IWBDA:2011 - International Workshop on Bio-Design Automation
San Diego Convention Center, San Diego, CA
June 6 - 7, 2011
http://www.biodesignautomation.org

48 DESIGN AUTOMATION CONFERENCE

Agilent Technologies

Autodesk

Raytheon
BBN Technologies

DNA 2.0

GINKGOBIOWORKS™

Hudson
Robotics, Inc.

life
technologies™

SynBERC
Synthetic Biology Engineering Research Center

The following students were provided financial support by our sponsors to attend the workshop

Laura Adam, 2nd year PhD, Virginia Bioinformatics Institute

Swapnil Bhatia, Postdoctoral researcher, Boston University

Benjamin Braun, 3rd year undergraduate, University of Texas at Austin

Deepak Chandran, 4th year PhD, University of Washington

Johann Desire Hai Elbaz, 3rd year PhD, The Hebrew University of Jerusalem

Michal Galdzicki, 4th year PhD, University of Washington School of Medicine

Roza Ghamari, 1st year MS, Boston University

Cristian Grecu, NSERC Postdoctoral Fellow, MIT

Victor Vasilev, 1st year PhD, Boston University

Foreword

Welcome to the Third International Workshop on Bio-Design Automation (IWBDA) at DAC.

IWBDA 2011 brings together researchers from the synthetic biology, systems biology, and design automation communities. The focus is on concepts, methodologies and software tools for the computational analysis of biological systems and the synthesis of biological systems.

Still in its early stages, the field of synthetic biology has been driven by experimental expertise; much of its success can be attributed to the skill of the researchers in specific domains of biology. There has been a concerted effort to assemble repositories of standardized components. However, creating and integrating synthetic components remains an ad hoc process. The field has now reached a stage where it calls for computer-aided design tools. The electronic design automation (EDA) community has unique expertise to contribute to this endeavor. This workshop offers a forum for cross-disciplinary discussion, with the aim of seeding collaboration between the research communities.

This year, the program consists of 15 talks and 11 poster presentations. These are organized into 5 sessions: *Gene Network Reconstruction, CAD Tools for Synthetic Biology, Biological Circuit Design, Biological Circuit Simulators, and Parts and Standardization*. In addition, we are very pleased to have three distinguished invited speakers: Adam Arkin, Chris Voigt, and Erik Winfree. Finally, we have a tutorial session by Ron Weiss, entitled “Circuit engineering principles for synthetic biology”.

We thank all the participants for contributing to IWBDA; we thank the Program Committee for reviewing abstracts; and we thank everyone on the Executive Committee for their time and dedication. Finally, we thank Agilent Technologies, Autodesk, Raytheon BBN Technologies, DNA 2.0, Ginkgo Bioworks, Hudson Robotics, Life Technologies, and the Synthetic Biology Engineering Research Center for supporting the workshop financially.

Doug Densmore, General Chair
Leonidas Bleris, General Secretary

Organizing Committee

IWBDA 2011 Program

Monday – June 6th

9am – 9:15am: Opening Remarks: Douglas Densmore (General Chair)

9:15am – 10am: Tutorial Session

Ron Weiss, "Circuit engineering principles for synthetic biology"

10:30am - 12pm: Tech. Talks Session 1 - *Gene Network Reconstruction*

1BDA.1 **Considerations for using integral feedback control to construct a perfectly adapting synthetic gene network**
Jordan Ang, Sangram Bagh, Brian P. Ingalls and David R. Mcmillen

1BDA.2 **Validation of Gene Network Models with Noisy Experimental Measurements using the Kalman Filter**
Yong-Jun Shin and Xiling Shen

1BDA.3 **A Coherent Feedforward Loop Robustly Regulating Asymmetric Cell Fate in Colon Cancer Stem Cells**
Pengcheng Bu, Yong-Jun Shin and Xiling Shen

12:00pm - 2pm: Lunch and Poster Session

2:00pm - 4:00pm: Tech. Talks Session 2 - *CAD Tools for Synthetic Biology*

2BDA.1 **GenoCAD 2.0: Thinking Inside the Box**
Mandy Wilson, Laura Adam and Jean Peccoud.

2BDA.2 **TASBE: A Tool-Chain to Accelerate Synthetic Biological Engineering**
Jacob Beal, Ron Weiss, Douglas Densmore, Aaron Adler, Jonathan Babb, Swapnil Bhatia, Noah Davidsohn, Traci Haddock, Fusun Yaman, Richard Schantz and Joseph Loyall

2BDA.3 **Toward Automated Selection of Parts for Genetic Regulatory Networks**
Fusun Yaman, Swapnil Bhatia, Aaron Adler, Douglas Densmore, Jacob Beal, Ron Weiss and Noah Davidsohn

2BDA.4 **A Software Stack for Specification and Robotic Execution of Protocols for Synthetic Biological Engineering**
Viktor Vasilev, Chenkai Liu, Traci Haddock, Swapnil Bhatia, Aaron Adler, Fusun Yaman, Jacob Beal, Jonathan Babb, Ron Weiss and Douglas Densmore

4:30pm - 6:00pm: Tech. Talks Session 3 - *Biological Circuit Design*

3BDA.1 **Programmed DNA computing Circuits as Cell-Mimicking systems for Nanomedical Applications**
Johann Elbaz, Fuan Wang and Itamar Willner

3BDA.2 **Automated Sequence Design for Nucleic Acid Circuits and Nanostructures using Structural Annotations**
Benjamin Braun, Xi Chen and Andrew Ellington

3BDA.3 **Robust Design of Genetic Circuits through Population Wide Error Detection and Correction**
Cristian Grecu, Jonathan Babb and Ron Weiss

7:00pm - 10:00pm: Dinner at Edgewater Grill
Address: 861 West Harbor Drive, San Diego, CA 92101
Phone: (619) 232-7581

Tuesday – June 7th

8:30am - 10:15am: General DAC Keynote

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

12:00pm - 2pm: Lunch and Poster Session

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"

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

5:00pm - 6:00pm: Panel Session

6:00pm - 6:30pm: Closing remarks and post-workshop future planning

Abstracts - Table of Contents

Monday – June 6th

9:15am – 10am: Tutorial Session
Ron Weiss, "Circuit engineering principles for synthetic biology"

Ron Weiss

Ron Weiss is an Associate Professor in the Department of Biological Engineering and in the Department of Electrical Engineering and Computer Science at the Massachusetts Institute of Technology. He received his PhD from MIT in 2001 and held a faculty appointment at Princeton University between 2001 and 2009. His research focuses primarily on synthetic biology, where he programs cell behavior by constructing and modeling biochemical and cellular computing systems. A major thrust of his work is the synthesis of gene networks that are engineered to perform *in vivo* analog and digital logic computation. He is also interested in programming cell aggregates to perform coordinated tasks using cell-cell communication with chemical diffusion mechanisms such as quorum sensing. He has constructed and tested several novel *in vivo* biochemical logic circuits and intercellular communication systems. Weiss is interested in both hands-on experimental work and in implementing software infrastructures for simulation and design work.

For his work in synthetic biology, Weiss has received MIT's Technology Review Magazine's TR100 Award ("top 100 young innovators", 2003), was selected as a speaker for the National Academy of Engineering's Frontiers of Engineering Symposium (2003), and received the E. Lawrence Keyes, Jr./Emerson Electric Company Faculty Advancement Award at Princeton University (2003). In addition, his research in Synthetic Biology was named by MIT's Technology Review Magazine as one of "10 emerging technologies that will change your world" (2004). He was chosen as a finalist for the World Technology Network's Biotechnology Award (2004), and was selected as a speaker for the National Academy of Sciences Frontiers of Science Symposium (2005). Over the last few years, Weiss has had several major publications in journals such as *Nature*, *Nature Biotechnology*, and *PNAS*.

Monday – June 6th

10:30am - 12pm: Tech. Talks Session 1 - *Gene Network Reconstruction*
1BDA.1 **Considerations for using integral feedback control to construct a perfectly adapting synthetic gene network** Jordan Ang, Sangram Bagh, Brian P. Ingalls and David R. Mcmillen
1BDA.2 **Validation of Gene Network Models with Noisy Experimental Measurements using the Kalman Filter** Yong-Jun Shin and Xiling Shen
1BDA.3 **A Coherent Feedforward Loop Robustly Regulating Asymmetric Cell Fate in Colon Cancer Stem Cells** Pengcheng Bu, Yong-Jun Shin and Xiling Shen

Considerations for using integral feedback control to construct a perfectly adapting synthetic gene network

[Extended Abstract] *

Jordan Ang†
Department of Chemical and Physical Sciences and Institute for Optical Sciences
University of Toronto Mississauga
Mississauga, Ontario L5L 1C6, Canada
j.ang@utoronto.ca

Sangram Bagh
Department of Chemical and Physical Sciences and Institute for Optical Sciences
University of Toronto Mississauga
Mississauga, Ontario L5L 1C6, Canada

Brian P. Ingalls
Department of Applied Mathematics
University of Waterloo
Waterloo, Ontario N2L 3G1, Canada

David R. McMillen
Department of Chemical and Physical Sciences and Institute for Optical Sciences
University of Toronto Mississauga
Mississauga, Ontario L5L 1C6, Canada

ABSTRACT

Homeostasis, a defining feature of living organisms, is the dynamic self-regulation of a system to maintain essential variables within limits necessary for acceptable performance in the presence of external disturbances [2]. Closely related is the phenomenon of sensory adaptation, in which a sensory system senses changes to its environment, responds with a change in its output signal, and then adjusts itself to enhance continued performance in the new environmental conditions. Familiar examples of sensory adaptation arise in the human visual and olfactory systems.

One of the major benefits of sensory adaptation is its respond-then-adjust behaviour. This ensures that the system is not caught permanently "responding" to a persistent stimulus. Consequently, responses to successive, additive stimuli are, over time, not additive themselves. This allows the system to respond to later-occurring, additional stimuli without overwhelming its response scope.

*A full version of this paper is available as [1].

†Presenting author.

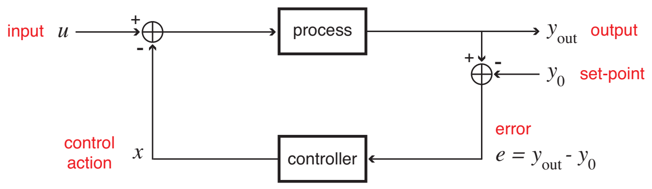


Figure 1: Block diagram circuit of a negative feedback-controlled system with an input signal, u, and an output signal, y_{out}, from the target system (the "process"). The error signal, e, is the difference between the current output and a target value (the "set-point"), and this is fed back through a "controller," which produces a "control action" signal, x, that is subtracted from the external input signal.

This study concerns itself with a special case of adaptation that occurs when a system's response signal relaxes back to its **exact** pre-stimulus value, irrespective of the value of a persistent stimulus. This is referred to as *perfect adaptation*. Beyond its application to sensory responses, perfect adaptation is important in control mechanisms that aim to maintain an output at a target value, whatever may happen to the input; in this case, perfect adaptation to a stimulus corresponds to perfect recovery of the desired target output after experiencing a perturbation to the input.

The importance of homeostasis in living organisms suggests that synthetic biology may benefit from an alliance with the more traditional engineering branch of control theory.

From a control theory perspective, adaptive self-regulation is achieved by negative feedback control. In its simplest representation, shown in Fig. 1, a "process" receives an input signal and produces an output signal. An additional component, the "controller," receives as input the output "error," that is, the difference between the process' current output and the desired output signal. The controller generates a control action that augments the original input signal prior to it reaching the process itself. A negative feedback controller continually draws the process output toward the desired output, promoting self-regulation and stability.

The effectiveness of a feedback a controller can be assessed in terms of (1) the *steady-state performance* and (2) the *transient performance*, after the onset of perturbation. The steady-state performance refers to the controller's ability to permanently eliminate the output error after a very long time (at steady state). If, after relaxation, the system output y_{out} does not agree with the desired output y_0 (the "set-point"), the difference is called the "steady-state error." If the steady-state error can be consistently eliminated, then the system is perfectly adapting, since its output always returns to its set-point. Transient performance, on the other hand, refers to the behaviour of the system's output response signal (and therefore the output error) during the system's relaxation to steady-state; this can be assessed via performance metrics such as the response curve's peak value, rise time, settling time, and possible oscillatory nature.

