A Scalable and Integrative System for Pathway Bioinformatics and Systems Biology

B. Compani [a,b], T. Su [a,b‡] , I. Chang [a,b,d], J. Cheng [a,b,g], K. Shah [a,c], T. Whisenant [a,c], Y. Dou [a,b], A. Bergmann [e], R. Cheong [e],L. Bardwell [a,c], A. Levchenko [e], B. Wold [a,f], P. Baldi [a,b], E. Mjolsness [a,b§]

1 Motivation:

Progress in systems biology critically depends on developing scalable informatics tools to predictively model, visualize, and flexibly store information about these complex biological systems. Scalability of these tools, as well as their ability to integrate within larger frameworks of evolving tools, is critical to address the multi-scale and size complexity of biological systems.

2 Results:

Here we describe a generative, scalable software infrastructure for pathway bioinformatics and systems biology. The Sigmoid modeling system is a three-tier architecture comprising distributed modules that implement pathway/cell model generation and simulation (xCellerator/Mathematica), a pathway modeling database (Sigmoid proper), a Web service-oriented middleware, a graphical user interface, and in the future, parameter optimization and other datamining technologies. Key to the design of the infrastructure is its scalability ensured by leveraging symbolic computer alge-

‡ B. Compani and T. Su share primary authorship

§ to whom correspondence should be addressed

[a]Institute for Genomics and Bioinformatics, University of California, Irvine, Irvine CA 92697, [b]School of Information and Computer Sciences, University of California, Irvine, Irvine CA 92697, [c]School of Biological Sciences, University of California, Irvine, Irvine CA 92697, [d]School of Engineering, University of California, Irvine, Irvine CA 92697, [e]Department of Biomedical Engineering, Johns Hopkins University, Baltimore MD 21218, [f]Division of Biology, California Institute of Technology, Pasadena CA 91125, [g]Department of Computer Science, University of Missouri-Columbia, Columbia, MO 65211-2060(current affiliation).

bra and self-generation of database and other code from high-level representations such as an UML schema.

3 Availability:

All sigmoid modeling software components and supplementary information are available through: `http://www.igb.uci.edu/servers/sb.html`.

4 Contact:

emj@ics.uci.edu,bardwell@uci.edu

5 Introduction

Although there are many kinds and levels of biological systems, such as immune systems, nervous systems, and ecosystems, the expression "systems biology" is used today mostly to describe attempts at unraveling molecular systems, above the traditional level of single genes and single proteins, focusing on the level of pathways and groups of pathways in a cell. Here we describe Sigmoid; a generative, scalable software infrastructure for systems biology to facilitate global modeling of biological systems. If deciphered as an acronym, SIGMOID would translate to; SIGnal MOdeling Interface and Database. Here the term Signal, in a biological sense, would be broadly interpreted. Sigmoid supports the process of cycling between model building, hypothesis generation, and biological experimentation and data gathering, by integrating the hypothesis and discovery phases in the research process.

Scalability of the software architecture is an essential and pervasive requirement given the underlying complexity of biological systems brought on by evolutionary tinkering and a large number of components and modules operating at multiple spatial and temporal scales. The scalability must be reflected in each component of the infrastructure. In Sigmoid, we address the problem of creating a scalable expert assistance system for modeling biological pathways, using current software technology to decrease the difficulty and cost of creating the system. The reason for building such a system is to provide computational support to biologists and computational scientists who need to create and explore predictive dynamical models of complex biological systems such as metabolic, gene regulation, or signal transduction pathways in living cells [Cheng *et al.*2005]. While the primary focus of the infrastructure is reverse engineering biological circuits, in the long-run we expect it to become applicable also to synthetic biology projects, that is for the more or less

de novo design of complex sets of molecular interactions with a particular computational, biomedical, or bio-synthetic focus.

5.1 Overview of the Software Infrastructure

The Sigmoid modeling system consists of (1) distributed modules implementing pathway/cell model generation and simulation (Cellerator; [Shapiro *et al.*2003]), (2) a pathway modeling database, (3) a Web service-oriented middleware, (4) a world wide web model browser, (4) a graphical user interface friendly to a biologist user, and (5) in the future, parameter optimization and other datamining technologies. These modules are organized in a classical three-tier architecture (Figure 1). The back-end currently consists of the database, the simulator, and other model manipulators. The GUI front-end does not access the back-end modules directly but rather via a Web service middleware module. The extra development overhead introduced by the middle layer is more than compensated by the advantages in terms of distributed computing, performance, flexibility, and scalability. With the exception of rapid model retrieval, the middleware layer brokers all communications between the GUI and the back-end components and also between the backend components themselves. We have found that storing binary instances of models in a database cache can provide significant improvements in model retrieval times in comparison to full model reconstruction and retrieval through the middleware layer. In the event that the rapid model retrieval interface is not accessible, the system will shift access to the database through the middleware. This infrastructure was created in a close collaboration between bioinformaticians and biologists by having the design of many of the essential software objects and their relationships be visible as implementation proceeded.

