An Efficient Parallel Stochastic Simulation Method for Analysis of Nonviral Gene Delivery Systems

Hiroyuki Kuwahara
Lane Center for Computational Biology
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA, U.S.A.
kuwahara@andrew.cmu.edu

Xin Gao*
Mathematical and Computer Sciences and Engineering Division
King Abdullah University of Science and Technology, Thuwal, Saudi Arabia
xin.gao@kaust.edu.sa

ABSTRACT

Gene therapy has a great potential to become an effective treatment for a wide variety of diseases. One of the main challenges to make gene therapy practical in clinical settings is the development of efficient and safe mechanisms to deliver foreign DNA molecules into the nucleus of target cells. Several computational and experimental studies have shown that the design process of synthetic gene transfer vectors can be greatly enhanced by computational modeling and simulation. This paper proposes a novel, effective parallelization of the *stochastic simulation algorithm* (SSA) for pharmacokinetic models that characterize the rate-limiting, multi-step processes of intracellular gene delivery. While efficient parallelizations of the SSA are still an open problem in a general setting, the proposed parallel simulation method is able to substantially accelerate the next reaction selection scheme and the reaction update scheme in the SSA by exploiting and decomposing the structures of stochastic gene delivery models. This, thus, makes computationally intensive analysis such as parameter optimizations and gene dosage control for specific cell types, gene vectors, and transgene expression stability substantially more practical than that could otherwise be with the standard SSA. Here, we translated the nonviral gene delivery model based on mass-action kinetics by Varga *et al.* [*Molecular Therapy*, **4**(5), 2001] into a more realistic model that captures intracellular fluctuations based on stochastic chemical kinetics, and as a case study we applied our parallel simulation to this stochastic model. Our results show that our simulation method is able to increase the efficiency of statistical analysis by at least 50% in various settings.

Categories and Subject Descriptors

I.6 [**Computing Methodologies**]: Simulation and Modeling; I.6.8 [**Simulation and Modeling**]: Types of Simulation—*Parallel*

*To whom correspondence should be addressed.

CMSB '11, September 21-23, 2011 Paris, FR

1. INTRODUCTION

Gene therapy is a promising therapeutic tool to treat a wide spectrum of acquired and inherited human diseases [1, 25]. Since the first human gene therapy trial in 1989 [30], over 1,300 gene therapy clinical trials have been reported worldwide, the vast majority of which have targeted cancers, heart diseases, and inherited monogenic disorders such as cystic fibrosis and severe combined immunodeficiency [33, 10, 8]. Since naked exogenous DNA molecules cannot be transported into the nucleus of target cells efficiently, one of the main challenges for gene therapy is the development of efficient and safe gene delivery methods [9, 1]. Since viruses have natural biological machineries for efficient transfer of their genetic materials within host cells, one approach to gene delivery is to utilize the gene transduction machinery of viruses for gene carriers (vectors) [39]. However, viral vectors have major drawbacks including their acute immune response and insertional mutagenesis, raising serious safety concerns in their use for gene delivery as it may ultimately lead to fatalities [33, 6, 17, 9]. Synthetic, nonviral gene delivery systems have safety advantages over viral counterparts as the materials for nonviral counterparts are generally less toxic, immunogenic and mutagenic [17, 1]. Despite these competitive advantages over viral vectors, nonviral delivery systems have been largely ignored until recently due to their inefficient transfection and expression of transgenes in target cells, especially *in vivo* [17, 1].

Computational modeling approaches can be a powerful tool to aid quantitative analysis of how nonviral vectors can be designed to optimize for the overall gene delivery process. Indeed, previous studies have shown that such a computational modeling approach can be useful to characterize and design optimized nonviral vectors for specific objectives [37, 4, 38, 40, 28]. While most, if not all, of the previous pharmacokinetic modeling studies describe the dynamics of intracellular gene delivery systems as continuous-deterministic processes based on mass-action kinetics, a more accurate picture of the dynamics of a gene delivery system can be attained through a discrete-stochastic process description based on *stochastic chemical kinetics* (SCK) [26, 14, 16], as it takes into account underlying discreteness and intrinsic fluctuations in the system. Unlike the deterministic process description, this approach can capture the time evolution of distribution of each molecular species, making more accurate analyses such as dosage control based on cytotoxicity from relatively high weight vectors [40] possible. While exact numerical

realizations of the time evolution of a SCK model can be generated via the *stochastic simulation algorithm* (SSA) [12, 13], the computational requirements of analysis based on the SSA can be substantial due to the statistical nature of the analysis which requires a large number of sample trajectories in order to estimate the system behavior at a reasonable degree of confidence.

In order to speed up statistical analysis of SCK models, including stochastic nonviral delivery models, parallel stochastic simulation methods have been studied [3]. Owing to intrinsically sequential process of the SSA, however, effective parallelizations at the single simulation level may be difficult for statistical analysis based on a large sample size. Perhaps, the most cost-effective parallelization of the SSA-based statistical analysis in general, to date, is to simply generate independent trajectories of a SCK model in parallel as this approach has no overhead from synchronization. By exploiting this independency of each sample trajectory, several tools and frameworks for statistical analysis of SCK models (e.g., those described in [34, 19, 2, 24, 41]) have been developed.

However, the traditional parallelization methods, which parallelize independent trajectories of simulations, have an obvious bottleneck. That is, such methods can never be more efficient than a single simulation run, which is the elementary unit for such parallelization methods. In this paper, we propose a novel parallelization method of the SSA of intracellular nonviral gene delivery models to decompose and parallelize at the single simulation run level. By focusing on shared properties in this specific subclass of SCK models, our approach is able to decompose a model into sub-models at a single molecular level, each of which is then efficiently simulated independently. Efficiency of the stochastic simulation of these "decomposed" sub-models comes from acceleration in both the next reaction selection scheme and the reaction update scheme. In this paper, we show evidence that this new approach can substantially increase the computational efficiency of statistical analysis of intracellular nonviral gene delivery systems. Our results from a specific system indicate that the proposed method is able to increase the efficiency of statistical analysis by at least 50% in various settings compared with the standard SSA.

2. NONVIRAL GENE DELIVERY MODELS

Several computational kinetic models have been constructed to quantitatively analyze design properties of specific nonviral gene delivery systems [28]. For example, Ledley and Ledley constructed a kinetic model for gene transfer events based on the half-lives of DNA, RNA, and the associated protein to understand the effect of changes in various gene delivery parameters [23]. Varga *et al.* modeled complex cellular and molecular processes of a lipofectamine-mediated gene delivery pathway to gain insights into optimal design of the polymer length [37]. Varga *et al.* further extended this pharmacokinetic model to capture an adenovirus-based vector and various polyethlenimine (PEI) vectors [38]. Their comparative analysis showed that the advantage of the adenoviral vector over PEI-based vectors might be transport efficacy for the nuclear import and PEI-based vectors could be significantly improved by optimizing for the endosomal escape rate. Banks *et al.* quantitatively modeled a gene delivery system for human epithelioid cells and monkey fibroblast cells to analyze how the values of parameters differ at each transgene delivery step between the two [4]. Zhou *et al.* modeled *in vitro* gene delivery for mouse melanoma cells using various PEI-based vectors to investigate how differences in the molecular weight of gene carriers can affect the quantitative profile of each transgene uptake step [40]. By analyzing parameter sensitivities of this model, they were, then, able to gain insights into the design of this particular nonviral gene delivery system by optimizing the parameter values for the overall transfection efficacy.

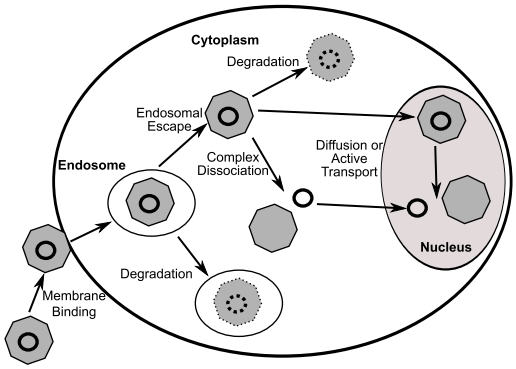

Figure 1: A cartoon to illustrate the rate limiting barriers of nonviral gene transfer using polyethlenimine carriers (based on Fig. 1 of ref. [40]).

