Journal of Computational Physics 228 (2009) 5908–5916

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

D-leaping: Accelerating stochastic simulation algorithms for reactions with delays

Basil Bayati, Philippe Chatelain, Petros Koumoutsakos *

Chair of Computational Science, ETH Zurich, CH-8092, Switzerland

ARTICLE INFO

Article history:
Received 20 December 2008
Received in revised form 26 April 2009
Accepted 4 May 2009
Available online 12 May 2009

Keywords:
Stochastic simulation
Delayed reactions
Tau-leaping
R-leaping

ABSTRACT

We propose a novel, accelerated algorithm for the approximate stochastic simulation of biochemical systems with delays. The present work extends existing accelerated algorithms by distributing, in a time adaptive fashion, the delayed reactions so as to minimize the computational effort while preserving their accuracy. The accuracy of the present algorithm is assessed by comparing its results to those of the corresponding delay differential equations for a representative biochemical system. In addition, the fluctuations produced from the present algorithm are comparable to those from an exact stochastic simulation with delays. The algorithm is used to simulate biochemical systems that model oscillatory gene expression. The results indicate that the present algorithm is competitive with existing works for several benchmark problems while it is orders of magnitude faster for certain systems of biochemical reactions.

1. Introduction

Several reactions in eukaryotic cells are not instantaneous, but rather their reactants are subject to transcription, processing, and synthesis before they can react with other chemical species [20,19]. These biochemical processes can be modeled via a system of reactions with delays. Delayed reactions can also be used as models of spatially dependent stochastic processes [9] when proteins may need to diffuse to a distant compartment in the cell in order to react with other proteins.

These reactions can be formulated as a continuous-time Markov process with a discrete set of states that can be expressed by the so-called *Master Equation* (ME) [15,11]. Exact realizations of the ME can be obtained via the *Stochastic Simulation Algorithm* [5,10,12] (SSA). The connections between SSA and Molecular Dynamics, as well as the classical Langevin, Fokker–Planck, and reaction-rate equations, have been recently reviewed in [14]. The SSA is exact since it independently simulates all reaction events but it can be computationally expensive for large systems.

In order to accelerate the SSA, several approximate algorithms have been proposed. These algorithms accelerate the SSA by either prescribing a larger time-step [13,22,7,9,8] or the number of reactions per time-step [2]. Recently, there has been interest in extending the SSA to incorporate delays. Delays in the stochastic process render it, by definition, non-Markovian, and suitable modifications to the SSA are necessary in order to produce the correct dynamics [6,3,1]. Cai [6] and Anderson [1] have proposed *exact, delayed SSAs* and, additionally, Leier et al. [16] have developed a *delayed, accelerated, approximate, SSA* (DAA-SSA).

In this paper, a time-adaptive generalization of DAA-SSA is proposed (D-leaping). D-leaping is shown to converge to the *Delay Differential Equation* (DDE) and preserve the correct statistical fluctuations. Furthermore, the algorithm is *adaptive* in time, and is shown to be, for certain chemical reactions, orders of magnitude faster than the algorithm presented in [16]. The D-leaping algorithm can be combined with *R-Leaping* [2] and τ-*Leaping* [13] as described in this work.

* Corresponding author. Tel.: +41 1 632 5258; fax: +41 1 632 1703.
E-mail address: petros@ethz.ch (P. Koumoutsakos).

0021-9991/$ - see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2009.05.004

This paper is organized as follows: Section 2 discusses continuum chemical kinetics and defines the reaction-rate and delay differential equations. Section 3 summarizes stochastic chemical kinetics as well as algorithms for incorporating delays. Section 4 introduces D-leaping and Section 5 presents numerical results and comparisons with other work. The work is summarized in Section 6.

2. Continuum chemical kinetics and delay differential equations

Continuum models of chemical kinetics systems with large numbers of molecules describe the evolution of concentrations with respect to time, instead of the evolution of discrete species. The concentration is defined as $\chi_i(t) := X_i(t)/\Omega$, where $X_i(t)$ is the number of molecules of species i in the volume Ω at time t. Under the assumption of a well stirred chemical system in thermal equilibrium, the evolution of concentrations can be expressed as

$$\frac{d\chi_i(t)}{dt} = f_i(t, \chi_1(t), \ldots, \chi_N(t)), \quad i = 1, \ldots, N, \tag{1}$$

where the functions f_i depend on the reactants of the chemical reaction, and N is the number of different species.

When the functions f_i depend not only on the current concentrations of the system, but also on concentrations in previous times, the time-delay differential equation for $\chi_i(t)$ can be written as

$$\frac{d\chi_i(t)}{dt} = f_i(t, \chi_1(t), \quad \chi_1(t-\tau_{d,1}), \ldots, \chi_N(t), \tag{2}$$

$$\chi_N(t-\tau_{d,N})), \tau_{d,i} > 0, \quad i = 1, \ldots, N, \tag{3}$$

where $\tau_{d,i}$, for $i = 1, \ldots, N$ are the delay times that, without loss of generality, are assumed to be constant throughout this paper.

