Workflows for Parameter Studies of Multi-Cell Modeling

Randy Heiland
Open Systems Laboratory
Indiana University
Bloomington IN 47405, USA

Maciek Swat
Biocomplexity Institute and
Department of Physics
Indiana University
Bloomington IN 47405, USA

Benjamin Zaitlen
Biocomplexity Institute and
Department of Physics
Indiana University
Bloomington IN 47405, USA

James Glazier
Biocomplexity Institute and
Department of Physics
Indiana University
Bloomington IN 47405, USA

Andrew Lumsdaine
Open Systems Laboratory
Indiana University
Bloomington IN 47405, USA

Keywords: Scientific workflows, scripting languages, cluster computing, visualization.

Abstract

Running simulations for multi-cell tissue models can involve numerous parameters and consume considerable computing resources. This paper presents an overview and use case of two open source projects - CompuCell3D, a multi-cell modeling framework, and VisTrails, a workflow system for parameter exploration and data management.

1. INTRODUCTION

Mathematical modeling and computer simulations continue to contribute to our understanding of a variety of complex phenomena. In this paper we are interested in processes involved in developmental biology. More specifically, we wish to model the dynamics of cells, cell clusters and tissues. There exist a variety of data structures and techniques for this particular domain of modeling, e.g., agents, regular lattices, unstructured meshes, in conjunction with finite-state machines (cellular automata) or PDEs solved via finite differences or finite elements.

We present an overview of the CompuCell3D modeling framework and explore the use of a workflow system, VisTrails, for the primary purpose of performing parameter explorations for CompuCell3D. Both packages are open source software.

Although there are several workflow systems that are now available, we shall see that VisTrails is an obvious choice for our particular situation. Scientific visualization plays a key role when developing a model that simulates multi-cell dynamics. VisTrails, as the name implies, comes bundled with visualization packages. In fact, both CompuCell3D and VisTrails have adopted the Visualization Toolkit (VTK, www.vtk.org) as one option for rendering data. Another reason that VisTrails was an obvious choice is the fact that it is built using the Python scripting language (www.python.org). CompuCell3D also uses Python as an interface to its underlying C++ code. Python is quite popular for developing interactive scientific applications [1].

2. COMPUCELL3D

CompuCell3D (www.compucell3d.org) is a multi-cell modeling framework [2]. It implements the Glazier-Graner-Hogeweg (GGH) model, also sometimes known as the cellular Potts model, and operates on a regular lattice (currently either square or hexagonal, in 2D and 3D). As a general-purpose framework, CompuCell3D is capable of modeling a broad range of phenomena, e.g., ferromagnetism, foams, and biological cells [3][4]. One noteworthy feature of CompuCell3D is that its most primitive object is a *generalized cell*. That is to say, it is not (typically) concerned with mechanisms at the sub-cellular (biological) level, e.g. reaction kinetic networks. And this feature is relevant for this study since it makes the parameter space more tractable.

CompuCell3D can be invoked as either an interactive application with a graphical user interface or a batch program that outputs data at user-specified intervals. In either case, a user creates an XML-formatted model-definition input file and then, in the interactive session, runs the simulation, watching the evolution of the generalized cells. In addition, it is also possible to visualize any underlying fields defined on the lattice, e.g., diffusion.

For a use case scenario, we choose one specific model, biological cell sorting [5] (in 2D), for which we will conduct parameter explorations. The general approach that we develop for these workflows can be applied to other models. Cell sorting is a well-known biological process and can be described as the reorganization of a random mix of two cell types. The cell types differ in their cell adhesivities (stickiness). One canonical outcome of a cell sorting simulation is that cell types with lower adhesivity

(Noncondensing) will engulf cell types with higher adhesivity (Condensing). However, as we shall see, by exploring a range of parameters associated with the model, we can obtain a wide variety of outcomes. Figure 1 depicts the CompuCell3D application displaying a snapshot of the time evolution for the 2D cell sorting model, showing Condensing cells (green), Noncondensing cells (red), and the background Medium (blue).

One goal in performing a workflow evaluation is to minimize the changes needed to run a simulation. Therefore, we wish to keep the basic approach of using an XML model-definition file as input. Our strategy will be to create workflows that automate the parameter exploration, edit the relevant XML values, execute the core CompuCell3D code and output raw data. The visualization of the data will be performed by VisTrails, in addition to managing the workflow data. For this initial study, we will only output simulation data at the end of some specified time.

