Tuesday – June 7th

10:30am - 12:00pm: Tech. Talks Session 4 - *Biological Circuit Simulators*
4BDA.1 **Biological Network Emulation in FPGA** Natasa Miskov-Zivanov, Andrew Bresticker, Deepa Krishnaswamy, Sreesan Venkatakrishnan, Diana Marculescu and James Faeder
4BDA.2 **Modeling and Visualization of Genetic Circuits** Tyler Patterson, Nicholas Roehner, Curtis Madsen and Chris Myers
4BDA.3 **Asynchronous Sequential Computation with Molecular Reactions** Hua Jiang, Marc Riedel and Keshab Parhi

Biological Network Emulation in FPGA

Natasa Miskov-Zivanov[1], Andrew Bresticker[2], Deepa Krishnaswamy[2], SreesanVenkatakrishnan[2], Diana Marculescu[2] and James R. Faeder[1]

***Short Abstract* — Models of biological networks have been studied through simulations using a number of software tools. However, the intrinsic disparity between the sequential nature of microprocessor architecture used in software-based simulations and the highly parallel nature of biological systems often results in prohibitively long simulation times for larger networks. In this work, we adopt an alternative approach to simulation of biological systems using hardware-based emulation. Our results on Boolean network models of three case studies show that such an approach can provide *speedup of five orders of magnitude* when compared to existing software simulation approaches.**

I. Introduction

When tackling important biological and medical problems, scientists usually design experiments that are focused on particular parts of a system under study, with a specified experimental setup and initial conditions. However, only by studying the system in a formal and systematic way, one can acquire sufficient knowledge about the system, so as to attain a comprehensive understanding of a system. In general, one can think of the main goals of modeling biological systems as following: ***explaining processes*** that result in observed phenomena; ***predicting previously unobserved phenomena***; ***identifying key generic reactions***; ***guiding experiments*** by suggesting new experiments, avoiding unneeded experiments and helping interpret experiments.

Over the past decade, a number of computational approaches have been proposed for the purpose of modeling and studying biological systems. However, the complexity of these models increases rapidly with the size of the network. Moreover, such simulations are computed sequentially on general purpose CPUs, which is in contrast to the highly parallel nature of information flow within biochemical networks. In general, models of biological networks are simulated by assuming an initial state (or distribution of initial states) as the model input, with dynamical trajectories and steady states as the model output. These simulations often result in prohibitively long run times.

The main goal of this work is to develop a hardware design methodology for emulating biological network behavior into Field Programmable Gate Array (FPGA) platforms, which are ideally suited to implement highly parallel architectures. We provide a framework for efficient and accurate analysis of complex signaling and regulatory networks, since their understanding is of great interest in medicine and biology. We outline here the contributions of our work when compared to both software-based and hardware-based approaches to studying signaling networks:

- In comparison to software-based approaches to simulate biological networks, our proposed approach is ***efficient***, because **hardware-based emulation** can give better performance than software-based simulation.
- In comparison to hardware-based approaches to studying biological networks, our approach: (*i*) ***Implements the logical (dynamic) model of the network***, while previous approaches only implemented the simulation algorithm for ODE models; (*ii*) ***Improves efficiency through implementation of several copies of the model***, allowing for simultaneous runs and faster comparison of several different implementations of the system, or outcomes of different initial conditions and different scenarios.

[1]Department of Computational and Systems Biology, School of Medicine, University of Pittsburgh, E-mail: {nam66,faeder}@pitt.edu
[2]Electrical and Computer Engineering Department. Carnegie Mellon University. E-mail: {abrestic, deepakri, sreesanv, dianam}@ece.cmu.edu.

II. Methodology

In this work, we focus on analyzing models that include Boolean or integer variables. In the past years, several groups have worked on developing Boolean network models for different signaling pathways [5] or gene regulatory networks [2]. These models are used to study the dynamics of the system and its behavior (state transitions) from the initial state until reaching a steady state or steady cycle. The model is composed of fixed sets of variables and rules.

Once the logical model is defined, the model is used as an input to the hardware design phase. This includes the design of a circuit, which usually starts with defining larger blocks. The next step in hardware design is to define modules within each block and describe them in a Hardware Description Language (HDL). The design can be simulated before the actual implementation in hardware, using an HDL simulator such as ModelSim [4]. Following are the main elements of the circuit designed to emulate the biological network model:

- **Inputs** - represent external stimuli or environmental effects that can occur anytime during circuit simulation (*e.g.*, changes in conditions outside of the system, inhibitor addition, gene knock-outs, *etc.*), as well as arrays of initial conditions (*e.g.*, element states or molecule counts).
- **Outputs** - used to collect the information about the behavior of the system as a response to initial conditions and changes in inputs.
- **Model Implementation**, consists of: (*i*) *Execution Unit*, which implements the network (elements and rules); (*ii*) *Control Unit*, which controls the simulation (and timing of interactions) of the implemented network.
- **Display** - outputs can be directed to a display to visually present the outcomes of model simulations.

When considering the translation of biological network models into hardware, one needs to define the time of the execution of individual interactions. FPGAs seem very suitable for implementing biological network models because of their parallel nature and in fact, we intend to develop design techniques that can facilitate correct relative timing when emulating signaling networks. For example, we approach the problem of implementing asynchronous network updates using synchronous hardware (that uses a clock signal) by designing a system with an embedded pseudo-random number generator.

III. Results

We applied our approach on following networks:

1. T cell large granular lymphocyte leukemia (T-LGL) [7];
2. Helper T cell differentiation [5];
3. Peripheral naïve T cell differentiation [3].

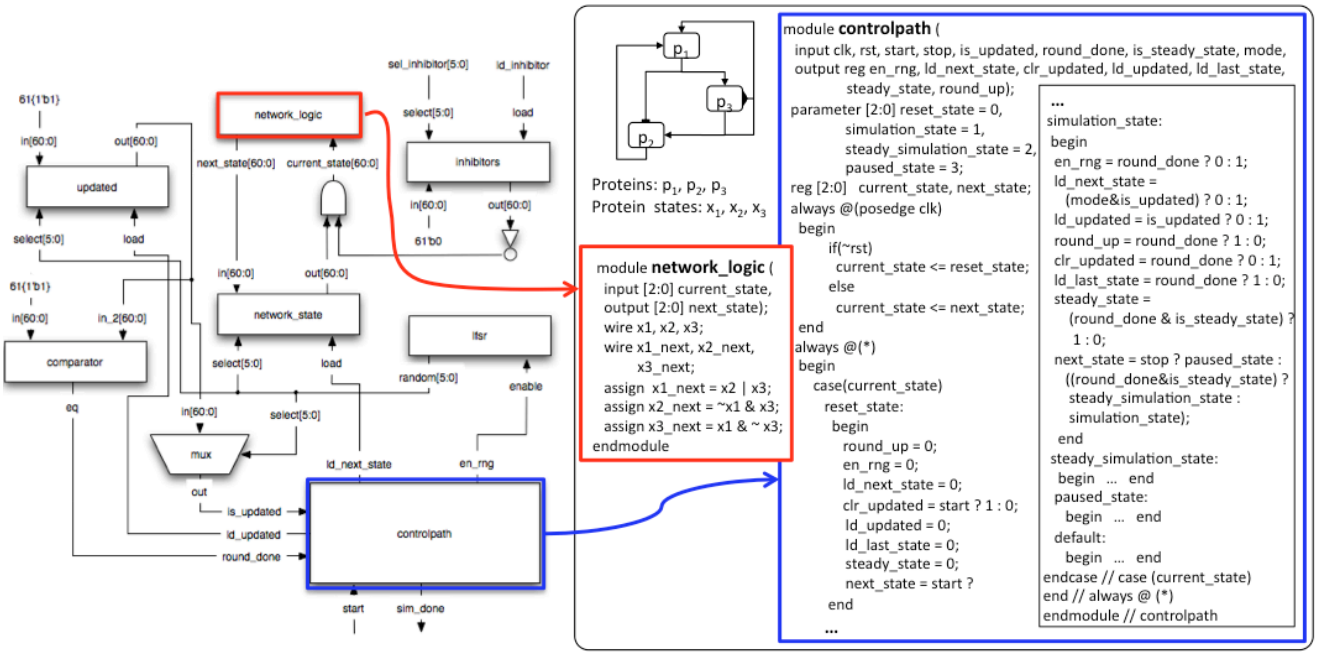


Figure 1. Designed circuit modules (left) and Verilog implementation (right) of two modules: `network_logic` module for the example network and `controlpath` module.

These models use discrete variables to represent elements of the network and logic update functions to represent interactions between elements. We first implemented these models in a hardware description language, Verilog. The stochasticity that exists in biological networks has been introduced through random ordering of updates of all elements. To compare hardware emulation results with software simulations, we ran simulations of models using BooleanNet [1]. After verifying the design through software simulation, it was synthesized for implementation on a Xilinx FPGA [6].

In the first example network, the runtime for 200 software simulations, with 16 update rounds in each simulation, was approximately 60s. Given the clock frequency for the implemented systems (50MHz), number of rounds necessary to reach the steady state (16), number of steps in each round (61), and number of clock cycles per step (approximately 5) and number of simulations (200), the estimated time for obtaining steady state values for the emulated network is 19.2 ms. In other words, the speedup from a single model implementation was approximately 3100X. Given the fact that multiple copies of the model can be implemented on a single FPGA and run in parallel, the speedup can scale further (possibly, linearly) with the number of model copies on the chip. In the case of the FPGA circuit used in the first study, five copies were implemented, providing an overall speedup of 15,000X when compared to software simulations.

Similar computations to those done in the first example, could be conducted for the second example network, where the model consists of 23 elements and therefore, each round consists of 23 update steps.

In the third example, we used a slightly different FPGA board that was operating at 1 GHz frequency. In a round, all 54 nodes are updated once and only once in a random order. We implemented six instances of the network on a Spartan 3 FPGA board simultaneously and executed them in parallel. The speedup of a single instance implementation when compared to software simulations in BooleanNet is approximately 60,000X, which in turn indicates six times 60,000X overall speedup for six instances.

Table I. Hardware emulation speedup (single instance).

Model	# nodes	# rounds	BooleanNet	FPGA
T-LGL	61	15	60s	0.019s
Th diff.	23	25	65s	0.022s
T cell diff.	54	20	60s	0.001s

IV. Conclusion

In this work, we proposed a methodology to implement logical models of signaling networks using FPGAs, in order to efficiently study these networks. We have shown that the speedup that an FPGA simulation provides over the classic simulation-based approach can be as high as six times 60,000X for up to six copies of a model implemented on a single FPGA These results demonstrate the tremendous potential of hardware-based emulation. As future work, we plan to create a general-purpose framework that can also translate models given in either logical rule-, reaction network- or reaction rule-based format into a faithful hardware implementation replica of the system that reproduces both dynamic and steady-state behavior. This will facilitate fast and accurate analysis of system behavior for many different initial conditions and scenarios, thus opening the possibility for scalable simulation and analysis of highly complex biological systems.

References

[1] I. Albert *et al.*, "Boolean network simulations for life scientists," in *Source Code Biol Med*, 2008.
[2] A. Garg, "Synchronous versus asynchronous modeling of gene regulatory networks," in *Bioinformatics*, 2008, pp. 1917-1925.
[3] N. Miskov-Zivanov *et al.*, "Logical Modeling of Peripheral T Cell Differentiation," in preparation.
[4] ModelSim - Advanced Simulation and Debugging, Mentor Grpahics, http://model.com/.
[5] J. Saez-Rodriguez et al., "A logical model provides insights into T cell receptor signaling," in *Plos Comput Biol*, 2007, pp. 1580-1590.
[6] XILINX, http://www.xilinx.com/.
[7] R. Zhang *et al.*, "Network model of survival signaling in large granular lymphocyte leukemia," in *Proc Natl Acad Sci* 2008, pp. 16308-16313.

Modeling and Visualization of Genetic Circuits *

Tyler Patterson,
Dept. of Elec. and Comp. Eng.
University of Utah
Salt Lake City, UT 84112
tpatterson80@gmail.com

Nicholas Roehner
Dept. of Bioengineering
University of Utah
Salt Lake City, UT 84112
n.roehner@utah.edu

Curtis Madsen
School of Computing
University of Utah
Salt Lake City, UT 84112
curtis.madsen@utah.edu

Chris J. Myers
Dept. of Elec. and Comp. Eng.
University of Utah
Salt Lake City, UT 84112
myers@ece.utah.edu

Recently, biologists and engineers have begun to work together on *synthetic biology* [2, 6]. These researchers are attempting to design synthetic genetic circuits that enable bacteria to consume toxic waste [4], destroy tumors [1], and produce drugs [11]. Although some *genetic design automation* (GDA) tools have been developed [3, 5, 8–10], there is still a need for more efficient methods for their modeling, analysis, and design. One such GDA tool is `iBioSim` which supports the representation of these circuits using a high-level *Genetic Circuit Model* (GCM) that can be translated into the *Systems Biology Markup Language* (SBML) for analysis. Several analysis methods are supported including differential equations, stochastic simulation, and Markovian analysis. The efficiency of these methods is enhanced by automatic reaction-based and logical abstraction methods [7].

Recently, we have created a user interface that is similar to those used to construct schematic diagrams which are familiar to electrical engineers. Promoters, chemical species, and biological relationships can be placed on a graphical schematic. Figure 1 shows an example screenshot for the schematic capture tool. This figure shows two identical promoters, P_{RFP} and P_{GFP}, which initiate transcription of two genes producing mRNAs that are translated into two proteins, *RFP* and *GFP*, as indicated by the edges connecting the promoters to the proteins. Although not shown, edges can also be connected from proteins to promoters to indicate that the protein activates or represses the promoter regulating downstream gene expression. In these cases, the promoter can often be hidden to simplify the diagram. Finally, complex formation reactions can be included to represent how proteins combine to form other proteins. In each case, kinetic rates and other parameters can be configured individually for use during analysis. Finally, GCM components can be placed on the schematic to support hierarchical modeling in which larger circuits are created from smaller circuits. These components can also represent entire genetic circuits running within individual cells in order to support population modeling.

Another enhancement was the creation of a new simulation visualization tool that leverages this schematic capture tool to allow the user to associate chemical species with color schemes, opacity, and cell size. Simulation data can then be associated with this tool to enable the user to effectively observe a population of cells, *in silico*. As an example, Figure 2 shows a screenshot of this visualization tool as applied to 40 copies of the genetic circuit shown in Figure 1. This example is inspired by the experiment by Elowitz et al. at Caltech in which individual cells of the bacterium *Escherichia coli* express red and green fluorescent proteins under the control of identical promoters. If gene expression were deterministic, all cells would express equal amounts of red and green proteins, and hence appear yellow. In the actual experimental data, the appearance of redder and greener cells in the population indicate the importance of intrinsic randomness, or "noise," in gene expression. As shown in Figure 2, these results are clearly reproduced.

This presentation will highlight these latest features of `iBioSim` by describing its application to the modeling, analysis, and design of a variety of synthetic genetic circuits. This presentation will also describe recent work to associate DNA sequences to GCM structures using the *synthetic biology open language* (SBOL). Finally, we will discuss a number of on-going ideas to further enhance this tool to represent the dynamics of large populations of cells that are growing, dividing, moving around, and communicating with their environment through the use of chemical signals. `iBioSim` is available at `http://www.async.ece.utah.edu/iBioSim/`.

*This research is supported by the National Science Foundation under Grant CCF-0916042.

1. REFERENCES

[1] J. C. Anderson, E. J. Clarke, and A. P. Arkin. Environmentally controlled invasion of cancer cells by engineering bacteria. *J. Mol. Biol.*, 355:619–627, 2006.

[2] A. Arkin. Setting the standard in synthetic biology. *Nature Biotech.*, 26:771–774, 2008.

[3] Y. Cai, M. L. Wilson, and J. Peccoud. GenoCAD for iGEM: a grammatical approach to the design of

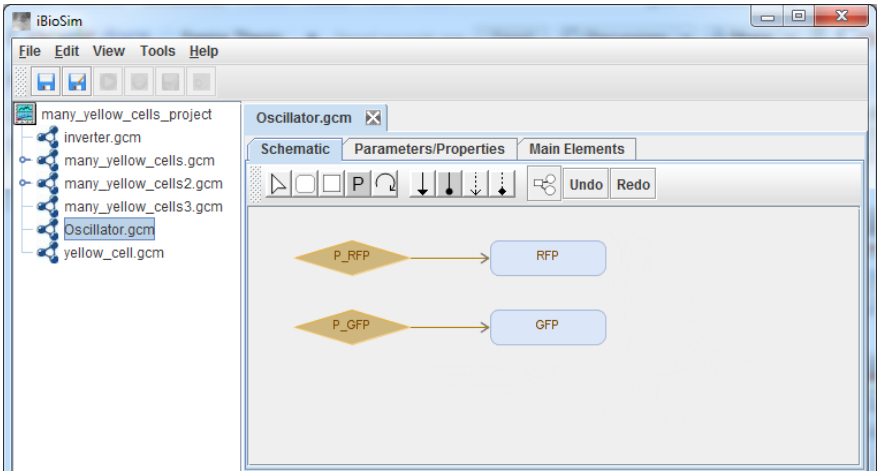


Figure 1: Schematic capture tool depicting a simple genetic circuit in which two promoters initiate the production of two species, red fluorescent protein (RFP) and green fluorescent protein (GFP).

Figure 2: Schematic visualization tool showing a simulation of the Elowitz et al. experiment in which the genetic circuit in Figure 1 was introduced into *E. coli* bacteria. The noisy nature of biological systems causes many shades of color ranging from green to red, with yellow colors in between. The visualization tool allows a researcher to see a simulation of a population in a way similar to looking through a microscope.

standard-compliant constructs. *Nucleic Acids Res.*, 38(8):2637–44, 2010.

[4] I. Cases and V. de Lorenzo. Genetically modified organisms for the environment: stories of success and failure and what we have learned from them. *International Microbiology*, 8:213–222, 2005.

[5] D. Chandran, F. T. Bergmann, and H. M. Sauro. TinkerCell: modular CAD tool for synthetic biology. *J. Biol. Eng.*, 3(19), 2009.

[6] D. Endy. Foundations for engineering biology. *Nature*, 438:449–453, 2005.

[7] H. Kuwahara, C. Myers, N. Barker, M. Samoilov, and A. Arkin. Automated abstraction methodology for genetic regulatory networks. *Trans. Comp. Syst. Biol.*, VI:150–175, 2006.

[8] M. A. Marchisio and J. Stelling. Computational design of synthetic gene circuits with composable parts. *Bioinform.*, 24(17):1903–10, 2008.

[9] C. J. Myers, N. Barker, K. Jones, H. Kuwahara, C. Madsen, and N.-P. D. Nguyen. iBioSim: a tool for the analysis and design of genetic circuits. *Bioinform.*, 25(21):2848–2849, 2009.

[10] M. Pedersen and A. Phillips. Towards programming languages for genetic engineering of living cells. *J. R. Soc. Interface*, 6((Suppl. 4)):S437–S450, 2009.

[11] D.-K. Ro, E. M. Paradise, M. Ouellet, K. J. Fisher, K. L. Newman, J. M. Ndungu, K. A. Ho, R. A. Eachus, T. S. Ham, J. Kirby, M. C. Y. Chang, S. T. Withers, Y. Shiba, R. Sarpong, and J. D. Keasling. Production of the antimalarial drug percursor artemisinic acid in engineered yeast. *Nature*, 440, 2006.

Asynchronous Sequential Computation with Molecular Reactions *

Hua Jiang
hua@umn.edu

Marc D. Riedel
mriedel@umn.edu

Keshab K. Parhi
parhi@umn.edu

Department of Electrical and Computer Engineering, University of Minnesota
200 Union St. S.E., Minneapolis, MN 55455

1. SUMMARY

Just as electronic systems implement computation in terms of voltage (*energy per unit charge*), molecular systems compute in terms of chemical concentrations (*molecules per unit volume*). In prior work, we described mechanisms for implementing logical and arithmetic functions, such as comparison, exponentiation and logarithms, with molecular reactions [1, 3, 4]. All such functions are memoryless – "combinational" functions in the jargon of circuit design. In current work, we are developing techniques for implementing computation with memory – "sequential" functions in the jargon of circuit design. In a paper to be presented at this year's Design Asynchronous Conference, we describe techniques for implementing *synchronous* sequential computations, that is to say, computation that is synchronized by a clock [2]. In this paper, we discuss methods for implementing *asynchronous* computation, that is to say, computation that is self-timed. We implement the computation with a multi-phase "handshaking" protocol that transfers quantities between molecular types based on the absence of other types. Our method produces computation that is accurate and independent of the reaction rates, assuming only that some reactions are faster than others. We validate our designs through ODE simulations of the mass-action chemical kinetics. We are exploring DNA-based computation via strand displacement as a possible experimental chassis.