We have coordinated the development of various software modules in Sigmoid by using the Universal Modeling Language (UML) to diagram the most important biological objects– notably reactions and molecular reactants. This UML diagram is used as a template to automatically generate several parts of Sigmoid, in particular a realization of the Sigmoid pathway modeling database (in SQL) and the corresponding Java object hierarchy along with support files for facilitating the object-relational mapping and end-user documentation. Also the Graphical User Interface (GUI) relies heavily on the Java reflection utility to automatically discover much of what it needs to know about the Sigmoid schema. Thus there is a guarantee that the software actually implements something very close to the UML construction of biological objects and, coding time for different modules of the system is reduced.

To keep the infrastructure flexible and manageable as it grows, we have resorted to a "generative" approach, that seeks to partially automate the generation of both executable code and mathematical models. We have applied this approach to as many of the modules in Figure 1 as possible, starting from high-level inputs such as UML diagrams and reaction notations understandable to non-computer scientists.

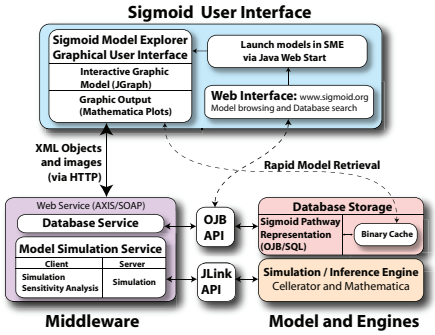

Fig. 1 Sigmoid three-tier architecture. Separation of modules into a communicating distributed system increases scalability of the architecture. Our simulator is the xCellerator model generator/simulator; the database is Sigmoid (autogenerated from UML schema in Figure **??**); user interface is the Sigmoid Model Explorer (SME).

We will now briefly describe the various components of this generative infrastructure, its main features, requirements, and current state of development. While we are developing the various components of this architecture together, it is important to notice that some components are more mature than others and that individual components which are more mature, such as the database or the simulator, are self-sufficient and can be used independently of the GUI or the middleware.

In overview, the main software, languages, and tools that are used in the architecture include:

- Front-end GUI: Java, Java reflection, JGraph, HTML, JavaScript, XML, WSDL, CSS, SVG, Java Webstart, Web browser;
- Middleware: Java, Apache, AXIS/SOAP, Java Servlet, JSP, XML, Apache Webserver, Tomcat, OJB, JLink;
- Back-end solver: JLink, Mathematica, xCellerator, Cellzilla, SBML;
- Back-end database: UML, AXgen, PostgreSQL, OJB, XML, VTL and Linux. We use publicly available open source, tools as much as possible. Sigmoid software components are available through: `www.igb.uci.edu/servers/sb.html`.

6 Methods

6.1 Model Generation and Simulation: xCellerator

Simulating a biological pathway often involves simulating dozens if not hundreds or thousands of elementary chemical reactions. Regardless of the details of the equations (typically differential equations) used to model an individual reaction, building a model containing a large numbers of reactions is a tedious and error-prone pro-

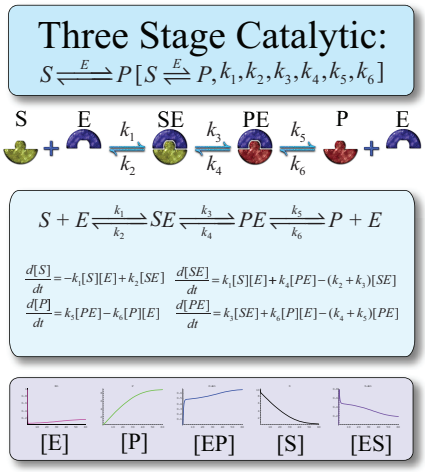

Fig. 2 Sigmoid Three Stage Catalytic model. From Top to bottom. xCellerator input notation, reaction cartoon, resulting differential equations and an example of numerical output.

cess if to be performed more or less manually. Note that unlike electronic circuits, such as those found in a computer, and comprising only a small number of elementary building blocks, chemical reactions in biology come in a large variety of elementary forms. What is needed therefore is to build a library of re-usable reaction models that can be expressed in a simple, higher-level language, specifying the molecular species and the type of reaction. For example, one can use syntax similar to "$A + B \rightarrow C$; mass action with rate k" to specify that molecular species A interacts with molecular species B to produce molecular species C according to the mass action kinetic law expressed by the differential equation $dC/dt = kAB$, whereby the rate of production of C is proportional to the product of the concentration of the reactants A and B. The primary problem is not a problem of numerical analysis: there are several packages that can be used to solve fairly large systems of such equations. The primary problem is a problem of model management and scalability. This problem is best addressed by using a symbolic mathematical language and numerical solver, such as Mathematica, based on computer- algebra objects and a rich set of well-implemented mathematical operations. Indeed,

Cellerator [Shapiro *et al.*2003] is implemented as a Mathematica notebook and is designed to facilitate biological modeling via automated equation generation. Sigmoid now supports xCellerator [B. Shapiro2007], the most recent version of Cellerator.

