Logarithmic Direct Method for Discrete Stochastic Simulation of Chemically Reacting Systems *

Hong Li † Linda Petzold ‡

July 27, 2006

Abstract

In biological systems formed by living cells, the small populations of some reactant species can result in dynamical behavior which cannot be captured by the traditional reaction rate equations. In that case, a more accurate simulation can be obtained by using the machinery of Markov process theory, specifically the Stochastic Simulation Algorithm (SSA). Since for realistic, practical biochemical systems the simulation by the SSA carries a high computational cost, several formulations have been proposed to increase the efficiency of this algorithm. In this paper we propose a highly efficient formulation of SSA, with computational complexity that is independent of the ordering of the reactions.

*This work was supported in part by the U.S. Department of Energy under DOE award No. DE-FG02-04ER25621, by the NIGMS under awards GM078993 and GM075297, by the National Science Foundation under NSF awards CCF-0428912, CTS-0205584, CCF-326576, and by the Institute for Collaborative Biotechnologies through grant DAAD19-03-D-004 from the U.S. Army Research Office.

†Department of Computer Science, University of California Santa Barbara, Santa Barbara, CA 93106. hongli@engineering.ucsb.edu

‡Department of Computer Science, University of California Santa Barbara, Santa Barbara, CA 93106. petzold@engineering.ucsb.edu

1 Introduction

Chemically reacting systems have traditionally been simulated by solving a set of coupled ordinary differential equations (ODEs).[1] Though traditional ODE-based approaches are sufficient for most systems, they fail to capture the stochasticity in some biochemical systems formed by living cells.[2–4] The dynamics of those systems can be simulated accurately by using the machinery of Markov process theory, specifically the stochastic simulation algorithm (SSA) of Gillespie.[2,5] SSA, which is an essentially exact procedure for well-stirred (spatially homogeneous) systems, is now in widespread use for the stochastic simulation of biochemical systems. For realistic modelling of biochemical systems, the computational cost of simulation by the SSA can be very high.

It is very important to have a highly efficient formulation of SSA, because this algorithm is the foundation of all discrete stochastic and multiscale algorithms for biochemical systems.[6] The Next Reaction Method (NRM),[7] the Optimized Direct Method (ODM)[8] and the Sorting Direct Method (SDM)[9] are three mathematically equivalent formulations of SSA which have been proposed recently to reduce the computational cost for simulation of biochemical systems.

Recent results[8] have shown that ODM is faster than NRM, unless the system is very nearly uncoupled. This was in contrast to a widely-held belief in the systems biology community that the Next Reaction Method formulation of SSA was the fastest. NRM uses an indexed priority queue [1] to reduce the search depth and reuse random numbers, and a dependency graph [2] to reduce the calculation of the propensities.[7] ODM is basically the original direct

[1]The indexed priority queue consists of a tree structure of ordered pairs of the index of reaction and the putative time when this reaction occurs.

[2]The dependency graph is a directed graph that describes how reactions affect each other. In other words, it is a data structure to tell which propensities need to be changed when a give reaction is executed.

SSA method, where the reactions have been re-ordered via the results of a single SSA simulation so that the most frequently accessed reactions are first. Subsequently, SDM[9] improved on ODM by dynamically ordering the reactions to eliminate the pre-simulation. Both ODM and SDM focused on the optimization of the system instead of the method itself. ODM loses some benefit when system behavior shifts during a simulation so that different sets of reactions are the most frequent ones to fire.[9] Similarly, frequent shifts during a simulation of the group of reactions which is fastest can defeat the efficiency of the bubble-up sorting of reactions in SDM. SDM's search depth[8] for locating the reaction which will fire in the next step (which is one of the time consuming parts of the simulation) is still $O(M)$, where M is the number of reactions.

In this paper we present an alternative formulation of the SSA, called the Logarithmic Direct Method (LDM). The computational cost of LDM is independent of the ordering of the reactions.

This paper is organized as follows. In Section 2 we briefly review SSA. Section 3 introduces LDM. Section 4 compares the performance of LDM with the previous best-performing algorithms.

2 Stochastic Simulation Algorithms

Gillespie's Stochastic Simulation Algorithm (SSA)[2,5,10] was the first rigorous numerical method using Monte Carlo techniques for stochastic chemical kinetics. It applies to a well-stirred (spatially homogeneous) chemically intact system, inside a fixed volume and at a constant temperature. The system contains M chemical reaction channels $\{R_1, \ldots, R_M\}$ involving N molecular species $\{S_1, \ldots, S_N\}$. The N molecular species $\{S_1, \ldots, S_N\}$ are represented by the dynamical state vector $X(t) = (X_1(t), \ldots, X_N(t))$, where $X_i(t)$ is the

number of molecules of species S_i in the system at time t. $X(t)$ is a Markov process.

