5th International Workshop on Bio-Design Automation

Imperial College, London

July 12-13, 2013

Platinum Sponsor

Gold Sponsor

Sponsor

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

Claudio Angione	University of Cambridge
Evan Appleton	Boston University
Aaron Berliner	NASA Ames
Federico Brunello	University of Bologna
Swati Carr	Boston University
Hui-Ju Katherine Chiang	National Taiwan University
Jole Costanza	University of Catania
Michal Galdzicki	University of Washington
Rishi Ganguly	Boston University
Owen Gilfellon	Newcastle University
Sonya Iverson	Boston University
Eric Kelsic	Harvard University
Faisal Khan	University of Oxford
Sungshic Park	Newcastle University
Pierre Parutto	EPITA (Ecole pour l'Informatique et les Techniques Avancées)
Ofir Raz	Weizmann Institute of Science
Nicholas Roehner	University of Utah
Jenhan Tao	Boston University

Foreword

Welcome to the Fourth International Workshop on Bio-Design Automation (IWBDA) at Imperial College, London. IWBDA 2013 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 and experimental development 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. 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 16 contributed talks and 16 poster presentations. Talks are organized into six sessions: Engineering, Parts, and Standardization; Characterization and System Identification; CAD Tools for Synthetic Biology; Biological Circuit Design and Assembly; and Applications. In addition, we are very pleased to have three distinguished invited speakers: Domitilla Del Vecchio, Alfonso Jaramillo, Richard Kitney, Jeff Hasty, and Howard Salis. Finally, we have an industrial panel session.

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 National Science Foundation, Synthetic Biology Engineering Research Center, CyBio, Agilent Technologies, Autodesk, ACS Synthetic Biology, Minres Technologies, Hudson Robotics, Raytheon BBN Technologies, and Benchling for their support.

Organizing Committee

Executive Committee

General Chair - Jonathan Babb (MIT)
Program Committee Chairs - Leonidas Bleris (UTDallas) and Ilias Tagkopoulos (UC Davis)
Publication Chair – Michal Galdzicki (University of Washington)
Industry Liaison Chair - Jose Pacheco (MIT)
Finance Chair - Aaron Adler (BBN Technologies) and Traci Haddock (Boston University)

Steering Committee

Douglas Densmore (Boston University)
Soha Hassoun (Tufts University)
Natasa Miskov-Zivanov (University of Pittsburgh)
Marc Riedel (University of Minnesota)
Ron Weiss (MIT)

Program Committee

Aaron Adler, BBN Technologies
J. Christopher Anderson, UC Berkeley
Shota Atsumi, UC Davis
Jonathan Babb, MIT
Jacob Beal, BBN Technologies
Leonidas Bleris, UT Dallas
Kevin Clancy, Life Technologies
Douglas Densmore, Boston University
Michal Galdzicki, University of Washington
Abhishek Garg, Harvard University
Traci Haddock, Boston University
Soha Hassoun, Tufts University
Dominique Loque, Joing BioEnergy Institute
Lan Ma, UT Dallas
Natasa Miskov-Zivanov, Carnegie Mellon University
Chris Myers, University of Utah
Kevin Oishi, University of Washington
Jose Pacheco, MIT
Dimitris Papamichail, University of Miami
Andrew Phillips, Microsoft Research
Cesar Rodriguez, Autodesk
Howard Salis, Penn State University
Casim Sarkar, University of Pennsylvania
Darko Stefanovic, University of New Mexico
Ilias Tagkopoulos, UC Davis

Additional Reviewers: B.A. Bartley, Rishi Ganguly, Alireza Goudarzi, Linh Huynh

IWBDA 2013 Program

Friday, July 12th

8:00 - 9:00 Registration
9:00 - 9:15 Opening Remarks

Keynote

9:15 - 10:15 **Domitilla Del Vecchio**. From Context-dependence to Modularity in Biological Circuits

10:15 - 10:30 Coffee Break

Talk Session 1: Engineering, Parts, and Standardization

10:30 - 10:55 (17) Michal Galdzicki, Ernst Oberortner, Mathew Pocock, Jacqueline Quinn, Mandy Wilson, Evan Appleton, Bryan Bartley, Jacob Beal, Swapnil Bhatia, Robert Cox, Raik Grunberg, Goskel Misirli, Hector Plahar, Nicholas Roehner, Larisa Soldotova, Guy-Bart Stan, Doug Densmore, **Chris Myers**, Herbert Sauro and Anil Wipat.
Recent Advances in the Synthetic Biology Open Language

10:55 - 11:20 (30) **Jacqueline Quinn**, Jacob Beal, Swapnil Bhatia, Patrick Cai, Joanna Chen, Kevin Clancy, Robert Sidney Cox, Michal Galdzicki, Nathan Hillson, Akshay Maheshwari, Chris Myers, Umesh P, Matthew Pocock, Cesar Rodriguez, Herbert Sauro, Larisa Soldatova, Guy-Bart Stan, Mandy Wilson, Drew Endy.
Synthetic Biology Open Language Visual: An Open-Source Graphical Notation for Synthetic Biology

11:20 - 11:45 (28) **Michal Galdzicki**, Bryan A. Bartley, Sean C. Sleight, Evren Sirin and John Gennari.
Version Control for Synthetic Biology

11:45 - 12:10 (33) **David Kong**, Jonathan Babb and Todd Thorsen, Scott Wick, Jeremy Gam, Ron Weiss and Peter Carr.
Microfluidic Genetic Circuit Assembly

12:10 - 13:30 Lunch

Keynote

13:30 - 14:30 **Alfonso Jaramillo**. Computational design of RNA circuits for signal processing in bacteria

Talk Session 2: Characterization and System Identification

14:30 - 14:55 (23) **Claudio Angione**, Jole Costanza, Giovanni Carapezza, Pietro Lio' and Giuseppe Nicosia.
Identifiability Analysis to Characterize Mitochondrial Diseases

14:55 - 15:20 (24) **Boyan Yordanov**, Neil Dalchau, Paul Grant, Jim Haseloff, Stephen Emmott and Andrew Phillips.
Automated Ratiometric Characterization using GEC

15:20 - 15:35 Coffee Break

Keynote

15:35 - 16:35 **Richard Kitney.** Foundational Resources in Synthetic Biology.

Industry Panel The Entrepreneurial Spirit in BioDesign Automation: from startups to spinouts to internal innovation.

16:35 - 17:35 Jacob Beal – BBN, Jaqueline Quinn – Autodesk, Wolf-Ekkehard Matzke – MINRES, Technologies, Debra Conway – Cybio, Malcom Wilson - Cronus Technologies on behalf of Hudson Robotics, Jim Hollenhorst – Agilent

18:30 - 20:00 Dinner

IWBDA 2013 Program

Saturday, July 13th

8:30 - 9:00 Registration

Keynote

9:00 - 10:00 **Jeff Hasty**. Engineered Gene Circuits: From Oscillators to Synchronized Clocks and Biopixels

10:00 - 10:15 Coffee Break

Talk Session 3: CAD Tools for Synthetic Biology

10:15 - 10:40 (19) **Evan Appleton**, Jenhan Tao and Douglas Densmore.
Interactive Experimentally Driven Algorithms for Optimized DNA Assembly

10:40 - 11:05 (16) Linh Huynh and **Ilias Tagkopoulos**.
Optimizing module matching for synthetic gene circuit design automation

11:05 - 11:30 (21) **Nicholas Roehner** and Chris Myers.
Directed Acyclic Graph-Based Technology Mapping of Genetic Circuit Models

11:30 - 11:55 (13) Jonathan Blakes, **Ofir Raz**, Natalio Krasnogor and Ehud Shapiro.
A New Algorithm for Combinatorial DNA Library Assembly

11:55 - 13:30 Lunch and Poster session

Talk Session 4: Biological Circuit Design and Assembly

13:30 - 13:55 (26) Noah Davidsohn, **Jacob Beal**, Aaron Adler, Fusun Yaman, Yinqing Li, Zhen Xie and Ron Weiss.
Accurate Predictions of Genetic Circuit Behavior from Part Characterization and Modular Composition

13:55 - 14:20 (12) Maximilian Weitz, Jongmin Kim, Korbinian Kapsner, Erik Winfree, **Elisa Franco** and Friedrich Simmel.
Dynamical diversity of a programmable molecular oscillator in droplet microreactors

14:20 - 14:45 (27) **Michael Guinn**, Yi Li and Leonidas Bleris.
A 2-input Decoder Circuit in Human Cells

14:45 - 15:00 Coffee Break

Keynote

15:00 - 16:00 **Howard Salis**. Clone Less, Know More: Efficient Optimization of Circuits and Pathways with Next Generation Biophysical Modeling

Talk Session 5: Applications

16:00 - 16:25 (14) **Wolf-Ekkehard Matzke**, Michael Stumpf and Thorsten Mascher.
Towards Industrial Strength BDA - A Short Essay

16:25 - 16:50 (4) **Pablo Carbonell**, Pierre Parutto and Jean-Loup Faulon.
RetroPath: Automated Pipeline for Embedded Metabolic Circuits.

16:50 - 17:15 (32) **Jole Costanza**, Claudio Angione, Giovanni Carapezza, Pietro Lio' and Giuseppe Nicosia.
Multi-objective Optimization of Escherichia coli for Direct Production of 1,4-butanediol

Closing remarks

17:15 - 17:30

Keynote Presentations

Domitilla Del Vecchio

From Context-dependence to Modularity in Biological Circuits

The ability to accurately predict a network's behavior from that of the composing modules is a major challenge in systems and synthetic biology due to individual modules exhibiting context-dependent behavior. One leading cause of context-dependence is retroactivity, a phenomenon similar to loading that affects the dynamic performance of a module upon connection to other modules. Retroactivity is particularly daunting for synthetic biology, in which working modules often fail to function as predicted once interacting with each other. Here, we illustrate analysis and design techniques to mitigate this problem along with experimental demonstrations. First, we introduce a simple analysis framework, conceptually analogous to Thevenin's electrical circuit theory, which accounts for retroactivity to quantitatively predict how a module's behavior will change after interconnection with other systems. To solve this problem, we introduce a load driver device to connect a circuit to a variable number of systems (load) in such a way that the circuit output is reliably transmitted to the load while preserving the unloaded system performance. The load driver design is based on fast regulatory elements that reach quasi-steady state quickly in response to slow changing inputs (principle of time-scale separation). These fast elements are in sufficiently high concentrations such that the quasi-steady state is unaffected by load. We built and experimentally tested in Saccharomyces cerevisae an instance of a load driver with circuitry based on (fast) phosphotransfer reactions. We first built and tested several transcriptional regulatory networks and demonstrated that their responses were severely impaired due to load. We then demonstrated that incorporation of our load driver restored response dynamics, thus remedying the load problem.

Keynote Presentations

Alfonso Jaramillo

Computational design of RNA circuits for signal processing in bacteria

The de novo engineering of living systems with targeted behaviour is one of the grand challenges of modern science. Such large-scale engineering requires high-throughput design methodologies to create novel cellular circuitry through the assembly of synthetic genetic elements. We use evolutionary computation algorithms with standard secondary structure methods to engineer RNA circuits working in E. coli. The advantage of RNA systems is that they are smaller than traditional protein circuits and more programmable with rational design techniques. We will assemble synthetic riboregulatory devices with ribozymes and aptamers to create RNA circuits for signal processing in bacteria. We have obtained RNA circuits for small-molecule riboregulator sensing and riboregulatory cascades, and have characterised their dynamics under varying environments using microfluidics time-lapse microscopy. Such circuits could be used as modular units to process endogenous or exogenous signals. Our work opens the door to the engineering of integrated circuits in living cells consisting of small DNA fragments with multifunctional behaviour.

Keynote Presentations

Richard Kitney

Foundational Resources in Synthetic Biology

The presentation will first consider some of the background issues in relation to synthetic biology and why the field has developed. Also, what is new about synthetic biology. Our approach to synthetic biology is via systematic design incorporating the engineering tenents of modularisation, standardisation and characterisation. Other important design issues are controlling complexity and responsible innovation. Our systematic design approach is based upon the synthetic biology design cycle. Our fundamental strategic approach comprises the development of platform or foundational technology leading through to a number of application areas. An important part of the work is the characterisation of bioparts. This is done in the context of the development of a web-based information system called SynBIS system. This system will be described. It comprises a four layer model: the HTML layer; the communication layer; the application layer; and the database layer. The system is now being extended to the device and systems levels. The process of part characterisation will be discussed in the context of the information system. An important part of the characterisation process is the production of standard data sheets which are incorporated into SynBIS. The next part of the talk will address standards, and particularly how standards relate to the industrialisation of synthetic biology. As part of a large European project we are developing a standard called DICOM-SB, this is a new Part of the international DICOM standard for biomedicine. DICOM-SB is compatible with SBOL. There will be a brief description of the development and aims of the UK Roadmap for Synthetic Biology and how latest development within the Centre for Synthetic Biology and Innovation are addressing the industrial translation process.

Keynote Presentations

Jeff Hasty

Engineered Gene Circuits: From Oscillators to Synchronized Clocks and Biopixels

Synthetic biology can be broadly parsed into the “top-down" synthesis of genomes and the “bottom-up" engineering of relatively small genetic circuits. In the genetic circuits arena, toggle switches and oscillators have progressed into triggers, counters and synchronized clocks. Sensors have arisen as a major focus in the context of biotechnology, while oscillators have provided insights into the basic-science functionality of cyclic regulatory processes. A common theme is the concurrent development of mathematical modeling that can be used for experimental design and characterization, as in physics and the engineering disciplines. In this talk, I will describe the development of genetic oscillators over increasingly longer length scales. I will first describe an engineered intracellular oscillator that is fast, robust, and persistent, with tunable oscillatory periods as fast as 13 minutes. Experiments show remarkable robustness and persistence of oscillations in the designed circuit; almost every cell exhibits large-amplitude fluorescence oscillations throughout each experiment. Computational modeling reveals that the key design principle for constructing a robust oscillator is a small time delay in the negative feedback loop, which can mechanistically arise from the cascade of cellular processes involved in forming a functional transcription factor. I will then describe an engineered network with intercellular coupling that is capable of generating synchronized oscillations in a growing population of cells. Microfluidic devices tailored for cellular populations at differing length scales are used to demonstrate collective synchronization properties along with spatiotemporal waves occurring on millimeter scales. While quorum sensing proves to be a promising design strategy for reducing variability through coordination across a cellular population, the length scales are limited by the diffusion time of the small molecule governing the intercellular communication. I will conclude with our recent progress in engineering the synchronization of thousands of oscillating colony “biopixels" over centimeter length scales through the use of redox signaling that is mediated by hydrogen peroxide vapor. We have used the redox communication to construct a frequency modulated biosensor by coupling the synchronized oscillators to the output of an arsenic sensitive promoter that modulates the frequency of colony-level oscillations due to quorum sensing.

Keynote Presentations

Howard Salis

Clone Less, Know More: Efficient Optimization of Circuits and Pathways with Next-Generation Biophysical Modeling

5000 ribosome binding sites fight for 30S, free energy minimized.

2000 promoters wield sigma factor against beta units, heat shock traumatized.

300 small RNAs find their partners and Hfq, kissing loops tantalized.

20 riboswitches morph their shape, ligands recognized.

One DNA Compiler to design them all, one DNA Compiler to find them,

And in the genome, where they belong, one DNA Compiler to predict what binds them.

:)

-Howard

Abstracts - Table of Contents

Oral Presentations

Abstracts - Table of Contents

Poster Presentations

Recent Advances in the Synthetic Biology Open Language

Michal Galdzicki Ernst Oberortner Matthew Pocock Jacqueline Quinn Mandy Wilson
Evan Appleton Bryan Bartley Jacob Beal Swapnil Bhatia Robert Cox Raik Grünberg
Goksel Misirli Hector Plahar Nicholas Roehner Larisa Soldotova Guy-Bart Stan
Doug Densmore Chris J. Myers Herbert Sauro Anil Wipat

1. INTRODUCTION

A significant concern in the synthetic biology community is the difficulty in reproducing results reported in the literature [5]. To address this problem, in 2008, a small group of researchers proposed the development of the *synthetic biology open language* (SBOL), an open-source standard for the exchange of genetic designs. In 2011, the first version of the SBOL core data model was released [2]. In 2013, the first version of a standard for visualization of genetic designs expressed in SBOL was also released [6]. Leveraging `libSBOLj`, a java-based library for SBOL's core data model, 18 software tools now support SBOL. While this represents excellent progress, there is still a lot of work to do. In April 2013, several members of the SBOL developers group met at Newcastle University to discuss SBOL's next steps. This abstract briefly describes two significant outcomes of this meeting: a plan to improve SBOL adoption and several significant enhancements to SBOL's data model.

2. SBOL ADOPTION PLAN

As mentioned above, SBOL evolved out of a desire to improve the reproducibility of research results. Figure 1 summarizes a plan discussed at the workshop that leverages SBOL to achieve this goal. We are working with the editors of the journal ACS Synthetic Biology to encourage authors of papers describing genetic designs to submit information about their design. The *Joint Bio Energy Institute* (JBEI) has created a repository called `JBEI-ICE` for storing information about genetic designs [3]. It has been proposed that designs be deposited at a public `JBEI-ICE` repository and that their EntryID be provided to the journal upon paper submission. This would be akin to submissions of DNA sequences to `GenBank` or `EMBL` but in this case authors submit designs. The submitted information would remain private, but reviewers of their paper would be given access in order to access the quality of the provided information. Upon paper acceptance, this submitted information would become publicly accessible. The `JBEI-ICE` tool supports export of SBOL RDF/XML files, so this would enable both the authors and other users to import these designs into their *genetic design automation* (GDA) tools. In the future, these GDA tools may be used directly to submit the designs to the repository. Another added benefit is the ability to export graphic design images in the SBOL visual standard using Graphviz or the software tool `Pigeon` [1]. We hope that these generated images are ultimately used in the publication of the design.

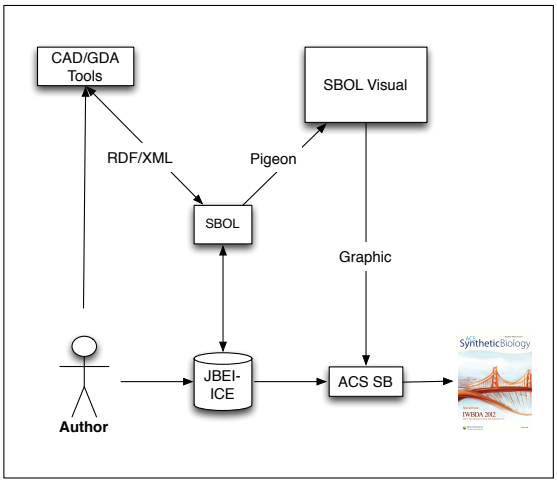


Figure 1: SBOL adoption plan.

3. ENHANCEMENTS TO SBOL

To improve the utility and adoption of SBOL, it is essential that it is able to capture all the relevant information for a genetic design. The first version of SBOL enables exchange of annotated DNA sequences. In particular, an SBOL file is composed of *DNA components* which have a *unique reference identifier* (URI), an id, a name, a description, a type taken from the Sequence Ontology, and a *DNA sequence*. These DNA components are annotated using *sequence annotations* which specify positions within a DNA sequence that are to be annotated with simpler DNA components.

There are several aspects of genetic design that cannot be exchanged using the first version of SBOL. At the workshop, several enhancements to the SBOL data model were proposed as shown in Figure 2 (note that most objects include URI, id, name, and description fields which are not shown to save space). First, the *component* class is generalized to allow it to not only represent annotated DNA sequences, but also annotated RNA and protein sequences. Next, the *device* class has been added to allow the description of groups of components that together perform a desired function. Devices can be either *primitive* (i.e., composed of components) or *composite* (i.e., composed of other devices). Next, the *system* class is added as a way to group a collection of devices with a description of their *context* under which they are to be used. The context class is used to provide information about the experimental methods including the

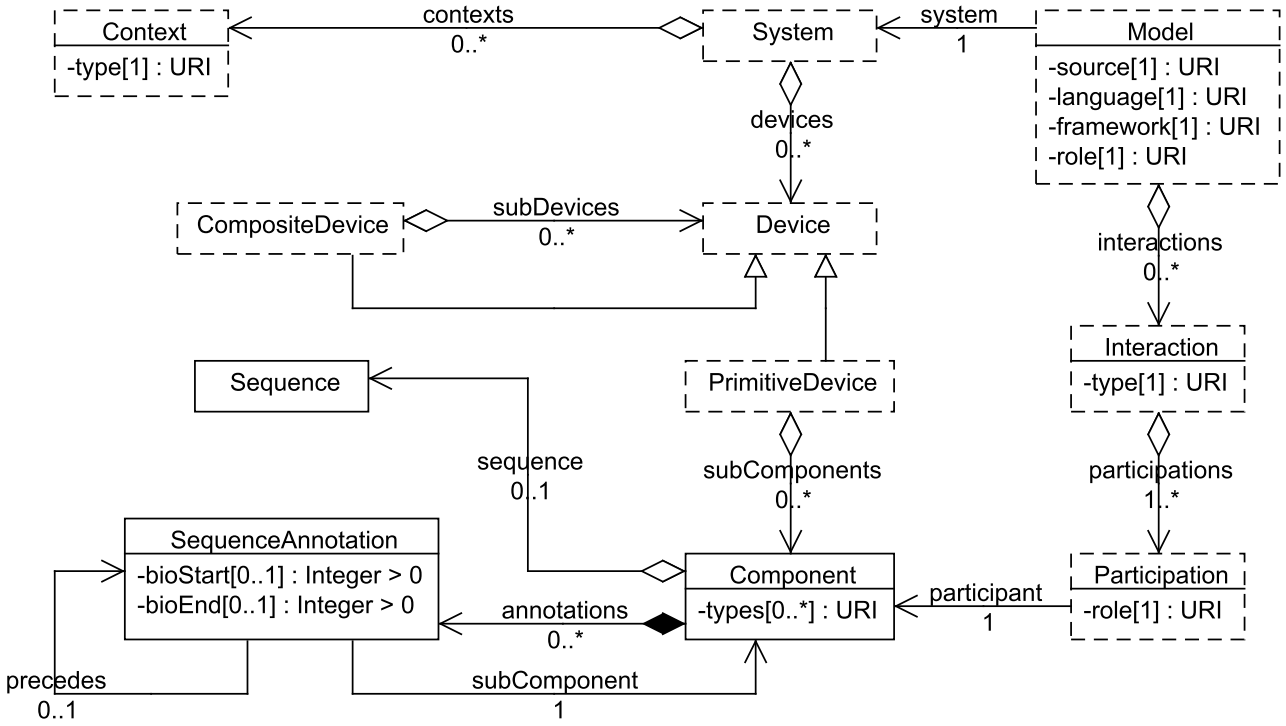


Figure 2: Proposed enhancements to SBOL's data model.

strain of the host, the medium in which the host resides, the container in which the medium is stored, the environmental conditions, and the measurement device used to study the system. Precise details about the experimental context are often essential to the reproducibility of laboratory results.