Many models of molecular interactions have been implemented in xCellerator using different formalisms, such as differential equations or stochastic molecular simulation formalism and ranging from the law of mass action and simple Michaelis-Menten models to more complex models of enzyme reactions (e.g. the Monod-Wyman-Changeaux or MWC model for allosteric enzymes [Najdi *et al.*2005]) and gene regulation [Segel1992]. The list of reaction models continues to expand along

with the library of actual pathway models comprising sets of coordinated reactions with parameters derived from the literature whenever possible. In addition, an extended set of enzyme mechanism models for single and multi-substrate, positively and negatively regulated and allosteric enzymes, called kMech, has been written for xCellerator and continues to develop[Yang *et al.*2005b]. Sigmoid currently supports all the available xCellerator and kMech reaction models. To illustrate xCellerator utility, consider the example of a three stage catalytic model. This reaction is a composite representation of 3 reversible reactions; substrate-enzyme complex formation, the conversion of the substrate to product within the complex and, subsequent disassociation of the enzyme-product complex into free enzyme and product. When presented with the correct input notation, xCellerator will translate the symbolic reaction to differential equations. The resulting differential equations and variable definitions are passed to Mathematica where they are solved by the numeric solver function (NDSolve) and time plots are generated. See example in Figure 2. The parameters for this enzyme mechanism are stored in the Sigmoid Pathways Database. In short, xCellerator converts symbolic reactions to mathematical equations, and solves the corresponding equations.

6.2 Sigmoid Pathway Database

The pathway model database is defined by a UML schema, Comprehensive UML class diagrams of the Sigmoid Schema can be found in the supplementary materials. The schema is organized into 4 main diagrams. The first diagram consists of the various top level container classes such as the Model Class and the Gene Ontology source class. The first diagram also contains the parameter set hierarchy, classes for graphical layout in SME and various classes to handle units and measures. The three remaining diagrams consist respectively of three major class hierarchies: Reactions, Reactants and Knowledge Sources. Reactions utilize Reactants for their products, substrates, and enzymes, Models are composed of parameterized Reactions, and these three class hierarchies utilize Knowledge Sources in order to reference external information about themselves. While initial versions of the Sigmoid database were implemented by hand, we wished to automatically transform the class descriptions contained in the high-level UML diagram of this hierarchy into a set of instantiable objects upon which applications may be built. Our current approach to the process of auto-generating software components from a master UML diagram relies on the capabilities of several existing open-source projects. These pre-existing projects remove much of the core software development responsibilities and allow us to focus on tying them together to produce the specific software products needed for our own use.

The Sigmoid database is no longer hand-coded. It consists of autogenerated, functionally equivalent code. Object-relational database code autogeneration from UML is itself a contribution of potentially general interest in database software en-

gineering. The current version of Sigmoid is implemented using PostgreSQL the main OpenSource database software.

In more detail, we currently use the AXgen (`http://axgen.sourceforge.net/`) open-source tool for reading UML diagrams and providing an API to access the diagram's structure. AXgen provides interfaces to both the Novosoft UML library (nsuml) as well as the NetBeans MDR library. This allows us to use one tool to read a much wider variety of UML than we would be able to otherwise. The AXgen API also provides many convenience functions for the process of auto-generating code from the UML. Once a UML diagram is loaded, a set of Java classes are generated for each corresponding UML class. As a spin-off we submitted new UML-interpretation features to the AXgen project to support field multiplicity as well as general code base improvements.

The actual process of generating the various classes is simplified by leveraging the Apache Velocity project and its associated Velocity Template Language (VTL). VTL allows one to create templates that interact with live Java code. In addition to the Java object class hierarchy, the auto-generation framework is also responsible for generating any auxiliary files. In the current implementation, this encompasses the creation of SQL files which create a database for the schema defined in the master diagram as well as a mapping (using the open-source OJB XML-based object relational bridge (`http://db.apache.org/ojb/`)) from the generated Java classes to the database. In the future, we may also be able to auto-generate UI widgets for each class.

An essential function of Sigmoid is to assist in the translation of biological knowledge into mathematical form. The representation of Reactions in Sigmoid is aimed at this goal. Sigmoid Reactions represent biochemical processes that transform molecular or other biological objects. These objects are in turn represented as Sigmoid Reactants. A major design feature of Sigmoid is that, to support translation of biology to mathematics, Reactions are defined in two ways: biologically, as Biological Reaction representations of various types, and mathematically, as Mathematical Reactions that constitute composable mathematical models. Because of the diversity of biochemical processes, there is an entire hierarchy of Biological Reaction types. Correspondingly there is a hierarchy of Mathematical Reaction models. This way the Sigmoid architecture can offer explicit support for the translation of biological processes into mathematical process models. Each type of biological reaction may in principle be translated into several alternative mathematical reaction models, and each mathematical reaction model can serve as the translation of several different biological reactions. Sigmoid will present consistent alternatives for each required translation from biology to mathematics.