Each reaction channel R_j is characterized by a propensity function a_j and state change vector $\nu_j = \{\nu_{1j}, \ldots, \nu_{Nj}\}$, where $a_j(x)dt$ is the probability, given $X(t) = x$, that one R_j reaction will occur in the next infinitesimal time interval $[t, t + dt)$, and ν_{ij} is the change in the number of species S_i due to one R_j reaction. The $N * M$ matrix ν is the stoichiometric matrix.

The SSA computes one *realization* of a dynamic trajectory of a chemically reacting system. Often an *ensemble* of trajectories is computed, to obtain an estimate of the probability density function of the system.

The SSA is based on the so-called *Next Reaction Density Function*,[10] which gives the joint probability that reaction R_j will be the next reaction and will occur in the infinitesimal time interval $[t, t + dt)$, given $(X(t) = x)$. The joint density function is formulated as follows:

$$P(\tau, j \mid \boldsymbol{x}_t, t) = a_j(\boldsymbol{x}_t) e^{-a_0(\boldsymbol{x}_t)\tau}, \tag{1}$$

where $a_0(\boldsymbol{x}_t) = \sum_{j=1}^{M} a_j(\boldsymbol{x}_t)$.

Based on the above *Next Reaction Density Function*,[10] the SSA creates random pairs (τ,j) via a Monte Carlo method, which answer the following two questions: "when will the next reaction occur?" and "Which reaction will occur next?"

The Direct Method is the original method proposed by Gillespie.[2,5] Starting from (1), we can write the probabilities $P(\tau|x, t)$ and $P(j|\tau, x, t)$ in the following form:

$$P(\tau|x, t) = a_0(\boldsymbol{x}_t) e^{-a_0(\boldsymbol{x}_t)\tau} \quad (\tau \geq 0), \tag{2}$$

$$P(j|\tau, x, t) = \frac{a_j(\boldsymbol{x}_t)}{a_0(\boldsymbol{x}_t)} \quad (j = 1, ..., M) \tag{3}$$

On each step of the simulation the random pairs (τ, j) are obtained based on the standard Monte Carlo inversion generating rules: Generate two uniform random numbers r_1 and r_2

from $U(0,1)$, the uniform distribution on $[0,1]$. τ is given by

$$\tau = \frac{1}{a_0(\boldsymbol{x}_t)} \ln\left(\frac{1}{r_1}\right). \tag{4}$$

The index j of the selected reaction is the smallest integer in $[1, M]$ such that

$$\sum_{j'=1}^{j} a_{j'}(\boldsymbol{x}_t) > r_2 a_0(\boldsymbol{x}_t). \tag{5}$$

Finally, the population vector X is updated by ν, and the simulation is advanced to the next reacting time.

3 Logarithmic Direct Method

Locating the reaction to fire next is the most expensive step of SSA. In all current SSA formulations, the search procedure accumulates the summation of $a_i(x)$ for $i = 1, \ldots, j$ by adding each propensity in turn until the sum is larger than the product of $a_0(x)$ and a uniform random number. The idea behind ODM and SDM is to reduce the average number of operations required to obtain the index of the next reaction to fire. This average is called the search depth.[8] The search depth of all current SSA algorithms is highly dependent on the biochemical system. For ODM and SDM, the search depth is O(M). LDM reduces the search depth to O($logM$).

On each step of DM, the total propensity a_0 is first calculated by adding all of the propensities a_i together. Then, in the selection of which reaction to fire next, the propensities a_i are accumulated by adding each propensity in turn until the partial sum is larger than the product of a_0 and a random number. Thus the propensities are summed almost twice. In the first step of LDM, we accumulate the partial sums of the propensities and store them in an array. With this ordered sequence of partial sums, we do a binary search to select the reaction to fire next. The basic idea of binary search is to divide the search interval in half

repeatedly,[11] so that each search step will select between two distinct alternatives. In the SSA, we find the position j such that $subtotal[j-1] \leq key < subtotal[j]$. The search key in LDM is the product of $a_0(x)$ and a uniform random number on $[0, 1]$. At the end of each LDM step, the ordered sequence of partial sums is updated from the point where the first partial sum has been changed.

