STOCHASTIC TRANSIENT ANALYSIS OF BIOCHEMICAL SYSTEMS AND ITS APPLICATION TO THE DESIGN OF BIOCHEMICAL LOGIC GATES

BIN CHENG AND MARC RIEDEL

Electrical and Computer Engineering
University of Minnesota
200 Union Street S.E.
Minneapolis, MN 55455
E-mail: bincheng@umn.edu, mriedel@umn.edu

In order to better characterize the behavior of biochemical systems, it is sometimes helpful and necessary to introduce *time-dependent input signals*. If the state of a biochemical system with such signals is assumed to evolve deterministically and continuously, then it can be readily analyzed by solving ordinary differential equations. However, if it assumed to evolve discretely and stochastically, then existing simulation methods cannot be applied. In this paper, we incorporate conditions for *transient analysis* into stochastic simulation and we develop the corresponding simulation algorithm. Applying our method to examples, we demonstrate that it can yield new insights into the dynamics of biochemical systems; specifically, it can be used to verify the design of biochemical logic gates.

1. Introduction

Certain biochemical systems appear to exploit randomness, choosing between different outcomes with a probability distribution – in effect, hedging their bets with a portfolio of responses. Examples include the *pap pili* epigenetic response of bacteria [1], the lentiviral positive-feedback loop in the HIV virus [2], and the lysis/lysogeny switch of the *lambda* bacteriophage [3].

Gillespie proposed the **stochastic simulation algorithm** (SSA) to characterize discrete, random biochemical reactions [4]. The SSA tracks integer quantities of the molecular species in a biochemical system, executing reactions at random based on propensity calculations. Repeated trials are performed to characterize the evolution of the system. Gillespie demonstrated that the SSA has a firmer physical basis than continuous, deterministic methods and that it provides more accurate simulation results [5]. The SSA works well for systems in which the quantities of the species are small. However, for larger systems, the computation time becomes prohibitive. Improved algorithms have been proposed [6]. Also, approximation methods have been applied [7], [8], [9], [10], [11].

In a typical application of the SSA, the system is assumed to be closed: after the initial state is fixed, the time evolution of the system only depends on the internal reactions. But what if we want to characterize the behavior of a system when there exist external mechanisms that modify the quantities of species? For instance, we may need to study what happens when an external source injects or drains certain species into or from the system. The external change might occur periodically or it might be dependent on the trajectory of the system, e.g., the quantities of species might be limited by threshold conditions. Existing methods, such as Gillespie's SSA, cannot handle such behaviors.

For electrical circuits, transient analysis consists of a time sweep using numerical methods to solve differential equations, with the operating points solved by setting all the time derivatives to zero. Similarly, in a biochemical system, various input signals can be defined and the system can be solved through ordinary differential equations (ODEs) or differential algebraic equations (DAEs). Indeed, several authors have suggested using a standard electrical simulation tool called SPICE to model biochemical reactions [12], [13], [14], [15], [16]. However, this presumes that the biochemical systems under investigation are continuous and deterministic.

In this paper, we propose an approach called **stochastic transient analysis** (STA). It incorporates time-dependent variations in the quantities of species into stochastic simulation. We consider pulse, piecewise-linear, and sinusoidal signals. The method can be readily generalized to include other free-form functions. The signals are either *forced* or *injected*. Further, threshold-triggered signals are incorporated.

We apply the STA method to analyze a simple model called *Lotka*. We also apply it to analyze the time-dependent behavior of the inverter model proposed in [17]. We propose designs for other types of biochemical logic gates (AND, OR, NAND and XOR) and we apply our STA method to verify the behavior of these.

2. Stochastic Transient Analysis

The power of transient analysis resides in the fact that the time behavior of the system can be simulated with different types of input signals to accommodate different analysis scenarios.

2.1. *Categorizing Input Signals for Stochastic Transient Analysis*

Although we do not discuss transient analysis of electric circuits here, we borrow from this field standard forms in which input signals can be categorized: pulse, piecewise linear (PWL) and sinusoidal.

(1) $PULSE(v_1, v_2, t_d, t_r, t_f, p_w, P)$, where v_1 is the initial value, v_2 is the pulsed value, t_d is the delay time, t_r is the rise time, t_f is the fall time, p_w is the pulse width and P is the period.
(2) $PWL(t_1, v_1, t_2, v_2, \ldots)$, where v_1 is the input value at time t_1, v_2 the input value at time t_2, etc.
(3) $SIN(v_0, v_a, F, t_d, \theta, \phi)$, where v_0 is the offset, v_a is the amplitude, F is the frequency, t_d is the delay, θ is the damping factor and ϕ is the phase.