These pharmacokinetic models quantitatively characterize rate-limiting barriers involved in specific nonviral vectors for specific cell types. Fig. 1 depicts several rate-limiting barriers involved in kinetics of specific PEI-based vectors for mouse melanoma cells, which were addressed by Zhou *et al.* [40]. Here, rate-limiting barriers include: (1) the binding of the polymer-plasmid complex to the cell membrane; (2) the internalization of the complex by endosomal uptake; (3) the degradation of the complex in endosome; (4) the endosomal escape of the complex into the cytoplasm; (5) the cytoplasmic degradation of the complex; (6) the cytoplasmic dissociation of the complex into the vector and the plasmid DNA; (7) the active/passive nuclear import of the complex or the plasmid alone; and (8) the dissociation of the complex in the nucleus.

In the aforementioned pharmacokinetic studies of nonviral gene delivery systems, the underlying process of transgene delivery is assumed to be characterized by a continuous-deterministic system description, which is governed by a set of ordinary differential equations based on mass-action kinetics [18, 21]. Aside from this shared model description, there are two common characteristics in these pharmacokinetic models: (1) each rate-limiting barrier is represented by a unimolecular reaction, which has only one reactant; and (2) a reaction network is sparse in that each species is used as a reactant in a relatively small number of reactions. Here, sparsity is defined using the definition of dependency graph [11] such that the system is sparse when the number

of edges in each vertex is small in its dependency graph. In the proposed parallel simulation of stochastic nonviral gene delivery models based on the SCK framework, thus, we take advantage of these features.

3. STOCHASTIC CHEMICAL KINETICS

Unlike mass-action kinetics, SCK characterizes the time evolution of biochemical systems as a discrete-stochastic process [26, 14, 16]. The system state of a SCK model is the molecule populations of $\mathbf{S} \equiv (S_1, \ldots, S_N)$, the N-dimensional vector of molecular species, whose values are updated by $\mathbf{R} \equiv (R_1, \ldots, R_M)$, the M-dimensional vector of biological reactions.

Definition 1. An SCK model is a 5-tuple $\langle t, \mathbf{X}, \mathbf{a}, \mathbf{V}, \mathbf{x_0} \rangle$ where:

- t is the independent, continuous variable that represents the time, which is initially set to 0;
- $\mathbf{X}(t) \equiv (X_1(t), \ldots, X_N(t))$ is the system state at time t which is given by the N-dimensional vector of integer random variables where $X_i(t)$ represents the molecule count of S_i at time t;
- $\mathbf{a} \equiv (a_1, \ldots, a_M)$ is the M-dimensional vector of functions whose j-th element, $a_j : \mathbb{Z}^N \mapsto \mathbb{R}$, is the propensity function of R_j which is defined such that, given $\mathbf{X}(t) = \mathbf{x}$, $a_j(\mathbf{x})dt$ is the probability that one R_j reaction occurs in the next infinitesimal time interval $[t, t + dt)$;
- $\mathbf{V} \equiv (\mathbf{v_1}, \ldots, \mathbf{v_M})$ is the M-dimensional vector whose j-th element, $\mathbf{v_j} \equiv (v_{j1}, \ldots, v_{jN})$ is the N-dimensional vector of integers whose i-th element v_{ji} specifies the change in X_i per one R_j event;
- $\mathbf{x_0} \equiv (x_{01}, \ldots, x_{0N})$ is the N-dimensional vector of integers whose i-th element represents the initial state of X_i as $X_i(0) = x_0^i$.

For example, the SCK model of the following biological reactions:

$$R_1 : \; S_1 \xrightarrow{c_1} S_2, \qquad R_2 : \; S_2 \xrightarrow{c_2} S_3,$$

where $\mathbf{x_0} = (3, 0, 0)$ is given by:

$$\mathcal{M}_e \equiv \langle t, \mathbf{X}, (a_1, a_2), \mathbf{V}, \mathbf{x_0} \rangle \tag{1}$$

where

$$\begin{aligned} \mathbf{X} &= (X_1, X_2, X_3), \\ a_1(\mathbf{X}) = c_1 X_1, \; a_2(\mathbf{X}) &= c_2 X_2, \\ \mathbf{V} &= ((-1, 1, 0), (0, -1, 1)) \\ \mathbf{x_0} &= (3, 0, 0). \end{aligned}$$

Here, $a_1(\mathbf{X})$ and $a_2(\mathbf{X})$ are said to follow first-order reaction kinetics because each of these reactions is a unimolecular reaction whose propensity function is linear and proportional to the molecular count of the reactant.

The system state of the SCK model is $\mathbf{X}$ which evolves over continuous time t where the system starts with $\mathbf{X}(0) = \mathbf{x_0}$. In the SCK model, each reaction R_j is completely characterized by $\mathbf{v_j}$ and a_j. Given that $\mathbf{X}(t) = \mathbf{x}$, if one R_j occurs next, then the system moves to state $\mathbf{x} + \mathbf{v_j}$. As described in Definition 1, the probability that one R_j event happens in the next infinitesimal time dt is characterized by a_j, which is dependent only on the current state, and not on any previous states. This memoryless property implies that the SCK model is a Markov process. This Markovian treatment can be justified by making the well-stirred assumption, where the molecules of species are uniformly distributed in the system.

From Definition 1, the time evolution of $P(\mathbf{x}, t \mid \mathbf{x_0})$, the probability that $\mathbf{X}(t) = \mathbf{x}$ given $\mathbf{X}(0) = \mathbf{x_0}$ as:

$$\begin{aligned} P(\mathbf{x}, t + dt \mid \mathbf{x_0}) = P(\mathbf{x}, t \mid \mathbf{x_0}) \left[1 - \sum_{j=1}^{M} a_j(\mathbf{x}) dt \right] \\ + \sum_{j=1}^{M} [P(\mathbf{x} - \mathbf{v_j}, t \mid \mathbf{x_0}) a_j(\mathbf{x} - \mathbf{v_j}) dt]. \end{aligned} \tag{2}$$

This can be rewritten as the following difference-differential equation as $dt \to 0$:

$$\begin{aligned} \frac{\partial P(\mathbf{x}, t \mid \mathbf{x_0})}{\partial t} = \sum_{j=1}^{M} P(\mathbf{x} - \mathbf{v_j}, t \mid \mathbf{x_0}) a_j(\mathbf{x} - \mathbf{v_j}) \\ - \sum_{j=1}^{M} P(\mathbf{x}, t \mid \mathbf{x_0}) a_j(\mathbf{x}), \end{aligned} \tag{3}$$

which is called the *forward chemical master equation* (CME) [26, 15, 35]. The CME governs the time evolution of the probability distribution of $\mathbf{X}(t)$ given the initial state $\mathbf{x_0}$.

While the numerical solution of the time evolution of a CME can be directly obtained by solving the underlying Markov chain, the state space can be enormous, if not infinite, as this approach needs to exhaustively search for all reachable states from $\mathbf{x_0}$. Thus, the CME itself may not be useful for analyzing the temporal behavior of nontrivial biochemical systems without reducing the system state space with approximations such as those described in [22, 27].

