A Parallel Implementation of Gillespie's Direct Method

Azmi Mohamed Ridwan, Arun Krishnan*, and Pawan Dhar

Bioinformatics Institute, 30 Biopolis Street, #07-01 Matrix, Singapore 138671.
{azmi,arun,pk}@bii.a-star.edu.sg

Abstract. Gillespie's Direct Method Algorithm (1977), is a well-known exact stochastic algorithm for simulating coupled reactions that requires the use of random numbers to calculate which reaction occurs next and when it occurs. However this algorithm is serial in design. For complex chemical systems, this will involve computationally intensive requirements with long simulation runs. This paper looks at decreasing execution times by attempting to parallelize this algorithm through splitting the computational domain into smaller units which will result in smaller computations and thus faster executions.

1 Introduction

Stochastic simulation has become an important tool for scientists in modeling complex chemical systems. Traditional methods of solving these systems usually involve expressing them mathematically through the use of ordinary differential equations which are notoriously difficult to solve. Gillespie's Direct Method [1] was a breakthrough in the sense that it could accurately and feasibly simulate these systems stochastically on what were then state-of-the-art computer systems. Since then, there have been improvements to the algorithm. One prominent recent modification is by Gibson[2].

The main disadvantage of Gillespie's algorithm is that it is essentially serial in nature. For complex chemical systems, this would result in computationally intensive executions and long simulation runs. The purpose of this paper is to study the feasibility of improving execution times through parallelization. The availability of compute clusters and parallel programming libraries (such as MPI, OpenMP and PVM) makes this possibility most attractive.

There are essentially two methodologies for achieving faster results. The first is known as **MRIP (Multiple Replication in Parallel)**[3]. The other method decomposes the problem domain into smaller sub-domains having fewer molecular species and having an instance of the Gillespie algorithm running. However there is a need to maintain the fundamental assumptions of the Gillespie algorithm while parallelizing in this manner. In this paper, we describe the procedure for using Domain Decomposition in order to parallelize Gillespie's Direct Method Algorithm. We will also show the application of the methods for a few chemical systems and the speedups obtained.

* To whom correspondence should be addressed

M. Bubak et al. (Eds.): ICCS 2004, LNCS 3037, pp. 284–291, 2004.

2 Methodology

2.1 Gillespie's Direct Method Algorithm

This section will briefly highlight the important aspects of Gillespie's algorithm. The reader is encouraged to read [1] for a more detailed description and proofs. The Gillespie algorithm is impressive in its simplicity. The algorithm begins by initializing the stochastic rate constants for the reactions and the initial populations of the various species. A loop is then started with the following steps. First, the probability of each reaction occuring at the current time is calculated. Then, random numbers are used to determine which reaction should occur as well as to calculate the next time step. The time is then incremented and the species' population are adjusted according to the reaction selected. Finally, the loop repeats itself until stopping criterions are met.

2.2 Data Collection

One of the primary concerns in writing a parallel version of the Gillespie algorithm is the collation of the data. This occurs as a result of the use of random numbers in the algorithm. The implementation of random numbers in computer programs is almost always pseudo-random which requires an initial seed. Thus for proper solutions, each instance of the program must use a unique initial seed. However this will mean that each instance will have a unique time evolution. This then implies that in all probability, there will be no corresponding data points for any of the instances for a specific time. One simple solution, termed here as 'nearest point', would be to use the various species population at the point of the latest reaction before that collection point.

2.3 Domain Decomposition (DD) Method

The Domain Decomposition method involves dividing the entire species into smaller independent populations. The fundamental assumption for Gillespie's algorithm is that for a fixed container of volume V, the system should be in thermal equilibrium thus implying that the molecules will at all times be distributed randomly and uniformly. It remains to be seen whether the Domain Decomposition method would lead to incorrect results due to a violation of this fundamental assumption.

To develop the Domain Decomposition method, the Gillespie algorithm needs to be examined. Although for the most part the algorithm remains unchanged, there is a need to reexamine the rate constants. While the deterministic rate constants k_i are assumed to be constant, the stochastic rate constants are not necessarily so. To illustrate this, we list general relationships between the deterministic (k_i) and stochastic (c_i) rate constants. The relationships are obviously dependent on the type of reaction involved. When the molecular species are divided by the number of sub-domains, the respective stochastic rate constants must be adjusted accordingly to maintain constant deterministic rate constants.

$$\sum_{\mathbf{j=1}}^{\mathbf{N}} \mathbf{S_j} \rightarrow \qquad\qquad \mathbf{NS_1} \rightarrow$$

$$k_i = V^{j-1} c_i \qquad\qquad k_i = \frac{V c_i}{N!}$$

2.4 Domain Decomposition Method with Synchronization (DDWS)

As stated previously, the fundamental assumption of the Gillespie algorithm is the fact that the system in the volume is supposed to be well-mixed. However the DD method could violate this if the sub-simulations produce large differences in the population of a species. Hence in order to improve the accuracy of the parallel Gillespie algorithm, there is a need to introduce some form of interaction between the sub-domains. Schwehm [4] implements this by randomly exchanging molecules between neighboring sub-domains. This form of diffusion is motivated by the way partial differential equations are solved numerically. This implementation however is very costly as large numbers of point-to-point messages must be used.

