Computational and Visualization tools for Genetic Analysis of Complex Traits

Mahen Jayawardena[1,2], Salman Toor[1] and Sverker Holmgren[1]

[1] Division of Scientific Computing, Department of Information Technology, Uppsala University, Sweden
(salman.toor,sverker.holmgren)@it.uu.se
[2] University of Colombo School of Computing, Colombo, Sri Lanka
mahen@cmb.ac.lk

Abstract. We present grid based tools for simultaneous mapping of multiple locations (QTL) in the genome that affect quantitative traits (e.g. body weight, blood pressure) in experimental populations. The corresponding computational problem is very computationally intensive. We have earlier shown that, using appropriate parallelization schemes, this type of application is suitable for deployment on grid systems.
A grid portal interface is often an ideal tool for biologists performing advanced genetic analysis. We describe an implementation of such a portal system and how it is used for performing multidimensional QTL searches efficiently.

1 Introduction

Most traits of medical or economic importance are quantitative. Examples are agricultural crop yield, growth rate in farm animals and blood pressure and cholesterol levels in humans. These traits are believed to be governed by a complex interplay between multiple genetic factors and the environment. One method to locate the genetic regions underlying a quantitative trait is known as *Quantitative Trait Locus (QTL) mapping*. A QTL is a DNA region (*locus*, pl. *loci*), harboring a gene or a regulatory element affecting a quantitative trait. In a standard QTL mapping study, genetic data (*genotype* data) from an experimental population is used as input to a statistical model of the measured trait (*phenotype* data). The computation of the model fit and significance tests are performed using numerical algorithms implemented in a QTL mapping software. A review of QTL mapping methods is given in e.g. [14].

Finding the most likely positions of d QTL influencing a trait corresponds to minimization of a d-dimensional non-convex objective function (the *outer problem*) which is defined by the QTL model fit (the *inner problem*). In standard QTL mapping software [11, 12, 25, 30], the outer problem is solved using an exhaustive grid search. The computational requirement for this type of algorithm is $\mathcal{O}(d^2G^d)$, where the number of grid points G is of the order 10^3. This type of scheme is robust, but because of the exponential growth of work it is prohibitively slow for $d > 2$. This has resulted in that simultaneous searches for many QTL

have so far not been practical. However, a more efficient global optimization algorithm that enables such searches has recently been presented [27, 28].

In previous work [19–21] we have shown that multi-dimensional QTL searches can be efficiently implemented on different large-scale computational resources, including grid systems.

A standard procedure for grid-enabling application software is to develop a set of scripts, e.g. in python, that manages the grid tasks. Such a script can take care of the submission, gather the results and perform error handling, and also possibly include post-processing of the results.

In most cases, these scripts are not generic and have to be modified for different tasks. It is a problem that users of such grid software are distracted from the main research issues by having to deal with such routine tasks which can still be rather involved. It would be preferable to have a more user-friendly frontend for grid-enabled QTL mapping software. In particular, many biologists and geneticists do not have much experience of grid systems and scripting languages and would prefer a tool with a standard interface and a single point of deployment and access. Thus, for such users, a web based interface to grid resources is appropriate.

In this paper, we combine the efficient optimization scheme presented in [27, 28] with the parallelization techniques presented in [19–21] and implement the resulting tool using the LUNARC grid portal framework [26] and the more efficient and reliable grid job submission and monitoring within the *Grid Job Management Framework* (GJMF) [15, 17] as presented in [16]. This tool is especially useful for biologists and geneticists for performing standard analysis tasks. This system can also be used as a learning tool for both the underlying QTL application as well as understanding how application portals work. We also provide tools for exporting the final data for visualization, and show how this is useful for understanding the results.

2 Grid Computing and Grid Portals

Grid technology promises to change the way we tackle computational problems and use data. In the last decade, a number of research projects have been started with the goal of implementing grid computing. These include Globus [3] (one of the underlying environments for many grid software packagers), gLite [2] , NorduGrid ARC [8] and KnowARC [7]. The main concept behind the grid framework is the use of in-homogenous networks of commodity class computers and clusters for performing large scale computational tasks. This is especially valuable for institutions where funding for dedicated HPC hardware cannot be obtained easily.

The grid middlewares currently available are often used in large projects like the LHC (Large Hadron Collider) at CERN. Still, more effort is required to make grid systems practical for the general user community. One of the major areas that limit the use of grid systems is that infrastructure for application handling is missing.