2. CONTEXT

Most prior schemes for molecular computation depend on specific values of the kinetic constants (the k's associated with each reaction.) This limits the applicability since the kinetic constants are not constant at all; they depend on factors such as cell volume and temperature.

We aim for robust constructs: in our methodology we use only coarse rate categories ("fast" and "slow"). Given such categories, the computation is exact and independent of the specific reaction rates. In particular, it does not matter how fast any "fast" reaction is relative to another, or how slow any "slow" reaction is relative to another – only that "fast" reactions are fast relative to "slow" reactions.

Digital circuits are generally composed of two types of components: those that implement *computation* and those that implement *memory*. The most common design paradigm is to use a global clock signal to synchronize transfers between computation and memory. The clock signal is generated by an oscillatory circuit that produce periodic voltage pulses. We advocate this approach for molecular computation in [2]. For a molecular clock, we choose reactions that produce sustained oscillations in the chemical concentrations. With such oscillations, a low concentration corresponds to logical value of zero; a high concentration corresponds to a logical value of one.

For computational constructs, we refer the reader to prior work [1, 3, 4]. Operations such as addition and scalar multiplication are straightforward. Operations such as multiplication, exponentiation, and logarithms are trickier. These can be implemented with reactions that implement iterative constructs analogous to "for" and "while" loops. (They do so robustly and exactly, without any specific dependence on the rates.)

An alternative to a global clock is to use self-timed constructs that synchronize transfers locally. In this work, we present such an asynchronous scheme for molecular computation. (Here "global" and "local" refer to how reactions are coupled, not to any kind of spatial structure of the chemical solution.) We model the molecular dynamics in terms of mass-action kinetics. We assume that the chemical solution is homogeneous and well-stirred. All *signals* are quantities of chemical types.

3. METHOD AND RESULTS

We implement transfers of signals with *delay* elements. A example of a two-delay-element chain is shown in Figure 1(a). Here D_1 releases its signal to T and accepts a new signal X. Meanwhile, D_2 releases its signal to Y and accepts the signal T that D_1 just released.

In spite of the fact that the chemical reactions fire at variable rates, we seek to realize *ordered* delay operations. We implement a delay element as a sequence of chemical transfers. We assign each of the delay elements three molecular types, R_1, G_1, and B_1 for D_1, and R_2, G_2, and B_2 for D_2, as shown in Figure 1(b). Reactions $X \rightarrow R_1$, $R_1 \rightarrow G_1$, $G_1 \rightarrow B_1$, $B_1 \rightarrow R_2$, $R_2 \rightarrow G_2$, $G_2 \rightarrow B_2$, $B_2 \rightarrow Y$ effect the transfer of the quantity of X to the quantity of Y.

A computation cycle, in which an input value is accepted and an output value is computed, completes in several phases. In each phase the signals are transfered from molecular types in one category to the next.

*This work is supported by an NSF EAGER Grant, #CCF0946601.

IWBDA 2011, June 5-6, 2011, San Diego, California, USA.

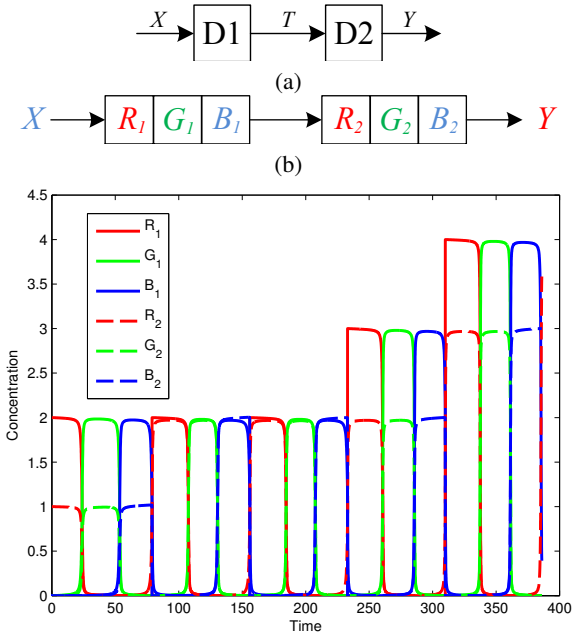


Figure 1. (a) A two-delay-element chain. (b) The chain labeled with molecular types; signals are represented by quantities of each type. (d) ODE simulation of the chemical kinetics.

All signals are assigned to one of the three categories: red, green, and blue. All operations transfer molecules in one of the three phases: red to green, green to blue, and blue to red.

We apply following technique to order the transfers of delay elements. Firstly, we use B_0 to represent X and R_3 to represent Y. Then, all molecular types are color coded into three categories: R_1, R_2, and R_3 are red; G_1 and G_2 are green; B_0, B_1, and B_2 are blue. Each transfer is now one of the three: red-to-green, green-to-blue, or blue-to-red. To control the order in which reactions fire, we enable reactions that transfer between two categories of molecules in absence of the third category.

Consider following reactions

$$\begin{aligned} \varnothing &\xrightarrow{k_{slow}} r \\ \varnothing &\xrightarrow{k_{slow}} g \\ \varnothing &\xrightarrow{k_{slow}} b \\ R_i + r &\xrightarrow{k_{fast}} R_i, \\ G_i + g &\xrightarrow{k_{fast}} G_i, \\ B_i + b &\xrightarrow{k_{fast}} B_i, \end{aligned} \tag{1}$$

The first three reactions have zero-order kinetics, generating molecular types r, g, and b constantly, at a slow rate. The other reactions quickly consume these types: red molecules quickly consume r; green molecules quickly consume g; and blue molecules quickly consume b. Molecules of r, g, and b can accumulate only when molecules in corresponding color categories are absent. Accordingly, we call these "absence indicators."

Consider the following reactions

$$\begin{aligned} 2G_i &\underset{k_{fast}}{\overset{k_{slow}}{\rightleftharpoons}} I_{G,i} \\ I_{G,i} + R_j &\xrightarrow{k_{fast}} 2G_i + G_j, \\ 2B_i &\underset{k_{fast}}{\overset{k_{slow}}{\rightleftharpoons}} I_{B,i} \end{aligned} \tag{2}$$

$$\begin{aligned} I_{B,i} + G_j &\xrightarrow{k_{fast}} 2B_i + B_j, \\ 2R_i &\underset{k_{fast}}{\overset{k_{slow}}{\rightleftharpoons}} I_{R,i} \\ I_{R,i} + B_j &\xrightarrow{k_{fast}} 2R_i + R_j, \end{aligned} \tag{3}$$

These reactions provide positive feedback, accelerating transfers once they begin. Consider the reactions:

Red-to-green Phase reactions:

$$\begin{aligned} b + R_i &\xrightarrow{k_{slow}} G_i \\ 2G_j &\underset{k_{fast}}{\overset{k_{slow}}{\rightleftharpoons}} I_{G,j} \\ I_{G,j} + R_i &\xrightarrow{k_{fast}} 2G_j + G_i \end{aligned} \tag{4}$$

where $j = 1, 2, \cdots, n$.

Green-to-blue Phase reactions:

$$\begin{aligned} r + G_i &\xrightarrow{k_{slow}} B_i \\ 2B_j &\underset{k_{fast}}{\overset{k_{slow}}{\rightleftharpoons}} I_{B,j} \\ I_{B,j} + G_i &\xrightarrow{k_{fast}} 2B_j + B_i \end{aligned} \tag{5}$$

where $j = 0, 1, \cdots, n$.

Blue-to-red Phase reactions:

$$\begin{aligned} g + B_i &\xrightarrow{k_{slow}} R_{i+1} \\ 2R_j &\underset{k_{fast}}{\overset{k_{slow}}{\rightleftharpoons}} I_{R,j} \\ I_{R,j} + B_i &\xrightarrow{k_{fast}} 2R_j + R_{i+1} \end{aligned} \tag{6}$$

where $j = 1, 2, \cdots, n+1$.

The absence indicators r, g and b only persist in the absence of the corresponding color-coded signal molecules, since they are quickly consumed by the signal molecules if these are present. This feature assures that as long as any reaction in a given phase has not fired to completion, the succeeding phase cannot begin. There are only these three absence indicators r, g and b, regardless of the number of delay elements. Through these common absence indicators, the corresponding phases of *all* delay elements are ordered: all the delay elements must wait for each to complete its current phase before they can all move to the next phase.

The result is a very crisp transfer of signal values across delay elements. Figure 1(c) shows an ODE simulation of the reactions $(1) - (6)$ for two delay elements (for unitless time). It shows the expected alternation of the phases of the transfer, from X to Y through red, green and blue. Here $k_{fast} = 1000$ and $k_{slow} = 1$. These results hold for a wide range of rates; the transfer characteristics are independent of the specific rates.

As discussed in [2], we can use delay elements together with computational constructs to implement general circuit functions. We are exploring the mechanism of DNA strand-displacement as an experimental chassis.

4. REFERENCES

[1] H. Jiang, A. P. Kharam, M. D. Riedel, and K. K. Parhi. A synthesis flow for digital signal processing with biomolecular reactions. In *IEEE International Conference on Computer-Aided Design*, pages 417–424, 2010.

[2] H. Jiang, M. D. Riedel, and K. K. Parhi. Synchronous sequential computation with molecular reactions. In *Design Automation Conference*, 2011.

[3] P. Senum and M. D. Riedel. Rate-independent biochemical computational modules. In *Proceedings of the Pacific Symposium on Biocomputing*, 2011.

[4] A. Shea, B. Fett, M. D. Riedel, and K. Parhi. Writing and compiling code into biochemistry. In *Proceedings of the Pacific Symposium on Biocomputing*, pages 456–464, 2010.

Tuesday – June 7th

2:00pm - 3:30pm: Joint IWBDA/DAC Session
Adam Arkin, "Scalable Parts Families, Context, and Computational Design for Gene Expression Engineering"
Chris Voigt, "Gene and Cellular Circuit Design"
Erik Winfree, "A Verifying Compiler for DNA Chemical Reaction Networks"

Joint DAC/IWBDA Special Session Design and Synthesis of Biological Circuits

Douglas Densmore
Boston University
Boston, MA

Mark Horowitz
Stanford
Stanford, CA

Smita Krishnaswamy
Columbia
New York, NY

Xiling Shen
Cornell
Ithaca, NY

Adam Arkin
University of California, Berkeley
Berkeley, CA

Erik Winfree
California Institute of Technology
Pasadena, CA

Chris Voigt
University of California, San Francisco
San Francisco, CA

ABSTRACT

With the growing complexity of synthetic biological circuits, robust and systematic methods are needed for design and test. Leveraging lessons learned from the semiconductor and design automation industries, synthetic biologists are starting to adopt computer-aided design and verification software with some success. However, due to the great challenges associated with designing synthetic biological circuits, this nascent approach has to address many problems not present in electronic circuits. In this session, three leading synthetic biologists will share how they have developed software tools to help design and verify their synthetic circuits, the unique challenges they face, and their insights into the next generation of tools for synthetic biology.

Categories and Subject Descriptors

J.3 [**LIFE AND MEDICAL SCIENCES**]

General Terms

synthetic biology, systems biology, computational biology, biological circuits

Keywords

bio-design automation, genetic compiler, chemical reaction networks, molecular computation, biological parts

1. INTRODUCTION

Electronic circuit design is the process of selecting components from a group of well understood electronic components (e.g. transistors, omp amps, diodes, etc) and combining these components together to implement the desired functionality. This design process currently benefits from having well-understood, characterized components. In addition, there are rules for the composition of circuit primitives, mature manufacturing processes, and computer software to help with this process.

By contrast, biological circuit design is in its infancy. While it is temping to broadly apply techniques and analogies from electronic circuit design, it is important to respect key differences between the two disciplines. Chief among these are the need for truly orthogonal signaling mechanisms, metabolic requirements of host organisms, and the lack of control and/or observability in the systems being designed. Biological circuit design will ideally integrate algorithms and techniques which help to address these issues while leveraging biological systems' ability to adapt, evolve, and be self maintaining/repairing. In addition, biological circuit design methodologies must also leverage biophysical modeling tools to create predictive design software to tune designs for specific performance in the face of varied, often loosely characterized primitives.

This special session begins to discuss these challenges and lay the foundation for designing biological circuits in a more disciplined manner. The first talk is about biological part characteristics and how to use these parts to create controllers for gene expression engineering. The second talk is about creating genetic compilers that incorporate biophysical models and promise to raise the level of abstraction at which biological systems are designed. Finally, the third talk is about techniques to create chemical reaction networks in order to provide a new computational paradigm in this space.

DAC 2011, June 5-10, 2011, San Diego, California, USA.

2. SPEAKER SUMMARIES

Scalable Parts Families, Context, and Computational Design for Gene Expression Engineering

Adam Arkin (Speaker)
Univ. of California, Berkeley

Our current ability to engineer biological circuits is hindered by design cycles that are costly in terms of time and money. Constructs invariably fail to operate as desired, or evolving away from the desired function once deployed. Synthetic biologists seek to understand biological design principles and use them to create technologies that increase the efficiency of the genetic engineering design cycle. Central to the approach is the creation of biological parts—encapsulated functions that can be composited together to create new pathways with predictable behaviors. We have defined five desirable characteristics of biological parts—independence, reliability, tunability, orthogonality and composability. We propose that the creation of appropriate sets of families of parts with these properties is a prerequisite for efficient, predictable engineering of new function in cells and will enable a large increase in the sophistication of genetic engineering applications.

We demonstrate these concepts with examples of gene expression controllers that exercise these properties and point to how the engineering goals of synthetic biology can be met. Using the 5' UTR as an controlling switchboard for gene regulation we show how a specially chosen set of families of RNA-based parts can create a nearly complete set of controllers for gene expression engineering. We further show how they composite together and are affected by the cellular context in which they find themselves. The latter measures point to host systems that might be engineered to increase the independence of our engineered systems from host physiology.

While there are many applications for these in basic metabolic engineering and production host optimization, we further argue that the true power of such a framework is only realized when engineering the complex behaviors of cells, such as required for operation beyond the bioreactor for applications, for example, in agriculture, cell/virus-based therapies, and bioremediation. We discuss a few of our results in this area.

Gene and Cellular Circuit Design

Chris Voigt (Speaker)
Univ. of California, San Francisco

A genetic compiler for synthetic biology would enable biological systems to be specified in a high-level language (akin to Java or Verilog), which is then automatically converted into a DNA sequence. I will describe several advances in our lab towards this goal. First, genetic circuits need to be designed specifically for assembly by computational algorithms. They need to be simple, modular, and easily rewired. To this end, we have developed several logic gates that can be layered. I will describe work to use DNA synthesis to rapidly create many orthogonal variants of these gates. Second, methods need to be developed to computationally predict sequences that will properly connect these gates. We have developed biophysical models that predict the strength of genetic parts to aid this process. Finally, relevant methods need to be developed to convert a desired higher-level language into integrated gates. Together, these approaches move towards the dream of being able to program living cells.

Compiling and Verifying DNA-Based Chemical Reaction Network Implementations

Seung Woo Shin
Erik Winfree (Speaker)
California Institute of Technology

One goal of molecular programming and synthetic biology is to build chemical circuits that can control chemical processes at the molecular level. Remarkably, it has been shown that synthesized DNA molecules can be used to construct complex chemical circuits that operate without any enzyme or cellular component. However, designing DNA molecules at the individual nucleotide base level is often difficult and laborious, and thus formal chemical reaction networks (CRNs) have been proposed as a higher-level programming language. So far, several general-purpose constructions have been described for designing synthetic DNA molecules that simulate the behavior of arbitrary CRNs, and many more are being actively investigated.

Here, we solve two problems related to this topic. First, we present a general-purpose CRN-to-DNA compiler that can apply user-defined compilation schemes for translating formal CRNs to domain-level specifications for DNA molecules. In doing so, we develop a language in which such schemes can be concisely and precisely described. This compiler can greatly reduce the amount of tedious manual labor faced by researchers working in the field. Second, we present a general method for the formal verification of the correctness of such compilation. We first show that this problem reduces to testing a notion of behavioral equivalence between two CRNs, and then we construct a mathematical formalism in which that notion can be precisely defined. Finally, we provide algorithms for testing that notion. This verification process can be thought of as an equivalent of model checking in molecular computation, and we hope that the generality of our verification techniques will eventually allow us to apply them not only to DNA-based CRN implementations but to a wider class of molecular programs.

Biographical Information

Adam Arkin

Adam Arkin is a leading authority on the evolutionary design principles of cellular networks and populations and their application to systems and synthetic biology. Arkin received his undergraduate degree in chemistry from Carleton College in 1988 and his Ph.D. in physical chemistry four years later from the Massachusetts Institute of Technology. He then pursued postdoctoral studies at Stanford University in chemistry with John Ross and in developmental biology with Harley McAdams and Lucy Shapiro. In 2000 he was featured in a special edition of Time magazine on “Future Innovators,” is a member of the first class of the Technology Reviews TR100, and in 2007 he was elected as a Fellow in the American Academy of Microbiology.

The thrust of Arkin’s research has focused on developing the physical theory, computational tools and experimental approaches for understanding cellular processes critical to life. The goal is to provide a framework that will facilitate the design and engineering of new functions and behaviors in cells through synthetic and systems biology. He once compared synthetic biology to computer design and said, “Most genetic engineering has been done by hook-or-by-crook. It takes a lot of trial-and-error to build simple things into cells, like the ability to produce a lot of a functional protein. We want to actually program cells as if they're computers or design them as if they were advanced aircraft so they can do much more complicated tasks of benefit to society."

Arkin has been serving Physical Biosciences Division as the head of its Synthetic Biology Department until his promotion in May to Division Director. In addition, he directs the Joint BioEnergy Institute’s Bioinformatics Group and Berkeley Lab’s Virtual Institute of Microbial Stress. He is a Professor of Bioengineering at the University of California (UC), Berkeley and was an investigator with the Howard Hughes Medical Institute (HHMI) until 2007.

Christopher Voigt

Christopher Voigt is an Associate Professor in the Department of Biological Engineering at the Massachusetts Institute of Technology. He holds a joint appointment as a Chemist Scientist at Lawrence Berkeley National Labs, is an Adjunct Professor of Chemical Engineering at the Korea Advanced Institute of Science and Technology (KAIST), and an Honorary Fellow at Imperial College.

Prior to joining MIT, he received his BSE in Chemical Engineering from the University of Michigan (1998), a PhD in Biochemistry/Biophysics at the California Institute of Technology (2002), performed postdoctoral work in the Bioengineering Department of the University of California - Berkeley (2003), and was a faculty member in the Department of Pharmaceutical Chemistry at the University of California - San Francisco (2003-2011).

Erik Winfree

Erik Winfree is Professor of Computer Science, Computation & Neural Systems and Bioengineering at Caltech. He is the recipient of the Feynman Prize for Nanotechnology (2006), the NSF PECASE/CAREER Award (2001), the ONR Young Investigators Award (2001), a MacArthur Fellowship (2000), the Tulip prize in DNA Computing (2000), and MIT Technology Review's first TR100 list of "top young innovators" (1999).

Prior to joining the faculty at Caltech in 1999, Winfree was a Lewis Thomas Postdoctoral Fellow in Molecular Biology at Princeton, and a Visiting Scientist at the MIT AI Lab. Winfree received a B.S. in Mathematics and Computer Science from the University of Chicago in 1991, and a Ph.D. in Computation & Neural Systems from Caltech in 1998.