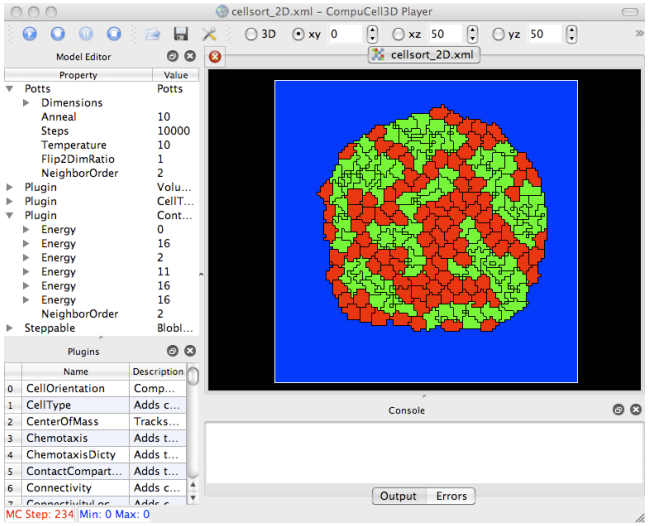

Figure 1. CompuCell3D application showing cell sorting

3. VISTRAILS

VisTrails is a scientific workflow and provenance management system [6]. Workflows are expressed as dataflows (pipelines) – a set of modules, with links connecting input and output ports, that get executed, in the simplest case, from top to bottom. In the most recent version (1.3, used here), it also provides constructs for functional loops and conditionals. As a standalone application, VisTrails provides an interactive Builder window in which one constructs a pipeline (via drag and drop), sets parameters and executes it. Figure 2 illustrates a very simple pipeline that performs an arithmetic binary operation and outputs the result to a console panel. On the left side of the Builder window, we have categories of built-in modules from which to drag and drop. The primary center panel is used to graphically build the pipeline and the smaller picture-in-picture panel in the upper-right is the (graph) history associated with this pipeline. Provenance data associated with a workflow (a *vistrail*) is maintained via XML files or a relational database. As mentioned before, VisTrails is capable of performing visualizations. These appear in a separate Spreadsheet window which is also interactive. The built-in visualization packages include VTK, matplotlib and Image Magick. VisTrails is written in Python, is open source and cross-platform.

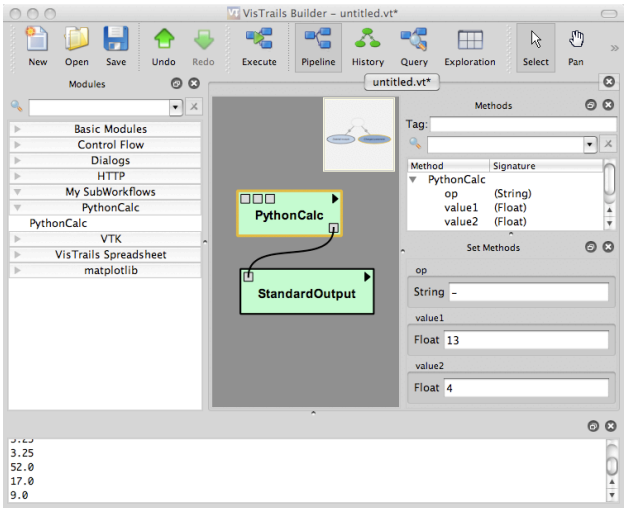

Figure 2. VisTrails Builder window and simple workflow

4. PARAMETER SWEEPS

Our primary goal in using a workflow system is to perform parameter sweeps for CompuCell3D models. We begin by taking a manual approach in VisTrails, illustrated in Figures 3-5: grab three Float modules, rename them to *p1,p2,p3* and connect them to another module, *edit_xml* which is a renamed instance of a general-purpose *PythonSource* module that can contain any user-specified Python script. In this case, it will contain the necessary script to edit the input XML model-definition file, replacing three chosen parameters with the values in *p1-p3*. It is quite easy to create input and output ports for a module (Figure 4). The ports will appear as small squares at the top (input) and bottom (output) of a module. When a user tries to connect an output port of one module to an input port of another, VisTrails will enforce datatype matching.