In addition to the *form* of the input signals, we distinguish between two ways of *imposing* them onto the system: they are either *forced* or *injected*. The former means that the quantity of the species is set by some external mechanism rather than by the internal reactions. The latter means that an external mechanism adds to or subtracts from the quantity of the species that are present.

Also, we allow for *threshold-triggered* input signals: these are signals that start or stop based on threshold quantities of species. This is useful in simulating boundary conditions, say between qualitatively different phases in the evolution of a biochemical system. For example, suppose that after reaching a threshold quantity of 1000 for a species, the system is deemed to go through a transition: above this threshold, an external mechanism begins injecting new species.

The format of an input signal for our STA method is summarized as follows. Here $<>$ means that the contents are mandatory while [] means that they are optional. (Some details are omitted; these will be implementation-dependent.)

$$\begin{array}{l} < SPECIES > < INITIAL > \\ [< FUNCTION > < FORCED \mid INJECTED >] \; [CONDITION] \end{array} \qquad (1)$$

where $SPECIES$ is the species name; $INITIAL$ is the initial quantity; $FUNCTION$ is the definition of an input signal function; $FORCED$ or $INJECTED$ is the way that the input is imposed; and $CONDITION$ is the definition of a threshold condition on the quantity of a species or a set of species.

2.2. *Stochastic Transient Analysis Algorithm*

Our transient analysis method could be incorporated into any of the proposed stochastic simulation algorithms, for instance that described in [6]. For simplicity, we describe its implementation in terms of Gillespie's direct SSA, proposed in [4].

Suppose we have a biochemical system consisting of n different species interacting through m different reactions. The species are denoted by $x_i (i = 1, \ldots, n)$ and their corresponding quantities are $X_i (i = 1, \ldots, n)$. The reactions are denoted by $R_i (i = 1, \ldots, m)$ and the corresponding rates are $c_i (i = 1, \ldots, m)$. The system state at time point t is denoted by $S_t = (X_1(t), \ldots, X_n(t))$. At the start of the simulation, the system state is $S_0 = (X_1(0), \ldots, X_n(0))$. At each step of the simulation, the next reaction to fire as well as the time point at which this reaction fires are decided. According to Gillespie, these are computed as follows:

$$\sum_{\nu=1}^{\mu-1} a_\nu < r_1 a_0 \leq \sum_{\nu=1}^{\mu} a_\nu, \tag{2}$$

$$\tau = (1/a_0) ln(1/r_2), \tag{3}$$

$$a_0 \equiv \sum_{\nu=1}^{m} a_\nu, \tag{4}$$

$$a_\nu \equiv h_\nu c_\nu (\nu = 1, \ldots, m), \tag{5}$$

where μ is the next reaction that fires; τ is the time elapsing from the present until the next reaction fires; r_1 and r_2 are two random numbers generated by a unit-interval uniform random number generator; h_ν is the number of distinct molecular reactant combinations available for reaction R_ν in state $(X_1, \ldots, X_n)$; and c_ν is the *rate* for reaction R_ν.

In transient analysis, different types of input signals can be specified.

- For forced signals, the value of the species is sampled from the function that is defined, taken at the current time point; the next reaction has no effect on its value.
- For injected signals, the amount to be injected is calculated by subtracting the function value at the current time point t_c from the function value at the next time point t_n:

  $$\text{injected number} = \lfloor f(t_n) \rfloor - \lfloor f(t_c)) \rfloor. \tag{6}$$

  This amount is added to the quantity of the corresponding species together with the change in quantity that the next reaction produces.

If a threshold condition exists, then a forced or injected signal is only applied if the threshold condition is met.

We note that with this increased flexibility in the simulation of input signals, the amount of data that must be recorded increases. If implemented as described above, the amount of data generated increases proportionally with the number of trials. To mitigate against this, we propose the following data structure; it requires constant memory, independent of the number of trials.