The final significant enhancement proposed is a mechanism for attaching models to genetic designs. There has been significant work in the development of standards for modeling biological systems, such as the *systems biology markup language* (SBML) [4]. The SBOL community does not wish to duplicate these efforts. Therefore, the *model* class allows one to reference a model for a system expressed in an external file using any modeling language of choice. The model class includes a URI to reference an external model file, a URI to identify the language of the model (SBML, CellML, matlab, BNGL, etc.), a URI to identify the modeling framework (ODE, stochastic, Boolean, etc.), and a URI to identify the purpose of the model (specification, back annotation, etc.). One important aspect of modeling for genetic designs is information about the interactions between the components in the design. For example, a protein component may repress a DNA component of type promoter. Typically, this information is stored in the model file, but it is useful for the user to have ready assess to the information, for example, to draw regulation constructs in a genetic circuit diagram. An *interaction* class is proposed to provide this type of information to the user. Each interaction includes a URI from the Systems Biology Ontology to specify its type (repression, activation, etc.). An interaction includes a number of component participants which are each given a URI for their role (repressor, repressed, etc.). While typically, these interactions are stored in an accompanying model file, if the source of the model is the SBOL file itself, then these interactions can be stored within the SBOL file.

4. CONCLUSION

While this abstract describes some exciting proposals being discussed in the SBOL community, it should be emphasized that they are just proposals, and we would value further input from people in the synthetic biology community. Many problems remain to be solved, and we look forward to further stimulating discussion at IWBDA.

5. ACKNOWLEDGMENTS

There are over 80 members of the SBOL developers group. While not all members were able to participate in the SBOL workshop, we would like to acknowledge their support of SBOL through numerous lengthy discussions on the SBOL mailing list.

6. REFERENCES

[1] S. Bhatia and D. Densmore. Pigeon: a design visualizer for synthetic biology. *ACS Syn. Bio.*, April 2013.

[2] M. Galdzicki et al. Synthetic biology open language (SBOL) version 1.0.0. BBF RFC #84.

[3] T. Ham et al. Design, implementation and practice of jbei-ice: an open source biological part registry platform and tools. *Nucleic Acids Res.*, 40(18), 2012.

[4] M. Hucka et al. The systems biology markup language (SBML): A medium for representation and exchange of biochemical network models. *Bioinfo.*, 9(4), 2003.

[5] J. Peccoud et al. Essential information for synthetic dna sequences. *Nature Biotechnology*, 29(22), 2011.

[6] J. Quinn et al. Synthetic biology open language visual (SBOL Visual) version 1.0.0. BBF RFC #93.

Synthetic Biology Open Language Visual: An Open-Source Graphical Notation for Synthetic Biology

Jacqueline Quinn, Jacob Beal, Swapnil Bhatia, Patrick Cai, Joanna Chen, Kevin Clancy, Robert Sidney Cox III, Michal Galdzicki, Nathan Hillson, Akshay Maheshwari, Chris Myers, Umesh P, Matthew Pocock, Cesar Rodriguez, Herbert M Sauro, Larisa Soldatova, Guy-Bart Stan, Mandy Wilson, Drew Endy

1. INTRODUCTION

The Synthetic Biology Open Language Visual (SBOL Visual) project is an effort toward developing a community-driven open standard for visual representation of genetic designs. Standardized visual notation for communicating designs has proven to be useful in many engineering disciplines. A de facto visual notation does exist in synthetic biology; however, it is incomplete, is often extended ad hoc, and exists as a poorly defined, voluntary, communal convention rather than an explicit standard. Because synthetic biology endeavors often require a multidisciplinary team, a common visual system of communication with well-defined semantics is vital. It is also important that the emerging ecosystem of biological design tools converge upon a common visual language to maximize adoption and minimize ambiguity in results. Given the central role and rich history of visual representation in the life sciences, a well-defined visual notation will also prompt the construction of the formal infrastructure needed to support effective ontologies, meaningful models, and tools tailored to community needs.[1]

promoter	primer binding site
cds	restriction site
ribosome entry site	blunt restriction site
terminator	5′ sticky restriction site
operator	3′ sticky restriction site
insulator	5′ overhang
ribonuclease site	3′ overhang
rna stability element	assembly scar
protease site	signature
protein stability element	user defined
origin of replication	

Figure 1: SBOL Visual version 1.0.0 symbols

[1] The introduction section was submitted as an abstract and will be presented as poster at SB6.0. Figures in this abstract overlap with figures in a paper submitted to the Bio-Ontologies 2013.

2. RELATIONSHIP TO SBOL STANDARD

SBOL Visual comprises a set of symbols, shown in figure 1, used to visually depict functional information encoded by nucleic acid sequences [1]. It serves as a human-readable counterpart to the Synthetic Biology Open Language data exchange standard [2]. The way in which SBOL Visual relates to the SBOL standard is depicted in Figure 2. Sequences of one or more SBOL Visual sequences can be used to depict DNA Components in the SBOL data model. Each symbol represents one or more ontological terms, which are used in the SBOL data model to establish DNA Component type, often referred to as part type. Currently, both SBOL Standard and SBOL Visual use terms from the Sequence Ontology (SO) [3].

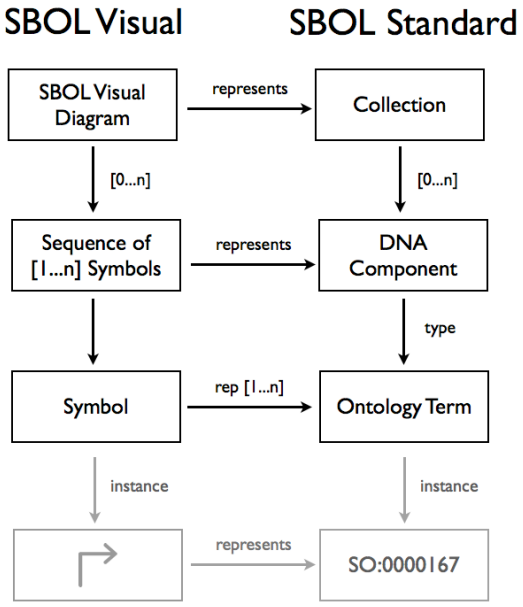


Figure 2: Relationship between SBOL Visual and SBOL Standard

In the specification of SBOL Visual version 1.0.0, each symbol is linked to a single SO term, which serves as a definition for the symbol. In April 2013, during a meeting of the SBOL Developers Group in Newcastle University, changes to the relationship between SBOL Visual symbols and ontological terms were explored. While use of a single ontological term to serve as a definition for a particular symbol allows for tighter design representation, allowing symbols represent of multiple terms widens the flexibility and range of the standard. As a proof of concept, terms in the Sequence Ontology Feature Annotation (SOFA), a subset of the SO containing only locatable sequence feature, were assigned to SBOL Visual symbols, displayed in Figure 3. Creating this mapping enables software tools to more

readily generate diagrams to depict annotated sequences by expanding the number of terms that can be represented by SBOL Visual. Further work is yet to be done to refine the mechanics for serialization of symbol-term mappings and how to avoid conflicts in potential interpretation of symbols.

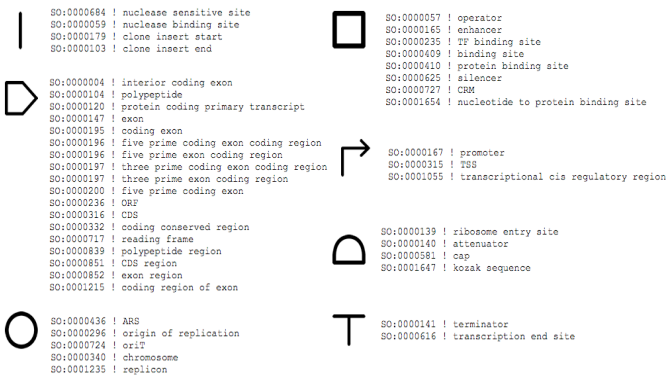


Figure 3: Draft mapping from SOFA to SBOL Visual symbols

3. ADOPTION OF SBOL VISUAL

SBOL Visual is intended for use in a range of circumstances and formalities, from casual whiteboard brainstorming to formal scientific publication. The standard is presently being used by synthetic biologists to depict genetic designs in peer-reviewed publications and presentations. Software developers in academia and industry are also adopting SBOL Visual in computer-aided design tools for synthetic biology. The tools that adopt SBOL Visual are listed in Table 1.

Table 1. Software tools that adopt SBOL Visual

Tool Name	Link
Tinker Cell	tinkercell.com
GenoCAD	genocad.org
DeviceEditor	j5.jbei.org
BioCompiler	synbiotools.bbn.com
SBOL Designer	clarkparsia.github.io/sbol
Pigeon	pigeoncad.org
GraphViz	graphviz.org
Vector NTI® Express Designer	lifetechnologies.com
Teselagen	teselagen.com

SBOL Visual can be used alongside other notation systems, such as Systems Biology Graphical Notation and with user generated symbols in order to depict aspects of design such as genetic regulation, as shown in Figure 4.

4. GROWTH OF SBOL VISUAL

The next step for SBOL Visual is the creation of a framework for supporting the synthetic biology community's involvement in growing and maturing SBOL Visual. While the symbols specified in the current version of SBOL Visual are drawn from the de facto visual notation of the synthetic biology community, SBOL Visual must ultimately be community driven if it is to meet the needs of synthetic biologists. The infrastructure for adding new symbols and term mappings to SBOL Visual is yet to be built.

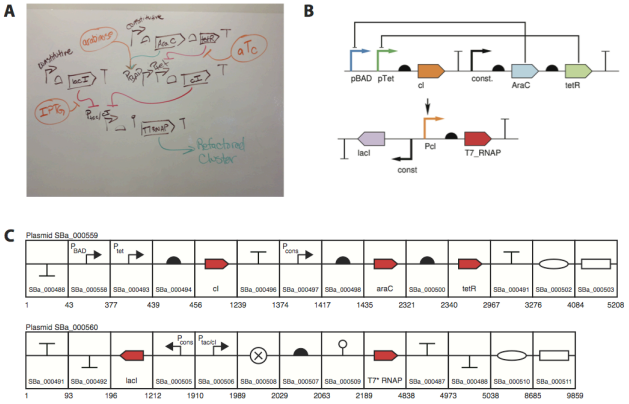


Figure 4: Use of SBOL Visual to represent an ANDN controller, designed by Temme, et al. [4] (A) as drawn on a whiteboard, (B) as produced by web-based SBOL Visual generator Pigeon [5], and (C) as formatted and by Temme for publication.

5. CONCLUSION

SBOL Visual is an open-source, community-driven response to the need for standardization of graphical notation for synthetic biology. It is a set of symbols to represent ontological terms related to functionality of nucleic acid sequences, focused on forward engineering of genetic systems. SBOL Visual is being used on whiteboards, slide presentations, scientific posters and publications, and in several CAD tools. The next steps for the SBOL Visual effort is to develop a consistent methodology for adding new symbols and ontological associations to the standard, so that SBOL Visual can leverage the input of the greater synthetic biology community and cover more use cases. Through active input from the synthetic biology community, SBOL Visual will mature into a foundational tool for the communication of genetic design.

6. AUTHOR AFFILIATIONS

Jacqueline Quinn, Autodesk, jacqueline.quinn@autodesk.com
Jacob Beal, BBN Technologies
Swapnil Bhatia, Boston University
Patrick Cai, University of Edinburgh
Joanna Chen, Joint BioEngergy Institute
Kevin Clancy, Life Technologies
Robert Cox, Technology Research Association of Highly Efficient Gene Design, Kobe
Michal Galdzicki, University of Washington
Nathan Hillson, Joint BioEnergy Institute
Akshay Maheshwari, University of California, San Diego
Chris Myers, University of Utah
Umesh P, University of Kerala
Matthew Pocock, Newcastle University
Cesar Rodriguez, Autodesk, Inc.
Herbert Sauro, University of Washington
Larisa Soldatova, Brunel University, London
Guy-Bart Stan, Imperial College London
Mandy Wilson, Virginia Bioinformatics Institute

Drew Endy, Stanford University

7. REFERENCES

[1] Quinn, J. et al. (2013). Synthetic Biology Open Language Visual (SBOL Visual), version 1.0.0. BBF RFC #93. doi:1721.1/78249

[2] Galdzicki, M. et al. Synthetic Biology Open Language (SBOL) version 1.1.0. BBF RFC #87. doi: 1721.1/73909

[3] Eilbeck, K., et al. (2005). The Sequence Ontology: a tool for the unification of genome annotations. *Genome biology*, *6*(5), R44. doi:10.1186/gb-2005-6-5-r44

[4] Temme, K. et al. (2012). Refactoring the nitrogen fixation gene cluster from Klebsiella oxytoca. *Proceedings of the National Academy of Sciences of the United States of America*, *109*(18), 7085–90. doi:10.1073/pnas.1120788109

[5] S. Bhatia and D. Densmore. Pigeon: a design visualizer for synthetic biology. ACS Syn. Bio., April 2013.

Version Control for Synthetic Biology

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

Bryan A. Bartley
Bioengineering,
University of Washington
3720 15th Ave NE Seattle, WA
bbartley@uw.edu

Sean C. Sleight
Bioengineering,
University of Washington
3720 15th Ave NE Seattle, WA
seansleight@gmail.com

Evren Sirin
Clark & Parsia, LLC
926 N St NW Rear #1
Washington, DC
evren@clarkparsia.com

John H. Gennari
Biomedical and Health Informatics,
University of Washington
1959 NE Pacific St Seattle, WA
gennari@uw.edu

ABSTRACT
In this paper, we describe a novel version control system for synthetic biological designs. For synthetic biology designs to scale up to larger projects, teams of engineers need to be able to coordinate their work. In software engineering, version control methodology is used to coordinate software programming across teams. We adapted this idea to the synthetic biology workflow. Through specific use cases and an initial implementation in a CAD design tool for synthetic biology called SBOL Designer, we demonstrate the management of changes to designs of genetic constructs. The application of version control as an engineering methodology in synthetic biology is critical to the development of design automation for biological systems.

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

General Terms
Management, Documentation, Design, Standardization.

Keywords
Synthetic biology, version control, standards.

1. INTRODUCTION
The use of version control in software engineering is pervasive and recognized as essential in practice. The most common use is for source code version control through a tool, which captures the changes to source code elements. The benefits of such systems are well accepted as they provide secure access to source and file history. Above all else, they facilitate team communication and baseline trace ability. Tools support automated merging of revisions and their use ensures no one over-writes someone else's code. In software project management these tools allow for better control and parallel development by team members. Additionally, such systems are also used more widely for management and organization of revisions to written documents and other files.

Synthetic biologists have benefited from the increasingly more efficient DNA synthesis technology and assembly techniques. Furthermore, new synthetic biology specific software tools used to design DNA constructs are in development. However, so far these tools do not take into account the repetition of the design, construction, and testing workflow. Tools that support the organization of results generated by the iterative engineering cycle are needed. For instance, the design evolves both as the ideas of the engineer develop and upon feedback from testing. Especially in biological engineering, the techniques for synthesizing, assembling, and deliberately introducing the DNA into host cells is costly and error prone. However, researchers may generate many variants of an original design to test multiple possible design solutions concurrently. Molecular biology techniques at the wet-bench are highly amenable to parallelization of the process to screen variants for the desired outcome. These techniques leave the engineer with many variant designs to organize and analyze. Therefore, the need to track, compare, and document design decisions is even more important than in software engineering, especially when coordinating the work of a team of researchers.

2. Use case: design revisions in building CMY
One of us (S.C.S) aimed to improve upon a previously engineered inducible multi-color reporter [4]. The improved design is composed of three expression cassettes, one for each fluorescent protein (Cyan - CFP, Magenta - mRFP, and Yellow - GFP). Each expression cassette was first built and tested independently and then combined into the final design. An analogous division of labor into feature *branches* is a common software engineering strategy to manage complexity and dependencies in the overall design. In Figure 1, each of these three modules is represented by a separate branch. Once each branch is sufficiently tested, it is *merged* into the final design, often known as the master branch.

Additionally, S.C.S tested two attempts at the “Cyan” color channel: first, the amilCP coding sequence, second the eCFP coding sequence. This is analogous to the case in software engineering where the programmer makes a *commit* after each change and saves the current state to the version control system. The versioning system records a snapshot of the source code, the time, and the author.

A design revision, such as use of eCFP, may be an improvement, but if it fails to perform effectively, the engineer should have the ability to rollback, or *revert*. Reverting to a previous version is common in synthetic biology. As another example at the high end of complexity, Bonnet et al report testing over 700 distinct constructs[1]. Thus if the subsequent changes turn out to be unwanted at a later time, the design can be *reverted* back to that commit point.

An important difference between synthetic biology designs and software engineering is the need to build the physical DNA and then introduce it into a cell. This process creates explicit commit points. However, there is a significantly higher cost of building in synthetic biology, compared to "compiling and building" software. This suggests that re-use of previously built constructs is highly valuable to the biological engineer. We observed that the biological engineer is very likely to re-use a module, such as a specific commit of a branch, in a new design. For example, S.C.S. did not begin the CMY project from scratch. He re-used components from his own prior work and from the publicly available DNA components in the Parts Registry (http://partsregistry.org). Additionally, his first Cyan attempt with the amilCP construct, may be useful in another design. This finding led us to include a *clone* operation, which imports the committed design as a new component in an independent design. Such a function is not present in traditional source code management software.

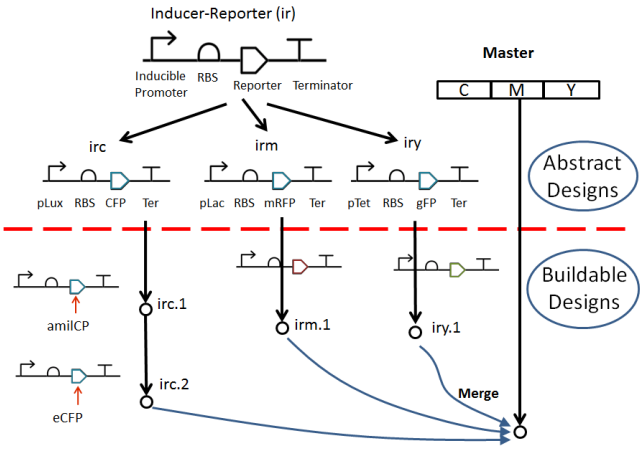


Figure 1. Details of the versioning of CMY, showing the distinction between abstract and buildable designs.

3. SBOL Designer

In order to demonstrate the use of version control in a synthetic biology application, we implemented the basic design revision operations within our CAD design tool for synthetic biology, the SBOL Designer. This tool enables the engineer to specify a new design using the SBOL Visual graphical symbols. Designs can be committed to and checked out from an online repository. Branch management features allow the engineer to create separate lines of design and to merge the resulting designs. As Figure 2 shows, we have implemented a display of the commit and branch history to view the progress of the project. The SBOL Designer uses the SBOL data standard [2]to store and retrieve the designs using Stardog (Clark & Parsia, LLC http://stardog.com/). Additionally, SBOL Designer provides the ability to retrieve DNA components from SBOL compliant repositories, such as the SBPkb [3]. In combination, these features constitute a functional modular design application for synthetic biologists to create scalable synthetic DNA designs.

4. Conclusion

In this paper we present the first application for version control of synthetic biological designs. The initial implementation of this system in SBOL Designer allows synthetic biologists to manage the iterative design process using a graphical interface. We implemented the basic version control operations and demonstrated their use on a specific use case for modular design. Our tool leverages the emerging SBOL standard to store and visualize the designs to support interoperability with other SBOL compliant tools. As synthetic biology research scales up, it must adopt methodologies and approaches from engineering, including standardization, modularity and revision control management systems. Here, we have presented our ideas and a prototype implementation of a version control system specific for synthetic biology and SBOL. These are new capabilities for synthetic biologists and we must evaluate these against the needs of synthetic biology engineers in the field.

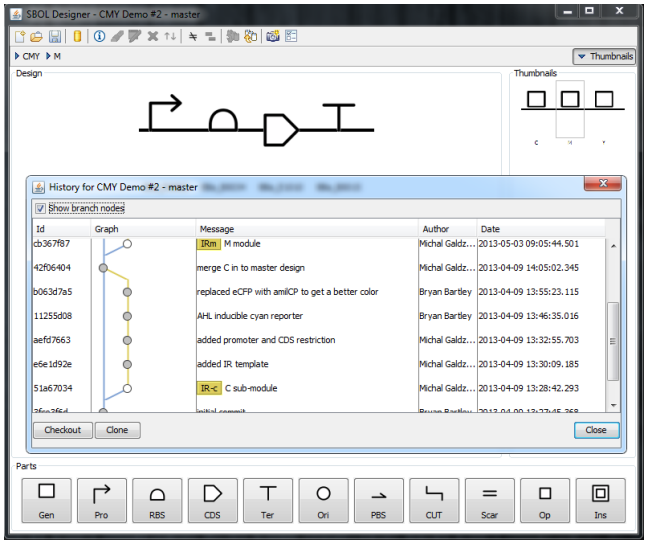


Figure 2. SBOL Designer showing the revision history of the CMY project

5. ACKNOWLEDGMENTS

Our thanks to Herbert Sauro for his participation in and support of this research. This work was partially funded by the NIH (grant #9R42HG006737).