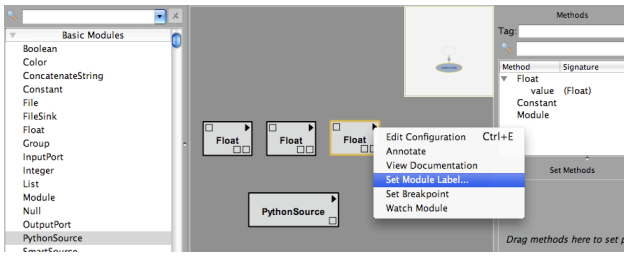

Figure 3. Drag & drop modules and renaming

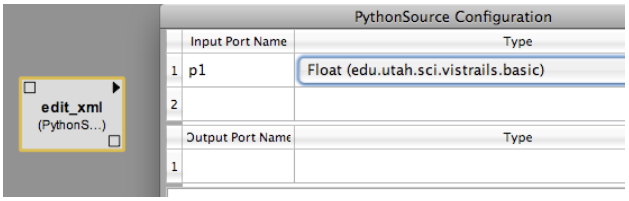

Figure 4. Defining input and output ports

Next, we create another *PythonSource* module, *run_cc3d*, that simply executes CompuCell3D with the edited XML file as input. This writes output data to a file, which is then processed by the MplPlot module (uses the matplotlib package) and is then rendered in the Spreadsheet. Figure 6 shows the results of two qualitatively different cell sorting results after a user has tediously entered a variety of values for *p1-p3* and manually executed the workflow.

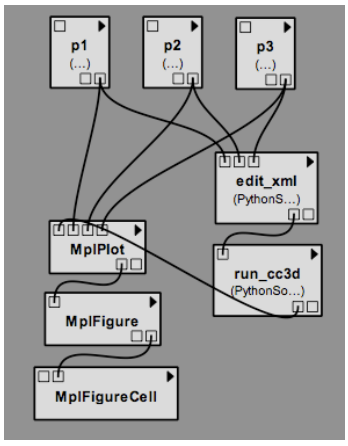

Figure 5. Pipeline to manually set parameters

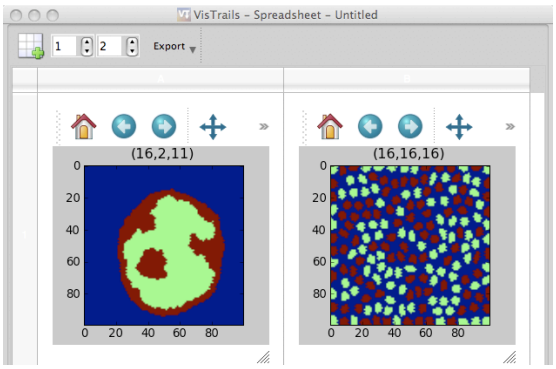

Figure 6. Spreadsheet results for 2 sets of parameters

The first improvement we make to our workflow is to use the *Cross* module (in the Control Flow category). This module will take the cross-product of two lists, offering a convenient method for performing parameter sweeps. By connecting multiple *Cross* modules together, we can easily build up sets of parameter values (Figure 7).

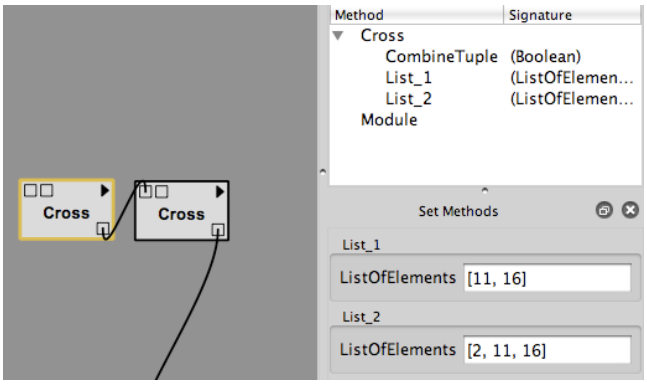

Figure 7. Combining Cross-product modules for parameters

The next improvement we make to the workflow is to use the *Map* module (also in the Control Flow category). This module will apply a generic function to a given input list, resulting in a sequence of results. Figure 8 depicts usage of the Map module. The generic function associated with the Map is a Group module that was created by graphically grouping all modules in Figure 5, except *p1-p3*. Rather than having the user manually supply parameter values as done previously, the workflow now generates a list of (5-tuple) parameters from combined Cross modules that will be inserted into the appropriate XML parameters.