It has long been known to control theorists and engineers that a particular type of control structure called *integral feedback control* (in which the controller takes action directly related to the sum of past output errors) both guarantees and is necessary for perfect adaptation in the face of step-input perturbations. Consequently, integral control is a component of many engineered systems; it has also been identified in biological contexts [3, 4]. Accordingly, the implementation of this control strategy in a synthetic gene network is an attractive prospect. However, the nature of genetic regulatory networks (density-dependent kinetics and molecular signals that easily reach saturation) implies that the design and construction of an in-cell device of this sort is not straightforward.

We propose a generic two-promoter genetic regulatory network for the purpose of exhibiting perfect adaptation (Figure 2); our treatment is theoretical and highlights the challenges inherent in the implementation of a genetic integral controller. We will also present results from a numerical case study for a specific realization of this two-promoter network, "constructed" using commonly available parts from the bacterium *E. coli* and will illustrate the possibility of optimizing this network's transient response via analogy to a linear, free-damped harmonic oscillator. Furthermore, we will discuss variations to this two-promoter network allowing for perfect adaptation under conditions where first-order protein removal effects would otherwise disrupt the adaptation. Finally, we will discuss ongoing work to build a synthetic integral controller *in vivo*.

Keywords
synthetic biology; control systems; regulatory feedback; gene regulation

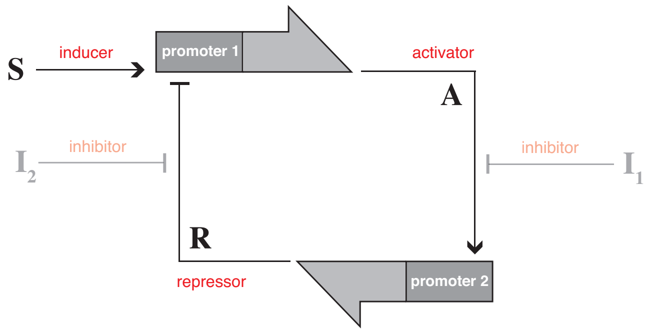


Figure 2: Gene network diagram of a two-promoter network capable of implementing integral control. We view this network as an input-output system with an external input signal S and output signal A, where italicized letters denote concentrations. Pointed arrows indicate activation of gene expression, while blunted arrows indicate repression. The wide block arrows represent the expression of proteins from promoters 1 and 2. A and R are their respectively expressed proteins. Promoter 1 is activated by S and repressed by R, the latter action completing a negative feedback loop; promoter 2 is activated by A. Each expression step functions as an integrator or sorts, and the expression from Promoter 2, specifically, that provides integration for the control action. I_1 and I_2 are optional regulators that serve as additional inputs to the system and can be used to tune the output set-point and control action levels.

1. ACKNOWLEDGMENTS
This work was funded by the Natural Sciences and Engineering Research Council of Canada (NSERC), the Ontario Research Fund (ORF), the Canada Foundation for Innovation (CFI), and the Ontario Graduate Scholarship in Science and Technology (OGS-ST).

2. REFERENCES

[1] J. Ang, S. Bagh, B. P. Ingalls, and D. R. McMillen. Considerations for using integral feedback control to construct a perfectly adapting synthetic gene network. *J. Theor. Biol.*, 266(4):723–738, 2010.

[2] W. B. Cannon. *A genetic switch.* W.W. Norton, New York, 1932.

[3] H. El-Samad, J. P. Goff, and M. Khammash. Calcium homeostasis and parturient hypocalcemia: An integral feedback perspective. *J. Theor. Biol.*, 214(1):17–29, 2002.

[4] T. M. Yi, Y. Huang, M. I. Simon, and J. Doyle. Robust perfect adaptation in bacterial chemotaxis through integral feedback control. *PNAS*, 97(9):4649–4653, 2000.

Validation of Gene Network Models with Noisy Experimental Measurements using the Kalman Filter

Yong-Jun Shin
Electrical and Computer Engineering
Cornell University
302 Phillips Hall
Ithaca, NY 14853
1-972-835-7376
yshin@cornell.edu

Xiling Shen
Electrical and Computer Engineering
Cornell University
411 Phillips Hall
Ithaca, NY 14853
1-607-254-8550
xs66@cornell.edu

ABSTRACT
Gene network dynamics are inherently stochastic and influenced by noise. Even though stochastic simulation algorithms, such as Gillespie's Stochastic Simulation Algorithm (SSA), can simulate the time evolution of various trajectories according to particular sequences of stochastic events, it is often difficult to compare the simulation results with experimentally measured data, which most likely correspond to a different set of random sequences. Unlike such algorithms, the Kalman filter, which is a well-established optimal estimation tool in science and engineering, allows model validation with real experimental data. Using the p53-MDM2 feedback loop as an example, we demonstrated in this work how the Kalman filter could be used to validate the mathematical model with noisy experimental measurement data. The validated model can also optimally predict the system state when new sets of experimental data are available.

Categories and Subject Descriptors
J.3 [Life and Medical Sciences]: Biology and genetics

I.6.5 [Simulation and Modleing]: Model Development (Modeling methodologies)

General Terms
Algorithms

Keywords
Gene network, Stochastic modeling, Optimal estimation, Kalman filter, p53-MDM2 feedback loop

1. INTRODUCTION
Cells consist of multiple, heterogeneous components such as genes and proteins that operate in a well-coordinated and robust manner. The interactions among these components are biochemical reactions governed by the law of mass action. For example, simple gene regulation, a two-gene network that serves as a basic building block for constructing more complex networks, can be expressed using the law.

$$A \xrightarrow{k_1} B \xrightarrow{k_2} null \tag{1}$$

Protein *A* is activating gene *b* as a transcription factor and, as a result, protein *B* is produced at the rate of k_1. Simultaneously, protein *B* is diluted or degraded at the rate of k_2. The coupled effect of these two paths on the rate of change (derivative with respect to time *t*) of protein B can be shown as an ordinary differential equation (ODE).

$$\frac{dB}{dt} = k_1 A - k_2 B \tag{2}$$

Biochemical reactions are inherently random processes and Eq. (2) is based on an assumption that there are a number of interacting molecules that average out the randomness so that the overall macroscopic behavior is "deterministic" or accurately predictable. However, this assumption is hardly justified in real cellular conditions as *in vivo* biochemical reactions involve a small number of molecules affected by the randomness to a greater extent [1,2]. In this context, there have been various approaches for modeling the random or stochastic feature of biochemical reactions (reviewed in [3]) and one of the most widely-used methods is Gillespie's Stochastic Simulation Algorithm (SSA). SSA is essentially a tool for numerically simulating the time evolution of well-stirred chemically reacting system by taking into account the randomness of molecular dynamics [4]. Based on the experimentally estimated or "guessed" rate constants and population size of each biochemical species, it can give us a more realistic view of the reaction dynamics compared to the deterministic approach.

In optimal control theory, there is a concept of "stochastic" optimal control, which recognizes the stochastic behavior (e.g., due to random disturbances) of a dynamic system and optimizes performance or stability on the average [5]. In order to optimally control in the presence of such randomness, a stochastic control system also needs to optimally estimate the system's dynamic state that provides "feedback" for closed-loop optimal control. The Kalman filter is a well-established method in various fields of science and engineering that optimally estimates the state of a dynamic system in real time by minimizing the difference between observed (true) and calculated (estimated) measurement values. The Kalman filter has many applications in aerospace and communication engineering, including the Global Positioning System (GPS). In this paper, we propose the Kalman filter as a novel way of modeling the dynamics of stochastic biological systems using the p53-MDM2 feedback loop as an example. Unlike SSA, the Kalman filter enables us to incorporate real experimental data into our model for various purposes such as model validation.

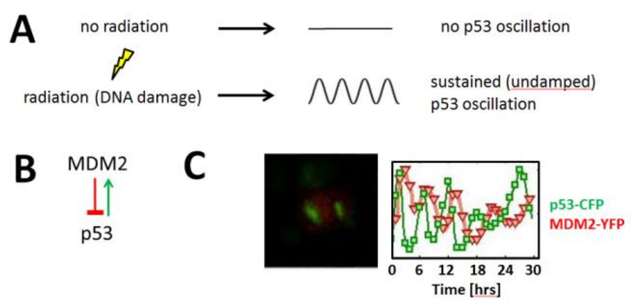


Figure 1. The p53 oscillation (A) When a cell is exposed to radiation, the p53 levels oscillate in a sustained (undamped) way. (B) The p53-MDM2 feedback loop. p53 activates MDM2 while MDM2 suppresses p53. (C) Oscillatory behavior of the p53-MDM2 feedback loop observed in MCF7 cell nucleus using functional p53-CFP and MDM2-YFP fusion proteins and time-lapse fluorescence microscopy [10].

The tumor suppressor p53 is one of the most studied proteins in cancer research [6,7]. Because cells are constantly damaged by various environmental and intrinsic factors, p53 is known to play a key role in deciding whether to repair the damage or activate apoptosis (programmed cell death). In cellular stress conditions, such as radiation-induced DNA damage, the p53 levels are reported to oscillate in a sustained (undamped) way as the p53 ubiquitination (suppression) by MDM2 is decreased (**Figure 1A**) [8]. In this loop, p53 transcriptionally activates MDM2, while MDM2 degrades p53 via ubiquitination (**Figure 1B**) [9]. The dynamic oscillatory behavior in MCF7 cell nucleus has been experimentally observed using functional p53-CFP and MDM2-YFP fusion proteins and time-lapse fluorescence microscopy (**Figure 1C**) [10].

2. RESULTS

A state-space representation of the p53-MDM2 feedback loop can be shown as

$$\begin{bmatrix} \frac{dx(t)}{dt} \\ \frac{dy(t)}{dt} \end{bmatrix} = \begin{bmatrix} -p_x & -p_{yx} \\ p_{xy} & -p_y \end{bmatrix} \begin{bmatrix} x(t) \\ y(t) \end{bmatrix} + \begin{bmatrix} w_x(t) \\ w_y(t) \end{bmatrix} \tag{3}$$

In Eq. (3), $x(t)$ and $y(t)$ stand for the state variables, p53 and MDM2, respectively. p_x, p_y, p_{xy}, and p_{yx} are the parameters of the model and $w_x(t)$ and $w_y(t)$ represent the process noise or random noise inherent in biological reactions. Eq. (3) basically illustrates how the components (x and y) of the system are dynamically coupled in the presence of stochastic noise (w_x and w_y). What we observe experimentally involves another type of noise, the measurement noise. For example, in case we are measuring x (p53), the measurement noise can be denoted as v_x as shown in Eq. (4).

$$x_m(t) = x(t) + v_x(t) \tag{4}$$

where x_m stands for the calculated value of measured x (p53), which is the sum of x and the measurement noise v_x. For fluorescence reporter imaging, v_x is dependent on the precision of the equipment (such as fluorescence microscope, camera, etc.) used for imaging.

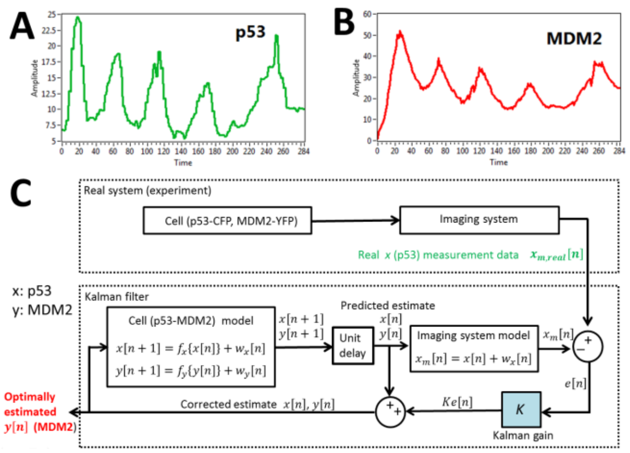


Figure 2. Optimal State Estimation Using the Kalman Filter (A) Experimentally measured p53 data. (B) Optimally estimated MDM2 values using the Kalman filter, which closely matches experimentally observed MDM2 (**Figure 1C**). (C) The block diagram of the real system and Kalman filter. The optimally estimated MDM2 values ($y[n]$) are gained by adding the predicted $y[n]$ estimate and $Ke[n]$, the product of the Kalman gain (K) and the error ($e[n]$) or difference between observed ($x_{m,real}[n]$) and calculated ($x_m[n]$) measurement values.

