BIOMOLECULAR NETWORK SIMULATOR: SOFTWARE FOR STOCHASTIC SIMULATION OF CELLULAR BIOLOGICAL PROCESSES

Yaroslav Chushak

Biotechnology HPC Software Applications Institute
US Army Medical Research and Materiel Command
ARLF/HEPB WPAFB, OH 45433-5707

Brent Foy

Department of Physics
Wright State University
Dayton, OH 45435

John Frazier

Air Force Research Laboratory
WPAFB, OH 45433-5707

ABSTRACT

We developed a software package called Biomolecular Network Simulator (BNS) to model and simulate complex biomolecular reaction networks. The software uses the Gillespie stochastic simulation algorithm for generating Monte Carlo trajectories that describe the evolution of a system of biochemical reactions. This software uses a combination of MATLAB and C-coded functions and is parallelized with the MatlabMPI library to run on multiprocessor architectures.

Simulations on HPC can be run either in the interactive or in the batch mode. The graphical user interface of BNS allows users to easily set parameters for the model and simulations. Furthermore, BNS contains a comprehensive set of tools for post-simulation analysis of the results.

1 INTRODUCTION

At the functional level, all biological processes in cells can be represented as a series of biochemical reactions. These processes can be divided into two groups - gene expression and metabolic reactions - that can be described by different approaches.

On the molecular level, all biochemical reactions require the interaction of the reactants with sufficient energy to overcome the activation energy barrier and fundamentally, they are stochastic in nature. However, metabolic reactions generally deal with large numbers of molecules and random fluctuations are negligible in magnitude comparing with the average number of molecules. Therefore, metabolic reactions can be modeled without a significant loss of accuracy by a deterministic approach in terms of Ordinary Differential Equations (ODE) [1]. This approach uses the concentration of molecules as variables and assumes that changes of state of the system are continuous. The ODE can be solved numerically with the help of specialized libraries of programs.

The situation is different for gene expression. Some of the reactions evolved in gene expression deal with small numbers of molecules. In this case, the modeling of these reactions as continuous flux of matter is no longer valid and the stochastic nature of chemical reactions now becomes important [2,3].

Since gene expression reactions are stochastic in nature, the best way to model them is to use a stochastic approach in terms of the Chemical Master Equation (CME), with the number of molecules of each molecular species as variables. The CME describes transitions of the system from one state to another state using probabilistic methods. Gillespie proposed a method to determine probabilistically-correct trajectories based on the CME through the use of Monte Carlo simulations [4].

We developed a software package – the Biomolecular Network Simulator (BNS) – that uses the Gillespie stochastic algorithm [4] to simulate the behavior of a system of biochemical reactions. It allows scientists to build a synthetic biomolecular network and to optimize its performance. BNS contains tools for both simulating the system and for analyzing the results of the simulation. Since the simulations are stochastic in nature, the user must run thousands of simulations to characterize the "ensemble" behavior of biological systems. Furthermore, the BNS

 ISBN 1-56555-313-(

code is parallelized allowing users to run simulations and store results on multiprocessor computers. In this paper, we present a brief description of the Biomolecular Network Simulator software along with some examples.

2 STOCHASTIC SIMULATION ALGORITHM

Let us consider a system composed of N well mixed chemical species, S_i $(i = 1,\ldots N)$, in a fixed volume V, which are involved in M reactions, R_μ $(\mu = 1,\ldots M)$. The dynamical state of the system can be specified by the state vector $\mathbf{X}(t) = (X_1(t), X_2(t),\ldots X_N(t))$, where $X_i(t)$ is the number of molecules of species S_i at time t.

The reaction rate for each reaction R_μ is defined by a propensity function $a_\mu(\mathbf{X}) = c_\mu \cdot H_\mu(\mathbf{X})$, where c_μ is the stochastic probability constant and $H_\mu(\mathbf{X})$ represents the number of possible combinations of reactants. Each reaction is also characterized by its state-change vector $\boldsymbol{\nu}_\mu = (\nu_{1\mu}, \nu_{2\mu}, \ldots \nu_{N\mu})$, where $\nu_{i\mu}$ is the change in the number of molecules S_i caused by one reaction R_μ.

To study the evolution of the state vector $\mathbf{X}(t)$, Gillespie proposed an algorithm for Monte Carlo generation of stochastic trajectories [4]. The direct simulation algorithm, (Table 1), which is implemented in the Biomolecular Network Simulator, answers two questions: (1) which reaction will occur next, and (2) what is the waiting time for the next reaction to occur.