At this point, the workflow has become quite computationally (and data) intensive. For the cell sorting model system being studied, our sample workflow generates 72 sets of parameters that affect the biological cells' (and Medium's) adhesivities.

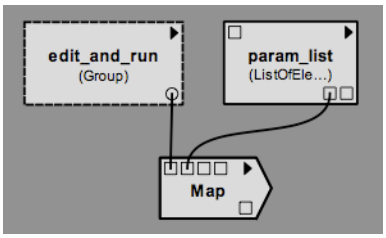

Figure 8. The Map module

To address the computational bottleneck, we modify our workflow so that the CompuCell3D execution is performed on a remote cluster as described in previous work [7]. Using Indiana University's Big Red cluster (a TeraGrid resource), together with Globus clients (*globus-job-run* and *globus-url-copy*) and the MyProxy service, we split the original workflow into two. The first automates the parameter sweep and job submissions to Big Red. The workflow submits a series of, in this case, 72 jobs that can be run simultaneously. Data files from the simulations are written to GPFS. Once the simulations have completed, the user executes a second workflow that automates retrieval of the data files and rendering into the VisTrails Spreadsheet. Figure 9 shows a portion of the 72 rendered results, obtained via the Export (an image) functionality of the Spreadsheet. These results reveal the two expected qualitative outcomes that were depicted in Figure 6, engulfment and dissociation. But they also reveal a third, checkerboard, pattern and variations of these three, including a checkerboard pattern where Condensing cells form the border (i.e., stick better to the Medium) and one where Noncondensing cells form the border.

At this point, we should state the obvious. The rendered results do not represent steady state solutions. We only know that these results represent a single solution at the same time T. Nevertheless, we are hopeful that such a study can offer some insight into model parameters being mapped into qualitative outcomes.

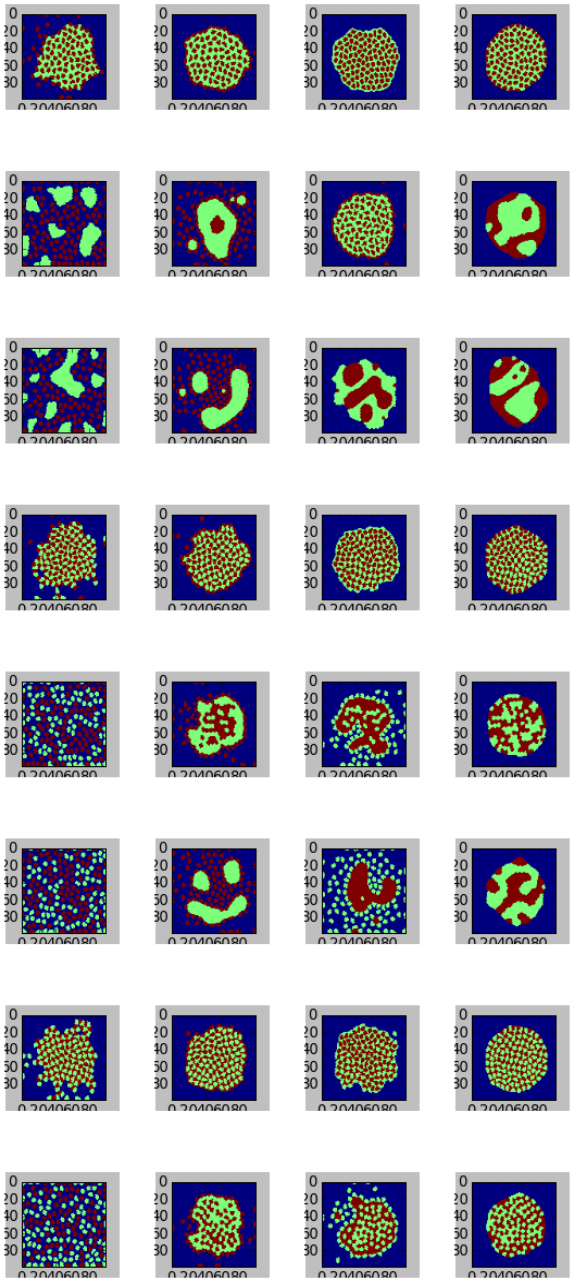

Figure 9. Exported image of Spreadsheet in Interactive mode

The VisTrails Spreadsheet window offers two modes, Interactive and Editing. Until now, we have only used the Interactive mode, which allows for graphical interaction within each rendered image (or geometry, if we had used VisTrails VTK modules, for example). In Figure 10, we show the first two rows of the Spreadsheet in Editing mode. This mode makes it possible to rearrange the layout of the