Suppose T is the simulation time, and N is the total number of simulation steps. We separate T into N time segments equally: $[(i-1)T/N, iT/N](i = 1, \ldots, N)$. A *accumulated* state $S^{(i)}(i = 1, \ldots, N)$ is attached to each corresponding time segment. For each time point in each trial, if the time point falls in i-th time segment, then the corresponding state is added to the i-th accumulated state $S^{(i)}$. At the end of simulation, each accumulated state is averaged by dividing it by the total number of trials that landed in the corresponding time segment. The final results can be plotted by using N general states at N time points. The data structure is shown in Fig. 1.

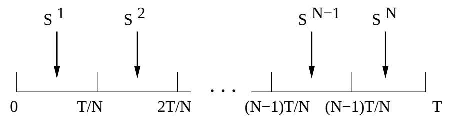


Figure 1. Time segmenting for multi-trial STA.

3. Stochastic Transient Analysis of the Lotka Model

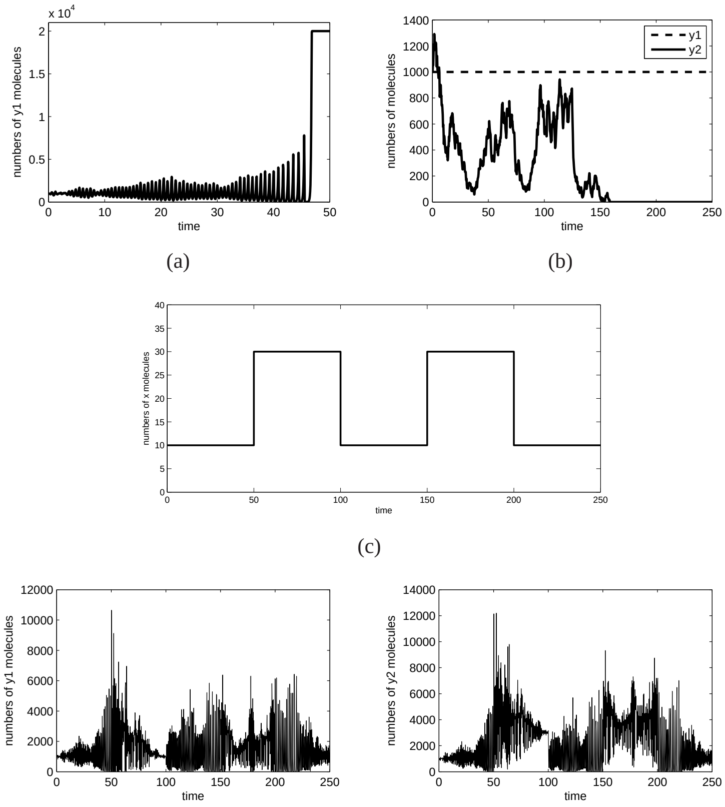
In this section, we apply our method to a simple biochemical model called *Lotka*, described by the following set of coupled reactions:

$$x+y_1 \xrightarrow{c_1} x+2y_1, \quad y_1+y_2 \xrightarrow{c_2} 2y_2, \quad y_2 \xrightarrow{c_3} z \qquad (7)$$

In [4], Gillespie analyzes the Lotka model. Some of the conclusions that he presents are qualitative deductions. For instance, he writes that “no matter what the state of the system is initially, it will eventually wind up in either the state ($y_1 = 0$, $y_2 = 0$) or the state ($y_1 = \infty, y_2 = 0$).” No simulation results are provided to support this. Here we show how transient analysis can be used to elucidate such behaviors.

Note that we can characterize the situation where the model winds up in the state ($y_1 = 0, y_2 = 0$) simply by simulating it for enough steps. However, we cannot do the same to characterize the situation where it winds up in the state ($y_1 = \infty, y_2 = 0$). In this case, as y_1 approaches infinity, the time steps between reactions firing become infinitesimal small; eventually, an infinite number of reactions fire per time unit. Gillespie’s SSA cannot handle such limiting conditions. However, with our transient analysis method, we can characterize such limits by defining threshold events. If we constrain the output of y_1 to be, say 20,000, as an approximation of infinity, we will get the waveform in Fig. 2(a).

