ELSEVIER

Computational systems biology in drug discovery and development: methods and applications

Wayne Materi[2] and David S. Wishart[1,2]

[1] Departments of Biological Sciences & Computing Science, University of Alberta, Canada, T6G 2E8
[2] National Research Council, National Institute for Nanotechnology (NINT) Edmonton, Alberta, Canada T6G 2E8

Computational systems biology is an emerging field in biological simulation that attempts to model or simulate intra- and intercellular events using data gathered from genomic, proteomic or metabolomic experiments. The need to model complex temporal and spatiotemporal processes at many different scales has led to the emergence of numerous techniques, including systems of differential equations, Petri nets, cellular automata simulators, agent-based models and pi calculus. This review provides a brief summary and an assessment of most of these approaches. It also provides examples of how these methods are being used to facilitate drug discovery and development.

Introduction

Systems biology is a newly emerging, multi-disciplinary field that studies the mechanisms underlying complex biological processes by treating these processes as integrated systems of many interacting components. In practice, systems biology involves collecting large sets of experimental genomic, proteomic or metabolomic data, generating predictive models of those data using computers, and then assessing or correcting those computer models by comparing the predicted data with newly derived experimental data [1,2]. A central tenant to systems biology is the concept of computer modeling or computer simulation. Indeed, this emphasis on computing has given rise to a new discipline called 'computational systems biology' [3]. The interest that drug companies and pharmaceutical researchers have in computational systems biology comes from the hope that these emerging simulation tools might make drug discovery and drug testing better, faster and cheaper [4–6].

This drive towards computational simulation is moving biology (and pharmaceutical research) away from being purely a descriptive science to being more of a predictive science [2,6]. However, as we have learned from other predictive sciences such as physics and chemistry, modeling and simulation are most useful when they can: (i) produce useful predictions or extrapolations that match experimental results; (ii) permit data to be generated that is beyond present-day experimental capabilities; (iii) enable experiments to be performed *in silico* to save time, cost or effort; (iv) yield non-intuitive insights into how a system or process works; (v) identify missing components, processes or functions in a system; (vi) enable complex processes to be better understood or visualized; and (vii) facilitate the consolidation of quantitative data about a given system or process.

If models or modeling systems that meet these criteria are developed, then it is not hard to see how computational systems biology could aid in the identification of novel drug targets or assist with the assessment or dosing of drug compounds. For example, using advanced modeling approaches, novel drug targets might be discovered by identifying the key or unanticipated control points in a disease-related pathway. Similarly, drug toxicity could, one day, be assessed rapidly and inexpensively by modeling organ specific metabolic stress responses. Furthermore, drug distribution and metabolism could be modeled through multi-organ, multi-tissue pharmacokinetic (PK) methods, and drug dosing regimes could be determined by modeling tumor (or viral, or immune) responses to different drug concentrations or different dose frequencies [4–6]. Obviously, these kinds of simulations would require experimental inputs and, eventually, some kind of experimental verification. But the fact that they could assist with the generation of a hypothesis or reduce experimental effort would mean they have done their job.

The types of simulations just described actually encompass many different temporal and spatial scales, ranging from

Corresponding author: Wishart, D.S. (david.wishart@ualberta.ca)

1359-6446/06/$ - see front matter doi:10.1016/j.drudis.2007.02.013

TABLE 1

Summary of different methods used in computational systems biology

Method	Description	Advantages	Disadvantages
ODEs	● Series of reaction-rate equations solved using numerical methods ● Produces graphs or tables of reagent production and consumption	● Well understood formalism ● Deterministic ● Fast ● Mathematically robust	● Limited to temporal modeling ● Assumes high concentrations and uniform mixing ● Brittle
Stochastic differential equations	● Series of reaction rate equations solved using 'master equation' and random number generator ● Handled using Gillespie algorithm	● Enables modeling low concentrations and low copy numbers ● Fast, robust	● Does not explicitly model individual random interactions ● Limited to temporal modeling Non-deterministic
S-system formalism or power law equations	● Uses Taylor approximation to simplify non-linear ODEs ● Enables steady-state DEs to be transformed to easily solved linear equations	● Permits enormous simplification ● Fast ● Enables rapid parameter testing	● Limited to modeling temporal events and processes
PDE or molecular dynamics	● Expresses spatial and temporal dependence through partial derivatives ● Solved using numerical methods ● Produces numeric output of concentrations and *x,y,z* coordinates	● Well understood formalism ● Can be fast ● Mathematically robust ● Enables modeling of time-and space-dependent process	● Complicated ● Difficult to implement or generalize ● Cannot model state of discontinuous transitions ● Brittle
Petri nets Colored Petri nets Hybrid Petri nets Stochastic Petri nets Timed Petri nets	● Uses a weighted firing process to activate events from multiple connections that are used as inputs ● Mimics telephone switchboard or power-grid load handling	● Simple to implement Nonmathematical ● Enables both quantitative and qualitative models	● Limited to temporal modeling ● Often uses linear approximations for non-linear processes ● Less developed than ODEs
CA or DCA	● Models time and space processes on a grid ● Neighboring objects interact with each other through simple rules of Boolean logic ● Generates movies and numerical output	● Simple to implement ● Non mathematical ● Quantitative and qualitative models ● Enables modeling of time and space process	● Computationally costly for large numbers of objects ● Less developed than ODEs or PDEs ● No simple conversion from rate constants
ABM	● Objects treated as intelligent agents ● Agents follow predefined rules of motion, behavior and interaction ● Generates movies and numerical output	● Simple to implement ● Non mathematical ● Quantitative and qualitative models ● Enables modeling of time and space process	● Computationally costly for large numbers of objects ● Less developed than ODEs or PDEs ● No simple conversion from rate constants
Pi calculus	● A language for concurrent computational processes ● Pairs of processes interact by sending and receiving synchronized messages	● Enables simple description of concurrent processes ● Syntax can be structured as graphs	● Less developed than all other modeling schemas

