Discovering a Broad Array of Functions for *Clock-Controlled Genes* Using Parallel Algorithm on Multi-GPGPU and MCMC

Ahmad Al-omari, Jonathan Arnold, Thiab Taha, Heinz-Bernd Schuttler
Institute of Bioinformatics, The University of Georgia, Athens, GA 30602, aomari@uga.edu

Introduction

A general purpose graphics processing unit (GPU) offers a solution to a very important and fundamental problem of interest to many researchers, a problem that would be prohibitive to solve without the technology of GPGPUs. The problem is how does the biological clock control the rhythms of ~2,418 genes in the genome (with 11,000 genes) of a model system, the filamentous fungus, *Neurospora crassa* (Dong, et al.,2008). Predicting and understanding the dynamics of all of these genes and their products in a genetic network, which is found from bacteria to humans(Lakin-Thomas, et al., 2011), describing how the clock functions is a challenge and beyond the current capability of the fastest serial computers. In order to overcome this problem, we developed two parallel algorithms that are described below(Al-Omari,et al.,2013,2014) that help us to discover a broad array of functions for *clock-controlled genes* using parallel algorithms on GPGPU.

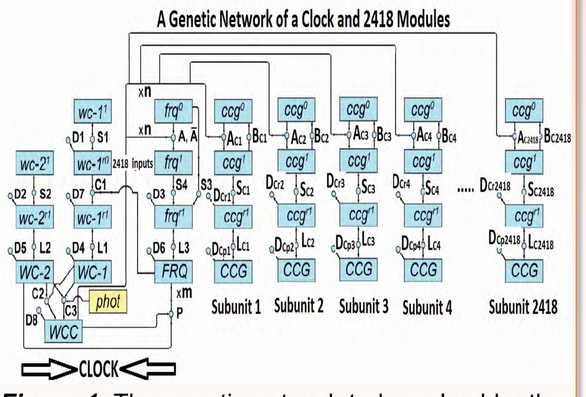

Figure 1. The genetic network to be solved by the GPUs[2,418 subunits]. The subunits are independent from each other but on the clock. Solving this huge network is beyond the fastest computer in existence.

Methods

Procedure Flowchart:

All of the slave modules appear in Figure1 has the same mathematical ODE form but different parameters:

$dg_0/dt=B_cg_1-A_cg_0w(t)$
$dg_1/dt=A_cg_0w(t)-B_cg_1$
$dg_r/dt=S_cg_1-D_{cr}g_r$
$dg_p/dt=L_cg_r-D_{cp}g_p$

↓

CPU sends the master module solution[w(t)] and the parameter set to each slave module

↓

GPU(s) solve(s) the 2,418 ODEs of the slave modules in parallel using **Adaptive Runge Kutta(ARK)** method, **or exact integral solution(EIS) using Gauss quadrature integrator**

↓

Each thread executes a kernel, which contains the **ARK** or **EIS**. Each thread executes the same system of ODEs for an assigned slave module but different data.

↓

GPU sends the solutions of the 2,418 systems of ODE back to the CPU

↓

CPU calculates a cost function (Chi-square using Metropolis procedure) that minimizes the difference between the experimental data and the fitted solution

Results

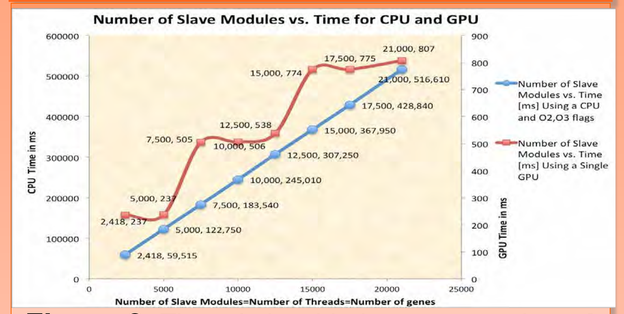

Figure 2. The relationship between number of thread blocks (slave modules) to be solved on the device (Red) and on the CPU (Blue) verses required time. The left Y-axis shows the CPU time while the right Y-axis shows the GPU time. The jump in the GPU time is due to exceeding the maximum number of TBs running simultaneously on the device. If a GPU is capable of running N blocks simultaneously, then from 1 to N blocks takes the same time to complete. The time of the CPU is monotonically increasing as a function of slave modules.