LDM achieves its efficiency by locating the reaction to fire next via binary search, which is well-known to have an average search depth of $O(logM)$. For LDM, the average search depth is independent of the ordering of the reactions. Hence there is no need for a pre-simulation.

In our implementation, we also improve the efficiency of the system state update stage through the use of sparse matrix techniques. The stoichiometric matrix is sparse, since each reaction involves only a few species. The use of sparse matrix techniques can reduce memory accesses, and can eliminate the subtraction or addition of terms which will always be 0.

4 Simulation Results

4.1 Models

In this paper we use three models as test problems to compare the efficiency of different SSA formulations.

Phage-λ: The lysis-lysogeny decision circuit in phage-λ infected *Escherichia coli* is a well-characterized competitive regulatory mechanism. The core of the regulatory mechanism is the four-promoter, five-gene regulatory network.[4] A stochastic model has been formulated based on this phenomenon. The stochastic model demonstrates that the stochastic variations in the populations of two independently produced regulatory proteins function as a control switch to select the pathway. In our numerical experiments, we use the simplified model,[12]

obtained from Dr. Chris Myers at the University of Utah, which consists of 75 reactions and 61 species.

M.tuberculosis: The stringent response in M.tuberculosis is the set of metabolic and regulatory changes which has been found to be crucial for potential survival in the infected host. The model we use in our numerical experiments is from the Biocomputation Group at the University of Pennsylvania, which is a comprehensive model for the stringent response.[13] There are 23 reactions and 17 species in this model.

Pap switch: The Pap epigenetic switch is an inherently stochastic genetic switch.[14] This switch determines whether or not E. coli bacteria will express Pap pili, the hairlike surface structures which enable E. coli to attach to host cells and establish infections. The model is very complicated if it includes all the details. In our experiments we use only the small subset of the full Pap system from.[14] There are 8 reactions and 6 species in this model.

4.2 Results

All codes were implemented on an Intel(R) Pentium(R) 4 CPU 2.80GHz 2 processor with cache size 1024 KB and 2G memory, running a Linux environment. We used the programming language C and compiled with the optimization flags. The different formulations of SSA share the same main procedure, but use different functions for the propensities calculation, the selection of the next reaction to fire, the update of the system state, and the calculation of a_0. SDM and ODM have been implemented with the dependency graph as described in,[8,9] while LDM has been implemented without a dependency graph because there is no explicit need for it in this formulation of the algorithm. In the performance comparison, LDM denotes LDM without the sparse matrix update, and SLDM is LDM with the sparse matrix update. ODM and SDM are not using the sparse matrix update.

For each of the test problems, we found that the accuracy of the simulation by the various

formulations of the SSA were quite comparable, as is to be expected.

We didn't compare the performance with the original DM, since the performance of DM is not well-defined, i.e. it depends on the ordering of the reactions, and ODM is always faster than DM. To compare the performance of the four formulations: ODM, SDM, LDM and SLDM, we used the above three models: *Phage-λ* model, *M.tuberculosis* model, and *Pap switch* model.

Figure 1 shows the performance of ODM, SDM, LDM and SLDM for the three test problems. LDM performs better than ODM and SDM for all of the models, even for the small *Pap switch* model. LDM with sparse matrix state updating performs better than LDM with original matrix state updating. The benefits of both LDM and the sparse matrix update increase as the system size increases.

5 Conclusion

Gillespie's stochastic simulation algorithm (SSA) is in widespread use for stochastic simulation of biochemical systems. It will undoubtedly play a role in any multiscale algorithm for biochemical simulation. Thus it is important to formulate this method as efficiently as possible. In this paper we proposed a new SSA formulation, the Logarithmic Direct Method (LDM), with a logarithmic search depth for locating the next reaction. LDM reduces the computation time and avoids the pre-simulation step of previous SSA formulations.

FIGURE CAPTION Figure 1: Performance comparison (with error bars) of ODM, SDM, LDM and SLDM on the three test problems.

References

[1] D.W. Oxtoby and N.H. Nachtrieb. *Principles of Modern Chemistry*. Saunders College Publishing, 2nd edition, 1990.

[2] D.T. Gillespie. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *J. Comp. Phys.*, 22:403–434, 1976.

[3] H.H. McAdams and A. Arkin. Stochastic mechanisms in gene expression. *Proc. Natl. Acad. Sci. USA*, 94:814–819, 1997.

[4] A. Arkin, J. Ross, and H.H. McAdams. Stochastic kinetic analysis of developmental pathway bifurcation in phage λ-infected *e. coli* cells. *Genetics*, 149:1633–1648, Aug 1998.