Now suppose there is some external mechanism which forces y_1 to stay at a constant value of 1000. The waveform of Fig. 2(b) illustrates the simulation results for such forcing. Interestingly, we observe that y_2 still gets to zero, even though this takes much longer. We expect that the value of y_2 should be proportional to x, while the value of y_1 should not change with x. This can be verified through the simulation results depicted in the two waveforms of Fig. 2(d) and Fig. 2(e), where x is set to be a pulse function of time: $PULSE(10\ 30\ 50\ 0\ 0\ 50\ 100)$, as displayed in Fig. 2(c). Here threshold events are defined: once y_1 or y_2 reaches zero, more of the corresponding species is injected



(a) (b)

(c)

(d) (e)

Figure 2. Stochastic transient analysis of the Lotka model, 3 trials. (a) System reaching state ($y_1 = \infty, y_2 = 0$); (b) y_1 is forced to a constant; (c) x is set to a pulse function of time; (d) y_1 does not change with x; (e) y_2 changes with x.

to keep the fluctuations going.

4. Stochastic Transient Analysis of Biochemical Logic Gates

In this section, we apply our STA method to the design of biochemical logic gates. First, we consider the inverter model proposed in [17]. Then, we propose a simple model for other logic gates: AND, NAND, OR and XOR. We simulate these using our STA method to characterize their behavior.

4.1. *Biochemical Inverter*

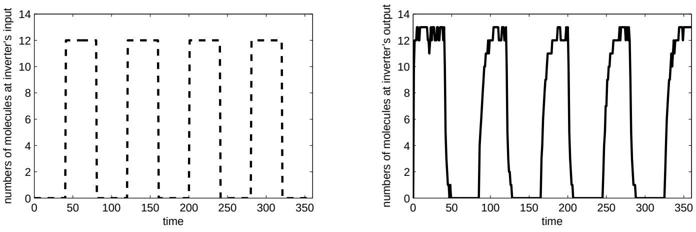
Some of the behaviors of gene regulation systems can be characterized as logical operations. For instance, RNA polymerase will stop transcribing a gene if there exists a repressor protein which binds to the operator of the gene's promoter. If one considers the concentration of the gene and the concentration of the repressor as two signals, then the relationship between them is like an inverter. In [17], the authors present the following set of coupled reactions modeling this behavior. They analyze their model, deterministically. Here we apply our stochastic transient analysis method.

$$\begin{array}{lll}
a+a \xrightarrow{K_{dim(a)}} a2, & a2 \xrightarrow{K_{sngl(a)}} a+a, & z+z \xrightarrow{K_{dim(z)}} z2, \\
z2 \xrightarrow{K_{sngl(z)}} z+z, & gz+a2 \xrightarrow{K_{rprs(a2)}} gza2, & gza2 \xrightarrow{K_{dis(a2)}} gz+a2, \\
gza2+a2 \xrightarrow{K_{rprs(a4)}} gza4, & gza4 \xrightarrow{K_{dis(a4)}} gza2+a2, & a \xrightarrow{K_{dec(a)}} ø, \\
a2 \xrightarrow{K_{dec(a2)}} ø, & z \xrightarrow{K_{dec(z)}} ø, & z2 \xrightarrow{K_{dec(z2)}} ø, \\
gza2 \xrightarrow{K_{dec(ga2)}} gz, & gza4 \xrightarrow{K_{dec(ga4)}} gza2, & mrnaz \xrightarrow{K_{dec(mrna)}} ø,
\end{array}$$

$$gz+rnap \xrightarrow{K_{xscribe}} gz+rnap+mrnaz, \; mrna2+rnaa \xrightarrow{K_{xlate}} mrnaz+rrna+z \tag{8}$$

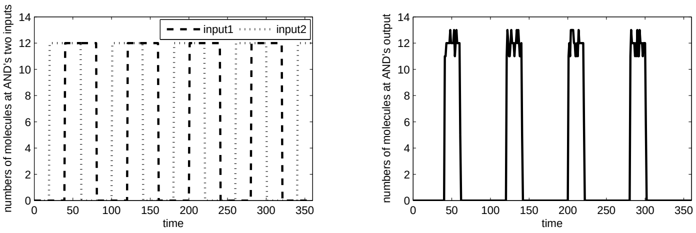
In the model, the species *a* represents the input to the inverter, and the species *z* its output. In the simulation, we first provide a logical value '0' as the input, i.e., we set the quantity of the species *a* to zero. We expect to get logical '1' at the output; this corresponds to a quantity of about 12 of *z*.

To impose logic '1' at the input, the authors in [17] suggest that an externally-imposed drive is needed to increase the quantity of species *a*. Based on the strength of the external drive, *a* achieves an equilibrium at the bottom or at the top of its signal range. Accordingly, the transfer curve of the inverter can be drawn by changing the external drive from weak to strong, gradually. Here we consider a pulse input signal, shown in Fig. 3(a). We simulate the model using our STA method; the results are plotted in Fig. 3(b). These waveforms clearly depict the transient behavior of an inverter: when the input signal is low, the output signal is high and vice versa.