rendered viewports and also provides a mapping to the underlying workflow (vistrail). In Figure 11, we show the top two rows of the Spreadsheet after rearranging so that distinctly different qualitative results are displayed.

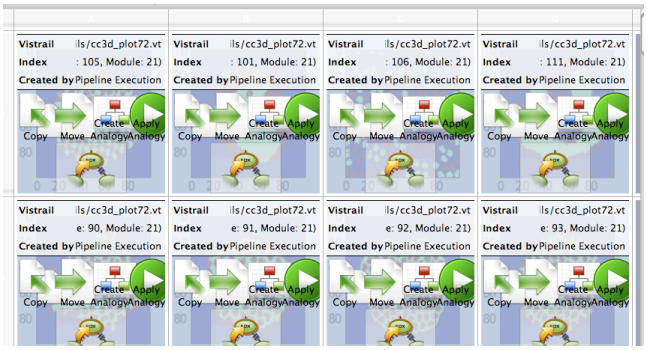

Figure 10. Spreadsheet in Editing mode

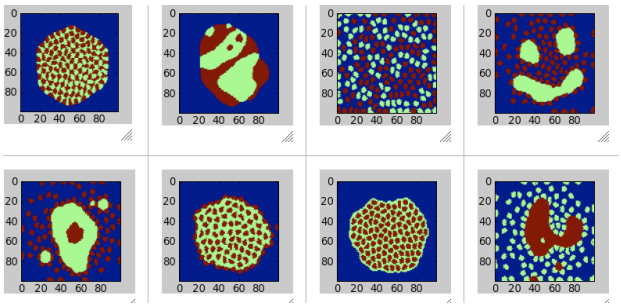

Figure 11. First two rows of Spreadsheet after rearrangement

5. FUTURE WORK

There are many different possibilities that one could explore in the future. The basic workflow approach used here was to programmatically edit an XML input file, run a simulation until time T and write data, thereby capturing a single solution. The next logical step might be to extend this to capture solutions at multiple time steps and generate movies for the parameter sweeps.

A more challenging task would be to devise techniques for classifying the qualitative outcomes – a taxonomy of solutions. If such a metric were possible, it might allow for programmatic workflow steering.

We have glossed over many details of using CompuCell3D as a general-purpose modeling framework. But a key idea is to select predefined *plugins* that map to specific behaviors of cells. One very significant challenge is to find appropriate values for these plugins' parameters that will result in the desired dynamics and structure. Whether or not we can incorporate VisTrails parameter sweeps within the system remains to be seen, but it would seem to be a worthy goal.

Faster, larger, and longer computations are always desirable. We are currently exploring options to address some of these challenges, including parallelizing CompuCell3D using MPI and looking at GPUs for speeding up parts of the code.

6. CONCLUSIONS

We have performed a parameter exploration for biological cell sorting, using CompuCell3D to perform the simulations and VisTrails to create and maintain a workflow. VisTrails was a natural choice for a workflow package since it is Python-based and CompuCell3D also makes extensive use of Python.

Both systems, CompuCell3D and VisTrails, are open source software (as is Python). A user can freely download and inspect all of the underlying code. This helps eliminate algorithm uncertainty that may be associated with closed source software. A user can, for example, see how a PDE solver is implemented in CompuCell3D and, moreover, insert their own solver into their copy of the code (and perhaps make it available to the community).

