# Stochastic Simulation Algorithm by Tau-Leaping

Shaila Mae Choa, Aakov Nikolei Dy, Yuki Enomoto, Jaylica Anne Tan
De La Salle University
2401 Taft Ave, Malate
Manila, 1004 Metro Manila
shaila_choa@dlsu.edu.ph, aakov_dy@dlsu.edu.ph, yuki_enomoto@dlsu.edu.ph,
jaylica_tan@dlsu.edu.ph

## ABSTRACT

This paper discusses Stochastic Simulation Algorithm (SSA) and Tau-Leaping that have been applied on simulating biochemical system formed by living cells. The efficiency and accuracy of these algorithms are compared.

## Keywords

algorithm; simulation; SSA; Tau-Leaping;

## 1. INTRODUCTION

The traditional deterministic approaches are although, sufficient for most systems, some cannot successfully compare the natural stochasticity of some biochemical system formed by living cells. The small population of a few critical reactant species can cause the behavior of the system to be discrete and stochastic. The dynamics of those kinds of systems can be simulated accurately by using a stochastic equation, specifically the Gillespie Stochastic Simulation Algorithm (SSA). For many realistic biochemical systems, the computational cost of simulation by the SSA can be very high. As computers have become faster, the algorithm has been used to simulate increasingly complex systems. The algorithm is particularly useful for simulating reactions within cells where some key reactant molecules may be present in small numbers[4]. It is used heavily in the field of computational systems biology. Significant amount of works have been focused on speeding up the SSA by reformulating the algorithm [6]. One of the most known strategy is the Tau-Leaping method, proposed by Gillespie as an approximate simulation strategy [4].

This paper is organized as follows. In section 2 we briefly review the Stochastic Simulation Algorithm and Tau-Leaping. A comparison of the simulation of SSA and Tau-Leaping is shown in section 3. In section 4, we present a sample implementation of SSA, and in section 5 we draw some conclusions.



© 2018 ACM. ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

## 2. BACKGROUND

### 2.1 Stochastic Simulation Algorithm

Gillespie Stochastic Simulation Algorithm (SSA) is a method for producing trajectories of finite well-mixed populations in continuous time that are statistically correct. The generated trajectory is stochastic that would only be gained by solving differential equations [7].

The SSA acquires a population consisting of a limited number of individuals scattered over a finite set of discrete states. As a result of reactions between interacting states, the number of individuals in each state changes. The time evolution of the state vector $X(t) \equiv (X_1(t), ..., X_N(t))$ is produced by SSA given an initial population state $X(t_0)$ and initial time $t_0$ where $X_i(t)$, $i = 1, ..., N$, is the population size of state $i$ at time $t$ with $N$ number of states. Through $M$ reactions $R_j$ where $j = 1, ..., M$ denotes the $j$th reaction, the states interact. Reaction takes place if the population size of at least one state instantaneously changes. There are two quantities that distinguishes each reaction $R_j$. Its state-change vector $j = ({}_1j, ..., {}_Nj)$ is the first quantity, where ${}_ij$ is the population change in state $i$ caused by one $R_j$ reaction. The second one is its propensity function $a_j(x)$ which is the probability of one $R_j$ reaction happening in the very extremely small time interval $[t, t + dt)$ [7].

The simplest procedure for constructing exact numerical realizations of the SSA is the direct method by Gillespie. From the uniform distribution in the unit interval, two random number $r_1$ and $r_2$ are drawn. The time step is defined as $\tau = \frac{1}{a_0(X)} ln(1/r_1)$ where $a_0(X) = \sum a_j(X)$ and the index of the next reaction to execute $R_j$ is the smallest integer $j$ satisfying $j = \sum_{i=1}^{j} a_1(x) > r_2 a_0(x)$. By replacing $t \leftarrow t + \tau$ and $x \leftarrow x + v_j$ the reaction is done.

The advantages of SSA algorithm is that the results are exact. Because the results are exact and computes each reaction one at a time, it uses more computing power thus making the process too slow if there are a large number of reactions or large population sizes and will take a very long time to complete its process [2]. SSA is numerically exact; however, it is necessarily inefficient for most practical applications [7].

### 2.2 Tau-Leaping

The Tau-Leaping method has been proposed to improve the efficiency of implementation of the SSA [3]. By using a Poisson random numbers, the Tau-Leaping method can leap over many reactions and approximate the stochastic behavior of the system very well [4]. The Tau-Leaping

method provides a natural bridge between the SSA in the discrete stochastic regime and the explicit Euler method for the chemical Langevin equation (CLE) in the continuous stochastic regime and to the reaction rate equation (RRE) in the continuous deterministic regime [3]. In this sense, the Tau-Leaping method is ideal for multiscale stochastic simulation.