To answer these questions, two random numbers uniformly distributed over the interval (0,1) – r_1 and r_2 – are generated. The first random number is used to determine the next reaction R_μ, where μ = i, such that

$$\sum_{i=1}^{\mu-1} a_i < r_1 \cdot a_0 < \sum_{i=1}^{\mu} a_i \,, \qquad (1)$$

where

$$a_0 = \sum_{\mu=1}^{M} a_\mu \,. \qquad (2)$$

The distribution of the waiting time is given by following probability density function:

$$P(\tau, \mu) = a_\mu \exp(-a_0 \tau) \,. \qquad (3)$$

Here, $P(\tau, \mu)$ is the probability that the waiting time for the reaction is τ and that it will be an R_μ reaction. The waiting time for the next reaction is calculated as [4]

$$\tau = \frac{1}{a_0} \log \frac{1}{r_2} \quad . \qquad (4)$$

After the next reaction and its waiting time are determined, the reaction is executed and the state of the system is changed according to the state-change vector $\boldsymbol{\nu}_\mu$. The simulation time is increased then by τ and the next simulation step is generated.

Table 1: Summary of the main steps in the Gillespie stochastic simulation algorithm.

Step 1:	Initialization. Set the state vector $\mathbf{X} = \mathbf{X}(t=0)$ to the initial number of molecules S_i and set the time t = 0.
Step 2:	Calculate the propensity $a_\mu(\mathbf{X})$ for each reaction R_μ and their sum $a_o(\mathbf{X})$.
Step 3:	Generate two random numbers r_1 and r_2 uniformly distributed over the interval (0,1).
Step 4:	Find the next reaction R_μ according to Eq. (1).
Step 5:	Calculate the waiting time τ for the next reaction according to Eq. (4).
Step 6:	Update the number of molecules according to the state-change vector $\boldsymbol{\nu}_\mu$.
Step 7:	Update the simulation time t = t + τ.
Step 8:	Calculate a new propensity $a_\mu(\mathbf{X})$ for each reaction that was affected by Step 6 and their sum $a_o(\mathbf{X})$.
Step 7:	Return to Step 3 or termination.
Termination. Simulations are terminated when the simulation time exceeds the maximal time of the simulations.	

The Gillespie stochastic algorithm is exact for the elementary reactions (uni-uni, uni-bi and bi-uni types of reactions) with any number of molecules in the system. For the system with large numbers of molecules, the trajectories generated by stochastic Monte Carlo simulations converge to the trajectory generated by deterministic differential equations [4].

3 BIOMOLECULAR NETWORK SIMULATOR

The Biomolecular Network Simulator uses a combination of MATLAB and C-coded modules. The front-end, graphical user interface (GUI) and analysis tools of BNS are written in MATLAB, while the simulation core engine is written in the C language. Such a combination allows one to use the interactive features and visualization tools of MATLAB, while achieving high speed for the computationally intensive part of the software with compiled C code. The parallelization of the code is done with the help of the MatlabMPI library [5]. BNS can be run on any computer platform where MATLAB 6.5 or newer is installed.

3.1 Input Data

A model is a set of mathematical relationships that describe the behavior of biochemical reactions that control cellular biological processes. Each of the 'Model' directories contains one or more subdirectories with model description files and/or different set of parameters for the same model and an 'Output' directory where the results of simulations are stored. There are two types of model directories: one for models defined in the Systems Biology Markup Language (SBML) format [5] and one for models defined as a set of MATLAB m-files. In addition, BNS allows one to perform simulations with multiple parameter sets, with each parameter set being run multiple times. Simulations with multiple parameter sets can be used for optimization and sensitivity analysis of the model.

3.2 Output Data

There are two types of output files: snapshot data and event log data. Both of these files are in MATLAB format. Snapshot data files contain the state of the system (number of molecules of each molecular species) at user specified time intervals. The information stored in the snapshot files are used to create runtime interactive graphics and for *post hoc* analysis of the data. The second type of output files – the event log files – contain the record of every discrete event that occurs during the simulation. The user should be aware that event log files may require considerable memory or hard disk space and, therefore, may create memory management problems for simulations involving a large number of long runs or for large reaction networks.

