Grid-Enabling an Efficient Algorithm for Demanding Global Optimization Problems in Genetic Analysis

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

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

Abstract. We study the implementation on grid systems of an efficient algorithm for demanding global optimization problems. Specifically, we consider problems arising in the genetic mapping of quantitative trait loci (QTL), i.e. locations in the genome that affect a trait that is measured on a continuous scale. The scheme used in standard serial codes for QTL-analysis is easy to parallelize, and we have earlier shown that it is highly suitable for implementation on grid systems when searching for a limited number of genetic loci. However, if this type of algorithm is employed for searching for larger numbers of loci the number of evaluations of the statistical model becomes overwhelming, making the computations infeasible using any type of computer system. In this paper, we instead consider the DIRECT algorithm, which has been shown to require much fewer objective function evaluations for high-dimensional problems. By removing selected parts of the frequent global communication and synchronization in the existing parallel versions of this algorithm, we show that it is possible to retain the efficiency of the original DIRECT algorithm and perform searches for QTL models involving many loci using grid systems.

1 Introduction

Some large and demanding computational problems can be efficiently solved using grid environments. Researchers in high-energy physics were early users of grids, performing Monte Carlo simulations and data analysis, but scientists in many other application fields are also currently starting to use grid-based algorithms and software.

To be suitable for implementation on grid systems, an algorithm must be pleasantly parallel in the sense that the main part of the work can be subdivided into sufficiently large chunks that can be executed in a highly independent and asynchronous way, but where possibly some local/serial preprocessing, synchronization at a few workflow stages, and/or some local/serial postprocessing may also be included.

However, for many demanding computational problems, the state-of-the art algorithms include a significant amount of inherent communication and/or synchronization. An alternative, pleasantly parallel algorithm may in principle exist, but often it results

in an overwhelming total computational complexity and the computations can not be performed using any type computational system.

Given that a parallel algorithm which originally is not suitable for grid environments is the only practical choice, grid systems still provide structured and cost-efficient access to powerful computational resources. Hence, it is an interesting challenge to attempt to introduce new versions of important non-pleasantly parallel algorithms where the global coupling and need for synchronization is removed at some level, while the efficiency of the original algorithm is still retained to a sufficiently high degree. Such new versions of the algorithms may form the basis of grid-enabled software for new application problem classes.

In this paper, we consider demanding global optimization problems arising in genetic analysis of complex traits. Global optimization is an example of a problem class where pleasantly parallel algorithms do exist; the most striking example is the exhaustive search scheme, where the objective function is independently evaluated at all points in a fine mesh in the search space. For large-scale problems, such an algorithm can easily be adopted for parallel computing on e.g. grid systems. We have also used this type of algorithm to solve some global optimization problems in genetic analysis using a grid system [16]. However, since the number of objective function evaluations performed by an exhaustive search algorithm grows very rapidly with the number of dimensions in the optimization problem, high-dimensional problems can not be tackled even if extreme computing resources would be available in the grid.

Alternative, potentially more efficient, global optimization algorithms involve more frequent and synchronized exchange of information about the progress of the optimization. One example is the DIRECT scheme [17], which has been found to be effective for problems arising in a number of application fields. A serial version of DIRECT has been developed and successfully applied to the computational genetics application [21, 22], but for high-dimensional problems the single-processor execution time still makes the analysis impractical even when using a basic statistical model. The inherent global communication and frequent synchronization in DIRECT results in that the algorithm is not easily parallelized. The parallel DIRECT scheme described in [5] is developed for more tightly coupled parallel computer systems using highly synchronized scheduling of processes. Hence, to be able to employ grid computing, a modified version of DIRECT is needed where some of the communication and synchronization is removed but where the serial efficiency of the original scheme is still retained to a sufficient degree. The purpose of this paper is to describe such an algorithm, to apply it to large-scale problems arising in the computational genetics application, and to show that the new scheme enables the geneticists to perform analysis of a type that has not been possible to tackle in a production environment before.