Unlike in SSA, reactions in Tau-Leaping are not simulated individually per time step. Instead, the Tau-Leaping method can proceed with many reactions in each step [3]. There are so called "leap" or "accelerated" methods which are described by the leaps over many reactions of a time step. Under the right conditions, the increase in speed of several orders of magnitude can be achieved. An important assumption in all accelerated SSA procedures is that the Leap Condition is satisfied. The Leap Condition requires that the time leap be small enough that the change in the propensity functions are negligible [8]. In other words, $aj(x) \approx constant$ in $[t, t+\tau]$ for all j. Three approximate SSA methods are included in this paper: the explicit tau-leap method (ETL); binomial tau-leap method (BTL); and optimized tau-leap method (OTL).

### 2.2.1 *Explicit Tau-Leaping (ETL)*

Explicit tau-leaping method is based on the Tau-Leaping algorithm that uses Poisson distribution. This method aims to use a larger time step than SSA while bounding the local error within the tolerance. Although the ETL procedure can significantly speed up the simulation, it is highly sensitive to the error control mechanisms and tolerance. It sacrifices the accuracy of the results in order to process them faster, thus it is also not good for stiff models [2]. Additionally, due to the unbounded nature of Poisson random variables and the lack of coordination between reactions during a time step, the population sizes of individual states can become negative [8].

### 2.2.2 *Binomial Tau-Leaping (BTL)*

Binomial tau-leaping method was introduced to address the issue of negative population sizes arising in the ETL method, which was developed by Tian (2004) and Chatterjee et. al. (2005) [8].

### 2.2.3 *Optimized Tau-Leaping (OTL)*

Optimized tau-leaping method is another Tau-Leaping algorithm that uses time increment of the simulation steps in an adaptive manner. Its advantages is that it cannot have a negative population as oppose to the Poisson Tau-Leaping method. It is also more accurate than fixed-step tau-leap methods because it can switch to the direct method when the leap method is violated and cannot be used. Because of this, optimized tau-leaping is a flexible algorithm that can adapt in various methods as the simulation progresses [9].

## 3. SIMULATION

Based on the parameters that Gillespie, Daniel T. (2001) has given, MathWorks has given a simulation to compare SSA Algorithm and Tau Leaping Algorithm. Based on Figure 1, the simulated plots are shown in solid lines for the SSA, and hollow circled lines for the Tau Leaping algorithm. Based on the final results, the number of steps took for SSA Algorithm is approximately 616,010 while the steps took for Tau Leaping Algorithm is approximately 620, which has a huge difference when it comes to optimization [5] [1].

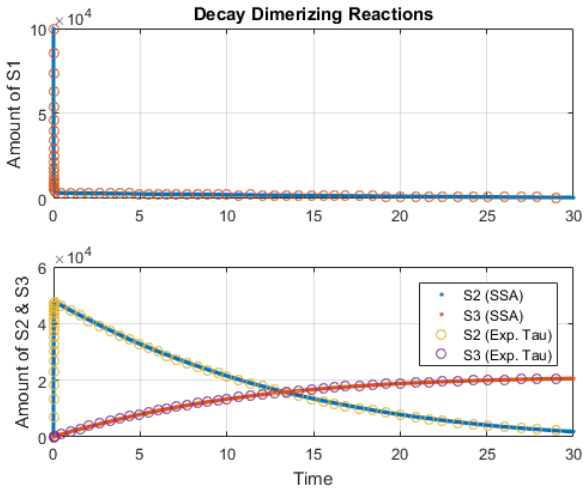


**Figure 1: Comparison of SSA Algorithm (solid line) and Tau-Leaping Algorithm (hollow circles) [1]**

## 4. IMPLEMENTATION

### 4.1 SSA and Tau-Leaping Example

Chemical kinetics is the study of chemical reactions with respect to reaction rates, effect of different variables, rearrangement of atoms, formation of intermediates etc. On the other hand, Kinetics is the study of motion. It came from the Greek word kinesis, meaning movement.

We want to be able to see the amounts reacted, formed, and the rates of their formation. In the following example by Cao et. al. the system involving N molecular species $S_1, ..., S_N$. The state vector is denoted by $X(t) = (X_1(t), ..., X_N(t))$, where $X_i(t)$ is the number of molecules of species $S_i$ at time $t$. $M$ reaction channels $R_1, ..., R_M$ are involved in the system. Assuming that the system is at thermal equilibrium which means there are no heat present in their state. The dynamics of reaction channel $R_j$ is characterized by the propensity function $a_j$ and by the state change vector $_j = (_1j, ..., _Nj) : a_j(x)dt$ gives the probability that one $R_j$ reaction will occur in the next following time interval $[t, t+dt)$, and $_ij$ gives the change in the Si molecular population induced by one Rj reaction. The dynamics of the system obeys the chemical master equation (CME):

$$\frac{\partial P(x,t|x_0,t_0)}{\partial t} = \sum_{j=1}^{M}[a_j(x-\nu_j)P(x-\nu_j,t|x_0,t_0) - a_j(x)P(x,t|x_0,t_0)], \quad (1)$$

where the function $P(x, t|x0, t0)$ denotes the probability that $X(t)$ will be x, given that $X(t_0) = x_0$. The CME is hard to solve, both theoretically and numerically. An equivalent simulation method is the SSA. Let . The time $\tau$ to the next occurring reaction is the exponentially distributed random variable with mean $1/a_0(x)$. The index $j$ of that reaction is the integer random variable with probability $a_j(x)/a_0(x)$. SSA is a kinetic Monte Carlo method based on these distributions. On each step, SSA generates two random numbers