3.3 Parallelization

The parallelization of the BNS code is accomplished using the MatlabMPI library developed at MIT Lincoln Laboratory [6]. MatlabMPI consists of a set of MATLAB scripts that implements a subset of the Message Passing Interface (MPI) standard and allows the MATLAB program to run in a multiprocessor architecture. In our parallelization scheme, the 'master' processor divides the total number of runs between the available processors, sends a set of jobs to each of the 'workers' and performs some of the simulation runs itself. The snapshot data from the worker's runs are sent back to the master processor for the interactive graphics but the event log files are saved to the hard drive by the workers. In this approach to parallelization, we reduce the communication between the nodes and increase the speedup of the simulations. It is very close to being "embarrassingly parallel", with the primary non-trivial parallelization being the sending of intermediate data back to the master processor for real-time visualization. To test of BNS scaling with the number of processors we ran 1000 simulations on an SGI Origin 3900 machine, which has a shared memory architecture. A simulation here is defined as a single run of the chosen biochemical reaction network. A 92-fold speedup was observed by running BNS on 100 processors (Figure 1).

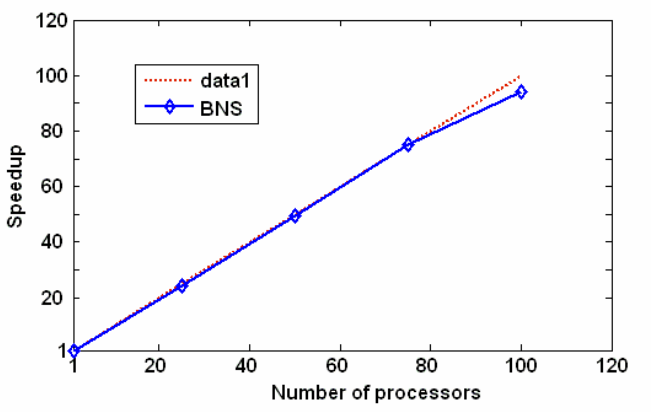

Figure 1. Scaling of BNS with the number of processors.

3.4 Running the Simulations

The BNS can be run either in command line mode or via a GUI. The GUI allows the user to modify model parameters at runtime and to execute simulations in the interactive or batch mode on HPC resources.

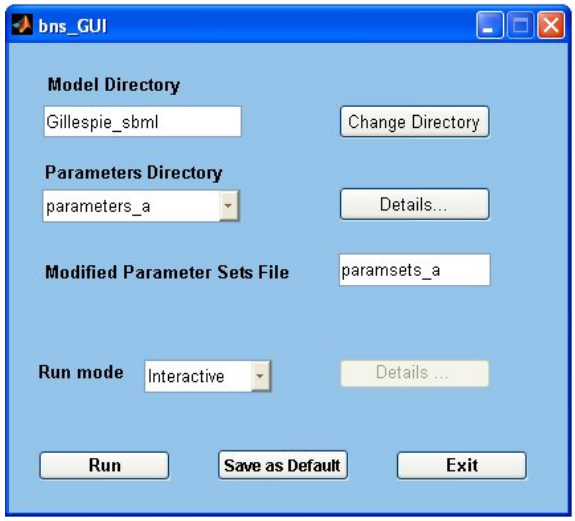

Figure 2. A screen shot of the main BNS GUI dialog window.

The main dialog window of the BNS GUI is shown in Figure 2. It allows the user to select the appropriate 'Model' and 'Parameters' directories and set the 'Run' mode.

A click on the 'Details' button next to the 'Parameters' directory opens a new window, shown in Figure 3. This

dialog window allows the user to modify model parameters and to set parameters for the simulation.

If simulations are run in the interactive mode, the results of the simulations will instantly appear on the screen. Usually, HPC centers allocate limited resources (in number of processors and running time) for interactive simulations, therefore BNS can be run in 'Batch' mode. In this mode all output data are stored on the hard drive for further analysis.