The structure of this paper is as follows: In Section 2, we give a brief introduction to the application field, and in Section 3 we describe the features of the global optimization problem. In Section 4, we introduce the DIRECT algorithm, and in Section 5 we describe our new parallelized version and how it is implemented on a grid system. After a brief description in Section 6 of the systems used, we in Section 7 present

experimental results for the new DIRECT scheme applied to typical problems in the application area. Finally, we draw some conclusions in Section 8

2 Genetic analysis of quantitative traits

Traits that vary continuously are called quantitative; examples of such traits are body weight and susceptibility to infections and diseases like diabetes in humans, growth rate and e.g. egg and milk production for farm animals, and crop yield for plants. In general, quantitative traits are affected by an interplay between multiple genetic factors and the environment. As indicated by the examples, most economically and medically important traits in humans, animals and plants are quantitative, and understanding the genetics behind them is potentially of great importance.

For a quantitative trait, the effect of segregation of alleles of one gene is at least to some degree concealed by the effect of other genes and environmental effects. Thus, individuals with identical genotypes may exhibit different phenotypes, and the dissection of quantitative traits is performed by statistical analysis of genetic and phenotypic data for a large population of individuals. The aim of such an analysis is to reveal *quantitative trait loci* (QTL), which are regions in the genome that affect a quantitative trait. Then, analysis at the molecular level can be used to perform a more detailed study of these genetic regions. Reviews of the field of QTL analysis are given e.g. in [12, 24].

In this paper, we employ a standard approach for QTL analysis based on interval mapping and a linear regression model for relating the genetic and phenotypic data [15, 18, 25]. 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 (normally measured in the unit *centimorgan*, cM). Let the vector $\mathbf{x} = [x_1 \quad x_2 \ldots x_d]$ denote potential positions of the d QTL. Also, let n be the number of individuals in the population, $\mathbf{y}$ the vector of n phenotype observations, $k \geq d$ the total number of parameters in the model and $\mathbf{b}$ the vector of k effects. A general linear QTL model can then be formulated in matrix form,

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

where the matrix $A(\mathbf{x})$ is the $n \times k$ design matrix and ϵ is the error vector. Given this model, any set of d hypothetical QTL positions $\mathbf{x}$ can be used as input when building the design matrix $A(\mathbf{x})$. The goal of a QTL search is to optimize the model fit over all possible positions $\mathbf{x}$ and to compute the corresponding residual sum of squares, RSS_{opt}, according to

$$RSS_{opt} = \min_{\mathbf{b},\mathbf{x}} (A(\mathbf{x})\mathbf{b} - \mathbf{y})^T (A(\mathbf{x})\mathbf{b} - \mathbf{y}), \tag{2}$$

The location $\mathbf{x}$ that minimizes (2) is denoted by $\mathbf{x}_{opt}$, and is the most probable set of QTL positions for the given model. The expression (2) is a separable non-linear least-squares problem where the model is a linear combination of non-linear functions. The

solution of (2) can be separated into two parts: The inner, linear problem,

$$RSS(\mathbf{x}) = \min_{\mathbf{b}} (A(\mathbf{x})\mathbf{b} - \mathbf{y})^T (A(\mathbf{x})\mathbf{b} - \mathbf{y}), \tag{3}$$

is referred to as evaluation of the objective, or kernel, function. If another more involved statistical model is employed, the objective function will be different and normally more expensive to compute. The outer, non-linear problem,

$$\min_{\mathbf{x}} RSS(\mathbf{x}), \tag{4}$$

is referred to as the global search problem and is in general independent of the choice of statistical model. In Figure 1, a surface plot of a part of a typical objective function in (4) for a problem where $d = 2$ is shown.