TABLE 2

Ordinary and partial differential equation (ODE) representations of simple chemical reactions

Reaction	Reaction type	System of ODEs or PDEs
$A \xrightarrow{k_1} B$	Monomolecular conversion	$\frac{d[A]}{dt} = -k_1[A]$, $\frac{d[B]}{dt} = k_1[A]$
$A \underset{k_2}{\overset{k_1}{\rightleftarrows}} B$	Reversible conversion	$\frac{d[A]}{dt} = -k_1[A] + k_2[B]$, $\frac{d[B]}{dt} = -k_2[B] + k_1[A]$
$A + B \underset{k_2}{\overset{k_1}{\rightleftarrows}} C + D$	Bimolecular reversible conversion	$\frac{d[A]}{dt} = -k_1[A][B] + k_2[C][D]$, $\frac{d[B]}{dt} = -k_1[A][B] + k_2[C][D]$, $\frac{d[C]}{dt} = k_1[A][B] - k_2[C][D]$, $\frac{d[D]}{dt} = k_1[A][B] - k_2[C][D]$
$A + B \xrightarrow{k_1} C$	Production	$\frac{d[A]}{dt} = -k_1[A][B]$, $\frac{d[B]}{dt} = -k_1[A][B]$, $\frac{d[C]}{dt} = k_1[A][B]$
$A \xrightarrow{k_1} B + C$	Degradation	$\frac{d[A]}{dt} = -k_1[A]$, $\frac{d[B]}{dt} = k_1[A]$, $\frac{d[C]}{dt} = k_1[A]$
$A + B \xrightarrow{k_1} C$ with diffusion of molecules at rate D_1	Production	$\frac{\partial[A]}{\partial t} = D_1\nabla^2[A] - k_1[A][B]$, $\frac{\partial[B]}{\partial t} = D_1\nabla^2[B] - k_1[A][B]$, $\frac{\partial[C]}{\partial t} = -D_1\nabla^2[C] + k_1[A][B]$, Note: $\nabla^2[A] = \frac{\partial^2[A]}{\partial x^2} + \frac{\partial^2[A]}{\partial y^2} + \frac{\partial^2[A]}{\partial z^2}$

nanometers to meters, and milliseconds to days. Processes that occur across small dimensions (nm) or short times (ms) are often referred to as 'fine grain' models, whereas events that concern longer times (s) or larger (mm or cm) dimensions are called 'coarse grain' models. A fundamental challenge to computational systems biology is how to develop models or modeling tools that can deal with this wide range of granularity. In this review we will describe some of the novel modeling techniques that are being developed to permit both temporal and spatiotemporal modeling throughout such a wide range of scales. Among the simulation techniques to be discussed or described are: (i) systems of ordinary differential equations (ODEs); (ii) Petri nets; (iii) pi calculus; (iv) partial differential equations (PDEs); (v) cellular automata (CA) methods; (vi) agent-based systems; and (vii) hybrid approaches. Table 1 presents a short summary of these techniques in addition to their potential advantages and disadvantages. We will focus our discussion primarily on the most commonly used techniques, particularly ODEs, Petri nets, CA methods, and agent-based systems. Examples of how some of these techniques are being used in drug target discovery or for evaluating drug efficacy, toxicity, distribution or dosing will also be provided and assessed.

Systems simulations using ordinary differential equations

Although they seem complex, biological systems can be viewed as networks of chemical reactions that can be analyzed mathematically using ODEs. The use of ODEs in biological modeling is widespread, making it far and away the most common simulation approach used in computational systems biology [3,7]. However, basic ODE methods are not amenable to modeling discontinuous state-changes, stochasticity, transport processes, diffusion, compartmentalization, cell migration, or other common biological events. By contrast, PDEs can be used to model processes that have spatial as well as temporal dependencies. Despite these limitations, ODEs are powerful simulation tools, and many ODE solvers now exist. In addition, numerous recent extensions to basic ODE methods (including stochastic ordinary differential equations and explicitly defined compartment exchange rates) have permitted the modeling of more realistic biological processes.

Simple ODEs can have exact solutions; however, more complex ODEs do not have exact solutions and must be solved numerically. Table 2 lists several examples of reaction ODEs, some of which are simple and some of which require numerical analysis. Based on methods first derived by Newton and Gauss (available in standard textbooks), numerical solutions use linear approximations of smooth curves over small time intervals to compute the subsequent values of reactant concentrations. Various methods have been derived to improve the computational accuracy of these approximate methods, including Runge–Kutta algorithms and implicit methods (for so-called 'stiff' differential equations).

When multiple reactants or multiple reactions are involved, as is the case with most systems biology simulations, it is necessary to work with multiple ODEs or systems of differential equations. These, too, must be solved numerically using programs such as the LSODA (Livermore solver for ODEs with automatic method switching for stiff and non-stiff problems) or CVODE (C variable-coefficient ODE solver) integrators. Systems of ODEs lie at the heart of several fine grain cellular simulation efforts including V-cell [8], E-cell [9] and the yeast cell cycle [10]. ODEs have also been used to generate coarse grain models of several different organs or organ systems [11–13].

