文章编号: 1000-0887(2007) 10-1213-10

应用数学和力学编委会, ISSN 1000-0887

随机模拟化学反应系统的加速 L-leap 算法

彭新俊, 王翼飞

(上海大学 数学系, 上海 200444)

(郭兴明推荐)

摘要: 提出了随机模拟化学反应系统的加速 L-leap 算法, 该算法根据 leap 条件确定具有最大倾向函数的反应通道的反应次数, 并利用二项分布随机数生成其它反应通道在当前 leap 时间区间内的反应次数 L-leap 算法可更好地满足 leap 条件 数值模拟实验表明该算法能取得更好的模拟性能

关 键 词: L-leap 算法; leap 条件; 随机模拟算法; 化学反应系统

中图分类号: O644 文献标识码: A

引 言

由于某些关键反应物的分子数目非常少, 生物细胞中的随机性和离散性显得非常重要, 因而化学反应系统的随机模拟已引起许多研究工作者的关注[1-2] Gillepsie 于 30 年前提出了精确但低效模拟化学反应系统的随机模拟算法(stochastic simulation algorithm, SSA)[3-4] 近年来, 一些研究工作者提出了一些等价算法[5-7], 但这些算法没有显著提高模拟速度

为提高随机模拟的速度, Gillepsie 提出了一种称为 τ-leap 近似模拟算法[8], 该算法根据 leap 条件(在 leap 时间区间内所有的倾向函数保持近似常量)确定 leap 时间区间, 并利用泊松分布随机数确定每个反应通道在当前时间区间内的反应次数 Chatterjee 等人进一步提出了二项 leap 模拟算法[9], 该算法用二项分布随机数代替泊松分布随机数 然而, 这些方法均有违反 leap 条件的概率 最近, Auger 等人提出了一种称为 R-leap 的近似算法[10], 该算法在每一步模拟中确定反应系统的总反应次数, 并利用伽玛分布随机数确定 leap 时间 Cai 与 Xu 也提出了一个类似的 K-leap 算法[11] 与早期的算法类似, 这两个算法也存在一定的违反 leap 条件的概率 例如, 当利用二项分布随机数确定反应通道的反应次数时, 可能出现某个通道的反应次数为非常大的情形

本文提出了一种模拟化学反应的 L-leap 算法 该近似算法首先根据 leap 条件确定当前时刻具有最大倾向函数的反应通道的反应次数 L, 而其它反应通道的反应次数则根据和倾向

收稿日期: 2007-04-18; 修订日期: 2007-06-27

基金项目: 国家自然科学基金资助项目(30571059); 国家高科技研究发展计划(863)专项资助项目(2006AA02Z190)

作者简介: 彭新俊(1980), 男, 湖南人, 博士生(E-mail: xjpeng@shu.edu.cn);

王翼飞(1948), 男, 教授, 硕士(联系人. E-mail: yifei_wang@staff.shu.edu.cn).

函数利用二项分布随机数确定, 并采用伽玛分布随机数确定 *leap* 时间 由于每个反应通道的反应次数均不超过 L, L–leap 算法可比其它算法更好地满足 leap 条件 文中给出了几种确定最大反应次数的有效方法 数值模拟实验结果表明 L–leap 算法可以取得比其它算法更好的模拟性能

1 背 景 知 识

对一个存在 M 个反应通道$\{R_1, \quad, R_M\}$及N 种化学物质$\{S_1, \quad, S_N\}$并达到热力学平衡的化学反应系统, 其 t 时刻的动力学状态可由状态向量$\boldsymbol{X}(t) = (X_1(t), \quad, X_N(t))^{\mathrm{T}}$表示, 其中 $X_i(t)$, $i = 1, \quad, N$, 表示物质 S_i 在t 时刻的分子数目 若 $a_j(\boldsymbol{x})$ 与 $\boldsymbol{v}_j = (v_{1j}, \quad, v_{Nj})^{\mathrm{T}}$ 分别表示 t 时刻反应通道 R_j 的倾向函数与状态改变向量, 其中 v_{ij} 表示 R_j 的一个事件发生时, 物质 S_i 的分子改变数量, 则对给定的 $\boldsymbol{X}(t) = \boldsymbol{x}$, 反应通道 R_j 在下一个无穷小时间区间$[t, t + \ )$ 内发生一次反应的概率为 $a_j(\boldsymbol{x})\mathrm{d}t$

Gillepsie 的 SSA[2-3] 在每一步模拟事件: 在时间区间 $[t, t + \ )$ 内不发生任何反应, 且 R_j 将在无穷小时间区间$[t + \ , t + \ + \mathrm{d} \ )$ 内发生一次反应 为此, SSA 在每一步产生两个标准均匀的随机数 r_1 与 r_2 不妨假设 $a_0(\boldsymbol{x}) = \sum_{m=1}^{M} a_m(\boldsymbol{x})$, 则下一个反应 R_j 将在$t + \ $ 时刻发生, 其中 由下式确定:

$$= -\ \ln(r_1)/(a_0(\boldsymbol{x})), \tag{1}$$

而 j 为满足下述不等式的最小正整数

$$\sum_{j=1}^{j} {}_j(\boldsymbol{x}) > r_2 a_0(\boldsymbol{x}) \tag{2}$$

从而系统更新其系统状态 $\boldsymbol{X}(t + \ ) = \boldsymbol{x} + \boldsymbol{v}_j$ 这一过程重复直到终点时刻或其它条件成立