Tuesday – June 7th

4:00pm - 5:00pm: Tech Talks Session 5 - *Parts and Standardization*
5BDA.1 **Evolution of SBOL- design information exchange standard** Michal Galdzicki, Cesar A. Rodriguez, Laura Adam, J. Christopher Anderson, Deepak Chandran, Douglas Densmore, Drew Endy, John H. Gennari, Raik Gruenberg, Timothy Ham, Matthew Lux, Akshay Maheshwari, Barry Moore, Chris J. Myers, Jean Peccoud, Nicholas Roehner, Guy-Bart Stan, Mandy Wilson and Herbert M. Sauro
5BDA.2 **Automated design of Synthetic Gene Circuits through Linear Approximation and Mixed Integer Optimization** Linh Huynh, John Kececioglu and Ilias Tagkopoulos

Evolution of SBOL- design information exchange standard

Michal Galdzicki
Biomedical and Health Informatics,
University of Washington
1959 NE Pacific St
Seattle, WA
mgaldzic@uw.edu

Cesar A. Rodriguez‡
BIOFAB
5885 Hollis St
Emeryville, CA 94608
cesarr@berkeley.edu

Herbert M. Sauro
Department of Bioengineering,
University of Washington
3720 15th Ave NE
Seattle, WA 98195
hsauro@uw.edu

ABSTRACT
The Synthetic Biology Open Language (SBOL) offers the ability to represent information describing synthetic DNA construct designs. Here we describe the evolution of SBOL, an open community project, to define a standard for electronic data exchange of synthetic biology information. We define a data model, its representation as SBOL-semantic, and SBOL-visual, a controlled vocabulary with corresponding symbols for the terms in the vocabulary. To reduce the cost of adoption we provide libSBOLj a Java implementation of SBOL. We report on the progress made in the development to provide data exchange capabilities for biological design and management tools.

1. INTRODUCTION
Current biological design research is limited by the availability of high quality descriptions of biological components. This dearth of biological engineering information is both a consequence of the early stages of research which aims to define the structural and functional characteristics needed and the inability of software tools to access information that does exist. Researchers in synthetic biology design, manage, and analyze their synthetic biological systems using software. But, these tools do not make efficient use of previously created DNA components.

Participants of the iGEM competition have generated a large collection of DNA components (partsregistry.org) amenable to standard BioBrick™ assembly techniques. The Parts Registry now provides XML-based downloads of the information; however, these components are scarcely characterized functionally. Furthermore, with every synthetic biology publication new DNA components are described, but the information required to use them in a new design is currently present as computationally inaccessible text, if at all [12]. More recently, the BIOFAB began to perform consistent, large scale, quantitative characterization of the behavior of professionally produced DNA components (biofab.org). To enable rational engineering of biological systems, such information is critical. Specifically, to improve automated design capabilities both quantitative mathematical models [10] and DNA sequence description is necessary. But for software design tools to interpret component descriptions from across sources, there is a need for their standardized representation.

Here we present the work of the Synthetic Biology Data Exchange group to define the Synthetic Biology Open Language (SBOL). This framework provides the capability to represent and exchange the descriptions of synthetic biological components and systems.

‡ For complete author list see Additional Authors

2. STANDARD FOR DATA EXCHANGE
The most recent evolution of SBOL as a community standard [8] builds on prior work to define PoBoL [5], incorporates lessons learned from the development of the SBPkb [6], and conforms to the specification developed at the January 2011 workshop at the Virginia Bioinformatics Institute. Using this participatory design process we established the criteria for the SBOL core data model.

2.1 Data Model
The updated elements and structure of the SBOL data model define information to describe DNA sequences intended for engineering novel synthetic biological systems. The model, see Figure 1, is organized around the *DNA Component*, which describes a *DNA sequence* intended to be used as an engineering element. The *DNA Component* object may have *Sequence Annotations* which describe the positions and orientations of the *Sequence Features*. A *Sequence Feature* describes the primary annotation of a sequence and its type, as defined by the Sequence Ontology [4]. The *Library* object serves as an organizational container and describes the collection of *DNA Components* and *Sequence Features* as a set. (For the complete model, please visit sbolstandard.org).

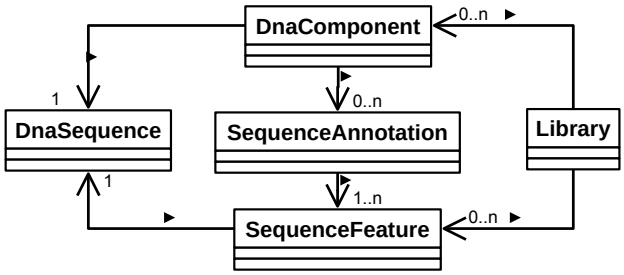


Figure 1. Schema of SBOL core model classes specifies the composition of information objects for data exchange.

2.2 SBOL-semantic
We express the data model for exchange using RDF, the W3C recommendation for representing information on the Web. This standard encoding allows the use of generic RDF tools to read, manipulate, and interpret SBOL data. For example, RDF provides the ability to use the standard query language SPARQL to retrieve DNA components from a repository of SBOL data as demonstrated using the SBPkb [6].

2.3 SBOL-visual
We are also developing a controlled vocabulary for describing DNA sequence-level features of a synthetic biology design. All the terms in the vocabulary have corresponding graphical symbols that enable unambiguous visual communication of a design. This combination of a controlled vocabulary and corresponding symbols for the terms in the vocabulary is called SBOL-visual

[13]. Please visit sbolstandard.org for a complete listing of the vocabulary and symbols. SBOL-visual is already in use in the graphical user interfaces of CAD tools such as TinkerCell [2], GenoCAD [1], and SynBioSS [14].

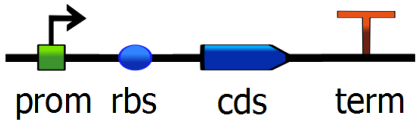


Figure 2. SBOL-visual symbols representing a constitutive promoter, ribosome binding site (RBS), coding sequence (CDS), and transcriptional terminator.

2.4 libSBOLj - Implementation

To simplify the adoption of SBOL for CAD developers, we developed a Java library, libSBOLj. The SBOL data model is implemented as Java objects managed by the Empire implementation of Java Persistence API for RDF [7]. Our library allows simple serialization and de-serialization of data to and from RDF/XML. We also provide the utility to express the object model as a JavaScript Object Notation (JSON), a popular data-interchange format. We plan to develop support for other programing languages in the future as part of our ongoing implementation work. The libSBOLj source code is available on GitHub (http://mgaldzic.github.com/libSBOLj).

3. CONCLUSIONS

Our main goal is to promote reuse of DNA components from which new and progressively complex systems can be built. To enable synthetic biologists to reap the benefits of design at higher-levels of abstraction, the Synthetic Biology Data Exchange group is committed to adopting and promoting SBOL as a standard representation both electronically and visually. We are currently working to embed SBOL capabilities into biological design and management tools, such as BIOFAB Data Access Client and Web Service, GD-ICE [9], Clotho [3], TinkerCell [2], iBioSim [11], and GenoCAD [1]. These SBOL-compliant tools should help engineers to reuse DNA components in new designs and apply them to solve novel problems.

4. ACKNOWLEDGMENTS

Our thanks to the Synthetic Biology Data Exchange Group, members of the Sauro and Peccoud Labs, Michael Grove, and Daniel L. Cook for their help throughout this project.

5. ADDITIONAL AUTHORS

Laura Adam (Virginia Bioinformatics Institute, Virginia Polytechnic Institute and State University, email: ladam@vbi.vt.edu), J. Christopher Anderson (Dept of Bioengineering, University of California Berkely, email: jcanderson@berkeley.edu), Deepak Chandran (Dept of Bioengineering, University of Washington, email: deepakc@u.washington.edu), Douglas Densmore (Dept of Electrical and Computer Engineering, Boston University, email: dougd@bu.edu), Drew Endy (BIOFAB and Dept of Bioengineering, Stanford University, email: endy@stanford.edu), John H Gennari (Biomedical and Health Informatics, University of Washington, email: gennari@uw.edu), Raik Gruenberg (email: raik.gruenberg@gmail.com), Timothy Ham (Joint BioEnergy Institute, email: tsham@lbl.gov), Matthew Lux (Virginia Bioinformatics Institute, Virginia Polytechnic Institute and State University, email: mlux@vbi.vt.edu), Akshay Maheshwari (email: ajmahesh@ucsd.edu), Barry Moore (Dept of Human Genetics, University of Utah, email: bmoore@genetics.utah.edu), Chris J. Myers (Dept of Electrical and Computer Engineering, University of Utah, email: myers@ece.utah.edu), Jean Peccoud, (Virginia Bioinformatics Institute, Virginia Polytechnic Institute and State University, email: jpeccoud@vbi.vt.edu), Nicholas Roehner (Dept of Electrical and Computer Engineering, University of Utah, email: n.roehner@utah.edu), Guy-Bart Stan (Dept of Bioengineering, Imperial College London, email: g.stan@imperial.ac.uk), Mandy Wilson (Virginia Bioinformatics Institute, Virginia Polytechnic Institute and State University, email: mandywil@vbi.vt.edu)

6. REFERENCES

[1] Cai, Y., Wilson, M.L., Peccoud, J. GenoCAD for iGEM: a grammatical approach to the design of standard-compliant constructs. Nucleic Acids Res, 38 (8). 2637-2644, 2010.

[2] Chandran, D., Bergmann, F. Sauro, H. TinkerCell: modular CAD tool for synthetic biology. J Biol Eng, 3 (1). 19, 2009.

[3] Densmore, D., Devender, A.V., Johnson, M. Sritanyaratana, N., A platform-based design environment for synthetic biological systems. in TAPIA, (2009), 24-29.

[4] Eilbeck, K., Lewis, S.E., Mungall, C.J., Yandell, M., Stein, L., Durbin, R. Ashburner, M. The Sequence Ontology: a tool for the unification of genome annotations. Genome Biol, 6 (5). R44, 2005.

[5] Galdzicki, M., Chandran, D., Nielsen, A., Morrison, J., Cowell, M., Grünberg, R., Sleight, S. Sauro, H. Provisional BioBrick Language (PoBoL) BBF RFC 31, 2009. http://hdl.handle.net/1721.1/45537.

[6] Galdzicki, M., Rodriguez, C., Chandran, D., Sauro, H.M. Gennari, J.H. Standard Biological Parts Knowledgebase. PLoS ONE, 6 (2). e17005, 2011.

[7] Grove, M. Empire: RDF & SPARQL Meet JPA Semanticweb.com, 2010. http://goo.gl/1dNxM.

[8] Grunberg, R. Draft of an RDF-based framework for the exchange and integration of Synthetic Biology data BBF RFC 30, 2009. http://hdl.handle.net/1721.1/45143.

[9] Ham, T., Dmytriv, Z., Hillson, N. Keasling, J., Towards Distributed Web of Registries: Design, Implementation and Practice of the JBEI Registry. in IWBDA, (2010).

[10] MacDonald, J.T., Barnes, C., Kitney, R.I., Freemont, P.S. Stan, G.-B. Computational design approaches and tools for synthetic biology. Integr. Biol., 3 (97-108), 2011.

[11] Myers, C.J., Barker, N., Jones, K., Kuwahara, H., Madsen, C. Nguyen, N.-P.D. iBioSim: a tool for the analysis and design of genetic circuits. Bioinformatics, 25 (21). 2848-2849, 2009.

[12] Peccoud, J., Anderson, J.C., Chandran, D., Densmore, D., Galdzicki, M., Lux, M.W., Rodriguez, C.A., Stan, G.-B. Sauro, H.M. Essential information for synthetic DNA sequences. Nat Biotech, 29 (1). 22, 2010.

[13] Rodriguez, C., Bartram, S., Ramasubramanian, A. Endy, D. Synthetic Biology Open Language Visual (SBOLv) Specification. BBF RFC 16, 2009.

[14] Weeding, E., Houle, J. Kaznessis, Y.N. SynBioSS designer: a web-based tool for the automated generation of kinetic models for synthetic biological constructs. Brief Bioinform, 11 (4). 394-402, 2010.

Automated Design of Synthetic Gene Circuits through Linear Approximation and Mixed Integer Optimization

Linh Huynh
Department of Computer Science
University of California, Davis
huynh@ucdavis.edu

John Kececioglu
Department of Computer Science
University of Arizona, Tucson
kece@cs.arizona.edu

Ilias Tagkopoulos
Department of Computer Science & UC Davis Genome Center
University of California, Davis
itagkopoulos@ucdavis.edu

ABSTRACT

Synthetic biology aspires to revolutionize the way we construct biological circuits, as it promises fast time-to-market synthetic systems through part standardization, model abstraction, design and process automation. However, the automated design of synthetic circuits remains an unsolved problem, despite the increasing number of practitioners in the field. One reason behind that, is the absence of an efficient mathematical formulation for the combinatorial optimization problem of selecting genes and promoters when synthesizing the candidate circuits. Here, we propose an optimization framework that is based on a linear relaxation of the non-linear optimization problem, which proves to be a good approximation of the non-linear dynamics present in biological systems. Further evaluation of the proposed framework in a real non-linear synthetic circuit (a toggle switch), and with the use of a mutant promoter library, resulted in a rapid and reproducible convergence to a synthetic circuit that exhibits the desired characteristics and temporal expression profiles. This work is a step towards a unifying, realistic framework for the automated construction of biological circuits with desired temporal profiles and user-defined constraints.

1. INTRODUCTION

When it comes to automated biological circuit design, CAD tools are still in their infancy despite notable developments in the field. In this context, the use of mathematical optimization has been very limited [2] and with mixed results, while the main challenge still remains: how can we develop algorithms that cope with the combinatorial explosion and complex models that describe biological behavior? Here, we introduce a novel optimization formulation for synthetic circuit design that finds the optimal part configuration, given a library of biological parts, an objective function (e.g. the desired temporal profile of the output protein), user-defined constraints (e.g. circuit size), and an existing topology that provides connectivity (e.g. gene A must positively regulate gene B) but not individual parts (e.g. gene A, gene B, regulation strength).The optimization method translates the circuit design problem into a nonlinear integer programming formulation that it solves using spatial branch and bound techniques.

2. METHODS

Linear formulation: For the current analysis, assume a database of parts that has m promoters and n proteins. We introduce the following equation to express the concentration of protein i as a function of the available promoters and proteins:

$$\frac{df_i}{dt} = \sum_{\substack{j=1 \\ j \neq i}}^{n} \sum_{k=1}^{m} a_{jk} y_{ik} f_j - (d_i + \mu) f_i + b_i \tag{1}$$

where the parameter a_{jk} is proportional to the production rate of protein i if protein j is bound at the promoter k upstream of gene i. Parameter d_i captures both the *degradation and auto-regulation* of protein i. The y_{ik} are binary variables defined as:

$$y_{ik} = \begin{cases} 1 & \text{If promoter } k \text{ is upstream of protein } i \\ 0 & \text{Otherwise} \end{cases}$$

Furthermore, we can add inducers in the system, by adding the term $-K_{inducer} f_i$ into equation 1. The solution of the linear ODE system is given by

$$\dot{F} = AF + B \tag{2}$$

where the elements of the A matrix are defined as

$$A_{ij} = \begin{cases} \sum_{k=1}^{m} a_{jk} y_{ik} & If\ i \neq j \\ -d_i - \mu & If\ i = j \end{cases} \tag{3}$$

and with F and B given as

$$F = (f_1(t), f_2(t), ..., f_n(t))^T,\ B = (b_1, b_2, ..., b_n)^T$$

Assuming that A is diagonalizable, there exists a matrix $S = (s_{ij})$, and a diagonal matrix D with its diagonal elements the eigenvalues of the system, i.e. $(\lambda_1, \lambda_2, ..., \lambda_n)$. Then $S^{-1}AS = D$ and thus $S^{-1}\dot{F} = DS^{-1}F + S^{-1}B$. Substituting $G = S^{-1}F$ and $E = S^{-1}B$, we end up with $\dot{G} = DG + E$, which has the following solution:

$$g_i(t) = C_i e^{\lambda_i t} - \frac{E_i}{\lambda_i} \tag{4}$$

As expected, the solution of a linear ODE system is non-linear, and thus can describe dynamics of basic biological functions, since the latter are usually expressed with exponential functions. Mapping back to F, and since $F = SG$ we end up with the solution :

$$f_i(t) = \sum_{j=1}^{n} s_{ij} g_j(t) \tag{5}$$

Objective function: Our optimization framework will try to obtain the parts set that minimize the difference between the desired temporal profile and the actual one. As such, if $f_p(t)$ and $f_p^*(t)$ are the estimated concentration and the desired concentration of protein p at the time point t, respectively, then our objective function is the following:

$$Z = \sum_{t \in T} (f_p(t) - f_p^*(t))^2 \tag{6}$$

Linear constraints: The user may add additional constraints to the optimization system. For example, if a specific gene i should be included (or conversely, should be absent) in the design, then we can introduce a binary variable x_i, that will denote the presence or absence of gene i in the final circuit. In addition, the user may restrict the number of promoters for any given gene, limit the number of genes per promoter, or disallow large polycistronic promoters in the circuit:

$$\sum_{k=1}^{m} y_{ik} \leq M_1 x_i \quad \forall i = 1...n \tag{7}$$

$$\sum_{i=1}^{n} y_{ik} \leq M_2 \quad \forall k = 1...m \tag{8}$$

$$\sum_{k=1}^{m} y_{ik} \geq x_i \quad \forall i = 1...n \tag{9}$$

Constraints are also in place due to stability issues of the resulting circuit. In order for the system to have stable dynamics, all eigenvalues must be distinct and their real parts must be negative. This leads to the following constraint:

$$Re(\lambda_i) \leq -\varepsilon \; and \; ||\lambda_i - \lambda_j|| \geq \varepsilon \quad \forall i \neq j = 1...n \tag{10}$$

Finally, after the addition of standard diagonalization equations, normalization of eigenvalue vector space, and the addition of boundary conditions as constraints on the initial concentration of each protein i, the concentration of the desired protein is given by:

$$\sum_{j=1}^{n} (C_j e^{\lambda_j t} - E_j/\lambda_j) = f_p(t) \tag{11}$$

Optimization problem : Now that we have defined all constraints in our system, we can formulate the optimization problem as solving for variables x_i and y_{ik} so that

$$\textbf{Minimize } Z \; subject \; to \; (4), (6) - (11)$$

This is a mixed integer non-linear programming (MINLP) problem which can be efficiently solved in practice using spatial branch & bound (e.g. Couenne [1]).

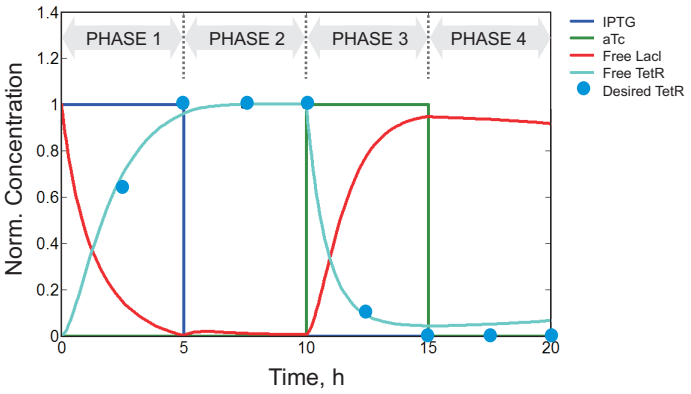


Figure 1: Expression profile of the resulting synthetic circuit, with promoters T7 (in upstream of LacI) and L3 (in upstream of TetR). The desired profile (input, depicted with blue dots) and actual profile (cyan line) for the TetR protein is shown. The temporal profile was split into four phases, based on changes in the inducer concentrations. Phase 1: IPTG high, aTc low; Phase 2: IPTG low, aTc low; Phase 3: IPTG low, aTc higt; Phase 4: IPTG low, aTc low.

3. RESULTS

To evaluate the capacity of our optimization framework, we assessed its performance in the case of a toggle switch design [4]. As an input to our optimization framework, we collected a mutant library for the TetO and LacO promoter that was experimentally characterized recently [3]. The values of a_{jk} are estimated from data in this library and other parameters such as degradation rates and association constants are chosen from [2]. As shown in figure 1, the system was able to find a set of parts (promoters T7 and L3, upstream of LacI and TetR, respectively) that led to a synthetic circuit that approximates well the desired transient dynamics.We further simulated these solutions, which resulted to circuits that also exhibit flip-flop characteristics at various degrees. From the complete solution space, less than 2% (7 sets) had this property.

4. REFERENCES

[1] P. Belotti, J. Lee, L. Liberti, F. Margot, and A. Waechter. Branching and bounds tighteningtechniques for non-convex MINLP. *Optimization Methods and Software*, 24(4):597–634, 2009.

[2] M. Dasika and C. Maranas. OptCircuit: an optimization based method for computational design of genetic circuits. *BMC Systems Biology*, 2(1):24, 2008.

[3] T. Ellis, X. Wang, and J. Collins. Diversity-based, model-guided construction of synthetic gene networks with predicted functions. *Nature biotechnology*, 27(5):465–471, 2009.

