BIOINFORMATICS APPLICATIONS NOTE Vol. 30 no. 8 2014, pages 1177–1179
doi:10.1093/bioinformatics/btt754

Genome analysis

Advance Access publication January 7, 2014

Regional heritability advanced complex trait analysis for GPU and traditional parallel architectures

L. Cebamanos[1], A. Gray[1,*], I. Stewart[2] and A. Tenesa[2]

[1]EPCC and [2]The Roslin Institute, The University of Edinburgh, Edinburgh, UK

Associate Editor: John Hancock

ABSTRACT

Motivation: Quantification of the contribution of genetic variation to phenotypic variation for complex traits becomes increasingly computationally demanding with increasing numbers of single-nucleotide polymorphisms and individuals. To meet the challenges in making feasible large-scale studies, we present the REgional heritability advanced complex trait analysis software. Adapted from advanced complex trait analysis (and, in turn, genome-wide complex trait analysis), it is tailored to exploit the parallelism present in modern traditional and graphics processing unit (GPU)-accelerated machines, from workstations to supercomputers.

Results: We adapt the genetic relationship matrix estimation algorithm to remove limitations on memory, allowing the analysis of large datasets. We build on this to develop a version of the code able to efficiently exploit GPU-accelerated systems for both the genetic relationship matrix and REstricted maximum likelihood (REML) parts of the analysis, offering substantial speedup over the traditional central processing unit version. We develop the ability to analyze multiple small regions of the genome across multiple compute nodes in parallel, following the 'regional heritability' approach. We demonstrate the new software using 1024 GPUs in parallel on one of the world's fastest supercomputers.

Availability: The code is freely available at http://www.epcc.ed.ac.uk/software-products

Contact: a.gray@ed.ac.uk

Received on September 4, 2013; revised on December 11, 2013; accepted on December 22, 2013

1 INTRODUCTION

The majority of traits of interest in relation to animal and human disease, livestock breeding and natural and artificial selection can be described as *complex*, in that they are determined by large numbers of genetic and environmental factors as well as their interactions. Statistical analysis techniques can be used to quantify the contribution of genetic variation to phenotypic variation for complex traits and genomic prediction. Large cohorts and dense single-nucleotide polymorphism (SNP) panels, including millions of SNPs, are required to achieve sufficient statistical power for mapping and increased accuracy at the expense of increased computational demand. It is of vital importance to tailor algorithmic implementation as closely as possible to modern computational hardware, to make the desired studies feasible.

*To whom correspondence should be addressed.

Modern workstations, or *nodes*, typically contain one or a few central processing units (CPUs), each with multiple cores, and a memory of limited size. Larger systems can be built by interconnecting multiple nodes. Many systems now also feature augmentation with graphics processing units (GPUs); traditionally used for computer gaming, these contain typically hundreds or thousands of simplistic cores and can offer substantial computational advantages when used in conjunction with CPUs as *compute accelerators*. Thus, increased computational performance comes at the cost of increased architectural complexity. Traditional programming languages can target only a single core of a single CPU. To fully exploit modern resources, one must embrace parallel programming techniques. For each of the different types of parallelism described above, different techniques must be used, and these can be combined to form *hybrid* programs able to perform well on systems ranging from simple workstations to the largest of supercomputers.

In (Gray *et al.*, 2012), we presented advanced complex trait analysis (ACTA), a software package for quantifying the contribution of genetic variation to phenotypic variation for complex traits using core functionality from the genome-wide complex trait analysis (GCTA) software (Yang *et al.*, 2011) but with dramatically improved computational performance on modern multi-core CPUs.

In this article, we introduce REgional heritability advanced complex trait analysis (REACTA), a derivative of ACTA offering several key advantages in addition to the preexisting functionality. First, as described in Section 2.1, the genetic relationship matrix (GRM) estimation algorithm has been adapted to remove memory size limitations and allow the analysis of large datasets. Building on this, we have developed a version able to exploit GPU-accelerated architectures, as described in Section 2.2. In Section 2.3, we describe development of functionality allowing a regional heritability approach, in which multiple compute nodes can be used in parallel to automatically analyze multiple small regions of the genome. Finally, in Section 3, we present results comparing the performance of the CPU and GPU versions and showcase a demonstrative run using 1024 GPUs in parallel on one of the world's largest supercomputers. Note that, whether using the regional heritability approach or not, our developments preserve numerical consistency (for both GRM files and REML results) with the original GCTA software. All our developments are thoroughly tested through use of an automated test suite containing a range of datasets and use-cases (where, of course, GCTA does not have the functionality to automatically analyze multiple regions, but one can compare single regions manually).