为提高随机模拟的速度, Gillespie 提出了一个称为 –leap 的近似模拟算法[8] 具体地, 该算法在每一步利用 leap 条件确定 leap 时间 , 在 确定后, 用 M 个独立的泊松分布随机数近似所有反应通道在该时间区间内发生的反应的次数 K_j:

$$K_j(\ ; \boldsymbol{x}) \quad P_j(a_j(\boldsymbol{x}), \ ), \qquad j = 1, \quad, M, \tag{3}$$

其中 $P_j(a_j(\boldsymbol{x}), \ )$ 表示均值为 $a_j(\boldsymbol{x})$ 的泊松随机变量 然后系统更新其状态

$$\boldsymbol{X}(t + \ ) = \boldsymbol{x} + \sum_{j=1}^{M} \boldsymbol{v}_j P_j(a_j(\boldsymbol{x}), \ )$$

–leap 算法的一个重要问题是如何根据 leap 条件确定时间 的值 记 $a_j(\ ; \boldsymbol{x}) = a_j(\boldsymbol{X}(t + \ )) - a_j(\boldsymbol{x})$, Gillespie 引入下面的约束以使得 leap 条件成立:

$$| \ a_j(\ , \boldsymbol{x}) | \qquad a_0(\boldsymbol{x}), \qquad j = 1, \quad, M, \tag{4}$$

其中 0< 1为误差控制参数 由于 $a_j(\ , \boldsymbol{x})$ 是一个随机变量, Gillespie 采用一阶Taylor 展开近似 $a_j(\ , \boldsymbol{x})$, 并令其均值的绝对值和标准差不超过 $a_0(\boldsymbol{x})$, 从而可以确定 的值

由于泊松随机变量可以是任意的非负整数, 这将导致在运行 –leap 算法时存在一定的风险, 如某些物种出现负状态或违反 leap 条件 Chatterjee 等人[9] 提出了一种称为二项 –leap 的模拟算法, 该方法利用二项随机变量确定每个反应通道的反应次数从而避免了负状态的出现, 但又产生了一些新的问题, 如过严格现象, 并且该算法无法推广到一般情形[13] 为此, Cao 等人[13] 最近提出了一个改进 –leap 算法 在模拟化学反应系统的每一步中, 该方法将反应通道分成 critical 和 non-critical 两类, 并分别用 SSA 和 –leap 算法进行模拟

由于对给定的 , 这些 -leap 算法无法真正保证模拟过程一定满足 leap 条件, 最近提出了 R-leap 与 K-leap 算法[10-11] 首先根据 leap 条件计算下一个 leap 时间区间内的所有反应通道的总反应次数 K, 然后利用二项或多项随机变量分配每一个反应通道的反应次数 $K_j, j = 1, \ , M$, 并利用伽玛随机数确定 leap 时间 然而, 若在分配所有反应通道的反应次数时出现某个通道的反应次数非常大, 这将会导致 leap 条件不再满足, 从而降低模拟精度

2 L-leap 算法

为避免在模拟过程中 K-leap 或 R-leap 算法有可能不满足 leap 条件这一情形, L-leap 算法首先根据 leap 条件确定系统中所有反应通道的最大反应次数 L, 其中 L 为满足 leap 条件的数 不失一般性, 我们假设 L 为具有最大倾向函数的反应通道对应的反应次数, 即 L 为第 个反应通道的反应次数, 其中 $= \arg\max_{1 \ j \ M}\{a_j(\boldsymbol{x})\}$ 事实上, 若 L 为具有较小倾向函数的反应通道的反应次数, 则会导致模拟过程以一个较大的概率违反 leap 条件 下文用 K 表示第 个反应通道的反应次数, 即 $K = L$ 对给定的 L, 根据 Gillespie 的结论[9], 对应的 leap 时间 为伽玛随机变量, 其概率密度函数为

$$p(\mid L) = \frac{a(\boldsymbol{x})\exp(- a(\boldsymbol{x}))[a(\boldsymbol{x})]^{L-1}}{(L-1)!}, \qquad 0 \tag{5}$$

另一方面, 对给定的 L, L-leap 算法需要解决的一个重要问题是如何确定每个反应通道的反应次数 K_j, j 事实上, 附录了证明 K 的条件概率密度函数 $p(K \mid)$ 为 Poisson 分布函数 $P(a(\boldsymbol{x}),)$, 进一步, 对给定的 , 条件概率密度函数 $p(K_j \mid , K) = p(K_j \mid K)$ 为二项分布函数

$$p(K_j \mid K) = \binom{K}{K_j} {}_j^{K_j}(1- {}_j)^{K-K_j}, \qquad j \qquad , \tag{6}$$