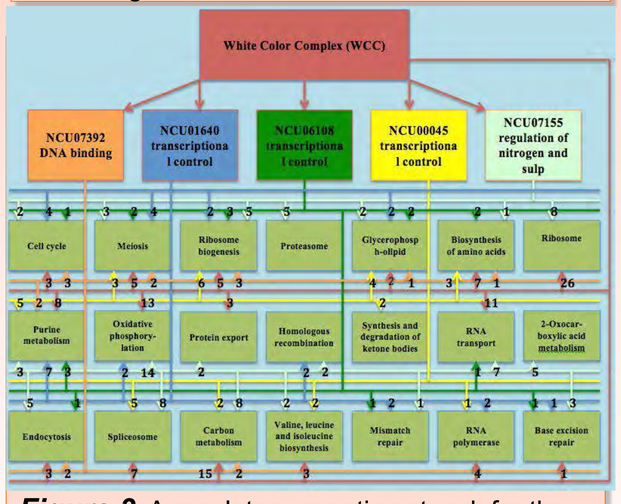

Figure 3. A regulatory genetic network for the six regulators (**WCC, NCU07392, NCU01640, NCU06108, NCU00045, NCU07155**) and the putative *clock-controlled genes*. The number on the arrow indicates how many annotated genes that are regulated by a particular regulator and participating in a particular pathway or function (small green boxes).

Conclusion

We have demonstrated that we can implement the ensemble method *(Yu et al., 2007)*. We have also demonstrated that we can solve the ODEs on GPUs for the slave module in ***Figure3.*** We propose to:

- Implement an ensemble method on the CPU and GPUs for identifying the network in ***Figure1.***

We have developed two algorithms on the GPGPU to solve the genetic network of 2,418 genes shown in ***Figure1*** within about 4 months using **ARK** algorithm instead of many years on the CPU and within 12 hours using **EIS** algorithm and as it is shown in **Figure 2**.

We have more than 31,668 data points for fitting the proposed large genetic network for the clock (Dong et al., 2008). We propose to:

- Fit the genetic network in ***Figure1*** by the best parallelization strategy on the CPU and GPUs to our microarray data.

The resulting network enable us to examine how the clock controls ribosome biogenesis and other pathways by a detailed molecular mechanism shown in **Figure 3**.

References

- Dong, W., et al. (2008) Systems biology of the clock in Neurospora crassa, PloS one, 3, e3105.
- Huang, Y. (2007) Parameter estimation of chemical reaction networks. A Master thesis,Physics Department, The University of Georgia.
- Lakin-Thomas, P.L.,Bell-Pedersen, D. and Brody, S. (2011) 3 The Genetics of Circadian Rhythms in Neurospora, *Advances in genetics*,**74**, 55.
- A. Al-Omari, J. Arnold, T. Taha, and H. Schuttler, "Solving Large Nonlinear Systems of First-Order Ordinary Differential Equations with Hierarchical Structure Using Multi-GPGPUs and an Adaptive Runge Kutta ODE Solver," 2013.
- A. Al-Omari, H.-B. Schuttler, J. Arnold, and T. Taha, "Solving Nonlinear Systems of First Order Ordinary Differential Equations Using a Galerkin Finite Element Method," *Access, IEEE*, vol. 1, pp. 408-417, 2013.
- Ahmad Al-Omari, James Griffith, Michael Judge, Thiab Taha, Jonathan Arnold, & H-Bernd Schuttler "Fitting and discovering a regulatory network function using EnsembleMethods on GPGPUs with special reference to the biological clock of Neurospora crassa", *Nature*, 2014 (under review).
- Yu, Y., *et al.* (2007) A genetic network for the clock of Neurospora crassa,*Proceedings of the National Academy of Sciences*,**104**, 2809-2814.

Acknowledgment

CUDA Teaching Center At The University of Georgia