2 BACKGROUND AND METHODS

2.1 Controlling memory usage

The ever-increasing numbers of SNPs and individuals desirable for analysis incur memory management challenges. The GRM is generated through the WW^T operation. The matrix W, derived from genotype data, can become prohibitively large, e.g. for 20 K individuals and 500 K SNPs, it consumes 37 GB of memory. The situation is even more of a problem for implementation on GPUs that typically offer relatively low memory, and the data sizes will rise as more data become available.

To automatically control memory usage for the GRM estimation within a user-definable limit, we decompose the above into multiple matrix multiplications, each involving a subset of SNPs, plus a final combination stage. A number of adaptations were required to facilitate this. To calculate W, the genotype data must be read from a (compressed PLINK format) file, expanded and processed. We created functionality to allow the compressed format to be kept resident in memory for the duration of the program, and the W subset corresponding to each subset of SNPs can be extracted as and when required. This improved methodology not only allows the exploitation of restricted memory machines (including GPUs) for those demanding large problems of interest, but also has the added benefit of allowing the GRM to be calculated 'on-the-fly' when required. When running an REML analysis, the user can optionally instruct the code to calculate the GRM rather than providing a file. This is not only user-friendly, but also saves the time in writing and reading files. It is also a vital component in the multi-region analysis (see Section 2.3).

2.2 GPU implementation

GPUs offer performance advantages over CPUs, at the expense of programming complexity. The GPU is used in tandem with the CPU as a 'compute accelerator', responsible for the most computationally demanding sections of the problem. The NVIDIA Compute Unified Device Architecture (CUDA) programming model provides the functionality necessary for the programmer to map the code to the hierarchical parallel GPU architecture and explicitly manage the memory transfers between the distinct CPU and GPU memory spaces.

The traditional version of ACTA uses BLAS and LAPACK linear algebra libraries (Gray *et al.*, 2012). For the GPU implementation, we exploit equivalent functionality in the GPU-enabled MAGMA library (Agullo *et al.*, 2009) combined with hand-written CUDA.

For the GRM creation, we use the GPU-enabled BLAS from MAGMA for each of the matrix multiplications described in Section 2.1. For the matrix inversions that dominate the REML analysis, we use the Cholesky factorization and matrix inversion routines provided by MAGMA.

We use a number of other CUDA kernels to obtain further speedup, and to avoid the repeated transfer of data between the CPU and GPU memory spaces, which would be prohibitively expensive. For the code sections remaining on the CPU, we use OpenMP to fully exploit all the cores.

2.3 Regional heritability analysis

It is often of interest to use variance component analysis for mapping disease loci or quantitative trait loci. Such an approach can facilitate mapping regions where there are multiple SNPs contributing to the trait of interest Nagamine *et al.* (2012). We have developed a new `--multi-region` option to automatically perform regional analysis where the size of and overlap between regions can be specified. For each region, files are written containing the analysis results for that region, a list of the SNPs used in that region and the standard output from the code. Although similar computationally efficient methods have previously been proposed (Listgarten *et al.*, 2013), their approach is only computationally efficient in situations where the number of individuals is larger than the number of SNPs included in the estimation of the relationship matrix. In addition, such methods have the caveat that a small subset of SNPs needs to be selected to correct for hidden population structure. Such a selection of SNPs, based on significance levels, depends on the sample size of the dataset available. Our option `--multi-region-with-polygenic` differs from this approach by fitting all available SNPs using two random effects. By fitting one local effect plus a polygenic effect, the software accounts for population structure without requiring the pre-selection of SNPs.

