Qnorm: A library of parallel methods for gene-expression Q-normalization

José Manuel Mateos-Duran[1]; Pjotr Prins[2]; Andrés Rodríguez[1] and Oswaldo Trelles[1].
(1) Computer Architecture Dept.; University of Malaga; Campus de Teatinos, 29071; Spain.
(2) Lab. of Nematology, Wageningen University, The Netherlands
{jmateos, andres, ortrelles}@uma.es, Pjotr.Prins@wur.nl

Project web site: http://www.bitlab-es.com/qnorm
Source code at: http://github.com/ots/qnormalization
OSS License: GPL version 3

Abstract for BOSC 2009 Session: Multicore and GPGPU computing

Qnorm is a library for exploring different strategies in parallelization of large scale computations, a generic approach in high performance computing (HPC).

The high amounts of molecular data produced by current high-throughput technologies in modern biology poses challenging problems in our capacity to process and understand data. Not only allows pyro-sequencing the production of overwhelming sets of data but even ultra high density microarrays jumped the previous thirty thousand genes contained, in a simple array, to more than 5 million genetic markers. Nowadays clinical studies include hundreds of thousand of patients instead of the thousands genetically fingerprinted a few years ago, in a typical study. Current sequential implementations of software are unable to deal with such enormous volumes. Here we show the impact of different high performance computing strategies using three different parallel approaches for shared memory, distributed memory and GPU architectures, that can be easily applied to other existing bioinformatics algorithms and show how benchmarking helps decide on strategy.

As proof of concept we chose the quantile based method[1] as it provides a fast and easy to understand procedure to normalize multiple gene-expression datasets, under the assumption of sharing a common distribution. The high computational cost and memory requirements ($p > 6$ millions and $N > 1000$ samples) of sequential Q-normalization in-core calculations are behind our interest in developing an HPC approach to this problem.

For shared and distributed memory architectures, we use a dynamic load distribution over the set of columns that are concurrently sorted and partially row averaged in a first step. A synchronization barrier is needed before global averaging in a second step. A number of indexes are managed to avoid a re-ordering of the experiments with good effects in the processing time. The same two steps approach can be mapped to a GPU solution. Every column is processed in parallel by the GPU and then the global average column is also computed in parallel. Performance results have shown a near perfect speed-up in the supercomputing parallel strategies. As expected, a good GPU (graphics card) can provide a working solution, obviously more modest than in a supercomputer.

By improving the quantile normalization algorithm large microarray datasets can now be normalized, previously not achievable on a single computer. These methods are generic, and our benchmarking strategy applies to all forms of parallelization of existing algorithms.

Our purpose is to provide mechanisms in a parallel library that compute static, dynamic and guided self scheduling and load distribution algorithms, as well as functions for matrix mapping on disk, in an open source package. These novel quantile normalization routines will be freely available with bindings for Perl, Python, Ruby and R, through Biolib mappings (http://biolib.open-bio.org/).

[1] Bolstad, B.M., Irizarry R. A., Astrand, M., and Speed, T.P. (2003), A Comparison of Normalization Methods for High Density Oligonucleotide Array Data Based on Bias and Variance. Bioinformatics 19(2):185-193