6. REFERENCES

[1] Bonnet, J., Subsoontorn, P. and Endy, D. 2012. Rewritable digital data storage in live cells via engineered control of recombination directionality. *PNAS*. 109, 23 (Jun. 2012), 8884–9.

[2] Galdzicki, M. et al. 2012. Synthetic Biology Open Language (SBOL) Version 1.1.0. *BBF RFC #87*.

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

[4] Sleight, S.C. and Sauro, H.M. 2012. Design and construction of a prototype CMY (Cyan-Magenta-Yellow) genetic circuit as a mutational readout device to measure evolutionary stability dynamics and determine design principles for robust synthetic systems. *Artificial Life 13* (Jul. 2012), 481–488.

Microfluidic Genetic Circuit Assembly

David Sun Kong
MIT Lincoln Laboratory
244 Wood Street
Lexington, MA 02420
dkong@mit.edu

Jonathan Babb
MIT Biological Engineering
77 Massachusetts Avenue
Cambridge, MA 02139
jbabb@mit.edu

Todd Thorsen
MIT Lincoln Laboratory
244 Wood Street
Lexington, MA 02420
thorsen@ll.mit.edu

1. INTRODUCTION

Synthetic biologists are learning to engineer cellular behavior in increasingly sophisticated ways. DNA-based genetic circuits have emerged as a powerful vehicle for embedding new regulatory networks in cells. To name only a few examples, simple organisms have been engineered to form patterns in multicellular systems [1], communicate from cell-to-cell [2], express genes in controlled patterns [3], and in therapeutics, genetic circuitry has been employed to identify and destroy cancer cells [4].

A major barrier for increased exploration and discovery with genetic circuitry remains their laborious construction process. Genetic circuit assembly is either performed by hand or via automated robotic liquid handlers, which are cost prohibitive for many laboratories. Our goal is to develop microfluidic tools to make genetic circuit assembly easy, fast, and cheap. Microfluidic technology offers advantages of automation, reduced reagent consumption, and a platform fabrication cost that is a fraction of robotic handlers.

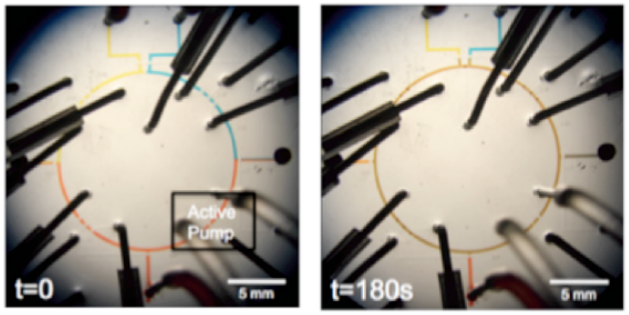


Figure 1. Mixing demonstration with food coloring in a microfluidic ring-mixer device.

Using microfluidics researchers have successfully miniaturized phosphoramidite-based DNA synthesis [5] and polymerase-mediated gene synthesis [6]. We demonstrate in this work, for the first time, the successful microfluidic assembly of genetic circuitry. Using the simple ring-mixer device shown in Figure 1, four widely used genetic circuit assembly biochemistries were miniaturized: ligation, Gateway [7], Gibson [8], and Golden Gate [9] assembly. The assemblies were evaluated in comparison to benchtop reactions and validated by screening, restriction digestion, and circuit testing in cells.

2. DEVICE DESIGN & FABRICATION

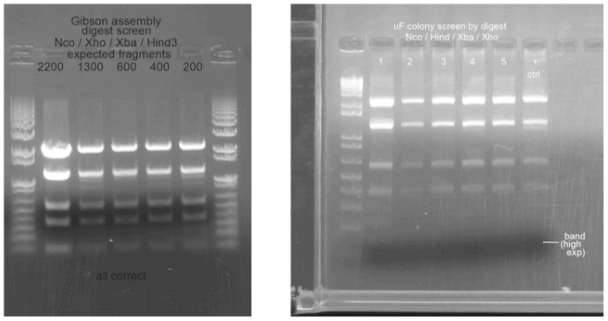
Polydimethylsiloxane (PDMS) based microfluidic devices, fabricated by multilayer soft lithography, enable thousands of flexible valves to be integrated into cm^2 devices and allow the end user to carry out complex fluid operations such as the parallel compartmentalization of thousands of picoliter-scale volumes, pumping, and mixing [10]. We have designed and fabricated a simple ring-mixer device (Figure 1) that enables mixing of up to 3 distinct components (shown as yellow, blue and orange food coloring) in a total reaction volume of 100-300 nanoliters. The device was fabricated by established multilayer soft lithography techniques.

3. GENETIC CIRCUIT DESIGN & ASSEMBLY TECHNIQUES

We chose four genetic circuits for assembly via our chosen biochemistries: (1) a plasmid backbone with a GFP BioBrick Insert [11] by ligation assembly; (2) a single-input miRNA sensor [4] assembled by Gateway assembly; (3) a circuit for generating quorum sensing in bacteria by Gibson assembly; and (4) an ATC-inducible circuit for GFP expression by Golden Gate. Assembly reactions were performed both at the benchtop at a 5 uL scale and in the microfluidic device, which was placed on an Eppendorf Mastercycler with *in situ* adapter for temperature cycling.

4. ASSEMBLY VALIDATION

Assembly reactions were validated by antibiotic screening and restriction digestion. An example of a restriction digestion comparing a benchtop and micrfluidic Gibson assembly is shown in Figure 2. Genetic circuitry was further validated by testing each circuit *in vivo* to verify its functionality.



Benchtop Assembly
(5 colonies mapped)

Microfluidic Assembly
(5 colonies mapped)

Figure 2. Restriction digestion verifying Gibson Assembly.

5. FUTURE WORK

We have also designed and fabricated microfluidic devices for massively parallel genetic circuit assembly. The device shown in Figure 3 features 256 reaction chambers, each ~120 nL in volume. Using such devices for high-throughput assembly, libraries of genetic circuits can be assembled in parallel in an automated fashion.

As an extension of research for classifying cancer cells by miRNA signatures [4], we seek to utilize such devices for the parallelized assembly of genetic circuits for the *in vivo* measurement of miRNA expression levels. Each reaction chamber is designed to yield a unique circuit for measuring a target miRNA, meaning with a handful of such devices it would be feasible to generate a set of sensors capable of measuring the majority of known human miRNAs.

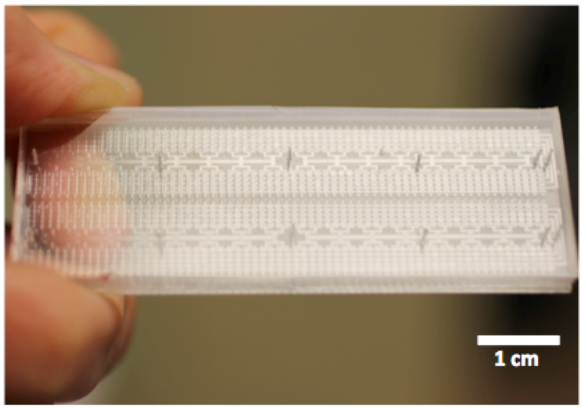


Figure 3. Optical image of a microfluidic device designed for the parallel assembly of 256 distinct genetic circuits.

6. ADDITIONAL AUTHORS

Scott Wick (MIT Lincoln Laboratory, swick@ll.mit.edu), Jeremy Gam (jgam@mit.edu), Ron Weiss (MIT Biological Engineering, rweiss@mit.edu), Peter Carr (MIT Lincoln Laboratory, carr@ll.mit.edu)

7. REFERENCES

[1] Basu, S., Y. Gerchman, C. H. Collins, F. H. Arnold, and R. Weiss. 2005. A synthetic multicellular system for programmed pattern formation. *Nature* 434:1130-1134.

[2] Weiss, R., and T. Knight. 2001. Engineered communications for microbial robotics. In *DNA computing*. Springer Berlin, Heidelberg. 1-16.

[3] Swinburne, I. A., D. G. Miguez, D. Landgraf, and P. A. Silver. 2008. Intron length increases oscillatory periods of gene expression in animal cells. *Genes & Development* 22:2342-2346.

[4] Xie, Z., L. Wroblewska, L. Prochazka, R. Weiss, and Y. Benenson. 2011. Multi-input RNAi-Based Logic Circuit for Identification of Specific Cancer Cells. *Science* 333, 1307-1311.

[5] Lee, C-C., Snyder T.M., and Quake, S.R. 2010. A microfluidic oligonucleotide synthesizer. *Nucleic Acids Research* 1:8.

[6] Kong, D.S., P.A. Carr, L. Chen, S. Zhang, and J.M. Jacobson. 2007. Parallel gene synthesis in a microfluidic device. *Nucelic Acids Research*, 35(8): e61.

[7] Life Technologies, *GATEWAY™ Cloning Technology Instruction Manual.*

[8] Gibson, D.G., L. Young, R.-Y. Chuang, J.C. Venter, C.A. Hutchinson, and H.O. Smith. 2009. Enzymaticassembly of DNA molecules up to several hundred kilobases. *Nature Methods*, 6(5): 343-347.

[9] C. Engler, R. Gruetzner, R. Kandzia, and S. Marillonnet. 2009. Golden gate shuffling: a one-pot DNA shuffling method based on type IIs restriction enzymes. *PloS one*, 4(5), e5553.

[10] Thorsen, T., S.J. Maerkl, and S.R. Quake, Microfluidic large-scale integration. *Science,* 2002. 298(5593): p. 580-584.

[11] T.F. Knight. 2003. Idempotent Vector Design for Standard Assembly of BioBricks. *MIT Synthetic Biology Working Group Technical Report*.

Identifiability Analysis to Characterize Mitochondrial Diseases

Claudio Angione
Computer Laboratory
University of Cambridge - UK
claudio.angione@cl.cam.ac.uk

Jole Costanza
Dept. of Maths & CS
University of Catania - Italy
costanza@dmi.unict.it

Giovanni Carapezza
Computer Laboratory
University of Cambridge - UK
giovanni.carapezza@cl.cam.ac.uk

Pietro Lió
Computer Laboratory
University of Cambridge - UK
pietro.lio@cl.cam.ac.uk

Giuseppe Nicosia
Dept. of Maths & CS
University of Catania - Italy
nicosia@dmi.unict.it

1. IDENTIFIABILITY ANALYSIS

Biological processes are usually modeled with components (e.g. parameters, variables) determined by measuring data and fitting to experiments. A non-identifiable component, i.e. a part of the system for which no unique solution exists, can be detected using the Identifiability Analysis (IA). The non-identifiability can be: (i) *structural*, i.e. some components are functionally related and therefore cannot be determined unambiguously, or (ii) *practical*, i.e. it is not possible to correctly estimate the component, due to low amount or quality of data available.

We apply the IA coupled with the flux balance analysis (FBA) model of the mitochondrion [4]. In particular, we perform flux balance analysis to obtain optimal production of ATP and NADH while varying the flux of the fumarate hydratase, responsible for the fumarase deficiency, a monogenic disorder in the TCA cycle. We consider the fluxes of the 135 reactions in the matrix as functions of the fumarase flux, and we infer functionally related reactions using the alternating conditional expectation algorithm (ACE) [1]. The connection between the identifiability analysis and a constraint-based structure stems from the fact that a non-identifiable constraint involving decision variables causes them to be functionally related. Given the space of the feasible fluxes, we apply IA in the regions with maximum ATP (healthy condition) and low ATP (pathological condition).

Identifiable solutions can help the decision-maker in a multi-objective optimization algorithm. Indeed, identifiability analysis can be used to augment optimization [2] and sensitivity analysis [5] in biochemical pathways. Our approach is general and could be easily tailored for other organisms.

The IA detects structural non-identifiable components of a model by fitting it repeatedly to experimental data and by analyzing the estimates of each component. Formally, let $K = [v_1, ..., v_m] \in \mathbb{R}^{n \times m}$ be the matrix of the n values for the m decision variables $\{x_1, ..., x_m\}$, where each column $v_i \in \mathbb{R}^n$ contains the n estimates for the ith variable. Let us suppose that the variables are related by unknown linear or non-linear functional relations. The true transformations that linearize these relations are denoted by α and β_j, namely $\alpha(x_i) = \sum_{j \neq i}^{m} \beta_j(x_j) + \xi$, where ξ represents a Gaussian noise. The ACE algorithm [1] estimates the optimal transformations $\hat{\alpha}(x_i)$ and $\hat{\beta}_j(x_j)$, $j \neq i$, such that $\hat{\alpha}(x_i) = \sum_{j \neq i}^{m} \hat{\beta}_j(x_j)$, where x_i is the response and all the other variables are the predictors.

Here we consider the FBA model of the mitochondrion by Smith et al. [4]. We choose the ATP and NADH productions as the objective functions to evaluate the distribution of fluxes in the network. The mitochondrial FBA model is composed of 423 reactions: 73 exchange reactions between the external environment and the mitochondrion, 135 reactions in the matrix compartment (e.g. the reactions of the Krebs cycle and beta-oxidation), and finally all the reactions that take place in the inner membrane space.

Without constraining the fumarate flux, the maximum ATP production is reached when the fumarate flux is equal to 6.9721 $\text{mmolh}^{-1}\text{gDW}^{-1}$. By calculating the maximum ATP and NADH production as function of the variation of each mitochondrial flux, we found that some reactions of citrate cycle have the maximum ATP production also at 6.9721 $\text{mmolh}^{-1}\text{gDW}^{-1}$. As described by Smith et al. [4], in the fumarase deficiency conditions the maximum ATP production reached after running the FBA is dramatically reduced. To evaluate the effect of the fumarate deficiency, we constrained the fumarate flux in the FBA model. We computed 1000 states of flux balance, constraining the fumarase flux by imposing at first the knockout condition (i.e., forcing the flux of the reaction to be zero), and then increasing the flux by 0.014 $\text{mmolh}^{-1}\text{gDW}^{-1}$, until 13.986 $\text{mmolh}^{-1}\text{gDW}^{-1}$. Therefore, we obtained a 135×1000 matrix V containing all the matrix fluxes in the model corresponding to fixed fluxes of fumarase.

The process of repeating estimates in the matrix K is replaced by taking into account all the points given as output by the FBA run in different fumarase conditions. We adopt the Mean Optimal Transformation Approach (MOTA) [3], by fixing at 5 the maximal number of parameters allowed to enclose a functional relation. The interdependent fluxes, which are non-identifiable, may be fixed at an arbitrary value in order to improve identifiability. This would not affect the model's dynamical properties, as the variables functionally related to the fixed variable change accordingly.

Healthy condition: maximum ATP.

In order to understand the changes in the functional relations with respect to the pathological condition, we apply the identifiability analysis to the arrays of the FBA output

CA and JC contributed equally to this work
IWBDA 2013, July 12-13, Imperial College, London, UK

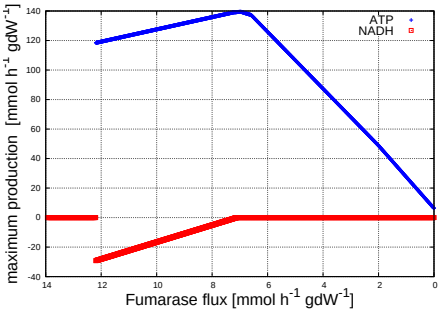


Figure 1: Maximum ATP and NADH production against different fumarate flux values. The healthy state and pathological state considered in our identifiability study show a phase transition in the middle.

fluxes whose fumarase flux belongs to $[8, 6]$ [mmolh^{-1}gdW^{-1}], which is the neighborhood of the maximum value of ATP (see Figure 1).

In the healthy condition we have strong functional relations between at most two variables, indicating that in the healthy state of the mitochondrion the fluxes are rather independent from one another, and there are no large groups of related parameters. This is also confirmed by the fact that several variables have no functional relations (i.e. the functional group of a variable is constituted only by the variable itself). Conversely, there are only four groups exhibiting a strong relation and constituted by two variables.

Pathological condition: decreasing ATP production.

Here we detect functional relations only in the pathological state of the mitochondrion. When the ATP decreases as function of the fumarase flux, in the interval $[-3, -1]$ mmolh^{-1}gdW^{-1} there is a signature of the onset of a pathological condition. The ATP decreases until it approximates 0, meaning that the mitochondrion is not working anymore.

Interestingly, the IA shows that the group in Figure 2 is the most recurrent, indicating that these four reactions strongly depend on each other. The most recurrent group is detected once for every reaction involved, indicating a very strong relation. All the reactions in this functional group are positively correlated with each other, except for R00512MM, which is negatively correlated.

The behaviors in the two conditions are significantly different in that the model shows a different number of strong relations, and also a different number of variables involved in these relations. Indeed, in the healthy status there were only four strong functional relations involving at most two variables. Most importantly, there is a strong functional group conserved in both conditions, i.e. R00713MM and R01648MM. The relation between these two parameters is shown in Figure 3 for the pathological status. Therefore, we can speculate that these two fluxes are structurally non-identifiable, and their non-identifiability does not depend on the mitochondrial status. This is due to the fact that both reaction share the succinate semialdehyde, and they are the most important reactions relying on this metabolite.

2. CONCLUSIONS

In this work we have analyzed how the genetic and the energy-converting pathways of mitochondria can be explored

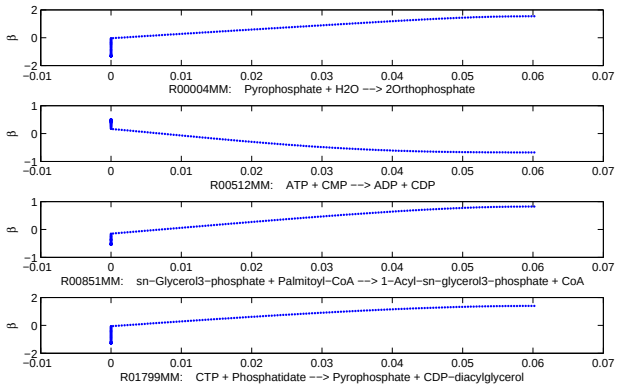


Figure 2: Functional relation among the fluxes [mmolh^{-1}gDW^{-1}] of the most recurrent functional group in the pathological state of mitochondria.

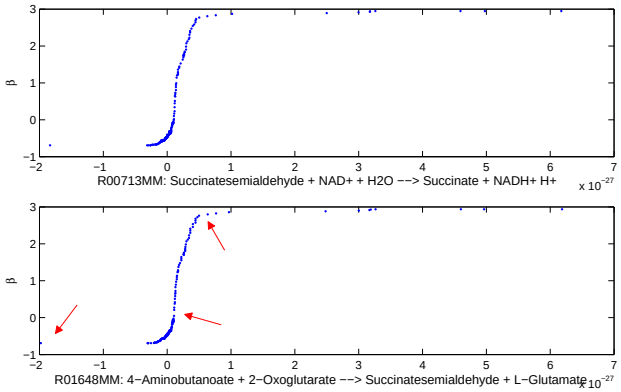


Figure 3: Functional relation in the pathological state in the group {R00713MM,R01648MM}, also detected as non-identifiable in healthy conditions. Therefore, their non-identifiability is conserved across mitochondrial conditions. The slight differences are indicated by red arrows.

using identifiability analysis. The identifiability analysis helps detect the differences between pathological and healthy conditions of the FBA model, inferring functional groups of variables. The elements of a functional group are variables functionally related with each other, which cannot be determined unambiguously. To our knowledge, this is the first time that the identifiability analysis is exploited in biological pathways in order to characterize a monogenic disorder.

The IA can be used in conjunction with multi-objective optimization, sensitivity, and ϵ-dominance. These techniques are framed in a unique pipeline applicable to a variety of conditions and organisms. The interplay between these techniques in a general-purpose framework can be exploited not merely to reach the optimal configuration for a model, but also to conduct tentative analyses on the variables and components of any model including ODEs, DAEs, FBA and GPR mappings. This allows to predict and characterize a disease status in a model by analyzing functionally related groups of fluxes. Striking applications of our pipeline could be in the field of metabolic engineering of a whole biological network.

3. REFERENCES

[1] L. Breiman and J. Friedman. Estimating optimal transformations for multiple regression and correlation. *Journal of the American Statistical Association*, 80(391):580–598, 1985.

[2] J. Costanza, G. Carapezza, C. Angione, P. Lió, and G. Nicosia. Robust design of microbial strains. *Bioinformatics*, 28(23):3097–3104, 2012.

[3] S. Hengl, C. Kreutz, J. Timmer, and T. Maiwald. Data-based identifiability analysis of non-linear dynamical models. *Bioinformatics*, 23(19):2612–2618, 2007.

[4] A. Smith and A. Robinson. A metabolic model of the mitochondrion and its use in modelling diseases of the tricarboxylic acid cycle. *BMC systems biology*, 5(1):102, 2011.

[5] G. Stracquadanio, R. Umeton, A. Papini, P. Liò, and G. Nicosia. Analysis and optimization of c3 photosynthetic carbon metabolism. In *BioInformatics and BioEngineering (BIBE), 2010 IEEE International Conference on*, pages 44–51. IEEE, 2010.

Automated Ratiometric Characterization using GEC

[Extended Abstract]

Boyan Yordanov
Biological Computation Group
Microsoft Research
yordanov@microsoft.com

Neil Dalchau
Biological Computation Group
Microsoft Research
ndalchau@microsoft.com

Paul Grant
Department of Plant Sciences
University of Cambridge
pg384@cam.ac.uk

Jim Haseloff
Department of Plant Sciences
University of Cambridge
jh295@cam.ac.uk

Stephen Emmott
Computational Science
Microsoft Research
semmott@microsoft.com

Andrew Phillips
Biological Computation Group
Microsoft Research
Andrew.Phillips@microsoft.com

1. INTRODUCTION

Recent DNA assembly methods have enabled the physical construction of large-scale biological devices but designing systems with a given behavior remains a challenge in synthetic biology. To enable the engineering of complex and robust systems, integrated experimental and computational methods are needed that allow precise experimental data to be collected and used to construct reliable computational models, that serve to make accurate predictions, leading to the design of functionally correct devices.