A simpler method of averaging out the species' populations at the appropriate step is used here. This would be more in line with the spirit of Gillespie's original algorithm. In the implementation of this algorithm, synchronizations are done at regular time intervals (in fact, in the same step when the population data are collected). This is easy to implement as the number of synchronizations done would be constant regardless of the total number of iterations for the Gillespie loop.

3 Chemical Reactions

To illustrate the parallel algorithms, we have chosen two types of chemical reactions: One whose simulations produces asymptotic results at steady states and the other that produces periodic results.

3.1 Michaelis-Menten Reactions

The Michaelis-Menten system is a set of well-known, enzyme catalyzed reactions that involves the binding of a substrate to an enzyme. These reactions are given below. The Michaelis-Menten system is an example of a 'deterministic' system.

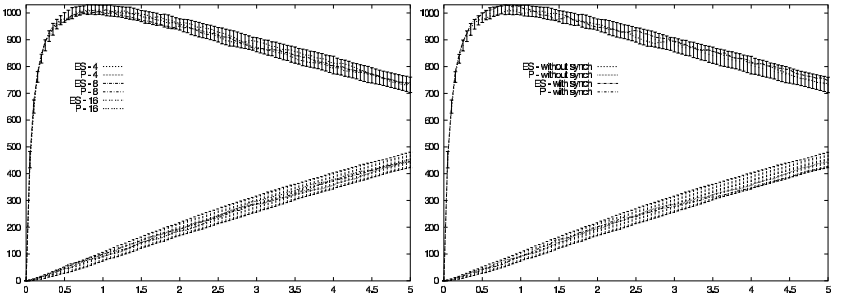
$$E + S \xrightarrow{k_1} ES \tag{1a}$$

$$ES \xrightarrow{k_2} E + S \tag{1b}$$

$$ES \xrightarrow{k_3} E + P \tag{1c}$$

For the implementation of the DD method, the rate constants must be modified as stated in Sect. 2.3. Looking at the MM equations (1b) and (1c), the

deterministic and stochastic reaction rate constants are directly related. Thus if the volume is divided into its sub-domains, the rate constants for these equations will remain unchanged. For (1a) the deterministic and stochastic rate constants are related by a volume factor. Therefore if the volume is divided into N sub-domains, then the stochastic rate constants must be increased by the same factor.



(a) Comparison of solutions from 3 DD plots with 4, 8 and 16 sub-domains for species ES and P.

(b) Comparison of a DD solution and a DDWS solution.

Fig. 1.

Figure 1a compares the results for a multiple replication (50 runs) simulation of the Serial algorithm (denoted by the verical error bars) and the DD method (with 4,8 and 16 sub-domains) for two of the molecular species. It can be seen that the DD method holds up well for these sets of reaction even for 16 sub-domains (with initial enzyme and substrate populations of 75 molecules each.) It would be difficult to distinguish the DD method results from those of the serial runs. Figure 1b shows a comparison between the serial solution , the DD method and the DDWS method (both of which uses the same initial random seeds). As can be seen, the addition of the synchronization does not lead to a qualitative difference in the results of the simulation.

3.2 Lotka Reactions

The Lotka Reactions[1], are an example of oscillatory reactions. It is a well-known system that has been adopted in many branches of science, most notably in ecology where it represents a crude predator-prey model.

$$\bar{X} + Y_1 \xrightarrow{k_1} 2Y_1 \tag{2a}$$

$$Y_1 + Y_2 \xrightarrow{k_2} 2Y_2 \tag{2b}$$

$$Y_2 \xrightarrow{k_3} Z \tag{2c}$$

Note that in the first reaction, the bar over the X indicates that X is *open* i.e. the molecular population level of this species is assumed to be constant. Different instances of the Direct Method will yield similar frequencies but they will be out of phase with each other. Also the amplitude variations may differ significantly.