In order to more practically analyze the time evolution of $\mathbf{X}(t)$ within an SCK model, the SSA was developed as a simulation method of SCK models [12, 13]. SSA is derived by defining a probability density function $p(\tau, \mu \mid \mathbf{x}, t)$ such that $p(\tau, \mu \mid \mathbf{x}, t)d\tau$ is the probability that, given $\mathbf{X}(t) = \mathbf{x}$, the next reaction occurs in the infinitesimal time interval $[t + \tau, t + \tau + d\tau)$, and it is R_μ. Then, it can be shown that:

$$p(\tau, \mu \mid \mathbf{x}, t) = a_\mu(\mathbf{x}) \exp(-a_0(\mathbf{x})\tau), \tag{4}$$

where $a_0(\mathbf{x}) \equiv \sum_{j=1}^{M} a_j(\mathbf{x})$. Thus, formally, the SSA is a Monte Carlo simulation procedure that selects τ and μ in a statistically accurate way according to the probability distribution defined in Eq. 4.

Perhaps, the most straightforward implementation of the SSA is the *direct method* which Gillespie proposed in the original paper on the SSA [12]. The direct method, as shown in Algorithm 1, rearranges Eq. 4 as follows:

$$p(\tau, j \mid \mathbf{x}, t) = [a_0(\mathbf{x}) \exp(-a_0(\mathbf{x})\tau)] \times [a_j(\mathbf{x})/a_0(\mathbf{x})]. \tag{5}$$

Since τ and μ are independent, from the first term of the right hand side, τ can be seen as an exponential random

variate with mean $1/a_0(\mathbf{x})$ while, from the second term of the right hand side, μ can be seen as a discrete random variate with probability $a_j(\mathbf{x})/a_0(\mathbf{x})$. Thus, by using two unit uniform random numbers n_1 and n_2, the direct method chooses τ and μ for each iteration as follows: $\tau = -\ln(n_1)/a_0(\mathbf{x})$, and μ is the smallest integer satisfying $\sum_{j=1}^{\mu} a_j(\mathbf{x}) \geq n_2 a_0(\mathbf{x})$.

Algorithm 1 Direct method implementation of the SSA [12]

1: $t \leftarrow 0$, $\mathbf{x} \leftarrow \mathbf{x_0}$
2: evaluate all $a_j(\mathbf{x})$, and calculate $a_0(\mathbf{x})$
3: **repeat**
4: generate two unit uniform random numbers n_1 and n_2
5: $\tau \leftarrow -\ln(n_1)/a_0(\mathbf{x})$
6: $\mu \leftarrow$ the smallest integer satisfying $\sum_{j=1}^{\mu} a_j(\mathbf{x}) \geq n_2 a_0(\mathbf{x})$
7: $t \leftarrow t + \tau$, $\mathbf{x} \leftarrow \mathbf{x} + \mathbf{v}_\mu$
8: update $a_j(\mathbf{x})$ and re-calculate $a_0(\mathbf{x})$
9: **until** $t > t_{max}$

While the direct method is simple, it may be very inefficient for simulation of large-scale models as its next reaction selection scheme runs in linear time with respect to the reaction size (line 6). To alleviate the computational complexity of simulating large-scale models, several streamlined versions of the SSA (e.g. [11, 5, 32, 29]) have been proposed.

4. PARALLEL SIMULATION

Our parallel simulation method, called *Decomposition of SSA* (DoSSA), exploits a shared feature of SCK models for nonviral gene delivery systems to decompose a model into independent sub-models and to effectively parallelize the stochastic simulation procedure. Specifically, DoSSA assumes that each reaction be represented as a first-order reaction. As described in Sec. 2, this assumption is seen to hold in nonviral gene delivery models including those that were experimentally validated.

When a SCK model, $\mathcal{M}$, has only first-order reactions, $\mathcal{M}$ can be decomposed into two independent SCK models, $\mathcal{M}^{(1)}$ and $\mathcal{M}^{(2)}$ so that the sum of these submodels can correctly generate sample trajectories of $\mathcal{M}$. Suppose the molecular population of each species in $\mathcal{M}$ is partitioned in $\mathcal{M}^{(1)}$ and $\mathcal{M}^{(2)}$. Then, since the propensity function of a first-order reaction is linear and additive, the value of each propensity function in $\mathcal{M}$ can be accurately described by the sum of the corresponding propensity functions in $\mathcal{M}^{(1)}$ and $\mathcal{M}^{(2)}$. This property leads us to statistically correct generation of sample trajectories of $\mathcal{M}$ via the sum of $\mathcal{M}^{(1)}$ and $\mathcal{M}^{(2)}$.

LEMMA 1. *Suppose $\mathcal{M} \equiv \langle t, \mathbf{X}, \mathbf{a}, \mathbf{V}, \mathbf{x_0} \rangle$ is a SCK model with N molecular species and M reactions where each of a_j in $\mathbf{a}$ follows first-order reaction kinetics. Then, there exist $\mathcal{M}^{(1)} \equiv \langle t, \mathbf{X^{(1)}}, \mathbf{a}, \mathbf{V}, \mathbf{x_0^{(1)}} \rangle$ and $\mathcal{M}^{(2)} \equiv \langle t, \mathbf{X^{(2)}}, \mathbf{a}, \mathbf{V}, \mathbf{x_0^{(2)}} \rangle$ such that, if $\mathbf{x_0} = \mathbf{x_0^{(1)}} + \mathbf{x_0^{(2)}}$, then $P(\mathbf{x}, t \mid \mathbf{x_0})$ is equivalent to $P_2(\mathbf{x}, t \mid \mathbf{x_0^{(1)}}, \mathbf{x_0^{(2)}})$ for all $t \geq 0$ where $P_2(\mathbf{x}, t \mid \mathbf{x_0^{(1)}}, \mathbf{x_0^{(2)}})$ is the probability that $\mathbf{X^{(1)}}(t) + \mathbf{X^{(2)}}(t) = \mathbf{x}$ given that $\mathbf{X^{(1)}}(0) = \mathbf{x_0^{(1)}}$ and $\mathbf{X^{(2)}}(0) = \mathbf{x_0^{(2)}}$.*

PROOF. The proof is by induction on t. We first show that the equality holds at $t = 0$. Then, by assuming that $P(\mathbf{x}, t \mid \mathbf{x_0}) = P_2(\mathbf{x}, t \mid \mathbf{x_0^{(1)}}, \mathbf{x_0^{(2)}})$, we show that $P(\mathbf{x}, t+dt \mid \mathbf{x_0}) = P_2(\mathbf{x}, t+dt \mid \mathbf{x_0^{(1)}}, \mathbf{x_0^{(2)}})$. The crucial aspect of the proof is that each propensity function, $a_j(\mathbf{x})$, is linear and additive so $a_j(\mathbf{x^{(1)}} + \mathbf{x^{(2)}}) = a_j(\mathbf{x^{(1)}}) + a_j(\mathbf{x^{(2)}})$. The detail of the proof is in Appendix A. □

Each of the sub-models, $\mathcal{M}^{(1)}$ and $\mathcal{M}^{(2)}$, also has the same set of first-order reactions. Thus, by applying Lemma 1 recursively to $\mathcal{M}$, we can further decompose the original model and ultimately end up with sub-models, each of which has the initial condition that the summation of the molecular population is exactly one at time 0. In other words, by letting $n_0 = \sum_i x_{0i}$, we generate as many as n_0 sub-models $\mathcal{M}^{(1)}, \cdots, \mathcal{M}^{(n_0)}$, each of whose initial molecule population, $\mathbf{x_0^{(k)}}$, has the condition that $\sum_i x_{0i}^{(k)} = 1$. Note that this decomposition can be applied even when $\mathcal{M}$ has reversible reactions.

Hereafter, we assume that $\mathbf{x_0} \equiv (n_0, 0, \ldots, 0)$ where $n_0 > 0$ to simplify the presentation of our method. In other words, at time 0, the molecular count of species S_1 is n_0 while the initial molecular counts of the other species are zero. Note that this assumption is seen to hold in the gene delivery models that are described in Sec. 2. Also note that this assumption is not necessary for the DoSSA to work. The decomposition of $\mathcal{M}$ for construction of sub-models is an efficient and simple process as all that changes from the original SCK model is the initial molecular count. From $\mathcal{M}$, then, we generate n_0 sub-models, whose k-th sub-model, $\mathcal{M}^{(k)}$, has the structure $\langle t, \mathbf{X^{(k)}}, \mathbf{a}, \mathbf{V}, (1, 0, \ldots, 0) \rangle$. That is, each sub-model has the initial condition that the molecular count of S_1 is one while the molecular counts of the other species are set to zero. Since each sub-model has the same structure, the time evolution of the probability distribution of the sub-models is identical.