While fluorescent reporter proteins provide a convenient experimental tool for characterizing gene expression *in vivo*, such measurements often depend on experimental conditions. The use of relative measurements based on a reference (standard) promoter has been proposed as a strategy for attenuating such variability [4]. Recently, an alternative experimental and mathematical characterization framework was proposed by J. Brown in [2]. There, the use of spectrally-distinct reporter proteins led to a ratiometric characterization protocol where an internal reference signal provided more direct measurements of relative promoter activity. This data was also integrated in a mathematical framework extending [5], which allowed the parametrization of gene expression and regulation models. As the cost of characterization decreases and more data becomes available, computational tools capable of handling such information are required as part of biodesign platforms.

In this abstract, we present our recent results on extending, implementing and applying the ratiometric characterization methods from [2]. Specifically, we developed alternative strategies for representing and processing experimental data, and incorporated a Bayesian characterization framework to allow the parametrization of mechanistic mathematical models that capture uncertainty. This allowed us to compare competing hypothesis about the influence of experimental conditions and the precise mechanisms involved in gene regulation. We implemented these procedures within the GEC environment [6], allowing the method to be used conveniently as part of a computer-aided design process in synthetic biology [3].

2. GEC CHARACTERIZATION

In the following, we present the key extensions to the characterization framework from [2] that were developed alongside our implementation of the method within GEC [6].

2.1 Data Handling

Using the experimental technique developed by J. Brown in [2], a characterization experiment is performed by simultaneously measuring optical density (OD) (representing the density of bacteria) together with the fluorescence F_i of multiple signals (representing a reference signal F_r and the outputs of a device). Such measurements are usually collected in replicates and under a number of different experimental conditions. Extending the mathematical strategy from [5], the characterization method described in [2] relies on the computation of steady-state fluorescence values of each individual signal as $F_i^s = \frac{\partial F_i}{\partial \mathrm{OD}}$ through a subset of the data where a linear relationship between fluorescence and absorbance holds. For ratiometric measurements, the ratio $R_i = \frac{F_i^s}{F_r^s}$ of a signal to the reference value is considered.

In our implementation, we provide a convenient way for experimental data to be loaded and browsed within GEC (Fig. 1-A). The measurements of bacterial density (Fig. 1-D) and individual fluorescence channels (Fig. 1-E) are visualized for each sample. The range of data used for characterization is easily adjustable, while visual feedback is provided (Fig. 1-D,E), which enables the characterization of devices using data collected from different bacterial growth phases. While the steady-state fluorescence of individual signals F_i is computed (Fig. 1-F), the method also allows a more direct computation of the ratio R_i as $R_i = \frac{\partial F_i}{\partial F_r}$ (Fig. 1-G). Besides decreasing the variability between individual measurements, this strategy allows larger ranges of data (where the individual fluorescence measurements no longer satisfy the required linear relationship) to be used for characterization.

2.2 Mechanistic Models

Once the experimental data is processed as in Sec. 2.1, it can be used to study the behavior of a host and the synthetic devices it carries under different conditions. We use one of several classical models [7] to describe bacterial growth during an experiment (Fig. 1-D). This strategy leads to the identification of biologically meaningful parameters such as growth

lag and maximal culture density, besides the maximal (exponential) growth rate, which is used directly in the method to relate the (ratiometric) fluorescence measurements from Sec. 2.1 to promoter activities. In the example from Fig. 1, this strategy allows us to identify that the studied experimental condition (increasing concentrations of a certain chemical) affects the maximal growth rate (Fig. 1-B).

Besides the bacterial growth models, mechanistic descriptions of gene expression and regulation are used to derive a number of device behavior hypothesis, which are tested against experimental data. For example, we have studied the expected promoter activity for various modes of regulation, capturing activation and repression by a single and multiple transcription factors, multimerization as well as more complicated mechanisms involving additional interactions. Mechanistic explanations formulated to describe condition effects (*e.g.* the maximal growth rate or the promoter activity is affected by an inducer as in Fig. 1-B or Fig. 1-C) are also tested against the experimental data.

2.3 Bayesian Parameter Inference

To fit the mechanistic models described in Sec. 2.2 to the experimental data from Sec. 2.1, we developed a Bayesian parameter inference procedure using Markov Chain Monte Carlo (MCMC) methods. This approach also allows us to capture the uncertainty in experimental measurements within the parameter estimates, where distributions over the possible parameters values are identified rather than singleton estimates (Fig. 1-H). Ultimately, this uncertainty is propagated to the modeling predictions, which allows a more direct understanding of how characterization experiments inform the computational analysis. In addition, different plausible mechanisms captured in the models from Sec. 2.2 can be compared using this Bayesian approach, which we illustrate briefly in Sec. 3.

2.4 Implementation

The method described in this abstract has been implemented within a prototype version of GEC [6], allowing it to be used in an online environment in addition to the functionality previously provided by the tool. Bayesian parameter inference is performed using Filzbach [1]. A screenshot of the "data sheet" visualization produced as part of a characterization experiment is presented in Fig. 1.

3. APPLICATION

We illustrate the method through preliminary data, characterizing the functional response of a synthetic device when exposed to varying concentrations of some chemical signal C_{12}, which is observed to induce the expression of fluorescent output (a number of other characterization experiments are currently underway). As a possible hypothesis, the promoter activity P is characterized by the Hill function $P = \frac{aC_{12}^n + bK^n}{C_{12}^n + K^n}$ (Fig. 1-C) with the parameter distributions in (Fig. 1-H). Despite requiring more parameters, this model is preferred (based on model selection) when compared to the alternative hypotheses where $n = 1.0$ or $n = 2.0$.

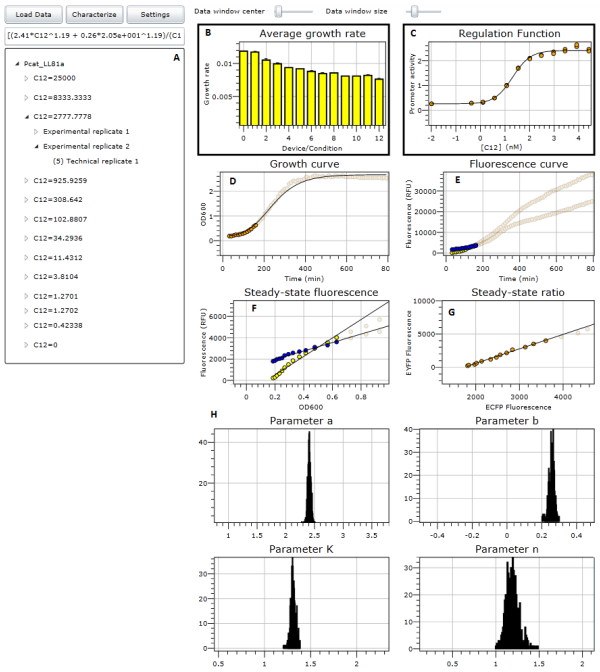


Figure 1: Ratiometric characterization in GEC. A: Experimental data can be inspected in a dataset viewer. B: Average growth rates across different experimental conditions. C: Functional response of device output to environmental conditions. D: Timecourse measurements of OD for the selected sample. E: Timecourse measurements of two separate fluorescence channels for the selected sample. F: Steady-state fluorescence computation for individual channels. G: Steady-state fluorescence ratio computation. H: Parameter uncertainty histograms for the identified functional response. Data selected for characterization is highlighted in D,E,F, and G.

4. REFERENCES

[1] Filzbach, Microsoft Research, Cambridge UK. `http://research.microsoft.com/en-us/um/cambridge/groups/science/tools/filzbach/filzbach.htm`.

[2] J. R. Brown. *A design framework for self-organised Turing patterns in microbial populations*. Ph.d. dissertation, University of Cambridge, 2011.

[3] N. Dalchau, M. J. Smith, S. Martin, J. R. Brown, S. Emmott, and A. Phillips. Towards the rational design of synthetic cells with prescribed population dynamics. *JR Soc Interface*, 9(76):2883–98, 2012.

[4] J. R. Kelly, A. J. Rubin, J. H. Davis, C. M. Ajo-Franklin, J. Cumbers, M. J. Czar, K. de Mora, A. L. Glieberman, D. D. Monie, and D. Endy. Measuring the activity of BioBrick promoters using an in vivo reference standard. *J Biol Eng*, 3:4, 2009.

[5] J. H. J. Leveau and S. E. Lindow. Predictive and Interpretive Simulation of Green Fluorescent Protein Expression in Reporter Bacteria. *J Bac*, 183(23), 2001.

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

[7] M. H. Zwietering, I. Jongenburger, F. M. Rombouts, and K. van 't Riet. Modeling of the bacterial growth curve. *App Env Microbiol*, 56(6):1875–81, 1990.

Interactive Experimentally Driven Algorithms for Optimized DNA Assembly

Evan Appleton
Graduate Program in Bioinformatics
Department of Electrical and Computer Engineering,
Boston University, 8 Saint Mary's St. Boston, MA, USA 02215
eapple@bu.edu

Jenhan Tao
Department of Electrical and Computer Engineering,
Boston University, 8 Saint Mary's St. Boston, MA, USA 02215
jenhantao@gmail.com

Douglas Densmore
Graduate Program in Bioinformatics
Department of Electrical and Computer Engineering,
Boston University, 8 Saint Mary's St. Boston, MA, USA 02215
dougd@bu.edu

1. MOTIVATION

Synthetic biologists build genetic devices from DNA parts. Once a device is designed, the problem of how to assemble it has a complex solution. Selection of the best assembly plan is error-prone and un-optimized when performed manually. Automated assembly is a key component of high-throughput cloning and can help biologists construct devices more efficiently and build part libraries with great modularity. DNA assembly algorithms exist to optimize the speed of assembly [2] but are agnostic to experimental constraints and only consider binary assembly methods. We present a method that uses experimental data and dynamic programming optimization algorithms in a computer-aided design tool called Raven that supports 'binary' and 'multi-way' ('one pot') DNA assemblies. Raven returns a hierarchical assembly plan optimized for time, modularity and efficiency, complete with assembly intermediates and their part junctions. For any cloning-based DNA assembly method [3, 6, 5], Raven automatically finds an optimized way to reuse existing parts and utilize experimental outcomes to inform future assembly optimizations to increase the speed and efficiency of assembly.

2. HIERARCHICAL ASSEMBLY PLAN

Raven breaks the DNA assembly problem into two major parts: selecting and generating optimized part junctions (or 'overhangs') and performing all cloning reactions necessary to achieve the final construct (Figure 1). The tool first determines an optimized set of assembly steps, then optimizes overhang selection (Algorithm 1); it is assumed that performing a successful assembly step is more costly and time consuming than performing a PCR reaction. This strategy is applied to binary [5], one-pot scarless [3] and one-pot modular [6] assembly methods to produce fast, modular and efficient assembly plans.

An optimized hierarchical assembly must minimize cloning steps, maximize part reuse and modularity and include high efficiency cloning steps to minimize the time and cost of assembly. All candidate steps are scored with heuristics to determine the best assembly plan. Time heuristics are based on assembly steps and stages. A step is the sum of all reactions required to assemble DNA parts with a specific assembly method and a stage is the collection of all steps that can be performed in parallel. Modularity heuristics are based on assembly intermediate sharing. If steps can be eliminated by sharing an intermediate, time and cost are saved. Finally, assembly efficiency heuristics are based on the number of parts connected in a single step, size of the target construct in a step and other secondary structure problems that will make a candidate step unsuccessful. The sum of all steps necessary to assemble a set of target parts is called a hierarchical assembly graph and the graph with the best score is passed to the overhang assignment algorithms.

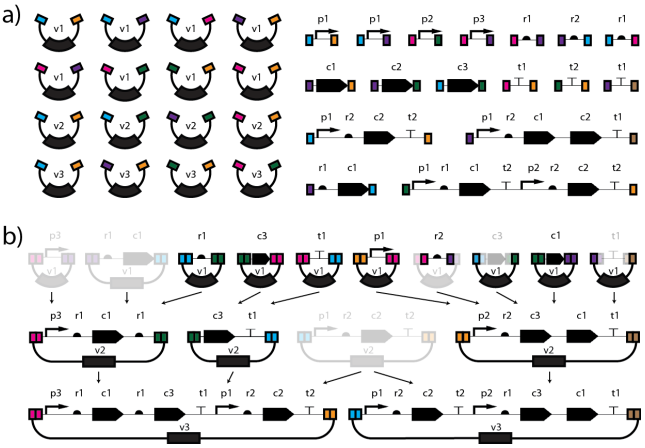


Figure 1: Example Optimized Assembly Plan a) Example existing library has a set of existing vector backbones, parts and devices with a variety of overhang pairs (shown as colored boxes) b) Example assembly plan re-uses parts and vectors from library (translucent) and shares intermediate parts. This assembly is performed in 2 stages, 6 steps and 13 PCRs.

3. PART JUNCTION SELECTION

Once an optimized hierarchical assembly graph is selected, optimized part junctions are selected. First, abstract part junctions place holders are selected based on specific assembly method constraints. Then, part junctions are optimized for time and modularity by selecting overhangs that may be shared the most in a given assembly graph and minimizing the introduction of new abstract part junctions. Finally, overhangs are mapped to concrete sequences, optimized for efficiency by selecting junctions that have diverse part junction sequences and do not introduce secondary structure conflicts.

Assembly Type	Steps	Stages	PCRs	Efficiency Heuristics
Binary	540	4	90	No
Modular One-Pot	300	2	187	No
Modular One-Pot	302	2	187	Yes
Scarless One-Pot	194	2	447	No
Scarless One-Pot	282	2	447	Yes

Table 1: Optimized assembly scores for 191 devices [4] for binary, modular one-pot and modular-scarless assembly with and without efficiency heuristics applied.

4. EXPERIMENTALIST INPUT

Sometimes an experimentalist might observe higher or lower assembly efficiencies than realized in the literature due to a number of experimental conditions such as cell competency, specific reaction conditions and other factors. So, with Raven a user is able to adjust the default efficiency numbers guiding the efficiency optimizations. A user may also wish to bias certain intermediates from appearing or not appearing, with or without mandatory inclusion or exclusion outside of the heuristics discussed above. For example, a user might wish to bias an assembly towards intermediates that fit a certain substructure such as a transcriptional unit. Another user might want to bias against performing PCR on large parts or parts with hairpins or other problematic structures. A user might also encounter unsuccessful cloning steps or PCRs (i.e. zero efficiency) or wish to test specific intermediate parts or part junctions. In all of these cases, it is imperative that these intermediates and/or part junctions are either biased or forced to appear or not appear. Raven optimizes around these additional constraints unless they are so restrictive that no valid solution can be found. Raven allows for lists of intermediates or overhangs to be biased for (recommended), biased against (discouraged), forced or forbidden. These parts and overhangs can be specified either manually in the GUI or automatically via Eugene [1] scripts.

5. RESULTS

We use Eugene [1] to generate a set of 1000 devices from a library of basic parts, each consisting of 5 transcriptional units and 20 basic parts. For a 20 basic part device, an unoptimized binary assembly of can be performed in 19 steps, a scarless one-pot assembly with up to 10 parts per reaction in 3 steps and a modular one-pot assembly with up to 6 parts per reaction in 6 steps. Thus, without optimization, these 1000 devices could be assembled in 19000 steps with binary assembly, 3000 steps with scarless one-pot assembly and 6000 steps with modular one-pot assembly. Using our tool without efficiency information, we find a binary assembly solution for 2707 steps (16293 step improvement), 1033 steps for scarless one-pot assembly (1967 step improvement) and 1201 steps for modular one-pot assembly (4799 step improvement). We also use our tool to generate optimized assembly plans for a contemporary device set of 191 devices

Algorithm 1 DNA Assembly (goalParts, partsLibrary, assmMeths, exptConstrs)

```
optimalGraphs = []
optimalHierarchicalGraphs = []
hierarchicalGraphs = []
for each part ∈ goalParts do
    bestGraph = []
    partitions = getPartitions(assmMeths, exptConstrs)
    hierarchicalGraphs = buildGraphs(partitions)
    for each graph ∈ hierarchicalGraphs do
        if score(graph) > score(bestGraph) then
            bestGraph = graph
        end if
    end for
    optimalHierarchicalGraphs.add(bestGraph)
end for
for each graph ∈ optimalHierarchicalGraphs do
    abstractAsgn = enforceOHRules(assmMeths)
    abstractAsgn = optOH(assmMethods, exptConstrs)
    finalAsgn = map(abstractAsgn, partsLibrary)
end for
return optimalGraphs
```

[4] for binary, scarless one-pot and modular one-pot assembly (Table 1). 11 additional libraries from the literature are also analyzed (not shown here) and show similar trends as efficiency heuristics are applied.

6. CONCLUSION

We demonstrate that our tool is capable of producing fast, modular and efficient DNA assemblies for large sets of devices. This is helpful for not only a human user to perform DNA assemblies at decreased time and cost, but also for automated approaches that rely on automated assembly planning and quality control.

7. REFERENCES

[1] Bilitchenko, L., Liu, A., Cheung, S., Weeding, E., Xia, B., Leguia, M., Anderson, J. C., and Densmore, D. Eugene: A domain specific language for specifying and constraining synthetic biological parts, devices, and systems. *PLoS ONE 6*, 4 (2011), e18882.

[2] Densmore, D., Hsiau, T. H. C., Kittleson, J. T., DeLoache, W., Batten, C., and Anderson, J. C. Algorithms for automated dna assembly. *Nucleic Acids Research 38*, 8 (2010), 2607–2616.

[3] Gibson, D. G., Young, L., Chuang, R.-Y., Venter, J. C., Hutchison, C. A., and Smith, H. O. Enzymatic assembly of DNA molecules up to several hundred kilobases. *Nature Methods 6*, 5 (2009).

[4] Lou, C., Stanton, B., Chen, Y.-J., Munsky, B., and Voigt, C. A. Ribozyme-based insulator parts buffer synthetic circuits from genetic context. *Nature Biotechnology 30*, 11 (2012), 1137–1141.

[5] Shetty, R. P., Endy, D., and Knight, T. F. Engineering biobrick vectors from biobrick parts. *Journal of Biological Engineering 2*, 5 (2008).

[6] Weber, E., Engler, C., Gruetzner, R., Werner, S., and Marillonet, S. A modular cloning system for standardized assembly of multigene constructs. *PloS ONE 6*, 2 (2010).

Optimizing module matching for synthetic gene circuit design automation

Linh Huynh
Department of Computer Science
& UC Davis Genome Center
University of California, Davis
huynh@ucdavis.edu

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

1. INTRODUCTION

An integral challenge in automated synthetic circuit design is to select the optimal set of parts to populate an abstract circuit topology, so that the circuit behavior best approximates the desired one. In some cases, it is also possible to reuse multi-part constructs, or *modules* that have been already built and experimentally characterized. Efficient part and module matching algorithms are essential to systematically search the solution space and their significance will only increase in the following years due to the projected explosion in part libraries and circuit complexity. Here, we present a matching method based on an iterative node traversal algorithm that can guarantee optimality in the set selection. Results from the integration of the proposed method in the SBROME CAD tool show scalable performance and optimal set selection in a database of experimentally constructed parts and modules. This work represents a fundamental departure from the previous heuristic-based part matching methods [1][2] and is a step towards maximizing efficiency in synthetic circuit design.

2. AN ILLUSTRATIVE EXAMPLE

Assume that we would like to design a circuit that has the topology specified in figure 1b based on an available module library, as shown in figure 1a. Since the final gate is an AND gate and the circuit output is *gfp*, there are only module 4 and module 5 that can match. If we use a greedy strategy based on the matched module size, then the module 4 will be chosen over module 5 (since the first match both A2 and N2) and the gate A1 can only match with the one in module 6, while there are three ways to match the gates N1 and O1. In the first case, where N1 and O1 are matched with module 1, this ultimately does not constitute a solution as there is no way to choose the common input y since module 1 and module 6 share no input. In the second case, N1 and O1 can be matched with module 2 and y can be set to be aTc. However this still does not lead to a solution, since module 2 uses cI as a signal and it will interfere with cI that connects gates A1 (matched with module 6) and N2 (matched with module 4). In the third and last scenario, if N1 and O1 are matched by two modules 8 and 7 as in figure 1c then this actually constitutes a solution since we can choose the common input y as aTc and there is no cross-talk. However this greedy selection requires a final construct with four modules, and it only presents a local optimum. For example, if the output *gfp* is matched with the module 5, then we need only two more modules (module 1 is matched with gates N1 and O1 while module 3 is matched with gates A1 and N2)

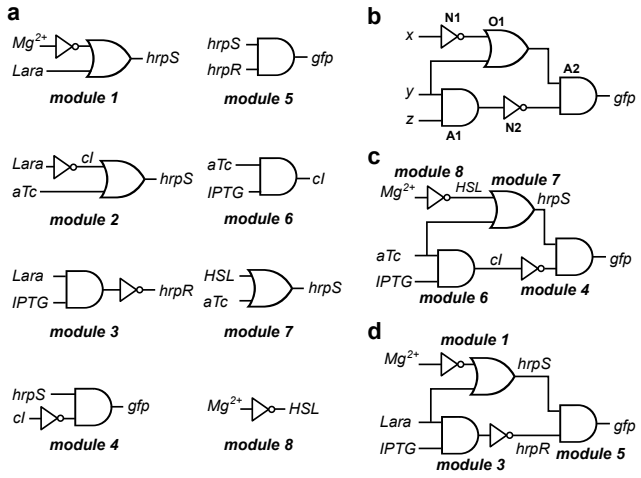


Figure 1: An illustrated example of the module matching problem. (a) The module library. (b) The abstract input circuit graph. (c) A non-optimal matching. (d) An optimal matching

to assemble the whole circuit as in figure 1d. This example illustrates why traditional heuristics are sub-optimal in module matching and it highlights two important features of biological circuits that have to be taken into account, namely the part compatibility and cross-talk.

