GridR: An R-based grid-enabled tool for data analysis in ACGT clinico-genomics trials

Dennis Wegener
Fraunhofer IAIS
Schloss Birlinghoven
53754 St. Augustin, Germany
+49 2241 14 2261
dennis.wegener@iais.fraunhofer.de

Thierry Sengstag
Swiss Institute of Bioinformatics and Swiss Institute for Experimental Cancer Research
Bâtiment Génopode
CH-1015 Lausanne, Switzerland
+41 21 692 4096
thierry.sengstag@isb-sib.ch

Stelios Sfakianakis
Biomedical Informatics Laboratory
Institute of Computer Science
FORTH, Greece
+30-2810-391650
ssfak@ics.forth.gr

Stefan Rüping
Fraunhofer IAIS
Schloss Birlinghoven
53754 St. Augustin, Germany
+49 2241 14 3512
stefan.rueping@iais.fraunhofer.de

Anthony Assi
OUEST-genopole Bio-Informatics Platform
Symbiose Team
IRISA-INRIA
Rennes, France
+33 (0)2 99 84 71 58
anthony.assi@irisa.fr

Abstract

In this paper, we describe an analysis tool based on the statistical environment R, GridR, which allows using the collection of methodologies available as R packages in a grid environment. The aim of GridR, which was initiated in the context of the EU project Advancing Clinico-Genomics Trials on Cancer (ACGT), is to provide a powerful framework for the analysis of clinico-genomic trials involving large amount of data (e.g. microarray-based clinical trials). As a proof of concept, an example of microarray-based analysis taken from the literature was reproduced using GridR. As GridR will ultimately be made available to the ACGT community as a web service, we are sketching the ACGT project and its architecture in the present article as well.

1. Introduction

1.1. Overview of ACGT

In recent years, the rapid development of high-throughput genomics and post-genomics technologies has provided clinicians fighting cancer with new discovery paths and has opened the possibility to develop patient-specific treatment strategies. However, the amount of information now available for each patient (e.g. in microarray context from 10'000s to 100'000s of variables summarizing up-to millions of array features) has rendered difficult the isolation of the clinically relevant information from all available data. Considering the current size of clinical trials (hundreds of patients), there is a clear need, both from the viewpoint of the fundamental research and from that of the treatment of individual patients, for a data analysis environment that allows the exploitation of this enormous pool of data [18]. This is the aim of the Advancing Clinico-Genomics Trials on Cancer (ACGT) project [1].

ACGT aims at developing an open-source IT infrastructure to provide the biomedical community with the tools needed to integrate complex clinical information and make a concrete step towards the tailorization of treatment to the patient.

On the data side, the ACGT environment is designed to be versatile and will allow the integration of high-throughput databases with data both from existing (e.g. microarrays, imaging) and future technologies (e.g. high-throughput proteomics). The design of the platform considers the integration of private (i.e. trial-specific) databases with public ones, thus making publicly available datasets potentially immediately available for hypothesis validation and meta-analyses.

On the methodology side, the ACGT platform is designed to be modular, allowing to integrate additional data analysis tools (software, both open-source and commercial, web services) as plugins, as they become available.

Considering that the amount of data generated is expected to rise to several gigabytes of data per patient in a close future access to high-performance computing resources will be unavoidable. Hence, Grid computing [8] appears as a promising technology. Access and use of Grid-based resources is thus an integral part of the design of the infrastructure.

There are a number of other projects that aim at developing Grid-based infrastructure for post-genomic cancer clinical trials, the most advanced of which are NCI's caBIG [5] (Cancer Biomedical Informatics Grid) in the USA and CancerGrid [4] in the UK. The overall approach in those projects is somewhat different from the one in ACGT. In caBIG, the bottom-up, technology-oriented, approach was chosen, in which the focus was put on the integration of a large number of analysis tools but with weak concern on data privacy issues. CancerGrid on the other hand addresses the very needs of the British clinical community. As a result some aspects of the project may not fully overlap with the European and international scope of ACGT.

In ACGT, with two on-going international clinical trials actually conducted in the framework of the project, the approach is top-down, with clinicians' and biomedical data analysts' needs at the heart of all technical decisions, considering data privacy issues as central as data analysis needs.

