GENETIC LINKAGE ANALYSIS CHALLENGES ON A DISTRIBUTED GRID ENVIRONMENT

A. Calabria, D. Di Pasquale, A. Orro, G. Trombetti, M. Gnocchi, L. Milanesi

Institute of Biomedical Technologies ITB-CNR, Milano, Italy, e-mail: luciano.milanesi@itb.cnr.it

The aim of the present work is to enable the use of high performance computing infrastructures, such as the EGEE-III Grid platform for the execution of genetic linkage analysis on very large SNPs (Single Nucleotide Polymorphism) markers data sets. The linkage analysis of SNPs has recently become a very popular approach for genetic epidemiology and population studies, aiming to discover the genetic correlation in complex diseases. It is a statistical method for detecting genetic linkage between disease loci and markers of known locations by following their inheritance in families through the generations. This is a NP-hard problem and the computational cost and memory requirements of the major algorithms proposed in literature grow exponentially with either pedigree size or number of markers. Implementations of the mentioned algorithms reflect these limits making analyses of medium/large data sets very hard on a single CPU. A web-based facility has been set up upon a high performance infrastructure, the EGEE Grid, in order to enable a tool for achieving a whole-genome linkage analysis. Test cases have been performed with 10.000 to 1 million SNPs per Chip.

Key words: linkage analysis, grid, distributed computing.

Introduction

The haploid human genome is a sequence of over 3 billion DNA base pairs and contains an estimated twenty to twenty five thousand protein-coding genes, therefore finding the relationships between the expression of a particular gene or genes and the outbreak of human diseases is a hard task. A common approach to this challenge starts with the Genetic Linkage Analysis, which is a statistical method used to identify the location on a chromosome of a given gene involved in a disease, relative to a known location of chromosome markers. This is obtained by comparing genetic data with information on pedigree structure, and following the inheritance of phenotypic alterations in families through the generations, exploiting the tendency for genes and genetic markers to be inherited together due to their location near one another on the same chromosome.

Markers used in this analysis are *Microsatellites* (polymorphic loci that consist of repeating units of 1–6 base pairs in length) and, recently, also *Single Nucleotide Polymorphisms* (SNPs, DNA sequence variations involving a single nucleotide). Two major algorithms (Lander, Green, 1987; Kruglyak *et al*., 1996; Elston, Stewart, 1997) have been proposed for the assessment of the genetic linkage, with several implementations (Schaffer, 1996; Markianos *et al*., 2001; Fishelson, Geiger, 2002); the problem is NP-hard and the computational cost of both algorithms grows exponentially for one of the two variables of the LOD Score equation, pedigree size and number of markers (Table 1). Implementations of the mentioned algorithms reflect these limits and the computational time for a medium size problem can require several CPU hours and huge RAM memory allocation. Therefore the role of high performance computing is getting more and more relevant in this field and more in general in the field of biological and medical scientific research, also due to the increasing quantity of data produced by the high throughput analysis techniques emerging nowadays. In this scenario, the use of distributed architecture environments can be an appropriate solution both from the computational point of view and for the data management, but submitting jobs, monitoring their status and retrieving the results can be challenging when working on a huge quantity of data.

Table 1

Main algorithms characteristics

Algorithms	Applications	Computational bounds	Increasing computational time by		
			individuals	loci	time
Elston-Stewart	Linkage, SLink, Fastlink, Vitesse	n° loci: ~8	linear	exponential	O((m2n)p)
Lander-Green	GeneHunter, Allegro, Merlin	n° loci: > 20	exponential	linear	O(m24p)
Bayesian Networks	Superlink	n° loci: nr	linear	linear	nr

Objectives

Actual technologies for chips (for example Illumina) provide SNP genotyping arrays, from 10.000 SNPs to more than 1 million; pedigree files, that collect the information of the family structure, are often large, counting more then 30 individuals. Computational time and space on a single CPU is unreasonable with these preconditions, therefore stands the need for a distributed and high performance computation infrastructure and a system that enables linkage analysis with large datasets.

The aim of the present work is to enable the use of the EGEE Grid Infrastructure for the execution of linkage analysis on very large SNPs data sets, creating a pipeline for the linkage process. These large calculation challenges are launched into the Grid infrastructure, distributing the needed processes among different computing elements. In this context, a user friendly web based access to grid resources is provided. This proposed facility has been tested with challenges performed with markers data collected using the largest chips currently available (up to 1M SNPs).