3. Chemical kinetics and stochastic simulation algorithms

3.1. SSA and accelerated, approximate algorithms

We consider a system of N species that react through M reaction channels. The number of molecules of species i at time t is a random variable $\mathbf{X}(t) = (X_1(t), \ldots, X_i(t), \ldots, X_N(t))$, as random molecular collisions give rise to chemical transformations described by the reaction channels $\{R_1, \ldots, R_M\}$. A propensity function $a_j(\mathbf{X})$ and state-change vector $v_j = (v_{1j}, \ldots, v_{Nj})$ specify the dynamics of a reaction channel R_j for $j = 1, \ldots, M$. The quantity $a_j(\mathbf{X})\tau$ represents the probability that a reaction of type R_j occurs in the infinitesimal time interval $[t, t+\tau)$ and v_{ij} denotes the change induced on the number of molecule i. The following vector is defined

$$\Theta := (a_1, a_2, \ldots, a_M, \epsilon), \tag{4}$$

where ϵ is an error control parameter such that $0 \leqslant \epsilon \ll 1$ [7]. All SSAs sample a time-step and the number of reactions that occurred within that time-step. Exact and accelerated SSAs can be expressed as:

$$\tau \sim \zeta(\Theta), \tag{5}$$

$$k_j \sim \Psi(\Theta, \tau), \tag{6}$$

$$\mathbf{X}(t+\tau) = \mathbf{X}(t) + \sum_{j=1}^{M} k_j \mathbf{v}_j, \tag{7}$$

where τ is the time-step, k_j are the number of reactions of type j, and the distributions ζ and Ψ vary depending on the algorithm used. In $\tau - Leaping$ [13] the time-step is prescribed and the total number of reactions is a random variable; ζ is a Delta distribution and Ψ is a Poisson distribution. Conversely in the case of *R-Leaping* [2], the total number of reactions per time-step is prescribed and the time-step is sampled from a probability distribution. More specifically, ζ is a Gamma distribution and Ψ is a Multinomial distribution. The algorithms proceed by iterating through Eqs. (5)–(7) until the predefined final time is reached.

3.2. Accelerated, approximate SSA for delayed reactions

In *consuming* delayed reactions the reactants are instantaneously annihilated, but the products do not manifest, in reactive form, until a constant delay time τ_d after the initial reaction. In turn for *non-consuming* reactions the reactants are not annihilated until the products appear [3].

Leier et al. [16] were the first to propose an accelerated scheme for SSA of delayed reactions. Their method extends the generic formulation presented in Section 3.1 by introducing a check, at every iteration, for delays. We let R_d denote a delayed reaction, and k_d the corresponding number of reactions executed at a given time-step. The algorithm proceeds by maintaining a queue structure of the delayed reactions (Fig. 1(A)). The queue is checked at every iteration to determine whether the products of the delayed reactions ought be manifested. If this is indeed the case, the products are created and the reaction is