Finally it should be mentioned - although these aspects are not described in the present text - that the ACGT project addresses legal and ethical issues related to clinical trials in distributed computing environment. Defining a sound legal and ethical framework for the use of clinical data is essential to gain public acceptance and support for such initiatives.

1.2. R in clinico-genomic context

The present article describes the initial implementation of GridR, one of the important analysis tools to be used in the ACGT environment. GridR is based on the open-source statistical package R [16]. The R environment provides a broad range of state-of-the-art statistical, graphical techniques and advanced data mining methods (including comprehensive packages for linear and non-linear modelling, cluster analysis, prediction, hypothesis tests, resampling, survival analysis, time-series analysis), it is easy extensible and turned out as the de facto standard for statistical research and many applied statistics project, especially in the biomedical field. (R itself is based on the S language [3] which is also implemented in the commercially available Splus system).

The associated project BioConductor [10] addresses the needs of the biomedical and biostatisticians community for genomic data-analysis oriented R packages. Numerous methods available as R/BioConductor packages and considered experimental a few years ago have been stabilized and became accepted standard in the analysis of high-throughput genomic data. Integrating R/BioConductor in an open-source clinical data-analysis environment is thus completely meaningful.

In the ACGT analysis environment, R is used as a user interface and as an analysis tool. R as user interface is supposed to serve as programming language interface to the ACGT environment. Used as analysis tool, the goal is to achieve a seamless integration of the functionality of R and the ACGT semantic data services in a grid environment, hiding complexity of the grid environment as the user might not want to or is not capable to deal with.

The rest of the paper is organized as follows. In Sec. 2 we sketch the ACGT architecture and give an overview over the key components. Sec. 3 describes the general scenario of statistical analysis in the context of clinico genomic trials and introduces a use case. In Sec. 4 we describe the grid enabled R environment GridR in ACGT, which will be evaluated in Sec. 5, based on the use case "Farmer scenario". Finally, in Sec. 6 we describe the relation to other project from the same area, discuss the results of the paper and give a future outlook.

2. ACGT Architecture

The ACGT requirements in terms of data management, efficient utilization of computational resources, and security can be matched by the adoption of a Grid infrastructure. The adopted architecture builds upon the Grid fabric and it is further enhanced by the deployment of Web Services and Semantic Web technologies. These technologies, although initially separated, are currently converging in a complementary way.

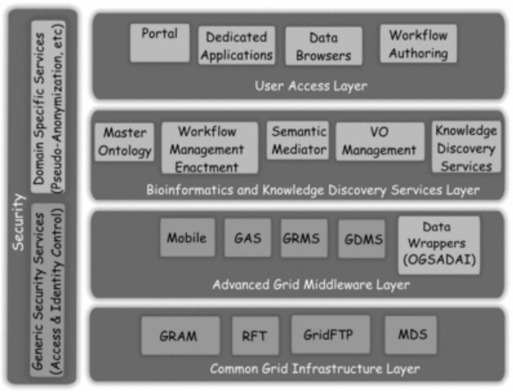

Security
Domain Specific Services (Pseudo-Anonymization, etc)
Generic Security Services (Access & Identity Control)
Portal
Dedicated Applications
Data Browsers
Workflow Authoring
User Access Layer
Master Ontology
Workflow Management Enactment
Semantic Mediator
VO Management
Knowledge Discovery Services
Bioinformatics and Knowledge Discovery Services Layer
Mobile
GAS
GRMS
GDMS
Data Wrappers (OGSADAI)
Advanced Grid Middleware Layer
GRAM
RFT
GridFTP
MDS
Common Grid Infrastructure Layer

Figure 1: The ACGT layered initial architecture.