So far, the general approach has been that researchers write job description scripts and job submission/management scripts for grid jobs. This diverts attention from the original task. Portal frameworks such as gridsphere [6] and GridBlocks [4] often provide almost all the functionality of the middleware. Since these portals are not specially designed for hosting grid applications, users need technical knowledge of object-oriented concepts and web development in order to extend the portal for their own applications.

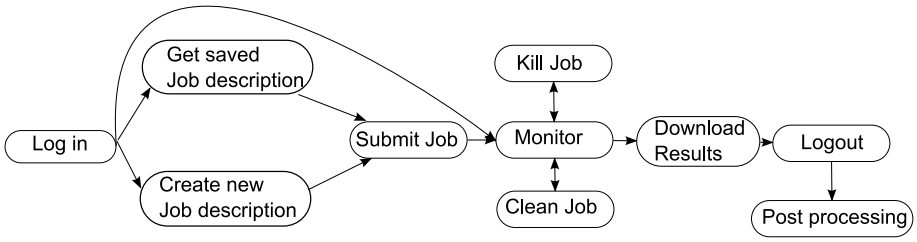

Fig. 1. QTL Application workflow

By having our QTL mapping code implemented within an application portal we make the grid related tasks transparent to the biologists. A workflow for a grid user of our code is given in Figure 1. Here a biologists would simply login to the application portal, choose the application (in this case the QTL application), fill in the necessary parameters and upload any data files and submit the computation. The more application specific parameters are discussed in section 4. The system will monitor the tasks and notify the user once the computation is completed. The user can then download the results. GridQTL [5] is a similar portal-based service which provides a grid enabled web interface to the QTL-express [30] software. Also, within the eQTL Infrastructure project [1], a grid portal for so called expression QTL analysis has been developed that uses the R/qtl software [11] for mapping of individual QTL.

3 The QTL Mapping Model

Assume that a model including d QTL is used and that a sequential map of the chromosomes and the genetic information within them is given. A position in the genome is identified by a number $x \in [0, G]$, where G is the total genome length. Let the vector $\mathbf{x} = [x_1 \quad x_2 \dots x_d]$ denote potential positions of the d QTL and let m be the number of individuals in the experimental population. Also, let k be the total number of parameters in the model and let the vector y contain the m phenotype observations, the vector b contain the k regression parameters and the vector ϵ contain the m noise components. A general QTL model can then be written as

$$y = A(\mathbf{x})b + \epsilon, \tag{1}$$

where the design matrix $A(\mathbf{x})$ is an $m \times k$ matrix of coefficients for fixed effects and QTL effects. Here, only the QTL effects depend on $\mathbf{x}$. The matrix A has one row per phenotype observations while the number of columns is given by $k = k_{fix} + k_{QTL}$.

Any set of d hypothetical QTL positions $\mathbf{x}$ can be used as input when building $A(\mathbf{x})$, but the genotypes of the individuals are (at best) only known at the marker loci. Several approaches can be used for estimating the regression parameters at loci between markers. One standard approach is maximum likelihood interval mapping [23, 24]. Another standard approach is the linear regression method [18, 29]. Since only a single, linear least-squares problem needs to be solved, this method is normally much less computationally expensive than if the maximum likelihood approach is used. When multiple QTL are included in the model, this becomes an important criterion. The linear regression model is also the approach used in the code used in our framework, which is a slightly updated version of the code that is presented in [27].

4 The QTL Mapping Portal: User Perspective

The original QTL mapping code is executed from the UNIX command line, and the input is given via a few text files. Some of these files are parameter files while the others contain data. The output generated is also as a text file.

In our first version of the web-based interface to the QTL mapping code the analysis is restricted to F_2 populations, but this restriction can easily be removed at a later stage. By using the functionality in the LUNARC application portal, the necessary phenotype, dataset, the number of QTL and the type of model can be set by the user in web form boxes. There are also several parameters that deal with the global and local search parameters that can be fine tuned by modifying default values in the same type of form boxes. These web forms are shown in Figures 2 and 3.