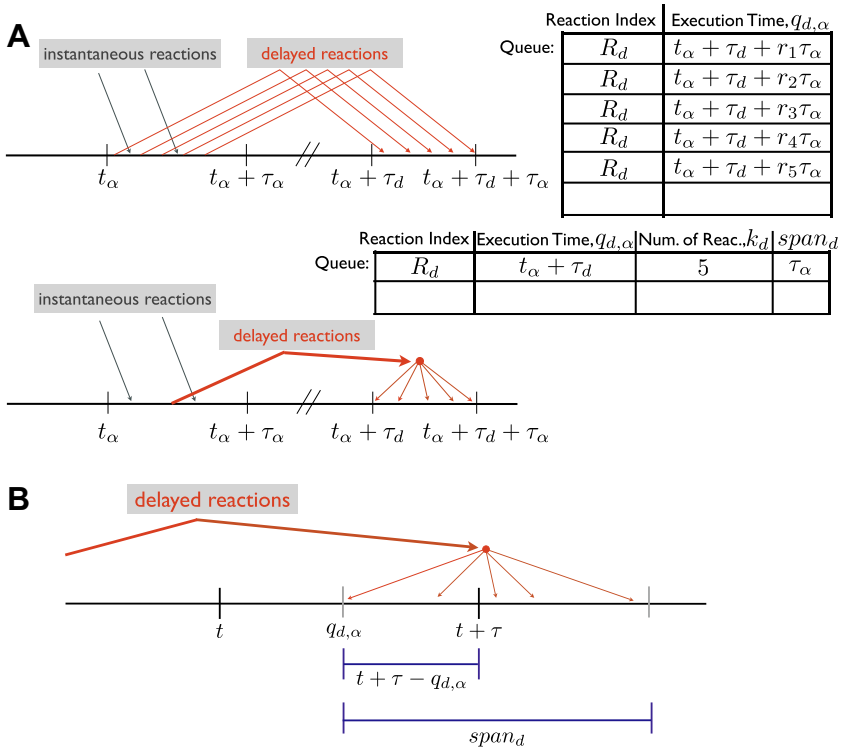

Fig. 1. (A) Comparison of approximate algorithms for the treatment of delayed reactions, time on the x-axis and their corresponding queues on the right. DAA-SSA in [16], top: the delayed reactions are uniformly distributed in the interval $[t_\alpha + \tau_d, t_\alpha + \tau_d + \tau_\alpha), r_i \sim \mathscr{U}(0,1)$, for $i = 1,\ldots,k_d$. D-leaping, bottom: the delayed reaction is stored with the number of executions, $k_d = 5$ (in this example), as well as the earliest possible execution time for the delayed reaction, $q_{d,\alpha} = t_\alpha + \tau_d$, and the time-step which generated the reaction, $span_d = \tau_\alpha$. (B) Perspective when the delayed reaction ought to be executed in the current time-step τ. If $((t + \tau - q_{d,\alpha})/span_d) < 1$, then a partial number of the k_d delayed reactions are executed, specifically $\hat{k}_d \sim \mathscr{B}(k_d, (t + \tau - q_{d,\alpha})/span_d)$.

removed from the queue. Reactions are added to the queue as a result of Eq. (6), where k_d reactions are placed on the queue, which have uniformly distributed execution times in the interval $[t_\alpha + \tau_d, t_\alpha + \tau_d + \tau_\alpha)$, where t_α is the current time and τ_α denotes the time-step (Fig. 1(A)).

4. D-leaping

The error in an accelerated stochastic simulation algorithm is $C_1\epsilon + C_2 n^{-1/2}$, where C_1, C_2 are constants, and n is the number of samples. The terms represent the time integration and statistical errors, respectively. The algorithm presented in [16] does not treat delayed reactions in an accelerated fashion since they are queued as individual events. The reason for doing so is that we have no *a priori* knowledge of the time-step in the future, i.e. whether the time-step when the delayed reaction is to be executed is on the same order of magnitude as the current time-step. If the time-step, at a later point, is substantially smaller than that used when the delayed reaction was queued, the simulation of a smaller time-step, which possibly executes single delayed reactions, would be necessary to preserve $\mathcal{O}(\epsilon)$ for the time integration error.

Therefore, single-reaction resolution would be necessary if the current time-step is smaller than that used to queue the delayed reaction. Here we propose a time-adaptive algorithm (D-leaping) which achieves singe-reaction resolution when necessary. The algorithm does not produce an error additional to the errors made by an accelerated stochastic simulation algorithm. Its computational savings are obtained by not sampling a uniform distribution for every delayed reaction. Therefore large computational savings can be procured when a substantial number of delayed reactions are queued over a time-step, i.e. when the number of molecules is large or when ϵ, the error parameter in an accelerated stochastic simulation algorithm, is large.

The time and time-step (t_α, τ_α) at which a delayed reaction would be queued is denoted here with a subscript α, whereas no subscript indicates the time and time-step (t, τ) at which a delayed reaction ought to be executed. Rather than queueing k_d delayed reactions, D-leaping queues a reaction that ought to be executed k_d times, as well as the first possible execution time for the delayed reaction, $q_{d,\alpha} = t_\alpha + \tau_d$, and the time-step, here called the span, which generated the reaction, $span_d = \tau_\alpha$ (Fig. 1(A)). In other words, $q_{d,\alpha}$ is the earliest possible execution time and $q_{d,\alpha} + span_d$ is the last possible execution time of the k_d delayed reactions.

Once the earliest possible execution time of a set of k_d delayed reactions is in the interval of the current time-step of the simulation, namely $q_{d,\alpha} \in [t, t + \tau)$, some of the k_d reactions may be executed. The essence of the D-leaping algorithm is that

only a *partial* number of the k_d reactions will be executed if the current timestep does not encapsulate the execution times of all k_d reactions, i.e. the span of the delayed reactions. If however the current time-step does indeed encapsulate the execution times, all of k_d reactions will be executed at once. The partial number of executions for the former case can be determined by considering a partitioning of the time domain.

The k_d reactions in $span_d$ follow the distribution:

$$k_d \sim \mathscr{P}(a_{d,\alpha} span_d), \tag{8}$$

where $a_{d,\alpha}$ was the propensity of the delayed reaction at time t_α and $\mathscr{P}(\lambda)$ is the Poisson distribution with parameter λ. If the span is partitioned into two arbitrary subintervals denoted by γ_1 and γ_2 such that $\gamma_1 + \gamma_2 = span_d$, it follows that the number of reactions in each subinterval, denoted by $\hat{k}_{\gamma_i}$, $i = 1, 2$ is

$$\hat{k}_{\gamma_i} \sim \mathscr{P}(a_{d,\alpha}\gamma_i) \text{ for } i = 1,2 \quad \text{such that } \hat{k}_{\gamma_1} + \hat{k}_{\gamma_2} = k_d. \tag{9}$$

