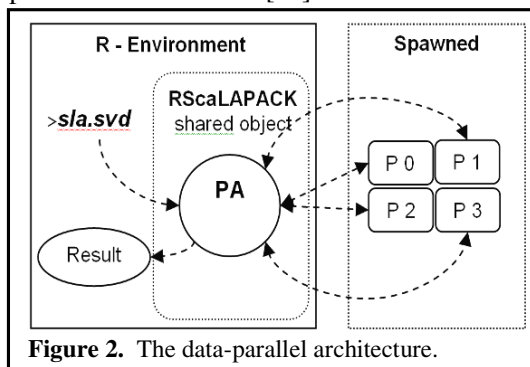
2. Parallel R

Our approach is driven by the need to provide a transparent parallel computational capability to the user. Through *Parallel R* the user can set up the parallel environment, distribute data and carry out the required parallel computation but maintain the same look-and-feel interface of the R system. Two major levels of parallelism within Parallel R are supported: *data parallelism* and *task parallelism*. With a data parallel approach (e.g. parallel clustering or PCA), the given data is divided among various processes that perform more or less the same task on their data chunks to obtain the desired result. With task parallelism (e.g. Markov Chain Monte Carlo), a given job is divided into various tasks that are executed in parallel to obtain the expected result. The Parallel R takes care of task/data partitioning, task scheduling, and mapping to computing resources requiring no or few modifications to the serial R codes from the user.



2.1. Data-Parallelism

With data-parallelism, we provided “hooks” to embed a parallel data analysis routine into an R environment with low performance overhead. As a proof-of-principle demonstration of how to use such hooks, we released an *RScaLAPACK* library [7-8] that replaced data analysis routines in R that are based on LAPACK [9] linear algebra solvers with the corresponding parallel, optimized and portable ScaLAPACK [10] routines. The new interface has mimicked the original R interface through



a single function call (figure 1). The introduced minor changes to the interface are not mandatory and are provided for the purpose of having user-level control over serial vs. parallel routines.

The architecture underlying data-parallel libraries such as *RScaLAPACK* is motivated by the need to perform efficient parallel computations on large data sets outside the parent R process and to manage the execution of these external processes. The computational entity running outside the parent R process (*Spawned Processes*) could be any parallel

computational system with a well-defined data distribution and result collection strategies, like the one provided by ScaLAPACK. The management entity (*Parallel Agent (PA)*) orchestrates the data flow between R environment and the parallel computational unit (figure 2). The *Parallel Agent* was designed to provide a single entry-point to the R environment for data handling while allowing to plug-in any generic parallel computational system.

2.2. Task-Parallelism

With task parallelism, our goal was to provide the task parallel capability, where a given job is divided into various tasks that are executed in parallel to achieve a reduction in the time involved in an analysis process. An initial prototyped library, called *taskPR* [11], provided a way to detect the out-of-order independent R tasks and intelligently delegate each task to remote worker processes in order to execute them concurrently. Again, the changes introduced to the interface were minimal to provide simple user-level control over task parallel vs. serial execution (figure 3).

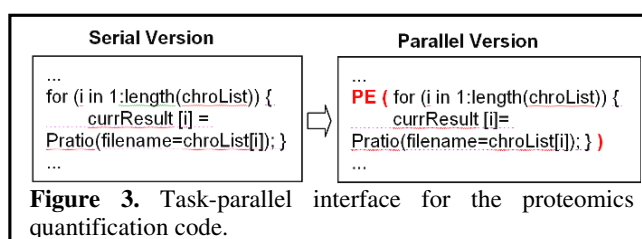


Figure 3. Task-parallel interface for the proteomics quantification code.

2.3. Analysis of Performance

The test runs were performed on the ORNL SGI Altix 3700 system with 256 Intel Itanium2 1.5 GHz processors, a total of 2TB shared memory, and a single system image of 64-bit Linux OS. The performance metrics include: (a) a *speedup factor*, $S(p)$, a measure of relative execution time between a p -processor system and a single processor system and (b) an *overhead*, $O(p)$, the percentage of additional execution time induced by the parallel R environment. The overhead is due to spawning p processes, distributing input data among them, and aggregating the results.