Using our tool, two types of global search can be performed. The first is the standard exhaustive search, which as has been mentioned above has been the method implemented in many of the QTL software due to its robustness. For multidimensional searches it has been shown in [27, 28] that the DIRECT algorithm [22] can be adopted for QTL searches to provide the results with the same robustness but with very much less computational demands, and our tool enables the use of this algorithm. In our previous work we have shown ways of implementing this scheme for demanding searches in parallel on grid systems. Using the parallelization settings in Figure 3 we enable partitioning of the search region and distribute the search across multiple nodes on a grid. This is possible for both the exhaustive and DIRECT search methods. Figures 4 and 5 show results presented by R scripts that perform post processing the results obtained from the portal. Figure 4 shows results of a four dimensional search using the 'DIRECT' search method while figure 5 shows results from another search performed using the exhaustive search algorithm. Figure 5 is only a subset

Fig. 2. Job description- Search parameters

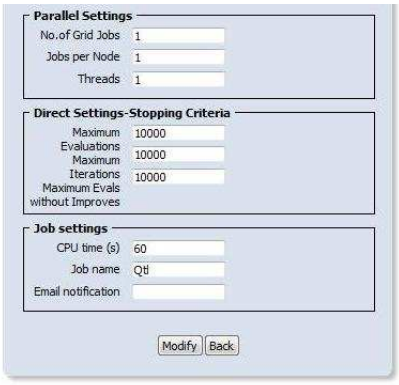

Fig. 3. Job description- Parallelization and Stopping Criteria

(a single partition, which runs on a specific node of the grid) of a two dimensional exhaustive search.

4.1 Randomization testing

Since QTL mapping involves multiple statistical tests throughout the genome, the selection of a significance threshold is a key issue of the procedure. Empirical estimation of the overall significance thresholds can be done in a wide range of population designs by re-sampling techniques such as randomization testing [13]. This is computationally quite expensive since 1000-10000 mapping problems must be solved, but with the use of grid systems this has been made feasible as the randomized tasks can be delegated to independent grid jobs. Using the parallelization settings given by the user in the web form in Figure 3 the number of randomization tasks and also the number of task sent to each node can be set. Figure 6 shows results of a randomization task after post-processing the results obtained from the portal. Here we use R to plot the different minima located by each of the randomized data in a probability density function. We can use other types of graphs or more advanced statistical analysis with the data also, since the scripts convert the datafiles generated from the portal into R readable format.

4.2 Visualization

The results generated from our QTL code is a plain text file with the different local and global minimum values with certain other data relating to their location. This text file is parsed for statistical post processing via R [9]. For some users, locating the global minima and its location may be sufficient. While for others it may be interesting to look at the distribution of the data.

