ACS SyntheticBiology

Letter

pubs.acs.org/synthbio

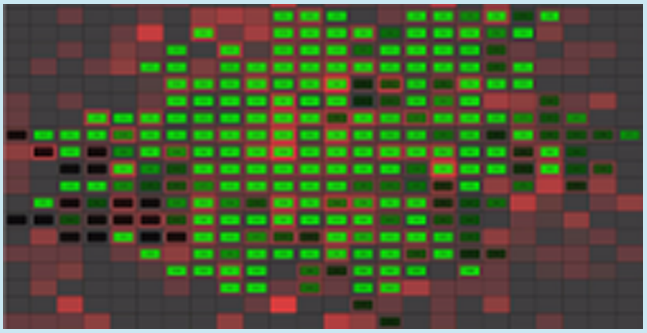
Dynamic Modeling of Cellular Populations within iBioSim

Jason T. Stevens† and Chris J. Myers*

Department of Electrical and Computer Engineering, University of Utah, Salt Lake City, Utah 84112, United States

Supporting Information

ABSTRACT: As the complexity of synthetic genetic circuits increases, modeling is becoming a necessary first step to inform subsequent experimental efforts. In recent years, the design automation community has developed a wealth of computational tools for assisting experimentalists in designing and analyzing new genetic circuits at several scales. However, existing software primarily caters to either the DNA- or single-cell level, with little support for the multicellular level. To address this need, the iBioSim software package has been enhanced to provide support for modeling, simulating, and visualizing dynamic cellular populations in a two-dimensional space. This capacity is fully integrated into the software, capitalizing on iBioSim's strengths in modeling, simulating, and analyzing single-celled systems.



KEYWORDS: *spatial modeling, dynamic modeling, multicellular modeling, stochastic simulation, genetic circuits, SBML*

Multicellularity and the resulting emergent behaviors give rise to much of the complexity found in nature. Biology is replete with elegant developmental programs that rely on the communication and cooperation of cells to form complicated structures such as tissues or biofilms. Synthetic biology promises to develop cellular populations for a number of exciting medical applications including treatment of cancer and infections, development of vaccines, microbiome engineering, and cell therapy and regenerative medicine.[1] Moreover, engineering at this scale enables biologists to create synthetic developmental programs in order to better understand how natural developmental programs work and the ways in which they fail.

While much of the initial focus of synthetic biology has been on engineering single-cell behaviors, a number of publications have shown the potential for multicellular engineering. Using a genetic bandpass filter and the quorum sensing circuitry from *Vibrio fischeri*, patterns can be generated in an *E. coli* population.[2] More recently, it has been shown that a genetic density sensor and an engineered chemotaxis pathway can be used to create concentric circle patterns with *E. coli*.[3] Along with pattern formation, distributed computation of logic gate functions has been demonstrated using populations of yeast and *E. coli*.[4]

As researchers' ability to engineer synthetic genetic circuits grows, tools for modeling and simulating multicellular systems are needed to augment the wet lab advances and catalyze the move toward multicellular engineering. Current software primarily caters to DNA assembly, plasmid design, and single-cell modeling.[5−9] Tools do exist for multicellular modeling;[10−12] however, most are insufficient for synthetic biologists because of their lack of support for design-oriented uses. Recently, gro[13] and CellModeller4,[14] software for multicellular modeling with a design orientation, have been published. iBioSim distinguishes itself from these two programs with its graphical user interface, analysis and abstraction tools, the use of standards representations, physics-agnostic spatial modeling, and Gillespie SSA-based simulation.

iBioSim has been in active development for several years.[15,16] Aimed at the synthetic biologist, it provides an integrated environment for modeling and analyzing genetic circuits, enabling efficient design space exploration. iBioSim supports the representation of these circuits using a model stored in the *Systems Biology Markup Language* (SBML)[17] that can be annotated with DNA components from a collection described using the *Synthetic Biology Open Language* (SBOL).[18] These models can be quickly created and modified using a drag-and-drop graphical user interface (see Figure 1, as well as a movie in the Supporting Information). Several analysis methods are supported, including ordinary differential equations, stochastic simulation, and Markovian analysis. The efficiency of these methods is enhanced by automatic reaction-based and logical abstraction methods.[19−23] Simulation data can be analyzed using built-in graphing and visualization tools or saved as one of several data formats (e.g., CSV) for import into other analysis programs.