When modeling the p53-MDM2 feedback loop, the values of the parameters (p_x, p_y, p_{xy}, and p_{yx}) and covariance of process/measurement noise (w_x, w_y, and v_x) are often not available and some arbitrary values are commonly used for simulation. However, this means that we are not sure if our model truly reflects the real biological system since the model is based on the arbitrary values. As stated earlier, the Kalman filter can be used to validate the uncertain values used in the model by incorporating real experimental measurement data into the model. For example, given the experimentally measured x (p53) data (**Figure 2A**), we optimally estimated the MDM2 (one of the state variables) values (**Figure 2B**) using the Kalman filter. In order to implement a Kalman filter using computers, we need to convert Eq. (3) into a discrete form based on the state transition matrix method. The discrete form equations (the discretized system functions are shown as $f\{\}$) and block diagram are shown in **Figure 2C**. Note that discrete-time index n is used instead of t, which indicates continuous time. The optimally estimated MDM2 values ($y[n]$) are obtained by adding the predicted $y[n]$ estimate and $Ke[n]$, the product of the Kalman gain (K) and the error ($e[n]$) or difference between observed ($x_{m,real}[n]$) and calculated ($x_m[n]$) measurement values. As this estimation closely matches experimentally observed MDM2 (**Figure 1C**), we can assume that the uncertain values used in the model are reflecting real biological counterparts. The computation of the Kalman filter for simple gene regulation model, including the Kalman gain, has also been described in detail previously [11].

3. CONCLUSION

In this paper, using the p53-MDM2 feedback loop as an example, we demonstrated how the Kalman filter, a well-established optimal estimation tool in science and engineering, can be used to validate mathematical model with uncertainty. One additional feature enabled by the Kalman filter is that the resulting model can predict realistic levels of a signaling factor based on experimental measurements of the other factors. In future work, we plan to systematically analyze the effects of different values of the parameters and noise covariance on the optimal estimation of the state variables.

4. REFERENCES

1. Kaern M, Elston TC, Blake WJ, Collins JJ. (2005) Stochasticity in gene expression: From theories to phenotypes. Nat Rev Genet 6(6): 451-464.
2. Raj A, van Oudenaarden A. (2008) Nature, nurture, or chance: Stochastic gene expression and its consequences. Cell 135(2): 216-226.
3. Ullah M, Wolkenhauer O. (2010) Stochastic approaches in systems biology. Wiley Interdiscip Rev Syst Biol Med 2(4): 385-397.
4. Gillespie DT. (2007) Stochastic simulation of chemical kinetics. Annu Rev Phys Chem 58: 35-55.
5. Stengel RF. (1994) Optimal control and estimation. Mineola, NY: Dover.
6. Vogelstein B, Lane D, Levine AJ. (2000) Surfing the p53 network. Nature 408(6810): 307-310.
7. Levine AJ, Oren M. (2009) The first 30 years of p53: Growing ever more complex. Nat Rev Cancer 9(10): 749-758.
8. Lev Bar-Or R, Maya R, Segel LA, Alon U, Levine AJ, et al. (2000) Generation of oscillations by the p53-Mdm2 feedback loop: A theoretical and experimental study. Proc Natl Acad Sci U S A 97(21): 11250-11255.
9. Piette J, Neel H, Marechal V. (1997) Mdm2: Keeping p53 under control. Oncogene 15(9): 1001-1010.
10. Lahav G, Rosenfeld N, Sigal A, Geva-Zatorsky N, Levine AJ, et al. (2004) Dynamics of the p53-Mdm2 feedback loop in individual cells. Nat Genet 36(2): 147-150.
11. Shin YJ, Bleris L. (2010) Linear control theory for gene network modeling. PLoS One 5(9): e12785.

A Coherent Feedforward Loop Robustly Regulating Asymmetric Cell Fate in Colon Cancer Stem Cells

Pengcheng Bu
Electrical and Computer Engineering
Cornell University
311 Weill Hall
Ithaca, NY 14853
1-607-255-4109
pb345@cornell.edu

Yong-Jun Shin
Electrical and Computer Engineering
Cornell University
302 Phillips Hall
Ithaca, NY 14853
1-972-835-7376
yshin@cornell.edu

Xiling Shen
Electrical and Computer Engineering
Cornell University
411 Phillips Hall
Ithaca, NY 14853
1-607-254-8550
xs66@cornell.edu

Monday – June 6th

2:00pm - 4:00pm: Tech. Talks Session 2 - *CAD Tools for Synthetic Biology*
2BDA.1 **GenoCAD 2.0: Thinking Inside the Box** Mandy Wilson, Laura Adam and Jean Peccoud.
2BDA.2 **TASBE: A Tool-Chain to Accelerate Synthetic Biological Engineering** Jacob Beal, Ron Weiss, Douglas Densmore, Aaron Adler, Jonathan Babb, Swapnil Bhatia, Noah Davidsohn, Traci Haddock, Fusun Yaman, Richard Schantz and Joseph Loyall
2BDA.3 **Toward Automated Selection of Parts for Genetic Regulatory Networks** Fusun Yaman, Swapnil Bhatia, Aaron Adler, Douglas Densmore, Jacob Beal, Ron Weiss and Noah Davidsohn
2BDA.4 **A Software Stack for Specification and Robotic Execution of Protocols for Synthetic Biological Engineering** Viktor Vasilev, Chenkai Liu, Traci Haddock, Swapnil Bhatia, Aaron Adler, Fusun Yaman, Jacob Beal, Jonathan Babb, Ron Weiss and Douglas Densmore

GenoCAD 2.0: Thinking Inside the Box

Mandy Wilson
Virginia Bioinformatics Institute (VBI)
Virginia Tech (0477)
Blacksburg, VA 24061
(540) 231-1380
mandywil@vbi.vt.edu

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 the year since it became available as an open-source application, GenoCAD has undergone many refinements to help the user customize their environment; with the addition of import and export capabilities, GenoCAD is well-positioned for inclusion as part of a suite of synthetic biology applications.

Categories and Subject Descriptors

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

General Terms

Algorithms, Management, Design, Economics, Reliability, Experimentation, Security, Languages, Verification.

Keywords

Synthetic Biology, Systems Integration, User Interface Design.

1. INTRODUCTION

GenoCAD is a web-based computer-assisted design tool that allows users to develop DNA sequences that are validated against biological design strategies (also called grammars) [1] and restricted by functional elements (libraries and parts). Since the initial release of the GenoCAD.org web site [2], the project has morphed into an open source software development effort. The features introduced since the original release have been described in details in a recent publication [3]. This abstract presents an overview of the most significant functionality.

2. BACKGROUND

GenoCAD was designed to help synthetic biologists streamline the process of developing sequences. Traditionally, synthetic biologists had to develop sequences by hand, using Excel, text editors, or specialized molecular packages like VectorNTI to assemble series of A, C, G, and Ts to create new sequences – a lengthy and error prone process. GenoCAD allows users to develop their own abstraction hierarchy instead of working at the base level [4]. GenoCAD's workflow invites users to develop libraries of functional part sequences, like promoters or terminators, then assemble the parts into designs under the guidance of grammars that define which functional units can be used and in what order. An example of the use of GenoCAD's grammar to model a particular design strategy is the development of a grammar to design constructs compliant with different assembly standards [5] Grammars are an intrinsic part of the system, and while users can use public libraries of parts provided by the system, they can also develop their own libraries using either the parts within GenoCAD or their own parts. When the user finishes developing their design, they can save the sequence for future refinement or export the sequence for ordering sequences or for loading into other applications.

3. NEW FEATURES

3.1 Parts Management Tool

The Parts Management Module has been rewritten to make it even easier to develop libraries and manage parts. It was modeled after the shopping cart paradigm used by ordering-commerce systems. The user selects parts, either by browsing through the Public Parts listing, or by using the new Search tool, to find parts that meet their criteria; they can then put the desired parts into their "Cart". When they are done selecting parts, users can use their "Cart" of parts to create libraries or to add to existing libraries of parts. They may also add their own parts, either by entering their own, or by importing parts from other sources. Users may only view the public libraries and parts or those that they loaded themselves, so their work is isolated from other users. If a user wishes to share their parts, load them into a separate library, or use them in another software application, they can export their parts to FASTA or TAB-DELIMITED format.

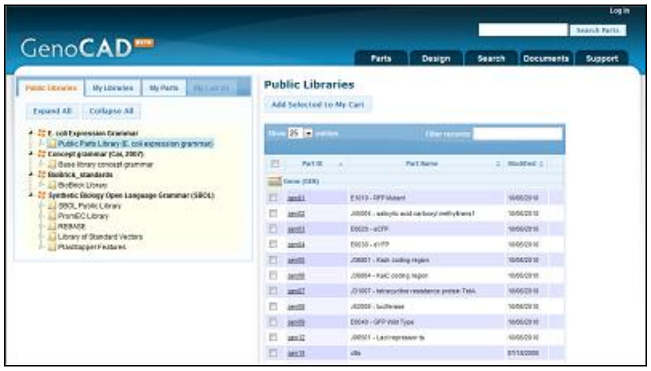


Figure 1: New Parts Management Module

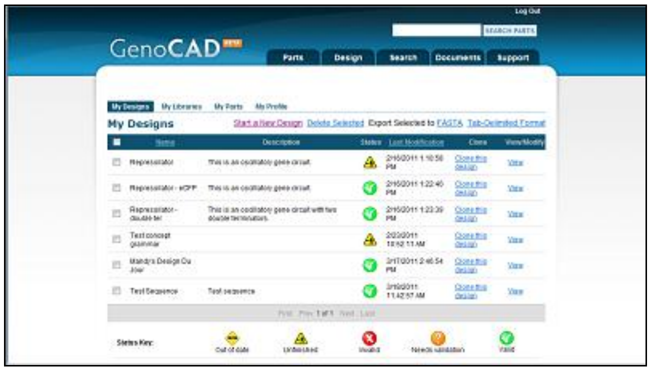
3.2 Search Capabilities

GenoCAD now provides three different ways to search the Parts Repository using the Lucene search engine. The user can do a quick search from any page, which does a text search on all of the fields associated with parts in the system. They can do a Basic Search which works in a similar fashion, except that Basic Searches can be saved for future queries, and, depending on the user's knowledge of Lucene syntax, can allow the users to build very convoluted queries. Finally, the user may use the Advanced

Search functionality to build fairly detailed searches without knowing Lucene syntax.

3.3 Enhancements to the My Designs Listing

Users have always been able to save their designs within GenoCAD for future review or modification. Now it is possible to export designs from the system in FASTA or Tab-Delimited formats, either for exchanging with other scientists or for loading into other systems. GenoCAD also supports a new validation feature from the My Designs page, so if an underlying change to the users' parts or parts libraries invalidates any designs (or changes the underlying design sequence associated with that design), the user can see a visual cue to let them know that something has changed, and they can respond accordingly.



My Designs

Start a New Design Delete Selected Export Selected to FASTA Tab-Delimited Format

	Name	Description	Status	Last Modification	Clone	View/Modify
☐	Repressilator	This is an oscillatory gene circuit.	⚠	2/16/2011 1:10:58 PM	Clone this design	View
☐	Repressilator - eGFP	This is an oscillatory gene circuit.	✓	2/16/2011 1:22:40 PM	Clone this design	View
☐	Repressilator - double ter	This is an oscillatory gene circuit with two double terminators.	✓	2/16/2011 1:23:39 PM	Clone this design	View
☐	Test concept grammar		⚠	2/22/2011 10:52:11 AM	Clone this design	View
☐	Mandy's Design Du Jour		✓	3/17/2011 2:46:54 PM	Clone this design	View
☐	Test Sequence	Test sequence	✓	3/9/2011 11:42:57 AM	Clone this design	View

First Prev 1 of 1 Next Last

Status Key: Out of date, Undefined, Invalid, Needs validation, Valid

Figure 2: Improved "My Designs" Screen with new functionality.