[4] T. Gardner, C. Cantor, and J. Collins. Construction of a genetic toggle switch in Escherichia coli. *Nature*, 403(6767):339–342, 2000.

Poster Session Abstracts

Low-Power Broadcast Electrode-Addressing for Disposable Digital Microfluidic Biochips Tsung Wei Huang and Tsung-Yi Ho
Using user-defined semantic languages in synthetic biology: generating DNA compilers Laura Adam and Jean Peccoud
Integration of Standardized Cloning Methodologies and Sequence Handling to Support Synthetic Biology Studies Maurice H T Ling, Dunqiang Liao, Angela Jean, Beng B Y Tew, Shanice Ho and Kevin P Clancy
Network Reconstruction Using Literature Curated and High Throughput Data Vishwesh Kulkarni, Kalyanasundaram Subramanian, Reza Arastoo, Mayuresh Kothare and Marc Riedel
ShReDing Biochemical Networks to Uncover Hierarchical Modularity Gautham Sridharan, Kyongbum Lee and Soha Hassoun
Pareto Optimal Fronts in Bacterial Knockout Strategies Jole Costanza, Luca Zammataro, Pietro Liò and Giuseppe Nicosia
Optimal Design of the Mitochondrial Bioenergetics Jole Costanza, Luca Zammataro, Pietro Liò and Giuseppe Nicosia
OREMPDB, A Semantic Dictionary of runnable biological circuits; system overview and first queries in the neurology field Renato Umeton, Raffaella Pizzolato and Giuseppe Nicosia
Stochastic Resonance Optimization of a Genetic Quorum-Mediated Trigger Circuit Eduardo Monzon, Chris Winstead, Abiezer Tejeda and Charles Miller
Applying Hardware Description Languages to Genetic Circuit Design Roza Ghamari, Brynne Stanton, Traci Haddock, Swapnil Bhatia, Kevin Clancy, Todd Peterson, Christopher Voigt and Douglas Densmore
Design, Implementation, and Deployment of Electronic Datasheets Cesar Rodriguez and Michal Galdzicki

Low-Power Broadcast Electrode-Addressing for Disposable Digital Microfluidic Biochips

Tsung-Wei Huang and Tsung-Yi Ho*
Department of Computer Science and Information Engineering
National Cheng Kung University, Tainan, Taiwan
twhuang@eda.csie.ncku.edu.tw; tyho@csie.ncku.edu.tw

ABSTRACT

The number of independent input pins used to control the electrodes in digital microfuidic biochips is an important cost-driver in the emerging marketplace, especially for medical purpose where chips here tend to be disposable for cross-contamination avoidance. A promising pin-count reduction technique, *broadcast addressing*, reduces the pin count by assigning a single control pin to multiple electrodes with mutually-compatible control signals. Prior works utilize this addressing scheme by minimally grouping electrode sets with non-conflict signal merging. However, merging control signals also introduces redundant actuations, which potentially cause a high power-consumption problem. Recent studies on PDMFBs have indicated that high power consumption not only decreases the product lifetime but also degrades the system reliability. Unfortunately, this power-aware design concern is still not readily available among current design automations of PDMFBs. To cope with these issues, we propose in this paper the *first* power-aware broadcast addressing for PDMFBs. Our algorithm simultaneously takes pin-count reduction and power-consumption minimization into consideration, thereby achieving higher integration and better design performance. Experimental results demonstrate the effectiveness of our algorithm.

1. INTRODUCTION

As the microfluidic technology advances, digital microfluidic biochips (DMFBs) have attracted much attention recently. These miniaturized and automated DMFBs provide various advantages including high portability, high throughput, high sensitivity, high immunity to human intervention, and low sample volume consumption. Due to these advantages, more and more practical applications such as infant health care, point-of-care disease diagnostics, environmental toxin monitoring, and drug discovery have been successfully realized on DMFBs [2, 4, 6].

*This work was partially supported by the National Science Council of Taiwan ROC under Grant No. NSC 99-2220-E-006-005 and 99-2221-E-006-220. We would like this abstract considered for oral presentation.

IWBDA'11, June 6-7, 2011, San Diego, California, USA

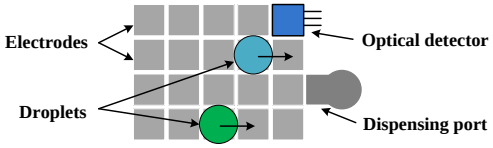


Figure 1: Schematic view of a digital microfluidic biochip.

Typically, a DMFB consists of a two dimensional (2D) electrode array, optical detector, and dispensing port, as schematically shown in Figure 1 [4]. In performing fluidic-handling functions, droplet-based operations are introduced on DMFB platforms. By generating electrohydrodynamic forces from electrodes, droplets can be dispensed from dispensing ports, moved around the 2D array for performing reactions (e.g., mixing or dilution), and then moved toward the optical detector for detection [5]. The entire operations are also called *reconfigurable* operations due to their flexibility in area and time domain [1].

In realizing fluidic controls, a primary issue is the control scheme of electrodes. To correctly control the electrodes, *electrode addressing* is introduced as a method through which electrodes are assigned by control pins to identify input signals. Early DMFB designs relied on *direct addressing*, where each electrode is directly and *independently* assigned by a dedicated control pin [3], as illustrated in Figure 2(a). This addressing maximizes the flexibility of electrode controls. However, for large arrays, the high pin-count demand complicates the electrical connections between the chip and the external controller, thus rendering this kind of chip unreliable and prohibitively expensive to package and manufacture [3, 4, 7].

Recently, *pin-constrained* DMFBs (PDMFBs) have raised active discussions to overcome this problem. One of the major approaches, *broadcast addressing*, provides high throughput for bioassays and reduces the number of control pins by identifying and connecting them with *compatible* control signals. In other words, multiple electrodes are controlled by a single signal source and are thus actuated simultaneously, as shown in Figure 2 (b). In this regard, much on-going effort has been made to group sets of electrodes that can be driven uniformly without introducing any signal conflict [7].

Although broadcast addressing serves as a promising solution to pin-constrained designs, yet the redundant actuations during signal merging have potentially caused a power-consumption problem. For example, in Figure 2(a), the direct-addressing result needs two exact actuations for mov-

ing the two droplets in this time step. In Figure 2(b), after applying the broadcast addressing, the pin count is greatly reduced from 20 to 7. Nevertheless, the addressing result needs two exact actuations, plus two *redundant* actuations, for moving the two droplets. As electrodes are controlled in a series of time steps, if control pins are not carefully assigned to electrodes, the addressing result will introduce a great number of redundant actuations. Hence, executing a bioassay may incur a high power-consumption problem [8].

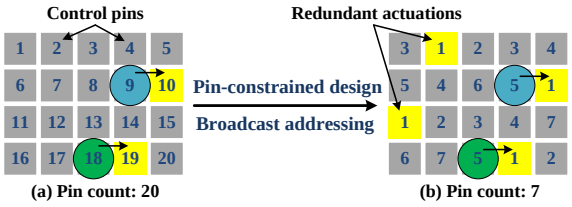


Figure 2: Moving two droplets in a specific time step. (a) A direct-addressing result uses two pins (pin 10 and pin 19) to generate two exact actuations. (b) A broadcast-addressing result uses one pin (pin 1) to generate two exact actuations, plus two redundant actuations.

As reported in recent studies, the power-consumption problem is especially critical for battery-driven applications, such as hand-held devices for point-of-care diagnosis and battery-operated sensors for environmental monitoring [2, 8]. Since these applications often require longer execution time, it is desirable to minimize the power consumption for longer battery lifetime. Besides, high power consumption reveals a fact of excessive actuations, which accelerates the dielectric breakdown of some electrodes. Such defects may result in unexpected executions and thus degrade the system reliability [1, 4].

Unfortunately, current broadcast addressing for PDMFBs neglects the induced number of redundant actuations during signal merging and pin sharing, which causes a significant power-consumption problem. As reported in [7], even the simplest broadcast addressing with pin-count minimization has been presented as NP-hard. And thus the design convergence imposed by simultaneously minimizing the pin count and power consumption has become the most difficult challenge. Due to the distinct nature from traditional VLSI technology, a specialized tool must be developed to solve this problem efficiently and effectively such that PDMFBs can be more feasible for practical applications.

1.1 Our Contributions

In this paper, we propose the *first* power-aware broadcast addressing for PDMFBs. By considering both pin-constrained and power-aware design issues, our algorithm can simultaneously minimize the pin count and power consumption to achieve high design performance. The contributions of this paper are summarized as follows.

- We introduce a new problem formulation of power optimization for broadcast-addressing PDMFBs. We also propose the *first* addressing algorithm to minimize the power consumption while considering the pin-count reduction. Our addressing technique comprehensively take power and pin-count saving issues into consideration to reduce the power-consumption and pin count.
- Unlike typical broadcast addressing which only deals with the compatibility for identical signals, our work can handle the integration between identical and complementary signals simultaneously. In this regard, further pin-count reduction can be achieved.
- We propose a progressive addressing algorithm based on a *minimum-cost maximum-flow* network to efficiently solve the entire power-aware addressing problem.

Experimental results demonstrate the effectiveness of our addressing algorithm. The evaluation performed on a set of real-life chip applications shows that our addressing algorithm achieves the best results in terms of pin count and power consumption.

2. ALGORITHM OVERVIEW

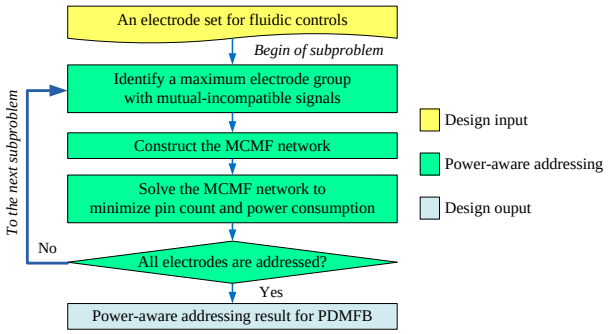


Figure 3: Overview of our algorithm.

Figure 3 shows the overview of our progressive network-flow based power-aware electrode-addressing algorithm. The essential intuition behind our algorithm is to reduce the design complexity by dividing the original problem into a set of manageable subproblems. In each subproblem, we identify a maximum electrode group with mutual-incompatible signals to facilitate the flow formulation. Then, pin-count and power-consumption minimizations are formulated to a minimum-cost maximum-flow (MCMF) network. By solving the flow network, we can optimally minimize the pin count and power consumption. Finally, iterations of subproblems end until all electrodes are addressed.

3. REFERENCES

[1] K. Chakrabarty, "Towards fault-tolerant digital microfluidic lab-on-chip: defects, fault modeling, testing, and reconfiguration," *Proc. IEEE ICBCS*, pp. 329–332, 2008.
[2] K. Chakrabarty, "Design automation and test solutions for digital microfluidic biochips," *IEEE TCS I*, pp. 4–17, 2010.
[3] J. Gong and C. J. Kim, "Direct-referencing two-dimensional-array digital microfluidics using multilayer printed circuit board," *IEEE J. MEMS*, no. 2, pp. 257–264, 2008.
[4] T.-Y. Ho, J. Zeng, and K. Chakrabarty, "Digital microfluidic biochips: A vision for functional diversity and more than Moore," *IEEE/ACM ICCAD*, pp. 578–585, 2010.
[5] J. H. Song, R. Evans, Y. Y. Lin, B. N. Hsu, and R. B. Fair, "A scaling model for electrowetting-on-dielectric microfluidic actuators," *Microfluidics and Nanofluidics*, pp. 75–89, 2009.
[6] F. Su, K. Chakrabarty, and R. B. Fair, "Microfluidics based biochips: Technology issues, implementation platforms, and design-automation challenges," *IEEE TCAD*, pp. 211–223, 2006.
[7] T. Xu and K. Chakrabarty, "Broadcast electrode-addressing for pin-constrained multi-functional digital microfluidic biochips," *Proc. IEEE/ACM DAC*, pp. 173–178, 2008.
[8] T. Xu and K. Chakrabarty, "A droplet-manipulation method for achieving high-throughput in cross-referencing-based digital microfluidic biochips," *IEEE TCAD*, pp. 1905–1917, 2008.

Using user-defined semantic languages in synthetic biology: generating DNA compilers

Laura Adam
Virginia Bioinformatics Institute (VBI)
Virginia Tech (0477)
Blacksburg, VA 24061
(540) 231-9652
ladam@vbi.vt.edu

Jean Peccoud
Virginia Bioinformatics Institute (VBI)
Virginia Tech (0477)
Blacksburg, VA 24061
(540) 231-0403
jpeccoud@vbi.vt.edu

ABSTRACT
In this paper, we describe a module that generates DNA compilers based on grammars, attributed or not, which are stored in a database. Software applications using a language-based approach can now be customized by their users.

Categories and Subject Descriptors
J.3.1 [LIFE AND MEDICAL SCIENCES]: Biology and genetics – *Biology and genetics, Specification Techniques, Design Management.*

General Terms
Algorithms, Design, Standardization, Languages, Theory, Verification.

Keywords
Code generation, compiler, attribute grammar, context-free grammar, compilation on-the-fly, synthetic biology

1. INTRODUCTION
DNA is often compared to language [5]. As a result, methods from computational linguistics have been proposed to approach the design of synthetic biological constructs. For instance, a language describing synthetic biology principles can be used to guide scientists through the design of a construct [1]. In the meantime, the construct's kinetic equations can be produced by the means of attribute grammars [2], allowing the scientists to simulate a design prior to any wet lab realization. However, such technology should be able to adapt to its users.

Indeed, synthetic biologists are likely to work with their own parts and parts' attributes, such as transcription rates, which are not fixed values for the entire community. Hence, both the words of the language and their semantic attributes change depending on the user. For even more flexibility, the users can be allowed to define their own grammars to explore a particular design space of interest. Consequently, the resulting analytic tool, the DNA compiler, should adapt to the user's defined language.

2. ARCHITECTURE OF THE COMPILER GENERATION MODULE

2.1 Components of the module
Figure 1 shows the components of the module as well as the workflow. The first step is to build the Prolog file for the compiler. The parts library ID and the grammar ID must be passed for the function to adequately query the database. A second function 'call compiler' is responsible for using the compiler. It needs the DNA sequence or the design to analyze and call Prolog with the just built compiler file to return the analysis results. The code to build a compiler relies on a few steps which are independent and can be suppressed as needed. This independence eases the modification, especially toward the maximum automation of the compiler construction, and will give more flexibility to the users, letting them perform only desired tasks (only lexical analysis for example). The modules that make up the compiler are sequential. Each of them is processed only if the previous module worked. They include:

'Write DNA': To be used if we want display the DNA sequence.

'DNA to parts': Interprets the DNA sequence into parts (lexer).

'Write parts': To be used to display the parts list

'Check grammar': Check the validity of the parts list against the grammar (parser). Returns true, and the equations (if attribute grammar), or false.

'Answer grammar': If it gets to this point in the program, it means a derivation tree could be built: the construct is valid against the grammar.

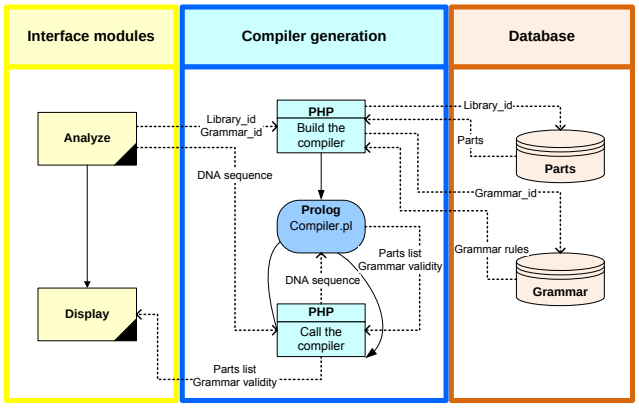


Figure 1 Overview of the compiler generation module

'DNA to parts' and 'Check grammar' are the only two modules that depend on the rules specifications and are hence to be generated from the database content, while the others may be hard-coded in the generation function to be fetched directly in the compiler file.

2.2 Storing grammars in databases
We developed a data model to store context-free grammars and, by extension, attribute grammars; Figure 2 shows the UML representation. Here, we describe the elements specific to attribute grammars. Categories can have attributes. Each part from a category has values for its category's attributes. In case the attribute is a list, this will be specified in the category attribute declaration and the parts of this category will have several entries for each of the elements from the list. The grammar contains several rules that may have a semantic action attached, i.e. a piece

of Prolog code that will be executed after the rules are applied. 'Attribute_to_pass' is a table that refers to category attributes that have to be passed by other categories, as-is, in order to be accessible in other rules where their original category did not appear: the inherited attributes. In addition, a table ('code_tbgenerated') stores the Prolog code corresponding to the rewriting of the compiler output into the target language. It is directly associated with a grammar.

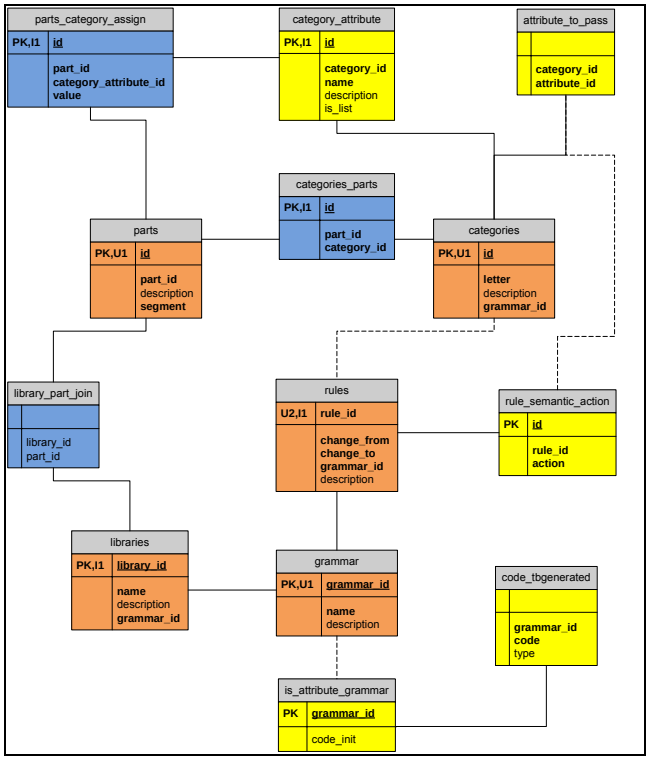


Figure 2 Data model for storing attribute grammar

3. COMPILER GENERATION ALORITHM

3.1 Syntactic compiler

The DCG syntax in SWI-Prolog is used to write the rules in the compiler. However, we need to pre-process the grammar rules before writing them.

First, the orphan rules must be removed. Orphans rules are those with a non-terminal symbol that only appear on the right side of the rules.

Among the remaining rules, the categories involved in a left recursive rule must be stored.

Finally, we write the subset of remaining rules, using the rewriting algorithm if needed, in the compiler file.

To avoid any Prolog syntax issue, all elements are designated using their database id.

3.2 Semantic compiler

The process is the same as for attribute grammars with additional steps for handling the semantics aspects of the language. The compiler code passes the parts attributes as arguments of their 'functor' i.e. category. In order to do the compilation in a single-pass fashion [3], which is easier for automatic code generation, global variables are used for computing the semantics. These variables are initiated at the start rule, which is stored in the code_init field from the table 'is_attribute_grammar'. From the computed semantics declarations stored in a variable, a fixed code, 'code_tbgenerated', specifies how to analyze the variables at the end of the compilation. For example, a list of function calls with their parameters values can be processed during the derivation and the fixed code will look at the list and execute the functions. Rules' semantic actions are used to specify what to store in the global variables. The different information related to the semantics must be stored using a Prolog syntax and will be fetched directly; no meta-language appeared to be usable for defining rules' semantic actions.

There is no solution for rewriting left recursive rules in the case of 'general' attribute grammars in a single-pass [4]: it is not clear how the semantics can be preserved. Consequently, we do not handle attribute grammars that are left-recursive.

4. CONCLUSION

All context-free grammars used in GenoCAD could be successfully used as-is to generate a compiler. A function to validate a GenoCAD design has been implemented and can be accessed from the 'My designs' screen. The attribute grammar published [2] has been stored in the data model proposed and a compiler could be generated. Yet, some constraints over the attribute grammars must be specified, such as no left-recursive rules. However, using a custom grammar editor will prevent the users to defined grammars that cannot be handled.