VisTrails is an easy to use workflow system. It lets a user graphically construct a pipeline of connected modules that can then be executed. Many useful modules come predefined in the system, including a general-purpose *PythonSource* that lets a user create their own Python script. Modules are user-configurable, allowing for the creation of (additional) input and output ports, easy selection of predefined datatypes on those ports, and subsequent type-checking when connecting one port to another. Two control flow modules were especially useful for our parameter exploration, one to generate a cross-product of parameters and another to map a generic function to a list of input values. Finally, it was critical that VisTrails provided visualization functionality. Fortunately, the bundled visualization packages were some with which we were already familiar. It was very exciting to see the variety of solutions, from our parameter sweep, appear in the Spreadsheet. Clearly, it is advantageous to be familiar with Python when using VisTrails. However, even if this is not the case, one can still construct useful workflows quite easily.

We have not truly compared CompuCell3D with other multi-cell modeling packages. The greatest obstacle to doing so has been the lack of having a standard modeling, e.g. markup, language, which in turn relies on having an underlying ontology. These are areas we have recently begun investigating.

7. ACKNOWLEDGEMENTS

We would like to thank the VisTrails developers for user support and the Research Technologies Division at IU for user support with Big Red.

This work was sponsored by National Institutes of Health, National Institute of General Medical Sciences, grant 1R01 GM076692-01 and the Biocomplexity Institute at Indiana University. The Open Systems Lab, as part of the Pervasive Technology Institute, gratefully acknowledges the generous support of the Lilly Endowment, Inc.

References

[1] Dubois, P. F. 2007. Python for Scientific Computing, Computing in Science & Engineering, vol. 9, no. 3, May/June 2007.

[2] Cickovski, T., Aras, K., Swat, M., Merks, R. M. H., Glimm, T., Hentschel, H. G. E., Alber, M. S., Glazier, J. A., Newman, S. A., and Izaguirre, J. A. 2007. From Genes to Organisms Via the Cell: A Problem-Solving Environment for Multicellular Development. Computing in Science and Engineering, 9: 50-60 (2007).

[3] Glazier, J. A., Balter, A., and Poplawski, N. J. 2007. Magnetization to Morphogenesis: A Brief History of the Glazier-Graner-Hogeweg Model. In Single-Cell-Based Models in Biology and Medicine, A. R. A. Anderson, M. A. J. Chaplain, and K. A. Rejniak, Ed. Birkhäuser, Basel, Boston and Berlin, 79-106.

[4] Swat, M. H., Hester, S. D., Balter, A. I., Heiland, R. W., Zaitlen, B. L., and Glazier, J. A. 2009. Multicell simulations of development and disease using the CompuCell3D simulation environment. In Systems Biology, I. V. Maly, Ed. volume 500 of Methods in Molecular Biology, pages 361--428. Humana Press, Clifton, N.J.

[5] Graner, F. and Glazier, J. A. 1992. Simulation of Biological Cell Sorting Using a Two-Dimensional Extended Potts Model. Physical Review Letters 69, 2013-2016 (1992).

[6] Callahan, S. P., Freire, J., Santos, E., Scheidegger, C. E., Silva, C. T., and Vo, H. T. 2006. VisTrails: visualization meets data management. In Proceedings of the 2006 ACM SIGMOD international Conference on Management of Data (Chicago, IL, USA, June 27 - 29, 2006). SIGMOD '06. ACM, New York, NY, 745-747. DOI= http://doi.acm.org/10.1145/1142473.1142574

[7] Heiland R., Mooney, S. D., Boverhof, J., Jackson, K., Swat, M., Balter, A., Christie, M., and Insley, J. 2007. Python for Scientific Gateways Development. In Proceedings of the International Workshop on Grid Computing Environments, Reno, NV, November 2007.

Biography

Randy Heiland is a Research Scientist in the Open Systems Lab at Indiana University. He received his M.S. in Computer Science at the University of Utah and his M.A. in Mathematics at Arizona State University.

Maciek Swat is a Research Scientist in the Biocomplexity Institute at Indiana University. He received his Ph.D. in Physics at Indiana University.

Benjamin Zaitlen is a Research Scientist in the Biocomplexity Institute at Indiana University. He received his M.S. in Physics at UC-Santa Cruz.

James Glazier is a Professor in the Department of Physics and Director of the Biocomplexity Institute at Indiana University. He received his Ph.D. in Physics at the University of Chicago.

Andrew Lumsdaine is a Professor in the School of Informatics and Computing and Director of the Open Systems Lab at Indiana University. He received his Ph.D. in Electrical Engineering and Computer Science at MIT.