It can be shown that the number of reactions in the first subinterval, namely $\hat{k}_{\gamma_1}$, is

$$\hat{k}_{\gamma_1} \sim \mathscr{B}\left(k_d, \frac{\gamma_1}{span_d}\right), \tag{10}$$

where $\mathscr{B}(N,p)$ represents a Binomial distribution of N trials with probability p. The conditional distribution of $\hat{k}_{\gamma_2}$ given $\hat{k}_{\gamma_1}$ reduces to

$$\hat{k}_{\gamma_2} \sim \mathscr{B}\left(k_d - \hat{k}_{\gamma_1}, \frac{\gamma_2}{span_d - \gamma_1}\right) \sim \mathscr{B}\left(k_d - \hat{k}_{\gamma_1}, \frac{\gamma_2}{\gamma_2}\right) \Rightarrow \hat{k}_{\gamma_2} = k_d - \hat{k}_{\gamma_1}. \tag{11}$$

The subinterval γ_1 contains the delayed reactions which will be executed in the current time-step and is defined as $\min(t + \tau - q_{d,\alpha}, span_d)$ (Fig. 1(B)). γ_2 represents the remaining time in which the delayed reactions can be executed. The queued event representing γ_2 has the updated parameters $span_d = span_d - (t + \tau - q_{d,\alpha})$, $q_{d,\alpha} = t + \tau$, and $k_d = k_d - \hat{k}_{\gamma_1}$. We note that the next time-step can subject γ_2 to further partitioning.

Specifically, two cases need to be considered in the algorithm:

Case 1, encapsulated. The time-step covers all possible execution times for all of the k_d reactions, namely $((t + \tau - q_{d,\alpha})/span_d) \geqslant 1$. The reactions do not need to be partitioned and can simply be executed all at once.

Case 2, not encapsulated. If $((t + \tau - q_{d,\alpha})/span_d) < 1$, then the portion of k_d reactions that fall in the interval $[q_{d,\alpha}, t + \tau)$ will be executed (Fig. 1(B)). Any unexecuted reactions among the potential k_d reactions will remain on the queue.

The reactions are separated into disjoint sets of delayed and non-delayed reactions, which are denoted by the indices d and j, respectively. The method can be expressed by the following pseudocode:

Algorithm 1.

while $t < t_{final}$ **do**
  $\tau \sim \zeta(\Theta)$
  $\mathbf{X}(t + \tau) = \mathbf{X}(t)$
  for all d such that $q_{d,\alpha} \in [t, t + \tau)$ **do**
    $\hat{k}_d \sim \mathscr{B}\left(k_d, \frac{\min(t+\tau-q_{d,\alpha}, span_d)}{span_d}\right)$
    $span_d = span_d - (t + \tau - q_{d,\alpha})$
    $k_d = k_d - \hat{k}_d$
    $q_{d,\alpha} = t + \tau$
    if $R_d == consuming$ **then**
      $\mathbf{X}(t + \tau) = \mathbf{X}(t + \tau) + \sum_d \hat{k}_d \boldsymbol{\nu}_d^{products}$
    else
      $\mathbf{X}(t + \tau) = \mathbf{X}(t + \tau) + \sum_d \hat{k}_d \boldsymbol{\nu}_d$
    end if
    if $k_d == 0$ **then**
      $Queue.remove([R_d, q_{d,\alpha}, k_d, span_d])$
    end if
  end for
  $k_{j \bigcup d} \sim \Psi(\Theta, \tau)$
  for all d such that $k_d \neq 0$ **do**
    $Queue.insert([R_d, q_{d,\alpha} = t + \tau_d, k_d, span_d = \tau])$
  end for
  $\mathbf{X}(t + \tau) = \mathbf{X}(t + \tau) + \sum_j k_j \boldsymbol{\nu}_j$
  for all d such that $R_d == consuming$ **do**
    $\mathbf{X}(t + \tau) = \mathbf{X}(t + \tau) + \sum_d k_d \boldsymbol{\nu}_d^{reactants}$
  end for
  $t = t + \tau$
end while

where $\nu_d^{reactants}$ and $\nu_d^{products}$ denote the stoichiometric changes for the reactants and products of delayed reaction d, respectively.

5. Numerical results

5.1. Hes1 biochemical model

Monk [20] has shown that the oscillatory expression of mRNA and protein sequences can be the result of a feedback inhibition model involving delays. The delays are believed to be a result of transcription, transcript splicing, transcript processing, and protein synthesis. The delay differential equations for the so-called Hes1 model are:

$$\frac{dm(t)}{dt} = \alpha_m \Lambda(p(t-\tau_d)) - \mu_m m(t), \tag{12}$$

$$\frac{dp(t)}{dt} = \alpha_p H(t-\tau_d) m(t) - \mu_p p(t), \tag{13}$$

$$\Lambda(p(t)) = \frac{1}{1+\left(\frac{p(t)}{p_0}\right)^h}, \tag{14}$$

where $m(t)$ and $p(t)$ are the mRNA and protein concentrations, respectively; μ_m and μ_p are the degradation rates of the mRNA and protein, respectively; α_m is the rate of transcription initiation in the absence of the protein; α_p is the rate at which the Hes1 protein is produced $H(.)$ is the Heaviside function, which here has been added to the model of Monk in order to initially inhibit the delays; $\Lambda(.)$ is the so-called *Hill function* with a *Hill coefficient* h; p_0 is the initial value of the protein [20].