5. ACKNOWLEDGMENTS

The development of GenoCAD is supported by NSF Award EF-0850100. Laura Adam is supported by a fellowship from SAIC.

6. REFERENCES

1. Cai, Y., Hartnett, B., Gustafsson, C., and Peccoud, J. A syntactic model to design and verify synthetic genetic constructs derived from standard biological parts. *Bioinformatics (Oxford, England) 23*, 20 (2007), 2760-7.

2. Cai, Y., Lux, M.W., Adam, L., and Peccoud, J. Modeling structure-function relationships in synthetic DNA sequences using attribute grammars. *PLoS computational biology 5*, 10 (2009), e1000529.

3. Koskimies, K. A specification language for one-pass semantic analysis. *ACM SIGPLAN Notices 19*, 8 (1984), 179-189.

4. Lohmann, W., Riedewald, G., and Stoy, M. Semantics-preserving Migration of Semantic Rules During Left Recursion Removal in Attribute Grammars. *Electronic Notes in Theoretical Computer Science 110*, (2004), 133-148.

5. Searls, D.B. Linguistic approaches to biological sequences. *Bioinformatics 13*, 4 (1997), 333.

Integration of Standardized Cloning Methodologies and Sequence Handling to Support Synthetic Biology Studies

Maurice HT Ling
Life Technologies, Inc.
33 Marsiling Ind. Est.Rd. 3
Singapore 739256
(65) 63629415
Maurice.Ling@lifetech.com

Angela Jean
Life Technologies, Inc.
33 Marsiling Ind. Est.Rd. 3
Singapore 739256
(65) 63629416
Angela.Jean@lifetech.com

Dunqiang Liao
Life Technologies, Inc.
33 Marsiling Ind. Est.Rd. 3
Singapore 739256
(65) 63629585
Dunqiang.Liao@lifetech.com

Ben BY Tew
Life Technologies, Inc.
33 Marsiling Ind. Est.Rd. 3
Singapore 739256
(65) 63629555
BengYong.Tew@lifetech.com

Shanice Ho
Life Technologies, Inc.
33 Marsiling Ind. Est.Rd. 3
Singapore 739256
(65) 63629307
Shanice.Ho@lifetech.com

Kevin Clancy
Life Technologies, Inc.
5791 Van Allen Way
CA 92008, USA
(01) 7602688356
Kevin.Clancy@lifetech.com

ABSTRACT

The assembly and downstream transformation of genetic constructs has been a fundamental scientific technology for the last thirty years. Synthetic biology is an engineering based approach to molecular biology as emphasizing the standardized assembly of characterized DNA fragments. The standards promoted by the BioBricks™ Foundation have enabled novel constructs to be developed based upon the expected function of these constructs. However scientists need a software environment that enables them to curate large collections of parts and assemblies, combined with appropriate tools to facilitate quick creation of constructs and identification of potential design issues *in silico*. In this paper, we present the implementation of BioBrick™ and GENEART® Assembly tools, coupled with an enhanced database to manage and develop such parts collections. Integration of these tools and data into the VectorNTI® software suite is a step towards implementation of BioCAD™, a computer based design approach to facilitate development of complex circuit based perturbation of cellular systems.

Categories and Subject Descriptors

J.3 [**Computer Applications**]: Life and Medical Sciences – biology and genetics

General Terms

Algorithms, Design, Experimentation

Keywords

Bioinformatics, Synthetic biology, BioBrick, GENEART Assembly, Vector NTI, Cloning, Parts, Devices

IWBDA'11, June 6-7, 2011, San Diego, CA, USA.

1. INTRODUCTION

The use of constructs has evolved from simple molecular cloning experiments to gene reporting [1], switch testing [2] and promoter characterization [3]. Assembly of constructs typically involves a backbone vector as well as the set of DNA fragments of interest. Depending on the DNA sequence and the assembly system in place, various combinations of restriction sites can be used for the extraction and recovery of the final target fragment [4].

Through the understanding of underlying gene expression mechanisms, novel and optimized biological systems can be created [5]. Although there are many ways to assemble constructs [6], typically biologists have not developed their materials with an eye to reuse or elaboration from simple to complex systems. This large scale assembly or reuse of constructs in series of experiments are often restricted and limited to the capabilities of the selected protocol [7]; leaving little room for interoperability or extensibility for future experiments.

A set of open cloning standards promoted through the BioBricks™ Foundation [8] can be used for the typing of constructs and the standardization of assembly protocols. Availability of various assembly standards [9] using combinations of restriction sites, amongst other criteria, also provides the option of selecting an appropriate assembly protocol for a given assembly. The resulting construct will have a restriction site scar between the 2 fragments which can have functional consequences, particularly when combing protein domains [10].

The GENEART® Assembly System is a homologous recombination based cloning system. This experimental design tool takes advantage of recombination in vitro or in yeast to join pre-existing DNA fragments or chemically synthesized fragments into a single recombinant molecule [11]. As the system relies on homologous recombination, the adjacent DNA fragments must share end-terminal homology [12] as shown in Figure 1 or they must be "stitched" together by means of primers [13], known as stitching primers. This system is thus reliant upon identification of possible issues that would disrupt homologous recombination and accurate design of the different types of primers needed in the experiment.

In conjunction with the BioBrick™ standards, the Repository of Biological Parts has developed to track DNA parts that have been generated by scientists and participants in the iGEM competition over the last several years [14]. The current collection has more than 15,000 entries, however many of these entries are poorly characterized and the lack of defined, parsable data records renders much of this data difficult to discover, use or validate in another laboratory. However these parts can be assembled using either traditional BioBrick or GENEART® Assembly approaches.

Ideally synthetic biologists would use software that can assist them in development of their projects. This would require the development of software that could 1. handle collections of parts and devices during various stages of development, managing the provenance of development of novel parts, characterization of existing parts and assembly of parts into more complex devices; 2. provide expert tools that could facilitate development of projects by automating or semi automating design decisions, and; 3. provide a means to identify design issues during in silico development. Enabling such a BioCAD™ environment in Vector NTI®, a widely used commercial software package, would greatly speed both development among existing synthetic biologists as well as introducing such practices to the larger population of existing molecular biologists.

2. ALGORITHMIC APPROACHES

The Vector NTI® Suite is a comprehensive desktop bioinformatics suite. The software uses a serialized, indexed flat file database for sequence record import, storage and retrieval. The database was modified from an ASCII format to a binary format and the indexing scheme was supplemented with an improved key value index. These changes both increased the numbers of records that can be stored from 10,000 to ~100,000 as well as decreased search and retrieval times by 100%. The improved database was used to import the entirety of the Standard Repository of Biological parts. These molecules were then functionally characterized based upon their BioBrick IDs, information encoded in the header lines of the files and sequence similarities. Subsets containing functionally characterized parts were used in subsequent *in silico* development projects.

The Vector NTI Molecule Viewer® was modified to identify sets of restriction enzymes important to the BioBrick Assembly standards. These sets were used to construct custom view and search sets for part characterization. This allowed us to quickly scan new and existing parts for the presence of these sites and helped us identify mutagenesis strategies to remove such sites from these affected parts.

Construct assembly using BioBrick™ standards was implemented as described in the BioBricks standards (Life Technologies, Inc.). To reflect the current Biobrick assembly approach, paired sets of molecules are submitted for assembly. Sequential runs using progressively assembled parts gives rise to a multi-fragment construct representing complex devices.

The implementation of construct assembly using BioBrick™ standards allows the final construct to be typed and assembled according to these standards. At the same time, the provision of chassis options and strand direction of assembly also allows various options to be visualized prior to the in vitro assembly of the construct. This facilitates rapid design, prototyping and debugging of potential design errors as in silico designs are developed.

The GENEART® Assembly System is a homologous cloning based tool which also supports synthetic biology cloning projects. There are 4 modes to the system – seamless cloning, high-ordered genetic assembly resulting in construct below 60 kilobases, high-ordered genetic assembly resulting in construct above 60 kilobases, and high-ordered stitching assembly. For 3 former modes required homologous ends between adjacent fragments and allow primers to be designed for use to amplify the original fragments. Seamless cloning allows up to 4 DNA fragments plus a vector totaling up to 13 kilobases in length while high-ordered genetic assembly allows up to 10 DNA fragments plus a vector totaling up to 110 kilobases in length. The stitching assembly does not require homologous ends between adjacent ends as up to 3 sets of stitching primers can be designed, implying a maximum of 5 fragments, including the vector, to be stitched together. The resulting clone can be displayed in a circular or linear form.

Both tools were used to design and subsequently construct constructs conforming to the two cloning methodologies. BioBrick™ constructs were validated to previously reported constructs in the Parts Repository. GENEART® Assembly constructs were verified by size, restriction patterns and through end and junction sequencing studies.

3. CONCLUSION

Experimental design tools have facilitated final results to be visualized before *in vitro* experiments are performed; allowing potential issues to be identified and resolved through *in silico* means. At the same time, the results provide baselines against which *in vitro* results can be compared. The implementation of a construct assembly algorithm utilizing open standards prescribed by the BioBricks™ Foundation has not only enabled the visualization and inspection of the final construct, but allowed various hypotheses and scenarios to be carried out without significant costs. The GENEART® Assembly tool permits the construction of 100 bp to 120,000 bp constructs without the use of restriction enzymes. The improvement of the Vector NTI database permits the improved use and reuse of parts and constructs during design. This integrated system is an initial step to providing robust, scalable and easy to use access to Synthetic Biology tools.

4. REFERENCES

[1] Wesley, S. V., Helliwell, C. A., Smith, N. A., Wang, M., Rouse, D. T., Liu, Q., Gooding, P. S., Singh, S. P., Abbott, D., Stoutjesdijk, P. A., Robinson, S. P., Gleave, A. P., Green, A. G. and Waterhouse, P. M. 2001, Construct design for efficient, effective and high-throughput gene silencing in plants. The Plant Journal 27, 581–590.

[2] Gardner, T. S., Cantor C.R., Collins J.J. 2000. Construction of a genetic toggle switch in Escherichia coli. Nature 403, 339-342.

[3] Wick M., Zubov D., Hagen G.. 1999. Genomic organization and promoter characterization of the gene encoding the human telomerase reverse transcriptase (hTERT). Gene 232(1), 97-106.

[4] Karn, J. and Matthes, H.W.D. and Gait, M.J. and Brenner, S. 1984. A new selective phage cloning vector, [lambda] 2001, with sites for XbaI, BamHI, HindIII, EcoRI, SstI and XhoI. Gene 32(1-2). 217-224.

[5] Agapakis, C.M. and Silver, P.A. 2009. Synthetic biology: exploring and exploiting genetic modularity through the design of novel biological networks. Molecular BioSystems 5(7), 704--713.

[6] Hebelstrup, K.H. and Christiansen, M.W., Carciofi, M. and Tauris, B. and Brinch-Pedersen, H. and Holm, P.B. 2010. UCE: A uracil excision(USER™)-based toolbox for transformation of cereals. Plant Methods 6(1), 15.

[7] Harbers, M. 2008. The current status of cDNA cloning. Genomics 91(3) 232-242.

[8] Smolke, C.D. 2009. Building outside of the box: iGEM and the BioBricks Foundation. Nature biotechnology 27(12) 1099-1102.

[9] Shetty, R.P. and Endy, D., Knight Jr, T.F. 2008. Engineering BioBrick vectors from BioBrick parts. Journal of biological engineering 2, 5.

[10] Muyrers, J.P.P., Zhang, Y., Stewart, A.F. 2001. Techniques: recombinogenic engineering-new options for cloning and manipulating DNA. Trends in Biochemical Sciences 26(5), 325-33

[11] Larionov, V., Kouprina, N., Graves, J., Chen, X. N., Korenberg, J. R., and Resnick, M. A. 1996. Specific cloning of human DNA as yeast artificial chromosomes by transformation-associated recombination. Proc Natl Acad Sci USA *93*, 491-496.

[12] Gibson D. G., Glass, J. I., Lartigue, C., Noskov, V. N., Chuang, R. Y., Algire, M. A., Benders, G., A., Montague, M. G., Ma, L., Moodie, M. M., Merryman, C., Vashee, S., Krishnakumar, R., Assad- Garcia, N., Andrews-Pfannkoch, C., Denisova, E. A., Young, L., Qi, Z. Q., Segall-Shapiro, T. H., Calvey, C. H., Parmar, P. P., Hutchison 3rd, C A., Smith, H. O., and Venter, J. C. 2010. Creation of a bacterial cell controlled by a chemically synthesized genome. Science *329*, 52-56.

[13] Raymond, C. K., Sims, E. H., and Olson M. V. 2002. Linker-mediated recombinational subcloning of large DNA fragments using yeast. Genome Res *12*, 190-197.

[14] http://partsregistry.org/Main_Page

Network Reconstruction Using Literature Curated and High Throughput Data

[Extended Abstract]

Vishwesh V. Kulkarni
Strand Life Sciences
Kirloskar Business Park
Bangalore, India
vishwesh@ee.iitb.ac.in

Kalyanasundaram Subramanian
Strand Life Sciences
Kirloskar Business Park
Bangalore, India
kas@strandls.com

Reza Arastoo
Department of Chemical Engineering
Lehigh University
Bethlehem, PA 18015, USA
reza308@lehigh.edu

Mayuresh V. Kothare
Department of Chemical Engineering
Lehigh University
Bethlehem, PA 18015, USA
mvk2@lehigh.edu

Marc Riedel
Department of Electrical and Computer Engineering
University of Minnesota
Minneapolis, MN 55455, USA
mriedel@umn.edu

ABSTRACT

We present a convex optimization approach to reconstruct gene regulatory networks using microarray data and literature curated data. Our approach builds on the theory developed recently in [13] to obtain a model of the network in terms of linear *ordinary differential equations* (ODE's), i.e., as $\dot{x} = Ax + Bu$ where x denotes the vector of gene expression values ans u denotes exogenous perturbation. However, whereas the solution proposed in [13] needs the microarray data as the outcome of a series of controlled experiments in which the network is perturbed by over expressing one gene at a time, our approach makes use of the available steady-state microarray data, as is the case in many real-world applications. First, we show how the results of [13] can be easily extended to derive the required stable model $\dot{x} = Ax + Bu$. We then demonstrate how a class of stable *nonlinear* ODE models of the given system can be derived by making use of piecewise quadratic Lyapunov functions.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.8 [**Software**]: Software Engineering

General Terms

Theory and Simulation Platform

Keywords

Gene regulatory network, modeling, microarray data, literature curated data, ODE's

1. EXTENDED ABSTRACT

Computational models are now increasingly used to analyze and predict the behavior of biochemical reaction networks (see [8, 6]). Among these networks, *gene regulatory networks* (GRN's), i.e., networks that capture interactions between genes and other cell substances such as proteins, are particularly interesting now since, today, RNA microarray assays are providing a wealth of gene expression information at increasingly lower costs. We are interested in understanding how this raw data, together with literature curated data, can be used to develop computationally efficient and reliable models of the underlying gene regulatory networks.

This problem has received considerable attention over the last 10 years. For example, a first order predictive model of a gene and protein regulatory network has been obtained in [3] using multiple linear regressions. In [3], it is assumed that the underlying GRN can be represented well enough by a system of nonlinear differential equations. Furthermore, it is assumed that the network is operating close enough to a steady-state point. Then, it follows that the GRN can be represented as a linear ODE $\dot{x} = Ax + u$ where $x \in \mathbb{R}^p$ is the vector of gene expression values, $\dot{x}$ is the time-derivative of x, u is an exogenous perturbation, and $A \in \mathbb{R}^{p \times p}$ is the interconnection matrix, the elements of which capture the regulatory interactions between entities of the GRN. The modeling problem thus reduces to that of estimating A. Under the steady-state assumption, $\dot{x} = 0$ so that the assumed model now reduces to $Ax = -u$. Now, if we induce N independent perturbations u and measure the corresponding steady-state values x of the RNA concentrations, then multiple linear regression can be easily applied to estimate A in the presence of measurement noise (see [3]). In [3], this approach has been applied to reconstruct a 9-gene subnetwork of the SOS pathway in *Escherichia coli*. In [13], this approach has recently

been compactly recast as a constrained convex optimization solution. The objective function is taken to be a convex combination of the network sparsity and the closeness of the predicted outcome to the test data. The literature curated data is accounted for as a set of constraints on the interconnection matrix A. To ensure that the solution is a stable network, an additional stability constraint is stipulated in [13]. However, [13] needs the microarray data as the outcome of a series of $p = N$ controlled experiments in which the network is perturbed by over expressing one gene at a time whereas in many real-world applications, it is impossible to create such raw data. We show that this requirement is actually unnecessary and that the algorithms presented in [13] can be readily modified to be compatible with available uncontrolled $p < N$ steady-state microarray data samples.

We next present a convex optimization solution to construct nonlinear ODE models of GRN's. Although for computational and otherwise convenience, many of the current modeling approaches resort to linear ODE models, it is well-known that the entities in biochemical reaction networks exhibit nonlinear interactions (see, for example, [1, 2, 9, 11]). To account for these nonlinearities, Hill-cube transformations are recommended in Odefy (see [7]) to transform a discrete logic model into a nonlinear ODE model. As a practical measure, this approach could be readily adopted to obtain the nonlinear ODE model in the following two steps: (1) use CNO (see [9]) to obtain a Boolean model using steady-state microarray data, and (2) use Odefy (see [7]) to transform this Boolean model into the desired nonlinear ODE model. Unfortunately, CNO requires a *cue and response* data and, hence, is not compatible with steady-state microarray data. Another approach to obtain a nonlinear ODE model is to use the *multivariate adaptive regression splines* (MARS) as follows: (1) obtain a steady-state description $y = f(x)$, where y is the vector of target variables, x is the vector of predictor variables, and f is the nonlinear MARS function (see [4]); (2) model the system as $\dot{y} = f(x)$ near the steady-state. However, this approach is not guaranteed to yield a stable GRN as its solution. We show that if the nonlinearity can be well approximated by a piecewise affine function, then piecewise quadratic Lyapunov functions (see [5]) can be appropriately used to obtain a stable nonlinear ODE model if it exists. Our solution is based on the fact that the piecewise affine function splits the state-space into cells, in each of which the system has a linear ODE representation. Then, as opposed to stipulating that $x^T(A^TP + PA)x < 0$, as done in [13], it is sufficient to stipulate that

$$x^T(A_i^TP + PA_i)x < 0 \quad \text{for } x \in X_i,$$

where we assume A_i is the linearization of the nonlinear ODE in the cell X_i of the state-space. Using the S-procedure (see [5]), the algorithm to compute a stable nonlinear ODE model, if it exists, is then derived. Our approach to network reconstruction differs from that of [12] in that [12] needs a large number of data samples that are in either a cue-response form or in a time-series form. Our approach to network reconstruction differs from that of [10] in that [10] mandates that the data samples should be the outcomes of independent perturbations to the so-called modules of the network. We have implemented our algorithms in MATLAB to successfully reconstruct a sparse 35-node network in which the maximum number of nodes adjacent to a node is 9. The primary disadvantage of our method is the speed since the solution to the bilinear matrix inequality of the form $A^TP + PA < 0$, where A and P are both unknown *a priori*, is resource intensive on computation.

2. REFERENCES

[1] R. Bonneau, D. Reiss, P. Shannon, M. Facciotti, L. Hood, N. Baliga, and V. Thorsson. The inferelator: An algorithm for learning parsimonius regulatory networks from systems biology data sets *de novo*. *Genome Biology*, 7(5), May 2006.

[2] W. W. Chen, M. Niepel, and P. K. Sorger. Classic and contemporary approaches to modeling biochemical reactions. *Genes and Development*, 24:1861–1875, November 2010.

[3] T. S. Gardner, D. di Bernardo, D. Lorenz, and J. J. Collins. Inferring genetic networks and identifying compound mode of action via expression profiling. *Science*, 301:102–105, July 2003.

[4] T. Hastie, R. Tibshirani, and J. H. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Verlag, New York, New York, 2001.

[5] M. Johansson and A. Rantzer. Computation of piecewise quadratic Lyapunov functions for hybrid systems. *IEEE Trans. on Automatic Control*, 43(4):555–559, April 1998.

[6] S. Klamt, J. Saez-Rodriguez, J. A. Lindquist, L. Simeoni, and E. D. Gilles. A methodology for the structural and functional analysis of signaling and regulatory networks. *BMC Bioinformatics*, 56(7):1471–1497, 2006.