This paper describes enhancements to iBioSim to support multicellular modeling, namely, a graphical interface for design, a spatial modeling framework, SBML annotations for dynamic process events, an SSA-based simulator with support for cellular population dynamics, and a visualization environment for

Special Issue: IWBDA 2012

Received: September 14, 2012
Published: November 21, 2012

   dx.doi.org/10.1021/sb300082b | *ACS Synth. Biol.* 2013, 2, 223−229

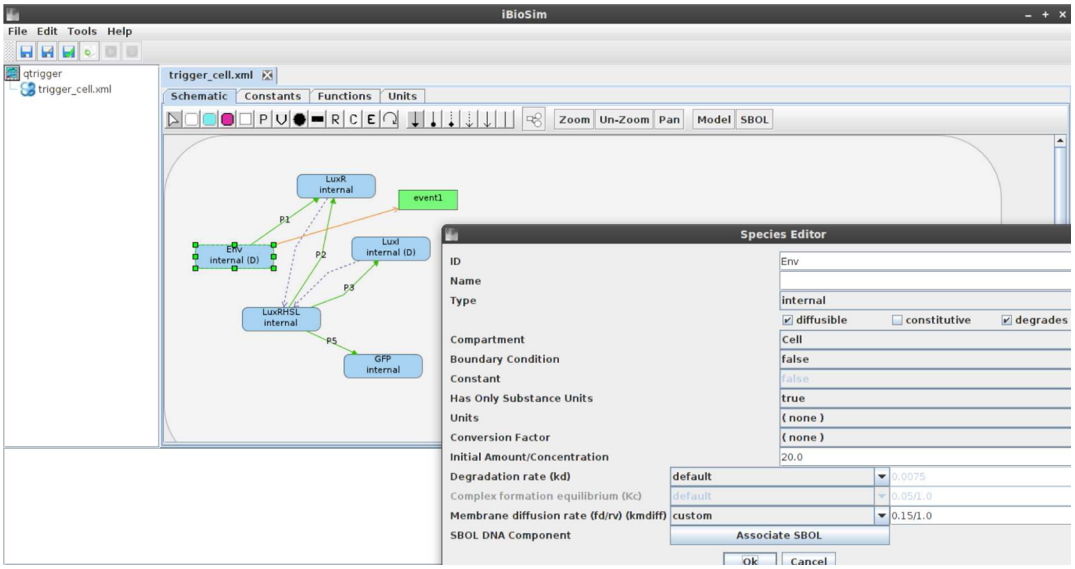


Figure 1. The iBioSim graphical user interface for model creation. To the left is the list of the models in the current workspace. The center panel shows the graphical interface for genetic circuit editing. A simple genetic circuit is shown with species in blue. The arcs represent repression or activation relationships between the species. The green rectangle represents an event. The panel in the foreground shows an editor for a particular selected species (the blue rectangle highlighted with green dots). This editor can be used to alter kinetic parameters and attributes for the species and to associate SBOL constructs with the species.

analysis of simulation data. A running example to illustrate these new features is the design and analysis of a *quorum trigger genetic circuit*,[24] which is shown in Figure 2. The quorum trigger

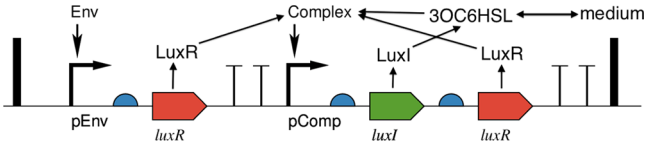


Figure 2. The genetic circuit diagram for the quorum trigger represented using SBOL visual symbols. This circuit is designed to produce a density-dependent response to an environmental signal (Env). This response is achieved using the quorum sensing molecule 3OC6HSL. The 3OC6HSL molecule can either come from LuxI generated at the basal rate of the pComp promoter, or it can come from diffusion into the cell from 3OC6HSL available in the medium that is produced by other cells.

circuit is a sensor that enables a cell to detect the presence of a signaling molecule in its environment (Env). The circuit decides that it is in the presence of the signaling molecule when it both senses the molecule itself and receives a signal from its neighbor cells, via quorum sensing, that they also believe the signal is present. Clearly, the utility of this circuit can be evaluated only by considering population dynamics.