The two reaction hierarchies can be differentiated and related as follows. First, the Biological Reaction hierarchy is intended to provide biologically oriented users with symbolic representations of a biochemical reaction or process. These representations include attributes that represent the basic reactants that participate in the reaction, but they do not specify the actual mechanics or rate law of the reaction. The primary function, along with participant roles (i.e. substrate, product, enzyme modifier) of each reactant in a given reaction are represented in a Biological Reaction

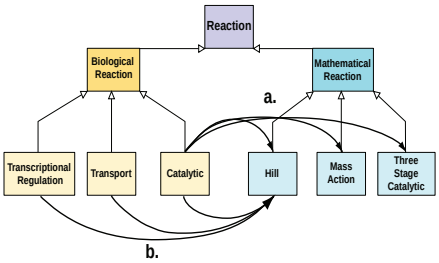

Fig. 3 Simplified version of the Sigmoid Schema Reaction hierarchy. (a.) There may exist a one to many relation between a particular biological reaction and potential functions (Mathematical-Reactions) that may be assigned to model the kinetics of the interaction. For instance numerous mathematical functions can be assigned to model a catalytic process. (b.) In reverse, the functional application of a particular set of differential equations may be conserved over a variety of biological phenomena so, there also may be a one to many association between a particular mathematical function (Reaction) and the biological scenarios it may be applied to. For instance a hill equation may provide useful in modeling a catalytic reaction, transcriptional regulation or even a transport process.

class as attributes. Second,the Mathematical Reactions constitute a type hierarchy of mathematical models of reactions or other processes in the Sigmoid schema. Such representations include particular rate laws, as well as the translation of compound reactions into a subnetwork of more elementary reactions each of which has a more elementary mathematical model. Most Mathematical Reactions currently have direct xCellerator/kMech implementation functions associated with them. Numerical parameters associated with each reaction are contained by reference, which enables key reaction parameters to be shared within a MathematicalReaction or across a full reaction network.

An example of the importance of many-to-many reaction translations is shown in Figure 3. A simplified fragment of the Sigmoid reaction hierarchy is shown. A catalytic Biological Reaction can be translated (a) into a Mass Action reaction, into the simpler (Michaelis-Menten like) approximation of a Hill function kinetics, or into the more detailed three-stage catalytic reaction. On the other hand a Hill function mathematical reaction could be the result of translating a catalytic reaction, a transport process, or a transcriptional regulation reaction.

6.3 Sigmoid Web Middleware for Distributed Computing and Web Services

A new distributed Web middleware layer was built which accesses the Sigmoid database and translates reaction sets into the input language of the xCellerator cell model generator, then calls xCellerator with requests for model generation and simulation and receives output plots in response. All these functions are exposed as Web services available to Java application programs and/or other clients. In addi-

tion to load balance and security management, the middleware provides a gateway between the front-end and the back-end of the architecture, allowing each one to evolve independently as long as the interface to the middleware is properly maintained. Furthermore, the middleware allows scalability in terms of the number of users that can be served simultaneously simply by increasing the computational and database server resources [Cheng *et al.*2005].

6.4 The Graphical User Interface: Sigmoid Model Explorer (SME) User Interface

The last component of the system to be initiated, and the most recent to achieve functional maturity, is the SME Web-compatible Graphical User Interface. The GUI allows the user to visualize, design, edit, and store pathway models, parameters, and initial conditions and their properties, to simulate the models by calling the simulator through the middleware, and to view and compare the properties of simulated models, for instance by viewing the temporal evolution of the concentration of chemical species under different conditions. The GUI runs from any Web browser as a Webstart or as a local client program.

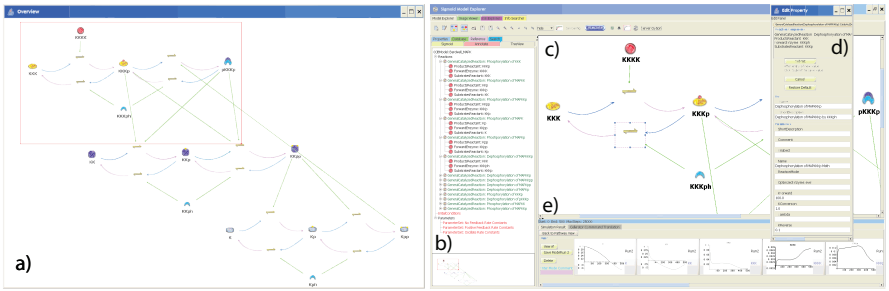

Fig. 4 Sigmoid Model Explorer showing portion of MAPK pathway. (a) Global Network View; (b) TreeView of compositional hierarchy; (c) network layout visualization; (d) parameter-editing panel. (e) output plot preview panel. Along the top are various action buttons for saving and running the model, and for switching the main panel to view output plots. User can select reaction icons.

The Sigmoid Model Explorer (SME) GUI is a Java application that is aware of the current Sigmoid object schema by using Java reflection. The SME GUI can be downloaded and also (as a Webstart) automatically updated through the Web. In addition Sigmoid uses Web-compatible Internet communication protocols (XML and SOAP) to perform three-tier distributed computing through the intelligent Web services middleware, which in turn communicates with the Sigmoid database and with xCellerator. Thus a variety of software platforms in addition to the SME Java application could use Sigmoid through its Web services. The SME GUI can display biological modeling objects in a compositional hierarchy, supports browsing and

selection from the model database, and supports editing of numerical parameters. It also supports display and editing of network layouts as bipartite labeled graphs with a user- definable mapping of object types to icons. Finally SME enables a simulation to execute remotely, or locally, and return sets of plots for side-by-side comparison with previous plot sets.