Fig. 2 illustrates this decomposing transformation for $\mathcal{M}_e$. $\mathcal{M}_e$ has two reactions that follow first-order kinetics. The original model has three molecules of S_1 while the molecule counts of other species are initially set to zero. Thus, we can decompose $\mathcal{M}_e$ into three sub-models. The state transition diagram of the underlying Markov process of this SCK model shows that there are 10 distinct states where each state has at most two possible transitions (Fig. 2A). By decomposing the original model, three identical sub-models as $n_0 = 3$ are generated. Each of the sub-models has three states where a single molecule of S_1 initially present transitions sequentially via the two reactions (Fig. 2B). Not only does each of these sub-models have smaller state space, but it also has fewer possible transitions than the original model. Another notable difference between the original model and the sub-models is that while the sequence of reaction events in the original model are stochastic (Fig. 2A), the transitions in each of the sub-models are found to be deterministic; each sub-model fires exactly two reaction events and the sequence of that is always R_1 first and then R_2 second (Fig. 2B).

The algorithm of the DoSSA is shown in Algorithm 2. The advantage of the DoSSA over the SSA is two-folds. Firstly, the average number of reaction events for a given time window is n_0 times smaller in the simulation of a sub-model than

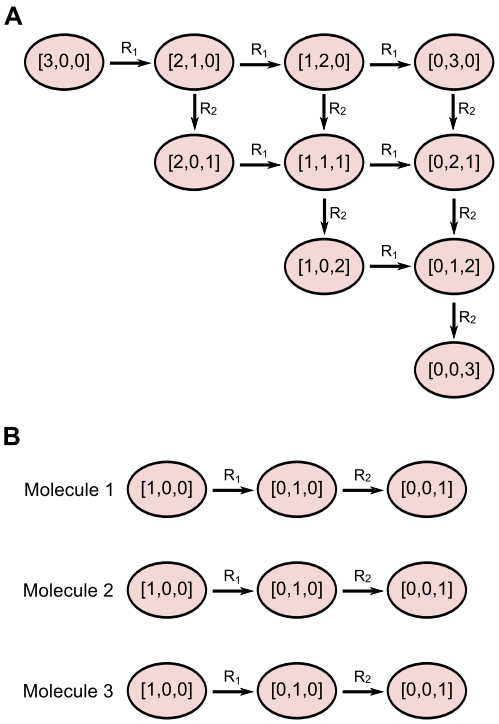

Figure 2: Markov chain state transition diagrams for $\mathcal{M}_e$. (A) The original SCK model. (B) The three sub-models decomposed from the original model.

that of the original model. Let E_t be the average number of reaction events in the time window $[0, t]$ in the original model, and E_t^s be the average number of reaction events in the time window $[0, t]$ in a sub-model. Then, since the process description of $\mathbf{X}(t)$ is identical to that of the sum of $\mathbf{X}^{(\mathbf{s})}(t)$ for all $t \geq 0$, we have $E_t = n_0 E_t^s$. This point is also illustrated in Fig. 2. Since state $[0, 0, 3]$ is the absorbing state in the original model, there are exactly six reaction events in the time windows $[0, \infty)$ from state $[3, 0, 0]$, while since $[0, 0, 1]$ is the absorbing state in each of the three sub-models, there are exactly two reaction events in the time windows $[0, \infty)$ from state $[1, 0, 0]$, which is three-times smaller than the original model. This feature alone can be translated roughly into n_0 times speedup in the simulation of each $\mathcal{M}^{(k)}$ compared with that of $\mathcal{M}$. Theoretically, thus, this approach is able to decompose and parallelize single simulations without synchronization and communication overhead.

Algorithm 2 The DoSSA with u simulation runs

1: Create a model $\mathcal{M}^{(s)} = \langle t, \mathbf{X}^{(\mathbf{s})}, \mathbf{a}, \mathbf{V}, (1, 0, \ldots, 0) \rangle$.
2: Simulate $\mathcal{M}^{(s)}$ with SSA in parallel $u \times n_0$ times
3: Combine the results for statistical analysis

Secondly, the evaluation of propensity functions in each state can be substantially accelerated. This is because, in each iteration, the SSA of each sub-model tends to have a much fewer number of species with a non-zero molecular count than the original model, which in turn, makes the number of active reactions (i.e., reactions with a non-zero reaction rate) in each sub-model smaller than that of the original model. For example, in Fig. 2, while some states of $\mathcal{M}_e$ have two active reactions (i.e., states: $[2, 1, 0]$; $[1, 2, 0]$; and $[1, 1, 1]$), sub-models only have states with single active reactions. That is, each transition event in Fig. 2B is deterministic in a sense that next reactions are predetermined. Since the number of active reactions are fewer in the simulation of sub-models, computational time for the evaluation of propensity functions in sub-models can also be reduced. Furthermore, when a reactant species has at most one molecule, which is often the case in the DoSSA, the evaluation of propensity functions becomes simplified; when R_j is active, rate constant c_j becomes the reaction rate. For example, the reaction rates of R_1 and R_2 are simply c_1 and c_2 when they are active in Fig. 2B. Thus, in these cases, speedup from the DoSSA becomes further pronounced.

5. APPLICATION

As a case study, we have applied the DoSSA to statistical analysis of a stochastic nonviral gene transfer model which is based on the mass-action kinetic model that was previously investigated by Varga *et al.* [37]. The Varga *et al.* model was constructed to study the contribution of each barrier process to the efficiency of the overall gene delivery system for transfection of C3A human hepatoblastoma cells with GFP-encoding plasmid by means of the Lipofectamine vector *in vitro* [37]. Using this model, they were particularly interested in predicting the effect of the length of a polymer in a polyplex delivery vector.

The Varga *et al.* model has 16 molecular species including the initial extracellular vector-plasmid complex (Table 2), and has the following 14 reactions:

$$
\begin{aligned}
R_1 &: S_1 \xrightarrow{c_1} S_2, & R_8 &: S_8 \xrightarrow{c_8} S_{10},\\
R_2 &: S_2 \xrightarrow{c_2} S_3, & R_9 &: S_9 \xrightarrow{c_9} S_{11},\\
R_3 &: S_3 \xrightarrow{c_3} S_4 + S_5, & R_{10} &: S_{10} \xrightarrow{c_{10}} S_{13},\\
R_4 &: S_3 \xrightarrow{c_4} S_6, & R_{11} &: S_{11} \xrightarrow{c_{11}} S_{12},\\
R_5 &: S_5 \xrightarrow{c_5} S_7, & R_{12} &: S_{13} \xrightarrow{c_{12}} S_{14},\\
R_6 &: S_5 \xrightarrow{c_6} S_8, & R_{13} &: S_{12} \xrightarrow{c_{13}} S_{14} + S_{16},\\
R_7 &: S_6 \xrightarrow{c_7} S_9, & R_{14} &: S_{14} \xrightarrow{c_{14}} S_{14} + S_{15}
\end{aligned}
$$

Further information on this gene delivery model is shown in Appendix B.

The overall gene delivery process of this model is as follows. Vector-plasmid complexes are initially in extracellular space, and binding to the cell surface followed by internalization leads to endosomal uptakes (R_1). The complex can then escape from endosome to cytoplasm (R_2). At this point the vector-plasmid complex can dissociate into a vector and a plasmid (R_3) or it can bind to cytoplasmic factors, which are required to facilitate nuclear-pore complexes (R_4). A plasmid dissociated from a vector can degrade in the cytoplasm (R_5), or it can bind to the cytoplasmic factors (R_6).