[7] J. Krumsiek, S. Polsterl, D. Wittmann, and F. J. Theis. Odefy — from discrete to continuous models. *BMC Bioinformatics*, 11:1861–1875, November 2010.

[8] M. K. Morris, J. Saez-Rodriguez, P. K. Sorger, and D. A. Lauffenburger. Logic-based models for the analysis of cell signaling networks. *Biochemistry*, 49:3216–3224, 2010.

[9] J. Saez-Rodriguez, L. Alexopoulos, J. Epperlain, R. Samaga, D. A. Lauffenburger, S. Klamt, and P. K. Sorger. Discrete logic modeling as a means to link protein signaling networks with functional analysis of mammalian signal transduction. *Molecular Systems Biology*, 5(331), 2009.

[10] E. Sontag. Network reconstruction based on steady-state data. *Essays Biochem.*, 45:161–176, 2008.

[11] D. Wittmann, J. Krumsiek, J. Saez-Rodriguez, D. A. Lauffenburger, S. Klamt, and F. J. Theis. Transforming Boolean models to continuous models: Methodology and applications to T-cell receptor signaling. *BMC Systems Biology*, 98(3), 2009.

[12] Y. Yuan, G. Stan, S. Warnick, and J. Goncalves. Robust dynamical network reconstruction. In *IEEE Conference on Decision and Control*, pages 180–185. Atlanta, GA, December 2010.

[13] M. Zavlanos, A. Julius, S. Boyd, and G. Pappas. Inferring stable genetic networks from steady-state data. *Automatica*, 2011 (In print).

ShReDing Biochemical Networks to Uncover Hierarchical Modularity

Gautham Sridharan[*], Kyongbum Lee[*], Soha Hassoun[Δ]

[Δ] Department of Computer Science, Tufts University
[*]Department of Chemical and Biological Engineering. Tufts University
{Gautham.Sridharan, Kyongbum.Lee, Soha.Hassoun}@tufts.edu

1. INTRODUCTION

Hierarchical modularity has emerged as an organizational principle of biochemical networks, providing insights into the coordinated regulation of reactions within and across pathways. In principle, the modularity of a biochemical network should allow the system to be partitioned into minimally interdependent parts, which in turn can facilitate detailed analysis of each part in the context of the overall system. In practice, modularity analysis has often relied on *ad hoc* decisions to find modules that are consistent with textbook biochemistry. While there is general agreement that a module should consist of a biologically meaningful group of connected components in the network, there is little consensus on the metric needed to quantitatively evaluate the quality of a partition. The goal of this study is to investigate a novel metric that can be used to systematically partition a biochemical network into functionally relevant groups of reactions. The metric, termed the **Sh**ortest **Re**troactive **D**istance (*ShReD*), characterizes the retroactive connectivity between any two reactions in a network arising from potential feedback interactions, thereby grouping together network components which mutually influence each other. It has been shown that feedback in biological systems contributes to modularity, which in turn affords robustness by limiting the propagation of perturbations (Kitano 2004; Stelling, Sauer et al. 2004). We evaluate *ShReD* as a partition metric on a model of liver metabolism augmented with drug transformation reactions and allosteric regulatory interactions.

2. METHODS

In this study, the focus is on understanding hierarchical relationships among reactions while treating metabolites as shared resources among modules. We thus abstract the network as a directed graph with nodes representing reactions, and edges indicating a coupled relationship between two reaction nodes. The interactions between nodes are derived as follows, and illustrated in Figure 1.

A) For each pair of series reactions, R_i and R_j, a directional edge from R_i to R_j is established. (Figure 1a, Figure 1b). A pair of series reactions is defined as one where the product of the first reaction is consumed by the second one.

B) For each reaction R_i regulated (through activation or inhibition) by a metabolite m_j, a directional edge from each reaction R_k producing metabolite m_j to R_i is established. (Figure 1c, Figure 1d).

If a reaction is reversible, then it is modeled as one node, and the interactions are derived as specified in (A) and (B) taking into account the reaction's bidirectionality.

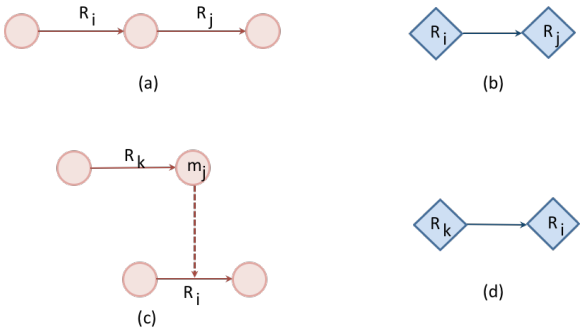


Figure 1: Derivation of reaction-centric network.

The *ShRed* of two nodes *i* and *j* is defined as the length of the shortest cycle that involves both nodes. A retroactive interaction exists between two nodes *i* and *j* if there is a path from node *i* to node *j* and a return path from node *j* to node *i*. The *ShReD* of nodes *i* and *j* is the sum of the shortest path distance from node *i* to node *j* and the shortest return path distance from node *j* to node *i*. Several well-known and efficient algorithms exist for computing the shortest path.

We adapted Newman's community detection algorithm (Newman 2006) to obtain hierarchical partitions based on the *ShReD* metric. Given a network of connected components, Newman's algorithm divides the network into two partitions so as to maximize the "modularity score," which scales with the number of connections within each partition relative to the expected number of connections between two randomly chosen components in the network.

To partition the network based on the *ShReD* metric, we modified Newman's algorithm as follows. Instead of computing the differences between the actual and expected number of connections between two nodes i and j, we computed the differences between $ShReD_{ij}$ and the *expected ShReD*, P_{ij}, between nodes i and j. The expected *ShReD*, P_{ij}, between two nodes is computed as the arithmetic mean of the average of all *ShReDs* involving node i and the average of all *ShReDs* involving node j.

We can then define a *ShReD-based modularity matrix*, $\boldsymbol{B}$, as follows:

$$B_{ij} = P_{ij} - SHRED_{ij}$$

The objective of our partition algorithm is similar to Newman's. We wish to find a vector $\boldsymbol{s}$, which assigns each node in the network to one of two subnetworks so as to maximize the modularity score, Q, which we re-define as follows:

$$Q = \sum_i \sum_j B_{ij} s_j s_i$$

where s_i is an element of a vector $\boldsymbol{s}$. Each s_i can have a value of -1 or 1. An increase in Q is obtained in two cases: if B_{ij} is positive, i.e. the *ShReD* between two nodes i and j is shorter than expected, and the two nodes are assigned to the same subnetwork ($s_i = s_j = 1$ or $s_i = s_j = -1$), or if B_{ij} is negative and the two nodes are assigned to the different subnetwork ($s_i = 1$ and $s_j = -1$ or vice versa). The vector $\boldsymbol{s}$ that maximizes Q be approximated by the leading eigenvectors of $\boldsymbol{B}$ (Pothen, Simon et al. 1990).

3. RESULTS

For the metabolic network, which included cofactors, the ShReD partition generated hierarchical modules whose compositions compared favorably with canonical associations based on textbook biochemistry (Figure 2). Interestingly, the ShReD partition revealed a 'redox' module involving reactions of glucose, pyruvate and lipid metabolism as well as drug transformation, which interact in a cyclical manner through shared production and consumption of NADPH.

4. CONCLUSION

Our novel metric ShReD, combined with Newman's algorithm, is to our knowledge the first modularity analysis technique that partitions a biochemical network to preserve cyclical interactions between reactions.

5. REFERENCES

Kitano, H. (2004). "Biological robustness." Nature reviews. Genetics **5**(11): 826-837.

Newman, M. E. (2006). "Modularity and community structure in networks." Proc Natl Acad Sci U S A **103**(23): 8577-8582.

Pothen, A., H. Simon, et al. (1990). "Partitioning Sparse Matrices with Eigenvectors of Graphs." SIAM J. Matrix Anal. Appl. **11**: 430-452.

Stelling, J. o., U. Sauer, et al. (2004). "Robustness of cellular functions." Cell **118**(6): 675-685.

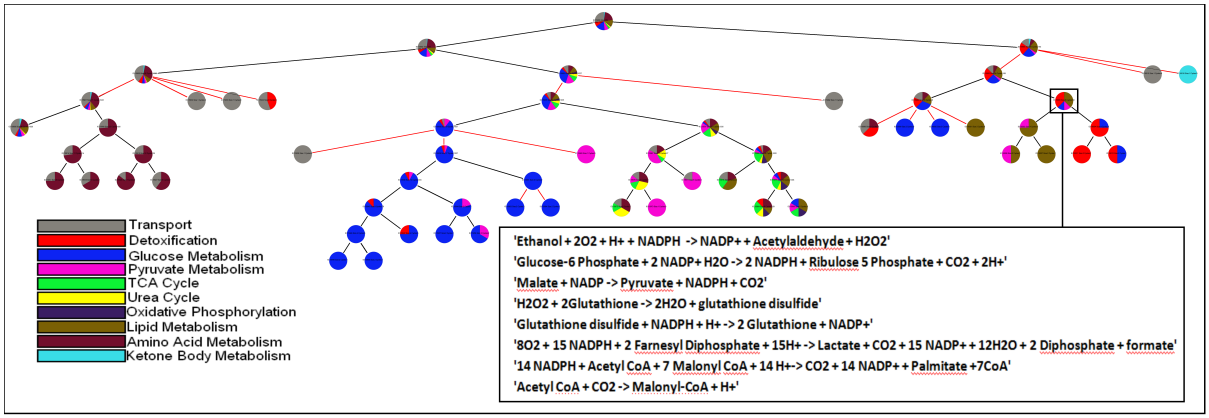


Figure 2. ShReD-based hierarchical partition of a hepatocyte metabolic network represented using a pie-in-tree chart. The legend in the lower left corner provides a color map representing the classification of reactions based on canonical associations (i.e. in the literature). Each pie represents a module, and the color of the pie represents proportional participation of reactions from various canonical associations. A single-colored pie indicates that all reactions in the module belong to a single canonical association. A multi-color indicates that *ShReD*-based partitioning found an alternative grouping.

Pareto Optimal Fronts in Bacterial Knockout Strategies

Jole Costanza and Giuseppe Nicosia
Dept. of Mathematics and Computer Science
University of Catania
Catania, Italy
costanza@dmi.unict.it
nicosia@dmi.unict.it

Luca Zammataro
Dept. of Translational Medicine
University of Milan
Milan, Italy
luca.zammataro@humanitasresearch.it

Pietro Liò
Computer Laboratory
University of Cambridge
Cambridge, UK
pl219@cam.ac.uk

ABSTRACT

In this work, we present a novel multi-objective optimization algorithm that computes Pareto optimal tradeoff surfaces for identifying genetic manipulations leading to targeted over- and under- productions. These surfaces provide key information of the phase space of the outcome of best genetic design strategies, i.e., the result of genetic knockouts. Our algorithm performs the simultaneous optimization of multiple cellular functions (i.e., multiple objectives), while minimizing the *knockout cost*; it also selects those genetic designs with greater in silico production of desidered metabolites. Knockouts are modeled in terms of gene sets that can affect one or more reduced reactions using gene-protein reaction mapping. We challenge the algorithm on several data test; here we show results on the latest genome-scale model of *Escherichia Coli* K-12 MG1655 [2] iAF1260.

Categories and Subject Descriptors

J.3 [**Life and Medical Sciences**]: [Biology and genetics]; G. 1.6 [**Optimization**]: [Global Optimization]

General Terms

Algorithms, Design

Keywords

Metabolic CAD, Metabolic engineering, Multi-objective Optimization, Biological Circuit Design, Escherichia Coli, Genetic Design Strategies

1. METHODS AND DATA

The genome-scale FBA model of E. coli, iAF1260 [2] consists of three parts. From m metabolites and n reactions, we form an $m \times n$ stoichiometric matrix S, whose ijth element S_{ij} is the stoichiometric coefficient of metabolite i in reaction j. The vector of flux values v, whose jth element v_j is the flux though reaction j, are constrained by a lower bound vector a and an upper bound vector b. Finally, the linear objective is formed by multiplying the fluxes by an objective vector f, whose jth element f_j is the weight of reaction j in the biological objective (Biomass). To allow the algorithms to function at the genetic level, we used gene-protein-reaction (GPR) mappings. GPR mappings define how certain genetic manipulations affect reactions in the metabolic network. For a set of L genetic manipulations, the GPR mappings is summarized with an $L \times n$ matrix G, where the ljth element G_{lj} of G is 1 if the lth genetic manipulation maps onto reaction j and is 0 otherwise [3]. To knockout the genes we declared the knockout vector y, whose lth element y_l is equal to 1 if the gene involved in manipulation l is knocked out and 0 otherwise.

IWBDA'11, June 6-7, 2011, San Diego, CA, USA

2. RESULTS

As matter of comparison we report the following: GDLS algorithm [3], OptFlux algorithm [5] [4] and OptKnock algorithms. [1]. GDLS algorithm [3] performs a single-objective optimization; it optimizes the synthetic objective function Acetate, obtaining 15.914 $\text{mmolh}^{-1}\text{gDW}^{-1}$ with knockout cost (kcost) equal to 15. For the second synthetic objective function, Succinate, GDLS obtains 9.727 $\text{mmolh}^{-1}\text{gDW}^{-1}$ with kcost=26. The biomass is constant: 0.050 h^{-1}. OptFlux algorithm [5][4] uses two meta-heuristics which obtain the following results: OptFlux with Evolutionary Algorithm reaches Acetate = 15.138 $\text{mmolh}^{-1}\text{gDW}^{-1}$ and Succinate = 9.874 $\text{mmolh}^{-1}\text{gDW}^{-1}$, while OptFlux with Simulated Annealing obtains Acetate = 15.219 $\text{mmolh}^{-1}\text{gDW}^{-1}$ and Succinate = 10.007 $\text{mmolh}^{-1}\text{gDW}^{-1}$. The designed algorithm performs a multi-objective optimization obtaining the following results: Acetate 21.901 $\text{mmolh}^{-1}\text{gDW}^{-1}$, Succinate 12.720 $\text{mmolh}^{-1}\text{gDW}^{-1}$ and Biomass 0.050 h^{-1} turning off a single gene b0918 (*kdsB*, reaction KDOCT2, reaction catalyzed by Lipopolysaccharide Biosynthesis), that is, kcost=1 (Figure 1 shows the results in anaerobic and aerobic conditions). The algorithm discovers hundreds of non-dominated solutions with above Acetate and Succinate values but different kcosts; with kcost=1 the algorithm discovered two distinct genetic design strategies (gene b0918, and gene b3867, *hemN*, reaction CPPPGO2, Oxygen Independent coproporphyrinogen-III oxidase), with kcost=2 there are 8 genetic design strategies, with kcost=3 it is possible to use 11 distinct genetic design strategies and so on. The study of genes and reactions of E. Coli has involved inferring 16 Pareto tradeoffs in *anaerobic conditions*: succinate vs. acetate; succinate

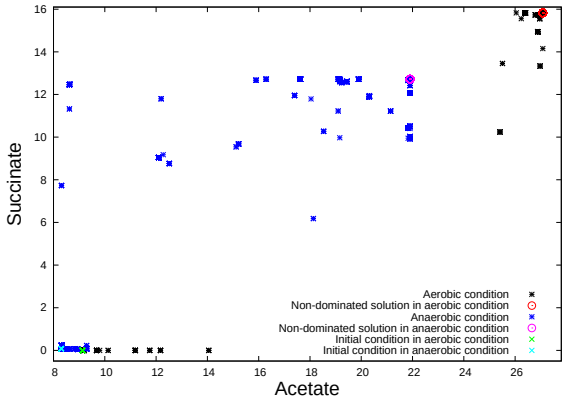


Figure 1: Feasible solutions and the non-dominated solution of Succinate vs. Acetate under aerobic (10 mmol/h oxygen) and anaerobic condition on a basis of 10 mmol/h glucose fed, GLF.

vs. biomass; acetate vs. biomass; ethanol vs. biomass; glycerol vs. biomass; 1,2-propanediol vs. biomass; lactate vs. biomass; formate vs. biomass; CO_2 vs. biomass; acetate vs. ethanol; acetate vs. succinate vs. biomass; glycerol vs. succinate vs. biomass; glycerol vs. succinate vs. acetate; glycerol vs. succinate vs. ethanol; glycerol vs. succinate vs. lactate; glycerol vs. succinate vs. formate. The main purpose is to find the genetic design that maximizes the desired fluxes (more than one) and biomass, with a lower kcost. We also performed a *four-objective optimization*: to maximize acetate, succinate, biomass and simultaneously to minimize kcost (Figure 3). *Anaerobic conditions* were simulated as zero oxygen and 10 mmolh^{-1} available glucose; aerobic condition as 5 and 10 mmolh^{-1} available oxygen and glucose. We computed: Succinate vs. Biomass and Acetate vs. Biomass trade-off (Figure 2).

In aerobic conditions (10 mmol/h oxygen), we compared the designed algorithm against GDLS. For Acetate, GDLS found 23.1522 mmolh^{-1}gDW^{-1} with kcost=4. while for Succinate 9.2704 mmolh^{-1}gDW^{-1} with kcost=13. The designed algorithm obtains the following results Acetate=27.0799 mmolh^{-1}gDW^{-1}, Succinate=15.8250 mmolh^{-1}gDW^{-1}, with kcost=6 (Figures 1 and 2). We found four combinations of knockout. As matter of example, we report the following combination (gene set and corresponding reactions): b2889, isopentenyl-diphosphate D-isomerase; b3198, 3-deoxy-manno-octulosonate-8-phosphatase; ((b0849 and b2762) or(b1064 and b2762) or (b1654 and b2762) or (b2762 and b3610)), phosphoadenylyl-sulfate reductase (glutaredoxin).

3. CONCLUSIONS

One of the goals of the present research work is to use the Pareto optimal solutions of E. Coli in order to produce useful metabolites and effective drugs. The algorithm scales effectively as the size of the metabolic system and the number of genetic manipulations increase. We clearly outperform the GDLS heuristic [3], OptFlux, [5] [4], OptKnock [1] and other heuristics, search methods, global and local optimization algorithms. Moreover, the results obtained show that the multi-objective approach is very suitable for the genetic design strategies (GDS) discovering. To our knowl-

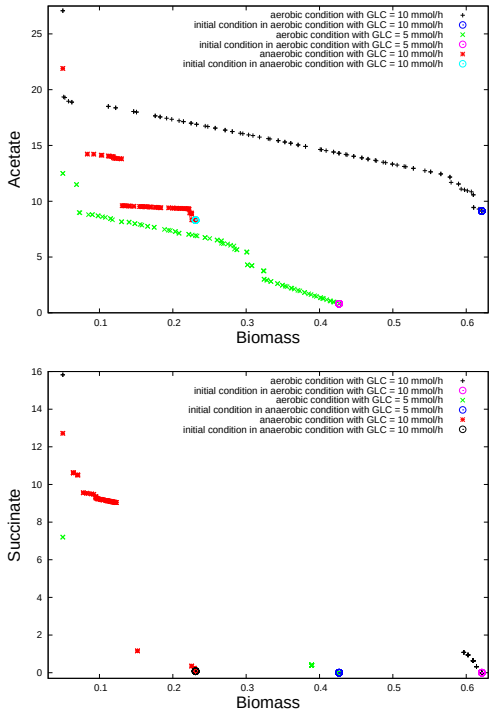


Figure 2: Biomass vs. Succinate and Biomass vs. Succinate aerobic (10 mmol/h oxygen) and anaerobic conditions with 10 mmol/h and 5 mmol/h GLF.

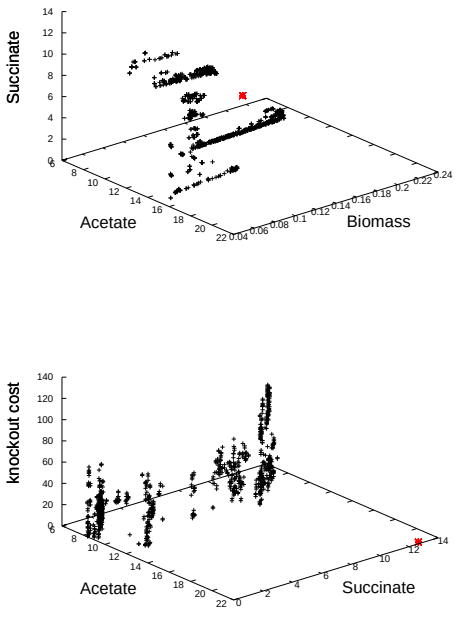


Figure 3: Biomass vs. Acetate vs. Succinate and Acetate vs. Succinate vs. kcost Pareto Fronts under anaerobic condition on a basis of 10 mmol/h GLF. In red the best solutions.

edge, this is the first study on multi-objective optimization for the GDS problem and in the characterizing of biological pathway in terms of Pareto optimal fronts.