Recent enhancements to SME are: (1) For model creation; There exists a new mechanism to create biological models completely from within SME and save them locally or, commit them to the database. To facilitate the construction of more complex biological processes, one to many mathematical reactions can be assigened to each biological reaction. Also, there are utilities to facilitate the use of webpages as source of information for data input and perform queries to the Gene Ontology database from within SME. Gene Ontology entities can either be used to tag Sigmoid objects or, instantiated directly as Sigmoid objects, ie. Reactants or Biological reactions. (2) Enhanced display features; In the biological network layout view, SME allows the user to hide parts of a model diagram individually or by an Object class and has new ease of use features like hiding edges between objects in a model diagram, collapsing multiple entities to a single node, one-click display of diagram object properties and, support for the display of multiple math reactions for a single biological reaction. Users can utilize a large library of Sigmoid JPEG/GIF icon sets to represent nodes in the network or easily retrieve images from the web using a URL. Layouts can now be saved as a separate file either locally or to the database and, model diagrams can be output as .dot, TIFF or JPEG formats for use in presentations. (3) Model translation; SME can preform local translation of Sigmoid models to xCellerator code and can perform translation of SBML 1.0 to Mathematica code. (4) Model simulation; SME supports simulation through local a Mathematica license using the JLink library as well as through the remote server and there is an option to retrieve and display the output graphs for intermediate complexes generated by xCellerator/kMech reaction types. (5) Connectivity; SME now supports the Web Services Description Language (WSDL), which is an XML grammar for describing network services. Supporting WSDL expedites adoption of supplementary datasets and functionalities from other systems that support this standard.

7 Results

7.1 Sigmoid Database Population

The generative version of Sigmoid has been successfully populated with over twenty published models that range from simple molecular interactions to complex cell fate decision networks. A majority of the models in the database focus on virtual representation of intracellular pathways that include examples in signaling, metabolism, the cell cycle, and gene regulation. Large-scale models of the signaling pathways include the mammalian Epidermal Growth Factor Recep-

tor (EGFR) pathway [Kholodenko *et al.*1999] and the yeast pheromone response pathway [Kofahl and Klipp2004], while other models represent common aspects of metabolism that include the anabolic Calvin cycle in plants [Poolman *et al.*2004], branched chain amino acid biosynthesis in bacteria [Najdi *et al.*2006], [Yang *et al.*2005a], and catabolic glycolysis [Nielsen *et al.*1998]. Furthermore, a simple model of the circadian clock [Tyson *et al.*1999] and two models of intracellular calcium flux [Borghans *et al.*1997] demonstrate oscillating outputs. Separate models of the NFkB [Hoffmann *et al.*2002], calcineurin [Hilioti *et al.*2004] and the p53 [Bullock and Fersht2001] regulatory networks demonstrate how transcription factors and their ability to activate or inhibit gene expression are regulated. Lastly, some models in the database represent diverse processes, including the mechanism of degradation of enzymes during industrial food processing [Brands and van Boekel2002] and the cell fate decisions of protists in the presence of far-red light under starvation conditions [Marwan2003].

Finally, computational models of the mitogen-activated protein kinase (MAPK) cascade are also present in the Sigmoid database. Several models derived from [Markevich *et al.*2004] examine the same MAPK cascade with two separate mechanisms, mass action and Michaelis-Menten, for each of the phosphorylation and dephosphorylation events. For each of these mechanisms, the models increase in complexity as the site and order of phosphorylation are taken into account in the set of reactions. In contrast to these models, Huang_1996_MAPK and its xCellerator notebook "MAPK cascade: Huang and Ferrell 1996", present the celebrated [1996] model that demonstrates the connection between a nonprocessive, two-collision dual-phosphorylation mechanism of kinase activation and an ultrasensitive, switch-like response. The model Bardwell_ 2007_MAPK_VariableFeedback and corresponding notebook "MAPK Cascade with Variable Feedback" extend this model to include a simple feedback phosphorylation of an upstream kinase by the MAPK (Figure 4). The effects of the feedback loop on the system depend upon the nature of the feedback: if feedback phosphorylation increases the activity of the upstream kinase (positive feedback), a bistable, all-or-none response may result [Ferrell and Machleder.1998]. In contrast, if feedback phosphorylation decreases the activity of the upstream kinase (negative feedback), then the result may be damped or sustained oscillation of the activity of the kinases in the cascade [Kholodenko2000]. The notebook contains examples of parameter values that will generate either of these outcomes, illustrating how complex, diverse and biologically useful behaviors can emerge from the combination of an ultrasensitive cascade architecture and a simple feedback loop.