The adopted architecture for ACGT is shown in more detail in Figure 1. A layered approach has been followed for providing different levels of abstraction and a classification of functionality into groups of homologous software entities [21]. In this approach we consider the security services and components to be pervasive throughout ACGT so as to provide both for the user management, access rights management and enforcement, and trust bindings that are facilitated by the Grid and domain specific security requirements like pseudonymization and anonymization. Apart from the security requirements, the Grid infrastructure and other services are located in the first (lowest) two layers: the Common Grid Layer and the Advanced Grid Middleware Layer. The upper layer is where the user access services, such as the portal and the visualization tools, reside. Finally, the Bioinformatics and Knowledge Discovery Services are the "workhorse" of ACGT and the corresponding layer is where the majority of ACGT specific services lie. Some characteristic examples of these services are:

- Mediator Services that offer uniform access to distributed and heterogeneous clinical and biomedical databases.
- Ontology Services that provide a conceptualization of the domain through the Master Ontology for Cancer and the "domain of discourse" for constructing complex queries for the mediator services.
- Workflow Enactment Services that support the efficient management and execution of complex biomedical workflows.
- Metadata Repositories and the corresponding services for the persistence and management of services' metadata descriptions.
- An assortment of data mining and knowledge discovery tools and services that fulfill the data analysis requirements of ACGT. In essence the GridR service is the most prominent tool for performing these data analyses in ACGT.

In the following sections the description of GridR is related to the current status of the integration into the ACGT environment, which is currently focused on the access to the ACGT grid services.

3. Scenario

In order to illustrate the working principles of GridR, we have selected the article by Farmer et al. [6], a simple clinical research project available from the literature and for which all data were available online [9]. This also provides a validation of GridR in a realistic usage.

In the "Farmer scenario", microarray data (Affymetrix U133A gene expression microarrays) obtained from breast cancer tumor samples of 49 patients are used to associate subtypes of breast cancer to patterns of gene expression and molecular signatures. R and BioConductor packages are used to load, normalize and analyze the data. The present work will be validated by showing that the results of the original paper can be reproduced using GridR.

The validation of GridR actually implements only a subset of the analysis steps presented in the original article, namely the principal component analysis (PCA). On the other hand, steps related to the quality control of the arrays are shown, which were not presented in the original paper. Figure 2 shows the plots related to those steps; in particular the plot illustrating the PCA is seen to be identical to that in [6].

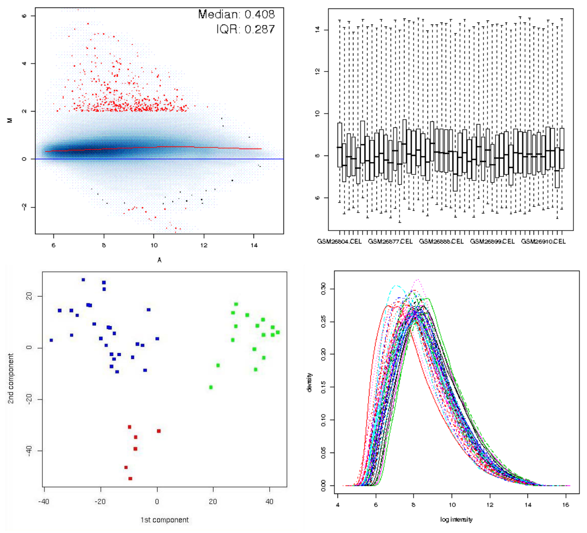


Figure 2: Farmer Scenario plots

4. Use case

The given scenario can be described from the technical point of view as follows: A researcher wants to perform interactive, grid-enabled data mining in the R environment. On his local machine (client) he develops algorithms using R as user interface. In clinical context, the data to be analyzed are usually so large (~800MB in the present, limited, scenario) that a transfer to the client might be ineffective or not possible. Besides, execution machines in the grid environment might have bigger computational power than client machines. It would be more efficient to just ship the algorithm to the execution machine (the best would be if the execution machine is the machine where the data is located in order to minimize transfer time), execute it remotely on this machine, and transfer the results back to the client.

The R environment is used both as a user interface (client) on the client side and as tool in the grid environment. GridR provides the execution of a single R function or a whole R script (wrapped as function) in the grid. The following subsections give a detailed description on the current implementation of GridR.

4.1. R as tool

In ACGT the R environment, namely GridR, is used as a tool for the remote execution of R code in the grid. More specifically, the task of the execution of the R code is submitted as a grid job to a remote grid machine. The current implementation of the server side GridR components that are related to the grid environment is based on several external software components, namely the Globus Toolkit 4 grid middleware [7], an installation of the R environment on the grid machines which will execute the functions remotely and a GRMS-Server installation from the Gridge toolkit [14] on a central machine in the grid environment that is responsible, for instance, for resource management. On the client side, GridR consists of a set of R functions and involves the Cog-kit [11], which is responsible for proxy generation and data transfer, and a GRMS-Client [14].

