Scaling the Gillespie Stochastic Simulation Algorithm Using Data-Parallel Architectures

Jose-Juan Tapia-Valenzuela and Roshan M. D'Souza
Dept. of Mechanical Engineering-Engineering Mechanics
Michigan Technological University
Houghton, MI, 49931
Email: jjtapiav@mtu.edu, rmjdsouza@gmail.com

May 8, 2009

Abstract

The Gillespie Stochastic Simulation Algorithm (SSA) is a stochastic equation simulation technique that generates a statistically correct solution of chemical reaction networks. Traditional coupled ordinary differential equation approaches to model reaction networks rely on bulk properties and assume interaction of millions of molecules. Therefore, they cannot predict the trajectory of reactants when the number of molecules is so low that the continuum assumption does not hold. The Gillespie SSA allows the explicit simulation of every reaction in the network and therefore can correctly handle systems with low molecule count.

In [1], Dan Gillespie introduced two implementations of this algorithm, the Direct Method (DM) and the First Reaction Method (FRM). The DM works, in broad terms, by calculating the probability each reaction has of occurring, and firing the one with the highest probability. The FRM works by calculating the reaction time each reaction will take to fire, and firing the one with the lowest waiting time.

Both the DM and the FRM are computationally very expensive. The algorithm execution time possesses a very high correlation to the number of reactions and reactants in the system. This is because most of the steps of the Gillespie algorithm are centered around operations such as finding the most probable reaction, or finding the reaction that will take place in the least time.

A number of alternatives and improvements on the algorithm have been proposed. One of such was presented by Dan Gillespie himself in [2], in the form of an approximate equivalent of his model, the τ-leaping method. This method sacrifices accuracy for performance by advancing in 'leaps' of time, which results in the firing of various reaction channels simultaneously. A certain amount of independence is assumed between different reaction channels even if they involve common reactants or products. While each step becomes more computationally expensive, this approach greatly reduces the number of steps that are necessary for a given simulation when compared with the exact implementations, the overall balance being greatly advantageous for the approximate method.

However, even these improvements cannot beat the inevitable slowdown the algorithm faces when dealing with large simulations with 10e5 to 10e6 reaction channels. An approach that solves or at least diminishes the influence the number of reactions and reactants have on the computational complexity of the algorithm is essential.

Stream computing presents a paradigm that can address many of these problems. Most of the Gillespie algorithm can be understood as an extension of the scan and reduction problem, which is one of the cornerstone operations of stream computing. As such, reworking the algorithm into this architecture presents a great opportunity for improvement and speedup.

To the best of our knowledge, there is one previous effort at using data-parallel computing for computing the Gillespie SSA by Li and Petzold in [3]. Their approach consisted on running many simultaneous, independent simulations of the same problem. While this certainly helped in speeding up statistical benchmarks of the algorithm, this does not constitute a true parallelization of the problem, and as such they are not really exploiting the full potential of the architecture.

We have implemented the three versions of the Gillespie algorithm described earlier on a stream computing platform, the Graphics Processing Unit (GPU). The implementation was done in such a way that every step of the algorithm was performed in a fully parallel way, either through the utilization of hierarchical scans, parallel sort and reductions, or through some specialized operation such as the implementation of a parallel Poisson random number generator. Preliminary results show a significant speedup of the parallel versions when compared to their serial counterparts when the simulation starts growing and including a larger number of reaction channels. First benchmarks demonstrate that the algorithm can easily deal with as many as 100,000 reactions in a single system, outperforming its serial equivalent by a complete order of magnitude.

References

[1] D. T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *The Journal of Physical Chemistry*, 81(25):2340–2361, 1977.

[2] D. T. Gillespie. Approximate accelerated stochastic simulation of chemically reacting systems. *The Journal of Chemical Physics*, 115(4):1716–1733, 2001.

[3] H. Li and L. Petzold. Efficient stochastic simulation of biochemical systems on the graphics processing unit. In *Proceedings of the 13th SIAM Conference on Parallel Processing for Scientific Computing*, 2007.