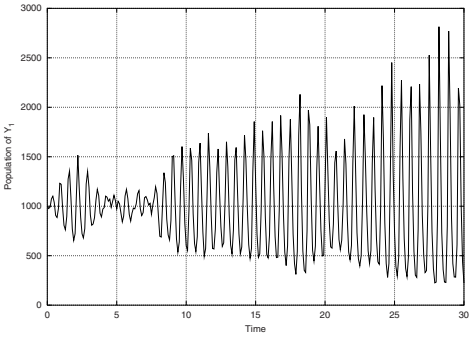
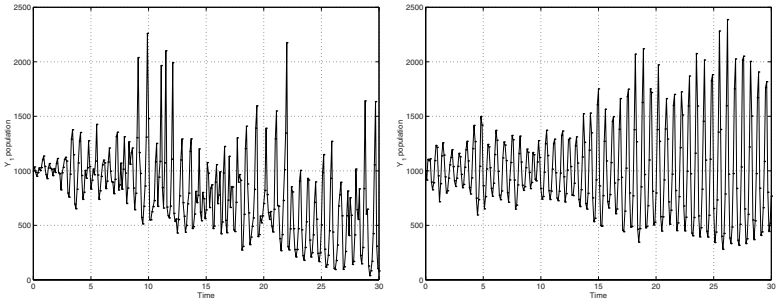


Fig. 2. Plot of species Y_1 for reaction (2) using the Serial implementation of the Direct method with $\bar{X} = 100, Y_1 = Y_2 = 1000, Z = 0, c_1X = 10, c_2 = 0.01, c_3 = 10$.

Figure 2 shows a plot of species Y_1 for a serial run . The Lotka reactions have steady-state solutions for Y_1, Y_2 at $Y_{1s} = c_3/c_2$ $Y_{2s} = c_1X/c_2$. Figure 2 demonstrates this with Y_1 oscillating around $Y_{1s} = c_3/c_2 = 1000$.

Figure 3a shows a plot of species Y_1 for the DD method applied to the Lotka reactions for 4 sub-domains. The solutions obtained are clearly incorrect as Y_1 does not oscillate around the steady-state solutions. To understand the reason for this, we must take note of what occurs in each sub-domain. As stated in Sect. 2.3, when the DD method is used, the stochastic rate constants must be modified appropriately. This results in a smaller steady-state solution for Y_1, Y_2 in each sub-domain. The oscillations will then occur around these values. If the value of Y_1 were to reach 0, only reaction (2c) is viable. Thus in the sub-domain method, imbalances may occur where some sub-domains may be void of any Y_1 and Y_2 species.

Figure 3b is an implementation of the Lotka reaction using the DDWS. The figure suggests that the correct solution has been derived as it resembles a serial solution (i.e. the solution oscillates around the steady-state value). This solution works because, synchronizations prevent the Y_1 species from being extinct in any one sub-domain (provided there is a nonzero population of Y_1 in any of the sub-domains).



(a) Plot of species $Y_1$for reaction (2) using the DD method.

(b) Plot of species Y_1 for reaction (2) using the DDWS

Fig. 3.

3.3 Brusselator Reactions

The Brusselator[1] is another set of well known reactions that represents oscillatory systems. Unlike the Lotka reactions previously, it is 'positively stable' and the amplitude of oscillations are more consistent with each other. The reactions can be expressed as:

$$\bar{X}_1 \xrightarrow{k_1} Y_1 \tag{3a}$$

$$\bar{X}_2 + Y_1 \xrightarrow{k_2} Y_2 + Z_1 \tag{3b}$$

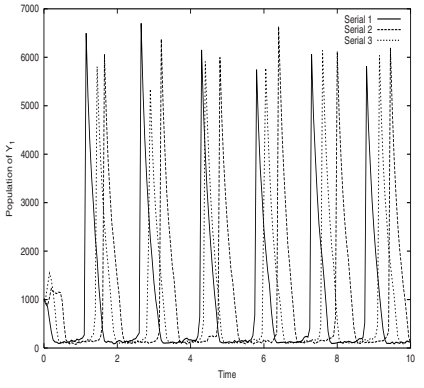
$$2Y_1 + Y_2 \xrightarrow{k_3} 3Y_1 \tag{3c}$$

$$Y_1 \xrightarrow{k_4} Z_2 \tag{3d}$$

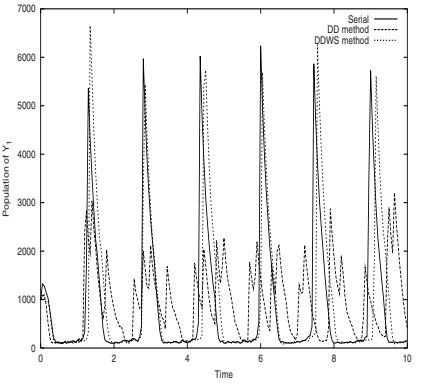
The serial plots (Fig. 4a) of the Brusselator reactions show that while the periods and amplitudes of the oscillations for the three plots are similar, the phases are not. As such when the DD method is used, the population of the species Y_1, Y_2) between the sub-domains are out of phase with each other, resulting in clearly inaccurate results (Fig. 4b - long dashed line). However once synchronizations are used, the solution obtained appears to be consistent with a serial solution of the reactions(Fig. 4b - short dashed line).