The client side part is structured around the components “RemoteExecution” (JobSubmission and JobDescription Generator) and “Locking”. The RemoteExecution component is responsible for the execution of R code as a job in the grid environment by transforming the R code to execute into a set of files, creating a job description file in the respective job description language, and submitting the job to the resource management system by the GRMS-client. During this process, the locking component takes care of the consistency of files/variables.

These components are based on R functions (see Table 1 and Table 2) so no changes in the core R implementation are necessary. The functions are based, among other, on the following R functionalities:

- callbacks - functions that are executed automatically by R after the user has issued a command (this is used when checking for results)
- active bindings - a variable is replaced by a function call which is handling the locking system and allows working interactively with that variable. When the variable is read, the predefined function is called and returns the value associated to the variable (or an error code if the variable is locked). When a value is assigned to the variable the function is called with the value as parameter for storage in an internal structure.
- parsing of error code - checks for missing values, variables and functions in the code which is executed remotely

Name	Action
grid.init	Initialization of all variables necessary for the grid execution
grid.exit	Unlocks all variables (grid.unlockAll) and removes all callbacks
grid.printJobs	Prints a summary of running jobs
grid.getLibraries	Returns the remote list of libraries (through grid.apply)
grid.getVersion	Returns the remote R version (through grid.apply)
grid.apply	Performs a remote execution of R functions; waits (callback) or sets a lock (grid.writeLock)

Table 1: List of functions for the users

Name	Action
grid.callback	Performs a callback
grid.waitForAll	Performs grid.callback for all jobs
grid.readLock	Read lock a variable
grid.writeLock	Write lock a variable
grid.unLock	Removes the lock from grid variable
grid.unlockAll	Unlocking of all grid variables
grid.catchObject NotFoundError	Error handling by parsing of error messages

Table 2: List of internal used functions

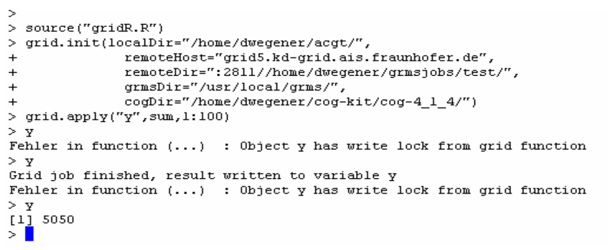
4.2. R as client

An R programming language interface that supports the access to the ACGT services is provided in the ACGT environment. This means that R users and developers will have access to distributed resources in a transparent fashion, as if those resources were local.

The complexity of the grid is thus hidden from the user.

Again, accessing the ACGT grid environment requires no changes in the core R implementation. In practice grid access is performed through the call of predefined R functions loaded from a package. R users can make use of the grid technology in a transparent way by passing the functions to be executed in the grid as input to one of those predefined functions (grid.apply) in their local code.

Figure 3 shows the execution of a simple sum function with the GridR client side components. Details on the steps of execution will be described in the following section.



```
>
> source("gridR.R")
> grid.init(localDir="/home/dwegener/acgt/",
+             remoteHost="grid5.kd-grid.ais.fraunhofer.de",
+             remoteDir=":2811//home/dwegener/grmsjobs/test/",
+             grmsDir="/usr/local/grms/",
+             cogDir="/home/dwegener/cog-kit/cog-4_1_4/")
> grid.apply("y",sum,1:100)
> y
Fehler in function (...)  : Object y has write lock from grid function
> y
Grid job finished, result written to variable y
Fehler in function (...)  : Object y has write lock from grid function
> y
[1] 5050
> 
```

Figure 3: Simple GridR Example

5. Evaluation and Testing

The implementation of GridR was validated on the basis of the "Farmer scenario", implementing some typical analysis steps of a microarray experiment.

In the present case, R was used in conjunction with the BioConductor packages affy, affyPLM and marray, which are specialized packages for microarray analysis, to build the individual modules of validation.