3.4 Improvements to the Design screen

The Design screen has been redesigned to make it more user-friendly. On load of a Design, the revalidation function is called to ensure that no recent changes have invalidated the existing design or sequence. The "step history" has been revised to allow users to track the selection of parts as well as categories (functional units), so it is even easier to go back to a specific point in the design process and make changes. The Save screen has been enhanced to make it easier for users to save a modified design as a new one instead of forcing the user to clone the existing design before applying modifications. In keeping with the goal of making GenoCAD flexible enough to become part of an integrated process, the user can now export the design sequence to GenBank format for easier loading into simulators and other synthetic biology applications

4. COMING FEATURES

4.1 Grammar Editor

One of the limitations of the current GenoCAD release is that, while users may develop their own parts and parts libraries, they are limited to the grammars provided by the GenoCAD system. This issue will be resolved in the form of a user-friendly grammar editor which will be made available to authenticated users. In addition to being able to build grammars tailored to the biologists' specific systems, the biologists will have the ability to share their grammars with their peers by exporting them, and they may also import grammars submitted by others. As with the users' personal libraries, user-defined grammars will only be available to the user who designed (or imported) them.

4.2 BLAST Search

Although the textual search capability provided in the current release of GenoCAD is a great asset for helping the users identify parts of relevance within the system, it is limited to textual matches. The BLAST search capability will allow users to search the GenoCAD database for analogous matches – sequences with regions of local similarity.

4.3 Improved Look and Feel

A new look and feel for GenoCAD is currently under development. One of the shortcomings of the current layout is that it is not readily clear to the user how Parts and Libraries are integral to the Design process (ie, in order to create a custom design, the user needs to first collect parts into specialized libraries.) The new GenoCAD interface will guide users through the process, and will also provide context-sensitive help screens tailored for each section.

4.4 Support for the New Synthetic Biology Open Language (SBOL)

VBI supports the development of the new Synthetic Biology Open Language, and is participating as one of the Beta testers. As SBOL is finalized, GenoCAD is pledged to providing SBOL support to promote integration with other SBOL-enabled software.

4.5 Availability of a GenoCAD Appliance

The publicly available instance of the application hosted at GenoCAD.org is a demonstration version used to illustrate the potential of the application. It is expected that GenoCAD users will want to host their own instance of GenoCAD within their organizations. This model makes it easier for IT personnel to ensure the integrity of the Intellectual Property included in the application database. GenoCAD users should also customize the backend database with custom grammars corresponding to the design strategies used in their organization. In order to facilitate GenoCAD deployment we are offering it as turnkey appliances that can be configured for workgroups of different sizes.

5. 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*, 23, 20 (Oct 15 2007), 2760-2767.

[2] Czar, M. J., Cai, Y. and Peccoud, J. Writing DNA with GenoCAD. *Nucleic Acids Res*, 37, Web Server Issue 2009), W40-47.

[3] Wilson, M. L., Hertzberg, R., Adam, L. and Peccoud, J. A Step-by-Step Introduction to Rule-Based Design of Synthetic Genetic Constructs Using GenoCAD. *Methods Enzymol.*, in press2011).

[4] Goler, J. A., Bramlett, B. W. and Peccoud, J. Genetic design: rising above the sequence. *Trends Biotechnol.*, 26, 10 (Oct 2008), 538-544.

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

TASBE: A Tool-Chain to Accelerate Synthetic Biological Engineering

Jacob Beal[1] Ron Weiss[2] Douglas Densmore[3] Aaron Adler[1]
Jonathan Babb[2] Swapnil Bhatia[3] Noah Davidsohn[2] Traci Haddock[3]
Fusun Yaman[1] Richard Schantz[1] Joseph Loyall[1]
[1]BBN Technologies 10 Moulton Street Cambridge, MA, USA 02138
[2]MIT 77 Massachusetts Ave Cambridge, MA, USA 02139
[3]Boston University 8 Saint Mary's St. Boston, MA, USA 02215
{jakebeal,fyaman, aadler, rschantz, jloyall}@bbn.com,
{rweiss, ndavidso, jbabb}@mit.edu, {dougd,swapnilb,thaddock}@bu.edu

1. MOTIVATION

There is a pressing need for design automation tools for synthetic biology systems. Compared to electronic circuits, cellular information processing has more complex elementary components and a greater complexity of interactions among components. Moreover, chemical computation within a cell is strongly affected both by other computations simultaneously occurring in the cell and by the cell's native metabolic processes and its external environment. This complexity implies an engineering work-flow that is currently highly iterative, error-prone, and extremely slow—critical problems that must all be addressed in order to realize the potential of synthetic biology.

In recent years, an assortment of tools have emerged, each independently addressing various parts of the design automation challenge. For example, the RBS calculator[10] helps design ribosome binding sites, the GeneDesign suite[9] optimizes coding sequences, and TinkerCell[5] is a graphical tool for visualizing and designing regulatory networks, to name only a few. A few projects, such as Eugene[4], GenoCAD[6], GEC[8], and Proto[2], have even begun extending design up to higher level languages. The time is now right to begin connecting and organizing such efforts together into a tool-chain for integrated end-to-end design and construction of synthetic biology systems.

In the TASBE project, we are developing one such approach to factoring the problem of design and assembly into sub-problems which can be more readily solved. Practitioners using our tool-chain will be able to design organisms using high level behavior descriptions, which are automatically transformed into genetic regulatory network designs, then assembled into DNA samples ready for *in vivo* execution. The tool-chain is also free and open software, which will allow researchers to incorporate their own design tools, thereby disseminating their results to the community and enhancing the capabilities of the tool-chain.

2. PROTOTYPE TOOL-CHAIN

TASBE is a modular tool-chain with both forward and backward information flow, designed to support multiple

Work sponsored by DARPA I2O under contract HR0011-10-C-0168; the views and conclusions contained in this document are those of the authors and not DARPA or the U.S. Government.
IWBDA '11 San Diego, California, USA

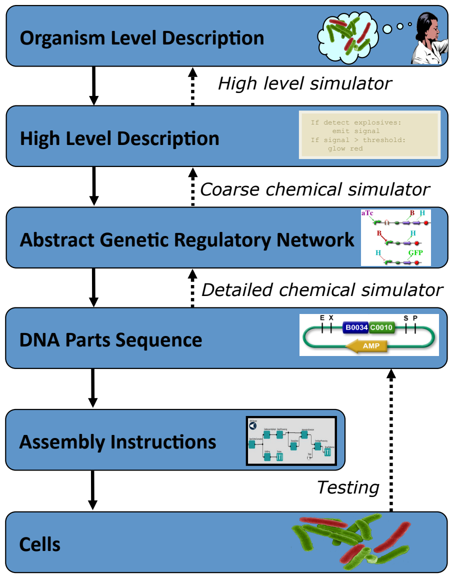


Figure 1: Prototype TASBE tool-chain architecture.

compatible workflows and integration of additional design tools developed by the broader research community. Figure 1 shows the prototype TASBE architecture.

At the highest level, TASBE allows expression of the desired system function using a biologically-focused high-level programming language. TASBE then uses a compiler to transform this design systematically into an abstract genetic regulatory network design (AGRN), which specifies the types of DNA parts and interactions needed to implement the network. Optimization is performed on this network to allow a parsimonious realization under constraints of metabolic load and biological part availability. The abstract network is then instantiated by mapping the network components to existing biological parts with the help of a parts library that documents the chemical properties of such parts. The properties of these parts are established through a process of DNA device characterization, in which the input/output transfer functions of biological devices are measured in support of abstractions such as the generalized digital static discipline. Finally, the desired DNA sequence is assembled using standard laboratory automation robotics, which execute an automatically generated sequence of biological protocols.

At each stage, simulation and testing tools help with system debugging and provide a counter-flow of information (black dotted arrows in Figure 1) to the human designer. This approach maximizes the chance of catching design flaws early and minimizes the number of times that an actual biological system must be assembled and tested. Our prototype implementation contains the following components:

- *A biologically-focused high-level language:* A high-level programming language is a critical component of a tool-chain because it allows cell behavior to be described succinctly, allowing non-biologists to participate in the design process. We have chosen Proto [1], a spatial computing language, since it offers a unique continuous parallel dataflow semantics that is a good match for current biological computating models.

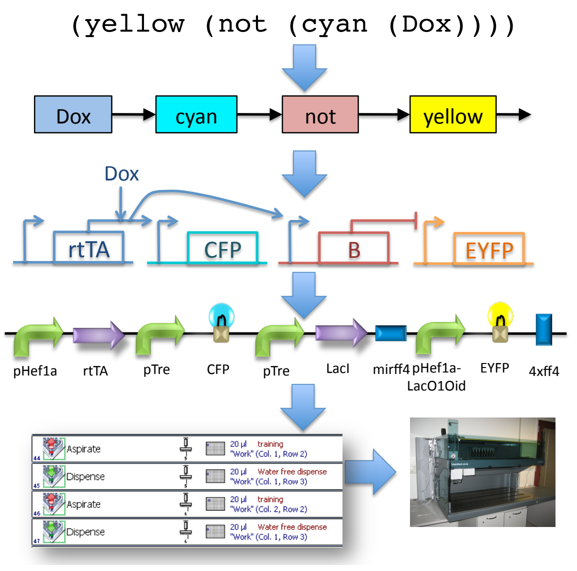
- *BioCompiler:* We have implemented a motif-based compiler[3] that transforms dataflow computations into AGRNs. The BioCompiler maps Proto primitives to genetic network motifs to produce an AGRN, then uses adapted versions of traditional compiler optimizations, such as dead code elimination and copy propagation, to reduce AGRN complexity.

- *Mapping between abstract and available DNA parts* To realize an AGRN, each AGRN element must be mapped to an available DNA part, and this mapping must be chosen such that the resulting network will function correctly within the bounds of chemical, physical, and metabolic constraints. Our MatchMaker system uses characterization data and a generalized version of the digital static discipline to guide a heuristic search for a correct implementation of an AGRN, then performs a greedy linearization of this network to yield a sequence of available DNA parts to be assembled.

- *DNA Assembly Planning* Laboratory automation equipment can execute DNA assembly protocols faster and more reliably than human technicians. Our Assembly Planner system, implemented in the Clotho framework[7], inputs a sequence of DNA parts and a protocol type and produces a set of assembly instructions for the laboratory robot to execute the protocol and produce a DNA sample ready for *in vivo* execution. By accessing laboratory inventory, this system can also optimize the number of assembly steps and the reusability of assembly mid-products.

- *Robotic Assembly* Finally, DNA assembly protocols are executed on a laboratory automation robot. At present, we are using a Tecan Evo 150, and two assembly protocols: BioBricks assembly for *E. coli* and a modified Gibson/Gateway protocol that uses magnetic beads and magnetic blocks to automate manual steps in DNA assembly.

Figure 2 shows a simple program, "If the cell detects the Dox molecule it fluoresces cyan, otherwise it fluoresces yellow," as it passes through the stages of the tool-chain. We have achieved an end-to-end integration of the software tools, and have used it to produce part sequence designs equivalent to known good designs executing *in vivo* in the Weiss laboratory. Once our current work on assembly protocol implementation is complete, these designs will be transformed into DNA samples, providing the first end-to-end compilation of high-level programs into DNA samples.



```
44 Aspirate  5  20 µl training
                "Work" (Col. 1, Row 2)
45 Dispense  5  20 µl Water free dispense
                "Work" (Col. 1, Row 3)
46 Aspirate  6  20 µl training
                "Work" (Col. 2, Row 2)
47 Dispense  6  20 µl Water free dispense
                "Work" (Col. 1, Row 3)
```

Figure 2: A simple program, to fluoresce cyan in the presence of Dox and yellow in its absence, passing through the stages of the tool-chain.

Note that although the prototype architecture is mostly linear, linearity is not an assumption of the architecture, but merely reflects the limited set of components in the initial prototype. Additionally, we have chosen Clotho[7] for an implementation framework for many of these components as its "app store" API and data management model facilitate integration of these tools and potentially many others. Finally, the components of our tool-chain are all designed for extensibility: Proto and the BioCompiler accept new primitive and motif definitions, MatchMaker can be augmented with additional design constraints, the Assembly Planner can be extended for new protocols, and the robotic automation systems are being controlled with a generalized API that should be portable to other robotic systems.

3. CONTRIBUTIONS

TASBE is an initial step towards achieving automation in synthetic biology. We have developed an initial set of prototype tools that can serve as a backbone for developing a larger, more comprehensive tool-chain. Our choice of decomposition attempts to minimize interaction between subtasks, thereby simplifying a potentially intractable problem of design at a cost of possible increased cost and inability to find solutions at the edge of design viability. Initial results point toward likely success of the TASBE approach. We intend to offer this tool-chain as an open platform for the research community, potentially multiplying the impact of each new design tool and significantly speeding the progress of synthetic biology research.