Figure 4 shows the performance results for the parallel R function called *sla.solve()* compared to the equivalent R function *solve()*. The following observations could be made: (a) the scalability in terms of both the problem size and the number of processors; (b) the speedup increase with higher input data size; and (c) the reduced overhead with increased input data size. Specifically, a higher speedup is observed for larger matrix sizes. For each matrix size, the best speedup is achieved at specific number of processors. This is the point where communication cost dominates computation cost. For the matrix of size 1024x1024, for example, the speedup at 4 processors is higher than that at 128 processors. Likewise, for the matrix of size 8192x8192 the best speedup of 116 is attained using 128 processors. Conversely, the overhead gets reduced with the increasing matrix size.

3. Parallel R Use in Scientific Applications

3.1. Biology: Discovery of Novel Genes and Pathways Involved in Aromatic Compound Degradation

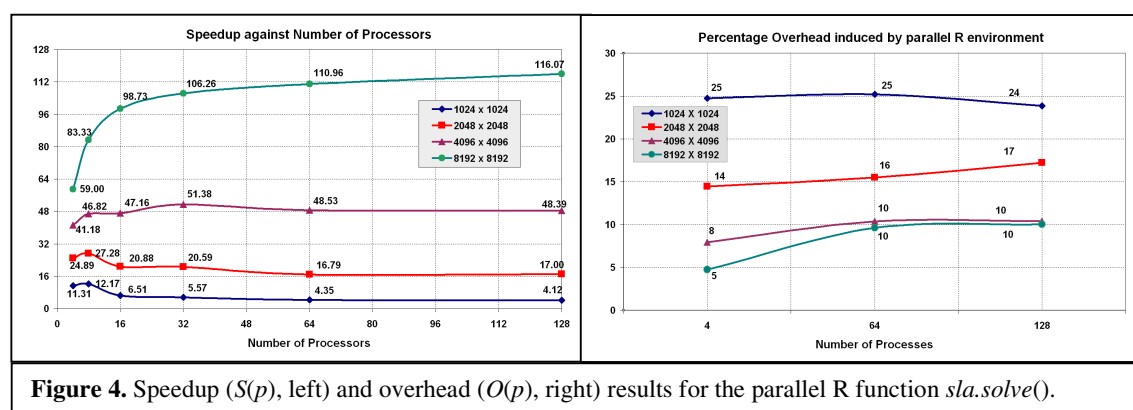


Figure 4. Speedup ($S(p)$, left) and overhead ($O(p)$, right) results for the parallel R function *sla.solve()*.

Aromatic compound degradation is a vital link in the global carbon cycle, involved in the process of plant residue decay. Large quantities of industrially generated aromatic contaminants in the environment are cleaned up by biodegradation using bacteria. *Rhodopseudomonas palustris* is a bacterium capable of anaerobic degradation of such aromatic contaminants. It is a DOE model organism for characterizing poorly understood aromatic compound degradation pathways. The diverse aromatic compounds are hypothesized to be converted to the central intermediate, benzoyl-CoA, via peripheral pathways, including coumarate pathway. Using quantitative shotgun proteomics and microarray measurements, we identified putative proteins in the poorly characterized coumarate pathway [12]. We also observed that aromatic compound degradation is accompanied with the regulation of many pathways in carbon metabolism, cross-membrane transportation and stress response. For this analysis we used our open source package ProRata [13-14] initially built on top of parallel R to address informatics challenges in quantitative proteomics: massive data management and extremely noisy data. We used *taskPR* to parallelize protein abundance ratio calculations and observed almost linear speed-up.