Spatial Modeling. All of the population-based enhancements to iBioSim rely on a spatial modeling framework. This framework is grid-based, with a single compartment (e.g., a cell) allowed at each grid location. A compartment can contain a model (e.g., a genetic circuit), allowing for an arbitrary number of these throughout the grid. Each grid location is considered distinct from every other, creating spatial separation. Moreover, if a compartment is present at a grid location, separation exists between the compartment and the grid location space. The compartment is considered to be a well-mixed space, as is each grid location. These separations thus create a representation of two-dimensional space and provide a basis for diffusion reactions.

Within a grid location, the separation between the grid space and compartment space can be bridged with reversible diffusion reactions. As these reactions go between an "intracellular" and "extracellular" space, they can be thought of as membrane diffusion reactions. Similarly, between grid locations, grid spaces can be connected with diffusion reactions, which can be thought of as spatial diffusion reactions. These allow species to travel across the grid and therefore from one cell to another, enabling the modeling of cellular signaling. Figure 3 shows the

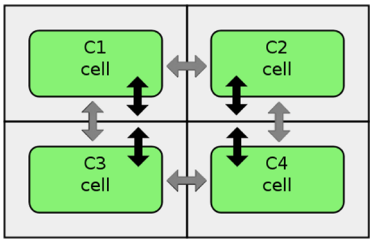


Figure 3. A two-by-two grid with each location containing a single cell model. The arrows indicate the diffusion reactions that can connect the discrete well-mixed spaces. Gray arrows indicate diffusion between two grid locations, and black arrows indicate diffusion between a cell's interior space and a grid location.

possible diffusion reactions on a 2 × 2 grid. Diagonal diffusion reactions between grid locations are not created due to the added computational requirements and the fact that such a diagonal diffusion reaction would occur with low probability.

Since the user only explicitly creates species within the compartments, each diffusible species has one *grid species* per grid location created automatically. Membrane diffusion reactions are added to transfer between the interior species and the grid species, and spatial diffusion reactions are added to transfer between two grid species. Species marked as diffusible within the model editor cause the software to automatically

dx.doi.org/10.1021/sb300082b | *ACS Synth. Biol.* 2013, 2, 223−229

```
<reaction metaid="Degradation_Env" id="Degradation_Env" reversible="false" fast="false" compartment="Grid">
  <annotation>
    <ibiosim:ibiosim xmlns:ibiosim="http://www.fakeuri.com" ibiosim:type="grid"/>
  </annotation>
  <listOfReactants>
    <speciesReference species="Env" stoichiometry="1" constant="false"/>
  </listOfReactants>
  <kineticLaw>
    <math xmlns="http://www.w3.org/1998/Math/MathML">
      <apply>
        <times/>
        <ci> kecd </ci>
        <apply>
          <ci> get2DArrayElement </ci>
          <ci> Env </ci>
          <ci> i </ci>
          <ci> j </ci>
        </apply>
      </apply>
    </math>
    <listOfLocalParameters>
      <localParameter id="i">
        <annotation>
          <array:array xmlns:array="http://www.fakeuri.com" array:min="0" array:max="2"/>
        </annotation>
      </localParameter>
      <localParameter id="j">
        <annotation>
          <array:array xmlns:array="http://www.fakeuri.com" array:min="0" array:max="2"/>
        </annotation>
      </localParameter>
      <localParameter id="kecd" value="0.005" units="u_1_second_n1"/>
    </listOfLocalParameters>
  </kineticLaw>
</reaction>
```

Figure 4. An example of the use of SBML dynamic array annotations. This block of SBML is an array of nine degradation reactions. The arrays are described in the annotation tags for the parameters and the reaction. Additionally, a function get2DArrayElement is called within the kinetic law of the reaction. This function takes array indices and returns the requested array element. By using annotations on parameters in the kinetic law to encapsulate information about the number of reactions, the model is more compact and is easier to expand and contract by simply modifying the annotations.

generate these species and reactions. Forward and reverse diffusion rates are associated with a particular species and can be specified by the user. These rates are applied to all diffusion reactions for that species, in every compartment. Diffusion rates can be set to zero if the user does not want the reactions to occur. In the future, the user will be able to specify unique diffusion rates for individual compartments.