4. REFERENCES

[1] J. Beal and J. Bachrach. Infrastructure for engineered emergence in sensor/actuator networks. *IEEE Intelligent Systems*, pages 10–19, March/April 2006.

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

[3] J. Beal, T. Lu, and R. Weiss. Automatic compilation from high-level languages to genetic regulatory networks. In *Proceedings of IWBDA: International Workshop on Bio-Design Automation at DAC*, June 2010.

[4] Berkeley Software 2009 iGem Team. Eugene. http://2009.igem.org/Team:Berkeley_Software/Eugene, October 2009.

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

[6] M. Czar, Y. Cai, and J. Peccoud. Writing DNA with GenoCAD. *Nucleic Acids Research*, 37(W40-7), 2009.

[7] D. Densmore, A. V. Devender, M. Johnson, and N. Sritanyaratana. A platform-based design environment for synthetic biological systems. In *TAPIA '09*, pages 24–29. ACM, 2009.

[8] M. Pedersen and A. Phillips. Towards programming languages for genetic engineering of living cells. *Journal of the Royal Society Interface*, 2009.

[9] S. Richardson, S. Wheelan, R. Yarrington, and J. Boeke. Genedesign: rapid, automated design of multikilobase synthetic genes. *Genome Res.*, 16(4):550–6, April 2006.

[10] H. Salis, E. Mirsky, and C. Voigt. Automated design of synthetic ribosome binding sites to control protein expression. *Nature Biotechnology*, 27:946–950, 2009.

Toward Automated Selection of Parts for Genetic Regulatory Networks

Fusun Yaman[1], Swapnil Bhatia[2], Aaron Adler[1],
Douglas Densmore[2], Jacob Beal[1], Ron Weiss[3], and Noah Davidsohn[3]
[1]BBN Technologies 10 Moulton Street Cambridge, MA, USA 02138
[2]Boston University 8 Saint Mary's St. Boston, MA, USA 02215
[3]MIT 77 Massachusetts Ave Cambridge, MA, USA 02139
{fusun, aadler, jakebeal}@bbn.com, {swapnilb, dougd}@bu.edu,
{rweiss,ndavidso}@mit.edu

1. INTRODUCTION

The state-of-the-art techniques in synthetic biology require practitioners to design organisms at the DNA level. This low-level manual process becomes unmanageable as the size of a design grows. In electronic computing, high-level languages and compilers have enabled computer scientists to produce more sophisticated programs more quickly and with less effort. The same principles can be applied to synthetic biology, making the design of large and complex systems tractable.

In this paper, we define the problem of going from high-level descriptions of behavior to DNA sequences, and develop an automated solution using constraint satisfaction and optimization algorithms. Our research builds on the BioCompiler [1], which compiles an organism-level behavioral description into a network of abstract biological parts. This paper focuses on transforming such an abstract network into a concrete network realized with specific DNA sequences.

2. BACKGROUND

There are several natural mechanisms that can be manipulated in a cell to achieve a desired behavior. Our work focuses on transcriptional logic systems, where the computation is through the execution of a transcriptional network. The steps of this execution are: 1) *transcription*—the copying of a region of DNA to RNA—a process that can be regulated by protein-promoter interaction; 2) *translation*—the linking together of amino acids in the order specified in the RNA sequence into a protein; and 3) *regulation*—the suppression or activation of DNA regions by the protein produced.

In the Clotho [2] ontology, a *feature* is a DNA sequence responsible for a specific biochemical behavior. We consider transcriptional networks comprising two types of features: promoters and sequences coding for regulating proteins. The relationship between a regulating protein and the promoter preceding the DNA region containing a gene determines if and when that gene can be transcribed and then translated. A regulating protein can repress or activate a promoter. Repressors disable the ability of a promoter to initiate transcription; activators enhance its ability to initiate transcription. Consequently, transcription of a gene downstream of a promoter is controlled by activators and repressors of the promoter. Moreover, a promoter may be regulated by multiple proteins, thus implementing relationships analogous to boolean logic operations like NOT, AND, and IMPLIES.

Work sponsored by DARPA I2O under contract HR0011-10-C-0168; the views and conclusions contained in this document are those of the authors and not DARPA or the U.S. Government.
IWBDA '11 San Diego, California, USA

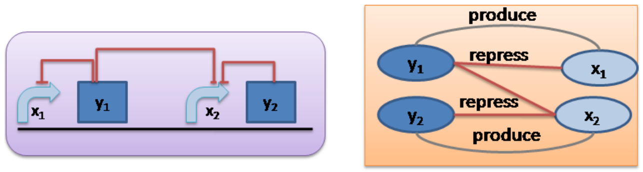


Figure 1: GRN visualizations: (Left) DNA sequence representation with rectangles as protein coding sequences (PCS) and block arrows as promoters. Red lines from PCS to promoters indicate repression. (Right) The same GRN in a graph representation with labeled edges.

A **genetic regulatory network** (GRN) is a bipartite graph with labeled edges and each vertex associated with a promoter (*i.e.*, promoter vertex) or a protein coding sequence (*i.e.*, protein vertex). In a GRN, the edges are always between a promoter vertex and a protein vertex. The edges have one of the following labels *produce, repress,* or *activate*. GRN visualizations are shown in Figure 1.

An **abstract genetic regulatory network** (AGRN) is similar to a GRN, with the difference that nodes are associated with a set of promoters or a set of protein coding sequences. An AGRN thus corresponds to a collection of GRNs. Our goal is to pick a near-optimal of these GRNs and choose a minimal set of available DNA parts covering the GRN so that it can be implemented in a cell. This translation of an AGRN to a GRN requires two kinds of solutions: a *topological solution*, choosing a single feature from the set associated with the node in the AGRN, such that all repression and activation relationships are satisfied, and a *quantitative solution*, which ensures that the choices also satisfy chemical signal compatibility constraints.

3. TOPOLOGICAL SOLUTION

The qualitative relationships between biological features are discovered by biologists experimentally. We define a *feature database* as a bipartite graph with each node associated with a single feature and the edges labeled from $\{repress, activate\}$. This is very similar to the GRN definition except the feature database does not have edges labeled

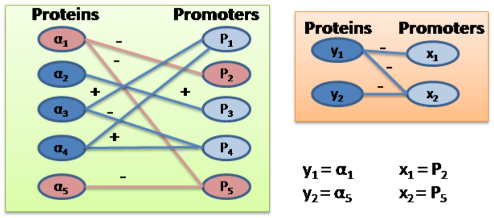


Figure 2: The constraint graph (right) is isomorphic to the strict subgraph induced by the vertices $\alpha_1, \alpha_5, P_2, P_5$ from the feature database (left).

produce. The left side of Figure 2 is a feature database (edge labels -/+ are short hand notations for *repress* and *activate* respectively), and the vertices are associated with proteins $\alpha_1, \ldots, \alpha_5$ and promoters $P_1, \ldots, P_5$.

We assume existence of a feature database containing such relationships. In translating an AGRN to a GRN by mapping each node to a feature, we need ensure that:

- The edges in the GRN are supported by the feature database. If the features we selected are not biologically capable of interacting in the desired manner, the GRN is not executable.
- The feature database does not imply additional relationships between the features of the GRN nodes. If two features are known to interact with each other, then whether or not we intended them to, they will interact when implemented in the cell, possibly disrupting the designed behavior.

For mapping the nodes of an AGRN to features we will ignore the production edges (edges labeled as *produce*) in the AGRN because they are unrelated to the two constraints above since any promoter can produce any protein. The **constraint graph**, induced by an AGRN is the same graph as the AGRN except the production edges are dropped. The right graph in Figure 2 is the constraint graph induced by the AGRN in Figure 1 where each x_i is associated with all promoters and y_i with all proteins.

In the *topological solution* of an AGRN w.r.t. a feature database, we look for a subset of vertices S in the feature database such that a *strict subgraph* of the feature database that contains only those vertices in S and all edges between them is isomorphic to the AGRN. More formally, a *topological solution* to an AGRN w.r.t. a feature database is a mapping between the nodes of the AGRN and a strict subgraph S of the feature database such that: 1) The mapping is an isomorphism between the induced constraint graph of the AGRN and the strict subgraph induced by S; 2)The labels of the edges in the constraint graph matches the edge labels in the strict subgraph induced by S; 3) For every node n in the AGRN, the set of features associated with n contains the feature associated with the node that n is mapped to. A topological solution to the AGRN in Figure 1 w.r.t. to the feature database in Figure 2 maps node y_1 to α_1, y_2 to α_5, x_1 to P_2 and x_2 to P_5.

4. QUANTITATIVE SOLUTION

Just like a digital circuit is composed of several devices, a GRN is a composition of **biological devices**. In a GRN, there is a device per promoter node. The inputs of the device are the protein nodes that are linked to the promoter with

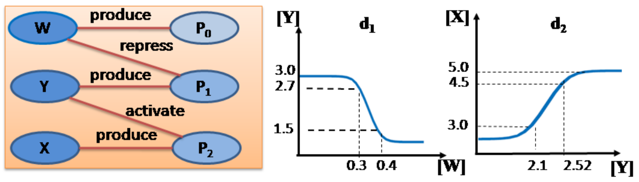


Figure 3: The concentration of Y is a function of W. The concentration of X is a function of Y. d_1 is signal compatible with d_2 because $2.7 > 2.52$ and $1.5 < 2.1$.

repression and activation edges. The outputs of the device are the protein nodes that are linked to the promoter with production edges. Without loss of generality, we will assume that each device has only one output. We will denote a device as $d = \langle \mathcal{I}, p, o \rangle$ where $\mathcal{I}$ is set of proteins which are inputs, p is a promoter and o is the output protein. In Figure 3, the GRN on the left has two devices: $d_1 = \langle \{W\}, P_1, Y \rangle$ and $d_2 = \langle \{Y\}, P_2, X \rangle$.

A device defines a function from the concentration of input proteins to concentration of output protein. The sigmoidal curves on the right side of Figure 3 are examples of such functions for devices d_1 and d_2 (single-input devices). The characteristics of the curve (slope, height, etc.) come from the biochemical properties of the features that make up the device. The slope (increasing vs. decreasing) is a function of repression or activation relationships.

Adapting from digital logic, any output o of the device higher (lower) than $high_o$ (low_o) will be considered as boolean true (false). Any output value between low_o and $high_o$ has an ill-defined truth value. Similar assumptions hold for the device inputs. In Figure 3, looking at the curve for d_1, the low value for the output Y is 1.5 and the high value is 2.7. The high and low values per input and output are the **specifications** of a device. We denote the specifications of a device $d = \langle \mathcal{I}, p, o \rangle$ as $S_d = \langle h, l \rangle$ where h (similarly l) is a function from $\mathcal{I} \cup \{o\}$ to reals for the high (similarly low) signal threshold.

Consider two devices, d with the specifications $\langle h, l \rangle$ and d' with specifications $\langle h', l' \rangle$. If the output o of d is an input of d' then d is **signal compatible** with d' iff $h(o) > h'(o)$ and $l(o) < l'(o)$. Note that if the output of first device is not an input to the second, by definition the devices are compatible. Finally, a GRN G is a **quantitative solution** to a AGRN A iff G corresponds to a topological solution of A w.r.t. a feature database and every device pair in G is signal compatible.

5. PROGRESS & RESULTS

By a reduction from subgraph isomorphism, we have shown that finding a topological solution is NP-Complete. To address this intractability, we have developed heuristic-based algorithms for topological and quantitative solutions and implemented them in a Clotho [2] app called MatchMaker.

6. REFERENCES

[1] J. Beal, T. Lu, and R. Weiss. Automatic compilation from high-level languages to genetic regulatory networks. In *Proceedings of IWBDA*, June 2010.

[2] D. Densmore, A. V. Devender, M. Johnson, and N. Sritanyaratana. A platform-based design environment for synthetic biological systems. In *TAPIA '09*, pages 24–29. ACM, 2009. (www.clothocad.org).