The computational problem

The Genetic Linkage Analysis method is based on the calculation of the *LOD scor*e, a test parameter defined as the Logarithm Of Odds ratio between hypothesis of linkage versus the null one. Its computation belongs to the class of NP-Hard problems (Piccolboni, Gusfiled, 1999), thus it has a complexity at least equal to the class NP-complete but it could also fall into the NP set. So far, approaches and algorithms proposed to solve the linkage analysis problem (quantitative and qualitative) (Table 1), are very computer intensive and the implementations of them suffer of hardware limitations. For example the Elston-Stewart algorithm, which has been adopted by applications such as Linkage, SLink, Faslink, Vitesse, grows exponentially for markers and linearly for individuals' parameters, whereas the Lander-Green one has an opposite behavior. Since we deal with SNP chips and thus with a very large number of markers, we adopt the algorithm proposed by Lander and Green: in this way we can handle a set of about 100 markers each time, but we have to monitor the pedigree size variable.

The computational infrastructure

The computational infrastructure used to implement this work is the Grid operated by the European Project EGEE (Enabling Grids for E-sciencE, Phase III) for the European scientific and research communities with more than 240 sites worldwide, providing access to more than 50.000 CPUs. It consists of a collection of computers, storages, special devices and services that are heterogeneous in every aspect, geographically distributed and dynamically linked by a wide-area network which can be accessed on-demand by a set of users with appropriate authentication and authorization. The Grid was originally invented as a practical solution to the problems of storing and processing the large quantities of data that are going to be produced by CERN's Large Hadron Collider (LHC). This computational facility can be seen as a service for sharing computer power and data storage capacity over the Internet, going beyond simple communication between computers, and aiming ultimately to turn the global network of computers into one vast computational resource. In this scenario, this infrastructure enables data sharing between thousands of scientists with multiple interests, tries to ensures that all data is accessible anywhere, anytime and

copes with different computer centers access policies, ensuring data security.

The EGEE Grid infrastructure runs upon a set of middleware services called gLite, which is integrated, certified and distributed by the project itself and world-widely deployed on the computational resources. The services available in the gLite distribution can be broadly classified into two categories: the Grid Foundation Middleware, covering the security of the infrastructure, the information, monitoring and accounting systems and the access to computing and storage resources; the other is a higher-level Grid Middleware, including services for job management, data catalogs and data replication, providing applications with end-to-end solutions. In order to use this infrastructure a personal authentication certificate is required. The certificate is released by a Certification Authority and associated to a Virtual Organization, a geographically independent group of collaborating scientists; the access to computational resources is executed through dedicated User Interfaces running the gLite middleware and its CLI commands.

The tests for the correctness of the linkage analysis algorithm and for the evaluation of its performances were performed on the computation cluster *Michelangelo* provided by the LITBIO project and is composed of 70 nodes, and 18.5TB of redundant storage. Each node is composed of two 275 dual-core AMD Opteron CPUs (total of 4 cores) and 8GB ram; all the nodes are connected via 10 Gbit/sec Infiniband for maximum scalability of the algorithms.

Methods

The system is designed in 3 different layers, as schematized in Fig. 1. The presentation layer shows a web based user interface which has been created to aid users to setup linkage challenges and to mask the complexity of low level interactions with the Grid middleware. The web page makes use of javascript as client side scripting language to increase interactivity and user friendliness, using asynchronous communications with back end PHP server pages to manage information exchange with the application layer and input file uploads. Through the web interface users can:

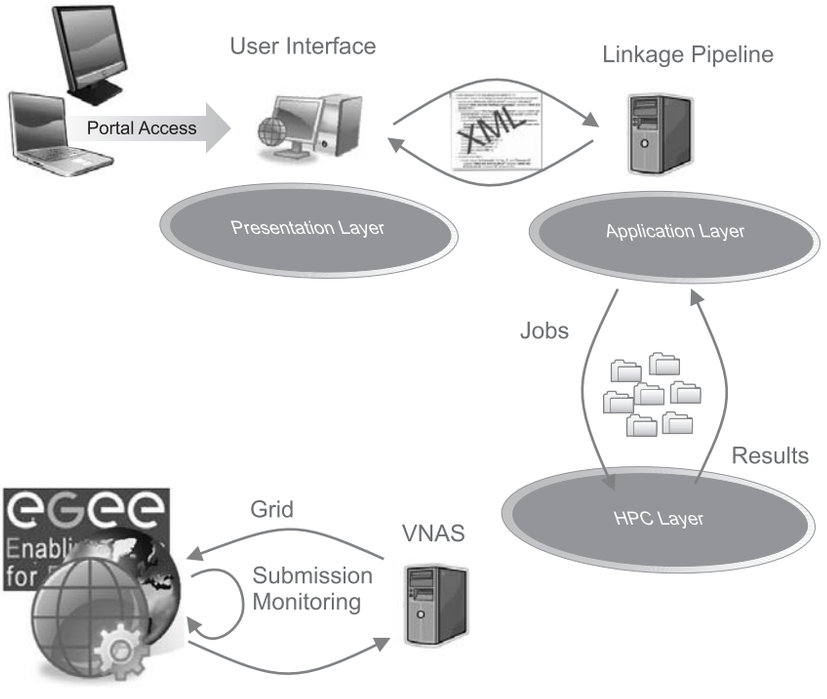

Fig. 1. Diagram of system's design.