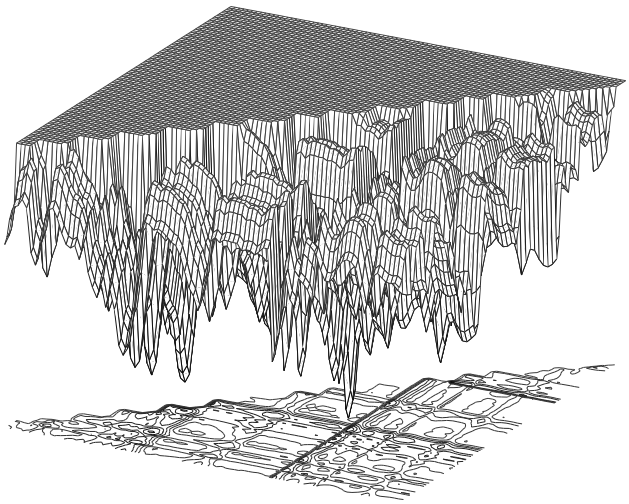

Fig. 1. Part of a typical optimization landscape for a model where $d = 2$

Note that there will always exist a value $\mathbf{x}_{opt}$ that optimizes (4), but the question remains if it can be established that these positions corresponds to true QTLs. It is necessary to compare models with different number of QTL and different sets of parameters and to perform significance tests for each model configuration. The significance tests are often performed by comparing RSS_{opt} to an empirically computed threshold using permutation tests [11, 13].

3 The global search problem in QTL analysis

It is generally believed that quantitative traits are governed by an interplay between multiple QTL and environmental factors. Using a model including d QTL implies that the global search problem is d-dimensional. So far, standard QTL mapping software [6,7,19,26] is not parallelized and uses an exhaustive search algorithm for solving the global search problem. This type of algorithm is robust, but the computational requirement grows very rapidly with d. This results in that currently only mapping of a single QTL ($d = 1$) can be easily performed using these codes. Models with multiple QTL are still regularly fitted, but then by using a forward selection technique where a sequence of one-dimensional exhaustive searches are performed. In this type of procedure, the identified QTL are successively included as known quantities when searching for additional QTL. However, it is not clear how accurate this technique is for general QTL models, and lately the interest in simultaneous mapping of multiple QTL, i.e. solving the original d-dimensional global search problem, has increased. Partly, the interest is motivated by analysis of real data sets [10,27,28] where certain interactions [9] between pairs of QTL have been found to only be detectable by solving the full two-dimensional optimization problem. Hence, solving the multi-dimensional global search problem in QTL analysis can potentially lead to new and interesting results in genetics of quantitative traits. However, for high-dimensional models to be tested and used in practice, more efficient algorithms and implementations are needed.

Previously used alternative optimization methods for QTL mapping problems include a genetic optimization algorithm, implemented for $d = 2$ using library routines [8], and a serial algorithm based on the DIRECT [17] scheme, implemented for $d = 1, \ldots 6$ [21] and later improved to include a more efficient local search [22]. For multi-dimensional QTL searches, the DIRECT-based optimization algorithms are many orders of magnitude faster than exhaustive search. However, as a first step towards a grid-enabled tool for QTL analysis, an exhaustive search algorithm was implemented in [16], were 3-dimensional searches were performed using the SweGrid [1] system.

The search space for the global search problem is in principle a d-dimensional hypercube, where the side is given by G, the size of the genome. The genome is organized in c chromosomes, which provides an implicit partitioning of the search space into a set of c^d d-dimensional unequally sized *chromosome combination boxes*, cc-boxes. Each cc-box can be identified by a vector of d chromosome numbers. The ordering of the loci does not affect the model fit, and by exploiting this symmetry the search space can be reduced by restricting the search to cc-boxes identified by non-decreasing sequences of chromosomes.

4 The DIRECT algorithm

The original DIRECT algorithms [17] was derived as a general tool for finding the global minimum of a multivariate function over a domain with simple bounds. DI-

RECT is an iterative, deterministic algorithm based on a standard Lipschitz optimization approach, but where the Lipschitz constant is treated as an unknown parameter used in the process to decide whether local or global search should be given priority. The name DIRECT is short-hand for the phrase *DIviding RECTangles*, which gives some indication on how the algorithm works; it recursively identifies 'potentially optimal' hyperboxes by evaluating the objective function in the center of a set of boxes, and then divides these into smaller boxes. The original algorithm is described in [17], and a pseudo-code version can be found in [5].