A Software Stack for Specification and Robotic Execution of Protocols for Synthetic Biological Engineering

Viktor Vasilev, Chenkai Liu, Traci Haddock, Swapnil Bhatia[1],
Aaron Adler, Fusun Yaman, Jacob Beal[2], Jonathan Babb, Ron Weiss[3], and
Douglas Densmore[1],
[1]Department of Electrical and Computer Engineering, Boston University, Boston, MA
[2]BBN Technologies, Cambridge, MA
[3]Department of Electrical Engineering and Computer Science, MIT, Cambridge, MA
{vvasilev, nykai55, thaddock, swapnilb, dougd}@bu.edu,
{aadler, fyaman, jakebeal}@bbn.com, {jbabb, rweiss}@mit.edu

1. INTRODUCTION AND MOTIVATION

Synthetic biology is an emerging field in which biologists modify or design the behavior of organisms to engineer systems that perform computation in diverse biological applications. Synthetic biologists design such a complex system by composing basic functional units—e.g., a promoter or a gene—into a regulatory network that exhibits the desired transcriptional behavior. As the desired behavior becomes more sophisticated, the size of the network grows, the complexity of the design becomes an impending concern [2], and its assembly and verification, an arduous task. The design complexity may be addressed by powerful design tools, large circuits may be assembled using novel assembly protocols, and the result may be verified by executing a comprehensive test suite. Performing design, assembly, or verification manually however, is tedious, error-prone, not easily reproducible, and hence unscalable. We address the problem with a chain of tools [3, 8], each tool providing an optimized solution to the problem at a discrete grade of abstraction. This report focuses on the assembly and verification stage—specifically, the design of a software stack for high level specification of biological protocols used in assembly and verification.

The primitive genetic parts comprising a larger system are constructed to be compatible with one of the standard assembly protocols such as BioBricks [7] or BioBytes [6]. Assembly planning algorithms [5] take a library of such assembly-ready parts and a list of genetic devices to be assembled, and produce an optimized hierarchy of assembly steps. The execution of each step in this sequence requires the execution of one or more basic biological procedures such as a ligation or a restriction digest. A critical gap exists between the specification of assembly plans at this level—a sequence of biological procedures—and the programming interface of liquid handling robots. Liquid handling robots are typically programmed by chaining together a sequence of basic actions such as pipetting from a specific well in one plate to another well, moving a plate or other labware from a specific location on the robot deck to another, or an action involving interfacing with auxiliary instruments such as incubators or mixers. While low-level access to a robot may afford greater flexibility, it has at least two disadvantages: 1) Low-level languages make the specification of complex protocols and their composition into higher-level human understandable units difficult. The absence of modularity, powerful and well-defined composition operators, and the need to manage labware minutiae makes low-level programming difficult, unscalable, and the resulting code unmaintainable. 2) Specifying protocols with a vendor-provided low-level language ties the resulting code to a particular robot architecture, removing the secondary advantages of programmed automation: portability, open exchange, and accelerated development through reuse. In this work, we bridge the gap between high-level assembly protocols and a low-level robot interface by designing a high-level language for biological protocol specification called Puppeteer, and a robot Hardware Abstraction Layer. The proposed design will allow high level specification and composition of protocols, accelerate protocol design, and make the assembly and verification of large systems tractable. Below, we describe our design, its advantages, the current state of its implementation, and plans for future work.

Work sponsored by DARPA I2O under contract HR0011-10-C-0168; the views and conclusions contained in this document are those of the authors and not DARPA or the U.S. Government.
IWDBDA June 2011, San Diego, CA

2. ARCHITECTURE

Our solution comprises a five-layer stack as illustrated in Figure 1. Using the Clotho platform [4], we develop two applications for specifying and executing biological protocols. The Assembly Planner [5] is the end-point of an end-to-end design workflow [3] that produces an assembly plan for synthetic biological devices, with each assembly step annotated with the name of a biological protocol. Each such protocol itself may be fully specified using another Clotho application called PuppetShow, which provides an environment for writing, testing, debugging, and executing biological protocols.

The protocols are written in a new high-level language called Puppeteer. The Language layer comprises the Puppeteer interpreter and linker. A protocol specified in Puppeteer may contain Puppeteer instructions as well as references to previously created Puppeteer programs available in a library. The Language layer expands and translates a Puppeteer protocol to a sequence of low-level commands expressed in a Common Robot Instruction Set (CRIS). CRIS provides a standardized instruction set that high level biological protocol languages like Puppeteer may assume to be supported by any robot. Any high-level language may produce CRIS programs and any robot vendor may support a superset of CRIS: this decouples robot hardware details from biological protocol and specification details and supports our goal of portability and protocol library reuse. The Hardware Layer—the external control and I/O interface of a robot—is wrapped under a Hardware Abstraction Layer (HAL). Vendor-provided software for programming the robot may be proprietary and is used to control the robot. An interface to it is provided by a software bridge, which maps protocols expressed in CRIS to sequences of native robot instructions.

The Resource Management layer maintains resource state information and provides a standardizable high-level interface for initializing, requesting, naming, aggregating, and accessing resources to the Language layer, analogous to a "system call" suite. This interface supports our goal of removing the minutiae of resource management from the protocol specification language.

3. IMPLEMENTATION

We have finished implementation of the Assembly Planner, with protocol name annotation, and will shortly begin work on PuppetShow. Our current implementation of the Puppeteer Language layer comprises an interpreter that recognizes ten primitive instructions and a library of two protocols. In its current embodiment, the interpreter is a command line program that accepts Puppeteer instructions and uses JSON objects for communication with the HAL. The interpreter can be run in one of two modes: *user* mode and *API* mode. In user mode, the user inputs a Puppeteer instruction, and the interpreter, after parsing and executing it, returns information about the result. In API mode, the interpreter uses JSON objects to communicate with the Applications layer allowing any application or programming language to interact with it. We have also begun implementation of a bridge for the Tecan Freedom Evo 150 robot. Our software stack is independent of the Tecan robot and API—we use the Tecan as a prototypical testbed for implementing the proposed stack. Our current implementation is capable of accepting a BioBrick assembly plan, linking it to a Puppeteer protocol library, and executing it on a Tecan robot or simulator.

4. RELATED AND FUTURE WORK

To our knowledge, Biocoder [1] is the only language for high-level specification of biological protocols with the goal of clarity, precision, and eventually, automation and reuse of protocol code. Biocoder is constructed as a C++ library thus allowing protocol specifications to leverage C++ features. When a protocol is compiled, a goal of the BioCoder library is to produce a human executable list of unambiguous English-language instructions equivalent to the original biological protocol. In addition to supporting these goals, Puppeteer will be the first to achieve an end-to-end translation from a high-level biological protocol specification to a sequence of actions on a robot.

Beyond protocol specification, Puppeteer also aims to aid the testing of biological protocols and ease the verification of large systems, through specialized instructions. The Resource manager may also exploit hardware parallelism and schedule actions or protocols on one or more robots efficiently. We plan to pursue these goals in subsequent versions of Puppeteer.

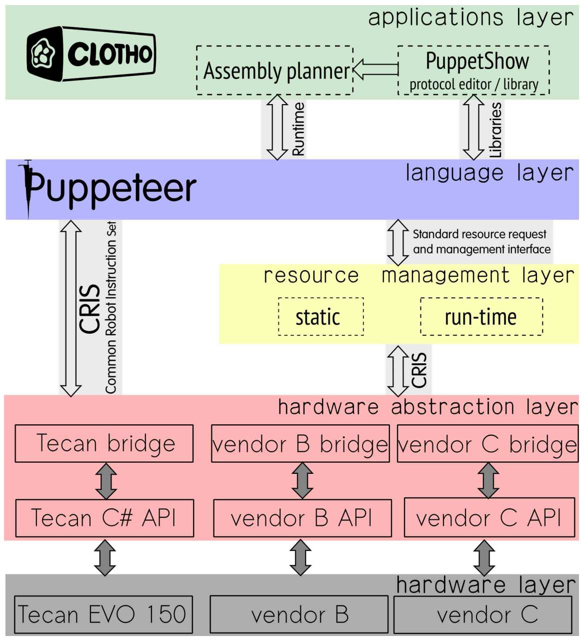


Figure 1: A software stack that abstracts away details of automated robotic assembly, enabling automated synthetic biological engineering on a variety of robotic platforms.

5. REFERENCES

[1] ANANTHANARAYANAN, V., AND THIES, W. Biocoder: A programming language for standardizing and automating biology protocols. *J. of Biological Engineering 4* (2010).

[2] ANDRIANANTOANDRO, E., BASU, S., KARIG, D. K., AND WEISS, R. Synthetic biology: new engineering rules for an emerging discipline. *Mol Syst Biol 2* (May 2006).

[3] BEAL, J., WEISS, R., DENSMORE, D., ADLER, A., BABB, J., BHATIA, S., DAVIDSOHN, N., HADDOCK, T., YAMAN, F., SCHANTZ, R., AND LOYALL, J. TASBE: A toolchain to accelerate synthetic biological engineering. In *IWBDA* (June 2011). (submitted).

[4] DENSMORE, D., DEVENDER, A. V., JOHNSON, M., AND SRITANYARATANA, N. A platform-based design environment for synthetic biological systems. In *TAPIA '09* (2009), ACM, pp. 24–29.

[5] DENSMORE, D., HSIAU, T. H. C., BATTEN, C., KITTLESON, J. T., AND DELOACHE, W. Algorithms for automated DNA assembly. *Nucleic Acids Research* (2010).

[6] ELLISON, M., RIDGWAY, D., FEDOR, J., GARSIDE, E., ROBINSON, K., AND LLOYD, D. BioBytes assembly standard. Tech. Rep. BBF RFC 47, The BioBricks Foundation, 2009.

[7] KNIGHT, T. Idempotent vector design for standard assembly of BioBricks. Tech. rep., MIT Synthetic Biology Working Group Technical Reports, 2003.

[8] YAMAN, F., BHATIA, S., ADLER, A., DENSMORE, D., BEAL, J., WEISS, R., AND DAVIDSOHN, N. Toward automated selection of parts for genetic regulatory networks. In *IWBDA* (June 2011). (submitted).

Monday – June 6th

4:30pm - 6:00pm: Tech. Talks Session 3 - *Biological Circuit Design*
3BDA.1 **Programmed DNA computing Circuits as Cell-Mimicking systems for Nanomedical Applications** Johann Elbaz, Fuan Wang and Itamar Willner
3BDA.2 **Automated Sequence Design for Nucleic Acid Circuits and Nanostructures using Structural Annotations** Benjamin Braun, Xi Chen and Andrew Ellington
3BDA.3 **Robust Design of Genetic Circuits through Population Wide Error Detection and Correction** Cristian Grecu, Jonathan Babb and Ron Weiss

Programmed DNA computing Circuits as Cell-Mimiking systems and Nanomedical Applications

Johann Elbaz, Fuan Wang and Itamar Willner
Institute of Chemistry, The Hebrew University of Jerusalem, Jerusalem 91904, Israel.
+972-2-6987737
e-mail: joelbaz@gmail.com

General Terms
Experimentation

Keywords
DNA, Programming devices, Biocomputing, Chemical recognition, Nanomedicine

Introduction

Nature performs complex information processing circuits, such as the programmed transformation of versatile stem cells into targeted functional cells. Man-made molecular circuits are, however, unable to mimic such sophisticated bio-machineries. Using the information encoded in the base sequence of DNA, and implementing catalytic functions of nucleic acids we develop programmable computing circuits that mimic cellular information processing pathways that find implications in future autonomous logic-nano-medecine.

Discussion

In a preliminary study [1], we reported on the use of catalytic nucleic acids (DNAzymes) that acted as functional elements for the design of logic circuits. We reported on the assembly of a library of Mg^{2+}-dependent DNAzyme subunits (box I) and theirs substrate (box II) that in the presence of the appropriate nucleic acids input(s) yield a functional nanostructure, computation unit, Figure 1. This computation unit included an input module, processing module and output module. The respective logic operation leads to the cleavage of the DNAzyme substrate and to fluorescence as the gate readout signal (box IV). Using this functional construct, universal set of logic gate was achieved and the use of this system for logic circuitry (gate cascade and fan-out gate cascade) was demonstrated. The logic construct was used to activate the release of the anti-thrombin aptamer, and to inhibit the hydrolytic activity of thrombin.