The discrete form of the delay differential equations are the following elementary reactions [3]:

$$\emptyset \xrightarrow{\beta\alpha_m \Lambda(P(t-\tau_d))} M, \tag{15}$$

$$M \xrightarrow{\mu_m} \emptyset, \tag{16}$$

$$\emptyset \xrightarrow{\alpha_p M(t)} P, \tag{17}$$

$$P \xrightarrow{\mu_p} \emptyset, \tag{18}$$

where M and P denote the mRNA and protein molecules, respectively, and β is a scaling parameter. The delayed reaction is represented by Eq. (15). The initial condition is defined as, $\mathbf{X}(0) = (M(0), P(0)) = (3\beta, 100\beta)$. The following parameters were used in the simulations:

$$(\alpha_m, \alpha_p, \mu_m, \mu_p, h, \tau_d) = \left(1\ \text{min}^{-1}, 1\ \text{min}^{-1}, 0.029\ \text{min}^{-1}, 0.031\ \text{min}^{-1}, 4.1, 19.7\ \text{min}\right), \tag{19}$$

and the simulation time was $t \in [0, 12\ \text{h})$.

5.1.1. Performance

Fig. 2 shows the speed-up in running times compared with the treatment of the delayed reactions proposed in [16]. Simulations were performed by leaving β invariant and varying ϵ (Fig. 2). The results indicate a substantial speed-up because the delayed reactions need not always be distributed, and in the case that they are, efficient use of the binomial distribution is employed instead of a uniform distribution. Specifically, large computational savings can be seen if the number of molecules (β) is large, since, in effect, the number of delayed reactions that would be executed is also large.

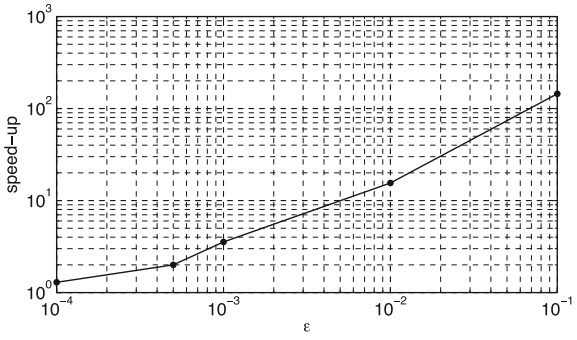

Fig. 2. Relative speed-up compared with the treatment of the delayed reactions in [16] for the Hes1 model. Logscale plot, where ϵ is varied from 10^{-4} to 10^{-1} while $\beta = 10^3$.

5.1.2. Validation

The accuracy of an approximate stochastic simulation algorithm can be evaluated by checking its convergence to a differential equation as the number of molecules becomes larger, and by its statistical correctness with respect to the exact SSA in the case of a small number of molecules.

In order to determine if the proposed algorithm converges to the delay differential equation, the parameter β was increased by factors of 10. The resulting simulations were compared to a numerical integration of Eqs. (12) and (13) using the DDE23 routine in Matlab 7.5.0 (R2007b). Fig. 3 shows the convergence of the protein concentration for three different simulations, where $\beta = 10^2, 10^3, 10^4$, and with an invariant $\epsilon = 0.005$ that bound the relative change for each species [7] using the *R-Leaping* [2] and $\tau -$ *Leaping* [13] methods. As shown, the process converged to the differential equation as the number of molecules was increased [15].

Simulations were also performed by prescribing $\beta = 10$ and comparing the D-leaping algorithm to the exact, delayed SSA [6]. Fig. 4 shows the mean of 10^3 runs for both methods as well as the standard deviation. The similarity reveals that the accelerated, adaptive method preserves the correct statistical dynamics of the system.

Furthermore, τ was plotted against the iteration number for the *R-Leaping* and $\tau -$ *Leaping* methods. Fig. 5 shows that the time-steps, and in effect the spans of the delayed reactions vary during the simulation. It should be noted that, since the $\tau -$ *Leaping* method does not sample the time-step, it does not produce a distribution in the *y*-axis as opposed to *R-Leaping*.

5.2. Delta notch 2-cell pathway

The Notch singalling pathway has been proposed as a mechanism by which oscillating gene expression of somatic cells occurs during embryonic development (for example in zebrafish [17]). The *her1* and *her7* genes encode for gene regulatory proteins, which are, in turn, regulated by another protein called the *delta Notch* protein. A *2-cell* model proposed by Lewis in [17] was simulated using D-leaping. The system consists of 3 mRNA and 3 protein species, denoted by the index *l*, and 2 adjacent cells, denoted by *l*. The indices $i = 1, 2, 3$ correspond to her1, her7, and delta, respectively. In total, there are 12 species subject to 24 reactions.