In the areas of drug discovery and development, the application of ODEs to coarse grain PK and pharmacodynamic simulations of the gut and liver is well known. Indeed, PK modeling might be regarded as one of the first and most successful examples of computational system biology [14,15]. Another coarse grain organ model that illustrates the potential of systems biology in drug discovery is the virtual heart, described by Noble [11]. This model exploits the huge body of macro- and microscopic knowledge that exists about the human heart. It also integrates organ level physiology with protein level transporter information. This model has already led to unexpected or counterintuitive predictions concerning the regulation of Na^{2+}/Ca^{2+} exchange in heart arrhythmias. The results actually suggest that drug discovery efforts should be focused on finding Na^{2+}/Ca^{2+} exchanger agonists for the treatment of arrhythmias, whereas in the past, most drug development focused on finding antagonists.

ODEs have also been used to model finer grain processes, such as intracellular and extracellular signaling. These cell signaling

TABLE 3

Partial list of computational systems biology simulation software packages

Package	Method[a]	URL and Reference	Remarks
Cell Designer	ODE, SDE	http://www.celldesigner.org/index.html [47]	Advanced GUI SBML, SBW
CellWare	ODE, SDE	www.cellware.org [48]	Simple GUI SBML
Dynetica	ODE, SDE	http://www.duke.edu/~you/Dynetica_page.htm [49]	Simple GUI
E-Cell	ODE, SDE, SSF	http://www.e-cell.org/ [50]	Object-oriented Python scripts SBML
Gepasi	ODE	http://www.gepasi.org/ [51]	Simple GUI SBML
SmartCell	ODE SDE	http://smartcell.embl.de/ [52]	Simple GUI, compartments
Vcell	ODE, PDE	http://www.vcell.org [53]	GUI, supports complex shapes, electric fields
Snoopy	Colored Petri Net	http://www-dssz.informatik.tu-cottbus.de/index.html?/software/snoopy.html	Simple GUI
CPN Tools	Colored Petri net	http://wiki.daimi.au.dk/cpntools/cpntools.wiki [54]	Simple GUI
Cell Illustrator- Animator	Hybrid Petri net	http://www.gene-networks.com [55]	Advanced GUI
MesoRD	Pseudo PDE	http://mesord.sourceforge.net/index.phtml [56]	Reaction–diffusion, SBML
SpiM	Pi calculus	http://research.microsoft.com/~aphillip/spim/	GUI
BioSPI	Pi calculus	http://www.wisdom.weizmann.ac.il/~biospi/index_main.html [57]	Sample models, Extensive manual
CancerSim	CA	http://www.cs.unm.edu/~forrest/software/cancersim/ [58]	GUI, 3D models, Movies
Mcell	DCA	http://www.mcell.cnl.salk.edu/ [59]	3D models, movies
SimCell	DCA	http://wishart.biology.ualberta.ca/SimCell/ [30]	GUI, 2D models, movies, SBML
AgentCell	ABM	http://flash.uchicago.edu/~emonet/biology/agentcell/ [60]	GUI, movies
Cell++	CA/PDE (hybrid)	http://www.compsysbio.org/CellSim/ [45]	GUI, movies

[a] Abbreviations: SDE, Stochastic differential equations; SSF, S-system formalism.

models are particularly useful for identifying drug targets, for finding the subcellular location of drug targets or for pinpointing key control points that would be useful for drug screening trials. They also enable multiple perturbations or multiple knockouts to be tested *in silico* without the need for genetic transformations or carefully controlled inhibitor dosing studies. One particularly interesting example involved the modeling of phosphorylation events in the Erbβ receptor-signaling pathway [6,16]: this pathway is known to be important in many different cancers and is the target for several drug candidates. Using this Erbβ model, Hendriks *et al.* [16] predicted the temporal phosphorylation processes of three different Erbβ receptors using 10 different ligands. This *in silico* modeling saved many weeks of experimental work, and also revealed that Erbβ phosphorylation occurred primarily in the endosomal compartments. This suggested that drug candidates that targeted the cell surface could be less effective than those that are more reactive in the endosome.

Historically, the time, programming knowledge and mathematical skill required to set up a useful ODE model put this approach beyond the reach of most experimental biologists. However, recent advances in graphical interface (GUI) design have improved standards in displaying and generating reaction models. In addition, the development of standardized mark-up languages, such as systems biology mark-up language (SBML) and cell mark-up language (Cell-ML), are making the generation and exchange of interesting biological models relatively simple [17]. A large repository of ODE-based Cell–ML metabolic and cell signaling models has been compiled at the Cell–ML model repository website (http://www.cellml.org/examples/repository/) as well as in the JWS online system (http://jjj.biochem.sun.ac.za), which is part of the silicon cell project [18]. Additionally, a large number of SBML models are located at the BioModels website (http://www.ebi.ac.uk/biomodels/). Table 3 provides a list of common ODE simulation packages, many of which are compatible with SBML or Cell-ML.