其中 $_j = a_j(\boldsymbol{x})/(a(\boldsymbol{x})), j$ 易见, 我们有 $E(K_j \mid K) = K a_j(\boldsymbol{x})/(a(\boldsymbol{x})), j$

为在实际的化学反应系统中实现用 L-leap 算法进行模拟, 下面发展了几种基于 leap 条件确定 L 的值的方法

2.1 L-selection 方法 1

在 -Leap 算法中, Gillespie 与 Petzold[8],[12] 提出了一种确定 的非常直接的方法 该方法要求对 $a_j(, \boldsymbol{x})$ 成立式(4) 对 $a_j(, \boldsymbol{x})$ 进行一阶 Taylor 展开, 其近似为

$$a_j(, \boldsymbol{x}) \quad \sum_{j=1}^{M} f_{jj}(\boldsymbol{x})K_j, \qquad j = 1, \ , M, \tag{7}$$

其中 $f_{jj}(\boldsymbol{x}) = \sum_{i=1}^{N}(a_j(\boldsymbol{x})/ x_i)v_{ij}, j, j = 1, \ , M$ 计算式(7)的条件期望值和方差, 有

$$E(a_j(, \boldsymbol{x}) \mid K) \quad [K_j(\boldsymbol{x})]/(a(\boldsymbol{x})), \qquad j = 1, \ , M, \tag{8}$$

$$\mathrm{var}(a_j(, \boldsymbol{x}) \mid K) \quad [K {}_j^2(\boldsymbol{x})]/(a^2(\boldsymbol{x})), \qquad j = 1, \ , M, \tag{9}$$

这里

$$_j(\boldsymbol{x}) = \sum_{j=1}^{M} f_{jj}(\boldsymbol{x}) a_j(\boldsymbol{x}), \qquad j = 1, \ , M, \tag{10}$$

$$_j^2(\boldsymbol{x}) = \sum_{j=1}^{M} (a(\boldsymbol{x}) - a_j(\boldsymbol{x}))f_{jj}^2(\boldsymbol{x}) a_j(\boldsymbol{x}), \qquad j = 1, \ , M, \tag{11}$$

类似于文献[12], 我们要求 $\mid E(a_j(, \boldsymbol{x}) \mid K) \mid$ $a_0(\boldsymbol{x})$ 与 $\sqrt{\mathrm{var}(a_j(, \boldsymbol{x}) \mid K)}$ $a_0(\boldsymbol{x})$

同时成立, 并保证 K 的值不小于 1, 从而可以得到满足 leap 条件的 K 的值

$$K = \max\left\{1, \min_{j\in M}\left[a(\boldsymbol{x})\frac{a_0(\boldsymbol{x})}{|\ _j(\boldsymbol{x})|}, a^2(\boldsymbol{x})\frac{^2a^2(\boldsymbol{x})}{^2_j(\boldsymbol{x})}\right]\right\} \tag{12}$$

最近, Cao 等人[14]在他们的工作中发展了一种更为精确地确定 的方法, 该方法用 $\max\{a_j(\boldsymbol{x}), c_j\}$ 代替原方法中的 $a_0(\boldsymbol{x})$ 这里同样用 $\max\{a_j(\boldsymbol{x}), c_j\}$ 代替式(12)中的 $a_0(\boldsymbol{x})$, 从而得到一种更为精确的确定 K 的方法

$$K = \max\left\{1, \min_{j\in M}\left[a(\boldsymbol{x})\frac{\max\{a_j(\boldsymbol{x}), c_j\}}{|\ _j(\boldsymbol{x})|}, a^2(\boldsymbol{x})\frac{\max\{a_j(\boldsymbol{x}), c_j\}^2}{^2_j(\boldsymbol{x})}\right]\right\} \tag{13}$$

2.2 L–selection 方法 2

由于在利用期望值与标准差计算 $a_j(\ ,\boldsymbol{x})$ 的界时必须对倾向函数微分, 这在 M 与 N 非常大时是非常耗时的, 为此, 本节发展了一种更为高效的确定 K 的方法

对每一个物种 S_i, 考虑 $X_i(t,\ ) = X_i(t+\ ) - x_i, i = 1,\ ,N$, 有

$$X_i(t,\ ) = \sum_{j=1}^{M} v_{ij}K_j, \tag{14}$$

其绝对值应当不超过 $\max\{ix_i, 1\}$, 即 $|\ X_i(t,\ )|\ \max\{ix_i, 1\}$, 其中 $_i = \ /g_i$, 且 g_i 的定义如下: 若记 HOR(i) 为 S_i 作为反应物的最高反应阶, 有( ) 若 HOR$(i) = 1$, 有 $g_i = 1$; ( ) 若 HOR$(i) = 2$, 则当存在某个二阶反应需要两个 S_i 分子时有 $g_i = 2 + 1/(x_i - 1)$, 否则 $g_i = 2$; ( ) 若 HOR$(i) = 3$, 则当存在某个三阶反应需要两个 S_i 分子时有 $g_i = 3 + 3/(2(x_i - 1))$, 当存在某个三阶反应需要 3 个 S_i 分子时有 $g_i = 3 + 1/(x_i - 1) + 2/(x_i - 2)$, 否则 $g_i = 3$