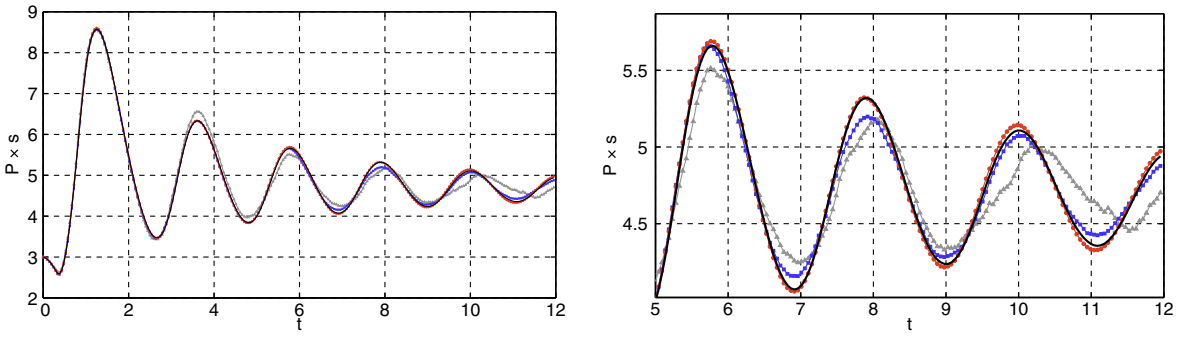

Fig. 3. Convergence with respect to the number of protein molecules for the Hes1 model [20] (left, entire simulation; right, region where $t \in [5, 12]$). Shown on the *y*-axis is the scaled number of protein molecules ($P \times s$), where $s = 0.03/\beta$, and on the *x*-axis time in hours (t); β is related to the number of molecules and was varied by factors of 10 and ϵ was fixed to 5×10^{-4}; $\beta = 10^2$, '-▲-'; $\beta = 10^3$, '—■—'; $\beta = 10^4$, '—●—'; delay differential equation (DDE), ' – '.

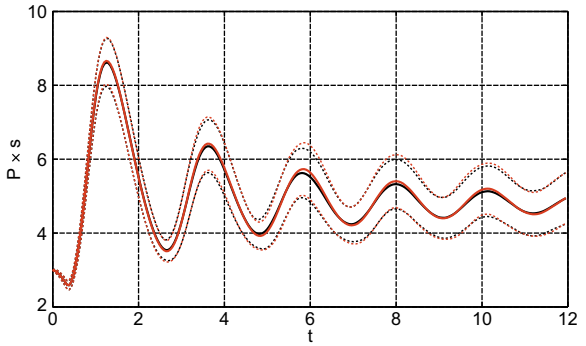

Fig. 4. Comparison of D-leaping with the exact delayed SSA [6]. Shown on the *y*-axis is the scaled number of protein molecules ($P \times s$), where $s = 0.03/\beta$, and on the *x*-axis time in hours (t); The mean over 10^3 runs (D-leaping, '—●—', delay SSA, '–') and ($\pm$) standard deviation (D-leaping, '– –', delay SSA, '- -') are shown for both methods, where $\beta = 10$. $\epsilon = 0.01$ was used for the D-leaping method.

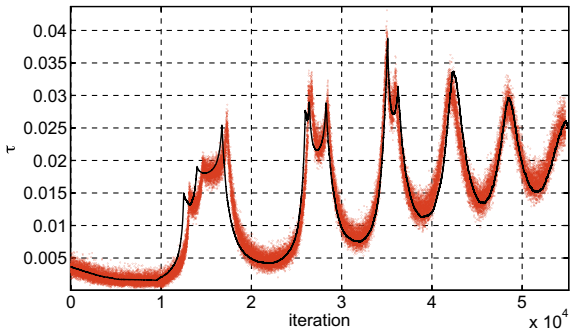

Fig. 5. τ versus the iteration number for a single run of the Hes1 model for the *R-Leaping*, '○', [2] and τ – *Leaping*, ' –', [13] methods. The parameters ϵ and β were fixed to 5×10^{-4} and 10^3, respectively. τ in the *R-Leaping* algorithm is a random variable, thereby producing a distribution in the *y*-axis, whereas τ is prescribed in τ – *Leaping*.