A key limitation of ODEs or systems of ODEs is the need for complete and quantitative data on concentrations, reaction rates, diffusion rates, degradation rates and many other parameters that are difficult to measure. Usually this kind of information is not available for all biological entities; therefore, approximations and estimates have to be used. Unfortunately, imprecise estimates can often lead to significant numerical instabilities in ODE simulators. These measurements, or their interpretation, is complicated by the fact that local concentrations of reactants or catalytic enzymes within compartments are often low and non-homogeneous. Even if local concentrations are explicitly modeled, the diffusion rates for species of considerably different physical size are often assumed to be identical. Obviously, under crowded conditions, small molecule diffusion will be less impeded than for larger proteins and protein complexes. Macromolecular crowding also affects reaction

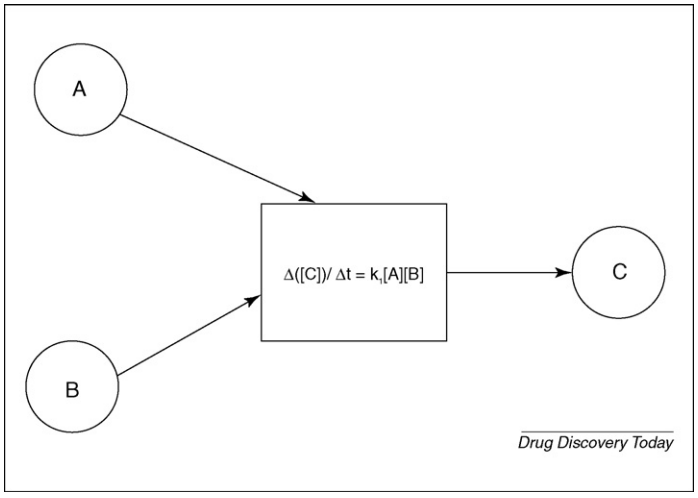

FIGURE 1

An illustration of a simple colored Petri net reaction of $A + B \xrightarrow{k_1} C$. Petri nets represent biochemical reactants and products as 'places' (circles) which react within 'transitions' (rectangles). Colored Petri nets represent molecular concentrations ('tokens' within places) using continuous values. Reaction kinetics are defined by equations within transitions.

kinetics. Equilibrium rate constants for macromolecular association reactions under crowded conditions can increase by two to three orders of magnitude compared with dilute concentrations [19]. Clearly these issues must be carefully considered when preparing or analyzing an ODE model of an intracellular system. For example, reaction rate constants can be empirically determined (or estimated from similar reactions) under conditions of macromolecular crowding, and corresponding adjustments made to the model where appropriate.

Systems simulations using Petri nets

Petri nets are a relatively non-mathematical alternative to ODEs for modeling time-dependent processes. Petri nets, which were originally developed in the 1960s, have long been used to model discrete distributed flow systems, such as data communications networks and manufacturing processes. It wasn't until 1993 that biologists realized that this modeling approach could be easily adapted to representing biological systems [20]. Petri nets were originally designed to function as discrete automata, but later enhancements have added the ability to deal with continuous quantities [21,22]. A Petri net contains two kinds of nodes, called 'places' and 'transitions', represented graphically by circles and rectangles, respectively. In a molecular model, each place is a species of molecule with some number of tokens inside, representing the number of molecules or concentration of that species, whereas transitions represent reactions. Places are connected to transitions by arrows (or 'directed arcs') either from source (input) places into the transition or from the transition to product (output) places. The stoichiometry of a reaction is indicated by a weight on the arc. Because Petri nets are a discrete system, they are driven in stepwise fashion by implicit time increments. A transition 'fires' (i.e. the reaction occurs) when the markings at all its input places are greater than the weights on its input arcs (i.e. when there are enough source molecules), producing the product of the appropriate weights on its output arcs (which are subsequently stored in the product places). As with ODEs, the output from a Petri net simulation is typically the time courses of molecular species.

The basic Petri net formalism has been extended to handle more complex modeling problems, many of which arise in biological simulations [23]. Hybrid Petri net and functional hybrid Petri net (FHPN) models enable the handling of either discrete or continuous values, thus permitting equivalent modeling power to more advanced ODE-based systems [22]. Timed Petri nets enable the implicit incorporation of deterministic delays in firing transitions. Stochastic Petri nets control transition firing with an exponentially distributed time delay, equivalent to 'chemical master equation' approximations of stochastic behavior [21]. Extensions of hybrid Petri nets are colored Petri nets, which enable the definition of mathematical relationships inside transitions governing the rate of firing [23]. The colored Petri net representation of a simple reaction is shown in Figure 1. Finally, hierarchical Petri nets are intended to support the composition of more complex models using combinations of previous models. No single implementation provides support for all variations.

The strong appeal for Petri nets among many biologists lies in the fact that they do not require a great deal of mathematical ability or knowledge, they do not always require precisely measured parameters before implementation (i.e. they are amenable to both qualitative and quantitative modeling), they can be used to generate compartmental models (although compartments must be represented explicitly, with different places representing the same chemical species in different compartments), they can describe both discrete and continuous events and they are amenable to construction using simple, easy-to-use graphical interfaces. Among the disadvantages of Petri nets are that, with the exception of the recently developed spatial hybrid variants, they cannot model spatially dependent processes (e.g. diffusion, growth or chemotaxis) and that for simpler Petri net models, the outputs are only linear representations of the expected exponential or sinusoidal processes. Table 3 includes several Petri net software packages used in computational systems biology.

Petri nets have been used to model a wide range of biological processes, including qualitative modeling of apoptosis [24], iron homeostasis [25] and the yeast mating response [25]. A particularly interesting application of Petri nets was recently demonstrated with the modeling and biomedical profiling of metabolic disorders [26]. Using the urea cycle as an example, Chen and Hoefstadt [26] built a hybrid Petri net that qualitatively modeled metabolite levels, transcription factor activity and signaling pathway changes for this complex pathway. This model successfully predicted the elevated arginine levels, hyperammonaemia, and mild increases in urine orotic acid found in patients with ornithine transcarbamalase deficiency (one of the key enzymes in the urea cycle). This model also enabled them to rationalize the potential therapeutic treatments for this disorder, including limited protein intake and the supplementation of the diet with arginine or citrulline. The authors argue that similar Petri net models could be used to assist with the diagnosis, drug development and treatment of a wide variety of metabolic disorders.