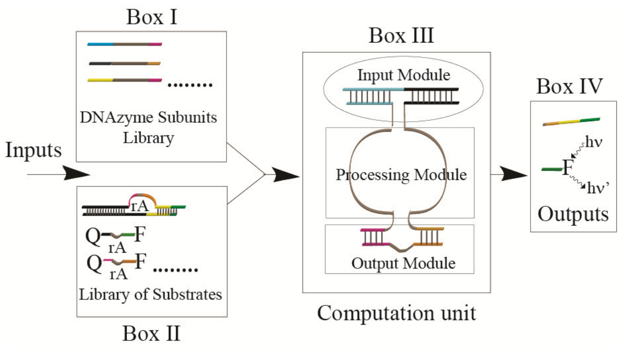
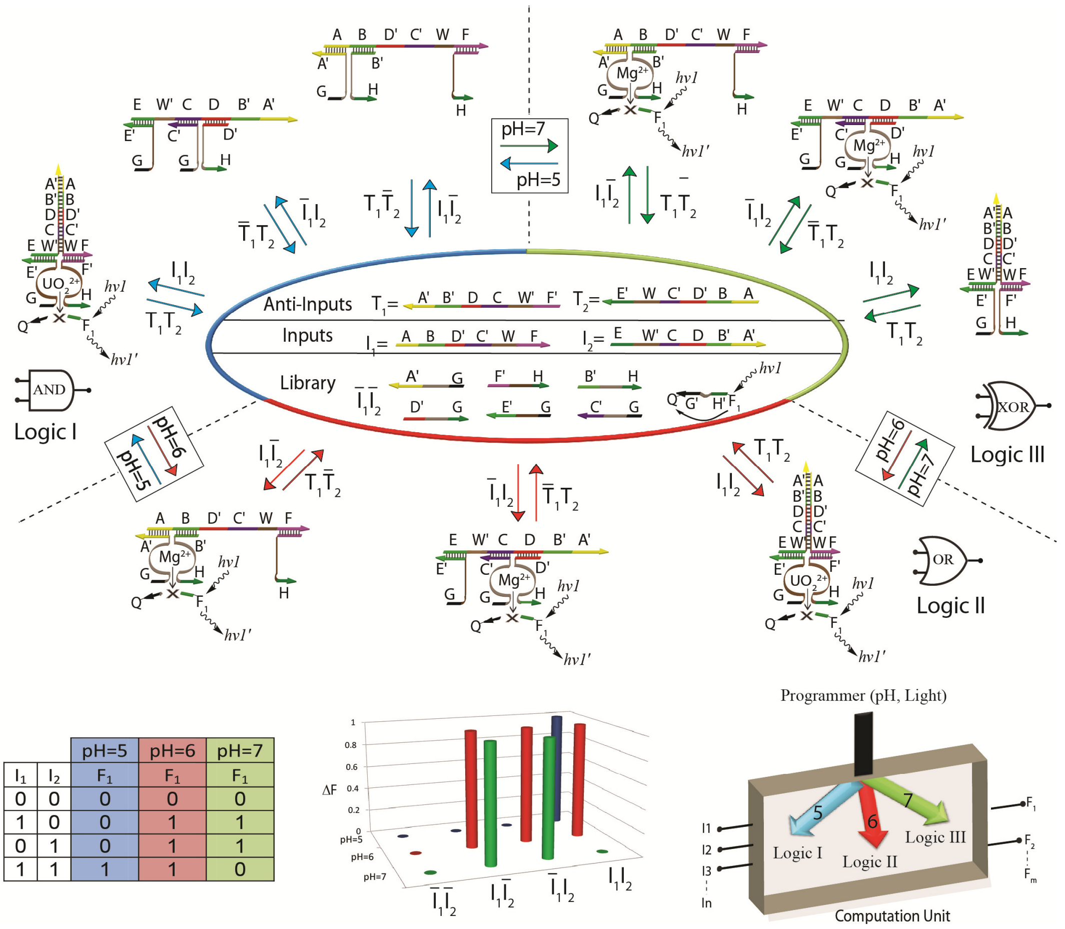


Figure 1- General design of the computation unit using libriries of DNAzyme and their substrates.

Complex information processing requires, however, the programming of the computational circuits by environmental triggered. The programming of the logic circuits was achieved by designing libraries of two different DNAzymes subunits (the Mg^{2+}- and the UO_2^{2+}-dependent DNAzymes), where the programming of the logic circuits was controlled by altering the pH of the system. While the UO_2^{2+}-dependent DNAzymes



		pH=5	pH=6	pH=7
I_1	I_2	F_1	F_1	F_1
0	0	0	0	0
1	0	0	1	1
0	1	0	1	1
1	1	1	1	0

Figure 2- DNA computing gates programmed by an environment stimuli (pH). By programming the system to pH=5, 6 or 7, an AND, OR, XOR gates are activated, respectively.

operates at pH=5.2, the Mg^{2+}-dependent DNAzyme operates at pH=7.2, where at pH=6 both DNAzymes exhibit partial activity. Thus, by the selection of the different subunits from the library three different logic circuits can be programmed, Figure 2. Using this method the programming of logic circuits of various complexities such as fan-out gate cascade, half-adder and half substractor were demonstrated. Since cancer cells exhibit low pH as compared to normal cells, the pH-programmable logic destruction of targeted m-RNA by the DNAzyme, may find future applications for cancer cell therapeutic nanomedicine.

REFERENCES

[1] Elbaz, J. et al. 2010. DNA computing circuits using libraries of DNAzyme subunits. *Nature Nanotechnol.* 5 (June 2010), 417-422.

I prefer to deliver the results in form of an oral presentation, yet a decision to present results as a poster is, also, acceptable.

Automated Sequence Design for Nucleic Acid Circuits and Nanostructures using Structural Annotations

Ben Braun Xi Chen Andrew Ellington
bjmnbraun@mail.utexas.edu, xichen@mail.utexas.edu, andy.ellington@mail.utexas.edu
The Center for Systems & Synthetic Biology
The University of Texas
Austin, TX 78712

ABSTRACT

The success of designed DNA-based chemical networks and nanodevices shows that DNA is as programmable as it is robust. Continued success on even more complex DNA-based systems relies on the design of primary sequence assignments which sufficiently promote desired hybridizations while preventing undesired hybridizations. Although a great deal of research has gone into this topic, DNA sequence design is still not standardized, partially because different systems have different requirements.

In this work, we present (1) a novel, structure-preserving representation of nucleic acid components at the domain-level abstraction in text format, (2) an improved set of quantified objectives for designing optimized sequences for general nucleic acid systems such as dynamic DNA circuits and DNA origami, and (3) a genetic algorithm to design sequences that approach these quantified objectives. We integrate these efforts in an automatic sequence design tool, CircDesigNA (`http://cssb.utexas.edu/circdesigna`), which interprets DNA circuits from a domain-based specification and produces a set of sequence-specified DNA parts which can be immediately synthesized and deployed. The implication of a program such as CircDesigNA is that DNA nanoscientists can develop design strategies which output a high level, structural, representation of a target system, and consider sequence design a black box. The development of modular DNA devices emerges as a consequence.

General Terms

Algorithms, Design, Standardization, Languages

Keywords

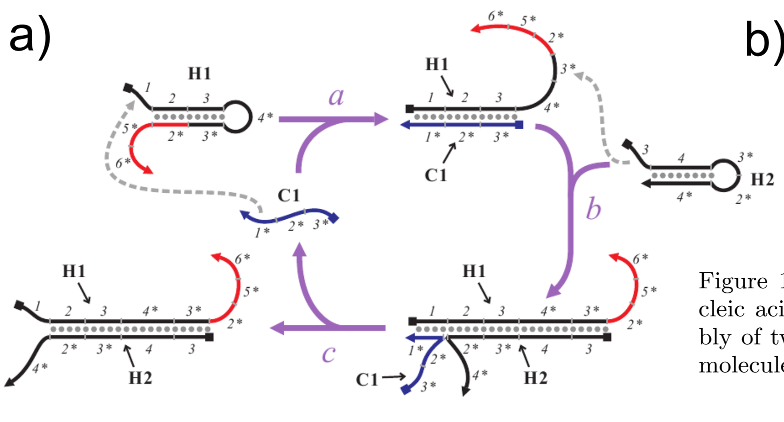
DNA chemical networks, DNA nanotechnology, Self-assembly

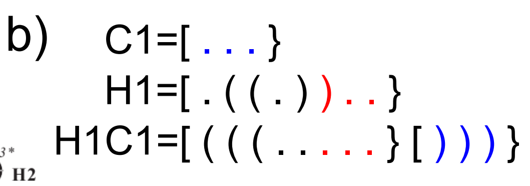
1. MOTIVATION AND INTRODUCTION

High level design of DNA-based chemical reaction networks, also known as DNA circuits [5][1], and nanodevices usually depends on the assumption that intended hybridization can be achieved by imposing sequence complementarity and that unwanted hybridization can be effectively avoided by careful sequence design. Sequence design for nucleic acid devices is not standardized, partially because of the sheer diversity of these devices, and many breakthroughs in DNA nanotechnology have been achieved without giving much focus to the sequence design problem. Standardizing sequence design would have the benefits of a more modular design process and more robust nanoscale devices, and could even lead to a better understanding of the kinetic models derived from nucleic acid interactions. Here, we aim to develop user-friendly, general purpose sequence design software which uses a structural representation of nucleic acid systems to evaluate sequence candidates using well-studied thermodynamic models.

2. A STRUCTURAL REPRESENTATION FOR NUCLEIC ACID SYSTEMS

We introduce a simple annotation system that preserves both sequence constraints and structural information, and is therefore well suited for representing general DNA circuits. This system currently only handles DNA components which are non-pseudoknotted. The structure of a nucleic acid molecule is expressed via an annotated version of the dot parenthesis model, where each character, a dot or a parenthesis, corresponds to a sequence domain (Fig. 1b). Multistranded unpseudoknotted complexes can be similarly represented [2].





```
C1=[ . . . }
H1=[ . ( ( . ) ) . . }
H1C1=[ ( ( ( . . . . . } [ ) ) ) }
```

Figure 1. A standard representation format for nucleic acid systems. This example depicts the assembly of two DNA molecules, H1 and H2, catalyzed by molecule C1.

This representation format preserves the structural information necessary for sequence design, but may also serve as a standard input format for the simulation of dynamic DNA hybridization reactions and as a standard output format for high level nucleic acid systems designers.

In the first step of designing a nucleic acid system with CircDesigNA, the user lists all domains and species which exist in the system's reaction pathway. For example, the circuit depicted in Fig. 1a involves domains 1 through 6 and species **H1, H2, H1C1, H1H2C2,** and **H1H2**. When defining a new domain, the user can choose to specify only its length or to provide a set of descriptive sequence constraints. The circuit depicted in Fig. 1a has been represented in this manner in Table 2 (see the online help file of CircDesigNA for a description of all possible sequence constraints).

```
Domain Definitions

1 8 -seq(G+C,40%,60%)   ->Composition Constraints
2 [SNNNNNNS]            ->Sequence Constraints
3 [SNNNNNNS]              Using IUPAC nomenclature
4 TTGTACCCACC           ->Immutable Bases
5 8 -seq(G+C,40%,60%)
6 8 -seq(G+C,40%,60%)

Molecules

H1   [.1|(2|(3|.4*|)3*|)2*|.5*|.6*}
H2   [.3|(4|.3*|.2*|)4*}
C1   [.3*|.2*|.1*}
H1C1 [(1|(2|(3|.4*|.3*|.2*|.5*|.6*}[)3*|)2*|)1*}
H1H2C1 [(1|(2|(3|(4*|(3*|.2*|.5*|.6*}
     [)3|)4|)3*|)2*|.4*}[.3*|.2*|)1*}
H1H2 [.1|(2|(3|(4*|(3*|.2*|.5*|.6*}
     [)3|)4|)3*|)2*|4*}
```

Table 2: The System in Fig. 1a converted to the standard textual input representation for CircDesigNA.