The discrete model for $i = 1, 2, 3$ and $l = 1, 2$ is as follows:

$$\emptyset \xrightarrow{k\rho(\mathbf{P^d}(t),i,l)} M_{i,l}, \tag{20}$$

$$M_{i,l} \xrightarrow{c} \emptyset, \tag{21}$$

$$\emptyset \xrightarrow{aM_{i,l}\left(t-\tau^d_{i,l}\right)} P_{i,l}, \tag{22}$$

$$P_{i,l} \xrightarrow{b} \emptyset, \tag{23}$$

where $\tau^d_{i,l}$ denote delay times, and the mRNA species are denoted by

$$\mathbf{M}(t) = (M_{i,l}(t))$$

and the protein species are

$$\mathbf{P}(t) = (P_{i,l}(t))$$

and the species evaluated at the originating times are

$$\mathbf{P^d}(t) = \left(P_{i,l}(t-\tau^d_{i,l})\right).$$

The function ρ is defined as

$$\rho(\mathbf{P^d}(t), i, l) = \gamma_{i,1} + \gamma_{i,2}\frac{\phi_{3,\hat{l}}}{1+\phi_{3,\hat{l}}} + \gamma_{i,3}\frac{1}{1+\phi_{1,l}\phi_{2,l}} + \gamma_{i,4}\frac{\phi_{3,\hat{l}}}{1+\phi_{3,\hat{l}}}\frac{1}{1+\phi_{1,l}\phi_{2,l}}, \tag{24}$$

where $\phi_{i,l} = P^d_{i,l}/P_{i,critical}$, where $P_{i,critical}$ = constant, l is the current cell, and $\hat{l}$ is the adjacent cell.

Initially $\mathbf{M}(0) = \mathbf{P}(0) = \mathbf{0}, k = 33(2\beta), a = 4.5, b = c = 0.23, P_{i,critical} = 2\beta \cdot (40, 40, 1000)$, where $\beta = 5$ is a scaling parameter and

$$\gamma = (\gamma_{i,j}) = \begin{pmatrix} 0.01 & 0 & 0 & 0.99 \\ 0.01 & 0 & 0 & 0.99 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

and

$$\tau^d = (\tau^d_{i,l}) = \begin{pmatrix} 11.4 & 12.6 \\ 6.745 & 7.455 \\ 16 & 20.5 \end{pmatrix}.$$

Additionally, another set of delay times are defined:

$$\tilde{\tau}^{\mathbf{d}} = (\tilde{\tau}^d_{i,l}) = \begin{pmatrix} 11.4 & 12.6 \\ 6.745 & 7.455 \\ 16 & 40.5 \end{pmatrix}.$$

Fig. 6 shows a simulation using D-leaping for $t \in [0, 2000)$. Depending on the delay times, the system can produce synchronous or asynchronous oscillations, as shown in [17]. When using the delay times defined by $\tau^{\mathbf{d}}$, the system undergoes

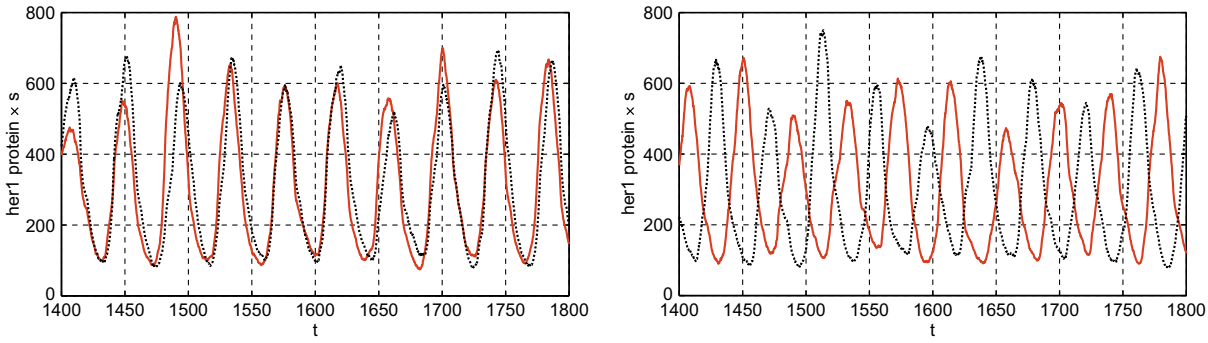

Fig. 6. Synchronous and asynchronous oscillations of the her1 proteins. Shown on the y-axis is the scaled number of protein molecules ($P_{her1} \times s$), where $s = 1/\beta$, and time (t) on the x-axis; '—●—' and '- -' denote the number of molecules in each of the two cells. (left) Using the delays times specified in $\boldsymbol{\tau}^{\mathbf{d}}$ results in synchronous oscillations, whereas (right) $\tilde{\boldsymbol{\tau}}^{\mathbf{d}}$ results in asynchronous oscillations.

synchronous osciallations (the her1 protein for both cells is show in Fig. 6). Asynchronous oscillations are the result of using $\tilde{\boldsymbol{\tau}}^{\mathbf{d}}$, where the delta Notch species has a longer delay time. The results indicate that D-leaping reproduces the correct dynamics of the system.

6. Conclusions

We presented D-leaping, a time-adaptive method for the accelerated simulation of stochastic processes with delays. The method was shown to be as fast as existing works for certain benchmark problems and in certain cases orders of magnitude more efficient. Furthermore, D-leaping does not introduce any additional errors in the accelerated stochastic simulation when incorporating delayed reactions.