The diffusion reactions are simple first-order reactions of the form:

$$s_e^{(i,j)} \xrightarrow{k_{diff}} s_e^{(i+1,j)}$$

$$s_e^{(i+1,j)} \xrightarrow{k_{diff}} s_e^{(i,j)}$$

$$s_e^{(i,j)} \xrightarrow{k_{diff}} s_e^{(i-1,j)}$$

$$s_e^{(i-1,j)} \xrightarrow{k_{diff}} s_e^{(i,j)}$$

$$s_e^{(i,j)} \xrightarrow{k_{diff}} s_e^{(i,j+1)}$$

$$s_e^{(i,j+1)} \xrightarrow{k_{diff}} s_e^{(i,j)}$$

$$s_e^{(i,j)} \xrightarrow{k_{diff}} s_e^{(i,j-1)}$$

$$s_e^{(i,j-1)} \xrightarrow{k_{diff}} s_e^{(i,j)}$$

$$s_e^{(i,j)} \xrightarrow{k_{idiff}} s^{(i,j)}$$

$$s^{(i,j)} \xrightarrow{k_{ediff}} s_e^{(i,j)}$$

where $s^{(i,j)}$ is the amount of species s within the cell in the location (i,j), $s_e^{(i,j)}$ is the amount of species s outside the cell in the location (i,j), k_{diff} is the diffusion rate of species s within the grid, k_{idiff} is the membrane diffusion rate entering the cell, and k_{ediff} is the membrane diffusion rate exiting the cell.

Dynamic Modeling. While static spatial modeling and diffusion can enable the creation of models for many interesting applications, certain important phenomena such as population control and artificial developmental programs (e.g., cells apoptosing to reveal a pattern) require dynamic events: cell movement, duplication, and death. To provide this capacity, iBioSim supports new dynamic events for compartment models. These are composed of regular SBML events with an additional annotation to specify a dynamic action that occurs when the event fires. SBML events themselves are composed of a trigger expression, a delay value, and assignments. During every time-step of a simulation, trigger expressions are evaluated for all untriggered events. If a trigger evaluates to true, the event becomes triggered and fires after the given delay. When the event fires, the specified assignments are performed. If an additional dynamic annotation is attached to an event, the simulator in iBioSim interprets and performs the dynamic action when it fires. There are three dynamic events: *death*, *duplication*, and *movement*.

A death event removes all traces of whichever compartment the event is triggered within from the simulation data structures. In other words, all species and other SBML elements associated with this compartment are removed from the simulation from this point forward. A future enhancement of the death event that we are considering is to potentially

dx.doi.org/10.1021/sb300082b | *ACS Synth. Biol.* 2013, 2, 223−229

move some portion of the diffusible species into the corresponding species in the grid location.

A duplication event creates a new copy of the compartment, using the existing compartment's SBML elements (i.e., species, reactions, etc.) as templates for the new version. Species are apportioned to the parent and child compartments via a 50/50 split by default, but event assignments associated with the duplication event can be used to override this. In the latter case, the child gets whatever is left over after the assignments to the parent take place in order to conserve species counts.

Symmetric and asymmetric versions of duplication events are provided. Symmetric duplication resets the duplication event trigger of the parent compartment, allowing it to fire again, while asymmetric duplication does not. Symmetric division can thus be thought of as binary fission, where the two resulting cells do not share a parent-child relationship. Asymmetric division can be used to model parent-child relationships, in which the parent is no longer able to divide.

For visualization purposes, the locations of compartments within the grid are tracked, and new child compartments after duplication are placed in a random neighboring location to the parent (including diagonally adjacent locations), with existing compartments shifted out of the way in the direction of movement to open up the adjacent location. If the new child compartment is placed in a location outside of the current grid bounds, or if a shifted cell is shifted outside of the current grid bounds, the grid automatically expands, creating new grid diffusion species and reactions for the new locations.

A movement event simply moves a compartment to one of the eight neighboring locations, either randomly or as specified by the user. As with duplication, compartments are shifted to make room for the moved compartment. To enable movement based on species amounts in neighboring locations, functions are provided for movement events, which can be used in the trigger expressions. For example, a threshold species amount in a neighboring location can be used to move the compartment in that direction, enabling chemotaxis modeling.

SBML Modeling. All of the models in iBioSim are represented using SBML Level 3 Version 1. The current version of SBML supports only static modeling. The creation or destruction of SBML objects cannot be specified, meaning that a model must remain constant throughout simulation. Dynamic models by definition change at runtime; thus, enhancements to SBML are required in order for it to support dynamic models.