Both a vector-plasmid complex and a plasmid bound the cytoplasmic factors can be associated with nuclear-pore complexes (R_7 and R_8), which in turn allows them to enter the nucleus (R_9 and R_{10}). Once in the nucleus, both a vector-plasmid complex and a plasmid can dissociate from nuclear-pore complexes and cytoplasmic factors (R_{11} and R_{12}). The vector-plasmid complex can then dissociate into a vector and a plasmid inside the nucleus (R_{13}). Finally, within the nucleus the GFP gene in the plasmid can be expressed, leading to GFP production (R_{14}).

Using this model, we are interested in analyzing the effect of the complex unpacking rate (i.e., k_{unpack}: the parameter for the dissociation of a vector-plasmid complex) on the distribution of the GFP level at various time snapshots. The values of the other parameters are based on the original Varga *et al.* model (Table 3). Each of our stochastic simulation begins with $n_0 = 90,000$, which corresponds to a fit for the total number of plasmids delivered to a cell as explained in [37]. We simulated the SCK version of the Varga *et al.* model 10,000 times and analyzed the time evolution of the distribution of the GFP level. Our results are in agreement with those of [37] in that by normalizing the protein production based on the mean value with the unpacking parameter (i.e., c_3 and c_{13}) being 10^{-2}min^{-1}, normalized protein production levels match those of [37] (Fig. 3). Furthermore, our results are able to show how the distribution of the GFP level changes over time — something that is not possible with the mass-action kinetic framework (Fig. 3).

To quantify the speedup achieved by the DoSSA, we first measured the average speedup achieved by the DoSSA over the SSA for a single simulation run for various values of unpacking parameter with parallel computing with 20 cores (Fig. 4). Here, we implemented both the SSA and the DoSSA of this gene delivery model in C to achieve high computational efficiency. We chose to use the direct method as the implementation of the SSA because the model has a relatively small number of reactions. In such small reaction systems, the direct method usually performs as well as, if not better than, the recently improved versions. For example, in [5], it was shown that when a system was small (reaction size = 68), the direct method could be much faster than the next reaction methods.

In this measure, we simply took the ratio of an average computational time for a single simulation of the Varga *et al.* model based on the DoSSA that is run over 20 cores and an average computation time for a single SSA run of the Varga *et al.* model. This runtime comparison shows that, at the single simulation run level, the speedup factor achieved by the DoSSA is consistently greater than the number of cores (i.e., 20) for various values of k_{unpack}. In this specific initial condition, the DoSSA is able to decompose the Varga *et al.* model into 90,000 sub-models. This high-level of decomposition here ensures that each of the 20 cores is effectively utilized for simulations of each sub-model. Indeed, each of the sub-model simulation runs costs smaller than 1/90,000 of the simulation of the original model, allowing the speedup factor of more than 30 to be realized in parallel computing with 20 cores (Fig. 4).

Interestingly, in the stochastic simulation of the Varga *et al.* model, the speedup factor of the DoSSA is an increasing function of k_{unpack}; as the value of k_{unpack} is increased, so is the speedup factor of the DoSSA. This positive relation can be explained by understanding the effect of dissociation reactions of vector-plasmid complexes in the Varga *et al.* model. That is, while the dissociation of a vector-plasmid complex in the nucleus is necessary for the gene in a plasmid to be expressed, the same dissociation process in the cytoplasm may lead to cytoplasmic degradation of the plasmid. Thus, higher value of the unpacking parameter can be translated to a higher level of cytoplasmic degradation of plasmids. Since simulation of each sub-model of the Varga *et al.* model samples the gene delivery process of a single vector-plasmid complex molecule, a higher chance of degradation of plasmids means that more simulation runs of sub-models terminate fast, which can, in turn, accelerate the overall simulation process of the DoSSA.

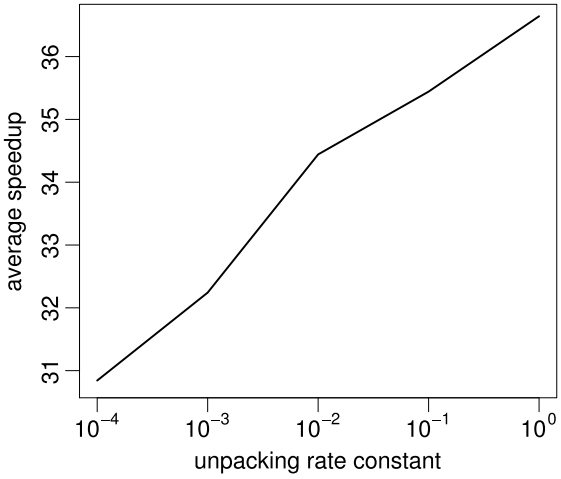

Figure 4: The average speedup factor of the DoSSA over the SSA for a single simulation run for various unpacking rate values with 20 cores.

Furthermore, the DoSSA substantially accelerates the next reaction selection and reaction update schemes. In the SSA of the Varga *et al.* model, the worst case of the reaction selection is that a next reaction event is chosen from 14 active reactions. In the DoSSA, however, the worst case is when a next reaction is chosen from two active reactions (i.e., when $X_3 = 1$ or $X_5 = 1$), and most of the time, next reaction is predetermined, making substantial speedup in the next reaction scheme. Also, since all of the species except for the GFP (i.e., S_{15}) have at most one molecule throughout the simulation of sub-models, evaluation of the propensity function is not necessary as the reaction rate of an active reaction is simply its rate constant. Taken together, these features of the DoSSA allows for substantial speedup at the single simulation level.

However, while important, the runtime comparison at the single simulation level may not reflect to the measure of computation time involved in statistical analysis with reasonable level of confidence as this often demands a large number of

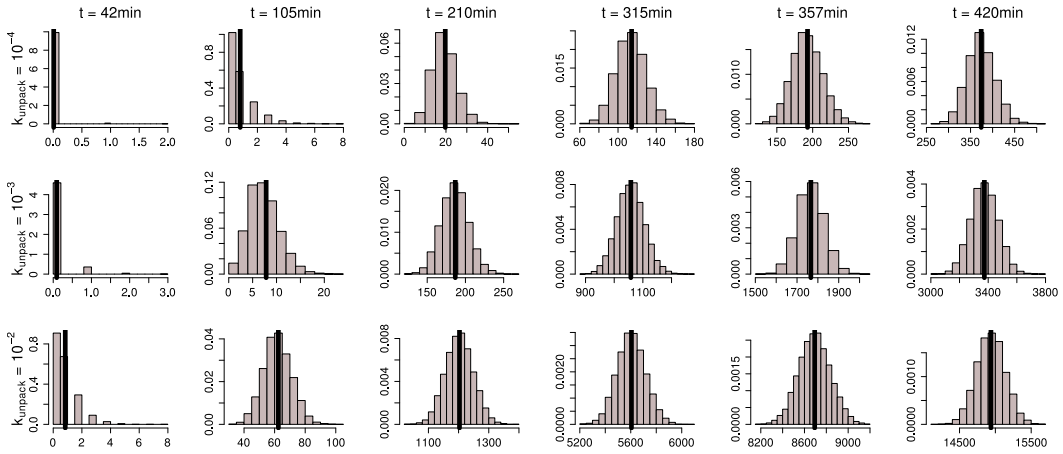

Figure 3: Time evolution of protein distribution for various unpacking rate constants. The black vertical line in each graph indicates the mean value of protein level. Here, snapshots of the protein distribution at six different time points are shown for the nonviral gene transfer model with three distinct values of the unpacking rate constant: 10^{-4}min^{-1}; 10^{-3}min^{-1}; and 10^{-2}min^{-1}. The values of the other parameters are specified in Table 3.

simulation runs. Thus, we next compared the computational time of this statistical analysis based on 10,000 runs of the SSA and the DoSSA. We measured the total simulation time based on a different number of cores by running on a cluster of computers. Our performance comparison shows that the DoSSA is consistently faster than the SSA independent of the number of cores, where the speedup factor of the DoSSA over the SSA is at least 1.5 (Table 1). An decrease in the

Table 1: Comparison of computational time between the standard SSA and the DoSSA for 10,000 runs.