计算每一个 $X_i(t,\ )$ 的条件期望和方差有

$$E(\ X_i(t,\ )\ |\ K\ )\ \ [K\ \hat{}_i(\boldsymbol{x})]/(a(\boldsymbol{x})), \qquad i = 1,\ ,N, \tag{15}$$

$$\mathrm{var}(\ X_i(t,\ )\ |\ K\ )\ \ [K\ \hat{}^2_j(\boldsymbol{x})]/(a^2(\boldsymbol{x})), \qquad i = 1,\ ,N, \tag{16}$$

其中

$$\hat{}_i(\boldsymbol{x}) = \sum_{j=1}^{M} v_{ij}a_j(\boldsymbol{x}), \qquad i = 1,\ ,N, \tag{17}$$

$$\hat{}^2_i(\boldsymbol{x}) = \sum_{j=1}^{M} (a(\boldsymbol{x}) - a_j(\boldsymbol{x}))v^2_{ij}a_j(\boldsymbol{x}), \qquad i = 1,\ ,N \tag{18}$$

同样, 我们要求 $|\ E(\ X_i(t,\ )\ |\ K\ )|\ \ \max\{ix_i, 1\}$ 以及成立 $\sqrt{\mathrm{var}(\ X_i(t,\ )\ |\ K\ )}\ \max\{ix_i, 1\}$, 从而得到下面的计算 K 的方法

$$K = \max\left\{1, \min_{i\in N}\left[a(\boldsymbol{x})\frac{\max\{ix_i, 1\}}{|\ \hat{}_i(\boldsymbol{x})|}, a^2(\boldsymbol{x})\frac{\max\{ix_i, 1\}^2}{\hat{}^2_i(\boldsymbol{x})}\right]\right\} \tag{19}$$

易见, 由于式(19)无需计算倾向函数的导数, 该式比由式(13)计算 K 更为高效, 特别是当 M 与 N 非常大时尤为明显

2.3 L–selection 方法 3

注意到 $a_j(\ ,\boldsymbol{x})$ 可用 $\sum_{j=1}^{M} f_{jj}(\boldsymbol{x})K_j$ 近似, 并且对所有的 $K_j, j = 1,\ ,M$, 有 $K_j\ \ K$ 成立, 故有

$$|\ \ a_j(\ ,\boldsymbol{x})|\ \ K\sum_{j=1}^{M}|f_{jj}(\boldsymbol{x})|, \qquad j = 1,\ ,M, \tag{20}$$

限制上述不等式的右边不超过 $a_0(\boldsymbol{x})$ 或 $\max\{a_j(\boldsymbol{x}), c_j\}$, 易见, 这可以保证 leap 条件成立, 从而得到下面的计算 K 的方法

$$K = \max\left\{1, \min_{1\le j\le M}\left[\frac{a_0(\boldsymbol{x})}{\sum_{j=1}^{M}|f_{jjc}(\boldsymbol{x})|}\right]\right\}, \tag{21}$$

$$K_A = \max\left\{1, \min_{1\le j\le M}\left[\frac{\max\{Ea_j(\boldsymbol{x}), c_j\}}{\sum_{j_c=1}^{6}|f_{jjc}(\boldsymbol{x})|}\right]\right\}, \tag{22}$$

易见, 由于在式(21)与式(22)中仍出现了$f_{jjc}(\boldsymbol{x})$, 当M与N非常大时, 这两个确定K_A的方法仍是相当耗时的1 但它们能严格地满足 leap 条件, 事实上, 在式(21)与式(22)中, 对$a_j(S, \boldsymbol{x})$的约束比方法 1 中的更为严格1 因而为了能获得一个比较满意的模拟速度, 通常要在式(21)与式(22)中取较大的 E 值1

2.4 L-selection 方法 4

这种确定K_A的方法与上述方法 3 类似, 但该方法是基于物种的1 注意到在式(14)中有$K_j \le K_A$成立, 故

$$|\Delta X_i(t, S)| = K_A\sum_{j=1}^{M}|v_{ij}|, \qquad i = 1, \cdots, N \tag{23}$$

同样, 限制不等式(23)的右边不超过$\max\{Ex_i, 1\}$, 我们可得到如下公式

$$K_A = \max\left\{1, \min_{1\le i\le N}\left[\frac{\max\{Ex_i, 1\}}{\sum_{j=1}^{6}|v_{ij}|}\right]\right\}\tau \tag{24}$$

这种方法是最简单的计算最大反应次数的方法, 每次运行需要最少的计算时间1 而且, 与式(21)与式(22)类似, 对一个较大的 E 值, 该方法能严格地满足 leap 条件1