If run for a long time, DIRECT will perform an exhaustive search over a uniform mesh defined by a resolution parameter. For optimization problems with a Lipschitz-continuous objective function, it can be proved that DIRECT will eventually find a global optimum up to some predefined accuracy. However, as for other iterative schemes for global optimization, no well-founded, practical stopping criterion is available for general problems. For the QTL mapping application studied in [21,22], a slightly modified DIRECT procedure is used, and it is executed until a predefined number N_f of objective function evaluations have been performed without any further improvement of the global minimum. For a model with d QTL, exploiting the symmetry described above results in that the size of the search space is $G^d/d!$. This motivates that the convergence parameter is chosen to have the form $N_f = (P_{alg}G)^d/d!$, where the number P_{alg} is determined by performing a large number of numerical experiments for each algorithm, adjusting P_{alg} so that the global optimum is located for all the tested data sets. In Table 1, the values of N_f as determined in [22] are given. For the global search problems, it was observed in [22] that the global optimum is often rather rapidly identified and the total number of function evaluations performed by the DIRECT algorithm is typically not much larger than N_f. Only in a few cases up to a magnitude more evaluations are actually performed. For comparison, Table 1 also shows the number of objective function evaluations when using exhaustive search, N_{exh}, exploiting a mesh with standard resolution.

Table 1. The values of N_f and N_{exh} for the mouse data set described in Section 7

d	# of cc-boxes	N_f	N_{exh}
2	171	841	2.645×10^6
3	1140	11487	2.089×10^9
4	5985	1.177×10^5	1.260×10^{12}
5	26334	9.655×10^6	0.612×10^{15}

5 Parallelization of DIRECT on a grid system

We now describe a modified version of the DIRECT algorithm applied to the global search problem in QTL analysis. The scheme can easily be used also for other problem

classes, and it is suitable for grid systems and other loosely-connected computer configurations. We also describe an implementation that can been adopted to most computer environments.

In the parallel DIRECT implementation described in [5], an MPI-based master-slave model is used, where the function evaluations are distributed among a pool of slave nodes while the global data structure containing the information about the hyper-boxes considered is maintained at the master node. This retains the original form (and hence serial efficiency) of the algorithm, but using such a scheme would incur too much communication and synchronization in a grid environment. For such systems, the work must be divided in more independent chunks, and we are forced to modify the form, and hence the serial efficiency, of the algorithm.

For the global search problems, the implicit partitioning of the search space into cc-boxes provides a natural basis for introducing a pleasantly parallel level in the DIRECT algorithm. The objective function (3) is continuous within a cc-box, but it is normally discontinuous at the cc-box boundaries. In the serial version of the algorithm for QTL analysis, the center of each cc-box is sampled when the DIRECT algorithm is initiated. The reason is that then the Lipschitz continuity condition in DIRECT is fulfilled, since in the convergence result the Lipschitz constant is used for bounding the objective function (3) only within boxes.

Using this cc-box initialization implies that we in principle could submit one independent job for each cc-box, each performing DIRECT optimization locally. However, for high-dimensional searches this would result in a very large number of jobs where the size of the local problems would vary a lot. Instead, we partition the global search space into p *sets* of cc-boxes, where p is a user-determined parameter. Each chromosome is of different size, and we do not a priori know how the sampling of search space points is performed by DIRECT. Hence, we distribute the cc-boxes over the sets using a block-cyclic distribution in an attempt to achieve good load balance. In Section 7 we present numerical experiments where the speedup and load balance for global search problems in QTL analysis is determined. For global optimization problems arising in other applications than QTL analysis, either some other implicit partitioning given by the application can be used instead of the cc-boxes, or an artificial blocking of the search space can be introduced for creating the p sets of local domains.

The new version of DIRECT can be described by the following pseudo-code:

```
Partition the cc-boxes into p sets;
do (in parallel) i=1:p
   l_sol(i) = DIRECT(cc-box-set(i));
end
Find the global solution among l_sol(:);
```

The cc-box sets are smaller than the total search space, and the form of N_f indicates that it might be possible to reduce the number of function evaluations needed for the

convergence test in the local searches. We use

$$N_f(p) = N_f \cdot a^{\log_2(p)}, \tag{5}$$

where $a \in [0.5, 1]$ is a parameter that has to be determined by numerical experiments for the problem type at hand. (note that N_f also needs to be determined using experiments, so these have to be performed anyway). The value $a = 0.5$ corresponds to the natural choice of halving the number of iterations in the convergence test when the size of the search space is halved, but a might have to be chosen larger than 0.5 to guarantee that a global optimum is found for all values of p used. This is an effect of that we have modified the original DIRECT algorithm by reducing the global coupling. It it is not a priori clear what effect this has on the performance in terms of the number of objective function evaluations. Again, we perform numerical experiments in Section 7 where we examine this for global search problems in QTL analysis.