$r1$ and $r2$ in $U(0,1)$, the uniform distribution on the interval (0, 1). The time for the next reaction to occur is given by $t+\tau$ , where $\tau$ is given by

$$\tau = \frac{1}{a_0(x)} \log(\frac{1}{r_1}). \quad (2)$$

The index $j$ for the next reaction is given by the smallest integer satisfying

$$\sum_{l=1}^{j} a_l(x) > r_2 a_0(x). \quad (3)$$

The system states are updated by $X(t+\tau) = X(t) + {}_j$ . The simulation proceeds to the next occurring time, until it reaches the final time. In principle, the SSA could be used to simulate all of the chemical species and reactions, except that because it must proceed one reaction at a time, it is much too slow for most practical problems.

Gillespie proposed a scheme called Tau-Leaping to accelerate the SSA. The basic idea of the Tau-Leaping method is to ask, "How many times does each reaction channel fire in each subinterval?"In each step, the tau-leaping method can proceed with many reactions. This is achieved at the cost of some accuracy. Define

$K_j(\tau; x, t)$ = the number of times, given $X(t) = x$, that reaction channel $R_j$ will fire94 in the time interval $[t, t+\tau)(j = 1, ..., M)$. (4)

In performing Tau-leaping, it assumes the condition that ÏĎ should be small enough that the change in the state during $[t, t+\tau)$ will be so small that no function will suffer an appreciable change in its value. $K_j(\tau; x, t)$ is then well approximated by the Poisson random variable with mean and variance aj $(x)\tau$ .

$$K_j(\tau; x, t) = P(a_j(x)\tau) \quad (j = 1, \ldots, M). \quad (5)$$

The basic tau-leaping method proceeds as follows: Choose a value for $\tau$ that satisfies the Leap Condition. Generate for each $j = 1, ..., M$ a sample value $k_j$ of the Poisson random variable $P(a_j(x)\tau)$, and update the state by

$$X(t+\tau) = x + \sum_{j=1}^{M} k_j \nu_j. \quad (6)$$

The implicit tau formula has been proposed to handle the stiffness. Using implicit tau formula allows much larger step sizes than the explicit tau formula, when applied to stiff stochastic systems. Using different kind of Tau formula will give you a better accuracy when applied to some stiff problems but the implementation details are similar.

## 5. CONCLUSION

Both SSA algorithms and Tau-Leaping algorithms have its own advantages and disadvantages. One of the advantages of SSA is that its results are more accurate as oppose to Tau-Leaping, however, it is not recommended to be used in processing a large number of reactions. Tau-Leaping is more efficient when it comes to processing the results and can be reliable in processing huge data, although it sacrifices the accuracy of results to compensate for the running time. Even though the results are not accurate, people are coming up with ways to make the Tau-Leaping as accurate and as efficient as possible (e.g. ETL, OTL, and BTL).

## 6. REFERENCES


[1] Comparing ssa and explicit tau-leaping stochastic solvers. http://www.mathworks.com/help/simbio/examples/comparing-ssa-and-explicit-tau-leaping-stochastic-solvers.html?requestedDomain=www.mathworks.com.

[2] Stochastic solvers. http://www.mathworks.com/help/simbio/ug/stochastic-solvers.html.

[3] Y. Cao, D. T. Gillespie, and L. R. Petzold. The adaptive explicit-implicit tau-leaping method with automatic tau selection. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.137.7316\&rep=rep1\&type=pdf".

[4] Y. Cao, D. T. Gillespie, and L. R. Petzold. Efficient step size selection for the tau-leaping simulation method. Technical report, THE JOURNAL OF CHEMICAL PHYSICS, January 30 2006.

[5] D. T. Gillespie. Approximate accelerated stochastic simulation of chemically reacting systems, 2001.

[6] H. Li and L. R. Petzold. Stochastic simulation of biochemical systems on the graphics processing unit. https://engineering.ucsb.edu/ cse/Files/SSA-GPU07.pdf, 2005.

[7] Nezar. Gillespie stochastic simulation algorithm. https://www.mathworks.com/matlabcentral/fileexchange/34707-gillespie-stochastic-simulation-algorithm?requestedDomain=www.mathworks.com, January 20 2012.

[8] M. Pineda-Krch. Adaptive tau-leaping in ssa. https://www.r-bloggers.com/adaptive-tau-leaping-in-ssa/, July 26 2007.

[9] M. Pineda-Krch. Gillespiessa: Implementing the stochastic simulation algorithm in r. http://epbi-radivot.cwru.edu/EPBI473/files/week5BioBasedRiskModeling/cranGi April 2008.