Systems simulation using cellular automata

Cellular automata (CA) are simple computer simulation tools that can be used to model both temporal and spatiotemporal processes

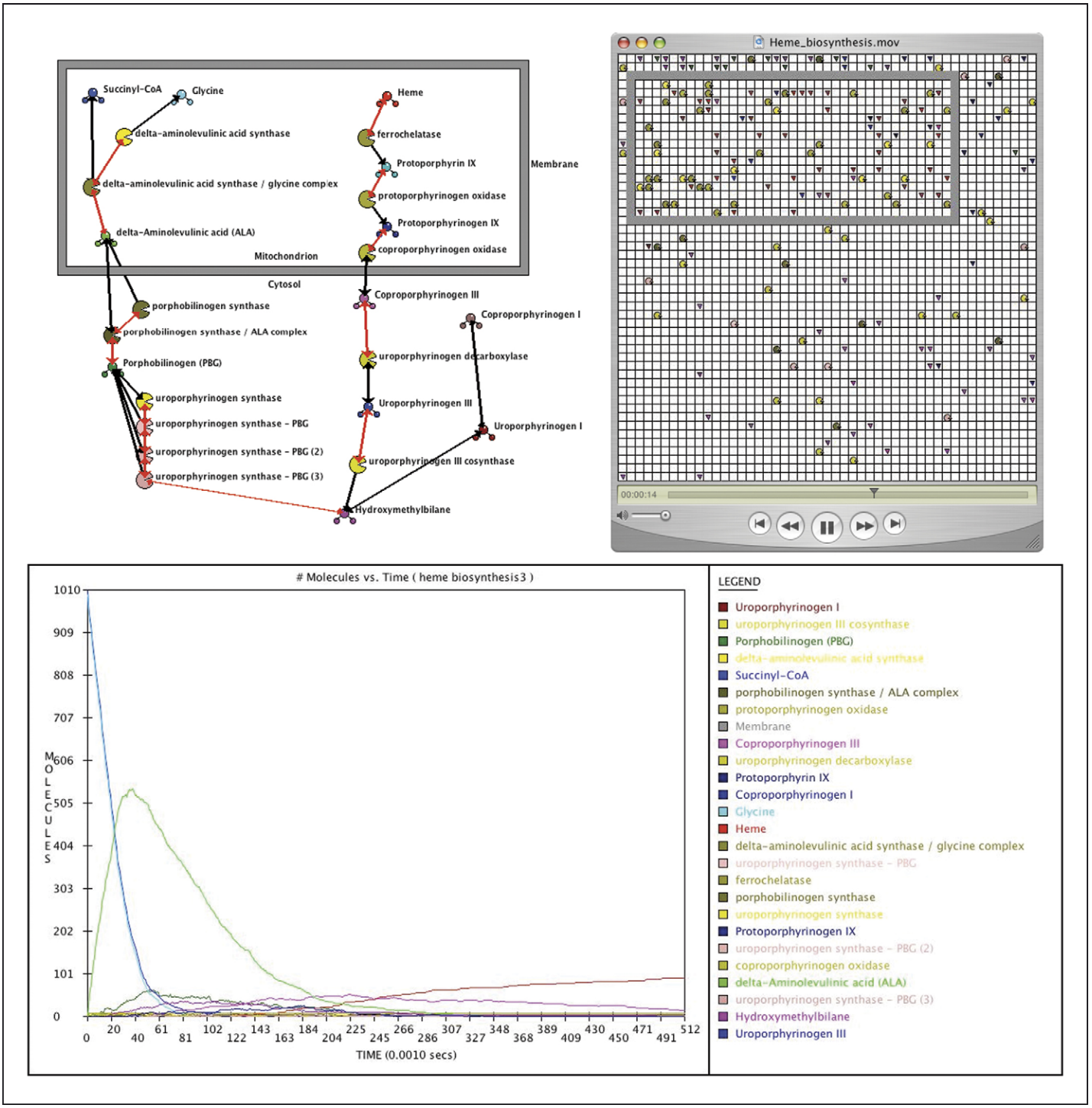

FIGURE 2

Screenshot montage of a CA simulation of Porphyria using SimCell. (a) The graphical definition of the compartments and reactions comprising the system are shown. **(b)** A sample 'movie shot' at a given time point is shown. **(c)** Representative graphic output of several species is shown.

using discrete time and/or spatial steps. Similar to Petri nets, CA models provide a relatively nonmathematical alternative to differential equations for spatiotemporal simulation. CAs normally consist of large numbers of near identical components with local interactions layered on a lattice or grid. The states or values of the components evolve synchronously in discrete time steps according to a set of rules. The value of a particular site is determined by the previous values or the states of the neighboring sites. Cellular automata were invented in the late 1940s by von Neumann and Ulam [27] and have been used to model a wide range of physical processes, including heat flow, spin networks and reaction–diffusion processes [28]. Cellular automata also have a long history in biological modeling. Indeed, one of the first computer applications in biology was a CA simulation called Conway's Game of Life [29]. This simple model simulated the birth, death and interaction between cells randomly placed on a square lattice or grid. The fate

of every cell was determined according to pairwise interaction rules. These interaction rules were typically Boolean logic conditions describing what a cell could do depending on the number of adjacent neighbors. From these simple rules some remarkably interesting behaviors or patterns would emerge.