The implementation of the parallel DIRECT algorithm described above is based on an existing serial C code [23]. The only modification needed is to enable the search to be performed over a set of cc-boxes instead of the whole search space. The parallel execution is implemented as a two step process. First, a small separate code is used to partition the global search space into sets of cc-boxes. The partitioning code stores the cc-box data on a separate file for each set, and these files are then used as input for the DIRECT code run in each of the p computational jobs. A job manager script runs the partitioning code and submits the DIRECT jobs together with the description files, waits for all the jobs to finish, gets the local results, and finally computes the final output. This type of job manager scripts can easily be produced for different computational environments, and we have currently developed versions for all three computer systems mentioned in the next section. Currently, the grid version does not include any handling of job failures, but we plan to use the robust multiple-job submission service described in [14] once it becomes available.

6 Computer systems

In the experiments presented in the next section we have used the Swegrid [1] system. The computational resources in SweGrid currently consist of six clusters distributed over the six national HPC centers in Sweden, connected via the 10 Gbit SUNET national network. Each cluster has 100 nodes, where each node is equipped with a 2.8 GHz Intel P4 processor and 2 GB RAM. Within the clusters, the nodes are interconnected by standard gigabit Ethernet. The clusters run different versions of Linux, and the grid middleware ARC [2] is currently used for submitting and scheduling grid jobs. Within the clusters, the jobs are scheduled using a standard queuing system, e.g. PBS. The ARC jobs are specified in XRSL-files, and our job submission script sets these up for the local DIRECT searches. For the comparisons presented later we have also created another version of the job submission script, using the GridEngine N1 queuing system. This script has been used for the SunFire 15k system Ngorongoro, which is

56-core shared-memory (SMP) server, and the cluster Isis, which is a 200-node, 800-core AMD Opteron cluster. These two systems are located at the Uppsala University HPC center UPPMAX [3].

7 Results

In this section we present results for QTL searches using models where $d = 3, 4$ and 5. The code can perform searches for $d = 1, 2$ and $d \geq 6$ as well. However, for one- and two-dimensional problems, the runtimes are so small so that no parallelization is needed, and for $d \geq 6$, the serial runtime becomes so large so that it is no longer practical to perform speedup experiments.

We have performed analysis for two data sets. The first consists of simulated data for an F_2 population consisting of 500 mice and is further described in [20]. The other data set consists of experimental data from an F_2 cross of two outbred lines of wild boars and domestic pigs with 191 individuals and is further described in [4]. All the studied traits are growth related, but our aim is to observe the behavior of the new version of the DIRECT algorithm and we do not attempt to draw any implications to genetics from the computations. Also, neither the statistical model nor the statistical significance has been evaluated for our test runs. However, the experiments carried out below can form the basis for serious work in genetics in the future.

7.1 Mouse data (simulated)

In Tables 2, 3 and 4, we present results for analysis of the mouse data set using models where $d = 3, 4$ and 5, respectively. The tables show the number of jobs used (p), the maximal wall clock time (in seconds) for one of the jobs on both the SweGrid and SunFire systems (T_{grid} and T_{SMP}), the maximal number of function evaluations for one of the jobs (N_{max}), and the load balance, defined as N_{max}/N_{avg}, where N_{avg} is the average number of function evaluations. Note that for the SweGrid system, the timings are only accurate up to a second, and the values for $d = 3$ are not very informative.

For this data set, we found that it is possible to fully reduce the parameter $N_f(p)$ according to the reduction in the size of the search space for the local searches, i.e., $a = 0.5$ was chosen in (5).

In Figure 2, we show the speedup for the analysis when using the SweGrid system, and in Figure 3 we show the corresponding values when the SunFire 15k system is used. From these figures, it is clear that the speedup is similar for the two types of systems. The algorithm does not make any use of the tightly interconnected shared memory in the SunFire system for communication between the jobs, and the differences in speedup can be attributed to differences in architectures of the processors and cache/memory systems which results in different performance behavior as the size of the local problems is reduced.