To circumvent these limitations, custom annotations are used to encode dynamic maps and the aforementioned dynamic events. Dynamic map annotations are used in several places. They are used to represent name-location pairs for compartments; they are used for representing arrays of dynamic events and grid-level reactions, that is, diffusion and degradation reactions for grid species; and they are used for representing arrays of identical or different submodels that comprise the grid. For simulation, these SBML annotations are used as templates for creating the simulator's data structures. Figure 4 shows an example of the annotations. These measures are intended to be temporary, as a more permanent solution for dynamic modeling within SBML is under development with the SBML community. Our annotation strategy should provide a foundation for the future designs of SBML packages to support these models.

Multicellular Simulation. The discrete nature of the multicellular modeling framework used lends itself to stochastic simulation and can thus be simulated using a standard Gillespie SSA simulator. However, dynamic events require the model to be altered at runtime, something that many simulators do not support. Our new simulator supports dynamic events as well as the large-scale nature of multicellular models (i.e., thousands of reactions). The simulator uses the recently developed SSA composition and rejection method.[25] The composition and rejection method creates and maintains groups of reactions according to their propensities during runtime. To choose a reaction, the algorithm randomly chooses a group and then randomly chooses a reaction and a propensity. If the propensity is less than the chosen reaction's propensity, that reaction is chosen; otherwise, a new reaction and propensity are chosen within the same group until the process finds a reaction to fire. This approach is equivalent to previous exact Gillespie algorithms, and can be programmed in a constant-time manner.

Our experience indicates that this algorithm is faster and scales better than the Gillespie SSA Direct method with a dependency graph. Figure 5 shows simulation time data for the

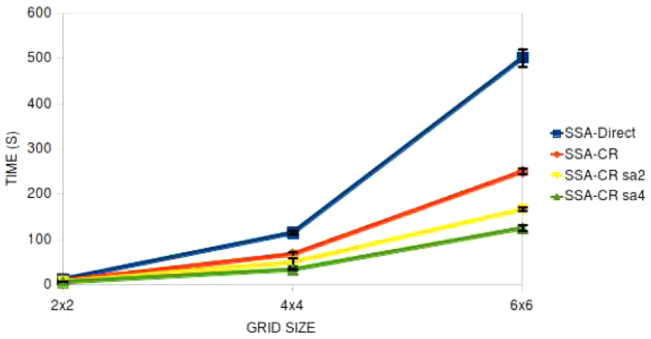


Figure 5. Runtime comparison between different Gillespie SSA algorithms on increasingly larger grid models of a population of cells containing a quorum trigger circuit. The sa2 and sa4 data are from using stoichiometry amplification values of 2 and 4, respectively. The results are the average runtimes over five simulation runs. Error bars are included to show 1 standard deviation.

SSA Direct method and the SSA-CR algorithm. These are run using increasingly large (2×2, 4×4, 6×6) static grids with 152, 640, and 1464 reactions, respectively. The SBML for the 2×2 model can be found in the Supporting Information.

When the counts of extracellular grid species grow large, the propensities of the corresponding diffusion reactions between the grid species during simulation grow large as well, resulting in a simulation bottleneck. Indeed, in excess of 99% of the reactions being fired can be grid diffusion reactions. To address this, our tool utilizes a simple *stoichiometry amplification abstraction*,[20] which allows users to group diffusion reactions and speed up simulation time. The algorithm works by multiplying the stoichiometry of the diffusion reaction by n and dividing the reaction's propensity by n. For instance, if a stoichiometry amplification value of 5 is chosen, extracellular grid reactions move five species per reaction, and this reaction's propensity is divided by five. So the reaction occurs one-fifth as frequently, but it moves five times the species. This speeds up simulation time significantly (up to n times as fast) for models with large quantities of diffusible species without appreciable macroscopic differences in the simulation outcome. Figure 5 shows the speedup from using stoichiometry amplification values of 2 and 4 and the SSA-CR algorithm. While the overall simulation times may not appear to be too significant, it should be emphasized that during genetic circuit design, one may need