(a) (b)

Figure 3. Stochastic transient analysis of the inverter, 3 trials. (a) Input, (b) Output.



(a) (b)

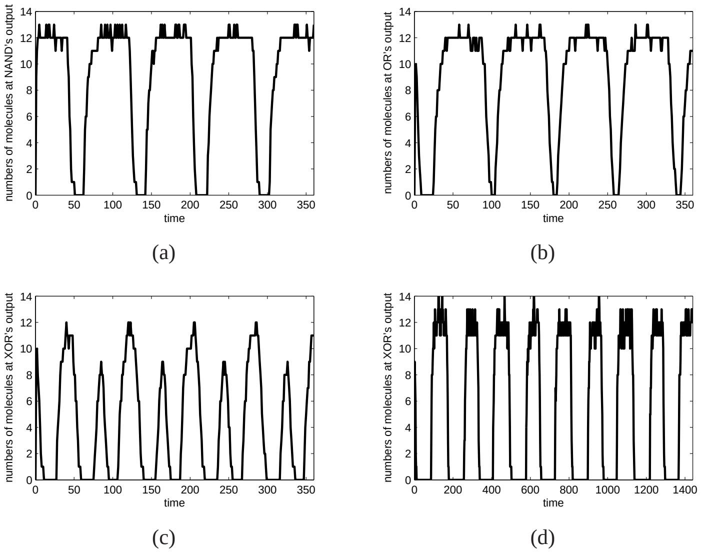
Figure 4. Stochastic transient analysis of the AND gate, 3 trials. (a) Input; (b) Output.

4.2. *Logic Gates*

In [17], the approach taken to create a NAND gate is to "wire-OR" the outputs of multiple inverters by assigning them the same output gene. However no reaction model is given for the NAND gate. Here, we explicitly design logic gates with reaction models and then use STA to verify our models.

First we design the reaction model of an AND gate. This is given in (9). The reaction constants c_1 and c_2 can be adjusted to get the same quantity level for logic '1' as for the inverter ($c_2/c_1 \approx 12$). To verify the model, we apply as input signals two forced pulse signals with the same period but different phases, as shown in Fig. 4(a). The simulation result is plotted in Fig. 4(b). The waveform clearly depicts the transient behavior of an AND gate.

$$\begin{array}{l} x+y \xrightarrow{c_1} x+y+z \\ z \xrightarrow{c_2} \emptyset \end{array} . \tag{9}$$



(a) (b)

(c) (d)

Figure 5. Stochastic transient analysis, 3 trials. (a) NAND, (b) OR, (c) XOR with high frequency, (d) XOR with low frequency.

Other logic gates are designed as follows.

- By simply connecting an inverter to the output of the AND gate, we obtain a NAND gate. Here the output of the AND gate acts as the external drive for the input of the inverter. Using the same input signals as for the AND gate simulation, we obtain the simulation results shown in Fig. 5(a).
- We can obtain an OR gate by hooking three inverters on an AND gate, one on each input and one on the output. The result is a total of 53 chemical reactions; these are not listed here. The simulation results are shown in Fig. 5(b), using the same input signals as for the AND gate.
- We can obtain an exclusive-OR (XOR) gate with two inverters, two AND gates, and one OR gate. The result is a total of 91 chemical reactions; these are not listed here. We show two different simulation results for the XOR gate: one in which the input signals are changing from low to high more rapidly, Fig. 5(c), and one where they are are changing less rapidly, Fig. 5(d). In Fig. 5(c), the output does not have enough time to reach the high level corresponding to logical ‘1’. In contrast, in Fig. 5(d), we obtain a clean XOR response.

5. Conclusions

The characterization of the XOR gate in the preceding section illustrates the sort of information that transient analysis provides: it allows us to characterize not only the input-output response of a system, but also its temporal dynamics. Our implementation of the method provides the flexibility to characterize such temporal dynamics for a variety of analysis scenarios. We are currently working on incorporating the method into simulation tools such as BioSPICE [18]. Also, we are applying it to problems in the computer-aided design of synthetic modules of biochemistry [19].

References

1. A. Hernday, B. Braaten, and D. Low. The Intricate Workings of a Bacterial Epigenetic Switch. *Advances in Experimental Medicine & Biology*, pages 547: 83–9, 2004.
2. L. Weinberger, J. Burnett, J. Toettcher, A. Arkin, and D. Schaffer. Stochastic Gene Expression in a Lentiviral Positive-Feedback Loop: HIV-1 Tat Fluctuations Drive Phenotypic Diversity. *Cell*, 122:169–182, 2005.
3. A. Arkin, J. Ross, and H. McAdams. Stochastic Kinetic Analysis of Developmental Pathway Bifurcation in Phage λ-Infected Escherichia Coli Cells. *Genetics*, pages 149:1633–1648, 1998.
4. D. Gillespie. Exact Stochastic Simulation of Coupled Chemical Reactions. *Journal of Physical Chemistry*, 81:No. 25, pp. 2340–2361, 1977.
5. D. Gillespie. Stochastic Simulation of Chemical Kinetics. *Annual Review of Physical Chemistry*, pages 58: 35–55, 2007.
6. M. Gibson and J. Bruck. Efficient Exact Stochastic Simulation of Chemical Systems with Many Species and Many Channels. *Journal of Physical Chemistry*, pages 105, 1876–1889, 2000.
7. D. Gillespie. Approximate Accelerated Stochastic Simulation of Chemically Reacting Systems. *J. Chem. Phys.*, pages 115(4), 1716–1733, 2001.
8. M. Rathinam, Y. Cao, L. Petzold, and D. Gillespie. Stiffness in Stochastic Chemically Reacting Systems: The Implicit Tau-leaping Method. *J. Chem. Phys.*, 2003.
9. Y. Cao, D. Gillespie, and L. Petzold. Avoiding Negative Populations in Explicit Tau Leaping. *J. Chem. Phys.*, pages 123, 054104–054112, 2005.
10. Y. Cao, D. Gillespie, and L. Petzold. Efficient Stepsize Selection for the Tau-leaping Method. *J. Chem. Phys.*, pages 124, 044109, 2006.
11. Y. Cao, D. Gillespie, and L. Petzold. The Adaptive Explicit-implicit Tau-leaping Method with Automatic Tau Selection. *J. Chem. Phys.*, pages 126(22), 224101–224109, 2007.
12. L. Nagel. SPICE2: A Computer Program to Simulate Semiconductor Circuits. *Memorandum No. UCB/ERL M520. Electronics Research Lab., College of Engi., UC, Berkeley*, 5 1985.
13. J. May and D. Mikulecky. Glucose Utilization in Rat Adipocytes. *Journal of Biological Chemistry*, pages 258, No. 8, pp. 4881–4888, 4 1983.
14. R. Seither, D. Hearne, D. Trent, D. Mikulecky, and D. Goldman. SPICE2 Network Simulation of Antifolate Effects on Purine and Pyrimidine Biosynthesis: Exploring the Role of Tetrahydrofolate Cofactor Depletion Versus Dihydrofolate Feedback Inhi–

bition. *Computera and Mathematics with Applications*, pages 20, No. 4–6, pp. 88–101, 1990.

15. R. Seither, D. Trent, D. Mikulecky, T. Rape, and D. Goldman. Effect of Direct Suppression of Thymidylate Synthase at the 5,l0-Methylenetetrahydrofolate Binding Site on the Interconversion of Tetrahydrofolate Cofactors to Dihydrofolate by Antifolates. *Journal Of Biological Chemistry*, pages 266, No. 8, pp. 18016–18023, 3 1991.
16. A. Brower, R. Ewing, R. Brower, and H. Abdel-Aty-Zohdy. SPICE for Modeling Biochemical Networks. *Circuits and Systems, 2005. 48th Midwest Symposium on*, 1:491–494, 8 2005.
17. R. Weiss, G. Homsy, and T. Knight Jr. Toward In-vivo Digital Circuits. *In Dimacs Workshop on Evolution as Computation Princeton NJ*, 1 1999.
18. S. Kumar and J. Feidler. BioSPICE: A Computational Infrastructure for Integrative Biology. *OMICS: A Journal of Integrative Biology*, pages 7(3): 225–225, 9 2003.
19. B. Fett, J. Bruck, and M. Riedel. Synthesizing Stochasticity in Biochemical Systems. *Proceedings of the 44th Annual Conference on Design Automation*, pages 640–645, 2007.