A striking feature observed in Figures 2 and 3 is that for many problems, the speedup is actually superlinear. Even though the load is not perfectly balanced over

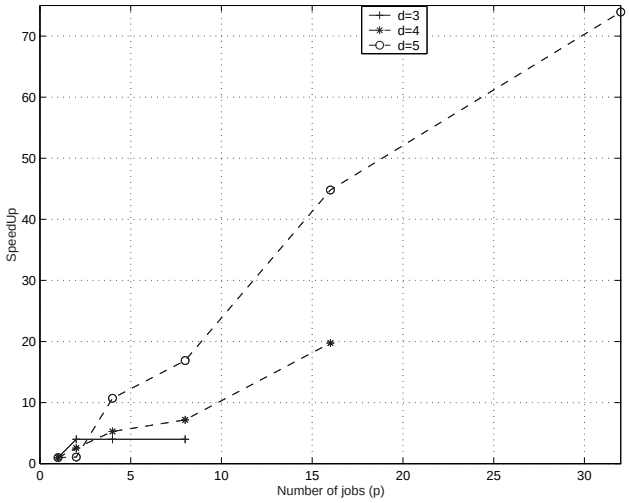

Fig. 2. Speedup for the analysis of mouse data when using SweGrid

the different jobs, we get significant speedup also for relatively large values of p. The reason for this is that first the minimum is located faster in the local searches (the search space is smaller), and then a smaller number of iterations are also performed in the convergence test. This type of result is a bit surprising, since it implies that for many global search problems in QTL analysis, the efficiency of DIRECT is *improved* by removing some of the global exchange of information. Further experiments are needed to examine if this property is shared by global optimization problems from other applications.

7.2 Pig data (experimental)

For this experimental data set, the objective function is noisy and has a very large number of local minima. In this case we found that for some combinations of d and p, the new version of the DIRECT scheme does not locate the correct global minimum if the "optimal" value $a = 0.5$ is used when computing $N_f(p)$ from formula (5). Instead we use $a = 0.75$, which corresponds to that the number of function evaluations in the stopping criterion is reduced by a factor of 0.75 when the size of the search domain is reduced by a factor of 0.5. In Figures 4 and 5 we show the speedup obtained using the SweGrid and Isis systems for the analysis of the pig data set.

From the figures, it is clear that also in this case the speedup archived using the grid system is comparable to a more tightly coupled system such as the Isis cluster. Again, the differences can be attributed to differences in architectures of the processors and cache/memory systems and not to the type of communication network used. Also, the results in Figures 4 and 5 show that, even though we had to increase the value of the

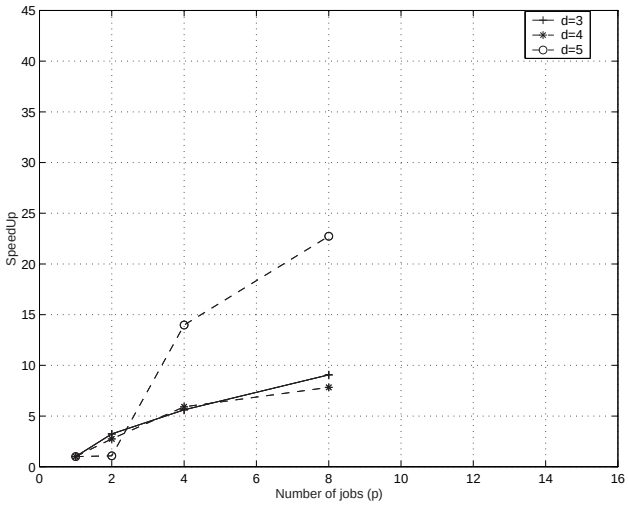

Fig. 3. Speedup for the analysis of mouse data when using SunFire 15k

parameter a in the stopping criterion to guarantee that the correct optimum was found, this did not result in a very large degradation in the speedup achieved compared to the mouse data set, where $a = 0.5$ could be used.

8 Conclusions

In this paper, we have developed a new, parallel version of the DIRECT scheme for global optimization and show how it can be implemented on most computer systems, including grids. We apply the new scheme to the global search problem in QTL analysis and show that we can solve this problem for high-dimensional statistical models efficiently using a grid system. This grid-enabled software provides a tool for geneticists to perform new types of analysis of quantitative traits, potentially leading to new interesting results in genetics.