4 Performance Results

Figure 5 shows speedup graphs for the Michaelis-Menten reactions with two different sets of intial values together with the Brusselator reaction simulation using the values in Fig. 4. For the MM simulations, 'small' corresponds to: $E = 12000, S = 12000, ES = 0, P = 0, c_1 = 0.01, c_2 = 1, c_3 = 100$ while 'large' corresponds to: $E = 36000, S = 36000, ES = 0, P = 0, c_1 = 0.01, c_2 = 250, c_3 = 1$.



(a) The Brusselator plot of species Y_1 for 3 Serial runs with initial populations as $Y_1 = 1000, Y_2 = 2000, c_1X_1 = 5000, c_2X_2 = 50, c_3 = 0.00005, c_4 = 5$.



(b) The Brusselator plot of species Y_1 for Serial, DD method, DDWS

Fig. 4.

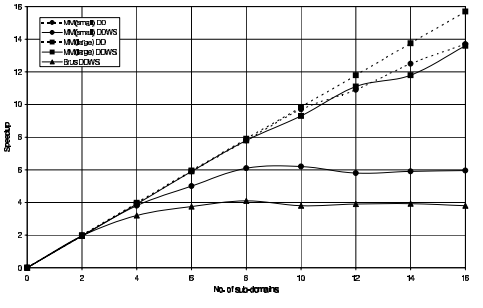


Fig. 5. Speedup for Michaelis Menten and Brusselator reactions.

It is quite apparent from the comparison of smaller and larger initial values for the MM reactions that, while the reactions remain the same, the speedup graphs differ significantly. The speedup for the MM reactions thus depend on the total number of iterations of Gillespie loop which in turn depends on the initial population values and rate constants used. The speedup for the DD method is much better than the DDWS method. However as the number of sub-domains are increased, the speedup will plateau and eventually decrease as the computation done in each sub-domain decreases.

This plateauing in the speedup is more apparent when synchronizations are introduced. This results in a constant number of synchronizations regardless of the initial population of the species and rate constants. As the number of sub-

domains increases, the number of iterations between synchronization decreases. Also the collective operation used ($MPI\_Allreduce$), has to handle an increasing number of processes thus increasing the cost of using it.

As a comparison, the speedup for the Brusselator equations using the values used previously are also shown. As stated before, a correct solution is only derived when synchronizations are used. The speedup shows the inevitable plateauing as the number of sub-domains is increased.

5 Summary

In this paper we have presented an approach to parallelizing Gillespie's Direct Method algorithm keeping in mind the need to remain consistent with the fundamental assumptions of the algorithm.The basic premise of the DD method is to divide the molecular population into smaller sub-domains where computations can be completed faster.

For oscillatory chemical systems, such as the Lotka reactions and the Brusselator, periodic synchronizations are needed. This introduces diffusion which prevents buildup of any particular species in a sub-domain thus ensuring the well-mixed nature of the whole domain.

The speedups obtained by the parallel Gillespie Algorithm show a "plateauing" effect in the presence of synchronizations (DDWS method). The DD method, without synchronizations shows very good speedup; however, its use is restricted to non-oscillatory systems.

Despite the fact that the methodology works for the systems under study here, it is not possible to state categorically as to whether it would work for any arbitrary system. Work remains to be done to study the efficacy of this method on larger, more highly coupled systems.

References

1. Gillespie, D.T.: Exact stochastic simulation of coupled chemical reactions. J. Phys. Chem. **81** (1977) 2340–2361
2. Gibson, M.A.: Computational methods for stochastic biological systems. Ph.D. Thesis, Calif. Inst. Technology (2000)
3. Ewing, G., McNickle, D., Pawlikowski, K.: Multiple replications in parallel: Distributed generation of data for speeding up quantitative stochastic simulation. In: Proc. of IMACS'97, 15th Congress of Int. Association for Mathematics and Computers in Simulation, Berlin, Germany (1997) 397–402
4. Schwehm, M.: Parallel stochastic simulation of whole-cell models. In: ICSB 2001 proceedings. (2001)