			Value of k_{unpack} (min^{-1})		
			10^{-4}	10^{-3}	10^{-2}
1 core	SSA	Run time[a]	14.6167	15.7333	19.4167
	DoSSA	Run time[a]	9.2333	9.8000	10.7167
		Speedup	**1.58**	**1.61**	**1.81**
10 cores	SSA	Run time[a]	1.3500	1.5167	1.7833
	DoSSA	Run time[a]	0.8333	0.9167	1.0333
		Speedup	**1.62**	**1.65**	**1.73**
20 cores	SSA	Run time[a]	0.6667	0.7500	0.9167
	DoSSA	Run time[a]	0.4333	0.4667	0.5333
		Speedup	**1.54**	**1.61**	**1.72**

[a]the unit is min/10^4runs.

speedup factor of the DoSSA here from the single simulation level is due to the fact that, like the DoSSA, the SSA can efficiently utilize 20 cores for the total of 10,000 runs. Nevertheless, the acceleration factors such as the next reaction selection and the reaction update still allow statistical analysis based on the DoSSA to be at least 1.5 times more computationally efficient than that based on the SSA in this study.

6. DISCUSSION

Nonviral gene delivery approaches have gained significant attention in gene therapy mainly because of their safety advantages over virus-mediated approaches. Although significant progress has been made in nonviral gene delivery systems, their transfection efficacy are still much less compared with those of viral counterparts. Thus, development of nonviral vectors with high transfection efficacy continues to be an important milestone for the success of gene therapy. The design process of synthetic vectors with experimental means alone, however, demands substantial time and efforts as it needs to consider multiple intracellular barrier processes for optimization of the overall gene transfer efficacy. Hence, computational modeling and analysis can be a powerful tool to aid wet-lab experiments in characterizing and predicting how gene vectors behave inside cells as previously emphasized by several studies (e.g., [37, 4, 38, 40]). However, computational analysis such as sensitivity analysis to identify rate-limiting barriers and to optimize parameters for the overall transfection efficacy can require substantial computational costs. Moreover, this high computational demand can further be increased by considering the intrinsic fluctuations of gene delivery processes. Therefore, efficient computational approaches to analyze gene delivery systems are essential to solving this biomedically important problem.

Parallel computing can play an important role in efficiently solving such computationally demanding problems. With multicore processors being the norm in personal computers, cluster computers becoming more accessible, and GPU computing becoming more popular, solving complex computational problems with the use of parallel computing is becoming not just essential but also inevitable. For analysis of gene delivery systems, thus, it is crucial to develop efficient methods for parallel computing to cope with the high computational requirement. In this paper, we have proposed

a novel parallelization method (DoSSA), that is capable of decomposing and parallelizing single simulations into a set of sub-simulations, each of which is able to generate a sample trajectory of a single vector-plasmid complex molecule.

Previous studies on paralleling single simulation runs (e.g., [7, 20]) focus on approximate reaction-diffusion models via space discretization to utilize the SSA for generating sample trajectories. These methods partitions the cell volume into many subcompartments and parallelizes at the single simulation level to take advantage of resulting independency among reactions in different compartments. However, these methods inevitably come with computational overhead from synchronization and communication among neighboring compartments, whose effects become significant especially when there are fast diffusion rates. Thus, while these approaches can significantly speed up single simulation runs of complex spatial-temporal models via parallelization, they are less efficient than a simpler method to parallelize generation of each sample trajectory when many simulation runs are required for statistical analysis. Our new approach based on decomposition of a SCK model into independent submodels, however, does not exhibit this computational drawback, and it can efficiently parallelize single simulations without synchronization and communication overhead. Furthermore, the deep decomposition of the SSA at the single-molecular level also enables acceleration in the next reaction selection and reaction update schemes, making single simulations faster even in the non-parallel setting. To the best of our knowledge, no previous method is able to parallelize simulation of a SCK model at the single simulation level as efficiently as the DoSSA is.

Our case study based on a previously studied gene delivery system has demonstrated that the average single simulation run of the DoSSA is at least 30 times faster than that of the SSA by utilizing 20 CPU cores. This is mainly due to the fact that the DoSSA is able to optimally utilize multiple cores at the single simulation level. Furthermore, it has shown that the computational time of the statistical analysis of a gene delivery model with 10,000 simulation runs is at least 50% more efficient with the DoSSA than with the SSA. Considering substantially high computational requirements involved in sensitivity analysis and parameter optimization of nonviral gene delivery models, such a speedup without loss of accuracy may be significant.

The DoSSA is developed especially to target effective parallelizations of nonviral gene delivery models by using biological insights of the structure and property of such models. That is, first-order reactions and sparseness property are the key to accelerate simulations of the DoSSA. This means that the DoSSA cannot be applied to general SCK models. However, considering the biomedical importance of the design of nonviral vectors with high transfection efficacy, we do not believe this tailored approach is such a limitation. In fact, this work highlights the usefulness of such specialized methods to solve important problems when a general solution is not feasible.

We are currently investigating the relationship between the model density and performance of our method. Also, we are exploring the possibility to apply the DoSSA to other interesting biological systems including pulse-response systems [36, 31]. Finally, we are considering a means to extend this parallel approach and further modify our method into a bounded approximation algorithm for SCK models with higher-order reactions.

Acknowledgments

HK is supported by Lane fellowship through the Ray and Stephanie Lane Center for Computational Biology at Carnegie Mellon University. XG is supported by a grant from King Abdullah University of Science and Technology.

7. REFERENCES

[1] M. Al-Dosari and X. Gao. Nonviral gene delivery: principle, limitations, and recent progress. *The AAPS journal*, 11(4):671–681, 2009.

[2] P. Ballarini, M. Forlin, T. Mazza, and D. Prandi. Efficient parallel statistical model checking of biochemical networks. *Electronic Proceedings in Theoretical Computer Science*, 14.

[3] P. Ballarini, R. Guido, T. Mazza, and D. Prandi. Taming the complexity of biological pathways through parallel computing. *Briefings in bioinformatics*, 10(3):278, 2009.

[4] G. Banks, R. Roselli, R. Chen, and T. Giorgio. A model for the analysis of nonviral gene therapy. *Gene therapy*, 10(20):1766–1775, 2003.

[5] Y. Cao, H. Li, and L. Petzold. Efficient formulation of the stochastic simulation algorithm for chemically reacting system. *Journal of Chemical Physics*, 121:4059–4067, 2004.

[6] J. Couzin and J. Kaiser. As gelsinger case ends, gene therapy suffers another blow. *Science*, 307(5712):1028, 2005.

[7] L. Dematté and T. Mazza. On parallel stochastic simulation of diffusive systems. In *Computational Methods in Systems Biology*, pages 191–210. Springer, 2008.

[8] M. Edelstein, M. Abedi, and J. Wixon. Gene therapy clinical trials worldwide to 2007—an update. *The Journal of Gene Medicine*, 9(10):833–842, 2007.

[9] X. Gao, K. Kim, and D. Liu. Nonviral gene delivery: what we know and what is next. *The AAPS journal*, 9(1):92–104, 2007.

[10] H. Gaspar, K. Parsley, S. Howe, D. King, K. Gilmour, J. Sinclair, G. Brouns, M. Schmidt, C. Von Kalle, T. Barington, et al. Gene therapy of X-linked severe combined immunodeficiency by use of a pseudotyped gammaretroviral vector. *The Lancet*, 364(9452):2181–2187, 2004.