In the presentation layer users interact with the application which produces an XML file; the application layer parses the XML file to extract the logics of execution and linkage parameters; in the final step the HPC layer interacts with the Grid middleware submitting jobs and monitoring their execution.

– Create the data pipeline using draggable modules that represent each operative step: the arrangement of logical blocks, managed by client-side scripting, makes building the data flow, from the input files to the outputs retrieving, a visual matter.

– Customize each step: choose and upload the input files, define optional data pre-processing to adapt inputs to different algorithms requirements, select one of the available algorithms and set the proper parameters.

– Launch the analysis and monitor the jobs' state, retrieve and download the results once the challenge is complete.

The communication between the Presentation and the Application Layers is obtained through the creation of an XML file that describes the data pipeline and all its parameters: this file provides a complete description of the challenge characteristics making possible successive quick resubmissions; the choice of XML language reflects the future possibility of the system to drift to web services technology, adapting the Application Layer to this protocol with a minor effort. The communication in the opposite direction, between the Application and the Presentation Layer, is obtained with asynchronous data exchanges polled by client scripts that request information to the server backend and display the available ones without the need of page reloads.

In the application layer the system first parses the XML file created by the presentation layer, then executes pre-processing operations (error detection, etc) and finally runs the workflow on the HPC layer, monitoring its status: real time information on jobs status is returned through the interface to the user. Preprocessing operations are related to select the correct SNPs from our genotype database, then formatting files splitting SNP markers on the basis of the parameters reported in the XML file, such as distance between two consecutive markers or interval in kilo-bases (nucleotides, DNA measure), and so forth. The last activity in this phase is the creation of execution files for the HPC infrastructure according to the setting specified by user. Once files have been created the challenge execution can begin and the monitoring process, which is demanded to the VNAS component of the HPC Layer, is activated. When each single job has finished its computation, all results are merged into a single CSV file, that is easily readable by biologists or domain experts, and a final global plot for each chromosome is created.

In the HPC layer, the submission engine splits the workload into small jobs and distributes analysis tasks over the available resources. This is achieved by a software layer, called VNAS (Trombetti, 2007), built on top of the grid middleware which monitors each single grid process and ensures the success of its computation by managing the resubmission of failed jobs automatically. When all tasks are computed the results are retrieved, merged and made available for downloading through the web interface.

The VNAS framework is an advanced system for the submission and monitoring of jobs onto the Grid environment. VNAS has integrated strategies for the detection of failures and hang-ups of Grid jobs and can perform automatic resubmission for jobs detected in such conditions. VNAS hence provides an abstraction with reliability over the Grid platform which significantly eases the task of developing new applications for the Grid. VNAS also significantly simplifies the development of certain complex Grid workflows by providing a callback system that eases the creation of arbitrarily complex multi-stage computational applications. In addition, VNAS provides an abstracted virtual sandbox which bypasses certain Grid limitations such as the maximum sandbox size, while simultaneously reducing the usage of Grid bandwidth and storage resources. The latter is achieved by transparently detecting equality of virtual sandbox files based on content, across different submissions, even when performed by different users, and by performing automatic garbage collection of files after N days of no-use.

The VNAS framework has been used in various projects including the present one, to raise the reliability and reduce the development time for new Grid applications and pipelines being developed.

Results

Tests were run to evaluate both efficiency on computational time and functionality of the proposed approach, obtaining an estimate of the Grid performances in comparison to a single 2 GHz CPU workstation and to the *Michelangelo* cluster composed by 280 CPU cores. The test, run using the Genehunter software, involved analysis of pedigrees composed by 25 subjects, including

individuals genotyped with different genotyping chips of markers from 10k up to 1 million of SNPs each. Considering the trade off between CPU load and memory requirements, a data subset of 50 SNPs was evaluated as the optimal workload for a Grid node and was assigned to each program run; the total number of runs needed to process all SNP data produced by each genotyping chip was split into jobs with an estimated duration of around 6 hours on a mid-range multicore CPU. Test results are summarized as follows: Table 2 shows the duration of the challenges in hours and Fig. 2 displays the relative plot.