2.5 L-leap 算法

基于前面的分析, 本节给出 L-leap 算法的具体过程1 在该算法中, 我们引入参数 n_A 用于提高模拟精度: 在时刻 t, 当 K_A 小于 n_A 时, L-leap 算法自动过渡到 SSA1 由于 L-leap 算法可以自适应地在两者之间选择合适的方法进行模拟, 可以期望该算法能得到较好的模拟精度1 数值实验中取 $n_A = 10$1

算法 1(L-leap 算法)

1) 初始化系统状态, 并令 $t = 0$;
2) 计算倾向函数 $a_j(\boldsymbol{x}), j = 1, \cdots, M$;
3) 确定反应通道 A 以及其倾向函数$a_A(\boldsymbol{x}) = \max_{1\le j\le M}\{a_j(\boldsymbol{x})\}$;
4) 采用式(12)、式(13)、式(19)、式(21)、式(22), 或式(24)计算 K_A;
5) 若 $K_A < n_A$, 运行一次 SSA, 并转步 8);
6) 若 $K_A \ge n_A$, 采用伽玛概率密度函数(5)生成 leap 时间 τ;
7) 对任给 $j \in A$, 利用二项分布(6)确定各自的反应次数 K_j;
8) 更新时间 $t = t + \tau$ 及系统状态$\boldsymbol{X}(t + \tau) = \boldsymbol{x} + \sum_{j=1}^{M}\boldsymbol{v}_jK_j$;
9) 若 $t < t_{end}$, 转 2), 否则程序停止1

算法 1 简要给出了模拟化学反应系统 L-leap 算法的步骤1 为了避免模拟过程中某些物种出现负分子数目的情形, 特别是采用前两种方法确定 K_A, 我们从反应通道 A 开始, 逐次在产生下一个反应通道的反应次数之前, 取 K_j 与 R_j 的最大允许反应次数之间的最小值1

3 数值模拟

为验证所提出的 L-leap 算法的模拟性能, 本节模拟 3 个具体的数值实例, 分别是独立双通道模型[11], 降解-二聚合模型[8,12,14] 和 LacZ/LacY 模型[13-14]1 实验中对比了 Gillespie 的 S-leap 算法[8,12]、Cao 等人的改进 S-leap 算法[14]、二项分布 S-leap 算法[9]、K-leap 算法[11] 和 L-leap 算法等的模拟性能1 我们采用不同算法和 SSA 得到的直方图之间直方图距离对比它们的精度1 这里所有的模拟工作均是用 MATLAB 实现的, 其运行环境为 WinXP 系统, 1. 99 GHz CPU 与 256Mb 内存1

3.1 独立双通道模型

该模型由两个反应通道组成

$$S_1 \xrightarrow{c_1} \varnothing, \quad S_2 + S_3 \xrightarrow{c_2} S_41 \tag{25}$$

这两个反应通道实际上是独立的, 但为了确定每一个 leap 时间区间内的反应通道 A, 我们考虑了两个反应通道 1 实验模拟的反应率常数分别为 $c_1 = 1$, $c_2 = 1 @10^{-4}$, 初始值为 $X_1 = X_2 = 3\,000$, $X_3 = 1 @10^4$, $X_4 = 01$ 对每一种方法, 选取不同 E 的值, 分别在时间区间 $[0, 2]$ 运行 $5 @ 10^4$ 次, 并计算各自在 $X_1(2)$ 直方图与 SSA 的直方图之间的直方图距离1

图 1 给出了 $X_1(2)$ 的直方图距离与 $5 @10^4$ 次模拟的 CPU 时间的关系1 可以看到方法 3 取得最好的模拟性能, 即对给定的 CPU 时间, 方法 3 得到的直方图距离最小1 方法 1 同样可得到较好的模拟性能, 该方法比原始 S-leap 算法和二项分布 S-leap 算法的模拟性能要好 1 对比方法 1 和方法 3, 可以发现方法 3 的模拟效率更高, 这是由于利用式(22) 计算 K_A 比式(13) 简单1 另外, 由于实验所用的二项分布随机数产生器比 Poisson 分布随机数产生器效率要低, 这导致方法 1 的模拟效率比改进 S-leap 算法效率低, 也导致二项分布 S-leap 算法的模拟效率最低1

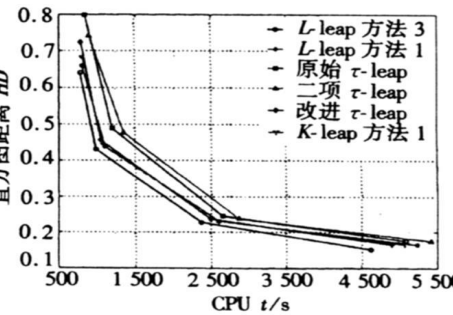

图 1 独立双通道模型(25) 的 $X_1(2)$ 直方图距离与 $5@10^4$ 次运行的 CPU 时间的关系

3.2 降解-二聚合模型

降解-二聚合模型包括 3 个反应物和如下的 4 个反应通道

$$\begin{cases} R_1: S_1 \xrightarrow{c_1} \varnothing, & R_2: S_1 + S_1 \xrightarrow{c_2} S_2, \\ R_3: S_2 \xrightarrow{c_3} S_1 + S_1, & R_4: S_2 \xrightarrow{c_4} S_31 \end{cases} \tag{26}$$