Strictly speaking, the objects (cells, proteins or reagents) in a CA simulation do not move: they only appear, change properties or disappear. Object properties, attributes or information are the only things that 'move' in CA simulations. Variations on the CA model, known as dynamic cellular automata (DCA), actually enable objects to exhibit motions [30]. DCA models permit 'Brownian-like' motion of individual molecules through a random number generator, which selects a direction of motion during each time step. Depending on the implementation of the DCA algorithm, molecules can move one or more cells in a single time step. DCA models permit considerably more flexibility in simulating biological processes.

In addition to their simplicity or nonmathematical nature, CA (or DCA)-modeling methods have several advantages compared with ODEs and Petri nets. This is because CA models permit the modeling of both continuous and stochastic spatial and/or temporal processes, including discontinuous state changes, transport processes, diffusion, compartmentalization, cell migration, cell death, viral infection and many other common biological events. Furthermore, CA methods generate both graphs and 'movies' of the process being modeled. These movies can be particularly informative when attempting to understand processes such as chemotaxis, drug partitioning or tissue remodeling. Relative to ODEs and PDEs, CA methods are particularly robust (i.e. not prone to divide-by-zero errors) and are easily scaled (from nm to meters, or from nanoseconds to minutes). They also enable qualitative and quantitative models to be easily generated. Similar to Petri nets, CA models are particularly amenable to construction using simple, easy-to-use graphical interfaces (Figure 2). Among the disadvantages of CA methods are the facts that they cannot model processes as efficiently or as quickly as ODEs, PDEs or Petri nets, and that the standard rate constants used in reaction modeling are not as easily translated to the probabilistic formalisms used in CA models. Table 3 also includes a list of several CA and DCA systems currently used in systems biology.

CA models have been used to model a wide variety of biological or processes including basic enzyme kinetics [30,31] oscillatory gene circuits [30], myxobacterial aggregation [32] and even predator–prey relationships [33]. The granularity of these system models varies enormously. However, one of the strengths of CA simulations is their scale-free nature [30]. In terms of pharmaceutical applications, CA methods have been used to model drug release in bio-erodible microspheres [34], lipophilic drug diffusion [30,35] and drug-carrying micelle formation [36]. More recently, CA methods have been used to model the progression of HIV/AIDS and HIV treatment strategies [37,38]. By accounting for general features of the host immune response, the localization of infection (lymph nodes) and the rapid viral mutation rate, the CA model described in [37] was able to reproduce accurately the three-phase pattern commonly found in patient T-cell counts and viral loads. Peer *et al.* [38] extended this model to show how the response to different drug therapies or combination therapies could be qualitatively simulated. These authors noted that one of the key advantages of a CA model compared with equivalent ODE or PDE models was its capacity to model efficiently the extreme time scales (days to decades) and to simulate the spatial heterogeneity of viral infection.

Systems simulation using agent-based models

Agent-based models (ABMs) are almost identical in concept and design to DCA. In ABMs, drugs, metabolites, proteins or cells are the 'agents', which are permitted to interact with each other through space and time, according to a predefined set of rules. The motions can be either directed or random (Brownian), and the rules can be simple (physics based) or highly complex (using previous knowledge). Unlike CA models, agent-based systems do not formally require spatial grids or synchronized time steps, although practical coding considerations usually force these constraints on ABMs. The same advantages and disadvantages exist for ABMs that do with DCA or CA models.

ABMs have been used to simulate bacterial chemotaxis [39], to model the calcium-dependent cell migration events in wound healing [40] and to develop optimal breast cancer vaccination protocols [41]. One particularly interesting application of ABMs involved the prediction of clinical trial outcomes of different anti-cytokine treatments for sepsis [42]. In this study an ABM model of the innate immune response was constructed using extensive literature data and information about all the relevant cell types, cell functions and cell mediators (cytokines). The resulting model was found, consistently, to match the literature observations about cytokine levels and innate immune responses to various immune insults, including sepsis. This model was then used to test a series of hypothetical anti-cytokine treatments (anti-TNF, anti-IL-1 or anti-CD-18) with varying treatment length, dosages and combinations, in an effort to identify an optimal sepsis treatment regime. The model unexpectedly revealed that none of the proposed therapies, neither alone nor in combination, improved outcome. These results were also shown to match the poor results of the INTERSEPT and NORASEPT phase III anti-TNF trials [42]. Had this model been developed earlier, it clearly could have provided some useful warnings about the possible failure of the phase III anti-TNF drug trials. It also presaged the lack of success for the anti-IL-1 and anti-CD-18 sepsis therapies that were entering phase I and II trials.

Conclusions

The proliferation of approaches and software implementations (Table 3) makes it difficult to select the 'best' approach for a particular modeling problem. The choices made often depend on the expertise of the user, the type of desired output, the knowledge of the model parameters (complete or incomplete) and the complexity of the system to be modeled [43]. CAs, DCAs and ABMs permit relatively simple, nonmathematical, visually appealing, spatiotemporal modeling, enabling compartmentalization, infection, diffusion and stochasticity to be easily simulated. However, these modeling methods quickly become computationally intensive, particulrly when the number of molecules increases [44]. Compared with CAs and ABMs, more traditional ODE methods, pi calculus approaches and Petri net systems can more efficiently model continuous time-dependent processes for well-mixed reactants in single compartments. However, these methods

do not easily capture spatiotemporal events nor do they mimic the true granularity or stochasticity of living systems.