Since the flexible but comprehensive schema of the Sigmoid database allows us to easily leverage other databases, we are developing "populator" programs which take data available from other sources and bring it into Sigmoid. This will considerably increase the power of Sigmoid by capturing community input from diverse sources and making it available to a biologist end-user in an integrated manner. For example, without much effort we were able to populate Sigmoid with the yeast GOnet database [Irwin *et al.*2005], which contains information about yeast ORFs and their annotations, gene ontology (GO), and protein-protein interactions. The

GOnet database itself is periodically updated and integrates information from three different sources: (1) ORFs (description, mutant phenotype, gene product, etc.) from the Saccharomyces Genome DataBase (SGD); (2) GO term annotation from the Gene Ontology Consortium arranged in the three categories of Molecular Function, Biological Process, and Cellular Component; and (3) genetic and physical interactions information from the General Repository for Interaction Datasets (GRID).

7.2 Parameter Optimization

A Simulated Annealing Optimizer [Zhang2008] has been integrated into Sigmoid through the web services interface. It uses a global optimization technique and Lam-Delosme schedule to make the optimization process faster and more efficient when compared with other general schedules available [Lam and Delosme1988]. It aims to reverse engineer model parameters(for example: kinetic rate constants) given both the model structure (represented as ordinary differential equations) and empirical system dynamics as expressed by time series experimental data.

This SA optimizer has been developed in a flexible, efficient and scalable manner. It is designed with a modular fashion to accommodate maximum reusability and flexibility. It has built-in support for high performance computing power- a feature often missing from other optimization packages.

7.3 Parameter Analysis

The Parameter Analysis routine in Sigmoid allows one to quickly sample the parameter space of a particular model and quantify the diversity of model outputs resulting from variation of the parameters in specified ranges. First, free parameters are defined within the model that will be part of the analysis. Then, a simulation function is defined that accepts a particular parameter variation and returns the model's output. Users have options to select Sigmoid output functions, such as the temporal sequence of a particular state variable. The output variation is measured using preset or user defined metrics aimed at focusing on particular aspects of output behavior. For example, one can measure the difference between the obtained output and some reference time state or determine the time points, at which the output might have peaks or troughs in an oscillatory response. The value of the metric might reflect on how sensitive a certain model is to simultaneous variation of any number of parameters, from one to all. This information can be then used in investigation of robustness of the model and the corresponding biological process. The values of the varied parameters, model output, and resulting metrics are stored in a database table using Mathematica's DatabaseLink package. Using a database provides a convenient method for storing the vast amounts of tabular data and allows for rapid remote access. Since model evaluations are independent, the procedure is easily

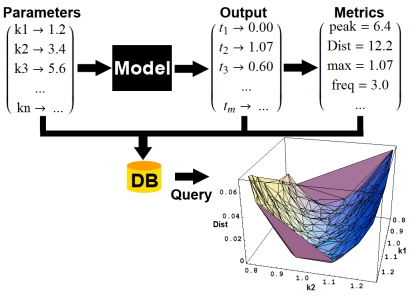

Fig. 5 Sensitivity of model output to parameter variations is handled by a set of operations integrated into the Sigmoid environment. These functions or their user-defined variants can allow fast and efficient generation of a set of solutions corresponding to variation of any parameter number from one to all and storage of these solutions in a database that can be queried to form various metrics of model performance. The results can be used to analyze the robustness of various models of a specific biochemical system of interest.

parallelized. The same notebook can run on multiple computers simultaneously, as long as all can connect to the same database. Lastly, Mathematica's powerful visualization and analysis features can be used to observe correlations between parameter values and associated metrics (See figure 5).

8 Conclusions

We have described the Sigmoid intelligent software infrastructure for systems biology. An initial version of each of the main components is available today and there are clear signs that the infrastructure can already be used to yield biologically relevant results. Since Sigmoid is based upon a computer algebra representation tool, it stands poised to serve as a formidable engine in model analysis. For instance, the *E. coli* metabolic pathway model correctly predicts the effect of certain mutations and, the MAP Kinase cascade model shows that, depending on the parameter sets and initial conditions chosen, it can generate a switch-like or graded input-output relationship, or even produce oscillatory behavior.

Development and expansion of Sigmoid continues at all levels. As the mediator of the user experience with Sigmoid, the GUI and web interface are bound to attract the largest number of feature requests from users. Because the overall architecture is now functional, many of these requests can be met at reasonable levels of effort and cost. There is also a need for new reaction types in xCellerator to deal with various kinds of (non-transcriptional) feedback. Other reaction types already in xCellerator and kMech (such as various enzymatic models, GMWC, GRN etc.) will need to be exposed for further pathway modeling. An essential aspect of the scale-up of Sig-

moid will be expert curation of the allowed and suggested mappings from biological reaction mechanisms to mathematical reaction models.