[11] M. Gibson and J. Bruck. Efficient exact stochastic simulation of chemical systems with many species and many channels. *J. Phys. Chem.*, A 104:1876–1889, 2000.

[12] D. T. Gillespie. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *Journal of Computational Physics*, 22:403–434, 1976.

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

[14] D. T. Gillespie. *Markov Processes An Introduction for Physical Scientists*. Academic Press, Inc., 1992.

[15] D. T. Gillespie. A rigorous derivation of the chemical master equation. *Physica A*, 188:404–425, 1992.

[16] D. T. Gillespie. *Handbook of Materials Modeling*, chapter 5.11, pages 1735–1752. Springer, 2005.

[17] D. Glover, H. Lipps, and D. Jans. Towards safe, non-viral therapeutic gene expression in humans. *Nature Reviews Genetics*, 6(4):299–310, 2005.

[18] F. Horn and R. Jackson. General mass action kinetics. *Archive for Rational Mechanics and Analysis*, 47(2):81–116, 1972.

[19] M. Jeschke and R. Ewald. Large-scale design space exploration of ssa. In *Computational Methods in Systems Biology*, pages 211–230. Springer, 2008.

[20] M. Jeschke, A. Park, R. Ewald, R. Fujimoto, and A. Uhrmacher. Parallel and distributed spatial simulation of chemical reactions. In *22nd Workshop on Principles of Advanced and Distributed Simulation*, pages 51–59. IEEE, 2008.

[21] J. Keener and J. Sneyd. *Mathematical Physiology*. Springer, 1998.

[22] H. Kuwahara, C. Myers, M. Samoilov, N. Barker, and A. Arkin. Automated abstraction methodology for genetic regulatory networks. *Transactions on Computational Systems Biology VI*, pages 150–175, 2006.

[23] T. Ledley and F. Ledley. Multicompartment, numerical model of cellular events in the pharmacokinetics of gene therapies. *Human Gene Therapy*, 5(6):679–691, 1994.

[24] H. Li and L. Petzold. Efficient parallelization of the stochastic simulation algorithm for chemically reacting systems on the graphics processing unit. *International Journal of High Performance Computing Applications*, 24(2):107, 2010.

[25] F. Mavilio. Gene therapy: back on track? *EMBO reports*, 11(2):75, 2010.

[26] D. A. McQuarrie. Stochastic approach to chemical kinetics. *Journal of Applied Probability*, 4:413–478, 1967.

[27] B. Munsky and M. Khammash. The finite state projection algorithm for the solution of the chemical master equation. *Journal of Chemical Physics*, 124, 2006.

[28] Z. Parra-Guillén, G. González-Aseguinolaza, P. Berraondo, and I. Trocóniz. Gene therapy: a pharmacokinetic/pharmacodynamic modelling overview. *Pharmaceutical research*, pages 1–11.

[29] R. Ramaswamy and I. F. Sbalzarini. A partial-propensity variant of the composition-rejection stochastic simulation algorithm for chemical reaction networks. *Journal of Chemical Physics*, 132(4):044102, 2010.

[30] S. Rosenberg, P. Aebersold, K. Cornetta, A. Kasid, R. Morgan, R. Moen, E. Karson, M. Lotze, J. Yang, S. Topalian, et al. Gene transfer into humans – immunotherapy of patients with advanced melanoma, using tumor-infiltrating lymphocytes modified by retroviral gene transduction. *New England Journal of Medicine*, 323(9):570–578, 1990.

[31] J. Ross. From the determination of complex reaction mechanisms to systems biology. *Annu. Rev. Biochem.*, 77:479–494, 2008.

[32] A. Slepoy, A. Thompson, and S. Plimpton. A constant-time kinetic monte carlo algorithm for simulation of large biochemical reaction networks. *The Journal of chemical physics*, 128:205101, 2008.

[33] C. Thomas, A. Ehrhardt, and M. Kay. Progress and problems with the use of viral vectors for gene therapy. *Nature Reviews Genetics*, 4(5):346–358, 2003.

[34] T. Tian and K. Burrage. Parallel implementation of stochastic simulation for large-scale cellular processes. In *Proceedings of the Eighth International Conference on High-Performance Computing in Asia-Pacific Region*, page 621. IEEE Computer Society, 2005.

[35] N. G. van Kampen. *Stochastic Processes in Physics and Chemistry*. Elsevier, 1992.

[36] W. Vance, A. Arkin, and J. Ross. Determination of causal connectivities of species in reaction networks. *Proceedings of the national academy of sciences of the United States of America*, 99(9):5816, 2002.

[37] C. Varga, K. Hong, and D. Lauffenburger. Quantitative analysis of synthetic gene delivery vector design properties. *Molecular Therapy*, 4(5):438–446, 2001.

[38] C. Varga, N. Tedford, M. Thomas, A. Klibanov, L. Griffith, and D. Lauffenburger. Quantitative comparison of polyethylenimine formulations and adenoviral vectors in terms of intracellular gene delivery processes. *Gene therapy*, 12(13):1023–1032, 2005.

[39] W. Walther and U. Stein. Viral vectors for gene transfer: a review of their use in the treatment of human diseases. *Drugs*, 60(2):249–271, 2000.

[40] J. Zhou, J. W. Yockman, S. W. Kim, and S. E. Kern. Intracellular kinetics of non-viral gene delivery using polyethylenimine carriers. *Pharmaceutical Research*, 24:1079–1087, 2007.

[41] Y. Zhou, J. Liepe, X. Sheng, M. Stumpf, and C. Barnes. GPU accelerated biochemical network simulation. *Bioinformatics*, 27(6):874, 2011.

APPENDIX

A. DETAIL OF PROOF OF LEMMA 1

Our proof is by induction on t. Since $\mathbf{x_0} = \mathbf{x_0^{(1)}} + \mathbf{x_0^{(2)}}$, $\mathbf{X}(0)$ is a sure variable that is equal to $\mathbf{X^{(1)}}(0) + \mathbf{X^{(2)}}(0)$, initially we have $P(\mathbf{x}, 0 \mid \mathbf{x_0})$ equal to $P_2(\mathbf{x}, 0 \mid \mathbf{x_0^{(1)}}, \mathbf{x_0^{(2)}})$.

Here, since each reaction is assumed to follow first-order reaction kinetics, let $a_j(\mathbf{X}) = c_j X_{I_j}$ where I_j is the index of the species that is used as the reactant of R_j.

Now, we assume that

$$P(\mathbf{x}, t \mid \mathbf{x_0}) = P_2(\mathbf{x}, t \mid \mathbf{x_0^{(1)}}, \mathbf{x_0^{(2)}})$$

to show that

$$P(\mathbf{x}, t + dt \mid \mathbf{x_0}) = P_2(\mathbf{x}, t + dt \mid \mathbf{x_0^{(1)}}, \mathbf{x_0^{(2)}}).$$

From Eq. 2, $P(\mathbf{x}, t + dt \mid \mathbf{x_0})$ with first-order reactions can be expressed as:

$$P(\mathbf{x}, t + dt \mid \mathbf{x_0}) = P(\mathbf{x}, t \mid \mathbf{x_0}) \left[1 - \sum_{j=1}^{M} c_j x_{I_j} dt\right] + \sum_{j=1}^{M} \left[P(\mathbf{x} - \mathbf{v_j}, t \mid \mathbf{x_0}) c_j (x_{I_j} - v_{jI_j}) dt\right]. \tag{6}$$

Now, we express $P_2(\mathbf{x}, t+dt \mid \mathbf{x_0^{(1)}}, \mathbf{x_0^{(2)}})$ by letting $\mathbf{X^{(1)}}(t) = \mathbf{x^{(1)}}$ and $\mathbf{X^{(2)}}(t) = \mathbf{x^{(2)}}$. Then, when $\mathbf{X^{(1)}}(t) + \mathbf{X^{(2)}}(t) = \mathbf{x}$, the probability that $\mathbf{X^{(1)}} + \mathbf{X^{(2)}}$ moves away from $\mathbf{x}$ in the next infinitesimal time dt is:

$$1 - \sum_{j=1}^{M} c_j x_{I_j}^{(1)} dt - \sum_{j=1}^{M} c_j x_{I_j}^{(2)} dt = 1 - \sum_{j=1}^{M} c_j x_{I_j} dt \tag{7}$$

because any reaction event from $\mathcal{M}^{(1)}$ or $\mathcal{M}^{(2)}$ can make this transition.