References

1. http://www.swegrid.se.
2. http://www.nordugrid.org.
3. http://www.uppmax.uu.se.
4. L. Andersson, C. Haley, H. Ellegren, S. Knott, M. Johansson, K. Andersson, L. Andersson-Eklund, I. Edfors-Lilja, M. Fredholm, and I. Hansson. Genetic mapping of quantitative trait loci for growth and fatness in pigs. *Science*, 263:1771–1774, 1994.

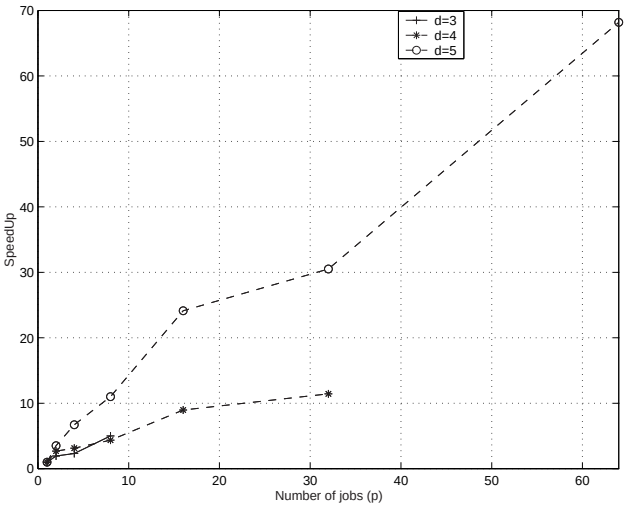

Fig. 4. Speedup for the analysis of pig data when using SweGrid

5. C.A Baker, L.T Watson, B Grossman, R.T Haftka, and W.H Mason. Parallel global aircraft configuration design space exploration. In *High performance symposium 2000*, pages 101–106, 2000.
6. C. Basten, B. Weir, and Z.-B. Zeng. *QTL Cartographer, Version 1.15*. Department of Statistics, North Carolina State University, Raleigh, NC, 2001.
7. K. Broman, H. Wu, S. Sen, and G. Churchill. R/qtl: QTL mapping in experimental crosses. *Bioinformatics*, 19:889–890, 2003.
8. Ö. Carlborg, L. Andersson, and B. Kinghorn. The use of a genetic algorithm for simultaneous mapping of multiple interacting quantitative trait loci. *Genetics*, 155:2003–2010, 2000.
9. Ö. Carlborg and C.S. Haley. Epistasis: too often neglected in complex trait studies? *Nature Reviews Genetics*, 5:618–625, 2004.
10. Ö. Carlborg, S. Kerje, K. Schütz, L. Jacobsson, P. Jensen, and L. Andersson. A global search reveals epistatic interaction between QTL for early growth in the chicken. *Genome Research*, 13:413–421, 2003.
11. G. Churchill and R. Doerge. Empirical threshold values for quantitative trait mapping. *Genetics*, 138:963–971, 1994.
12. R. Doerge. Mapping and analysis of quantitative trait loci in experimental populations. *Nature Reviews Genetics*, 3:43–52, 2002.
13. R. Doerge and G. Churchill. Permutation tests for multiple loci affecting a quantitative character. *Genetics*, 142:285–294, 1996.
14. 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 *Proceedings of CoreGrid symposium 2007*. Springer Verlag, LNCS series. In press.