3. METHODS

Gene circuit representation: Each circuit is represented by a *circuit graph* $G = (V, E, V_I, V_O, \tau_1, \tau_2)$ where V and E are the vertex set and the edge set, V_I and V_O are subsets of V that represent inputs and outputs respectively, τ_1 and τ_2 are vectors that describe the type of each node (ligand, mRNA, protein, gate) and each edge (activatory or inhibitory), respectively.

Problem formulation: Given a circuit graph and a library that contains parts and modules, find a minimum set of parts/modules that can be assembled together to match with the given input circuit graph so the final graph contains no cross-talk between components. In this research, we limit our study for only single output directed acyclic graphs.

A dynamic programming based algorithm: Module matching is similar to topology mapping [3] in electric circuit design which can be solved efficiently by using the dynamic programming approach. However, in contrast to electrical circuits, biological circuits do not have a universal carrier

Table 1: Running time (seconds) for sample circuits

	Module library size		
Design	25	50	97
`not_and_1`	0.01	0.05	0.06
`not_and_2`	0.02	0.28	10.64
`2_to_1_mux_1`	0.02	0.16	0.18
`2_to_1_mux_2`	0.06	0.14	0.15
`2_to_1_mux_3`	0.02	0.54	19.81

of information and they have to utilize multiple molecular species to do so, since there is no explicit wiring to isolate components. This constraint breaks optimal substructure condition of any dynamic programming approach since the optimal solution sometimes has to be constructed from non-optimal sub-solutions because the optimal sub-solutions may be not be compatible or have cross-talk effects.

To overcome these obstacles, we have to store both non-optimal solutions and the information about the connection compatibility and cross-talk effects to ensure the optimal solution is found. For each node v, the *transitive fan-in subgraph* of v is a subgraph that contains v itself and all nodes connected to v by some directed paths. A *matching at* v is a matching of a set of parts and modules with the transitive fan-in subgraph of v that satisfies the constraints of connection compatibility and cross-talk effects. The set $\mathcal{R}(v)$ stores all possible matchings at v and their necessarily related information. In particular, each element $r \in \mathcal{R}(v)$ is a 4-component vector in which i) COST is the number of parts and modules, ii) SIGNAL contains all internal molecular species inside parts and modules that are used to match with the transitive fan-in subgraph of v iii) OUTPUT is the output signal at v with this matching, and iv) information about the recursion so that the procedure TRACEBACK can trace back to find the final solution.

The key idea here is that the set $\mathcal{R}(v)$ will be constructed recursively following algorithm 1. All nodes are traversed by their topological order. At each node v, $\mathcal{R}(v)$ stores all matching combinations that satisfy the constraints of compatibility and cross-talk (checked by procedures COMP and CROSS respectively), each of these combinations contains a matching m at v together the matching at each input of m recursively. This algorithm can be thought of as an exhaustive search that investigates all possible solutions systematically but it's performance is improved by eliminating solutions that violate the constraints of the connection compatibility and cross-talk effects as early as they occur.

Algorithm 1 Module matching algorithm

Input: $G = (V, E, V_I, \{v_o\}, \tau_1, \tau_2)$
Module & part library $\mathcal{L}$
Output: List of fixed circuits *Sol*
for each node $v \in V$ in the topological order **do**
  $\mathcal{R}(v) \leftarrow \emptyset$
  $\mathcal{M} \leftarrow \text{MATCH}(v, \mathcal{L})$
  for each match $m \in \mathcal{M}$ **do**
    $\mathcal{I} = \{v_{i_1}, v_{i_2}, \ldots, v_{i_k}\} \leftarrow \text{INPUT}(m)$
    for $r = (r_{i_1}, r_{i_2}, \ldots, r_{i_k}) \in \prod_{j=1}^{k} \mathcal{R}(v_{i_j})$ **do**
      if $(\text{COMPATIBLE}(m, r) \wedge \neg\text{CROSS}(m, r))$ **then**
        $R(v) \leftarrow R(v) \cup \{\big(\sum_{j=1}^{k} \text{COST}(r_{i_j}) + 1,$
        $\bigcup_{j=1}^{k} \text{SIGNAL}(r_{i_j}) \cup \text{SIGNAL}(m),$
        $\text{OUTPUT}(m),$
        $(m, r_{i_1}, r_{i_2}, \ldots, r_{i_k})\big)\}$
      end if
    end for
  end for
end for
$\mathcal{O} = \{r \in \mathcal{R}(v_o) \mid \text{COST(r) } \textit{is miminum}\}$
$Sol = \text{TRACEBACK}(\mathcal{O})$

4. RESULTS

We built a library of totally 97 parts and modules where 47 are reported in literature and 50 other are synthetic modules (similar constructs with alternative promoters). Since the algorithm will always return the optimal solution, we only consider the running time to evaluate its performance. The running time for each circuit with different library sizes is reported in table 1. In general, the running time scales well in the case where the input topology contains logical gates only (`not_and_1`, `2_to_1_mux_1` and `2_to_1_mux_2`) because their solution space is small and designs are eliminated due to compatibility constraints. When the input topology contains both logic gates and elementary parts, the algorithm has more flexibility to choose smaller modules to match, which leads to an extended solution space and running time (`not_and_2` and `2_to_1_mux_3`).

5. DISCUSSION

We present a module matching strategy that returns the optimal set of parts for synthetic gene circuit design automation. There are several extensions of this work that warrant further investigation. First, we can improve the computational performance by reducing the number of records in each $\mathcal{R}(v)$. To do that, we can apply a branch and bound technique by using heuristics to find some initial solutions and then prune out the records that cannot lead to an optimal solution. Second, we can extend this approach for general graphs that may contain loops or multiple outputs. More far-reaching and ambitious extensions of this method include its application in multi-cellular system design, where the objectives are to minimize both the number of cell types and the size of the resulting circuit.

6. REFERENCES

[1] L. Huynh, A. Tsoukalas, M. Köppe, and I. Tagkopoulos, "Sbrome: A scalable optimization and module matching framework for automated biosystems design," *ACS Synthetic Biology*, 2012.

[2] F. Yaman, S. Bhatia, A. Adler, D. Densmore, and J. Beal, "Automated selection of synthetic biology parts for genetic regulatory networks," *ACS Synthetic Biology*, vol. 1, no. 8, pp. 332–344, 2012.

[3] K. Keutzer, "Dagon: technology binding and local optimization by dag matching," in *DAC*, pp. 617–624, ACM, 1988.

Directed Acyclic Graph-Based Technology Mapping of Genetic Circuit Models

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

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

1. INTRODUCTION

As engineering foundations such as standards and abstraction begin to mature within synthetic biology, one limiting factor for the design of more complex genetic circuits will be the availability and effectiveness of *genetic design automation* (GDA) tools. Ideally, a synthetic biologist could design genetic circuits at a fairly high level of abstraction, focusing on a desired behavior rather than the DNA components used to implement said behavior. Such a synthetic biologist would use GDA tools to automatically map behavioral descriptions for genetic circuits to standardized sets of DNA components, thereby decreasing the expert knowledge and time required during the design process. To this end, we have developed a genetic technology mapping algorithm and implemented it in our GDA tool `iBioSim` [5].

Previous research on the problem of automated selection of DNA components to meet a behavioral specification has resulted in GDA tools such as the Genetic Engineering of Living Cells [6] (GEC), MatchMaker [7], Cello [8], and the Synthetic Biology Reusable Optimization Methodology [3] (SBROME). Broadly speaking, our approach to genetic technology mapping is most similar to the Cello approach, as both build on the *directed acyclic graph*-based (DAG-based) mapping techniques originally used to map electronic circuits. However, we also make copious use of standards such as the *systems biology markup language* (SBML), an established standard for describing biochemical models that is supported by over 250 software tools [2], and the *synthetic biology open language* (SBOL), a nascent standard for describing DNA components that has growing support among GDA tools [1]. By incorporating standards for the function and structure of DNA into our approach, we hope to provide for the possibility of greater interoperability between GDA tools in the future.

2. DAG-BASED TECHNOLOGY MAPPING

Our algorithm can be divided into three steps. First, the algorithm constructs regulatory DAG representations from a *genetic circuit model* (GCM) specification written in SBML and a SBOL-annotated GCM library, and it partitions the DAGs into rooted DAGs. Second, the algorithm decomposes these DAGs to a canonical form and matches the library DAGs to each node in the specification DAGs. Lastly, DAG matches are selected via dynamic programming and recursive backtracking to obtain the corresponding solution set of SBOL-annotated GCMs that may be composed to satisfy or "cover" the input GCM specification in the fewest base pairs of DNA.

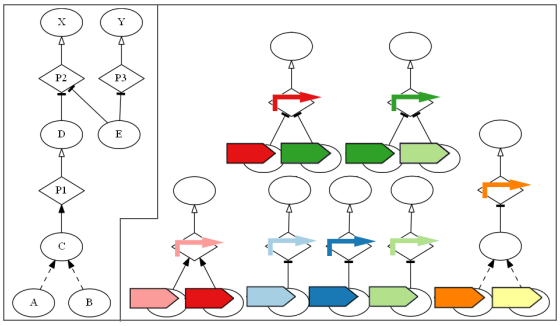


Figure 1: A regulatory DAG specification (left) and library (right) constructed from GCMs. Circles are species nodes, diamonds are promoter nodes, open arrows are production edges, closed arrows are activation edges, tee arrows are repression edges, and dotted arrows are complexation edges. Colored bent arrows and box arrows represent the different SBOL promoter and coding sequence DNA components that annotate the source GCMs for the DAG library.

2.1 DAG Construction and Partitioning

A GCM written in SBML consists of a set of molecular species and a set of reactions, the latter of which are annnotated as genetic production or complex formation reactions with the appropriate *systems biology ontology* (SBO) terms. Within the genetic production reactions, species modifiers are similarly annotated with SBO terms as repressors or activators. During regulatory graph construction, nodes are created for each species and genetic production reaction, and are labeled as species and promoters, respectively. Directed edges are created between nodes in accordance with the SBO annotations on species modifiers, and are labeled as repression, activation, or production. Lastly, directed edges are also created between nodes for reactant and product species of complex formation reactions, and are labeled as complexation. Fig. 1 displays DAGs constructed from a sample GCM specification and library.

During partitioning (see Fig. 2), the library and specification DAGs are split at nodes with more than one outgoing edge into n rooted DAGs, where n is the total number of outgoing edges at these nodes. This enables us to match the partitions in linear time and optimally cover them using *depth-first search* (DFS) and dynamic programming. The tradeoff is that there is no guarantee of global optimality when the covered partitions are re-composed to form a solu-

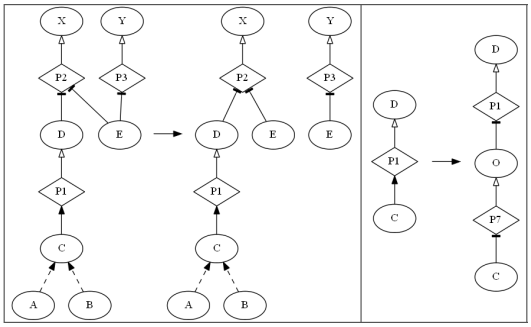


Figure 2: Examples of partitioning (left) and decomposition (right) of regulatory DAGs.

tion. As with any heuristic, the hope is that the non-globally optimal solution is found more quickly and that it is still of fairly high quality.

2.2 DAG Decomposition and Matching

Decomposition (see Fig. 2) transforms the specification and library DAGs so that they only contain edges labeled as repression or production and no nodes with more than two incoming edges. So long as our library contains GCMs describing single genes that are regulated in this manner, decomposition guarantees that a match can be found at each node in our specification DAG. Decomposition also increases the number of matches that can be made at each node, thereby increasing the possibilities for covering and improving the quality of our final solution.

When matching DAGs, our approach resembles that taken in Keutzer's foundational technology mapping system for electronic circuits, DAGON [4]. Like DAGON, our tool also uses the Aho-Corasick algorithm to match strings of characters encoding paths through our library DAGs with strings encoding paths through our specification DAG. Through string representation and the construction of a discrete finite automaton, the Aho-Corasick algorithm achieves a worst-case run time that scales linearly with the size of the specification and independently of the size of the library.

2.3 DAG Covering

Due to the possibility of cross-talk resulting from shared transcription factors, our algorithm is restricted to selecting a particular matching library DAG only once while covering the specification DAGs. Consequently, each covering decision has implications that may effect future covering decisions, which prevents us from guaranteeing that an optimal solution is found with a single application of the DFS. Hence, our approach adds recursive backtracking to Keutzer's approach to covering which effectively creates a *branch-and-bound* algorithm.

In our branch-and-bound algorithm, whenever a solution or dead end is encountered, the DFS steps back to the last time a decision was made and makes the next best decision provided that the theoretical optimum cost of a solution using this decision does not exceed the cost of the best solution found so far. The theoretical optimum cost from a given point in the DFS is determined by looking ahead and covering without regard to component cross-talk. While branch-and-bound guarantees that the optimal solution is eventually found, it also has a worst-case run time that scales exponentially with input size. In practice, the average run time is made reasonable through heuristics to prune the decision tree such as bounding the solution cost.

3. CONCLUSIONS

We have developed and implemented an algorithm for DAG-based genetic technology mapping in our GDA tool, `iBioSim`. It is among the first genetic technology mapping algorithms to adapt heuristics from electrical engineering and perhaps that which makes the greatest use of established and developing standards for describing the function and structure of DNA. While our approach currently only handles genetic circuits that are directed and acyclic in nature (i.e. no feedback), we intend to handle cyclic genetic circuits by extending our partitioning step with existing algorithms for efficiently cutting cyclic directed graphs into DAGs.

4. ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant No. CCF-0916042 and CCF-1218095. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

5. REFERENCES

[1] M. Galdzicki et al. Synthetic biology open language (SBOL) version 1.0.0. BBF RFC #84.

[2] M. Hucka et al. The Systems Biology Markup Language (SBML): A medium for representation and exchange of biochemical network models. *Bioinform.*, 19(4):524–531, 2003.

[3] L. Huynh, A. Tsoukalas, M. Koppe, and I. Tagkopoulos. SBROME: A scalable optimization and module matching framework for automated biosystems design. *ACS Synthetic Biology.*

[4] K. Keutzer. DAGON: Technology binding and local optimization by DAG matching. In *Proceedings of the 24th ACM/IEEE design automation conference*, pages 341–347. ACM, 1987.

[5] C. Madsen, C. Myers, T. Patterson, N. Roehner, J. Stevens, and C. Winstead. Design and test of genetic circuits using iBioSim. *IEEE Design and Test*, 2012.

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

[7] F. Yaman, S. Bhatia, A. Adler, D. Densmore, and J. Beal. Automated selection of synthetic biology parts for genetic regulatory networks. *ACS Synth. Biol.*, 1:332–344, 2012.

[8] B. Yordanov, E. Appleton, R. Ganguly, E. A. Gol, S. B. Carr, S. Bhatia, T. Haddock, C. Belta, and D. Densmore. Experimentally driven verification of synthetic biological circuits. In *Design, automation, and test in Europe conference and exhibition (DATE), 2012*, pages 236–241, 2012.

A New Algorithm for Combinatorial DNA Library Assembly

Jonathan Blakes
School of Computer Science
University of Nottingham
Nottingham NG8 1BB, UK
+44 115 84 68403
jonathan.blakes@nottingham.ac.uk

Ofir Raz
Department of Computer Science and Applied Math.
Weizmann Institute of Science
Rehovot 76100, Israel
+972 8 934 4506
ofir.raz@weizmann.ac.il

Natalio Krasnogor *
School of Computer Science
University of Nottingham
Nottingham NG8 1BB, UK
+44 115 84 67592
natalio.krasnogor@nottingham.ac.uk

Ehud Shapiro *
Department of Computer Science and Applied Math.
Weizmann Institute of Science
Rehovot 76100, Israel
+972 8 934 4506
ehud.shapiro@weizmann.ac.il

ABSTRACT

We describe a novel greedy algorithm for computing near-optimal DNA assembly graphs and show empirically that it runs in linear time, enabling almost instantaneous planning of DNA library sizes exceeding the capacity of today's biochemical assembly methods. We compare assembly graph quality and algorithmic performance to the results obtained in [1], demonstrating that they are significantly faster to obtain and equivalent to the best results for DNA library assembly with intermediate reuse found in the literature.

Categories and Subject Descriptors

F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems – *sequencing and scheduling, computations on discrete structures*.

General Terms

Algorithms, Performance, Design, Experimentation.

Keywords

DNA assembly, combinatorial DNA libraries, gene design, genomic libraries, synthetic biology.

1. INTRODUCTION

Combinatorial DNA libraries, composed of natural and synthetic biological parts, can be assembled in an efficient manner. This is achieved by recognising that certain combinations of primitive and composite parts can occur multiple times within a library. Thus ordering the necessary assembly steps into stages so that these common, reusable intermediates are assembled first, in parallel, allows for reuse of correctly assembled composites between library members.

* Corresponding authors.

Fifth International Workshop on Bio-Design Automation (IWBDA 2013), July 12–13, 2013, London, United Kingdom.

1.1 Formal Problem Definition and Example

Let S be a set of available DNA parts and T a set whose elements are ordered sequences of elements in S which constitute a DNA library. Define a library assembly graph (as in Figure 1) as a hierarchical clustering, representing the binary convergence of nodes in S to nodes in T.

Given an assembly graph quality/cost function(stages,steps) that describes the ratio in penalties between the number of stages (depth of the construction graph) and number of intermediate steps (nodes that are neither in S nor in T), the problem is finding the assembly graph with optimal quality/minimal cost. In what follows we use the same cost function as [1] which seeks to minimize the number of stages and then the number of steps.

For S={A,B,C,D,E} and T={ABE,ABDE,ACDE,ADE} Figure 2 shows a suboptimal (2,8) and the optimal (2,7) assemblies.

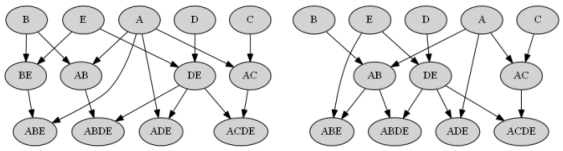


Figure 1: Suboptimal (left) and optimal library assembly.

1.2 State-of-the-Art

The authors of [1] contributed an iterative refinement algorithm based on dynamic programming for finding the optimal binary assembly tree (minimum number of stages and steps) of a *single* library sequence ('goal-part'), with worst case $O(n^3)$ scaling where n is the number of subparts, and where the minimum number of stages is $\text{ceil}(\log_2(n))$. They extended this to multiple-goal-parts by introducing 'slack' and 'sharing' factors that assemble all goal-parts by sharing nodes between trees to create a graph which reduces the number of steps, with $O(k^2 n_{max}^2)$ scaling where k is the number of single goal-part assembly graphs created, n_{max} is the maximum number of subparts across all goal-parts, and the minimum number of stages is $\text{ceil}(\log_2(n_{max}))$. In what follows we refer to this algorithm as A_1, and our new algorithm as A_{new}.

Greedy algorithms apply locally optimal choices at each stage to approximate, or even obtain, a globally optimal solution in reasonable time; ideally for problems exhibiting 'optimal substructure' (also necessary for dynamic programming).

2. ALGORITHM

Unlike [1] we address libraries of many sequences directly. Our algorithm leverages the fact that all concatenation steps occur between a *pair* of adjacent parts (ordered 2-tuples of the *p*-tuples described in section 2.1). Therefore, when applied greedily to all sequences, each pair is mutually exclusive of any other pairs, either to the left or right of an occurrence that contain the 1st or 2nd part respectively; i.e. overlapping pairs constituting a 3-tuple *triple* which cannot both be concatenated in the same stage.

We rank concatenations using a *scoring* function that is simply the number of occurrences of that pair throughout the library, although other biological or manufacturing constraints could be taken into account by alternative scoring functions.

2.1 DNA Library Encoding and Example

Sequences of the DNA library to assemble are encoded in terms of their constituent subparts, i.e. as a list of *s*-tuples (or any hashable ordered sequence) of *p*-tuples of hashable objects representing subparts. Table 1 shows the state of the four sequence example library as it evolves under the optimal assembly plan shown in Figure 1 (sequences are shown as hashable Python objects conforming to this library encoding).

Table 1: Library state and resulting steps in a run of A_{new}.

Initial input	After stage 1	After stage 2
`library=[ ((A,),(B,),(E,)), ((A,),(B,),(D,),(E,)), ((A,),(C,),(D,),(E,)), ((A,),(D,),(E,))]`	`library=[ ((A,B),(E,)), ((A,B),(D,E)), ((A,C),(D,E)), ((A,),(D,E))]`	`library=[ ((A,B,E)), ((A,B,D,E)), ((A,C,D,E)), ((A,D,E))]`
`steps=[ ((A,),(B,)), ((D,),(E,)), ((A,),(C,))]`	`steps=[ ((A,B),(E,)), ((A,B),(D,E)), ((A,C),(D,E)), ((A,),(D,E))]`	`steps=[]`

Note that since optimal steps were derived in stage 1, the `steps` of stage 2 are the same as the sequences in `library` after stage 1.

2.2 Description

For each stage, *maps* from pairs and triples to *multisets*, counting the number of times they occur in a sequence, are computed and from these, *multimaps* of pairs to containing triples and triples to contained pairs, are derived.

Using the per-pair scoring function, a list of (pair, score) items is computed, shuffled (to randomize the order of equal scoring pairs in a stable sort - allowing multiple runs with different resultant assembly graphs) and sorted by descending score.

For each (pair, score) a pair is added to a list of *excluded* pairs if it has a score ≤ 0 (allowing custom scoring functions to exclude certain pairs by design) or, if not already excluded, it is added to the set of steps for this stage and all other pairs that are in a triple with the current pair are excluded instead. If no steps were added in this stage then the list of stages, containing the lists of steps for each stage, is returned.

Lastly, each of the highest scoring pairs that are not excluded by a higher scoring pair are then concatenated by seeking for occurrences in the sequences of the multiset mapped to that pair and moving elements of the right p_r -tuple into the left p_l-tuple, leaving a pair of $(p_l + p_r)$- and 0-tuples, so as to not create new occurrences of any pairs.