Besides loading the expression data matrix and associated clinical data, those modules contribute in:

- Producing some figures required for the quality control of the chips (e.g. RNA degradation plots)
- Producing "MvA plots" to obtain an overall view of the fraction of differentially expressed genes.
- Using a variance filter to pick unique probeset per gene and performing a principal component analysis, to verify that samples with similar subtypes group together.
- Extracting symbols of genes most correlated to molecular markers relevant to the analysis (androgen receptor, AR, and estrogen receptor, ESR1).

The analysis steps are wrapped into functions for remote execution with GridR. In the following the process of executing a single R function in the grid is described. The different steps of execution are briefly shown in Figure 4 (see also Figure 3 for a view from the R client).

- **Function loading.** The GridR functions are loaded from a file or package into the workspace of the R client.
- **Grid initialization.** The grid environment is initialized by calling the function grid.init. This function sets the paths to the cog-kit and the grms-client as well as the remote host that will be taken as execution machine.
- **Code writing.** The R code which is to be executed in the grid is written and wrapped as single R function in the local R environment
- **Grid submission.** The grid.apply function is called, that launches the process of submission. At first, the function to be executed in the grid and the needed parameters are written into a file (fx.bin). Then the R script which is actually executed on the remote machine is generated (script.R), which is followed by the creation of the job description for the grid-job-submission by GRMS (job.xml). The job description file contains information for the resource management system, e.g. which application to execute in the grid (R in the present case), a dedicated execution machine, which files to stagein (-out) from (to) the grid to (from) the execution machine etc. Next a shell file is created, which specifies the "workflow" which is performed on the client side (shell.sh). After that, the R client executes the created shell file in the background. That is, while the user can directly continue to work in the R session, the shell file creates a new proxy if necessary, performs the file copy from the client to the grid (to a GridFTP server later used for file stagein), submits the job to the grid system and, after the execution is finished, performs the file copy of the result files (staged out by the grid system) back to the client machine.

  During the remote execution the created R script is executed on the remote machine, which reads the parameters and the function from fx.bin, executes y=f(x) and writes the result into a file (y.dat).
- **Waiting for result.** While the remote execution is active and the R client waits for result (by checking if the file y.dat is created) the variable y is locked.

- **Result processing.** If file y.dat was created on the remote machine and, together with the result files, transferred back to the client, the file is loaded. The exit status is checked and – if the job was successful – the value is assigned to y and the variable is unlocked. Further more, if e.g. one of the result files is a Postscript file, ghostview is started for result visualization.

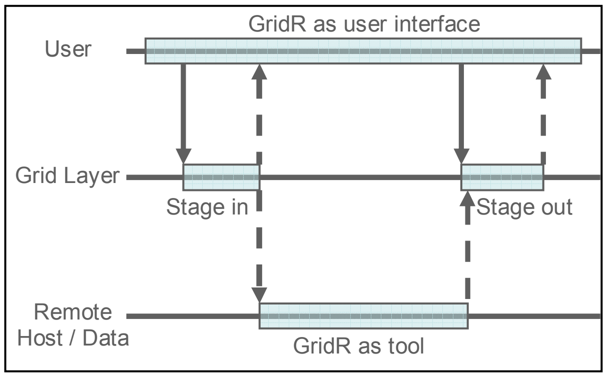


Figure 4: Steps of execution

6. Discussion, Related Work, Conclusion and Future Work

In this paper we have presented the integration of R in a Grid environment. In the biomedical and biostatisticians community R is widely used and turned out as de facto standard. GridR, as one of the important analysis tools in the ACGT environment, enables users to run experiments in the grid and profit from the advantages of grid technology by providing a grid enabled environment for R. The presented use case showed the execution of a complex analysis scenario. It should be considered a proof of concept, as measurements and statistics about the practical gains in terms of computational power still need to be done in the context of large clinical trials. However, it becomes clear that the availability of GridR will be of great use to clinicians and clinical-data analysts interested in computationally heavy data-mining, such as resampling techniques, full cross-validation of classifiers or meta-analyses.