Comparing the results of different computation infrastructures we can see that distributed analysis pipelines with number of data (linkage variables) close to the computational limits for a mid-range workstation achieved improvements of about 65–70 % in computational time compared to dual-core 2 GHz CPU execution and considering markers chips greater than 100 k, the advantage of the distributed architecture gets proportionally bigger, due to the difference between linear increase of computational time for the sequential run on the single CPU and the saturation trend of the parallelized data flow obtained distributing the workload on the Grid's computing elements. It must also be highlighted that computations with a higher number of individuals in the pedigree tree, were still performed on the distributed infrastructure, but resulted infeasible on the desktop workstation due to memory overflow. The tests also show that the average performance of the proposed system can't compete with a cluster environment in terms of pure computing time, even if it shows a comparable trend due to the similar workload distribution technique adopted (not MPI).

Conclusions

This application enables the user to launch genetic linkage analysis computations for medium to large challenges over a distributed computa-

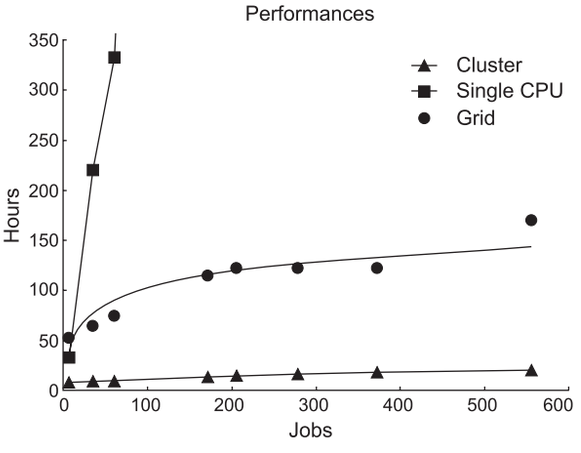

Fig. 2. Graph of the test results.

The computation efficiency of the distributed approach of our Grid-based system grows with the size of the challenge, being lower than a dedicated Cluster's but much higher than a sequential run on a single processor unit.

Table 2

Duration of the challenges. Test results

Genotyping chip (# of SNPs)	Runs (50 SNP)	Jobs (6 h)	Computational cost (hours)		
			single 2GHz CPU	cluster (70 nodes 280 CPU Cores)	grid
10 k	200	6	33	8	53
66 k	1320	35	220	9.5	64
100 k	2000	60	333	10	75
317 k	6340	172	1056	13	115
370 k	7400	206	1233	15	120
500 k	10000	278	1665	16	122
670 k	13400	373	2233	18	122
1 M	20000	556	3332	20	170

Notes. Data derived from different genotyping chips were analyzed with the Genehunter software using 3 computational infrastructures: our Grid-based system, a 280 cores Cluster and a single 2GHz CPU Workstation. The left side columns show the challenges characteristics, on the right the challenges durations expressed in hours as resulted for different environments.

tional infrastructure such as the EGEE Grid. The application offers an interface to customize the data pipeline and achieve, through the submission engine, the parallel processing of the pipeline tasks on the distributed resources. The low-level interactions with the Grid environment are managed with a reliable software layer that hides any complexity for the final user, allowing monitoring and results retrieving to be easily managed with the web interface. Tests made on the proposed system showed that this approach is mostly useful in high-end challenges, where Grid overheads are affecting overall execution times less compared to single CPU performances; only very small challenges may show higher efficiency when run in a single CPU workstation, while very large analyses are made accessible even without a dedicated cluster, resulting as a good and affordable solution when considering the cost-benefit ratio of the two infrastructures.

Acknowledgments

This work has been also supported by the Enabling Grids for E-sciencE (INFSO-RI-222667), CNR-BIOINFORMATICS, LITBIO and ITALBIONET and Italian-Canada FIRB-MUR Projects.

References

Elston R.C., Stewart J. A general model for the analysis of pedigree data // Hum. Hered. 1997. № 21. P. 523–542.

Fishelson M., Geiger D. Exact genetic linkage computations for general pedigrees // Bioinformatics. 2002. V. 18. № l. S189–S198.

Lander E.S., Green P. Construction of multilocus genetic linkage maps in humans // Proc. Natl Acad. Sci. USA. 1987. № 84. P. 2363–2367.

Kruglyak L., Daly M.J., ReeveDaly M.P., Lander, E.S. Parametric and nonparametric linkage analysis: a unified multipoint approach // Amer. J. Hum. Genet. 1996. № 58. P. 1347–1363.

Markianos K., Daly M., J., Kruglyak L. Efficient multipoint linkage analysis through reduction of inheritance space // Amer. J. Hum. Genet. 2001. V. 68. № 4. P. 963–977.

Schaffer A. Faster linkage analysis computations for pedigrees with loops or unused alleles // Hum. Hered. 1996. V. 46. № 4. P. 226–235.

Trombetti G.A. *et al*. Data handling strategies for high throughput pyrosequencers // BMC Bioinformatics. 2007. № 8(1). P. S22.

Piccolboni A., Gusfiled D. On the complexity of fundamental computational problems in pedigree analysis. Technical report of University of California, Computer Science Department, 1999.