在该系统中, 单体分子 S_1 能聚合得到一个不稳定形式 S_2, 该聚合过程具有可逆性, 而 S_2 可转化为稳定状态 S_3, 同时, 分子 S_1 可以降解1

Gillespie 曾详细分析了反应率取 $c_1 = 1$, $c_2 = 0.02$, $c_3 = 0.5$, $c_4 = 0.04$ 的模拟情形 1 这里我们采用相同的反应率, 并设系统状态初值为 $X_1 = 4\,150$, $X_2 = 39\,565$, $X_3 = 3\,4451$ 对不同的 E 值, 分别在时间区间 $[0, 10]$ 内运行 $5 @10^4$ 不同的 leap 算法和 SSA, 并计算各自的直方图与 SSA 的直方图之间的直方图距离1

图 2 给出了 $X_2(10)$ 与 $X_3(10)$ 的直方图距离与 $5 @10^4$ 次模拟的 CPU 时间的关系1 同样可以看到, 对给定的 CPU 时间, 方法 4 取得最小的直方图距离, 而方法 2 的直方图距离仅仅在较

长的 CPU 时间下只比改进 S-leap 算法差 1 这些结果表明本文提出的 L-leap 算法能取得更好的模拟性能, 特别是当 CPU 时间较短时, 该方法能取得比其它已建立的近似方法更好的精度, 这对应着较大的 E值1 这主要是由于 L-leap 算法在整个模拟过程中能更好地满足 leap 条件1

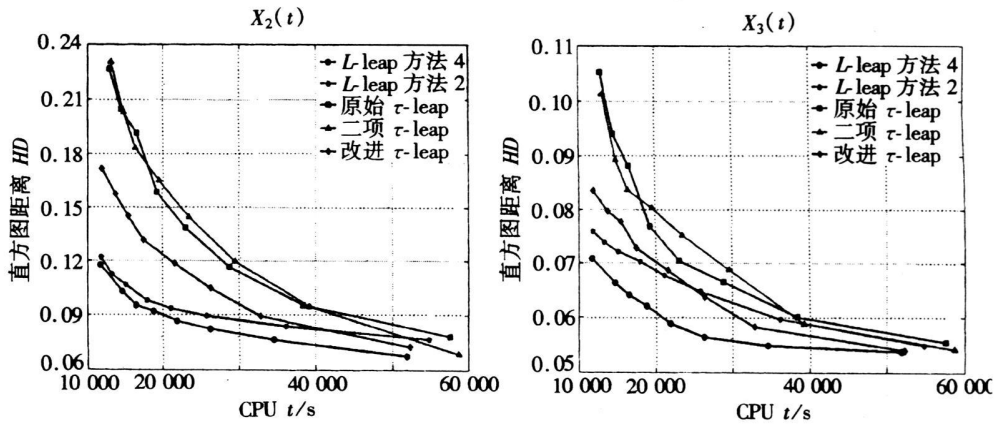

图 2 降解-二聚合模型(26)的 $X_2(10)$ 与 $X_3(10)$ 的直方图距离与 $5@10^4$ 次运行的 CPU 时间的关系

3.3 LacZ/LacY 模型

LacZ/LacY 模型是用于描述 E. coli 中的 LacZ、LacY 基因表达和 LacZ、LacY 蛋白质活性化学反应系统, 它包括 22 个反应通道和 19 种反应物1 该模型可用于验证 L-leap 算法是否能应用于中等大小的刚性反应系统1 模型的初值如下: 除 PLac= 1, RNAP~ $N(35, 3.5^2)$ 和 Ribosome~ $N(350, 35^2)$ 外, 其它物质的初始值均为 0, 且反应体积随着反应时间和倾向函数的增加而线性递增1

由于采用 SSA 进行模拟耗时过大, 我们无法得到 L-leap 算法在第一个细胞代(时间区间为[0, 2 100])内的模拟精度1 为简单计, 模拟中我们仅采用式(19)计算 K_A 的值1 表1给出了 L-leap 算法在时间区间[0, 300]内 100 次模拟相对于 SSA 的加速情形1 对比可见, 当 E= 0.4 时, L-leap 算法可提高大约 18 倍的模拟速度, 而当 E= 0.8 时, 则可提高大约 70 倍的模拟速度1 模拟实验的第 2 部分考虑了几种物种在时间区间[150, 180]内的均值和标准差的轨迹拟合情形1 为此, 首先运行一次 SSA 到 t = 150 得到一个状态初值, 然后在[150, 180]内分别运行 SSA 和 E= 0.7 时的 L-leap 算法1 图3给出了几种物质在这一时间区间内的两种算法的轨迹图, 可以看出, L-leap 算法取得了相当好的近似结果1 实验的最后一部分首先运行一次 SSA 获得 t = 150 的状态初值, 然后运行 SSA 和 L-leap 算法得到二者之间的直方图距离1 图4给出了物种 TrRbsLacZ 在 t = 151 时刻对应 $1@10^4$ 次模拟的直方图距离1 很明显, L-leap 算法在这个模型中取得了较好的模拟精度1