4. REFERENCES

[1] A. P. Burgard, P. Pharkya, and C. D. Maranas. Optknock: a bilevel programming framework for identifying gene knockout strategies for microbial strain optimization. *Biotechnol. Bioeng.*, 84(6), December 2003.

[2] A. M. Feist, C. S. Henry, J. L. Reed, M. Krummenacker, A. R. Joyce, P. D. Karp, L. J. Broadbelt, V. Hatzimanikatis, and B. Ø. Palsson. A genome-scale metabolic reconstruction for escherichia coli k-12 mg1655 that accounts for 1260 orfs and thermodynamic information. *Mol. Syst. Biol.*, 3(121), June 2007.

[3] D. S. Lun, G. Rockwel, N. J. Guido, M. Baym, J. A. Kelner, B. Berger, J. E. Galagan, and G. M. Church. Large-scale identification of genetic design strategies using local search. *Mol. Syst. Biol.*, 5(296), August 2009.

[4] K. Patil, I. Rocha, J. Forster, and J. Nielsen. Evolutionary programming as a platform for in silico metabolic engineering. *BMC Bioinformatics.*, 6(308), December 2005.

[5] M. Rocha, P. Maia, R. Mendes, J. P. Pinto, E. C. Ferreira, J. Nielsen, K. R. Patil, and I. Rocha. Natural computation meta-heuristics for the in silico optimization of microbial strains. *BMC Bioinformatics.*, 9(499), November 2008.

Optimal Design of the Mitochondrial Bioenergetics

Jole Costanza and Giuseppe Nicosia
Dept. of Mathematics and Computer Science
University of Catania
Catania, Italy
costanza@dmi.unict.it
nicosia@dmi.unict.it

Luca Zammataro
Dept. of Translational Medicine
University of Milan
Milan, Italy
luca.zammataro@humanitasresearch.it

Pietro Liò
Computer Laboratory
University of Cambridge
Cambridge, UK
pl219@cam.ac.uk

ABSTRACT

Mitochondrial oxidative phosphorylation is the major ATP synthetic pathway in eukaryotes. In this process, electrons liberated from reducing substrates (NADH and $FADH_2$) are delivered to O_2 via a chain of respiratory H^+ pumps. These pumps establish a H^+ gradient across the inner mitochondrial membrane, and the electrochemical energy of this gradient is then used to drive ATP synthesis. In addition to ATP synthesis, mitochondria are the site of other important metabolic reactions. Mitochondria also play central roles in cellular Ca^{2+} homeostasis which affects numerous other cell signaling pathways [2]. In this work, we used a multi-objective optimization algorithm that computes Pareto Optimal Tradeoff for maximizing the mitochondrial bioenergetic in different matrix Ca^{2+} concentration. It appears that Ca^{2+} is a global positive effector of mitochondrial function, and thus any perturbation in mitochondrial or cytosolic Ca^{2+} homeostasis will have profound implications for cell function, for example, at the level of ATP synthesis and NADH generation [2].

Categories and Subject Descriptors

J.3 [**Life and Medical Sciences**]: [Biology and genetics]; G. 1.6 [**Optimization**]: [Global Optimization]

General Terms

Algorithms, Design

Keywords

Metabolic CAD, Metabolic engineering, Multi-objective Optimization, Biological Circuit Design

1. METHODS AND DATA

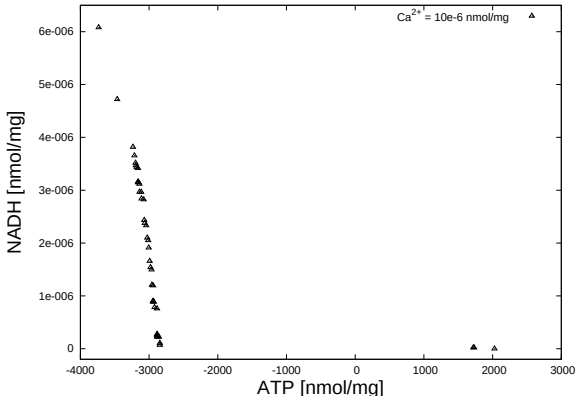
The mitochondrial model is a 73 state system of DAEs, each of which represents the metabolites involved in bioenergetic reactions of mitochondria. The state variables was initialized to achieve the fully oxidized state [1]. In this work, we calculated the metabolites concentration that leads to maximize the matrix ATP and NADH, maintaining constant oxidized cytochrome c, (cytc $^{3+}$), reduced cytochrome c, (cytc^{2+}), ubiquinone, ubiquinol, NAD_{mtx}, $NADH_{mtx}$, GTP_{mtx}, GDP_{mtx} (mtx = matrix), the mitochondrial membrane potential (1 mV), the matrix O_2 (0.0652 nmol/mg), the total CO_2 (21.4762 nmol/mg) and Ca^{2+} concentration. We initialized Ca^{2+} with 5 different value to evaluate the behavior of mitochondria. First we used Ca^{2+}=10e-6, then 10e-5, 10e-7, 10e-6×1.5 and 10e-6/1.5 nmol/mg.

IWBDA'11, June 6-7, 2011, San Diego, CA, USA

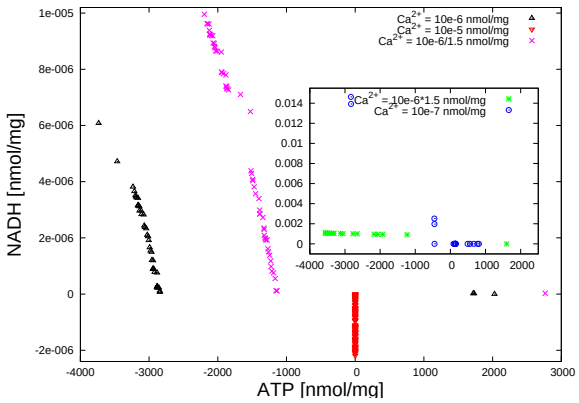
2. RESULTS

In initial condition, we obtained NADH = 1.5987e-010 nmol/mg (formation) and ATP = -0.0014 nmol/mg (consumption). After optimization with Ca^{2+}=10e-6 nmol/mg, we obtained the Pareto Front shown in Figure1 (a) and we analyzed three non-dominated solutions: the point with maximum ATP synthesis (and lower NADH formation), the point with maximum NADH formation (and lower ATP synthesis), and, finally, the tradeoff point. The first solution provides ATP = 2027.34 nmol/mg and NADH = 6.17e-015 nmol/mg, with over-production of SUC_{mtx}, PYR_{mtx}, $CoASH_{mtx}$, H^+_{mtx}, ATP_{ims} ADP_{ims} (ims = intermembrane space) and Mg^{2+}_{cyt} (cyt= cytosolic space), and under-production of $ISOC_{mtx}$, aKG_{mtx}, MAL_{mtx}, $AcCoA_{mtx}$, CIT_{ims}, $ISOC_{ims}$, aKG_{ims}, SUC_{ims}, MAL_{ims} and GLU_{cyt}, ASP_{cyt}, see Figure 2. The second solution provides ATP = -3734.6 nmol/mg (consumption) and NADH = 6.07e-006 nmol/mg, over-producing the following metabolites: H^+_{mtx}, $ISOC_{mtx}$, SUC_{mtx}, FUM_{mtx}, $CoASH_{mtx}$, ATP_{ims}, MAL_{ims} whereas CIT_{mtx}, $SCoA_{mtx}$, MAL_{ims}, ADP_{ims}, AMP_{ims}, Pi_{ims}, PYR_{ims}, $GLU_{ims,cyt}$ and $aKG_{ims,cyt}$ are totally consumed, as shown in Figure 2. The tradeoff point is equal to the first analyzed point. Increasing the matrix calcium concentration from 10e-6 to 10e-5 nmol/mg, ATP synthesis and NADH formation are stopped and both molecules are consumed by mitochodrial metabolism, Figure 1 (b). This achievement demonstrates that a perturbation in mitochondrial Ca^{2+} homeostasis will have profound implications for cell function at the level of ATP synthesis and NADH generation. If calcium increases by a little step, 10e-6×1.5 nmol/mg we obtain an increase in NADH formation, while ATP is invariant. In case of drastically matrix calcium depletion, 10e-7 nmol/mg, there is a lower ATP synthesis, Figure 1 (b), but with Ca^{2+}= 10e-6/1.5 nmol/mg both objectives are maxi-

mized.



(a)



(b)

Figure 1: (a) Mitochondrial NADH vs. ATP Pareto Front with Ca^{2+}=10e-6 nmol/mg. (b) Mitochondrial NADH vs. ATP Pareto Fronts with different Ca^{2+} concentration. Internal plot in B present different y scale.

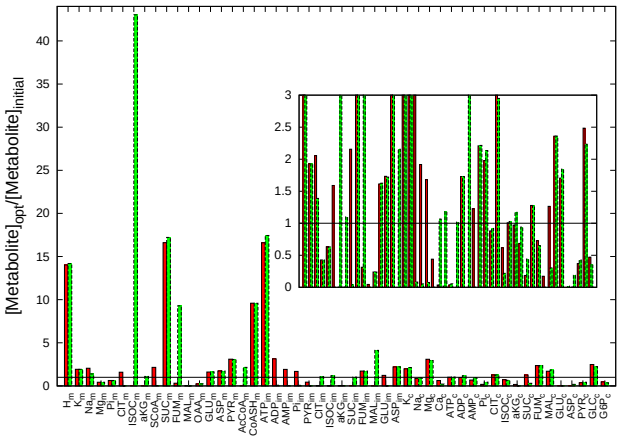


Figure 2: The ratio of the metabolite concentrations optimized by the multi-objective algorithm compared to the initial concentrations for maximum ATP synthesis, in red, and maximum NADH formation, in green. Small plot has focused y-axis in [0,3] range.

3. CONCLUSIONS

Ca^{2+} in a ubiquitous messenger and participates in many signaling pathways that are crucial for cell survival. Data from literature demonstrate that mitochondria are implied to play a crucial role in neuronal cell survival since they are regulators of ATP metabolism, Ca^{2+} homeostasis, NAD+, NADH, and of endogenous reactive oxygen species production. All these metabolites are key protagonists in cellular mechanisms leading to degenerative process and, consequently, the programmed cell death. Moreover there are evidences that mitochondrial dysfunction occurs prior to the onset of symptoms of several neurodegenerative diseases, such as Parkinson's disease (PD) [3], Alzehimer disease (AD) [5] and Amyotrophic Lateral Sclerosis (ALS) [4]. PD and AD are linked by the same mitochondrial Ca^{2+} homeostasis dysfunction hypothesis, leading to ATP depletion and NAD+/NADH ratio imbalance, and promoting cellular microtubule disruption. Moreover, mitochondrial membrane potential is lost and calcium homeostasis is deregulated, this event leads to calpains activation, high levels of mitochondrial ROS are oxidative stress. Finally, oxidative stress culminate in neurodegeneration [5] [3]. Disrupted calcium homeostasis and subsequent excitotoxicity are to be interrelated mechanisms that collectively contribute to motor neuron degeneration in ASL [4].

By means of the computational quantitative framework here proposed, we are able to mathematically infer dynamics of mitochondrial bioenergetic metabolism, changing the specific behavior pathways; in particular, we have to maximize the ATP, and NADH uptake rates of the mitochondria metabolism examining the role of Ca^{2+}. The results demonstrate that at different levels of mitochondrial Ca^{2+} the organelle changes its bioenergetic activity, and this is bounded by specifics lower and upper matrix Ca^{2+}, beyond which NADH and ATP production are knocked out. In particular, it's shown that high calcium levels lead to dysfunction of ATP synthesis and NADH generation.

4. SELECTED REFERENCES

[1] J. N. Bazil, G. T. Buzzard, and A. E. Rundell. Modeling mitochondrial bioenergetics with integrated volume dynamics. *PLoS Comput. Biol.*, 6(1), January 2010.

[2] P. S. Brookers, M. W. A. Y. Yoon, J. L. Robotham, and S. Sheu. Calcium, atp, and ros: a mitochondrial love-hate triangle. *Am J Physiol Cell Physiol*, 287(4), October 2004.

[3] A. R. Esteves, D. M. Arduíno, D. F. F. Silva, C. R. Oliveira, and S. M. Cardoso. Mitochondrial dysfunction: The road to alpha-synuclein oligomerization in pd. *Parkinsons Dis.*, 2011, January 2011.

[4] P. Shi, J. Gal, D. Kwinter, X. Liu, and H. Zhu. Mitochondrial dysfunction in amyotrophic later sclerosis. *Biochim Biophys Acta*, 1802(1), January 2010.

[5] C. Supnet and I. Bezprozvanny. The dysregulation of intracellular calcium in alzheimer disease. *Cell Calcium*, 47(2), February 2010.

Stochastic Resonance Optimization of a Genetic Quorum-Mediated Trigger Circuit

[IWBDA Abstract] *

Eduardo Monzon, Abiezer Tejeda, Chris Winstead and Charles Miller †
Utah State University
Dept. of Electrical and Computer Engineering
† Dept. of Biological Engineering
Logan, UT
eduardo.monzon@usu.edu

ABSTRACT

A method is presented for characterizing quorum-sensing circuits using a measure of stochastic resonance. The method is applied to a quorum-mediated trigger circuit, which is designed to synchronize a population of cells in response to a supplied exogenous signal. The stochastic resonance analysis identifies an optimal basal production rate for reliable operation of the circuit. The production rate can be adjusted by a number of methods including sequence modifications in the RBS. By combining these methods, a design optimization procedure may be obtained for synthetic quorum-type circuits.

General Terms

design, theory, reliability

Keywords

stochastic resonance, quorum sensing, genetic circuits

1. INTRODUCTION

Genetic systems are subject to noise within the environment of the cell, and some circuits rely on stochastic behavior for proper function. Growth temperature and available nutrients are factors that can affect the basal production rate of a promoter, which may appear as noise in the output signals of a logic genetic circuit. In some cases, certain levels of noise may improve the operation of a genetic circuit. In this situation, the circuit may be said to exhibit stochastic resonance.

Stochastic resonance often occurs in bistable systems or circuits that have a sensory threshold [2]. Genetic circuit constructs are nonlinear systems and are frequently designed to operate based on thresholds for some concentration of a substance. For example, it is currently popular to design genetic circuits based on Boolean logic, which implies the existence of thresholds that distinguish the logical states within the circuit. If a signal's molecular population is greater than a specified threshold, then its logical state is "1". Its logical state is assumed to be "0" otherwise.

There are currently several alternative definitions for stochastic resonance. In this work, we consider a definition based on Shannon's mutual information. Under this approach, we select two signals to represent the input and output of the circuit. Let X be the *logical* state of the input signal (i.e. $X \in \{0, 1\}$), and let Y be the logical state of the output signal. Then the mutual information $I(X; Y)$ is evaluated as a function of some parameter $\eta > 0$ that indicates the amount of noise present in the system. The circuit is said to exhibit stochastic resonance at the value of η for which $I(X; Y)$ is maximized.

In this work, we consider the input X as an externally supplied transcription factor, and the output Y is the quorum signal, HSL, exchanged between cells. The parameter η is the basal production rate of HSL, i.e. the rate at which molecules of HSL are randomly produced when the external signal is absent. The mutual information is found to have a clear maximum at non-zero value of η. This information can be used to optimize the circuit's robustness by altering the basal production rate of HSL.

2. STOCHASTIC RESONANCE IN QUORUM TRIGGER

In this work, we characterize stochastic resonance in the quorum trigger circuit shown in Fig. 1. The quorum trigger design is based on a genetic Muller C-element circuit described by Nguyen et al [4]. This device is called a "quorum trigger" because it can be used to activate or "trigger" the coordinated behaviour of a colony of cells based on an applied environmental signal. The environmental signal (called *Env* in Fig. 1) could be any transcription factor able to activate the production of LuxR via the leftmost promoter. The LuxR molecules combine with HSL molecules to activate a feedback loop on the second promoter, which induces production of both LuxR and HSL. HSL acts as a cell-to-cell signal, hence contributing to the quorum activation.

*The authors would like this abstract to be considered for both a poster and oral presentation.

When the *Env* signal is absent, molecules of HSL and LuxR are randomly produced at a low rate, referred to as the basal production rate of the rightmost promoter. This is a source of noise in the system, which could give rise to false activation. If the basal production rate is zero, then the circuit is never able to be triggered due to a total absence of HSL. There must consequently be an optimal basal production rate at which the circuit achieves its most reliable function.

In order to predict the optimal basal production rate, the circuit's reliability was characterized using the mutual information measure, defined as

$$I(X;Y) = \sum_y \sum_x p(x,y) \log\left(\frac{p(x,y)}{p_x(x)\,p_y(y)}\right).$$

In practice, the mutual information is computed numerically using Monte Carlo simulations. For this work, the Monte Carlo runs were obtained from the Gillespie Stochastic Simulation Algorithm using the iBioSim tool [3].

Figure 2 shows the plot of the mutual information between the *Env* signal and the output signal for basal production rates varying from 1×10^{-4} up to 4×10^{-2}. The information transfer is maximum around 4×10^{-3} achieving a value close to 1.

3. DESIGN OPTIMIZATION

The stochastic resonance analysis indicates a procedure for optimizing the quorum trigger circuit at design time. Based on the results from Sec. 2, an optimal basal production rate is identified. The basal production rate can be adjusted by manipulating base pairs in and around the RBS. Slight changes in the RBS can alter the expression levels across a wide range [1]. When the basal rate is altered, the active production rate is also altered, which may affect the mutual information curve and shift the optimal rate to a different value.

Because of this complex interdependence of variables, an iterative procedure may be used: (1) Evaluate the design's basal production rate for HSL, and evaluate the mutual information curve. (2) Identify the optimal rate and adjust the basal rate toward that optimal point. (3) Repeat steps 1 and 2 until improvements are no longer obtained.

While this procedure provides a plausible method for optimizing the robustness of the quorum trigger circuit, it is not yet ideal. At present, basal production rates are difficult to predict at design time, and must be determined through experimental measurement. Each iteration of the design optimization must therefore be executed experimentally. It is also difficult to predict the precise amount by which the basal rate is adjusted when the RBS is modified, which may interfere with convergence of the iterative procedure.

4. CONCLUSIONS

We presented a formal method to characterize the stochastic resonance of a genetic circuit used for quorum sensing. This method could be integrated in the design automation of genetic circuits to produce more robust devices. However, for this method to be useful a prediction or actual measure of the basal production rate of the promoter is needed as well as its range of variations.

5. ACKNOWLEDGMENTS

This work was supported by the National Science Foundation under award CCF-0916105. The authors would like to thank Prof. Chris Myers and Mr. Curtis Madsen, of the University of Utah, for their helpful discussions and collaborative activities related to the quorum trigger circuit.

6. REFERENCES

[1] C. Anderson. http://partsregistry.org/ribosome_binding_sites/prokaryotic/constitutive/anderson, 2010.

[2] M. D. McDonnell, N. G. Stocks, C. E. M. Pearce, and D. A. (Author). *Stochastic Resonance: From Suprathreshold Stochastic Resonance to Stochastic Signal Quantization*. Cambridge University Press, 2008.

[3] C. J. Myers, N. Barker, K. Jones, H. Kuwahara, C. Madsen, and N.-P. D. Nguyen. ibiosim: a tool for the analysis and design of genetic circuits. *Bioinformatics*, 25(21):2848–2849, 2009.

[4] N. phuong Nguyen, C. Myers, H. Kuwahara, C. Winstead, and J. Keener. Design and analysis of a robust genetic muller c-element. *J Theor Biol*, 264(2):174–187, May 2010.

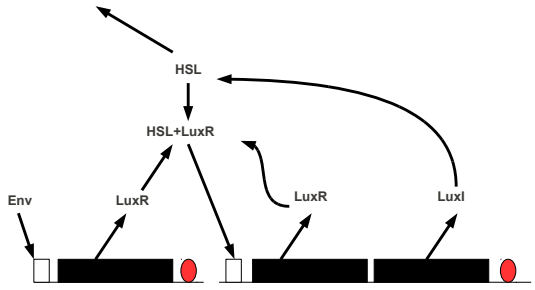


Figure 1: Quorum trigger genetic circuit design.