Likewise, we continue to expand and populate the Sigmoid database. It is possible to develop database "populator"codes for importing relevant data from other sources, depending on their accessibility to software agents, such as KEGG, Systems Biology Markup Language (SBML), Systems Biology Workbench (SBW), SiBML/GeneNet, Cytoscape, The Reactome, Biomodels, Biopax, SGD, Biocyc, and others. Increased standardization and inter-operability through, for instance, SBML (an XML-based protocol for systems biology information interchange [http://www.sbml.org for further information]) is possible xCellerator. xCellerator now both reads and writes SBML and thereby opens the gates to communication with 100 other SBML compatible simulators. Incorporation of the Gene Ontology (GO) *de facto* standard provides additional inter-operability. New modules and interfaces to analysis tools, for instance to optimize models, learn from data, or perform graph analysis, are under development in the back-end and are expected to enhance end-user capabilities.

Furthermore, the modularity of Sigmoid along with its separation of biological and mathematical representations, enables us to build interfaces to additional computer algebra systems outside of the Mathematica/xCellerator superstructure. SAGE([SteinLast accessed October 2009]) for instance, an open source mathematics software program largely constructed upon the Python framework, provides a gateway to a broad array of open source math programs such as Axiom, GAP, GP/PARI, Macaulay2, Maxima, Octave, and Singular. In addition, the SAGE language includes interfaces to commercial math programs like Magma, Maple, Mathematica, MATLAB, and MuPAD as well. Constructing a language interface to SAGE or some similar project would enable Sigmoid to harness the additional functionality provided by these packages.

Other packages, such as VCell, Sigpath, and JDesigner for example, have functionalities that might be similar to some of the features contained in Sigmoid. While it is sound to have a number of parallel efforts across multiple research groups, there are several features of the Sigmoid architecture that, in aggregate, position it uniquely within realm of the currently available systems biology software systems. Sigmoid introduced the web services framework [Cheng *et al.*2005] to create a truly distributed system. This flexible framework offers powerful modularity that, in conjunction with the generative nature of the Sigmoid coding cycle, offers a significantly reduced development time for integration of new components and data structures. Also, the OJB object relational bridge offers the advantages of oriented programming in conjunction with relational databasing. Sigmoid capitalizes on the robust mathematical software tools and problem solving environment that Mathematica offers (along with the xCellerator/kMech packages designed to facilitate biological modeling via automated equation generation) while remaining open to other simulation and analysis tools. The synthesis of these features yields a flexible scalable architecture that not only allows for manageable, cost effective, adoption of new system components, but may open the ability to play within yet larger bioinformatics frameworks.

Ultimately constructing multi-scalar, predictive models of multicellular organisms would be a healthy ambition for the field of systems biology. If we are to reverse engineer biological organisms, scaling up from the pathway level to cells and multicellular systems presents a formidable challenge. Tools must be designed that can handle and integrate multiple temporal and spatial scales over several orders of magnitude while modeling combinations of continuous, stochastic, and discrete events with different levels of compartmentalization. As our understanding of these biological systems progresses, the schema we use to model them must evolve in pace.

Acknowledgment

This work has been supported by NSF grant EIA-0321390 and NIH grant T15 LM007443 to PB; a Laurel Wilkening faculty innovation award to PB; a UC Systemwide Biotechnology Research and Education Program 2002-06 award to PB; NIH grant GM069013 to EM; NCI Director's Challenge support to Children's Hospital Los Angeles for EM; B.C. was supported by grant T15LM07443 from the National Library of Medicine at the National Institutes of Health; A.L. was supported by NIH grants: GM69013 and GM072024. NASA Intelligent Systems Program support of EM, and by the Institute for Genomics and Bioinformatics at UCI. We would like to thank students, programmers, and colleagues that have provided us with valuable feedback or have helped implement particular components of the infrastructure. These include B. Bornstein, G. Wesley Hatfield, P. Hebden, E. Meyerowitz, K. Petrov, L. Scharenbroich, T. Najdi, L. Zhang, B. Shapiro, D. Trout, C. Yang.

References

[B. Shapiro2007] M. Hucka E. Mjolsness B. Shapiro, J. Lu. Mathematica platforms for modeling in systems biology: Recent developments in mathsbml and cellerator. 2007.

[Borghans *et al.*1997] J. M. Borghans, G. Dupont, and A. Goldbeter. Complex intracellular calcium oscillations. a theoretical exploration of possible mechanisms. *Biophys Chem.*, 66(1):25–41, 1997.

[Brands and van Boekel2002] C. M. Brands and M. A. van Boekel. Kinetic modeling of reactions in heated monosaccharide-casein systems. *J Agric Food Chem.*, 50(23):6725–39, 2002.

[Bullock and Fersht2001] A. N. Bullock and A. R. Fersht. Rescuing the function of mutant p53. *Nat Rev Cancer*, 1(1):68–76, 2001.

[Cheng *et al.*2005] J. Cheng, L. Scharenbroich, P. Baldi, and E. Mjolsness. Sigmoid: Towards a generative, scalable software infrastructure for pathway bioinformatics and systems biology. *IEEE Intelligent Systems*, 20(3):68–75, 2005.

[Ferrell and Machleder.1998] J. E. Ferrell and E. M. Machleder. The biochemical basis of an all-or-none cell fate switch in xenopus oocytes. *Science*, 280:895–898, 1998.