When run on a single compute node, REACTA will automatically loop over all regions. However, the computational cost of the approach can be high, especially if one wishes to obtain empirical levels of significance by permutation. However, the parallelism inherent to the method can be mapped to large-scale parallel machines. We have developed functionality, using the Message Passing Interface, to automatically run across multiple compute nodes in a parallel cluster, where each node is responsible for a subset of regions. Within each node, the CPU or GPU version is invoked (depending on the hardware resources available) to analyze each region, where each GRM is calculated 'on-the-fly' before the associated REML analysis, as described in Section 2.1. Therefore, we provide a hierarchical parallelization: message passing inter-node communications combined with OpenMP or CUDA intra-node parallelization.

3 RESULTS

In Figure 1, we compare the CPU and GPU versions for three different test cases. The CPU version is run on an Intel Xeon E5-2620, fully using all six cores via the use of six OpenMP threads. The GPU version is run on an NVIDIA Tesla K20 GPU. It can be seen that GPU has an overall performance advantage of around a factor of 3, where the advantage is more profound for the GRM section. We stress that (unlike many other published GPU–CPU comparisons) our baseline performance results are for a highly optimized CPU version, which makes optimal use of all the cores, and SIMD lanes within each core (Gray *et al.*, 2012). We provide both CPU and GPU versions of the software, such that users can decide which to use based on their specific resources and requirements. A cost analysis is beyond the scope of this article, as to be comprehensive it would need to include factors such as electricity usage, which are difficult to determine and may vary with location. Furthermore, the GPU

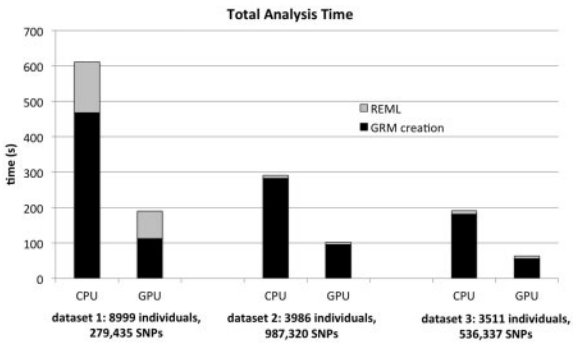

Fig. 1. The analysis time taken by the CPU and GPU versions of the code, for three different test cases, where results are decomposed into the GRM estimation and REML stages. Note that the REML time is independent of the number of SNPs

purchase cost is dependent on memory requirements, which, in turn, are dataset-specific.

To demonstrate the effectiveness of the multi-region functionality, we have used the world's second fastest (at the time of writing) open-access supercomputer, *Titan*. This Cray XK7 machine is based at Oak Ridge National Laboratory in the USA and comprises 18 688 nodes, each with a single NVIDIA K20X GPU augmenting an AMD Opteron 6274 16-core CPU, where the nodes are connected via the Cray Gemini network.

We split the 987 320 SNPs into regions of size 100 SNPs, with an overlap of 50 SNPs between regions: a total of 19 746 regions (where a separate GRM creation and REML analysis are performed for each). We analyze these in parallel using 1024 GPUs, where each GPU automatically iterates through its subset of regions. This run took 15 min 26 s. From the individual region timings, we can infer that on a single GPU, this would have taken a factor of 842 longer: >9 days. It is noteworthy that in the same time, our software would allow us to determine empirical levels of statistical significance by permutations. The deviation from *ideal* scaling (of a factor of 1024) is due to the fact that the time per region varies and the load is not perfectly distributed between GPUs. This may be improved in future work through redistribution strategies at run-time.

Funding: This work was supported by BBSRC [BB/K000195/1] and CRUK [C12229/A13154], and used resources of the ORLCF at ORNL, supported by DoE [DE-AC05-00OR22725].

Conflict of interest: None declared.

REFERENCES

Agullo,E. (2009) Numerical linear algebra on emerging architectures: the PLASMA and MAGMA projects. *J. Phys.*, **180**, 1.

Gray,A. *et al.* (2012) Advanced complex trait analysis. *Bioinformatics*, **28**, 3134–3136.

Listgarten,J. *et al.* (2013) A powerful and efficient set test for genetic markers that handles confounders. *Bioinformatics*, **29**, 1526–1533.

Nagamine,Y. *et al.* (2012) Localising loci underlying complex trait variation using regional genomic relationship mapping. *PLoS One*, **7**, e46501.

Yang,J. *et al.* (2011) GCTA: a tool for genome-wide complex trait analysis. *Am. J. Hum. Genet.*, **88**, 76–82.