The expected benefits for the users are twofold based on the duality of using the R as a client tool and executing the R scripts on the Grid. Firstly, we argue that the R environment is a popular tool among biostaticians and something that a lot of users are familiar with. Therefore using R as a user access environment to interact with the ACGT platform is beneficial for the acceptance and the active use of the project's infrastructure. Secondly, the "gridification" of the R execution layer implies that the users' analysis tasks are executed in an efficient and secure way relieving the client side of the computational burden and the need to download all the input data sets. This architecture also leaves room for many possible optimizations in the future, e.g. the implementation of a scheduler that in collaboration with the Grid scheduling facilities determines the best place to execute a specific R task based on the data that it requires.

To our knowledge there has not been so far any attempt for grid enabling the R environment itself and using it as user interface for accessing the Grid resources and invoking the Grid services. Analytical services in the Grid have been the focus of work in various previous projects. In particular the Grid Weka [11] toolkit was an effort to extend the Weka toolkit [23] so that it uses the Grid computational resources for data analysis tasks. The Weka4WS toolkit [20] is a similar work to support the data analysis on the Grid by offering a WSRF compliant interface to the machine learning algorithms provided by Weka. Also, the DataMiningGrid project [19] aims to deliver Grid interfaces and middleware components to facilitate the discovery and execution of data mining tools in distributed Grid environments. One of the results of this project is a "Data Mining Application Enabler" which grid-enables existing data mining applications. Nevertheless, the focus is on generic, mostly command line, applications (e.g. Bash or Python scripts, C applications, etc.), although R scripts could be also handled in the same way: as inputs to the command line R application.

In [10] a short overview over support for concurrent computation in R is given. Packages such as snow (Simple Network Of Workstations), based on the adoption of an article about benefits and problems with parallel processes in R [17], and rpvm provide interfaces in order to support the parallel computation on clusters. In contrast to the message-passing interface (MPI) [13] or the parallel virtual machine (PVM) [15], the snow package provides a higher level of abstraction that is independent of the communication technology.

Some rudimentary support for building client server applications that use R in the server side has been offered by toolkits like Rserve [22] or Rweb [2] but these efforts do not offer a seamless integration with the Grid technologies.

As future work we plan to extend the functionality offered by GridR to support the interfacing with the rest of the ACGT services, e.g. the mediator. This will enable the use of GridR as an all-encompassing user interface to the whole ACGT environment. Additionally, the provision of a WSRF compliant interface to the GridR service functionality will give the ability to compose higher level workflows that include GridR and will also make it more compatible with the service oriented view of the Grid.

7. Acknowledgments

The authors gratefully acknowledge the financial support of the European Commission for the Project ACGT, FP6/2004/IST-026996.

8. References

[1] ACGT (EU): http://eu-acgt.org/

[2] J. Banfield, "Rweb: Web-based Statistical Analysis", *Journal of Statistical Software*, Vol. 4, Issue 1, 1999 (http://www.math.montana.edu/Rweb/)

[3] R. Becker, J. Chambers, and A. Wilks, *The New S Language*, Chapman & Hall, London, 1988.

[4] CancerGrid (UK): http://www.cancergrid.org/

[5] Cancer Biomedical Informatics Grid, caBIG (USA): https://cabig.nci.nih.gov/

[6] P. Farmer, H. Bonnefoi, V. Becette, M. Tubiana-Hulin, P. Fumoleau, D. Larsimont, G. Macgrogan, J. Bergh, D. Cameron, D. Goldstein, S. Duss, AL. Nicoulaz, C. Brisken, M. Fiche, M., R. Iggo, "Identification of molecular apocrine breat tumours by microarray analysis", *Oncogene*, 24, 2005, 4660-4671

[7] I. Foster, "Globus Toolkit Version 4: Software for Service-Oriented Systems", *IFIP International Conference on Network and Parallel Computing*, Springer-Verlag LNCS 3779, pp 2-13, 2006

[8] I. Foster, C. Kesselman, S. Tuecke, "The Anatomy of the Grid: Enabling Scalable Virtual Organizations," *International Journal of High Performance Computing Applications*, vol. 15, no. 3, 2001, pp. 200—222.