Once all possible concatenations have been applied, all 0-tuples are removed and the algorithm loops.

3. RESULTS

We compare the performance of our algorithm to that of A_1 in terms of assembly graph quality (smallest number of stages then steps, Table 2) and running time (Table 3) using the one synthetic and two real world libraries of [1]: a small dataset for exhaustive search, 'phagemid' and 'iGEM-2008' (containing 2 duplicates).

Table 2: Quality (|stages|,|steps|) of assembly graph

Library |size| / Algorithm	Exhaustive |5|	Phagemid |131|	iGEM-2008 |395|
A_1	(3,11)	(4,202*)	(5, 808*)
A_{new}	(3,11) best (3,15) worst	(4,202) or (4,208) †	(5,808) best (6,841) worst

* A_1 values were estimated from [1] Figure 7. † Only outcomes.

Table 2 shows that our algorithm A_{new} obtains the optimal solutions found by A_1. Therefore we conclude that this greedy approach can optimally assemble real-world DNA libraries.

Table 3: Running time (seconds) for subsets of iGEM-2008

Library size / Algorithm	50	100	150	200	250	300	350	395
A_1 ‡	0.1	1	45	120	290	405	620	720
A_{new} mean of 10 runs	.007 $\pm 10^{-5}$	.014 $\pm 10^{-4}$	.020 $\pm 10^{-5}$	.027 $\pm 10^{-4}$	.033 $\pm 10^{-4}$	.039 $\pm 10^{-4}$	.045 $\pm 10^{-4}$	.053 $\pm 10^{-4}$

‡ A_1 values were estimated from [1] Supplemental Figure 1.

Table 3 shows the running times of A_{new} were at least two orders of magnitude faster than A_1 and imply that A_{new} scales linearly with the number of sequences. The implementation language, runtime and hardware these timings were obtained with may differ; we used Python running on a single i7-2670 2.2Mhz core.

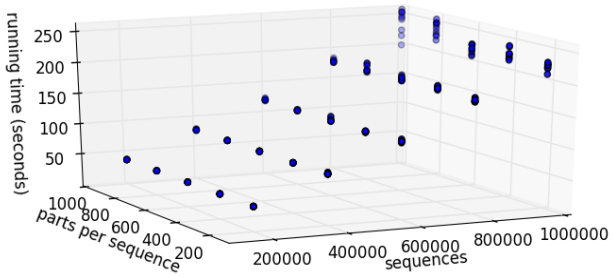


Figure 2: Running times of generated large-scale combinatorial DNA libraries for A_{new} (10 runs each).

To investigate the growth of running time with library size further, we generated synthetic combinatorial libraries with up to 1 million sequences and 1000 parts per sequence. Figure 2 shows how the running times on these synthetic libraries continue the trend observed in Table 3: increasing linearly with the number of sequences; and that |parts| per sequence has only a minor effect.

4. PROJECTS

This work was supported by CADMAD (FP7 STREP 265505), ROADBLOCK (EP/I031642/1) and AUDACIOUS (EP/J004111/1).

5. REFERENCES

[1] Densmore, D., Hsiau, T. H.-C., Kittleson, J. T., DeLoache, W., Batten, C. and Anderson, J. C. 2010. Algorithms for automated DNA assembly. Nucleic Acids Research 38, 8 (Mar. 2010), 2607-2616. DOI=http://dx.doi.org/10.1093/nar/gkq165

Accurate Predictions of Genetic Circuit Behavior from Part Characterization and Modular Composition

Noah Davidsohn
MIT
ndavidso@mit.edu

Jacob Beal
Raytheon BBN Technologies
jakebeal@bbn.com

Aaron Adler
Raytheon BBN Technologies
aadler@bbn.com

Fusun Yaman
Raytheon BBN Technologies
fusun@bbn.com

Yinqing Li
MIT
yinqingl@mit.edu

Zhen Xie
Tsinghua University
zhenxie@tsinghua.edu.cn

Ron Weiss
MIT
rweiss@mit.edu

1. OVERVIEW

A long-standing goal of synthetic biology is to rapidly engineer new regulatory circuits from simpler regulatory elements [8, 16, 2, 7]. As the complexity of engineered circuits increases, it becomes increasingly important to utilize quantitative models to guide circuit construction effectively, but previous efforts have been hindered by lack of accuracy in predictions of circuit behavior [13, 10]. To address this shortcoming, we have developed Empirical Quantitative Incremental Prediction (EQuIP), a new method for accurate prediction of genetic regulatory network behavior. EQuIP predictions are based on a composable black-box model derived solely from empirical observations of steady-state and dynamic behavior. We have validated the precision of EQuIP predictions for six transcriptional cascades in transiently transfected mammalian cells. These cascades exhibit over 1,000-fold cell-cell variation in fluorescence, yet EQuIP's predictions have a mean error of only 1.6-fold compared to experimental data. Such accurate predictions will foster reliable forward engineering of complex biological circuits from libraries of standardized devices.

2. EMPIRICAL PRODUCTION/LOSS MODEL

Effective construction of genetic circuits in synthetic biology involves choosing parts that combine to produce a desired behavior—an task that rapidly increases in importance and complexity for larger circuits. Current biological circuit models often explicitly encode all the biochemical reactions in the system believed to be significant (e.g., [11, 1, 9]), but their accuracy is impaired by difficult to estimate parameters and inadequate incorporation of cellular context effects. The main alternative, black-box abstractions, have been frequently explored (e.g., [7, 17, 14, 3, 12]), but until now there has been no black-box model that successfully predicted the experimental behavior of a multi-gene circuit.

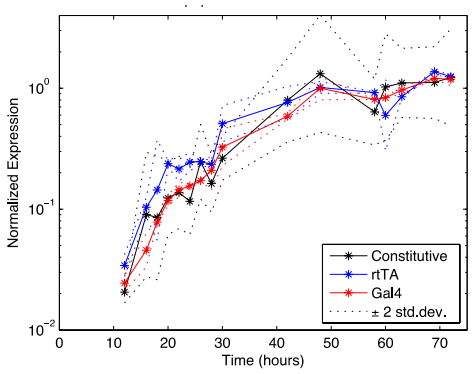
Our EQuIP black box model improves on prior approaches by combining three key ingredients: an appropriate level of device abstraction, obtaining detailed and internally validated characterization data, and an effective framework for modeling circuit behavior from part characterization. In our abstraction, a device encapsulates a small set of regulatory interactions that transduce input values into corresponding outputs. For each device, we collect empirical observations of expression dynamics and device steady-state behavior within the appropriate cellular context. This information is then used to create a composable device model with commensurate units describing the relationship between input levels, the circuit (i.e., plasmid) copy number, and the corresponding production and loss rates for the output biological species. EQuIP thus abstracts away biochemical details and captures significant interactions with the cellular context without requiring either explicit or well-understood models of those interactions. Using a piecewise linear approximation of input/output transfer curves, rather than making strong biochemical assumptions (e.g., using Hill kinetics), allows us to build a model that includes the effects of interactions with the cell even when an explicit model is not available.

EQuIP's device production and loss functions are derived from three sets of characterization data: fluorescent protein calibration and normalization, constitutive gene expression measured over time, and input/output transfer curves. The calibration and normalization data are an extended set of controls used to map each color's fluorescence measurements to absolute MEFL units [15] and to compensate for systematic variation between data sets, following the calibration protocols that we have previously laid out in [5].

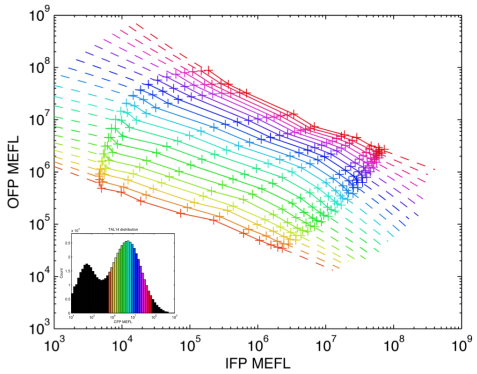
Fundamental production and loss dynamics for the system are determined from measurement of the time course of constutive and transcriptional activator driven expression. Data from a 72-hour experiment to determine these dynamics is shown in Figure 1(a). Our analysis of this model gives a mean behavior of cell division every 20 hours through 47 hours of expression, convolved across a variable initial delay of up to one division time. From this, we determine a loss function based on exponential dilution from cell growth (expected to dominate, given stable proteins) and a relative production function uniquely determined on an inverse model of the dilution of protein and plasmids over time.

This constitutive production model is transformed into a model of regulated production with the aid of an input/output transfer curve: a snapshot of the relationship between in-

Work partially 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 '13 London, UK



(a) Expression Time Series



(b) TAL14 Transfer Curve

Figure 1: EQuIP uses two sets of input data: expression time series (a) are used to quantify incremental fluorescence production and loss rates, and input/ouput transfer curves (b) are used to quantify regulatory relationships (dashed lines are projection; inset shows binning by constitutive fluorescence).

put and output concentrations at a particular time. We determine this relation using a device characterization circuit with three fluorescent reporter proteins that approximate input concentration (IFP), output concentration (OFP), and number of circuit copies in a cell (CFP). We estimate the relative circuit copy number for an individual cell based on the intensity of a fluorescent protein (CFP) constitutively expressed from one of the co-transfected plasmids [6]. To account for behavior variance due to variance plasmid copy number, we split the flow cytometry characterization of each sample into logarithmically partitioned subpopulations ("bins") based on CFP fluorescence. The mean expression level of each fluorescent protein is then computed independently for each subpopulation [4]. This allows us to diminish the impact of noise associated with single cell data while still capturing the differential behavior due to circuit copy number.

3. EXPERIMENTAL VALIDATION

To validate EQuIP, we gathered transfer curves for three hybrid repressor devices: TAL14, TAL21, and LmrA. Each experiment was performed with a co-transfection, following the characterization method in [5]: triplicate samples of

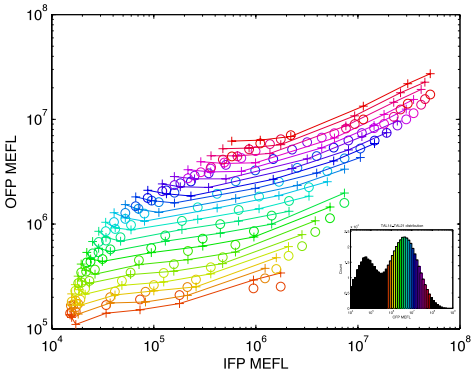


Figure 2: Experimental testing of two element cascades, such as this TAL14-TAL21 cascade, shows a good agreement between EQuIP predictions (circles) and observed experimental data (lines and crosses).

each circuit were induced with a series of Dox concentrations covering the full range of rtTA3 activation, and fluorescence was measured using flow cytometry at $t = 72$ hours post-transfection. Figure 1(b) shows the transfer curve for TAL14, segmented by circuit copy number, with the segmentation indicated on the inset histogram of constitutive expression.

We then used these data to predict the behavior of all two-repressor cascades comprising TAL14, TAL21, and LmrA, and compared these predictions to experimental results. The observed mean output levels for different input/plasmid combinations have more than a 1,000-fold range for each of the six cascades across the different cell sub-populations; over this wide range, the mean error of predicted versus observed output is only 1.6-fold for all six transcriptional cascades measured. Figure 2 shows one example of these six: the 72 hour input/output predictions vs. experimental data for the TAL14-TAL21 cascade. EQuIP is also effective at predicting which combinations of repressors are best matched to provide the greatest differential expression between fully-induced and un-induced states: although the induction predictions from EQuIP are conservative (they under predict the gain), they correctly identify the TAL14-TAL21 and TAL21-TAL14 cascades as having significantly stronger gain than all cascades involving LmrA.

4. FUTURE DIRECTIONS

The ability to make accurate predictions without detailed biochemical models of genetic regulatory devices makes EQuIP a powerful abstraction, as it only uses readily available single-cell fluorescence data with minimal assumptions or indirect inference. As may be expected, predictions are less accurate for extreme expression levels and circuit copy numbers. Anticipated future work includes improving models and gathering additional data to allow prediction of there more extreme cases, validation of EQuIP on more complex circuits, application of predictions to improve regulatory protein design, and adaptation of EQuIP to other organisms. On the basis of such expansion, we anticipate that EQuIP may become a critical building block in a new era of exponential growth in our ability to engineer biological systems.

5. REFERENCES

[1] C. M. Agapakis and P. A. Silver. Synthetic biology: exploring and exploiting genetic modularity through the design of novel biological networks. *Molecular BioSystems*, 704, 2009.

[2] E. Andrianantoandro, S. Basu, D. K. Karig, and R. Weiss. Synthetic biology: new engineering rules for an emerging discipline. *Mol Syst Biol*, 2006.

[3] G. Batt, B. Yordanov, R. Weiss, and C. Belta. Robustness analysis and tuning of synthetic gene networks. *Bioinformatics*, 23:2415–2422, 2007.

[4] J. Beal, R. Weiss, D. Densmore, A. Adler, E. Appleton, J. Babb, S. Bhatia, N. Davidsohn, T. Haddock, J. Loyall, et al. An end-to-end workflow for engineering of biological networks from high-level specifications. *ACS Synthetic Biology*, 1(8):317–331, 2012.

[5] J. Beal, R. Weiss, F. Yaman, N. Davidsohn, and A. Adler. A method for fast, high-precision characterization of synthetic biology devices. Technical Report MIT-CSAIL-TR-2012-008, MIT, April 2012. Technical Report: MIT-CSAIL-TR-2012-008 http://hdl.handle.net/1721.1/69973.

[6] L. Bleris et al. Synthetic incoherent feedforward circuits show adaptation to the amount of their genetic template. *Molecular Systems Biology*, 7:519, 2011.

[7] B. Canton, A. Labno, and D. Endy. Refinement and standardization of synthetic biological parts and devices. *Nature Biotechnology*, 26(7):787–793, 2008.

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

[9] S. Hooshangi and R. Weiss. The effect of negative feedback on noise propagation in transcriptional gene networks. *Chaos*, 16, 2006.

[10] R. Kwok. Five hard truths for synthetic biology. *Nature News*, 463:288–290, 2010.

[11] T. S. Moon, C. Lou, A. Tamsir, B. C. Stanton, and C. A. Voigt. Genetic programs constructed from layered logic gates in single cells. *Nature*, 491:249–253, 2012.

[12] V. K. Mutalik et al. Precise and reliable gene expression via standard transcription and translation initiation elements. *Nature Methods*, 10:354–360, 2013.

[13] P. E. M. Purnick and R. Weiss. The second wave of synthetic biology: from modules to systems. *Nature Reviews Molecular Cell Biology*, 10(6):410–422, 2009.

[14] N. Rosenfeld, J. Young, U. Alon, P. Swain, and M. Elowitz. Accurate prediction of gene feedback circuit behavior from component properties. *Molecular Systems Biology*, 13(3):143, November 2007.

[15] Measuring molecules of equivalent fluorescein (mefl), pe (mepe) and rpe-cy5 (mepcy) using sphero rainbow calibration particles. Technical Report SpheroTechnical Notes: STN-9, Rev C 071398, SpheroTech, October 2001.

[16] B. Wang, R. Kitney, N. Joly, and M. Buck. Engineering modular and orthogonal genetic logic gates for robust digital-like synthetic biology. *Nature Communications*, 508:465–471, 2009.

[17] R. Weiss and S. Basu. The device physics of cellular logic gates. In *NSC1 The First Workshop of NonSilicon Computing*, pages 54–61, 2002.

Dynamical diversity of a programmable molecular oscillator in droplet microreactors

[Extended Abstract - Submitted for oral presentation]

Maximilian Weitz
Systems Biophysics and Bionanotechnology
Technische Universität München
85748 Garching, Germany
maximilian.weitz@tum.de

Jongmin Kim
Department of Bioengineering
California Institute of Technology
Pasadena, CA 91125
jongmin@dna.caltech.edu

Korbinian Kapsner
Systems Biophysics and Bionanotechnology
Technische Universität München
85748 Garching, Germany
korbinian.kapsner@ph.tum.de

Erik Winfree
Computation and Neural System
Computer Science
Bioengineering
California Institute of Technology
Pasadena, CA 91125
winfree@caltech.edu

Elisa Franco
Department of Mechanical Engineering
University of California, Riverside
Riverside, CA 92521
efranco@engr.ucr.edu

Friedrich C. Simmel
Systems Biophysics and Bionanotechnology
Technische Universität München
85748 Garching, Germany
simmel@tum.de

ABSTRACT

We study the emergence of diverse dynamic behaviors in a synthetic molecular oscillator encapsulated in microdroplets with volumes ranging from 33 fL to 16 pL. We observe strong variations in oscillation period and amplitude, which depend on the droplet size and on the operating point of the oscillator in its phase space. The observed variability is not attributed to stochastic reaction effects, but to partitioning of components arising during the emulsification process. This partitioning noise could be used for systematic experimental characterization of circuit function robustness with respect to parametric and initial conditions variations.

Keywords

Oscillator, emulsion, synthetic molecular circuits

INTRODUCTION: Emulsions are a powerful experimental platform: thousands of microscopic reactors can be generated with simple protocols, enabling the parallelization of otherwise lengthy and expensive serial bulk experiments [7]. In addition, encapsulation of species in small compartments favors partitioning variability, and therefore enhances the probability of diverse experimental outcomes. For example, emulsion-based techniques for selection of ribozymes or proteins rely specifically on this phenomenon [5]. Here we describe the compartmentalization of a programmable, synthetic molecular oscillator [1, 4] in a water-in-oil emulsion (Fig. 1 A), and we highlight the role of partitioning "noise" as a significant source of variability in the *dynamic* behavior of the system. While partitioning noise occurring in droplets for the purpose screening simple processes such as transcription/translation may be regarded as a disturbance, we believe that this phenomenon is a helpful characterization tool of an operating point for nonlinear, complex dynamical systems.

Our interest in oscillators stems from the fact that they are canonical signal generators in both biological [2] and computational systems, where they orchestrate and synchronize events. A variety of synthetic molecular oscillators have been built in the past decades [6], ranging from purely chemical systems (very robust, but not tunable), to *in vitro* biochemical clocks, to genetic clocks hosted in bacterial and mammalian cells (tunable, but extremely complex). We consider the transcriptional *in vitro* oscillator proposed in [4] (Fig. 1 B), whose architecture is inspired from *in vivo* genetic clocks but implemented using DNA molecular programming principles. This system offers an ideal compromise in terms of complexity and tunability, because it is built using a reduced number of components which could be easily scaled and expanded to improve performance.

RESULTS: For the oscillator in bulk solution we choose different operating points, yielding "sustained", "damped", and "strongly damped" traces. The bulk solution consists of a mixture of DNA, transcription buffer, and enzymes for transcription and degradation of components, and is used as the water phase in the generation of a water-in-oil emulsion. The oil phase is prepared using a non-ionic surfactant [3]. Emulsification is done by vortexing the sample. Droplets are observed on average for 20 hours under a microscope, tracking a fluorescent marker (Texas Red) used in our oscillator system [1, 4].

We observe a significant diversity of period and amplitude (Fig. 1 C), in particular for droplets with radius 2–10 μm. Interestingly, the "strongly damped" bulk solution results in a number of encapsulated oscillators that show sustained behaviors.

To explain our data, we build numerical models based on the reactions constituting the oscillator and on candidate stochastic parititioning processes. Because the concentrations of reagents in the bulk oscillator solution are in the order of 5–500 nM, stochasticity of the chemical kinetics is

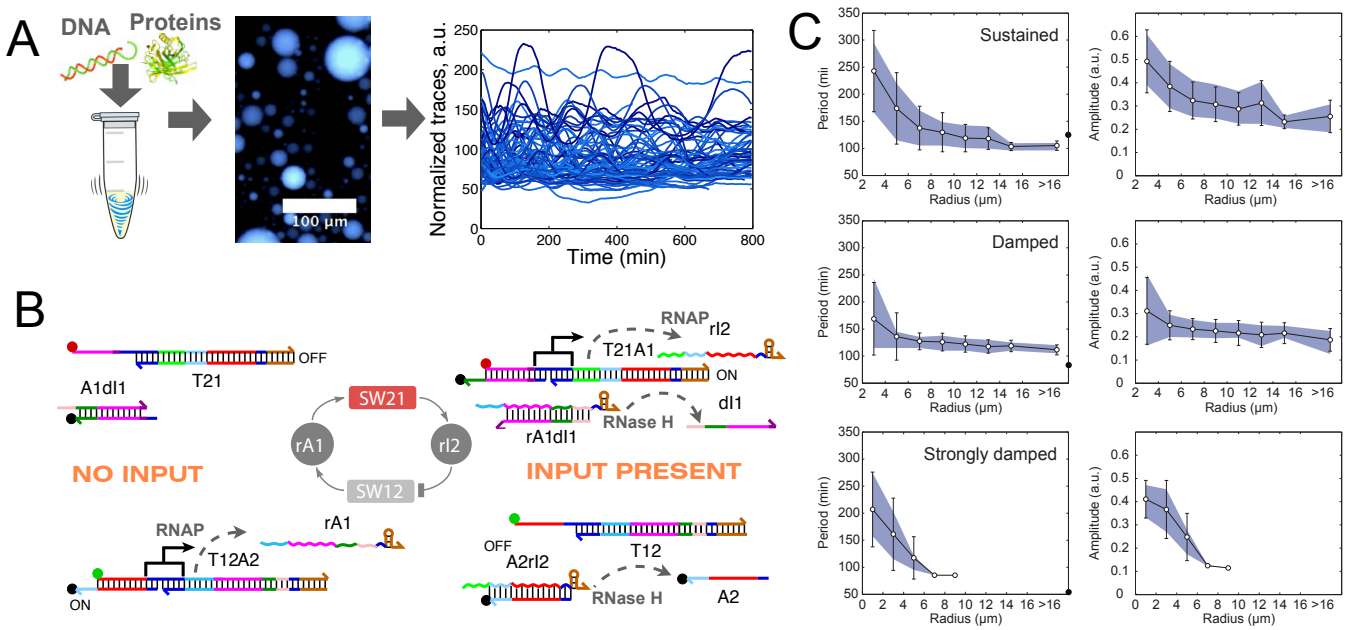


Figure 1: A: Scheme of our experimental method. B: Scheme of the compartmentalized oscillator [1, 4]. C: Period and amplitude diversity (for measured oscillatory traces only).

negligible. Thus, we examine the role of partitioning noise. Partitioning of reagents is expected to follow a Poisson distribution: thus, the mean and variance of the number of particles in a droplet should be equal to $\lambda = c_0 N_A V$, where c_0 is the bulk concentration, N_A is Avogadro's number, and V is the volume of the droplet. However, we find that this model does not explain the experimental amplitude and period variability. Control experiments estimating the concentration of fluorescently labeled DNA and proteins in vortexed emulsions suggest that partitioning causes a variance larger than the expected Poisson variance λ. Phenomenologically, we find that a Gamma distribution is a better fit for our partitioning process. A broader-than-Poisson variance, captured by the β parameter in the Gamma distribution, may be caused by a variety of processes, such as losses in the oil phase during vortexing, aggregation of components, and multiplicative noise introduced in the random breakage and coalescence of droplets. Numerical simulations and partitioning control experiments also support the hypothesis that enzyme losses are the most important phenomenon underlying the observed diversity. Two enzymes, T7 RNA polymerase and RNase H, are the "engines" of our molecular oscillator (Fig. 1 B); in particular, the concentration of RNase H in the bulk solution is low relative to other components, and thus partitioning effects would result in significant variations of number of RNase H molecules, thus affecting the degradation rates in the oscillator.