[5] D.T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *J. Phys. Chem.*, 81:2340–2361, 1977.

[6] Y. Cao, D. Gillespie and L. Petzold. Multiscale stochastic simulation algorithm with stochastic partial equilibrium assumption for chemically reacting systems. *J. Comput. Phys.*, 206:395–411, 2005.

[7] M. Gibson, J. Bruck. Efficient exact stochastic simulation of chemical systems with many species and many channels. *J. Phys. Chem.*, 104:1876–1889, 2005.

[8] Y. Cao, H. Li and L. Petzold. Efficient formulation of the stochastic simulation algorithm for chemically reacting systems. *J. Chem. Phys.*, 121(9):4059–4067, 2004.

[9] J. M. McColluma, G. D. Peterson, C. D. Cox, M. L. Simpson, N. F. Samatova. The sorting direct method for stochastic simulation of biochemical systems with varying reaction execution behavior. *J. Comput. Biol. Chem.*, 2005.

[10] D.T. Gillespie. Approximate accelerated stochastic simulation of chemically reacting systems. *J. Chem. Phys.*, 115(4):1716–1733, 2001.

[11] http://www.nist.gov/dads/HTML/.

[12] H. Kuwahara, C. Myers, N. Barker, M. Samoilov, and A. Arkin. Asynchronous abstraction methodology for genetic regulatory networks. *To appear,* Transactions on Computational Systems Biology, 2006.

[13] http://www.cis.upenn.edu/biocomp/new_html/stringent.php3.

[14] B. Munsky and M. Khammash. The finite state projection algorithm for the solution of the chemical master equation. *J. Comput. Phys.*, 124, 2006.

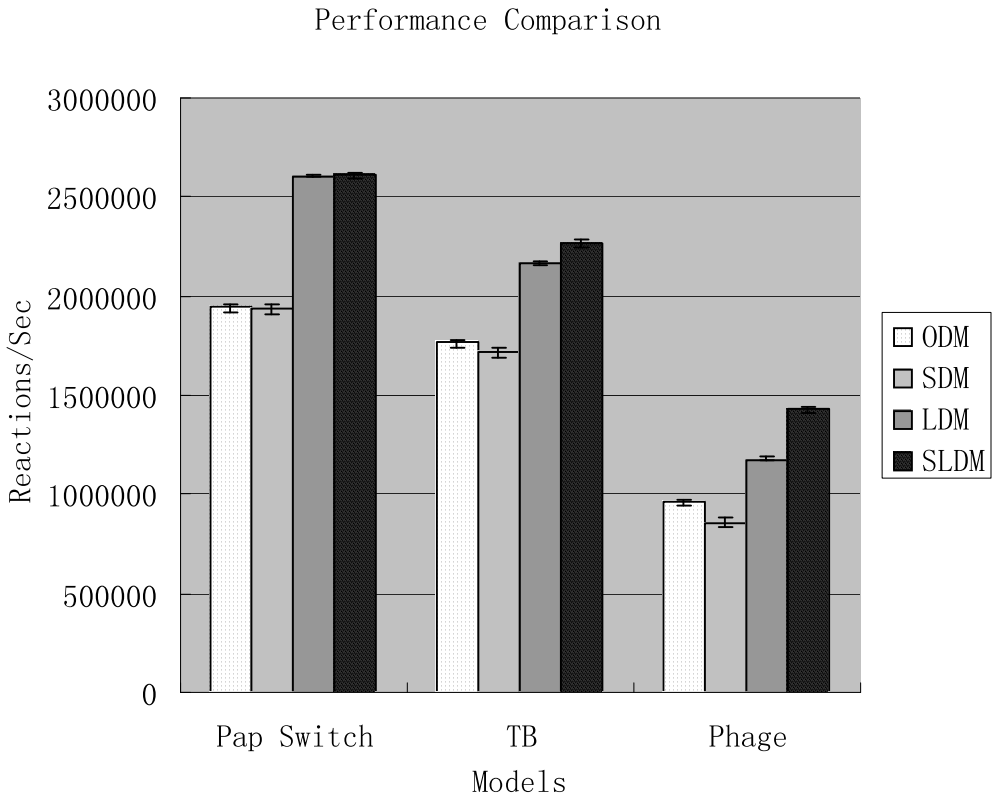
Performance Comparison
3000000
2500000
2000000
1500000
1000000
500000
0
Reactions/Sec
Pap Switch
TB
Phage
Models
ODM
SDM
LDM
SLDM

Figure 1: