BIOINFORMATICS APPLICATIONS NOTE Vol. 26 no. 1 2010, pages 134–135
doi:10.1093/bioinformatics/btp608

Gene expression

The gputools package enables GPU computing in R

Joshua Buckner[1,*], Justin Wilson[1], Mark Seligman[3], Brian Athey[2], Stanley Watson[1] and Fan Meng[1,2]

[1]Department of Psychiatry and Molecular and Behavioral Neuroscience Institute, [2]National Center for Integrative Biomedical Informatics, University of Michigan, Ann Arbor, MI 48109 and [3]Rapid Biologics, P.O. Box 31989, Seattle, WA 98103, USA

Received on July 8, 2009; revised on October 7, 2009; accepted on October 17, 2009

Advance Access publication October 22, 2009

Associate Editor: John Quackenbush

ABSTRACT

Motivation: By default, the R statistical environment does not make use of parallelism. Researchers may resort to expensive solutions such as cluster hardware for large analysis tasks. Graphics processing units (GPUs) provide an inexpensive and computationally powerful alternative. Using R and the CUDA toolkit from Nvidia, we have implemented several functions commonly used in microarray gene expression analysis for GPU-equipped computers.

Results: R users can take advantage of the better performance provided by an Nvidia GPU.

Availability: The package is available from CRAN, the R project's repository of packages, at http://cran.r-project.org/web/packages/gputools More information about our gputools R package is available at http://brainarray.mbni.med.umich.edu/brainarray/Rgpgpu

Contact: bucknerj@umich.edu

1 INTRODUCTION

Microarray gene expression analysis often involves CPU intensive computation requiring days or even months on mainstream desktop computers. Examples are the Granger test for detecting causal relationships in gene expression time course data (Kaminski *et al*., 2001), pairwise similarity calculations involving permutations, and machine learning-based approaches for microarray sample classification.

R is the most popular statistical environment for microarray data analysis. By default, R does not perform calculations in parallel. Modern graphics processing units (GPUs) have made parallel processing on a single desktop computer practical. Using a GPU, parallel processing may be applied to data from high-throughput biological experiments to increase analysis efficiency (Mateos-Duran *et al*., 2009). To facilitate the application of parallel processing using a GPU to such data, we developed the gputools R package. With the gputools R package and a CUDA compatible GPU, the efficiency of our own gene expression analyses may be improved.

The gputools package for the R statistical environment provides a collection of functions that make use of an Nvidia GPU and Nvidia's CUDA toolkit to achieve parallelism on a consumer grade desktop computer. This application note describes the functions and the performance of the gputools R package.

*To whom correspondence should be addressed.

2 DESCRIPTION OF THE PACKAGE

Table 1 contains a summary of the primary and secondary functions of the package. The primary functions include the functions gpuCor, gpuGranger, gpuHclust, gpuSvmTrain and gpuMi. For example, gpuCor calculates Pearson and Kendall correlation coefficients and may be used as a substitute for R's cor function for large datasets. The function gpuGranger applies the Granger causality test (Kaminski *et al*., 2001) to time sequences of variables and is comparable with the granger.test function from the MSBVAR R package. Refer to Table 1 for descriptions of the rest of the primary functions.

The secondary functions include functions that extend or assist primary functions: gpuDistClus, gpuSvmPredict, getAucEstimate (which is not GPU enabled), gpuTtest, chooseGpu and getGpuId. The secondary functions also include functions for GPU assisted matrix algebra: gpuMatMult, gpuQr and gpuSolve. The gpuQr and gpuSolve functions are relatively inefficient. The two functions have been retained in the package in the hope that others may find the source code instructive.

3 RESULTS

We tested GPU enabled functions against non-GPU enabled versions with biomedical data on a desktop computer. The desktop computer has an Intel Core i7 920 processor and an Nvidia GeForce GTX 295 GPU card. The desktop computer's operating system is the CentOS 5.3 Linux distribution.

The primary functions of the gputools package were tested against corresponding non-GPU enabled R solutions (Table 2). We chose to use a single thread of the R environment in our test, since this is the way that most users interact with R. The function gpuMi was tested against the same algorithm coded in the C programming language with an R interface and without any parallelism.

The gpuGranger function was tested using a dataset from a P19 cell line differentiation gene expression profiling study from Dr Michael Uhler's laboratory of the University of Michigan (unpublished data) with 12 time points using the Illumina MouseRef-8 BeadChip. Each trial of the gpuGranger function used a subset of the 28 843 probes in the chip. The other functions were tested with subsets of GSE6306 (Li *et al*., 2006).

Table 1. Functions of the gputools R package

Function	Description
gpuGranger	Granger-causality (Kaminski *et al*., 2001)
gpuHclust	Hierarchical clustering
gpuCor	Pearson and Kendall correlation coefficients
gpuSvmTrain	Support vector machine training (Carpenter, 2009)
gpuMi	B-spline based mutual information (Daub *et al*., 2004)
gpuDist	Distance between vectors
gpuDistClust	Correlation coefficients from vector data
gpuSvmPredict	Predict classes using output of gpuSvmTrain
getAucEstimate	Estimate the quality of an svm (Hand and Till, 2001)
gpuTtest	Perform Student's t-test
gpuMatMult	Matrix multiplication
gpuQr	QR decomposition (Bjorck, 1996)
gpuSolve	Solve linear systems and find matrix inverses
chooseGpu	Select a GPU in case the desktop has more than one
getGpuId	Get the number of the currently selected GPU

Table 2. Primary functions and their pre-existing R solutions

Function	Pre-existing R counterpart
gpuGranger	granger.test of package MSBVAR
gpuHclust	hclust
gpuCor	cor
gpuSvmTrain	svm of package e1071
gpuMi	none known

Table 3. Test data sizes

Test Number	1	2	3	4	5
gpuGranger	200	400	600	800	1000
gpuHclust	1000	2000	4000	6000	8000
gpuCor	20	30	40	50	60
gpuSvmTrain	9000	10 000	11 000	12 000	13 000
gpuMi	2000	4000	6000	8000	10 000

The performance test results are shown in Figure 1. For gpuCor, the test number in Table 3 and Figure 1 indicates the number of components of each of 10 000 vectors. For the rest of the primary functions, the test number indicates the number of variables. In the gpuGranger test, lag was set to 2 and each variable had 12 samples (Kaminski *et al*., 2001). In the gpuCor test, 10 000 vectors were used for Kendall correlation for each trial. In the gpuMi test, each variables had 100 samples, there were 10 bins of equal width, and spline order was set to 3 (Daub *et al*., 2004). In the gpuSvmTrain test, each variable had 500 features (Carpenter, 2009).

4 CONCLUSION

Our approach significantly speeds up several algorithms frequently used in gene expression data analysis. Since the code for the gputools

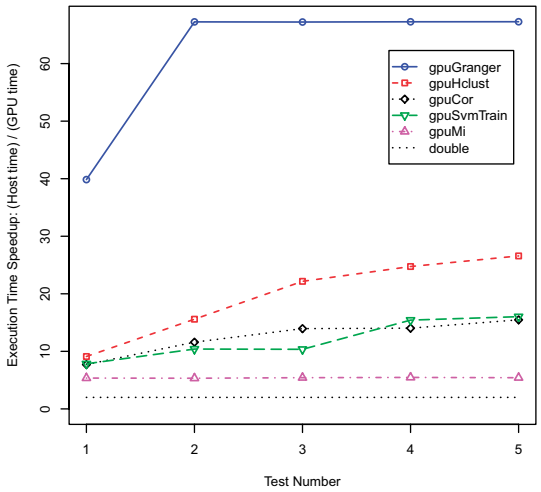

Fig. 1. Single GPU versus single thread.

package is open source and distributed with the package, researchers may customize it to suit individual needs. We welcome collaboration with the research community and contributions to the package from other researchers and developers.

Funding: J. Buckner, J. Wilson, and F. Meng are members of the Pritzker Neuropsychiatric Disorders Research Consortium, which is supported by the Pritzker Neuropsychiatric Disorders Research Fund L.L.C. This work is also partly supported by the National Center for Integrated Biomedical Informatics through National Institutes of Health grant 1U54DA021519-01A1 to the University of Michigan.

Conflict of Interest: none declared.

REFERENCES

Bjorck,A. (1996) *Numerical Methods for Least Squares Problems*. SIAM, Philadelphia.

Carpenter,A. (2009) cuSVM: a CUDA implementation of support vector classification and regression. Available at http://patternsonascreen.net/cuSVM.html (last accessed date November 17, 2009).

Daub,C. *et al*. (2004) Estimating mutual information using B-spline functions an improved similarity measure for analysing gene expression data. *BMC Bioinformatics*, **5**, 118.

Kaminski,M. *et al*. (2001) Evaluating causal relations in neural systems: granger causality, directed transfer function and statistical assessment of significance. *Biol. Cybern.*, **85**, 145–157.

Li,J.Z. *et al*. (2006) Sample matching by inferred agonal stress in gene expression analyses of the brain. *BMC Genomics*, **8**, 336.

Mateos-Duran,J.M. *et al*. (2009) Qnorm: a library of parallel methods for gene-expression Q-normalization. In *Bioinformatics Open Source Conference 2009, Stockholm, Sweden*, Available at http://www.open-bio.org/w/images/c/c7/BOSC2009_program_20090601.pdf (last accessed date November 17, 2009).

Hand,D.J. and Till,R.J. (2001) A simple generalisation of the area under the ROC curve for multiple class classification problems. *Mach. Learn.*, **45**, 171–186.