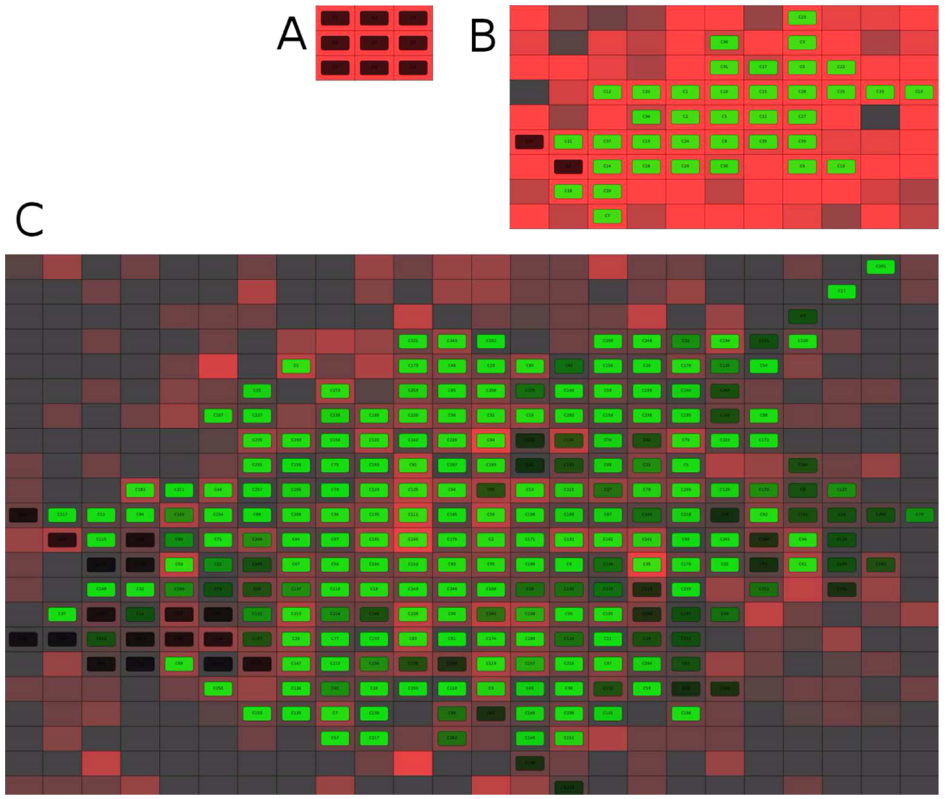


Figure 6. Three separate time points (a being first, c being last) in a dynamic simulation of the quorum trigger model shown in Figure 2 with an added GFP production reaction that is activated by the complex. The rectangular shapes on the grid represent cells, the population of which grows throughout the simulation. Extracellular 3OC6HSL has been associated with a red color scheme. Intracellular GFP is associated with a green color scheme. Red thus shows 30C6HSL diffusing outside of the cells, which turn green in a density-dependent manner. In this simulation, there is a significant amount of the environmental trigger molecule available, and the result is that the population becomes activated.

to perform many simulations for different circuit configurations and parameter sets. In this case, a speedup of 5 times over the SSA-direct is significant. Furthermore, the benefits are even more pronounced for simulations of dynamic models that can grow to be very large.

Visualization. iBioSim's visualization environment has been enhanced for both static and dynamic multicellular models. A multicellular model is displayed as a grid (see Figure 3), which can be animated over time using time series data from a simulation's output. Appearances (e.g., colors) can be attached to species counts of the model to allow for visualizing the model's behavior in a way that approximates fluorecent markers. Appearances can be associated with species both within compartments (which change the appearance of the compartment itself) and in the extracellular space to visualize diffusible species moving around the grid. For dynamic models, child compartments inherit the appearance scheme of the parent, and grid appearances are extended automatically as the grid expands, making it easy to visualize a population as it grows.

Figure 6 shows this process occurring over the time-course of a dynamic model with the quorum trigger. This figure and the movies in the Supporting Information show how a population of cells with a quorum trigger circuit behaves over time under different environmental conditions. This visual presentation aids the designer in determining if the dynamic response of the circuit achieves desired goals. Furthermore, statistics for the time-series data files can be generated for dynamic models, which can be used with the graphing functionality within iBioSim to further analyze the system, e.g., in aggregate over multiple simulation runs.