3.2. Climate: Characterization of Extreme Events under the Increased Carbon Dioxide Scenario

A climate extremes analysis was done on the daily precipitation in the Chicago area using results from CCSM3 simulations using the A2 scenario of the IPCC 4th Assessment. The A2 scenario is one of the most extreme scenarios involving increasing carbon dioxide during the period 2000-2099. For the extremes analysis, the daily precipitation value is defined as an “extreme” when it exceeds the 99th

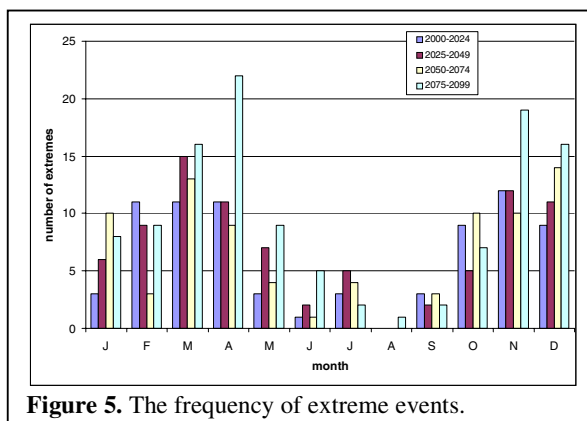


Figure 5. The frequency of extreme events.

percentile of daily precipitation time series. The rainfall in Chicago is the greatest during the summer months, and the number of extreme precipitation days is the lowest during this time period and the highest during the spring and fall. The scenario time frame was broken into four quarters. The total extremes during those periods were similar for the first three, but the last quarter showed 50% more extremes. Monthly details of these quarters are shown in figure 5. During seven of the twelve months, the last quarter of the simulation (2075-2099) showed the highest number of extremes. The temporal variability of intensity of precipitation extremes in Chicago was also studied using R libraries, such as *ismev* [15] and *extRemes* [16], for four quarters from 2000-2099. Analysis showed that the last quarter of this century will experience the most intense extremes. This experiment was done on a single grid point in the model results. The *taskPR* library was used to extract 6-hourly precipitation data of the whole world at $1.4^0 \times 1.4^0$ grid resolutions from 2000-2099 comprising 7,301 netCDF files totaling 2.9 TB of data. An almost linear speedup with the number of processors was observed, since *taskPR* was performing multiple serial reads in parallel as opposed to single process reading of all the files sequentially one after the other. The same libraries are used to perform similar analysis for every land point in the model and other variables in addition to precipitation.

4. Future Research

Our ultimate plan for achieving high performance statistical computing is to develop the *Parallel Scientific Data Analysis (PSDA)* library that will

Figure 6: Architecture diagram of the Parallel Scientific Data Analysis Library. It shows a stack of components: R, GDL/IDL, and MATLAB at the top, connected to Language Specific Middleware. Below that is the Generic Parallel Engine, which contains a DAG Analyzer, Language Optimizer, and Task Scheduler.

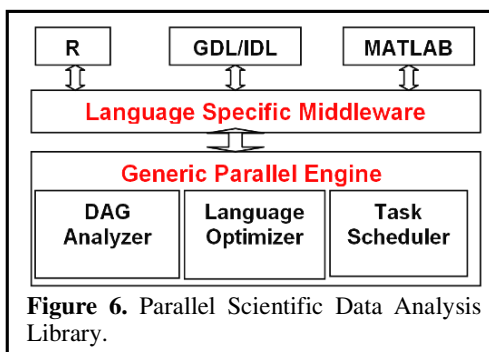


Figure 6. Parallel Scientific Data Analysis Library.

incorporate these task/data parallel techniques in a common parallel statistical computing environment. This library would allow application scientists to use their traditional data analysis codes but execute them on a parallel platform with minor modifications to the codes. The PSDA library comprises *Language Specific Middleware (LSM)* and *Generic Parallel Engine (GPE)* (figure 6). LSM will provide language specific parser routines along with a unified interface to the parallel statistical analysis functionalities provided by GPE. The GPE would no longer be limited to R, but rather would be pluggable to similar mathematical analysis environments like MATLAB, IDL/GDL, etc. A *Directed Acyclic Graph (DAG) Analyzer* is being developed to identify out-of-order tasks using data dependency techniques. Loop and other control structure optimizations are done through a *Language Optimizer*.