3. SCORING SEQUENCE DESIGN CANDIDATES

The net objective score of a sequence assignment ϕ is a function measuring the degree to which branch migration, strand association, and strand disassociation can occur as specified in a nucleic acid system. We propose the following framework for such a objective function, with justification of each term following. It is minimized during the process of sequence design. For a positive weight w,

$$\begin{aligned} Score(\phi) \quad = \quad & (\text{Net } -\Delta G_{MFE} \text{ of Cross Interactions}) + \\ & (\text{\# of Domain Helixes that Breathe}) * w + \\ & (\text{Net } -\Delta G_{MFE} \text{ of Undesired Self-Foldings}) \end{aligned}$$

, where a free energy of 0 is assigned to the ideal conformation where no unwanted base pairing occurs. These objectives were inspired by the work of David Yu Zhang, and a similar score function is used in his automatic designer, Domain Design [7]. *Term 1*: Branch migration is highly dependent on the structure of a toehold region for kinetics, hence spurious hybridization at a toehold region must be avoided when branch migration is desired. *Term 2*: The breathing of DNA helices can create a temporary toehold, which can result in undesired branch migration. Hence, the ends of helices should be prevented from breathing. *Term 3*: Strand association is inhibited if the target pairing domain is already engaged in some stable structure, either with itself or surrounding domains.

4. SEQUENCE DESIGN USING STRUCTURAL ANNOTATIONS

CircDesigNA uses a genetic algorithm to optimize sequences within the imposed constraints and minimize the *Score* function. Evaluating the *Score* function requires locating the undesired interactions of a nucleic acid system, and then quantifying each interaction using well-developed thermodynamic models. In order to enumerate undesired interactions, regions among all reaction species that exist in a single stranded conformation are identified using a stack-based algorithm. Loops and internal bulges are considered as single stranded regions, and two single stranded regions joined by a duplex are converted to one long single stranded region. These regions are then combined with the set of individual domains to form a set of 'generalized exposed regions'.

Pairs of generalized exposed regions are then classified as either *interactivity-desired* or *interactivity-undesired*, based on whether the two regions in each pair contain a domain and its complement, respectively.

Undesired intermolecular interactions are penalized with a free energy value corresponding to the intermolecular association penalty, $\Delta G_{Assoc.}$. Undesired self-foldings are similarly penalized. Unwanted base pairs that exist as a continuation of a helix in one of the reaction species are given an unusually high penalty corresponding to their base stacking energy and the lack of any entropic reward. Bases that participate in unwanted base pairs are more readily mutated during the design process of CircDesigNA. Sequence constraints can also help to ensure that desired interactions are maintained. Hence, CircDesigNA produces sequences which prevent cross interactivity, but retain desired interactions.

We have used CircDesigNA to design the sequences for the catalyzed hairpin assembly reaction presented in [6] and have experimentally verified some designs. Moreover, CircDesigNA can be efficiently applied to large or complex systems, such as designing a DNA origami [4] scaffold of length 7.2 kb . Our thermodynamic analysis showed that this designed scaffold results in less spurious interactions during origami folding than genomic M13 DNA. CircDesigNA can also be used in the design of structure-free extender sequences for aptamer design [3].

5. REFERENCES

[1] L. Cardelli and A. Phillips. A programming language for composable dna circuits. *J. R. Soc. Interface*, 2009.

[2] R. M. Dirks, J. S. Bois, et al. Thermodynamic analysis of interacting nucleic acid strands. *SIAM Rev*, 2007.

[3] L. N, J. Ebright, et al. Technical and biological issues relevant to cell typing with aptamers. *J Proteome Res*, 2009.

[4] P. W. K. Rothemund. Folding dna to create nanoscale shapes and patterns. *Nature*, 2006.

[5] D. Soloveichik, G. Seelig, and E. Winfree. Dna as a universal substrate for chemical kinetics. *Proc Natl Acad Sci U S A*, 2010.

[6] P. Yin, H. Choi, C. Calvert, and N. Pierce. Programming biomolecular self-assembly pathways. *Nature*, 2008.

[7] D. Y. Zhang. Towards domain-based sequence design for dna strand displacement reactions. *LNCS*, 2011.

Robust Design of Genetic Circuits through Population-Wide Error Detection and Correction

Cristian Grecu
Deparment of Electrical Engineering and Computer Science
MIT
Cambridge, USA
cgrecu@mit.edu

Jonathan Babb
Deparment of Electrical Engineering and Computer Science
MIT
Cambridge, USA
jbabb@mit.edu

Ron Weiss
Deparment of Electrical Engineering and Computer Science
MIT
Cambridge, USA
rweiss@mit.edu

1. INTRODUCTION

It is natural to inquire whether biological systems can inspire the design of their man-made, silicon counterparts. All organisms, either uni- or multi-cellular, use networking concepts at the very core of their existence: transcription/translation networks govern the mechanics of protein production and regulation; metabolic networks control their biochemical processes; biological neural networks support the flow of information in higher level organisms. Conversely, when designing synthetic biological systems [1], bioengineers look for inspiration to electrical engineering for addressing issues like composability, reusability, robustness, etc. As synthetic bio-circuits become more complex, networking concepts can be used to ease their design and improve their predictability. Fueled by similar needs for predictable performance, large scale integration, and improved robustness, synthetic biology and digital systems design can develop synergies that benefit each other in unexpected ways. The challenges that biological substrates impose on systematic, structured design of biological networks are not easy to tackle. For instance, the use of multiple identical components is generally restricted or even prohibited due to recombination; isolation is difficult or impossible to achieve in many cases; noise and crosstalk are inherent to the nature of biological substrates; failure modes of engineered biocircuits are not sufficiently understood.

Here, we demonstrate a network-inspired solution for improving the robustness of synthetic biological circuits using global signaling and population-wide voting mechanisms. We use the genetic toggle switch [2] as a model error generator for developing a voting mechanism coupled to the states of the toggle. A global communication signal synchronizes the whole cell population operating in a fault-free regime. When individual cells deviate from the synchronized behavior, an additional genetic circuit determines the loss of resistance to a particular antibiotic; by applying the antibiotic periodically to the system, faulty cells are pruned out and the correct behavior is preserved.

2. SIGNALING

In their natural environment, biological cells act as populations of individuals, with specific communication functions. Among such mechanisms we can enumerate small molecule signaling, protein transfer, and DNA transfer. In this work, communication channels are implemented using a particular class of small molecules called acyl-homoseryne lactone (AHL) produced enzymatically by the LuxI synthase. These small molecules can diffuse through the membrane of cells and bind to the receptor protein LuxR. The interaction between AHL and LuxR determines the latter to activate a specific promoter (pLux) and initiate transcription of the subsequent DNA region. In effect, this is a uni-directional communication channel, with AHL acting as a global communication signal [3].

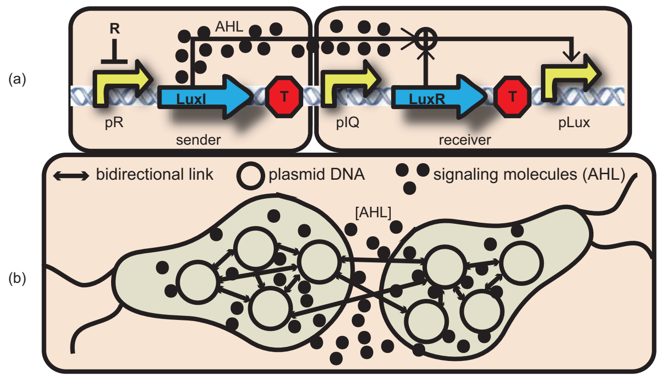


Figure 1. Global cellular communication based on small molecule (AHL) signaling. a: sender and receiver circuits; b. Inter-cellular signaling by AHL diffusion.

3. FAULT MODEL

We use a high-level fault model that is agnostic of detailed failure mechanisms that cause the genetic circuit to malfunction. The fault model employed assumes that a faulty cell that transitions from a correct state to a faulty state simply stops producing the protein (or combination of proteins) that characterizes the state of the circuit. This model was developed by performing time-lapsed flow-cytometry on a population of E. Coli cells that contained the plasmid with the toggle circuit. For instance, in Figure 2, after applying an initial pulse of IPTG, the cell population transitions from a fault free state in which the majority of the cells are fluorescing green to an intermediate state in which a significant number of cells lose green fluorescence, and finally to a failed state in which most cells either do not fluoresce or exhibit red fluorescence.

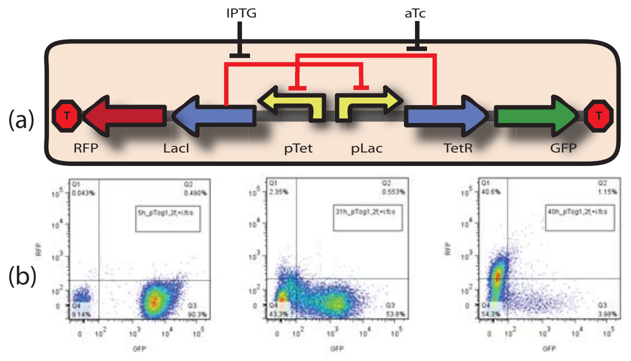
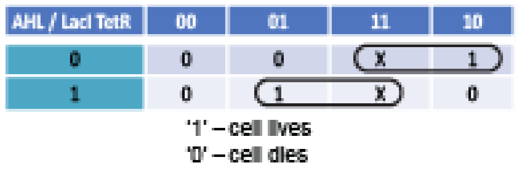


Figure 2. a. Genetic toggle switch; b. loss of function by fluorescence shift from Green to Red.

The advantage of using high-level fault models is that it can cover lower level faults of either phenotypic or genotypic nature. We hypothesize that by stopping the proliferation of cells that transition more rapidly toward the “faulty” state and encouraging the replication of cells that hold to the intended state, the circuit can be maintained in a “fault-free” state for a significantly longer amount of time, effectively error-correcting the original circuit.



AHL / LacI TetR	00	01	11	10
0	0	0	X	1
1	0	1	X	0

‘1’ – cell lives
‘0’ – cell dies

Figure 3. Design of error correcting genetic circuit using a high-level fault model. A binary value indicates that the corresponding protein species is either produced (‘0’) or not (‘1’).

The output of the error correcting circuit controls an antibiotic resistance gene that is expressed only when the cell operates correctly, in synchrony with the whole population. When the cell falls out of synchrony, the antibiotic resistance is shut off and the cell dies when the antibiotic is applied to the growth media.

4. PRELIMINARY RESULTS

We built DNA plasmids [4] that contain error detection and correction circuits designed as described above, with chloramphenicol as the antibiotic that kills faulty cells. Initial results show that our circuitry is able to confer high antibiotic resistance to cells that have a highly active pLux promoter (high cell density and correct state) and low resistance to low density populations (both correct and failed state). Also, the resistance of failed cells in high density populations is reduced. This indicates that it is indeed possible to synchronize the antibiotic resistance of the circuit with the correct behavior of the overall population by means of the AHL-based global communication mechanism.

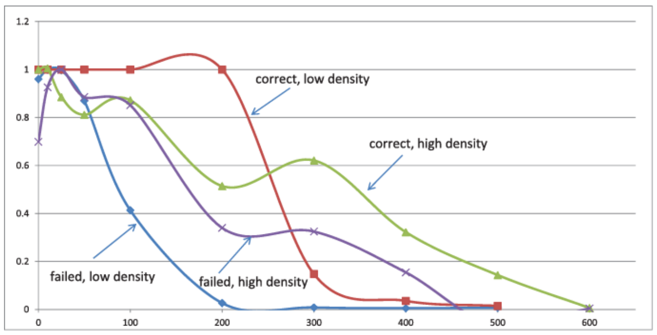


Figure 4. Antibiotic resistance vs. antibiotic concentration of cell populations in correct and failed states showing synchronization with cell density.

Future improvements include better separation between antibiotic resistance ranges of correct and failed cells within populations, and extension of the method to employ built-in suicide mechanisms that do not require external user intervention.

5. ACKNOWLEDGMENTS

This work is supported in part by NSERC and SynBERC.

6. REFERENCES

[1] Andrianantoandro, E., Basu, S., Karig, D. K., & Weiss, R. (2006). Synthetic biology: new engineering rules for an emerging discipline. *Molecular Systems Biology*, *2*(1), 2006.0028.

[2] Gardner, T. S., Cantor, C. R., & Collins, J. J. (2000). Construction of a genetic toggle switch in Escherichia coli. *Nature*, *403*(6767), 339-342. Nature Publishing Group.

[3] Ng, W.-L., Bassler, B. L. (2009). Bacterial quorum-sensing network architectures. *Annual Review of Genetics*, *43*(1), 197-222. Annual Reviews.

[4]] Ellis, T., Adie, T., & Baldwin, G. S. (2011). DNA assembly for synthetic biology: from parts to pathways and beyond. *Integrative biology quantitative biosciences from nano to macro*, 109-118. The Royal Society of Chemistry.