Discussion. The enhancements to iBioSim to support modeling, simulating, and visualizing dynamic cellular populations in a two-dimensional space improve our ability to explore design trade-offs in genetic circuits that must work within cellular populations. These enhancements include a new graphical interface for design, a spatial modeling framework, SBML annotations for dynamic process events, a new SSA-CR-based simulator with support for cellular population dynamics, and a visualization environment for analysis of simulation data. Possible future directions include modeling extensions to support a broader range of cellular behavior such as physical interactions, support for physics-based spatial modeling, simulation speedup via parallelization and GPU optimization, and creating a feedback loop between our software's development and collaboration with experimentalists.

The software space for CAD tools aimed at synthetic biologists is a new one, and a recent wave of tools for population-level design is now coming online.[13,14] iBioSim distinguishes itself from these other tools with its graphical interface, stochastic simulation algorithms, and support for the SBML and SBOL languages. We hope that our efforts are instructive to the community and help guide the development of this new generation of software tools for cellular population engineering.

dx.doi.org/10.1021/sb300082b | *ACS Synth. Biol.* 2013, 2, 223−229

METHODS

iBioSim itself and the simulation algorithms described are written in Java. The SSA Composition and Rejection algorithm is based on the algorithm described by Slepoy et al.,[25] with extensions. Simulations are run on a dual-core Celeron SU2300 with 1 GB of memory. The visualization framework is written using JGraphX,[26] an open-source Java graph visualization library. Additionally, the GNU Trove data structure library,[27] the Jafama fast math library,[28] and the Flanagan scientific and numerical library[29] are used. iBioSim is freely available from http://www.async.ece.utah.edu/iBioSim/.

ASSOCIATED CONTENT

Supporting Information

A movie demonstrating model creation in iBioSim, SBML models of the quorum trigger and a 2 × 2 grid of the quorum trigger submodel, and a timelapse movie of a dynamic quorum trigger grid model. This information is available free of charge via the Internet at http://pubs.acs.org.

AUTHOR INFORMATION

Corresponding Author

*E-mail: myers@ece.utah.edu.

Present Address

[†]Department of Bioengineering, University of Washington, Seattle, WA.

Author Contributions

C.J.M. and J.T.S. designed and tested the software. J.T.S. wrote the software. C.J.M. developed the example presented in this paper.

Notes

The authors declare no competing financial interest.

ACKNOWLEDGMENTS

Thanks to Tyler Patterson for developing the initial visualization engine, to Hiroyuki Kuwahara for coming up with stoichiometry amplification, and to Nicholas Roehner and Curtis Madsen for helpful and constructive discussion during the project's development. We would also like to thank, in addition to the aforementioned four, Nathan Barker, Kevin Jones, and Namphuon Nguyen for their previous work in developing iBioSim. This material is based upon work supported by the National Science Foundation under Grant No. CCF-0916042 and CCF-1218095. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