Acknowledgements

This work was funded by the DOE Scientific Data Management Center (<http://sdmcenter.lbl.gov>). Oak Ridge National Laboratory is managed by the University of Tennessee-Battelle, L.L.C. for the Department of Energy under contract DOE-AC05-00OR22725.

References

- [1] Ostrouchov G. and Samatova NF 2003 High End Computing for Full-Context Analysis and Visualization: when the experiment is done *Proc. of the High End Computing Revitalization Task Force (HECRTF)* (Washington, DC, June 16-18 2003)
- [2] Mount R *et al.* 2004 *The Office of Science Data-Management Challenge*, <http://www-user.slab.stanford.edu/rmount/dm-workshop-04/Final-report.pdf>
- [3] Venables WN and Smith DM 2002 *An Introduction to R* (Network Theory Ltd.)
- [4] Yu H 2006 *The Rmpi Package* <http://stat.cmu.edu/R/CRAN/doc/packages/Rmpi.pdf>
- [5] Li N and Rossini AJ 2005 *The rpvm Package* <http://cran.r-project.org/doc/packages/rpvm.pdf>
- [6] Tierney L, Rossini AJ, Li N and Sevcikova H 2006 *The Snow Package* <http://cran.r-project.org/doc/packages/snow.pdf>
- [7] Yoginath S, Samatova NF, Bauer D, Kora G, Fann G and Geist A 2005 RScalAPACK: high-performance parallel statistical computing with R and ScaLAPACK *Proc. 18th Int'l Conf. on Parallel and Distributed Computing Systems* (Las Vegas, Nevada, September 12 – 14, 2005)
- [8] Samatova NF, Yoginath S, Bauer D, Kora G 2005 *The RScalAPACK Package* <http://cran.r-project.org/doc/packages/RScalAPACK.pdf>
- [9] Anderson E, Bai Z, Bischof C, Demmel J, Dongarra J, Croz JD, Greenbaum A, Hammarling S, McKenney A and Sorensen S 1999 *LAPACK Users' Guide* (SIAM)
- [10] Blackford LS, Choi J, Cleary A, D'Azevedo E, Demmel J, Dhillon I, Dongarra J, Hammarling S, Henry G, Petitet A, Stanley K, Walker D and Whaley RC 1997 *ScaLAPACK User's Guide* (SIAM)
- [11] Samatova NF, Bauer D, Yoginath S, Kora G 2005 *The taskPR Package* <http://cran.r-project.org/doc/packages/taskPR.pdf>
- [12] Pan C, Lankford PK, Zhang B, Oda Y, Samatova NF, Harwood CS, Pelletier DA, Hettich RL 2006 *Proc. the 54th ASMS Conference on Mass Spectrometry* (Seattle, WA, May 28 - June 1)
- [13] Pan P, Kora G, Tabb DL, Pelletier DA, McDonald WH, Hurst GB, Hettich RL and Samatova NF 2006 Robust estimation of peptide abundance ratios and rigorous scoring of their variability and bias in quantitative shotgun proteomics *Anal. Chem.* (In press)
- [14] Pan P, Kora G, McDonald WH, Tabb DL, VerBerkmoes NC, Hurst GB, Pelletier DA, Samatova NF and Hettich RL 2006 ProRata: a quantitative proteomics program for accurate protein abundance ratio estimation with confidence interval evaluation, <http://www.MSProRata.org> *Anal. Chem.* (In press)
- [15] Coles S 2006 *The ismev Package* <http://cran.r-project.org/doc/packages/ismev.pdf>
- [16] Gilleland R, Katz R and Young G *The extRemes Package* <http://cran.r-project.org/doc/packages/extRemes.pdf>