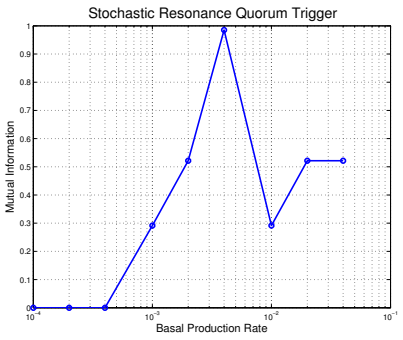


Figure 2: Stochastic resonance observed in the quorum trigger circuit.

Applying Hardware Description Languages to Genetic Circuit Design

Roza Ghamari[1], Brynne Stanton[2], Traci Haddock[1], Swapnil Bhatia[1],
Kevin Clancy[3], Todd Peterson[3], Christopher A. Voigt[2], Douglas Densmore[1],
[1]Department of Electrical and Computer Engineering, Boston University, Boston, MA
[2]Department of Pharmaceutical Chemistry, University of California, San Francisco, CA
[3]Life Technologies, Carlsbad, CA
{rozagh, thaddock, swapnilb, dougd}@bu.edu,
bcstanton@gmail.com, cavoigt@picasso.ucsf.edu,
{kevin.clancy, todd.peterson}@lifetech.com

1. INTRODUCTION

In electronic circuit design, a specific class of circuits are classified as "digital logic." These circuits abstract voltages as either "1" (present) or "0" (absent). These 0-1 circuits may be viewed as a network of switches. The behavior of these networks can be expressed using Boolean Algebraic equations. This formalism provides a number of advantages including proving circuit equivalence, reducing the size/complexity of the circuit, and translating higher level behaviors into a set of well defined boolean primitives. Digital logic can be further refined into combinational and sequential logic. Combinational logic differs from sequential logic in that the output produced by the circuit is solely a function of its current inputs (without regard to input history). Combinational logic forms the basis for much of digital logic and is the building block for sequential logic. Its automated creation, optimization, and verification are all mature areas of research.

Synthetic biology has recently gained attention with its focus on engineering biology [7]. In particular, many researchers have investigated how to assemble genetic regulatory networks. These networks can be viewed as a collection of elements which aid in DNA transcription (promoters) and elements which produce proteins (genes). Promoters can be arranged in such a way that small molecules or transcription factors can act upon them to turn specific genes on and off. If one views transcriptional signals as either present (1) or absent (0) then analogies can been made between certain genetic regulatory networks and combinational logic circuits [5], [8]. Figure 1 illustrates a NOR gate [11], [4] in a variety of representations (electrical and genetic).

However, discrepancies exist in this broad analogy between electrical digital logic systems and biological systems. These include the need for orthogonal transcription factors (electronic signals are physically separated), the selection of compatible promoter/transcription factor combinations (electronic signals 1 and 0 are universal), and the tuning

IWBDA June 6-7, 2011, San Diego, CA

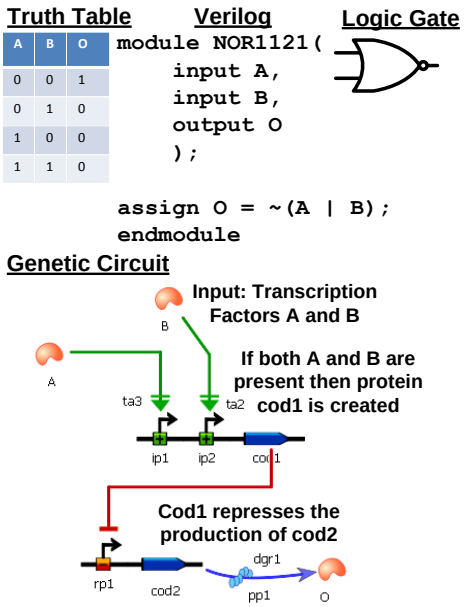


Figure 1: Various representations of a NOR Gate

of other DNA elements to produce protein concentrations needed for correct circuit functionality. However, these issues notwithstanding, if tools are built which allow synthetic biologists to use the combinational logic paradigm, then the wealth of knowledge on how to transform, optimize, and manipulate behavioral requirements into this paradigm could be leveraged for biological systems.

This work outlines our approach to taking a hardware description language (HDL) called Verilog and compiling this using a library of genetic circuit primitives. Using Verilog has several potential advantages:

- It provides a very mature, expressive language with a large user base, tremendous library support, and synthesis tools from higher level languages.
- Algorithms and techniques for logic synthesis of electronic circuits can now be re-cast more easily to apply to genetic circuits.
- It connects electrical engineers to the field of synthetic biology providing increased collaborative opportuni-

ties.

What follows is our proposed design flow from Verilog to genetic circuits. We have implemented this as a Clotho (`clothocad.org`) [6], [12] "App" named Cello (a play on the term Cell Logic). Cello provides both the graphical user interface for the design as well as a method to specify design constraints and execute the "genetic compiler".

2. DESIGN FLOW

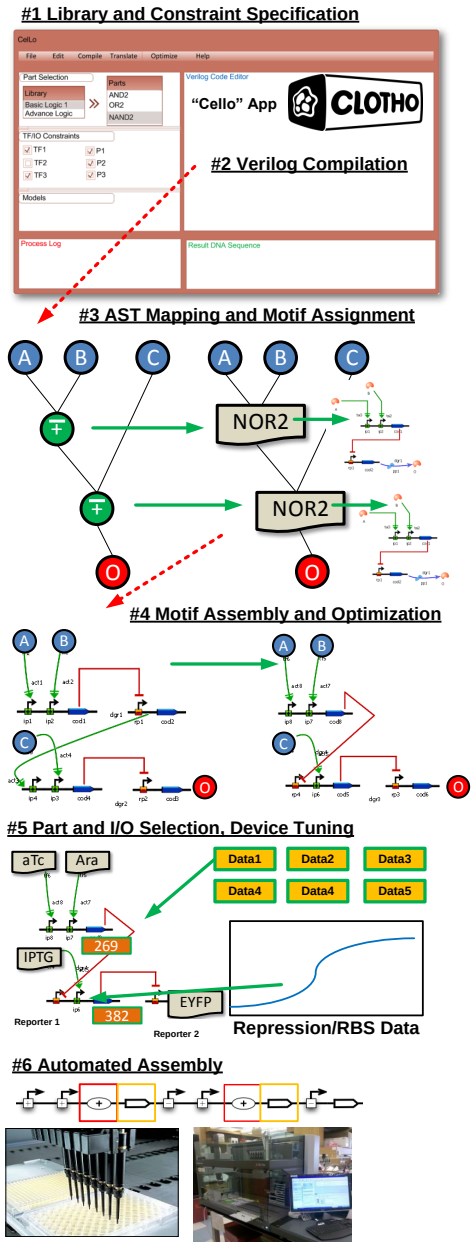


Figure 2: Proposed HDL to Genetic Circuit Design Flow

We propose the following design flow (Figure 2):

1. Select a library of biological parts from available databases. Constraints specify which parts and transcription factors can be used as well as which should be avoided. Parts are organized in database "Families" according to their function and potential composition.

2. Compile the Verilog program into an abstract syntax tree (AST). This data structure captures the desired logical function as well as the input and output relations.

3. Assign nodes in the AST to biological functional operators and map those operators to genetic motifs. This graph "coloring" can use a variety of optimizations involving desired network size as well as known design complexity limits.

4. Stitch the collection of motifs together via input and output signals. Reduce the assembled motifs to an equivalent motif which uses fewer biological resources.

5. Select physical parts to implement the motifs based on characterization data. Tune ribosome binding sites based on the level of gene expression desired. This will ensure appropriate transcriptional and translational performance requirements.

6. Send selected parts to liquid handling robots for automated assembly. The Cello App has full use of Clotho's suite of automated assembly algorithms.

3. RELATED AND FUTURE WORK

For designing genetic circuits, other language and design environments such as Eugene [2], ProMoTo [10] and Proto [1] have been proposed. None of these are explicitly bridging the gap from electronic circuit design to genetic circuit design. Lower level two-level logic descriptions like Espresso [9] and ABC [3] do not have the rich set of tools provided by more mature HDLs.

Future work involves the experimental verification of designs created with Cello, the inclusion of more behavioral Verilog constructs, and the improvement of the design optimization and tuning steps.

4. REFERENCES

[1] Beal, J., and Bachrach, J. Cells are plausible targets for high-level spatial languages. In *Spatial Computing Workshop* (2008).

[2] Bilitchenko, L., Liu, A., Cheung, S., Weeding, E., Xia, B., Leguia, M., Anderson, J. C., and Densmore, D. Eugene - a domain speci↓c language for specifying and constraining synthetic biological parts, devices, and systems. In *PLoS One* (2011).

[3] Brayton, R. K., and Mishchenko, A. Abc: An academic industrial-strength veri↓cation tool. In *CAV* (2010), pp. 24–40.

[4] Chandran, D., Bergmann, F., and Sauro, H. Tinkercell: modular cad tool for synthetic biology. *Journal of Biological Engineering 3*, 1 (2009), 19.

[5] Densmore, D., and Anderson, J. Combinational logic design in synthetic biology. In *Circuits and Systems, 2009. ISCAS 2009. IEEE International Symposium on* (May 2009), pp. 301–304.

[6] Densmore, D., Van Devender, A., Johnson, M., and Sritanyaratana, N. A platform-based design environment for synthetic biological systems. In *TAPIA '09: The Fifth Richard Tapia Celebration of Diversity in Computing Conference* (New York, NY, USA, 2009), ACM, pp. 24–29.

[7] Endy, D. Foundations for engineering biology. *Nature 438*, 7067 (Nov. 2005), 449–453.

[8] Marchisio, M. A., and Stelling, J. Automatic design of digital synthetic gene circuits. *PLoS Comput Biol 7*, 2 (Feb. 2011).

[9] Mcgeer, P., Sanghavi, J., Brayton, R., and Sangiovanni-Vincentelli, A. L. Espresso-signature: a new exact minimizer for logic functions. In *Design Automation Conference* (1993), vol. 1, pp. 618–624.

[10] Mirschel, S., Steinmetz, K., Rempel, M., Ginkel, M., and Gilles, E. D. PROMOT: modular modeling for systems biology. *Bioinformatics 25* (Mar 2009), 687–689.

[11] Tamsir, A., Tabor, J. J., and Voigt, C. A. Robust multicellular computing using genetically encoded NOR gates and chemical "wires". *Nature 469*, 7329 (Jan. 2011), 212–215.

[12] Xia, B., Bhatia, S., Bubenheim, B., Dadgar, M., Densmore, D., and Anderson, J. Clotho: A software platform for the creation of synthetic biological systems, a developer's and user's guide for clotho v2.0. In *Methods in Enzymology* (2011), vol. 498.

Design, Implementation, and Deployment of Electronic Datasheets

Cesar A. Rodriguez
BIOFAB
5885 Hollis St
Emeryville, CA 94608
cesarr@berkeley.edu

Michal Galdzicki
Biomedical and Health Informatics,
University of Washington
1959 NE Pacific St
Seattle, WA
mgaldzic@uw.edu

ABSTRACT
In the fields of engineering, datasheets are used by engineers to determine whether the behavior of a part or device will meet the requirements of a system in which the part or device might be used. In Synthetic Biology, a rapidly increasing number of standard biological parts are becoming available. Manual retrieval and selection of parts will become progressively time consuming. We proposed that electronic datasheets can facilitate the automation of part retrieval and selection. We hypothesize that automated part retrieval and selection will support the design of new biological devices and systems with increased functionality while decreasing development time. In this report, we describe how we designed, implemented, and deployed electronic datasheets at the BIOFAB. These new software artifacts lay the foundation for testing our hypothesis.

Keywords
Synthetic Biology, Information Management, Datasheets, Automation

1. INTRODUCTION
Canton and his colleagues note that "Quantitative descriptions of devices in the form of standardized, comprehensive datasheets are widely used in electrical, mechanical, structural and other engineering disciplines. A datasheet is intended to allow an engineer to quickly determine whether the behavior of a device will meet the requirements of a system in which the device might be used." Given this perspective, they developed a prototype datasheet for the standard biological part F2620 [1]. The "Canton" datasheet provides design information and performance data. The design of F2620 is depicted graphically using high-level symbols that indicate, for example, that the promoter R0040 is upstream of the RBS B0034. Performance data are provided in four sections on the same page that depict the static and dynamic performance, input compatibility, and reliability of F2620. The Canton datasheet is visually appealing and contains useful information.

At the International Open Facility Advancing Biotechnology (BIOFAB), we are producing collections of professionally engineered, high quality standard biological parts [2]. We have publically released data and information for 330 parts. As the production capacity of the BIOFAB matures and other biofabs around the world come on line, we can expect that the number of available standard biological parts will grow geometrically if not exponentially. This expected abundance of parts poses a new problem. How does a human engineer review and select the best parts for new biological devices and systems? Building on the work of Canton and his colleagues, we propose the development of electronic datasheets. Electronic datasheets can provide human-centered access to design information and performance data for the growing number of standard biological parts [3]. They have the added benefit of providing an application interface to the same data and information. By providing computers ready-access to the data and information, computers can be used to automate the process of retrieving and selecting the best parts for new biological devices and systems [4]. We hypothesize that the automated retrieval and selection of parts will eventually facilitate the design of new biological devices and systems with increased functionality while decreasing development time. The test of that hypothesis will be left for future work. This report describes how we designed, implemented, and deployed electronic datasheets at the BIOFAB.

2. SOFTWARE DEVELOPMENT
2.1 Design
The high-level requirements for the BIOFAB's electronic datasheets were:

- Provide design information and performance data for the BIOFAB's parts and constructs
- Provide the data and information in human- and machine-readable formats
- Make access to the data and information as simple as possible
- The data and information should be useful to biological device and system engineers

In meeting these requirements, it was decided to build a three-tiered application composed of a rich internet application called the Data Access Client (DAC) [5], a RESTful web service called the Data Access Web Service [6], and a relational database called the BIOFAB DB. Both the DAC and DAWS provide design information and performance data for BIOFAB parts and constructs. The DAC provides a human-readable interface. The DAWS provides the machine-readable interface. All that is required for using the DAC and DAWS is a modern web browser and an Internet connection. Evaluating if the data and information provided is useful to biological device and system engineers is an ongoing process. We directly elicit requirements from individuals using BIOFAB parts. We also provide means for users to communicate use cases and comments.

The DAC was designed as an information "dashboard" [3,7]. It integrates information about the entire inventory of available collections, parts, and constructs with design information and performance data of the individual components. For example, with the DAC a user is informed that there is a Random Promoter Library available. When the user selects the Random Promoter Library, the DAC displays the parts and constructs in the library. If the user selects an individual construct the DAC provides the DNA sequence, bulk gene expression, and gene expression per cell for the construct.

The DAWS is a RESTful web service [6]. It provides a simple and platform-independent application interface (API). Any modern software development kit has the necessary classes for developing a client application that can retrieve data and information from the DAWS. For example, the DAC uses Javascript and AJAX to retrieve data and information from the DAWS. It is straightforward to develop an Apple iPhone/iPad application with a similar capacity. There are efforts underway to have TinkerCell [8] , Clotho [9], and Gene Designer [10] retrieve data and information from the DAWS. The DAWS, provides data and information in standard formats which are described in more detail in section 2.2.

The data and information displayed by the DAC and provided by the DAWS is maintained in a relation database called the BIOFAB DB.

2.2 Implementation

The Data Access Client is developed with Javascript, Ext JS 3.3 and 4.0 [12], and Adobe Flex [13]. The GUI widgets for displaying a construct's DNA sequence and annotations are components from the Vector Editor project [14].

The Data Access Web Service is developed with Java, J2EE, BioJava [15], libSBOLj [16], and Gson [17] libraries. The DAWS, presently provides collections and parts information in Javascript Object Notation (JSON) [11]. Construct design information is provided in Genbank, INSD [18], Synthetic Biology Open Language (SBOL) JSON [16], and SBOL RDF [16] formats. Please attend the talk or visit the poster "Evolution of SBOL-design information exchange standard'' for a more detailed discussion of SBOL [19]. The DAWS provides construct performance summary data in CSV and JSON formats and raw data in JSON format. We are working on integrating construct design information, performance summary and raw data into one unified data model that can be serialized into SBOL JSON and SBOL RDF formats.

The DAC and DAWS are provided via the World Wide Web using the Glassfish application server [20].

2.3 Deployment

The Data Access Client and Web Service can be accessed from the BIOFAB website at: http://biofab.org/data. The source code for the DAC and DAWS is available at Github: https://github.com/BIOFAB/DataAccess. Issues with the DAC and DAWS are being tracked at Github: https://github.com/BIOFAB/DataAccess/issues.

3. CONCLUSIONS

We designed, implemented, and deployed electronic datasheets at the BIOFAB. These new datasheets provide human-centered access and an application interface to design information and performance data. They also facilitate the automation of part retrieval and selection. We hypothesize that automated part retrieval and selection will support the design of new biological devices and systems with increased functionality while decreasing development time. The electronic datasheets are new software artifacts that lay the foundation for testing our hypothesis.

4. ACKNOWLEDGMENTS

We are indebted to (in alphabetical order) Adam Arkin, Gaymon Bennett, Guillaume Cambray, Deepak Chandran, Marc Juul Christoffersen, Kevin Costa, Drew Endy, Joao Guimaraes, Tim Ham, Nathan Hillson, Leonard Katz, Allan Kuchinsky, Colin Lam, Quynh-Anh Mai, and Vivek Mutalik,

5. REFERENCES

[1] Canton, B., Labno, A., Labno, and Endy, A. 2008. "Refinement and standardization of synthetic biological parts and devices," *Nature Biotechnology*, vol. 26, no. 7, p. 787-793, Jul. 2008.

[2] "About the BIOFAB". (2011). Retrieved April 19, 2011, from http://www.biofab.org/about

[3] Dertouzos, M. 2001. *The Unfinished Revolution*. HarperCollins Publishers, Inc., New York, NY.

[4] Galdzicki, M., Rodriguez, C., Chandran, D., Sauro, H.M. Gennari, J.H. Standard Biological Parts Knowledgebase. PLoS ONE, 6 (2). e17005, 2011.

[5] "Rich Internet Applications". (2011). Retrieved April 19, 2011, from http://en.wikipedia.org/wiki/Rich_Internet_application

[6] "Representational State Transfer". (2011). Retrieved April 19, 2011, from http://en.wikipedia.org/wiki/Representational_State_Transfer

[7] "Dashboards". (2011) Retrieved April 19, 2011, from http://en.wikipedia.org/wiki/Dashboards_(management_information_systems)

[8] Chandran, D., Bergmann F.T., Sauro H.M. *TinkerCell: modular CAD tool for synthetic biology*. J Biol Eng. 2009 Oct 29;3:19.

[9] Densmore, D., Van Devender, A., Johnson, M., Sritanyaratana, N. 2009. Platform-based design environment for synthetic biological systems. In *Proceedings of TAPIA '09*. ACM, New York, NY, 24-29. DOI =10.1145/1565799.1565806

[10] Villalobos, Ness, Gustafsson, Minshull, Govindarajan. *Gene Designer: a synthetic biology tool for constructing artificial DNA segments.* BMC Bioinformatics 2006 7:285.

[11] "JSON". (2011). Retrieved April 19, 2011, from http://www.json.org/

[12] "Ext JS". (2011). Retrieved April 19, 2011, from http://www.sencha.com/products/extjs/

[13] "Adobe Flex". (2011). Retrieved April 19, 2011 from http://www.adobe.com/products/flex/

[14] "Vector Editor". (2011). Retrieved April 19, 2011 from http://code.google.com/p/vectoreditor/

[15] "BioJava". (2011). Retrieved April 19, 2011 from http://biojava.org/wiki/Main_Page

[16] "libSBOLj". (2011). Retrieved April 19, 2011 from https://github.com/mgaldzic/libSBOLj

[17] "google-gson". (2011). Retrieved April 19, 2011 from http://code.google.com/p/google-gson/

[18] "INSDC". (2011). Retrieved April 19, 2011 from http://www.insdc.org/documents.html

[19] "SBOL". (2011). Retrieved April 19, 2011 from http://www.sbolstandard.org/

[20] "Glassfish". (2011). Retrieved April 19, 2011 from http://glassfish.java.net/

Notes