[Hilioti *et al.*2004] Z. Hilioti, D. A. Gallagher, S. T. Low-Nam, P. Ramaswamy, P. Gajer, T. J. Kingsbury, C. J. Birchwood, A. Levchenko, and K. W. Cunningham. Gsk-3 kinases enhance calcineurin signaling by phosphorylation of rcns. *Genes Dev.*, 18(1):35–47, 2004.

[Hoffmann *et al.*2002] A. Hoffmann, A. Levchenko, M. L. Scott, and D. Baltimore. The ikappab-nf-kappab signaling module: temporal control and selective gene activation. *Science*, 298(5596):1241–5, 2002.

[Irwin *et al.*2005] B. Irwin, M. Aye, P. Baldi, N. Beliakova-Bethell, H. Cheng, Y. Dou, W. Liou, and S. Sandmeyer. Retroviruses and yeast retrotransposons use overlapping sets of host genes. *Genome Research*, 15:641–654, 2005.

[Kholodenko *et al.*1999] B. N. Kholodenko, O. V. Demin, G. Moehren, and J. B. Hoek. Quantification of short term signaling by the epidermal growth factor receptor. *J Biol Chem.*, 274(42):30169–81, 1999.

[Kholodenko2000] B. N. Kholodenko. Negative feedback and ultrasensitivity can bring about oscillations in the mitogen-activated protein kinase cascades. *Eur J Biochem*, 267:1583–1588, 2000.

[Kofahl and Klipp2004] B. Kofahl and E. Klipp. Modelling the dynamics of the yeast pheromone pathway. *Yeast.*, 21(10):831–50, 2004.

[Lam and Delosme1988] J. Lam and J. Delosme. Performance of a new annealing schedule. pages 306–311. 1988.

[Markevich *et al.*2004] N. I. Markevich, J. B. Hoek, and B. N. Kholodenko BN. Signaling switches and bistability arising from multisite phosphorylation in protein kinase cascades. *J Cell Biol*, 164(3):353–9, 2004.

[Marwan2003] W. Marwan. Theory of time-resolved somatic complementation and its use to explore the sporulation control network in physarum polycephalum. *Genetics*, 164(1):105–15, 2003.

[Najdi *et al.*2005] T. S. Najdi, C. R. Yang, B. E. Shapiro, G. Wesley Hatfield, and E. D. Mjolsness. The generalized Monod, Wyman, Changeux model for mathematical modeling of metabolic enzymes with allosteric regulation. In *Proc. IEEE Computational Systems Bioinformatics Conference*, Stanford University, CA, 2005.

[Najdi *et al.*2006] T. S. Najdi, C. R. Yang, B. E. Shapiro, G. W. Hatfield, and E. D. Mjolsness. Application of a generalized mwc model for the mathematical simulation of metabolic pathways regulated by allosteric enzymes. *J Bioinform Comput Biol.*, 4(2):335–55, 2006.

[Nielsen *et al.*1998] K. Nielsen, P. G. Sarensen, F. Hynne, and H. G. Busse. Sustained oscillations in glycolysis: an experimental and theoretical study of chaotic and complex periodic behavior and of quenching of simple oscillations. *Biophys Chem.*, 72(1-2):49–62, 1998.

[Poolman *et al.*2004] M. G. Poolman, H. E. Assmus, and D. A. Fell. Applications of metabolic modelling to plant metabolism. *J Exp Bot.*, 55(400):1177–86, 2004.

[Segel1992] I. H. Segel. *Enzyme Kinetics. Behavior and Analysis of Rapid Equilibrium and Steady State Enzyme Systems*. Wiley, New York, NY, 1992.

[Shapiro *et al.*2003] B. E. Shapiro, A. Levchenko, E. M. Meyerowitz, B. J. Wold, and E. D. Mjolsness. Cellerator: Extending a computer algebra system to include biochemical arrows for signal transduction simulations. *Bioinformatics*, 19(5):677–678, 2003.

[Stein2009] W. Stein. *SAGE: Software for Algebra and Geometry Experimentation*. http://www.sagemath.org/ and http://sage.scipy.org/ Last access, Oct. 2009, 2009.

[Tyson *et al.*1999] J. J. Tyson, C. I. Hong, C. D. Thron, and B. Novak. A simple model of circadian rhythms based on dimerization and proteolysis of per and tim. *Biophys J.*, 77(5):2411–7, 1999.

[Yang *et al.*2005a] C. R. Yang, B. E. Shapiro, S. P. Hung, E. D. Mjolsness, and G. W. Hatfield. A mathematical model for the branched chain amino acid biosynthetic pathways of escherichia coli k12. *J Biol Chem.*, 280(12):11224–32, 2005.

[Yang *et al.*2005b] C. R. Yang, B. E. Shapiro, E. D. Mjolsness, and G. W. Hatfield. An enzyme mechanism language for the mathematical modeling of metabolic pathways. *Bioinformatics*, 21:774–780, 2005.

[Zhang2008] L. Zhang. *Dynamic Biological Signaling Pathway Modeling and Parameter Estimation Through Optimization*. PhD thesis, Information and Computer Science: University of California, Irvine, 2008. LD 791.9 I5 2008 Z43, OCLC:276454918.