[9] Gene Expression Omnibus (GEO), accession number GSE1561, http://www.ncbi.nlm.nih.gov/

[10] RC. Gentleman, VJ. Carey, DM. Bates, B. Bolstad, M. Dettling, S. Dudoit, B. Ellis, L. Gautier, Y. Ge, J. Gentry, K. Hornik, T. Hothorn, W. Huber, S. Iacus, R. Irizarry, F. Leisch, C. Li, M. Maechler, AJ. Rossini, G. Sawitzki, C. Smith, G. Smyth, L. Tierney, JY. Yang, J. Zhang, "Bioconductor: open software development for computational biology and bioinformatics", *Genome Bio.* (2004) 5(10):R80 (http://www.bioconductor.org/)

[11] R. Khoussainov, X. Zuo and N. Kushmerick, "Grid-enabled Weka: A Toolkit for Machine Learning on the Grid", *ERCIM News*, No. 59, October 2004

[12] G. von Laszewski, I. Foster, J. Gawor, P. Lane,."A Java Commodity Grid Toolkit", *Concurrency and Computation: Practice and Experience*, 13, 2001.

[13] MPI Forum: http://www.mpi-forum.org

[14] J. Pukacki, M. Kosiedowski, R. Mikołajczak, M. Adamski, P. Grabowski, M. Jankowski, M. Kupczyk, C. Mazurek, N. Meyer, J. Nabrzyski, T. Piontek, M. Russell, M. Stroiński, M. Wolski "Programming Grid Applications with Gridge", *Computational Methods in Science and Technology* vol. 12, Poznan 2006.

[15] PVM: http://www.csm.ornl.gov/pvm/pvm_home.html

[16] R Development Core Team (2005), "R: A Language and Environment for Statistical Computing", R Foundation for Statistical Computing, Vienna, Austria, ISBN 3-900051-07-0 (http://www.r-project.org/)

[17] A. Rossini, L. Tierney and N. Li, "Simple Parallel Statistical Computing in R", *UW Biostatistics Working Paper Series*, Working Paper 193, (March 5, 2003).

[18] S. Rüping, S. Sfakianakis, and M. Tsiknakis, "Extending workflow management for knowledge discovery in clinico-genomic data", in Nicolas Jacq et. al., editor, *From Genes to Personalized HealthCare: Grid Solutions for the Life Sciences, Proceedings of HealthGrid 2007*, volume 126 of Studies in Health Technology and Informatics, pages 184–193. IOS Press, April 2007.

[19] V. Stankovski, M. Swain, V. Kravtsov, T. Niessen, D. Wegener, J. Kindermann and W. Dubitzky, "Grid-enabling data mining applications with DataMiningGrid: An architectural perspective", *Future Generation Computer Systems*, accepted for publication

[20] D. Talia, P. Trunfio, O. Verta, "Weka4WS: a WSRF-enabled Weka Toolkit for Distributed Data Mining on Grids", *Proc. of the 9th European Conference on Principles and Practice of Knowledge Discovery in Databases* (PKDD 2005), Porto, Portugal, October 2005, LNAI vol. 3721, pp. 309-320, Springer-Verlag, 2005. (http://dx.doi.org/10.1007/11564126_32)

[21] M. Tsiknakis, M. Brochhausen, J. Nabrzyski, J. Pucacki, S. Sfakianakis, G. Potamias, C. Desmedt, D. Kafetzopoulos, "A Semantic Grid Infrastructure Enabling Integrated Access and Analysis of Multilevel Biomedical Data in Support of Post-Genomic Clinical Trials on Cancer", Digital Object Identifier: 10.1109/TITB.2007.903519 (to appear), http://ieeexplore.ieee.org/xpl/tocpreprint.jsp?isnumber=26793&punumber=4233

[22] S. Urbanek, "Rserve - A Fast Way to Provide R Functionality to Applications" in: *Proc. of the 3rd International Workshop on Distributed Statistical Computing* (DSC 2003), ISSN 1609-395X, Eds.: Kurt Hornik, Friedrich Leisch & Achim Zeileis, 2003 (http://rosuda.org/Rserve/)

[23] I. Witten and E. Frank, *Practical machine learning tools and techniques*. 2nd Edition, Morgan Kaufmann, 2005