The accuracy of the method was verified by numerically showing convergence to the delay differential equation as the number of molecules in the system is increased and by comparing the fluctuations in the system to those produced by the exact, delayed SSA. The algorithm was also shown to be capable of simulating complex systems of delayed reactions, i.e. the delta Notch signalling pathway. Additionally, the method is general in that it can be combined with different accelerated stochastic simulation algorithms.

The algorithm presented herein does not incorporate a specific change to the selection of the time-step for the leaping methods. A simple modification might be to create an additional bound on the changes which the incoming delayed reactions (Fig. 1(B)) would engender. For example, the bounds that Cao et al. [7] have proposed may provide a promising avenue for this particular endeavour. Ongoing work aims to extend the present algorithm to spatially developing systems [21] with multiresolution capabilities [4] with delays and in extending the recently proposed exact R-leaping algorithm [18] to systems with delays.

References

[1] D.F. Anderson, A modified next reaction method for simulating chemical systems with time dependent propensities and delays, J. Chem. Phys. 127 (21) (2007) 214107.
[2] A. Auger, P. Chatelain, P. Koumoutsakos, R-leaping: accelerating the stochastic simulation algorithm by reaction leaps, J. Chem. Phys. 125 (8) (2006) 084103.
[3] M. Barrio, K. Burrage, A. Leier, T.H. Tian, Oscillatory regulation of hes1: discrete stochastic delay modelling and simulation, PLOS Comp. Bio. 2 (9) (2006) 1017–1030.
[4] B. Bayati, P. Chatelain, P. Koumoutsakos, Multiresolution stochastic simulations of reaction–diffusion processes, Phys. Chem. Chem. Phys. 10 (39) (2008) 5963–5966.
[5] A.B. Bortz, M.H. Kalos, J.L. Lebowitz, A new algorithm for monte carlo simulation of using spin systems, J. Comput. Phys. 17 (1) (1975) 10–18.
[6] X. Cai, Exact stochastic simulation of coupled chemical reactions with delays, J. Chem. Phys. 126 (12) (2007) 124108.
[7] Y. Cao, D.T. Gillespie, L.R. Petzold, Efficient step size selection for the tau-leaping simulation method, J. Chem. Phys. 124 (4) (2006) 044109.
[8] Y. Cao, D.T. Gillespie, L.R. Petzold, Adaptive explicit–implicit tau-leaping method with automatic tau selection, J. Chem. Phys. 126 (22) (2007) 224101.
[9] A. Chatterjee, D. Vlachos, An overview of spatial microscopic and accelerated kinetic monte carlo methods, J. Comput.-Aid. Mater. Des. (2007) 14253–14308.
[10] D.T. Gillespie, A general method for numerically simulating the stochastic time evolution of coupled chemical reactions, J. Comput. Phys. 22 (4) (1976) 403–434.
[11] D.T. Gillespie, A rigorous derivation of the chemical master equation, Phys. A: Stat. Theor. Phys. 188 (1-3) (1992) 404–425.
[12] D.T. Gillespie, Exact stochastic simulation of coupled chemical reactions, J. Phys. Chem. 81 (25) (1977) 2340–2361.
[13] D.T. Gillespie, Approximate accelerated stochastic simulation of chemically reacting systems, J. Chem. Phys. 115 (4) (2001) 1716–1733.
[14] D.T. Gillespie, Stochastic simulation of chemical kinetics, Annu. Rev. Phys. Chem. 58 (2007) 35–55.
[15] N.G. Van Kampen, Stochastic Processes in Physics and Chemistry, second ed., North Holland, 2001.
[16] A. Leier, T.T. Marquez-Lago, K. Burrage, Generalized binomial tau-leap method for biochemical kinetics incorporating both delay and intrinsic noise, J. Chem. Phys. 128 (20) (2008) 205107.
[17] J. Lewis, Autoinhibition with transcriptional delay: a simple mechanism for the zebrafish somitogenesis oscillator, Curr. Biol. 13 (16) (2003) 1398–1408.

[18] Eric Mjolsness, David Orendorff, Philippe Chatelain, Petros Koumoutsakos, An exact accelerated stochastic simulation algorithm, J. Chem. Phys. 130 (14) (2009) 144110.
[19] H. Momiji, N.A.M. Monk, Dissecting the dynamics of the hes1 genetic oscillator, J. Theor. Biol. 254 (4) (2008) 784–798.
[20] N.A.M. Monk, Oscillatory expression of hes1, p53, and nf-kappa b driven by transcriptional time delays, Curr. Biol. 13 (16) (2003) 1409–1413.
[21] D. Rossinelli, B. Bayati, P. Koumoutsakos, Accelerated stochastic and hybrid methods for spatial simulations of reaction–diffusion systems, Chem. Phys. Lett. 451 (1–3) (2008) 136–140.
[22] T.H. Tian, K. Burrage, Binomial leap methods for simulating stochastic chemical kinetics, J. Chem. Phys. 121 (21) (2004) 10356–10364.