Several researchers have recently recommended hybrid or hierarchical hybrid systems to combine the strengths of both discrete and continuous approaches [44–46]. For instance, a recent hybrid CA-PDE model was used to simulate tumor growth and tumor interaction with the immune system [46]. Cells were represented on a square grid supplied by nutrients along the top and bottom edges. PDE equations governed the diffusion of two small-molecule nutrients, one required for cancer cell survival, the other for division. At each time cycle, diffusion of the nutrients was first simulated and the new concentrations imposed on the grid. Then cells responded to both the nutrient level as well as to neighboring cells according to specified CA rules. The authors found that, although basic tumor growth was accurately modeled, different recruitment and killing parameters for CTLs lead to oscillations in both the tumor and immune cell populations.

A primary benefit from this hybrid model was the ability to integrate processes that occur rapidly (diffusion) with processes that can take days (tumor growth and migration). This mixing of widely differing scales of time and space is normally one of the largest impediments to modeling accurately biological processes using reasonable computer power [38,44]. Given the success of these newly emerging hybrid models, it is probable that the future of computational systems biology, just like systems biology itself, will depend on finding ways to better integrate a multiplicity of tools and methods.

Acknowledgements

The authors wish to thank the National Institute for Nanotechnology (NINT), the National Research Council of Canada (NRC) and Genome Alberta, a division of Genome Canada, for financial support.

References

1 Hood, L. (2003) Systems biology: integrating technology, biology, and computation. *Mech. Ageing Dev.* 124, 9–16
2 Ideker, T. *et al.* (2001) A new approach to decoding life: systems biology. *Annu. Rev. Genomics Hum. Genet.* 2, 343–372
3 Kitano, H. (2002) Computational systems biology. *Nature* 420, 206–210
4 Bugrim, A. *et al.* (2004) Early prediction of drug metabolism and toxicity: systems biology approach and modeling. *Drug Discov. Today* 9, 127–135
5 Cho, C.R. *et al.* (2006) The application of systems biology to drug discovery. *Curr. Opin. Chem. Biol.* 10, 294–302
6 Kumar, N. *et al.* (2006) Applying computational modeling to drug discovery and development. *Drug Discov. Today* 11, 806–811
7 de Jong, H. (2002) Modeling and simulation of genetic regulatory systems: a literature review. *J. Comput. Biol.* 9, 67–103
8 Slepchenko, B.M. *et al.* (2003) Quantitative cell biology with the Virtual Cell. *Trends Cell Biol.* 13, 570–576
9 Nakayama, Y. *et al.* (2005) Dynamic simulation of red blood cell metabolism and its application to the analysis of a pathological condition. *Theor. Biol. Med. Model.* 2, 18
10 Sveiczer, A. *et al.* (2004) Modelling the fission yeast cell cycle. *Brief. Funct. Genomic. Proteomic.* 2, 298–307
11 Noble, D. (2006) Systems biology and the heart. *Biosystems* 83, 75–80
12 Shorten, P.R. and Upreti, G.C. (2005) A mathematical model of fatty acid metabolism and VLDL assembly in human liver. *Biochim. Biophys. Acta* 1736, 94–108
13 Crampin, E.J. *et al.* (2004) Computational physiology and the Physiome Project. *Exp. Physiol.* 89, 1–26
14 Huisinga, W. *et al.* (2006) The virtual laboratory approach to pharmacokinetics: design principles and concepts. *Drug Discov. Today* 11, 800–805
15 Mager, D.E. (2006) Quantitative structure-pharmacokinetic/pharmacodynamic relationships. *Adv. Drug Deliv. Rev.* 58, 1326–1356
16 Hendriks, B.S. *et al.* (2006) Computational modelling of ErbB family phosphorylation dynamics in response to transforming growth factor alpha and heregulin indicates spatial compartmentation of phosphatase activity. *Syst. Biol.* 153, 22–33
17 Schilstra, M.J. *et al.* (2006) CellML2SBML: conversion of CellML into SBML. *Bioinformatics* 22, 1018–1020
18 Snoep, J.L. *et al.* (2006) Towards building the silicon cell: a modular approach. *Biosystems* 83, 207–216
19 Ellis, R.J. (2001) Macromolecular crowding: obvious but underappreciated. *Trends Biochem. Sci.* 26, 597–604
20 Reddy, V.N. *et al.* (1993) Petri net representations in metabolic pathways. *Proc. Int. Conf. Intell. Syst. Mol. Biol.* 1, 328–336
21 Goss, P.J. and Peccoud, J. (1998) Quantitative modeling of stochastic systems in molecular biology by using stochastic Petri nets. *Proc. Natl. Acad. Sci. U. S. A.* 95, 6750–6755
22 Matsuno, H. *et al.* (2003) Biopathways representation and simulation on hybrid functional Petri net. *In Silico Biol.* 3, 389–404
23 Pinney, J.W. *et al.* (2003) Petri net representations in systems biology. *Biochem. Soc. Trans.* 31, 1513–1515
24 Heiner, M. *et al.* (2004) Model validation of biological pathways using Petri nets – demonstrated for apoptosis. *Biosystems* 75, 15–28
25 Sackmann, A. *et al.* (2006) Application of Petri net-based analysis techniques to signal transduction pathways. *BMC Bioinformatics* 7, 482
26 Chen, M. and Hofestadt, R. (2006) A medical bioinformatics approach for metabolic disorders: biomedical data prediction, modeling, and systematic analysis. *J. Biomed. Inform.* 39, 147–159
27 von Neumann, J. (1966) *The Theory of Self-reproducing Automata* (Burks, A., ed.), University of Illinois Press
28 Wolfram, S. (2002) *A New Kind of Science*. Wolfram Media
29 Gardner, M. (1970) Mathematical games. *Sci. Am.* 223, 120–123
30 Wishart, D.S. *et al.* (2005) Dynamic cellular automata: an alternative approach to cellular simulation. *In Silico Biol.* 5, 139–161
31 Kier, L.B. *et al.* (1996) A cellular automata model of enzyme kinetics. *J. Mol. Graph.* 14, 227–231
32 Sozinova, O. *et al.* (2005) A three-dimensional model of myxobacterial aggregation by contact-mediated interactions. *Proc. Natl. Acad. Sci. U. S. A.* 102, 11308–11312
33 Kondoh, M. (2003) High reproductive rates result in high predation risks: a mechanism promoting the coexistence of competing prey in spatially structured populations. *Am. Nat.* 161, 299–309
34 Zygourakis, K. and Markenscoff, P.A. (1996) Computer-aided design of bioerodible devices with optimal release characteristics: a cellular automata approach. *Biomaterials* 17, 125–135
35 Kier, L.B. *et al.* (1997) A cellular automata model of diffusion in aqueous systems. *J. Pharm. Sci.* 86, 774–778
36 Kier, L.B. *et al.* (1996) A cellular automata model of micelle formation. *Pharm. Res.* 13, 1419–1422
37 Zorzenon dos Santos, R.M. and Coutinho, S. (2001) Dynamics of HIV infection: a cellular automata approach. *Phys. Rev. Lett.* 87, 168102
38 Peer, M.A. *et al.* (2004) Cellular automata and its advances to drug therapy for HIV infection. *Indian J. Exp. Biol.* 42, 131–137
39 Emonet, T. *et al.* (2005) AgentCell: a digital single-cell assay for bacterial chemotaxis. *Bioinformatics* 21, 2714–2721
40 Walker, D.C. *et al.* (2004) Agent-based computational modeling of wounded epithelial cell monolayers. *IEEE Trans. Nanobioscience* 3, 153–163
41 Lollini, P.L. *et al.* (2006) Discovery of cancer vaccination protocols with a genetic algorithm driving an agent-based simulator. *BMC Bioinformatics* 7, 352
42 An, G. (2004) *In silico* experiments of existing and hypothetical cytokine-directed clinical trials using agent-based modeling. *Crit. Care Med.* 32, 2050–2060
43 Alves, R. *et al.* (2006) Tools for kinetic modeling of biochemical networks. *Nat. Biotechnol.* 24, 667–672
44 Ridgway, D. *et al.* (2006) Accommodating space, time and randomness in network simulation. *Curr. Opin. Biotechnol.* 17, 493–498
45 Sanford, C. *et al.* (2006) Cell++ – simulating biochemical pathways. *Bioinformatics* 22, 2918–2925