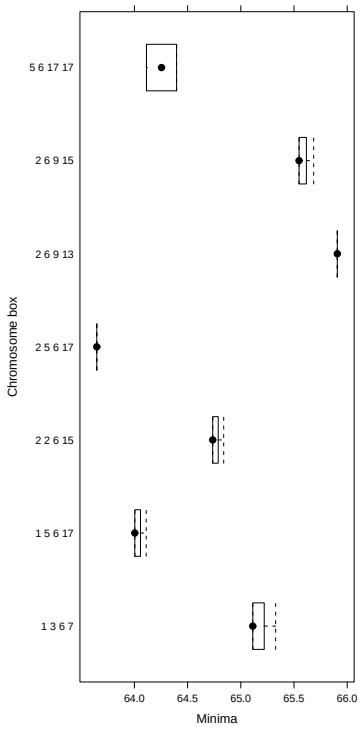

Fig. 4. Results-DIRECT Search

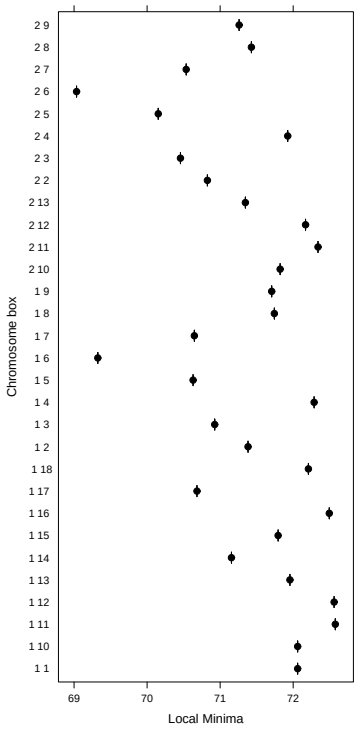

Fig. 5. Exhaustive Search Results

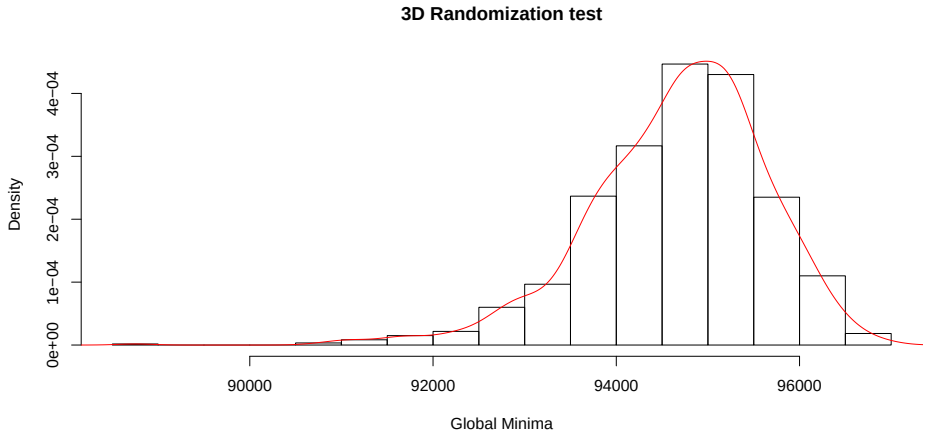

Fig. 6. 3D Randomization test

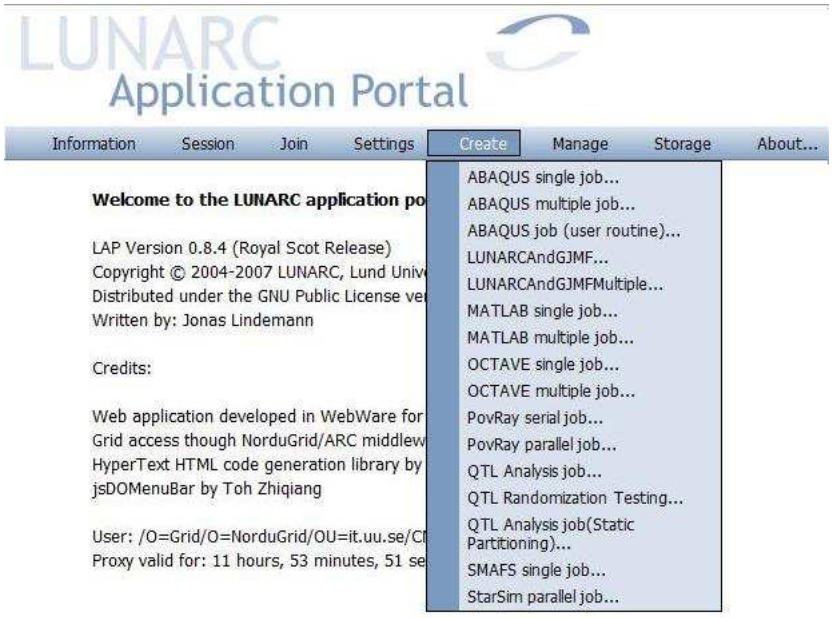

Fig. 7. The different Lunarc grid jobs available

Also when performing randomization tests it may be useful to have the results in some form of graphics. To enable this we have developed several scripts, though not integrated into the portal, these can be used for post processing of the results. Once the results are downloaded onto the local machine, these scripts can be used to transform the data in a R [9] compatible data file format. R is a open source statistical software that is widely used by biologist and others for data analysis and visualization.

5 The QTL Mapping Portal: Implementation

For the QTL application presented in this paper, we use the open source Lunarc Application Portal (LAP). The LAP project [26] is an effort to provide an application oriented web based environment, providing targeted user interfaces for commonly available applications. The portal currently provides user interfaces for applications such as MATLAB, OCTAVE, ABAQUS (Structural analysis) and MOLCAS (Computational chemistry) as given in Figure 7.

The portal can also be viewed as a python-based framework for implementing web interfaces to user applications. Additional user interfaces are added as plugins to an installed portal instance. We have added our own separate plugin to the portal for the QTL application. As the plugin code is python-based it makes the implementation, modification and maintenance of the code more efficient.