DISCUSSION: This abstract describes the successful encapsulation in a water-in-oil emulsion of a synthetic molecular oscillator, a complex programmable circuit exhibiting out-of-equilibrium dynamics. Emulsions generated by sample vortexing result in significant radius-dependent variability of oscillation frequency and amplitude, at each operating condition of the bulk system. Our models support the hypothesis that diversity can be explained by partitioning noise, which is presumably caused by a variety of factors such as loss and aggregation of components during the droplet formation process.

Partitioning noise may be simply considered an undesired disturbance in a variety of molecular/synthetic biology applications. In contrast, we believe that this source of randomness in initial conditions could be a useful tool to explore the robustness of a molecular dynamic circuit with respect to perturbations in their initial conditions. The development of robust, tunable clocks is central in several emerging fields (such as molecular nanofabrication, bioinspired energy and information storage, and biosensing), where these clocks will serve as timing devices for rationally designed complex, autonomous molecular operations.

1. REFERENCES

[1] E. Franco, E. Friedrichs, J. Kim, R. Jungmann, R. Murray, E. Winfree, and F. C. Simmel. Timing molecular motion and production with a synthetic transcriptional clock. *Proceedings of the National Academy of Sciences*, 108(40):E784–E793, 2011.

[2] A. Goldbeter, C. Gérard, D. Gonze, J.-C. Leloup, and G. Dupont. Systems biology of cellular rhythms. *FEBS Letters*, 586:2955–2965, 2012.

[3] C. Holtze, A. C. Rowat, J. J. Agresti, J. B. Hutchison, F. E. Angile, C. H. J. Schmitz, S. Koster, H. Duan, K. J. Humphry, R. A. Scanga, J. S. Johnson, D. Pisignano, and D. A. Weitz. Biocompatible surfactants for water-in-fluorocarbon emulsions. *Lab Chip*, 8:1632–1639, 2008.

[4] J. Kim and E. Winfree. Synthetic *in vitro* transcriptional oscillators. *Molecular Systems Biology*, 7:465, 2011.

[5] W.-C. Lu and A. D. Ellington. In vitro selection of proteins via emulsion compartments. *Methods*, (0), 2007.

[6] B. Novak and J. J. Tyson. Design principles of biochemical oscillators. *Nat Rev Mol Cell Biol*, 9(12):981–991, 2008.

[7] A. B. Theberge, F. Courtois, Y. Schaerli, M. Fischlechner, C. Abell, F. Hollfelder, and W. T. S. Huck. Microdroplets in microfluidics: An evolving platform for discoveries in chemistry and biology. *Angewandte Chemie-International Edition*, 49(34):5846–5868, 2010.

A 2-input Decoder Circuit in Human Cells

Michael Guinn
University of Texas at Dallas
Richardson, TX 75080
mtguinn@utdallas.edu

Yi Li
University of Texas at Dallas
Richardson, TX 75080
yxl121030@utdallas.edu

Leonidas Bleris
University of Texas at Dallas
Richardson, TX, USA
bleris@utdallas.edu

1. ABSTRACT

We present the implementation of combinatorial circuits, decoders, that convert binary information from N cellular inputs into 2^N unique outputs. As a proof-of-principle implementation we provide a 2-input architecture. To debug and optimize the device we use small molecules as inputs and fluorescent reporters as outputs. The experiments are carried out in human embryonic kidney cells and the characterization is performed using fluorescence microscopy and flow cytometry.

2. INTRODUCTION

Biomolecular computing systems [1] have the potential for a range of key technological and medical applications. Recent developments allow for the implementation of complex logic in cells [2]. Examples include circuits that respond to input stimulus using recombinase based mechanisms [3] and multilayered transcriptional and post transcriptional regulation circuits [4]. Multi-output architectures are equally important to multi-input [5], yet the former is an unsolved problem. As a means of enabling the use of multiple outputs with only few inputs, we propose the implementation of decoder architectures. Here we present the results of a novel combinatorial genetic architecture that can convert information from two chemical molecules into four observable responses in the form of fluorescent proteins (Figure 1). The circuit utilizes several transcription factors and synthetic microRNAs (details to be presented in the workshop). The logic of the decoder can be extended to implement more than 2 inputs. This decoder can be used as a modular component for higher-order circuits as well as serve as the interface for multi-output experiments.

3. EXPERIMENTS

The experiments were performed in a human embryonic kidney derivative cell line (Tet-3G ON). The circuit allows for conditional expression of one of three fluorescence proteins: YFP, CFP, or mKate. To validate the circuit, approximately 200K cells were plated per well, transfected 24 hours later, induced immediately following transfection, and grown an additional 48 hours for analysis using flow cytometry. Various combinations of the input ligands led to orthogonal states of the decoder. Each perspective well was induced with either no Y and no X, no Y and X, Y and no X, or Y and X immediately after transfection.

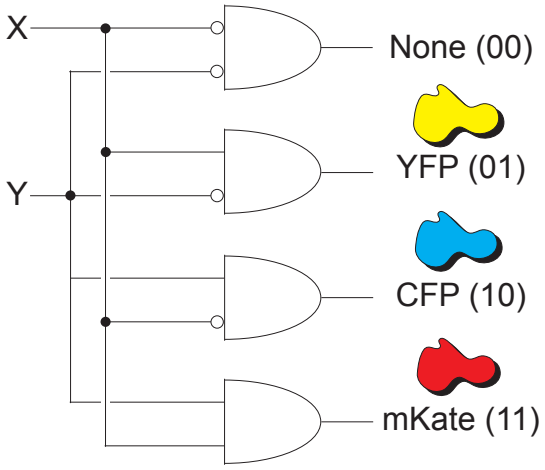


Figure 1: Boolean logic construction of a biological decoder with two inputs which are converted to fluorescent protein outputs.

4. RESULTS AND DISCUSSION

The aim of the circuit is to produce outputs which are completely orthogonal and singular per input. In the case of the fluorescence decoder this meant producing null fluorescence with no induction, producing YFP with input X induction, producing CFP with input Y induction, and producing mKate when the circuit is induced with both X and Y. The normalized microscopy images for each input (00, 01, 10, 11) are shown in Figure 2a. Each image contains the YFP, CFP, and mKate fluorescence fields overlapped, with the dominant response in the correct location (01 = YFP, 10 = CFP, 11 = mKate). Flow cytometry was performed to probe the population statistics. In all experiments performed, 100K events were collected. Measured fluorescence data was appropriately compensated using controlled experiments. A gate for each fluorescence protein was created that distinguished between transfected cells and untransfected cells (Figure 2b). Each scatter plot shows a various state of the decoder. We use integrated mean fluorescence intensity (iMFI) to report the population measurements. iMFI takes two quantities into consideration; the frequency of cells

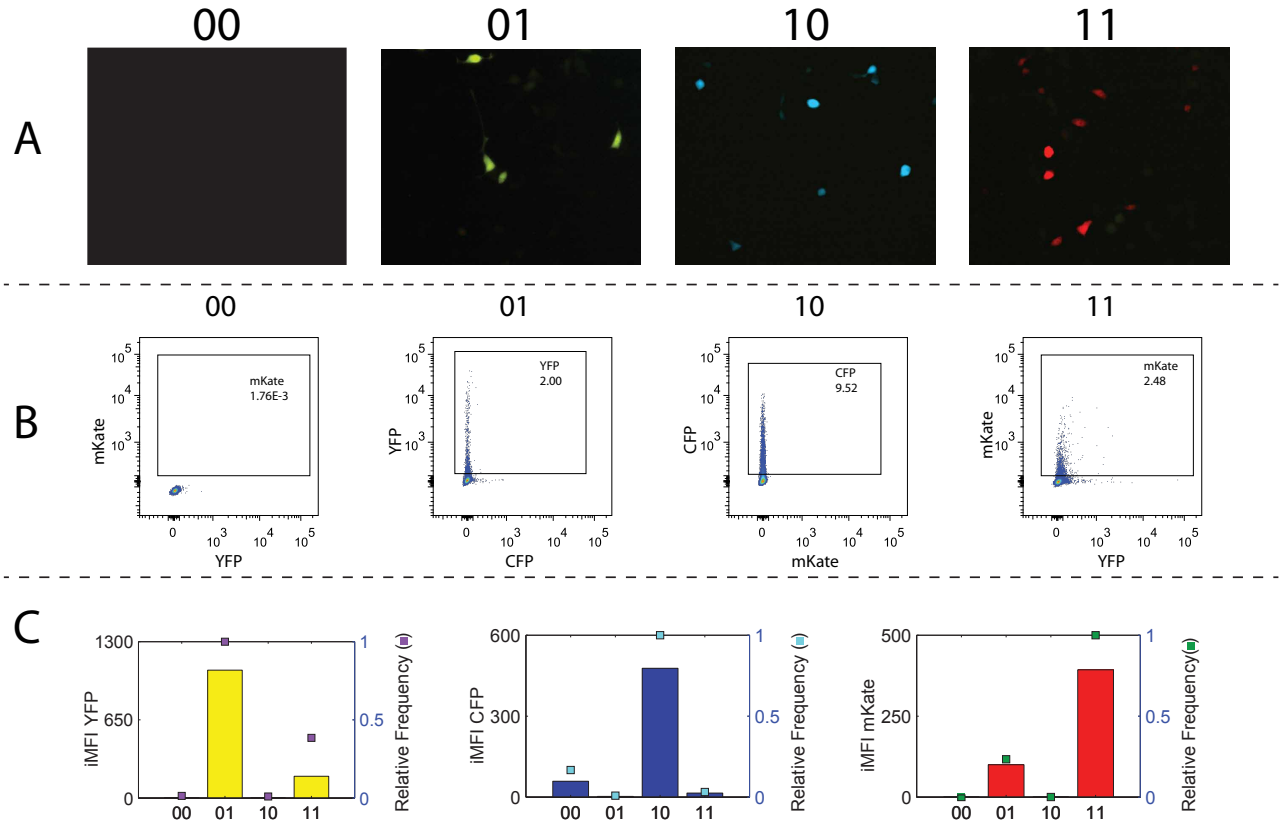


Figure 2: (a) Fluorescence microscopy of four possible outputs: 00, 01, 10, 11. (b) Flow cytometry scatter plots showing four possible states of the circuit. (c) Integrated mean fluorescence intensity (iMFI) of the cells transfected with the circuit under the different states.

within the transfected gate and the mean fluorescence intensity (MFI) of the cells within the gate. The frequency of the cells within the gate is the number of cells above the gate divided by the parent population (the number of cells alive) multiplied by one hundred. The iMFI is the product of the frequency and MFI and yields a more indicative representation of the cells with an operating circuit. The iMFI (Figure 2c) correlates with the microscopy images showing the highest iMFI values among the correct states (01 = YFP, 10 = CFP, 11 = mKate). The leakage observed can be attributed to the inefficiency of the multi-plasmid transfection (some cells receiving only a portion of the entire circuit). We are currently performing Gibson assembly and placing the entire circuit within one delivery vector. General purpose decoder circuits will allow us to apply combinatorial control in cells. A future objective of this work is to replace the fluorescence proteins within the architecture with proteins that interface with endogenous cellular pathways [6].

5. ACKNOWLEDGMENTS

This work was funded by the US National Health Institute (NIH) grant GM098984, the US National Science Foundation (NSF) grant CBNET-1105524, SRC/TxACE, and the University of Texas at Dallas. We would like to thank Taek Kang for comments and discussions.

6. REFERENCES

[1] Y. Benenson, "Biomolecular computing systems: principles, progress and potential.," *Nature Reviews Genetics*, vol. 13, pp. 445–468, 2012.

[2] T. Moon, C. Lou, A. Tamsir, B. C. Stanton, and C. Voigt, "Genetic programs constructed from layered logic gates in single cells.," *Nature*, vol. 491, pp. 249–253, 2012.

[3] P. Siuti, J. Yazbek, and T. Lu, "Synthetic circuits integrating logic and memory in living cells.," *Nature Biotechnology*, no. doi:10.1038/nbt.2510, 2013.

[4] K. Rinaudo, L. G. Bleris, R. Maddamsetti, S. Subramanian, R. Weiss, and Y. Benenson., "A universal rnai-based logic evaluator that operates in mammalian cells.," *Nature Biotechnology*, no. 25, pp. 795–801, 2007.

[5] Z. Xie, L. Wroblewska, L. Prochazka, R. Weiss, and Y. Benenson, "Multi-input rnai-based logic circuit for identification of specific cancer cells.," *Science*, vol. 333, pp. 1307–11, 2011.

[6] Y. Li, R. Moore, M. Guinn, and L. G. Bleris, "Transcription activator-like effector hybrids for conditional control and rewiring of chromosomal transgene expression.," *Scientific Reports*, vol. 2, no. 897, 2012.

Towards Industrial Strength BDA – A Short Essay

Wolf-Ekkehard Matzke
MINRES Technologies GmbH
Keltenhof 2
85579 Neubiberg
+49-17663374295
wolf@minres.com

Michael Stumpf
Imperial College London
317 Sir Ernst Chain Building
London SW7 2AZ
+44-20-75945114
m.stumpf@imperial.ac.uk

Thorsten Mascher
Ludwig-Maximilians-University Munich
Grosshaderner Str. 2-4
D-82152 Planegg-Martinsried
+49-89-218074622
mascher@bio.lmu.de

ABSTRACT
In this paper, we seek answers to the question "What does it take for bio-design automation (BDA) to become an industrial strength discipline?" Given the enabling role of BDA in synthetic biology, this central question needs to be addressed and answered. Our goal is to stimulate a community discussion around this question. To jump-start the debate, we will provide some core hypotheses covering what we believe are the most important aspects to be considered. Given that industrial strength is a composite aggregate of several variables, we have chosen to take a holistic approach and not restrict ourselves a priori to any particular viewpoints.

Keywords
Synthetic Biology, Bio-Design System, Bio-Design Automation, Rational Design, Industrial Strength, Stochasticity.

1. INTRODUCTION
Exploitation of the industrial potential of synthetic biology has only recently begun. Currently, we witness the progression of synthetic biology from an emerging technology to an emerging industry. However, it will take many years for synthetic biology to evolve into a fully established industry. To which extent and when these transitions will be completed depends on a multitude of challenges and constraints facing synthetic biology. The siliconization of synthetic biology – that is the degree to which in silico-based design penetrates synthetic biology – is amongst the fundamental technical challenges to be addressed. Siliconization is driven by the necessity to increase the productivity of design, which is in turn a requirement for the successful industrialization of synthetic biology. Eventually, in silico design has to become the principal design approach of synthetic biology. In silico design is the realm of bio- design automation (BDA). BDA's ultimate promise is to increase the productivity of synthetic biology. To fulfill this promise, BDA needs to develop industrial strength solutions. Industrial strength is defined as a system's ability to work capably and dependably in an operational setting [1]. In the remaining paper we will present what we believe are the four most important aspects to reach industrial strength for BDA. Note that the authors come from three different perspectives: BDA industry, BDA research, and synthetic biology.

2. IT'S THE BUSINESS MODEL, STUPID
"It's the economy, stupid" was the famous 1992 U.S. presidential campaign slogan of Bill Clinton that helped him get elected. The slogan was meant to emphasize the economy as the number one issue to be addressed. Paraphrasing this motto, the title of this section aims to draw attention to the fact that business models are of prime concern for industrial strength. It is worth mentioning that the intrinsic relation between industrial strength and business models matches most people's basic intuition. However, it might not be obvious to everyone in the BDA community why he or she should care about business models, which is in most cases a distant concept for engineers and researchers alike. After all engineers develop technical artifacts and managers are supposed to develop business models. The straightforward answer is that businesses are the foundation of any professional activity. The more specific answer is that BDA is in its early formative stage, where everyone has an important role in defining the trajectories for tomorrow's business models, knowingly or unknowingly, willingly or unwillingly. Today's dominant BDA business model is public funding. Clearly, this is an unsustainable model. We already witness much of the "free" software developed in public funding settings end up as internet-zombies. Nevertheless, the public funding model defines the starting point and direction of trajectories to future business models. But if we don't manage to develop sustainably business models, we might very well end up in an unproductive open source jungle. After all, the success or failure of BDA is a question of the right business models and their ability to adapt to changes.

3. THE RATIONAL DESIGN FALLACY
Synthetic biology is founded on the notion of rational design, which implies the following propositions [2]: (1) Design processes (what to do) and design methods (how to do it) are known explicitly and in detail; (2) Given a well-defined goal, the design process progresses along the optimal path on the basis of logical principles. Clearly, neither of these propositions will pass the reality test, which renders rational design a utopian vision. This is not a bad thing in itself, provided it is recognized for what it is: a guiding dream with positive transformative significance. However, if rational design is used as a mental model that directs practical actions [3], then it can turn straight into a fallacy. The two major pitfalls for BDA caused by this fallacy are over-automation and forward-engineering tyranny. The labeling of these pitfalls is meant to express the respective BDA experience of synthetic biologists. The over-automation pitfall occurs when the level of abstraction is raised beyond the essence horizon for the sake of automation: You cross the essence horizon if your abstractions alienate the substance of the system to be designed. The forward-engineering tyranny pitfall occurs when a unidirectional forward design process is enforced without any limitation. To avoid these pitfalls industrial strength BDA should adhere to the following principles and guidelines: (1) Keep in mind that all design is rationally bounded [4]; (2) Focus on the development of solutions for today's synthetic biology problems. Of course there also need to be investments in longer-term R&D efforts that may not mature for years; (3) Respect that the (baseline) design processes are established by synthetic biologists; (4) Support both forward and reverse engineering. Note, that in

today's synthetic biology settings, reverse-engineering dominates; (5) Identify the right (essence preserving) levels of computational abstractions along the design process; (6) Automate with care. Last but not least, the perhaps most important rule is this: (7) If your automations do not improve the BDA experience for synthetic biologists', start all over again!

4. THE BIO-DESIGN SYSTEM PARADIGM

As Martin et al. stated [5]: "If we wish to build systems that perform correctly and consistently, we will want to be clear, concise, and unambiguous in our specifications" and concepts. Note that all existing design automation industries, such as electronic design automation (EDA), architecture, engineering and construction (AEC), mechanical computer aided design (MCAD), etc. maintained their towers to Babel down to the present day, namely at the cost of productivity. The way to avoid the tower of Babel effect is to agree on a unifying framework within which the various resources and processes of design can be situated and their roles and relations can be identified. In Fig. 1 we propose such a hierarchical framework for a bio-design system. Note that BDA technology constitutes the in silico lab and BDA Engineering is part of design environment engineering.

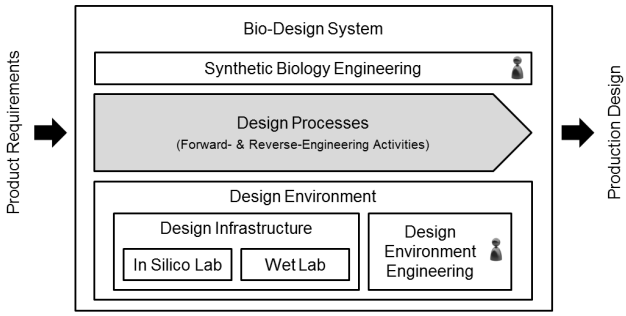


Figure 1. Bio-Design System Framework.

5. STOCHASTICITY RULES

This section emphasizes that BDA technology must be capable to cope with stochasticity. Generally, stochasticity is a second order effect in many industrial engineering design settings, and is mostly related to concerns such as product yield, product lifetime, etc. The opposite is true, although often not explicitly acknowledged, for synthetic biology. Quantum physics aside, the world is knowable and deterministic. It follows that stochasticity is due to ignorance of the underlying causes. Ignorance is currently ubiquitous in synthetic biology, and it does not seem that this situation will change in the near future. Not surprisingly, the prevailing sentiment in the world of synthetic biology is that the inherent complexity of the field largely prevents the use of computational design approaches. This position caused a considerable debate between the BDA and the synthetic biology community [6]. Unfortunately, discussants often seem to talk at cross-purposes when they might well be seeing the same problem from different angles. To arrive at some level of consensus among the communities consider the following: (1) If we interpret complexity as a function of our ignorance about the reality's work principles, then synthetic biology is devilishly complex; (2) Complex systems are inherently stochastic in nature. Both statements should be viewed as consensual propositions. The immediate conclusion is that stochasticity is an essential property of biological systems. Indeed, more recent research has shown that stochasticity is a generic feature of many biological systems, and may, in fact, be actively exploited by ensuring heterogeneity in a population of cells. Understanding the causes – in addition to the effects – of stochasticity is thus pivotal for applications in synthetic biology. Statistical inference is providing us with the computational tools necessary to dissect noise and stochasticity in biological systems; such a statistical angle is essential as we are no longer dealing with a single type of behavior but with a probability distribution over different (though related) types of behavior; this poses severe challenges to e.g. conventional optimization procedures. Let us consider the design of biochemical reaction network according to some stated design objectives. Let us further assume we have identified a set of potential designs, but we don't know the one that best meets the design objectives. Let us finally assume we have captured these designs by some appropriate modeling formalism. However, we face an additional challenge. In practice, it will be the rule rather than the exception that we have to cope with parameter uncertainty, often quite considerable (i.e. spanning orders of magnitude). The task is now to identify the design that best meets the design objectives (model selection) under conditions of parameter uncertainty (parameter inference). This requires an inference engine that is capable to handle model selection and parameter inference. Conceptually the inference engine needs to "sit" on top on whatever computational model we apply. We strongly believe that for any industrial strength BDA solution an inference engine is a prerequisite to cope with stochasticity. One superior inference engine (both in terms of conceptual and computational performance) is the ABC-SMC algorithm presented in [7]. The algorithm supports both forward- and reverse engineering and is flexible with respect to the underlying computational models. Perhaps the most important feature is that the algorithm can handle parameter uncertainty and model selection at once.