46 Mallet, D.G. and De Pillis, L.G. (2006) A cellular automata model of tumor–immune system interactions. *J. Theor. Biol.* 239, 334–350
47 Kitano, H. *et al.* (2005) Using process diagrams for the graphical representation of biological networks. *Nat. Biotechnol.* 23, 961–966
48 Dhar, P. *et al.* (2004) Cellware – a multi-algorithmic software for computational systems biology. *Bioinformatics* 20, 1319–1321
49 You, L. *et al.* (2003) Modeling biological systems using Dynetica – a simulator of dynamic networks. *Bioinformatics* 19, 435–436
50 Tomita, M. *et al.* (1999) E-CELL: software environment for whole-cell simulation. *Bioinformatics* 15, 72–84
51 Mendes, P. (1993) GEPASI: a software package for modelling the dynamics, steady states and control of biochemical and other systems. *Comput. Appl. Biosci.* 9, 563–571
52 Ander, M. *et al.* (2004) SmartCell, a framework to simulate cellular processes that combines stochastic approximation with diffusion and localisation: analysis of simple networks. *Syst. Biol.* 1, 129–138
53 Loew, L.M. and Schaff, J.C. (2001) The Virtual Cell: a software environment for computational cell biology. *Trends Biotechnol.* 19, 401–406
54 Lee, D.-Y. *et al.* (2006) Colored Petri net modeling and simulation of signal transduction pathways. *Metab. Eng.* 8, 112–122
55 Peleg, M. *et al.* (2005) Using Petri net tools to study properties and dynamics of biological systems. *J. Am. Med. Inform. Assoc.* 12, 181–199
56 Hattne, J. *et al.* (2005) Stochastic reaction–diffusion simulation with MesoRD. *Bioinformatics* 21, 2923–2924
57 Lecca, P. *et al.* (2004) A BioSpi model of lymphocyte–endothelial interactions in inflamed brain venules. *Pac. Symp. Biocomput.* 521–532
58 Abbott, R.G. *et al.* (2006) Simulating the hallmarks of cancer. *Artif. Life* 12, 617–634
59 Stiles, J.R. and Bartol, T.M. (2001) Monte Carlo methods for simulating realistic synaptic microphysiology using MCell. In *Computational Neuroscience: Realistic Modeling for Experimentalists* (De Schutter, E., ed.), pp. 87–127, CRC Press
60 Emonet, T. *et al.* (2005) AgentCell: a digital single-cell assay for bacterial chemotaxis. *Bioinformatics* 21, 2714–2721