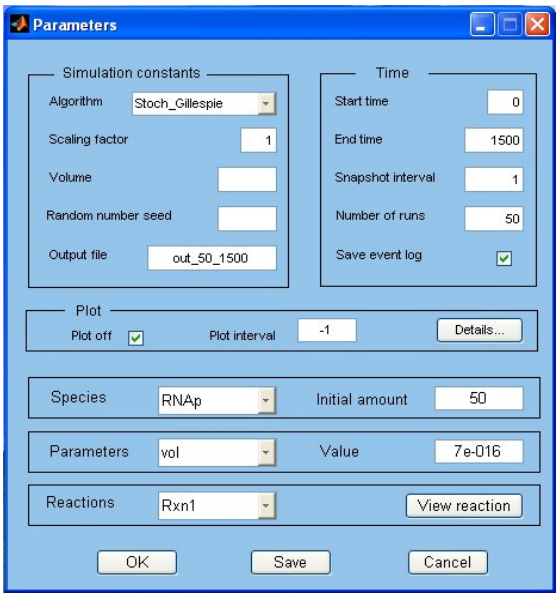

Figure 3. The parameters dialog window of the BNS GUI allows the user to modify the model parameters and to set simulation parameters.

3.5 Analysis

BNS has a comprehensive set of tools for post-simulation analysis. A GUI for the analysis tools allows the user to easily select the data and to set conditions for the analysis. Multiple types of post-simulation analyses are available.

3.5.1 Plots of number of molecules vs. time

The most frequently used type of analysis is a plot of the number of molecules vs. time. Such plots are available in the interactive mode or as post-simulation analysis. There are two ways to create plots: each compound is plotted on a separate figure or multiple compounds are plotted on the same figure window (grouping mode). The plot in Figure 4 is in grouping mode and shows the behavior of two molecular species, S1 and S2, over the time interval of 1500 seconds for a biomolecular reaction network containing transcription, translation and metabolic reactions. The number of molecules vs. time plots can be created with both types of output files: snapshot data or event log data.

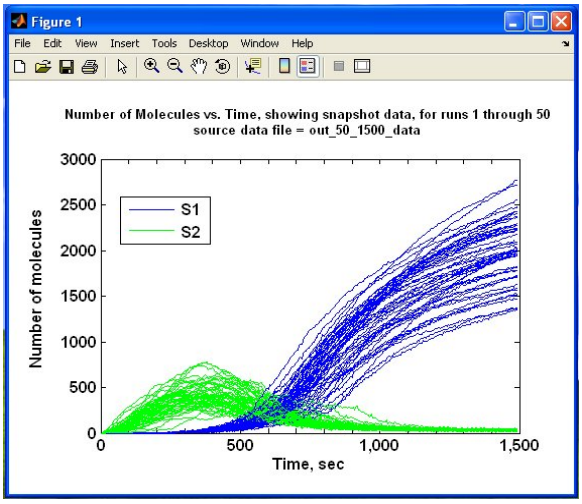

Figure 4. The evolution of the number molecules of molecular species S1 and S2 with time. The snapshot data for 50 runs are shown.

3.5.2 Time-weighted average analysis

A time-weighted average analysis refers to the calculations of the average number of molecules of a particular molecular species during a user selected time-bin. The average is weighted according to the amount of time the compound exists in each state during the selected time-bin. The time weighted average is then plotted versus time. The averaging analysis can be performed for a single run or for a selected set of runs.

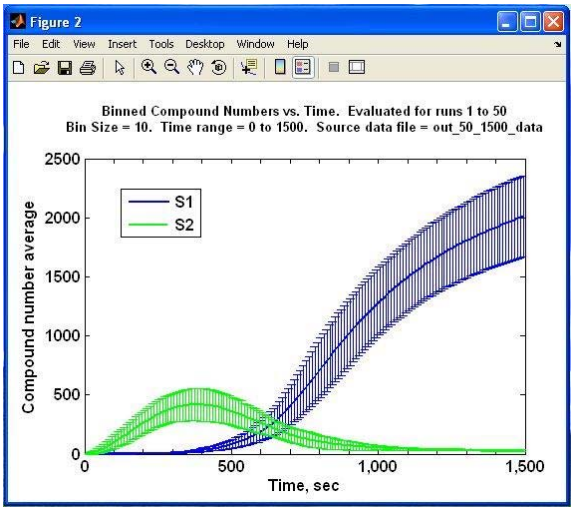

Figure 5. The averaged number of compounds S1 and S2 in the time interval (0, 1500) for the same simulation runs as in Figure 4. For each simulation, the time weighted average was calculated using a 10-sec time-bin and the time weighted averages were averaged over the 50 simulation runs. Data for the mean ± SD are shown.

When the analysis is applied to multiple runs, the plot shows the between run average (the average of each run's time-weighted average) and standard deviation. As in the previous case, the user can plot each compound on a separate figure or multiple compounds on the same figure. Figure 5 shows the between run average of the time weighted average number of molecules for the same 50 runs as shown on Figure 4 using a time averaging time-bin of 10 sec.