The implementation goals of the LAP framework have been:

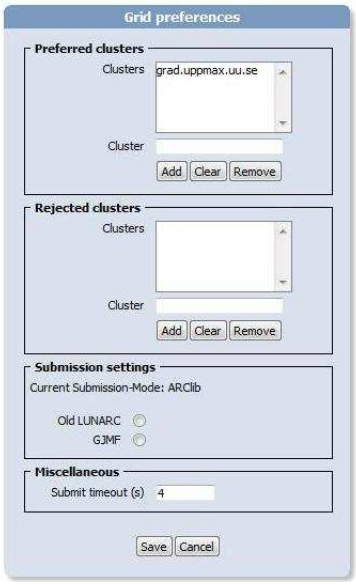

Fig. 8. Grid Preferences

- **Lightweight** – Easy to understand without large dependencies on other libraries. Easy to deploy and maintain.
- **Extendible** – It should be easy to extend the portal using a built in plugin-architecture.
- **Customizable** – The graphical design should be customizable to adapt to existing web designs.

The portal framework is implemented using the python web application framework Webware. This is a lightweight framework for developing object-oriented web applications. The framework contains design patterns for applications servers, server pages, servlets, session management and many other features. The framework is modular and easily extendable.

The portal application server is integrated with the Apache webserver using a special Apache module, mod_webkit provided with Webware. For security reasons the Apache webserver serves the web pages using the HTTPS protocol.

The grid resource that we have utilized is the Swegrid system [10]. As given in Figure 8 we can add the different grid resources that we would like the system to submit our jobs to. The portal can be utilized to submit the jobs to other grid resources as well via the use of translation services.

The underlying QTL mapping application code is implemented in C, but is not built into the grid portal, and the application code can be updated and replaced seamlessly without any action by the biologists.

This layered structure of the system as seen in Figure 9 has also provides us the opportunity to integrate a more robust submission system while still

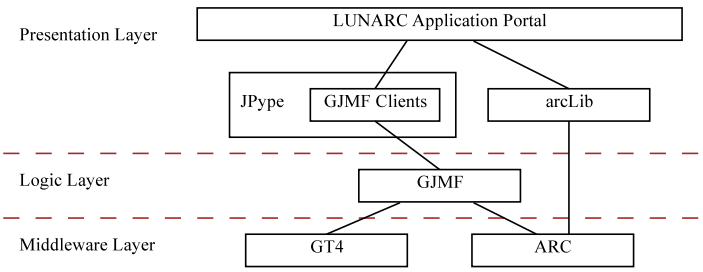

Fig. 9. The LAP - GJMF Integration

maintaining the original interfaces. With the integration of GJMF [15] the portal provides a more reliable job model. This is essential, as even though the biologist sees only one computation being executed in reality this work may be composed of a multitude of grid jobs.

We internally split the data files and partition the global search across multiple nodes on the grid. Each of these jobs have to be monitored, and the failure of a single job negates the total search task. A robust submission which can handle job failures is thus preferred. This has been fulfilled via the integration of GJMF with the portal [16]. The Grid Job Management Framework (GJMF), implemented using Java and Globus as a service oriented architecture (SOA), is a toolkit for automating some of the job management tasks in Grid environments. Using the grid preference page as given in Figure 8 we can select which submission method we would prefer. GJMF also makes it possible for the integration of multiple grid systems where the jobs can be submitted. It will handle the translation of the job description files and other details without having to make any modifications to the portal. Also it helps to reduce the work load of the node running the application portal by taking over the job management task, since the GJMF service does not need to run on the same machine or even the same cluster.

6 Conclusion

The QTL analysis portal makes it possible for biologists and students to perform multi-dimensional QTL scans within reasonable time via the use of grid resources. This portal provides a user-friendly interface that does not distract the user from the analysis task by having to manage the grid jobs. The only tool the user needs now is a standard web browser; there is no need for installation of any external tools. The user is also able to use any computer with any OS and platform as an access node.

The portal also makes it possible to perform randomization testing using grid resources and to make use of R for visualization of the results in a post-processing step.

The use of the robust job submission system GJMF has made it possible to increase the reliability of the system and to reduce the job failure rates. We anticipate that this system can be used by biologist as a production tool for QTL analysis.