表 1 L-leap 算法相对于 SSA 的模拟加速情况

L-leap 方法	加速度
E= 0.4	18.7
E= 0.5	27.9
E= 0.6	40.5
E= 0.7	53.1
E= 0.8	70.5

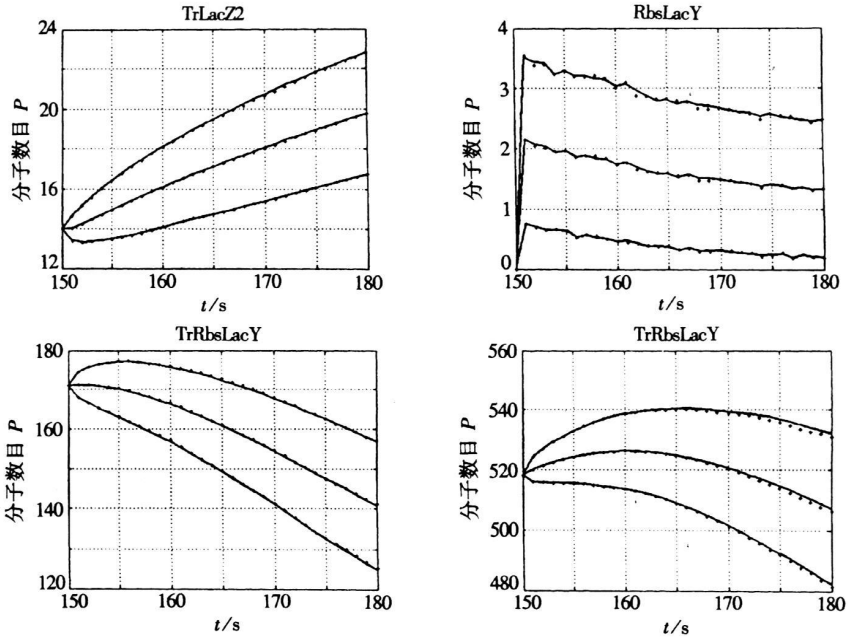

图 3 LacZ/LacY 模型中 4 个物种的均值轨迹和标准差的轨迹图

4 结 论

Gillespie 的 SSA 可以精确地模拟生化系统的反应过程, 但由于其效率低而不实用1 最近提出的一系列 S-leap 算法基于 leap 条件可有效地提高模拟速度, 但这些算法均存在一定的违反 leap 条件的概率, 从而导致模拟精度下降1

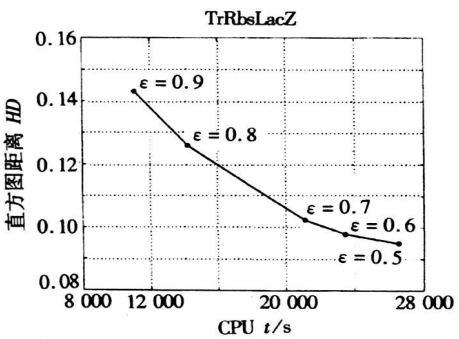

图 4 LacZ/LacY 模型中物种 TrRbsLacZ 在 t = 151 时刻的直方图距离

本文发展了一种模拟化学反应系统的称为 L-leap 算法的全新方法, 该方法首先根据 leap 条件确定当前时刻具有最大倾向函数的反应通道的反应次数 L, 而其它反应通道的反应次数则根据 L 和倾向函数利用二项分布随机数确定 1 这一确定方法可以保证更好地满足 leap 条件, 从而提高模拟精度 1 若将其它算法引入到 L-leap 算法, 可以进一步地提高模拟精度 1 我们下一步的工作将包括探讨 L-leap 算法中对具有最大倾向函数的反应通道的反应次数的约束对模拟精度的影响1

附录 I

对给定的 S, Gillespie[14] 指出任意反应通道在时间区间$(t, t+\ S)$ 内的反应次数 K_j 为独立泊松随机变量 $P_j(\ a_j(\boldsymbol{x}), S)$, 其密度函数为

$$p(K_j) = \frac{a_j(\boldsymbol{x})S)^{K_j}}{K_j!}\exp\{-\ a_j(\boldsymbol{x})S\}, \qquad j = 1, , , M1 \tag{A1}$$

类似于 Gillespie[9] 的证明, S 相对于由 L-leap 方法得到的 K_A 的条件概率密度函数为伽玛分布$(5)1$ 反之, 由中心极限定理, 对给定的 S, K_A 为泊松随机变量, 其条件概率密度函数为 $P_A(\ a_A(\boldsymbol{x})S)1$