3.5.3 Reaction frequency analysis

Complex biomolecular reaction networks usually contain reactions that occur on different time scales: some reactions have a low propensity and occur rarely; other reactions are very fast and occur frequently. The data stored in the event log files allow the user to perform various reaction frequency analyses of the simulation data to learn more about the basic nature of the system. One type of analysis creates plots of the total number of times each reaction occurred during the simulation. Figure 6 shows an example of a histogram of the average number and standard deviation of the number of reaction occurrences in each reaction channel averaged over the 50 runs in Figure 4. The number of reactions is shown in the logarithmic scale.

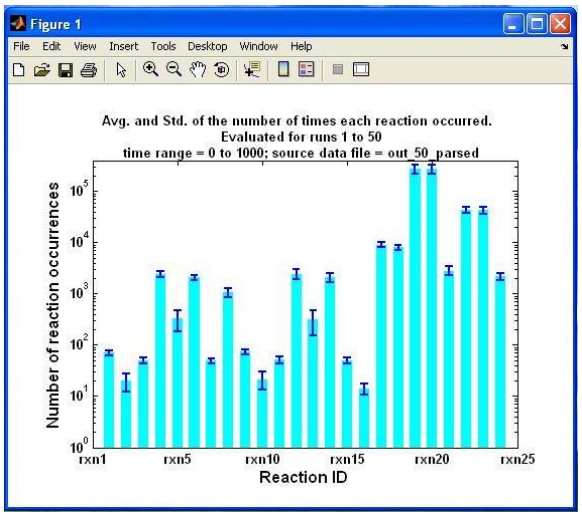

Figure 6. Histogram of the average total number of reaction occurrences in each reaction channel. The data are for the mean ± SD for the 50 simulation runs.

Reaction rates (number of reactions per unit time) in each reaction channel can also be calculated for user-selected time-bins and plotted versus time. Such types of analyses provide important information about the behavior of the system.

4 CONCLUSIONS

The Biomolecular Network Simulator allows the users to simulate the behavior of complex biological processes utilizing the capacities of high performance computers. Some of the features that distinguish BNS from similar tools are:

- usage of MATLAB and C-coded functions allows the user to combine intensive visualization of data with high speed computations;
- parallelized code for multiple simultaneous simulations allows the user to run BNS on multiprocessor machines;
- options to run the code in the interactive or batch mode;
- user friendly graphical user interface allows the user to easily set and modify parameters of the model, simulation and analysis; and
- comprehensive tool sets provide for post-simulation analysis of results.

ACKNOWLEDGMENT

The work of Yaroslav Chushak was made possible by a grant from the Department of Defense High Performance Computing Modernization Program office (HPCMP). The work of Brent Foy was made possible by a grant from the Air Force Office of Scientific Research (AFOSR) and by the Air Force Summer Faculty Fellowship Program.

REFERENCES

[1] Voit, E. O., 2000. *Computational Analysis of Biochemical Systems*. Cambridge University Press, Cambridge, UK.

[2] Elowitz, M. B., Levine, A. J., Siggia, E. D., and Swain, P. S., 2002, "Stochastic Gene Expression in a Single Cell." *Science*, v.297, no. 5584, (August): 1183 – 1186.

[3] Kaern, M., Elston, T. C., Blake, W. J, and Collins, J. J., 2005, "Stochasticity in Gene Expressions: From Theories to Phenotypes." *Nature Reviews Genetics*, v. 6, no, 6 (June): 451-464.

[4] Gillespie, D., 1977, "Exact Stochastic Simulation of Coupled Chemical Reactions." *J. Phys. Chem*. v. 81: 2340-2361.

[5] Huska, M., Finney, A., Sauro H. M., Bolouri, H. et al., 2003, "The Systems Biology Markup Language (SBML): A medium for representation and exchange of biochemical network models." *Bioinformatics*, v. 19, no. 4: 524-531.

[6] Kepner, J., 2001, "Parallel Programming with Matlab-MPI." Proceedings of the High Performance Embedded Computing workshop, HPEC 2001, (Lexington, MA, Sep 25-27)

DISCLAIMER

The opinions or assertions contained herein are the private views of the authors and are not to be construed as official or as reflecting the views of the U. S. Army or of the U. S. Department of Defense. "This paper has been approved for public release; distribution is unlimited."