(1) Ruder, W. C., Lu, T., and Collins, J. J. (2011) Synthetic biology moving into the clinic. *Science 333*, 1248−1252.
(2) Basu, S., Gerchman, Y., Collins, C. H., Arnold, F. H., and Weiss, R. (2005) A synthetic multicellular system for programmed pattern formation. *Nature 434*, 1130−1134.
(3) Liu, C., Fu, X., Liu, L., Ren, X., Chau, C. K., Li, S., Xiang, L., Zeng, H., Chen, G., Tang, L. H., Lenz, P., Cui, X., Huang, W., Hwa, T., and Huang, J. D. (2011) Sequential establishment of stripe patterns in an expanding cell population. *Science 334*, 238−241.
(4) Tamsir, A., Tabor, J., and Voigt, C. (2011) Robust multicellular computing using genetically encoded NOR gates and chemical wires. *Nature 469*, 212−215.
(5) Cai, Y., Hartnett, B., Gustafsson, C., and Peccoud, J. (2007) A syntactic model to design and verify synthetic genetic constructs derived from standard biological parts. *Bioinformatics 23*, 2760−2767.
(6) Weeding, E., Houle, J., and Kaznessis, Y. N. (2010) SynBioSS designer: a web-based tool for the automated generation of kinetic models for synthetic biological constructs. *Briefings Bioinf. 11*, 394−402.
(7) Chandran, D., Bergmann, F. T., and Sauro, H. M. (2009) TinkerCell: modular CAD tool for synthetic biology. *J. Biol. Eng. 3*, 19.
(8) Hillson, N. J., Rosengarten, R. D., and Keasling, J. D. (2012) j5 DNA Assembly Design Automation Software. *ACS Synth. Biol. 1*, 14−21.
(9) Xia, B., Bhatia, S., Bubenheim, B., Dadgar, M., Densmore, D., and Anderson, J. C. (2011) in *Synthetic Biology, Part B Computer Aided Design and DNA Assembly* (Voigt, C., Ed.) Methods in Enzymology, Vol. *498*; pp 97−135, Academic Press, New York.
(10) Cickovski, T. M., Huang, C., Chaturvedi, R., Glimm, T., Hentschel, H. G., Alber, M. S., Glazier, J. A., Newman, S. A., and Izaguirre, J. A. (2005) A framework for three-dimensional simulation of morphogenesis. *IEEE/ACM Trans. Comput. Biol. Bioinf. 2*, 273−288.
(11) Meir, E., Munro, E. M., Odell, G. M., and Von Dassow, G. (2002) Ingeneue: a versatile tool for reconstituting genetic networks, with examples from the segment polarity network. *J. Exp. Zool. 294*, 216−251.
(12) Le Novere, N., and Shimizu, T. S. (2001) STOCHSIM: modelling of stochastic biomolecular processes. *Bioinformatics 17*, 575−576.
(13) Jang, S. S., Oishi, K. T., Egbert, R. G., and Klavins, E. (2012) Specification and simulation of synthetic multicelled behaviors. *ACS Synth. Biol. 1*, 365−374.
(14) Rudge, T. J., Steiner, P. J., Phillips, A., and Haseloff, J. (2012) Computational modeling of synthetic microbial biofilms. *ACS Synth. Biol. 1*, 345−352.
(15) Myers, C. J., Barker, N., Jones, K., Kuwahara, H., Madsen, C., and Nguyen, N.-P. D. (2009) iBioSim: a tool for the analysis and design of genetic circuits. *Bioinformatics 25*, 2848−2849.
(16) Madsen, C., Myers, C., Patterson, T., Roehner, N., Stevens, J., and Winstead, C. (2012) Design and test of genetic circuits using iBioSim. *IEEE Design Test Comput. 29* (3), 32−39.
(17) Hucka, M., et al. (2003) The Systems Biology Markup Language (SBML): A medium for representation and exchange of biochemical network models. *Bioinformatics 19*, 524−531.
(18) Galdzicki, M. et al. (2011) Synthetic Biology Open Language (SBOL) Version 1.0.0., DOI: 1721.1/66172.
(19) Kuwahara, H., Myers, C., Barker, N., Samoilov, M., and Arkin, A. (2006) Automated abstraction methodology for genetic regulatory networks. *Trans. Comp. Syst. Biol. VI*, 150−175.
(20) Kuwahara, H. (2007) Model abstraction and temporal behavior analysis of genetic regulatory networks. Ph.D. thesis, University of Utah.
(21) Kuwahara, H., Myers, C., and Samoilov, M. (2010) Temperature control of fimbriation circuit switch in uropathogenic Escherichia coli: quantitative analysis via automated model abstraction. *PLoS Comput. Biol. 6*, e1000723.
(22) Kuwahara, H., Madsen, C., Mura, I., Myers, C., Tejeda, A., and Winstead, C. (2010) in *Stochastic Control* (Myers, C., Ed.) Sciyo, http://sciyo.com/.
(23) Madsen, C., Myers, C., Roehner, N., Winstead, C., and Zhang, Z. (2012) Utilizing stochastic model checking to analyze genetic circuits, in *2012 Computational Intelligence in Bioinformatics and Computational Biology*; IEEE, New York.
(24) Nguyen, N., Myers, C., Kuwahara, H., Winstead, C., and Keener, J. (2010) Design and analysis of a robust genetic Muller C-element. *J. Theor. Biol. 264*, 174−187.
(25) Slepoy, A., Thompson, A. P., and Plimpton, S. J. (2008) A constant-time kinetic Monte Carlo algorithm for simulation of large biochemical reaction networks. *J. Chem. Phys. 128*, 205101.
(26) JGraphX. http://www.jgraph.com/jgraphdownload.html.
(27) GNU Trove. http://sourceforge.net/projects/trove4j/.

(28) Jafama. http://sourceforge.net/projects/jafama/.
(29) Flanagan, M. T. Michael Thomas Flanagan's Java Scientific Library. http://www.ee.ucl.ac.uk/mflanaga/java/.