There are two mutually exclusive cases when $\mathbf{X^{(1)}}(t) + \mathbf{X^{(2)}}(t)$ moves from $\mathbf{x} - \mathbf{v_j}$ to $\mathbf{x}$ by reaction R_j. The first case is when the transition is caused by a reaction event from $\mathcal{M}^{(1)}$, that is, when $\mathbf{X^{(1)}} = \mathbf{x^{(1)}} - \mathbf{v_j}$ and $\mathbf{X^{(2)}} = \mathbf{x^{(2)}}$ where the probability of $\mathbf{X^{(1)}}(t) + \mathbf{X^{(2)}}(t)$ moves from $\mathbf{x} - \mathbf{v_j}$ to $\mathbf{x}$ in the next infinitesimal time dt is:

$$c_j(x_{I_j}^{(1)} - v_{jI_j})dt + c_j x_{I_j}^{(2)} dt = c_j(x_{I_j} - v_{jI_j})dt. \tag{8}$$

The second case is when the transition is caused by a reaction event from $\mathcal{M}^{(2)}$, that is, when $\mathbf{X^{(1)}} = \mathbf{x^{(1)}}$ and $\mathbf{X^{(2)}} = \mathbf{x^{(2)}} - \mathbf{v_j}$ where the probability of $\mathbf{X^{(1)}}(t) + \mathbf{X^{(2)}}(t)$ moves from $\mathbf{x} - \mathbf{v_j}$ to $\mathbf{x}$ in the next infinitesimal time dt is:

$$c_j(x_{I_j}^{(2)} - v_{jI_j})dt + c_j x_{I_j}^{(1)} dt = c_j(x_{I_j} - v_{jI_j})dt. \tag{9}$$

By letting $P_{2,2}(\mathbf{x^{(1)}}, \mathbf{x^{(2)}}, t \mid \mathbf{x_0^{(1)}}, \mathbf{x_0^{(2)}})$ be the probability that $\mathbf{X^{(1)}}(t) = \mathbf{x^{(1)}}$ and $\mathbf{X^{(2)}}(t) = \mathbf{x^{(2)}}$ given that initially $\mathbf{X^{(1)}}(0) = \mathbf{x_0^{(1)}}$ and $\mathbf{X^{(2)}}(0) = \mathbf{x_0^{(2)}}$, we can express $P_2(\mathbf{x}, t + dt \mid \mathbf{x_0^{(1)}}, \mathbf{x_0^{(2)}})$ as:

$$\begin{aligned} P_2(\mathbf{x}, t + dt \mid \mathbf{x_0^{(1)}}, \mathbf{x_0^{(2)}}) &= P_2(\mathbf{x}, t \mid \mathbf{x_0^{(1)}}, \mathbf{x_0^{(2)}}) \left[1 - \sum_{j=1}^{M} c_j x_{I_j} dt\right] \\ &+ \sum_{j=1}^{M} \left[P_{2,2}(\mathbf{x^{(1)}} - \mathbf{v_j}, \mathbf{x^{(2)}}, t \mid \mathbf{x_0^{(1)}}, \mathbf{x_0^{(2)}}) c_j (x_{I_j} - v_{jI_j}) dt\right] \\ &+ \sum_{j=1}^{M} \left[P_{2,2}(\mathbf{x^{(1)}}, \mathbf{x^{(2)}} - \mathbf{v_j}, t \mid \mathbf{x_0^{(1)}}, \mathbf{x_0^{(2)}}) c_j (x_{I_j} - v_{jI_j}) dt\right]. \end{aligned} \tag{10}$$

Here, we can express $P_2(\mathbf{x} - \mathbf{v_j}, t \mid \mathbf{x_0^{(1)}}, \mathbf{x_0^{(2)}})$ as the sum of the two mutually exclusive cases as follows:

$$\begin{aligned} P_2(\mathbf{x} - \mathbf{v_j}, t \mid \mathbf{x_0^{(1)}}, \mathbf{x_0^{(2)}}) &= P_{2,2}(\mathbf{x^{(1)}} - \mathbf{v_j}, \mathbf{x^{(2)}}, t \mid \mathbf{x_0^{(1)}}, \mathbf{x_0^{(2)}}) \\ &+ P_{2,2}(\mathbf{x^{(1)}}, \mathbf{x^{(2)}} - \mathbf{v_j}, t \mid \mathbf{x_0^{(1)}}, \mathbf{x_0^{(2)}}). \end{aligned}$$

Thus, we can simplify Eq. 10 as follows:

$$\begin{aligned} P_2(\mathbf{x}, t + dt \mid \mathbf{x_0^{(1)}}, \mathbf{x_0^{(2)}}) &= P_2(\mathbf{x}, t \mid \mathbf{x_0^{(1)}}, \mathbf{x_0^{(2)}}) \left[1 - \sum_{j=1}^{M} c_j x_{I_j} dt\right] \\ &+ \sum_{j=1}^{M} \left[P_2(\mathbf{x} - \mathbf{v_j}, t \mid \mathbf{x_0^{(1)}}, \mathbf{x_0^{(2)}}) c_j (x_{I_j} - v_{jI_j}) dt\right]. \end{aligned} \tag{11}$$

Therefore, it follows from induction that

$$P(\mathbf{x}, t \mid \mathbf{x_0}) = P_2(\mathbf{x}, t \mid \mathbf{x_0^{(1)}}, \mathbf{x_0^{(2)}}) \quad \text{for all } t \geq 0.$$

B. FURTHER MODEL DESCRIPTION

Table 2: Species in the gene transfer model.

Species	**Compartment**	**Description [37]**
S_1	extrarcellular	initial complexes
S_2	cytoplasm	endosomal complexes
S_3	cytoplasm	cytoplasmic complexes
S_4	cytoplasm	cytoplasmic vectors
S_5	cytoplasm	cytoplasmic plasmids
S_6	cytoplasm	cytoplasmic import protein bound complexes
S_7	cytoplasm	cytoplasmic degraded plasmids
S_8	cytoplasm	cytoplasmic import protein bound plasmids
S_9	cytoplasm	nuclear pore associated import protein bound complexes
S_{10}	cytoplasm	nuclear pore associated import protein bound plasmids
S_{11}	nuclear	nuclear import protein bound complexes
S_{12}	nuclear	nuclear complexes
S_{13}	nuclear	nuclear import protein bound plasmids
S_{14}	nuclear	nuclear plasmids
S_{15}	intracellular	protein
S_{16}	nuclear	nuclear vectors

Table 3: Parameters of the gene transfer model.

Parameter	**Name in ref. [37]**	**Value** (min^{-1})
c_1	$k_{internalization}$	1^a
c_2	k_{escape}	1×10^{-2}
c_3	k_{unpack}	ranged
c_4	k_{bind}	2×10^{-2}
c_5	$k_{degradation}$	5×10^{-3}
c_6	k_{bind}	2×10^{-3}
c_7	k_{NPC}	1×10^{3}
c_8	k_{NPC}	1×10^{3}
c_9	k_{in}	3×10^{-3}
c_{10}	k_{in}	3×10^{-3}
c_{11}	$k_{dissociation}$	1×10^{3}
c_{12}	$k_{dissociation}$	1×10^{3}
c_{13}	k_{unpack}	ranged
c_{14}	$k_{protein}$	1×10^{-2}

[a]The value of $k_{internalization}$ is derived from Eq. 1 in [37].