现考虑条件概率密度函数 $p(K_j \mid K_A)$. 注意到式(A1) 中的 $p(K_j)$ 可重写如下

$$p(K_j) = \frac{a_j(\boldsymbol{x})S)^{K_j}}{K_j!}\exp\{-a_j(\boldsymbol{x})S\} =$$

$$\frac{a_j(\boldsymbol{x})S)^{K_j}}{K_j!}\exp\{-a_A(\boldsymbol{x})S\}\exp\{(a_A(\boldsymbol{x})-a_j(\boldsymbol{x}))S\} =$$

$$\frac{a_j(\boldsymbol{x})S)^{K_j}}{K_j!}\sum_{K_A\geqslant K_j}\frac{\exp\{-a_j(\boldsymbol{x})S\}}{(K_A-K_j)!}\{(a_A(\boldsymbol{x})-a_j(\boldsymbol{x}))S\}^{K_A-K_j} =$$

$$\sum_{K_A\geqslant K_j}\binom{K_A}{K_j}D_j^{K_j}(1-D_j)^{K_A-K_j}p(K_A), \tag{A2}$$

其中 $D_j = a_j(\boldsymbol{x})/(a_A(\boldsymbol{x}))$, $j = 1, \cdots, M$. 另一方面, 我们有

$$p(K_j) = \sum_{K_A} p(K_j, K_A) = \sum_{K_A\geqslant K_j} p(K_j, K_A) = \sum_{K_A\geqslant K_j} p(K_j \mid K_A)p(K_A), \tag{A3}$$

式(A3)的第 2 个等式成立的原因是由于我们假设第 A 个反应通道的反应次数最多 1. 注意到函数集 $\{(x^n/n!)\exp(-x): n \geqslant 0\}$ 为 $C(R)$ 中的基函数集, 故有

$$p(K_j \mid K_A) = \binom{K_A}{K_j}D_j^{K_j}(1-D_j)^{K_A-K_j}. \tag{A4}$$

从而由式(6) 得到的 K_j 等价于概率密度函数为式(A1) 的 Poisson 分布随机数.

[参 考 文 献]

[1] Ideker T, Galitski T, Hood L. A new approach to decoding life: systems biology[J]. Annu Tev Genom Human Genet, 2001, 2(1): 343-372.

[2] Fedoroff N, Fontana W. Genetic networks: Small numbers of big molecules[J]. Science, 2002, 297(5584): 1129-1131.

[3] Gillespie D T. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions[J]. J Comput Phys, 1976, 22(4): 403-434.

[4] Gillespie D T. Exact stochastic simulation of coupled chemical reactions[J]. J Chem Phys, 1977, 81(25): 2340-2361.

[5] Gibson M, Bruck J. Efficient exact stochastic simulation of chemical systems with many species and many channels[J]. J Chem Phys, 2000, 104(9): 1876-1889.

[6] Cao Y, Li H, Petzold L R. Efficient formulation of the stochastic simulation algorithm for chemically reacting systems[J]. J Comput Phys, 2004, 121(9): 4059-4067.

[7] McCollum J M, Peterson G D, Cox C D, et al. The sorting direct method for stochastic simulation of biochemical systems with varying reaction execution behavior[J]. Comput Biol Chem, 2006, 30(1): 39-49.

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

[9] Chatterjee A, Vlachos D G, Katsoulakis M A. Binomial distribution based tau-leap accelerated stochastic simulation[J]. J Chem Phys, 2005, 122(2): 024112.

[10] Auger A, Chatelain P, Koumoutsakos P. R-leaping: Accelerating the stochastic simulation algorithm by reaction leaps[J]. J Chem Phys, 2006, 125(8): 084103.

[11] Cai X D, Xu Z Y. K-leap method for accelerating stochastic simulation of coupled chemical reactions[J]. J Chem Phys, 2007, 126(7): 074102.

[12] Gillespie D T, Petzold L R. Improved leap-size selection for accelerated stochastic simulation[J]. J Chem Phys, 2003, 119(16): 8229-8234.

13] Cao Y, Gillespie D T, Petzold L R. Avoiding negative populations in explicit tau leaping method[J]. J Chem Phys, 2005, 123(5): 054104-054112.

[14] Cao Y, Gillespie D T, Petzold L R. Efficient step size selection for the tau-leaping simulation method [J]. J Chem Phys, 2006, 124(4): 044109.

L-leaping: Accelerating the Stochastic Simulation of Chemically Reacting Systems

PENG Xin-jun, WANG Yi-fei

(Department of Mathematics, Shanghai University, Shanghai 200444, P.R. China)

Abstract: Presented here is an L-leap method for accelerating stochastic simulation of well stirred chemically reacting systems, in which the number of reactions occurring of a reaction channel with the largest propensity function is calculated from the leap condition and the numbers of reactions occurring of the other reaction channels are generated by using binomial random variables during a leap. The L-leap method can better satisfy the leap condition. Numerical simulation results indicate that the L-Leap method can obtain better performance than established methods.

Key words: L-leap algorithm; leap condition; stochastic simulation algorithm; chemically reacting system