References

1. eQTL Project. http://eqtl.berlios.de.
2. gLite. http://glite.web.cern.ch/glite.
3. Globus. http://www.globus.org.
4. GridBlocks. http://sourceforge.net/projects/gridblocks/.
5. The GridQTL project. www.gridqtl.org.uk/.
6. gridsphere. http://www.gridsphere.org.
7. KnowARC. http://www.knowarc.eu.
8. Nordugrid project. http://www.nordugrid.org.
9. The R project for statistical computing. http://www.r-project.org.
10. Swegrid. http://www.swegrid.se.
11. C. Basten, B. Weir, and Z.-B. Zeng. *QTL Cartographer, version 1.15*. Dept. of Statistics, North Carolina State University, Raleigh, NC, 2001.
12. K.W. Broman, H. Wu, S. Sen, and G.A. Churchill. R/qtl: QTL mapping in experimental crosses. *Bioinformatics*, 19(7):889–890, 2003.
13. G. Churchill and R. Doege. Empherical threshold values for quantitative trait mapping. *Genetics*, 138:963–971, 1994.
14. R.W. Doerge. Mapping and analysis of quantitative trait loci in experimental populations. *Nature reviews-Genetics*, 3:43–52, 2002.
15. E. Elmroth, P. Gardfjäll, A. Norberg, J. Tordsson, and P-O. Östberg. Designing general, composable, and middleware-independent grid infrastructure tools for multi-tiered job management. In T. Priol and M. Vaneschi, editors, *Towards Next Generation Grids*, pages 175–184. Springer-Verlag, 2007.
16. E. Elmroth, S. Holmgren, J. Lindemann, S. Toor, and P-O. Östberg. Empowering a flexible application portal with a SOA-based grid job management framework. Accepted for publication in proc. PARA'08, May 13-16, Trondheim, Norway, 2008.
17. E. Elmroth and P-O. Östberg. GJMF - A Service-Oriented Grid Job Management Framework. Submitted for journal publication, 2009.
18. C.S. Haley and S.A. Knott. A simple regression method for mapping quantitative trait loci in line crosses using flanking markers. *Heredity*, 69:315–324, 1992.
19. M. Jayawardena and S. Holmgren. Grid-enabling an efficient algorithm for demanding global optimization problems in genetic analysis. In *3rd IEEE International Conference on e-Science and Grid Computing, IEEE Conference Proceedings 10.1109*, pages 205–212. IEEE, 2007.
20. M. Jayawardena, S. Holmgren, and K. Ljungberg. Using parallel computing and grid systems for genetic mapping of multifactorial traits. In B. Kågström, E. Elmroth, J. Dongarra, and J. Waśniewski, editors, *Proc. PARA 2006, Applied Parallel Computing: State of the Art in Scientific Computing, Lecture Notes in Comp. Sci*, pages 627–636. Springer-Verlag, 2007.
21. M. Jayawardena, S. Holmgren, and H. Löf. Efficient optimization algorithms and implementations for genetic analysis of complex traits on a grid system with multicore nodes. Accepted for publication in proc. PARA'08, May 13-16, Trondheim, Norway, 2008.

22. D. Jones, C. Perttunen, and B. Stuckman. Lipschitzian optimization without the lipschitz constant. *J. Optimization Theory App*, 79:157–181, 1993.
23. C.-H. Kao, Z.-B. Zeng, and R. Teasdale. Multiple interval mapping for quantitative trait loci. *Genetics*, 152:1203–1216, 1999.
24. E. Lander and D. Botstein. Mapping mendelian factors underlying quantitative traits using RFLP linkage maps. *Genetics*, 121:185–199, 1989.
25. S. Lincoln, M. Daly, and E. Lander. Mapping genes controlling quantitative traits with MAPMAKER/QTL1.1. Technical report 2nd edition, Whitehead Institute, 1992.
26. J. Lindemann and G. Sandberg. *Advances in Grid Computing - EGC 2005*, chapter An Extendable GRID Application Portal, pages 1012–1021. Springer Berlin / Heidelberg, 2005.
27. K. Ljungberg, S. Holmgren, and Ö. Carlborg. Simultaneous search for multiple QTL using the global optimization algorithm DIRECT. *Bioinformatics*, 20:1887–1895, 2004.
28. K. Ljungberg, K. Mishshenko, and S. Holmgren. Efficient algorithms for multidimensional global optimization in genetic mapping of complex traits. *Journal of Advances and Applications in Bioinformatics and Chemistry*. (Accepted for publication).
29. O. Martinez and R. Curnow. Estimating the location and the sizes of effects of quantitative trait loci flanking markers. *Theor Appli Genet*, 85:480–488, 1992.
30. G. Seaton, C. Haley, S. Knott, M. Kearsey, and P. Visscher. QTL express: mapping quantitative trait loci in simple and complex pedigrees. *Bioinformatics*, 18:339–340, 2002.