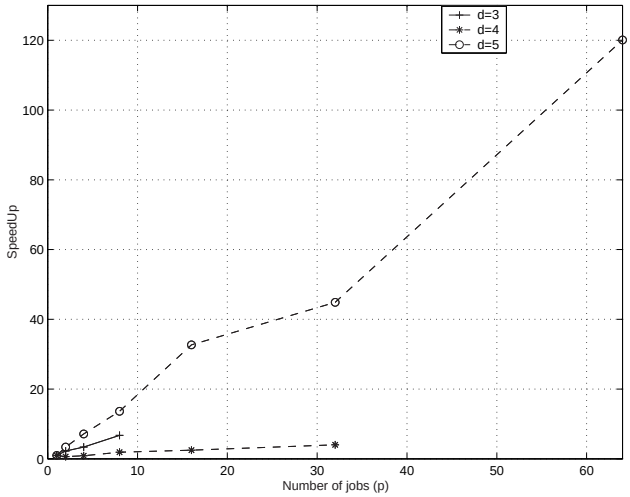

Fig. 5. Speedup for the analysis of pig data when using the cluster Isis

15. C. Haley and S. Knott. A simple regression method for mapping quantitative trait loci in line crosses using flanking markers. *Heredity*, 69:315–324, 1992.
16. M. Jayawardena, S. Holmgren, and K. Ljungberg. Using parallel computing and grid systems for genetic mapping of quantitative traits. In *Proceedings of PARA06*. Springer Verlag, LNCS series. In press.
17. D. Jones, C. Perttunen, and B. Stuckman. Lipschitzian optimization without the Lipschitz constant. *Journal of Optimization Theory and Application*, 79:157–181, 1993.
18. S. Knapp, W. Bridges, and D. Birkes. Mapping quantitative trait loci using molecular marker linkage maps. *Theoretical and Applied Genetics*, 79:583–592, 1990.
19. S. Lincoln, M. Daly, and E. Lander. Mapping genes controlling quantitative traits with MAPMAKER/QTL 1.1. Technical report, Whitehead Institute, 1992. Technical report. 2nd edition.
20. K. Ljungberg. Efficient evaluation of the residual sum of squares for quantitative trait locus models in the case of complete marker genotype information. Technical Report 2005-033, Division of Scientific Computing, Department of Information Technology, Uppsala University, 2005.
21. K. Ljungberg, S. Holmgren, and Ö. Carlborg. Simultaneous search for multiple QTL using the global optimization algorithm DIRECT. *Bioinformatics*, 20:1887–1895, 2004.
22. K. Ljungberg, K. Mishchenko, and S. Holmgren. Efficient algorithms for multi-dimensional global optimization in genetic mapping of complex traits. Technical Report 2005-035, Division of Scientific Computing, Department of Information Technology, Uppsala University, 2005.
23. Kajsa Ljungberg. *Numerical Methods for mapping of Multiple Quantitative Trait Loci in Experimental Populations*. Phd thesis, Uppsala University, 2005.
24. M. Lynch and B. Walsh. *Genetics and Analysis of Quantitative Traits*. Sinauer Associates, Inc, 1998.

25. O. Martinez and R. Curnow. Estimating the locations and the sizes of effects of quantitative trait loci using flanking markers. *Theoretical and Applied Genetics*, 85:480–488, 1992.
26. 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.
27. K. Shimomura, S. Low-Zeddies, D. King, T. Steeves, A. Whiteley, J. Kushla, P. Zemenides, A. Lin, M. Vitaterna, G. Churchill, and J. Takahashi. Genome-wide epistatic interaction analysis reveals complex genetic determinants of circadian behavior in mice. *Genome Research*, 11:959–980, 2001.
28. F. Sugiyama, G. Churchill, D. Higgins, C. Johns, K. Makaritsis, H. Gavras, and B. Paigen. Concordance of murine quantitative trait loci for salt-induced hypertension with rat and human loci. *Genomics*, 71:70–77, 2001.

Table 2. Results for the analysis of the mouse data set with a model where $d = 3$

p	T_{grid}	T_{SMP}	N_{max}	N_{max}/N_{avg}
1	8	13.7	23281	1
2	2	4.22	7380	1.02
4	2	2.44	4218	1.15
8	2	1.51	2521	1.28

Table 3. Results for the analysis of the mouse data set with a model where $d = 4$

p	T_{grid}	T_{SMP}	N_{max}	N_{max}/N_{avg}
1	79	155	124964	1
2	31	56.4	62760	1
4	35	26.3	33434	1.03
8	11	19.8	25590	1.45
16	4	-	11557	1.21

Table 4. Results for the analysis of the mouse data set with a model where $d = 5$

p	T_{grid}	T_{SMP}	N_{max}	N_{max}/N_{avg}
1	2957	8550	991030	1
2	2743	7924	928713	1
4	276	612	251794	1.09
8	175	376	191513	1.24
16	66	-	96751	1.57
32	40	-	64141	1.87