6. CONCLUSIONS

The four hypotheses of this essay aimed at highlighting the most important challenges for industrial strength BDA, at least in our view. There might be other important concerns that we have not touched. We ought to hear about them. We might have covered some aspects insufficiently or inaccurately to some reader. We want to learn about those. But most importantly, and in spite of these conceivable gaps, this essay will hopefully foster a community discussion about industrial strength BDA.

7. REFERENCES

[1] M. Rouse, "Industrial Strength," in WhatIs.com vol. 2013.

[2] D. L. Parnas and P. C. Clements, "A rational design process: How and why to fake it," Software Engineering, IEEE Transactions on, vol. SE-12, pp. 251-257, 1986.

[3] D. Gentner and A. L. Stevens, Eds., Mental Models. Hillsdale, NJ: Lawrence Erlbaum Associates, 1983.

[4] H. A. Simon, The sciences of the artificial: MIT press, 1996.

[5] J. Martin and J. J. Odell, Object-oriented methods (UML ed., 2nd ed.): a foundation: Prentice-Hall, Inc., 1998.

[6] D. Densmore, "Bio-Design Automation: Nobody Said It Would Be Easy," ACS Synthetic Biology, vol. 1, pp. 296-296, 2012/08/17 2012.

[7] C. P. Barnes, D. Silk, X. Sheng, and M. P. H. Stumpf, "Bayesian design of synthetic biological systems," Proceedings of the National Academy of Sciences, vol. 108, pp. 15190-15195, September 13, 2011 2011.

RetroPath: Automated Pipeline for Embedded Metabolic Circuits.

Pablo Carbonell
Univ. Evry, CNRS, institute of Systems & Synthetic Biology, Évry, France
pablo.carbonell@issb.genopole.fr

Pierre Parutto
Univ. Evry, CNRS, institute of Systems & Synthetic Biology, Évry, France
pierre.parutto@issb.genopole.fr

Jean-Loup Faulon
Univ. Evry, CNRS, institute of Systems & Synthetic Biology, Évry, France
jean-loup.faulon@issb.genopole.fr

ABSTRACT

Metabolic circuits are a promising alternative to other conventional genetic circuits as modular parts implementing functionalities required for advanced synthetic biology applications. To date, metabolic design has been mainly focused on production circuits. Emergent applications such as smart therapeutics however require circuits that enable sensing and regulation. Here we present RetroPath, an automated pipeline for embedded metabolic circuits that explores the circuit design space from a given set of specifications and selects the best circuits to implement based on desired constraints. In that way, design and implementation of advanced applications in the metabolic space such as therapeutic circuits that autonomously diagnose and treat disease, are enabled and their optimization streamlined.

Keywords

synthetic biology, circuits, metabolic network

1. INTRODUCTION

Synthetic biology aims at translating the methods and techniques from engineering into biology in order to streamline the design and implementation of devices through standardized parts [5]. To that end, metabolic circuits stand out as a promising alternative to other proposed genetic circuits [6] for developing modular parts implementing most of the basic functionalities required in advanced synthetic biology applications. To date, metabolic design has been mainly focused on production circuits that perform synthesis of exogenous compound in chassis organisms [7, 1]. Beyond such well-established practices, emergent applications such as smart therapeutics require of circuits that enable sensing and regulation of compound delivery [8].

Addressing the challenge of such advanced metabolic circuit design requires of an appropriate modeling of metabolic processes that should enable the efficient computation of the design space associated with a desired circuit. For that purpose, we have recently proposed a fingerprint coding representation based on changes in atom bonding environments where enzymatic reactions are processed [3]. Here, we present RetroPath, an automated pipeline allowing the design and optimization of metabolic circuits upon a required set of specifications.

IWBDA '13 London UK

2. DETERMINING THE DESIGN SPACE

Metabolic circuits are basically composed of either one or a combination out of production, sensor and processing modules. Each module implements, in turn, one or several metabolic pathways inducing the desired function, as well as their interfaces with genetic elements. Altogether, the problem does not differ greatly from the problem of design space exploration often found in embedded systems [4]. The basic steps are the following: a) Define input, output and metabolic space; b) Define specifications of the desired circuit; c) Compute scope connecting input with output sets; d) Enumerate the circuit design space; and e) Explore the design space in order to find the optimal solution(s). We describe in the following briefly each of the steps.

2.1 Metabolic space definition

In order to delimitate our problem, the fraction of the chemical space that can be processed *in vivo* using natural and synthetic biology devices, **M**, needs to be determined. Toward that end, we have recently addressed the issue of complexity that has been hindering any efficient full computational description of the metabolic space by proposing a solution based on extended connectivity fingerprints (ECFPs) which code for changes in the atom bonding environments where the reactions are taking place [3]. Moreover, by combining such modeling framework with the information from biological databases, the metabolome of the chassis organism **S** is also defined.

2.2 Circuit specifications

Metabolic circuit specifications define the problem that needs to be solved in the metabolic space. Although several types of metabolic circuits with different functionalities can be conceived, all of them conceptually belong to the design space of solutions to the general problem of finding the metabolic scope between a given source **S** and the target set of metabolites **T**. Figure 1 outlines typical input/output specifications. Moreover, other specifications can be envisaged such as number maximum of involved enzymes or desired intermediates, which altogether constitute a set of structural constraints imposed to the solutions.

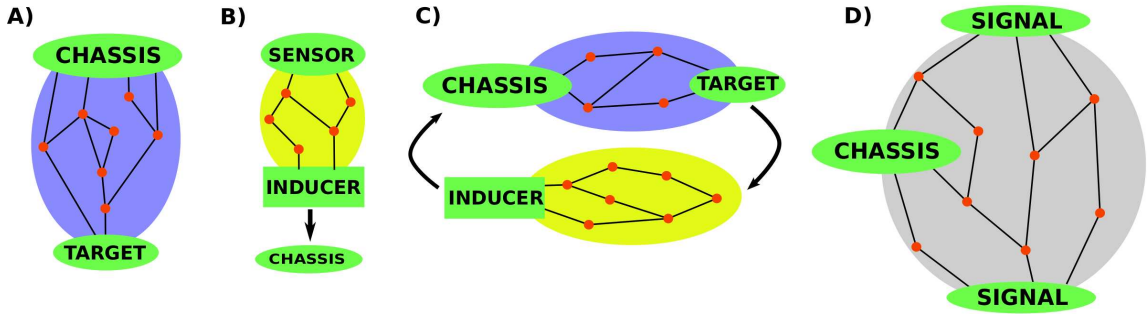


Figure 1: Metabolic circuit I/O specifications for: A) Production; B) Sensing; C) Regulation; D) Logics.

2.3 Computing the circuit scope

The circuit scope is the solution to a problem that can be stated in the following way: given an initial set of source metabolites **S** and a final set of target metabolites **T**, find the set of enzymes that are at least involved in one minimal pathway connecting elements of **T** to the source **S**, i.e. the scope should contain only enzymes that are at least essential for establishing one of the metabolic pathways. We have recently shown that the scope problem can be appropriately approached through the hypergraph formalism and the retrosynthesis technique [2].

2.4 Circuit enumeration

Enumerating metabolic pathways connecting a source to a target is a computationally complex problem which we have recently address [2] through different approaches such as elementary mode or topological analyses. Such approach, originally proposed for biosynthetic pathways, can be seamlessly extended to the general problem of enumerating the metabolic circuits C that are solutions to the problems shown in Figure 1 in their corresponding metabolic scopes.

2.5 Circuit optimization

Metabolic circuits are often multi-step processes involving many enzymes whose activities have to be controlled for optimal performance. To that end, a circuit cost function is defined $J(C)$ that typically is defined as an aggregate function of individual costs of elements in the circuit, i.e. enzymes and metabolites, as well as topological properties of the network, such as maximum expected flux, cross-talk, redox balance, etc.

3. RESULTS

This methodology was applied to the KEGG and MetaCyc databases. We found 2427 heterologous compounds that could be connected through production circuits to the *E.coli* chassis. Approximately 15% of the cases contained more than 10 circuits, and 3% more than 10^2. Similarly, metabolic sensor circuits that transform biomarkers through heterologous transformations into a metabolite that can induce the expression of a desired gene were computed in the network. Through the application of retrosynthesis, we found a number in excess of 105 metabolic circuits connecting 7 transcription factor ligands to a list of 120 biomarkers of human diseases showing the potential of this approach to the design of biosensors.

4. CONCLUSIONS

As metabolic engineering and synthetic biology are progressing and more chemicals are being produced and are possible to be detected in chassis organisms, it is critical to rationalize the design-construction-characterization process. Here, we introduced RetroPath, an automated pipeline for designing embedded metabolic circuits into chassis organisms, a tool that should pave the road toward the rationalization of metabolic circuits development for advanced synthetic biology applications.

5. ACKNOWLEDGMENTS

Genopole®through an ATIGE grant; abSYNTH facilities; and ANR through a Chair of Excellence.

6. REFERENCES

[1] P. K. Ajikumar, W.-H. Xiao, K. E. J. Tyo, et al. Isoprenoid pathway optimization for taxol precursor overproduction in escherichia coli. *Science*, 330:70–74, 2010.

[2] P. Carbonell, D. Fichera, S. Pandit, and J. L. Faulon. Enumerating metabolic pathways for the production of heterologous target chemicals in chassis organisms. *BMC Systems Biology*, 6:10+, 2012.

[3] P. Carbonell, A.-G. Planson, D. Fichera, and J.-L. Faulon. A retrosynthetic biology approach to metabolic pathway design for therapeutic production. *BMC Sys Biol*, 5(1):122+, 2011.

[4] E. Kang, E. Jackson, and W. Schulte. An approach for effective design space exploration. In *Proc. 16th Monterey Conf. Found. of Comp. Soft.*, FOCS'10, pages 33–54, Berlin, Heidelberg, 2011. Springer-Verlag.

[5] A. S. Khalil and J. J. Collins. Synthetic biology: applications come of age. *Nat Rev Genet*, 11:367–379, 2010.

[6] N. Nandagopal and M. B. Elowitz. Synthetic biology: Integrated gene circuits. *Science*, 333:1244–1248, 2011.

[7] C. J. Paddon, P. J. Westfall, D. J. Pitera, et al. High-level semi-synthetic production of the potent antimalarial artemisinin. *Nature*, advance online publication, Apr. 2013.

[8] N. Saeidi, C. K. K. Wong, T.-M. M. Lo, et al. Engineering microbes to sense and eradicate pseudomonas aeruginosa, a human pathogen. *Mol Syst Biol*, 7, 2011.

Multi-objective Optimisation of Escherichia coli for Direct Production of 1,4-butanediol

Jole Costanza
Dept. of Maths & Computer Science
University of Catania - Italy
costanza@dmi.unict.it

Claudio Angione
Computer Laboratory
University of Cambridge - UK
claudio.angione@cl.cam.ac.uk

Giovanni Carapezza
Computer Laboratory
University of Cambridge - UK
giovanni.carapezza@gmail.com

Pietro Lió
Computer Laboratory
University of Cambridge - UK
pietro.lio@cl.cam.ac.uk

Giuseppe Nicosia
Dept. of Maths & Computer Science
University of Catania - Italy
nicosia@dmi.unict.it

In the last 20 years, using of microorganisms, microbes and algae for the synthesis of natural or synthetic substances is increased and revealed crucial for industrial, biotechnological and natural processes. The synthesis of pharmaceutical molecules is usually very costly, therefore their production by means of genetic and metabolic engineering of microorganisms could be cost effective [2]. Additionally, the decrease of oil reserves and the request of new sources of energy have moved the interest of many researchers to the study of biofuels production techniques and other petroleum-derived chemicals from biological systems. Through metabolic engineering, many strains have been systematically designed for producing natural and synthetic products. Synthetic products are obtained by inserting in an organism enzymes that naturally the organism does not own. In the recent work of Yim et al. [4], the authors have engineered the Escherichia coli in order to produce 1,4-butanediol (BDO), a chemical compound industrially used and not produced naturally by organisms. BDO is currently manufactured entirely from petroleum-based feedstocks such as acetylene, butane, propylene and butadiene.

In an in silico step, strains are designed by computational techniques and successively realised in vitro. Genome-scale metabolic networks are useful to investigate the capability of organisms, since they include information about metabolite interactions, reactions, genes and enzymes control.

In this work, we seek genetic strategies in order to maximise BDO production from the genome-scale metabolic network of *E. coli* composed of 904 genes and 941 unique biochemical reactions, used in Yim et al. [4]. Genetic strategy stands for knockout manipulation, i.e., turning off one or more genes in order to block reactions. Searching for the optimal knockout set is a hard problem, since an organism such as a bacterium, contains more than 4000 genes. Importantly, a knockout strategy must also guarantee the survival of the organism. By using Pareto optimality and multi-objective optimisation, we simultaneously ensure the biomass formation and find several solutions that represent trade-offs [1]. Knockout manipulations are represented in the model by means of a Boolean string y, where the lth element is 1 if the corresponding gene set is turned off. The aim is to find the optimal Boolean string y that maximises or minimises simultaneously two or more biological functions (in this case, BDO production and biomass formation).

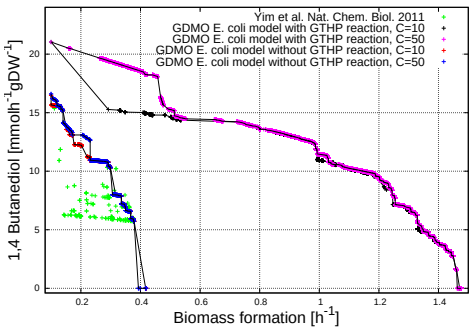


Figure 1: Pareto fronts obtained to maximise BDO production and biomass formation in the *E. coli* metabolic network used in [4]. In black, purple, red and blue the results obtained by our algorithm, and in green the results published in [4]. C is a parameter of GDMO algorithm and represents the maximum number of knockout allowable.

We compare our optimisation solutions with those published by Yim et al. [4]. The authors of Yim et al. [4] use the *E. coli* model containing 941 reactions, but they exclude by default the reaction *GTP amine hydrolysis* (GTHP). We conduce the experiment considering both the original model (with the reaction GTHP) and that without the reaction. We implement the method in matlab and adopt the algorithm published in [1] by using two values of the parameter C, C=10 and C=50. C indicates the maximum knockout number allowable. The results of BDO and biomass maximisation are shown in Figure 1. The front in black, purple, red and blue represent the results of our search, and in green we plot the solutions proposed by the authors of Yim et al. [4]. Each point represents an optimal *E. coli* strain, designed to maximise BDO. Between solutions in green, the authors of Yim et al. [4] propose that with a BDO equal to 15.166 mmol/h/gdW and biomass 0.140 1/h with a knockout cost

IWBDA 2013 July 12-13, London - UK

(kcost) 8.

Our algorithm finds several suitable solutions. By considering the original model and the Pareto in black, we select the point that reaches BDO = 13.132 mmol/h/gdW and biomass = 0.889 1/h with a kcost = 5. Moreover, with a kcost = 8, we find BDO = 20.488 mmol/h/gdW and biomass = 0.163 1/h, incrementing the BDO amount of +35% with respect to [4] (Pareto front in purple). Additionally, by deleting the reaction GTHP we find the same solution of Yim et al. (BDO = 15.169 mmol/h/gdW and biomass = 0.140 1/h) with a kcost equal to 6 instead of 8. With a kcost = 7 we find BDO = 15.520 mmol/h/gdW and biomass = 0.132 1/h, and with kcost=9 BDO = 20.490 mmol/h/gdW and biomass = 0.162 1/h.

We observe that by fixing C=50, the Pareto front on the left section owns better solutions in terms of BDO with respect to Pareto front obtained with C=10. Additionally, some green points overpass Pareto fronts in red and blue on the right section. The method of Yim et al. [4] is based on an algorithm that does not consider gene sets and the Gene-Protein-Reaction (GPR) map, but turns off the fluxes through the reactions. Therefore, these solutions are obtained turning off reactions that in our analysis are not linked to any gene. On the contrary, our method considers gene sets network and implements GPR relationship.

To consider more genes, we also take into account of the most recent genome-scale metabolic network of *E. coli* [3]. We have added the synthetic pathway composed of 10 new reactions that forms BDO, starting form Succinyl-CoA and 2-Oxoglutarate, produced naturally in *E. coli* metabolism. The model is composed of 1366 genes, 1041 gene sets, 1136 metabolites and 2261 reactions. By maintaining the same environment conditions used by Yim et al. [4] (a glucose uptake rate equal to 20 mmol/h/gdW), we maximise BDO production and biomass formation in anaerobic and aerobic conditions. With a knockout cost equal to 6, we obtain an overproduction of +662.669% of BDO (from 1.425 mmol/h/gdW to 10.869 mmol/h/gdW). This genetic strategy represents the best trade off design. The genes knocked out are: (i) the set (PflBec) or (TdcEec) or (PflDec) or (PflBec and YfiD) belonging to the Pyruvate Metabolism (pyruvate formate lyase reaction), (ii) the gene YgiN belonging to the Oxidative Phosphorylation (quinol monooxygenase reactions) and (ii) the gene Tpi belonging to the Glycolysis/Gluconeogenesis (triose-phosphate isomerase reaction). The maximum BDO production is reached with a knockout cost equal to 11, providing 10.933 mmol/h/gdW BDO production and 0.133 h^{-1} biomass formation. In Table 1, we report the maximum BDO production obtained with knockout costs from 2 to 8.

Genetic strategies search is a combinatorial optimisation problem, since the aim is to find the optimal binary string of 1041 values. Our optimisation algorithm is also able to handle real variables. Figure 3 shows the evolution of BDO and biomass maximisation during the optimisation of the environment conditions. In blue the dominated solutions, and in black the final Pareto front. Environment conditions are represented by 330 uptake rate fluxes. In this experiment, we maintain glucose and oxygen uptake rates respectively fixed at 20 mmol/h/gdW and 0 mmol/h/gdW. Changing uptake rates is a crucial element and can significantly improve the production.

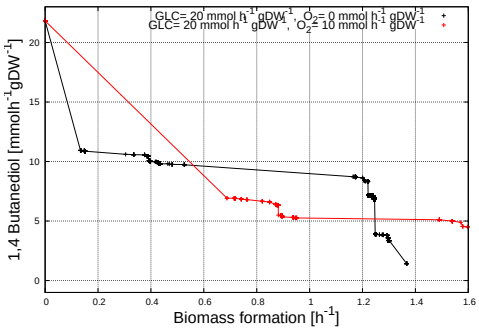


Figure 2: Genetic strategies optimisation to overproduce BDO in *E. coli* metabolic network [3].

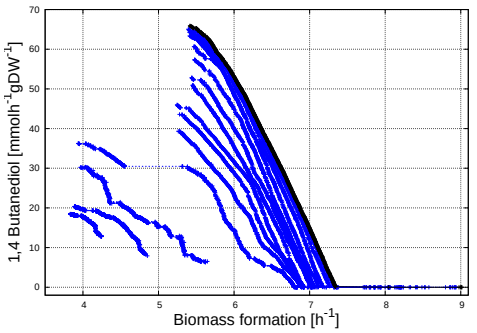


Figure 3: Optimisation of environment conditions for maximising BDO production and biomass formation in *E. coli* metabolic network [3].

BDO [mmol/h/gdW]	Biomass [h^{-1}]	kcost
1.425	1.367	wild type
3.358	1.301	2
3.367	1.298	3
6.809	1.246	4
8.319	1.220	5
10.869	0.151	6
10.922	0.137	7
10.923	0.136	8

Table 1: Minimum knockout cost and corresponding solutions obtained from the Pareto front of Figure 2 in black.

1. REFERENCES

[1] J. Costanza et al. Robust design of microbial strains. *Bioinformatics*, 28(23):3097–3104, 2012.

[2] J. Keasling. Manufacturing molecules through metabolic engineering. *Science*, 330(6009):1355 – 1358, 2010.

[3] J. D. Orth et al. A comprehensive genome-scale reconstruction of Escherichia coli metabolism-2011. *Molecular Systems Biology*, 7:53+, Oct. 2011.

[4] H. R. Yim et al. Metabolic engineering of escherichia coli for direct production of 1,4-butanediol. *Nature Chemical Biology*, 7(7):445–452, 2011.

j5 and DeviceEditor: DNA assembly design automation

Joanna Chen, Rafael Rosengarten, Douglas Densmore, Timothy Ham, Jay Keasling and Nathan Hillson

We have developed two on-line software tools, j5 and DeviceEditor, that automate and visualize the design of combinatorial and non-combinatorial, sequence agnostic, scar-less, multi-part assembly methodologies. Together, the tools offer a visual canvas for spatially arranging abstractions of genetic components; provide automated oligo, direct synthesis, and cost-optimal assembly process design; and integrate with liquid-handling robotic platforms to set up the PCR and multi-part assembly reactions. This work reduces the time, effort, and cost of large scale cloning and assembly tasks, as well as enables research scales otherwise unfeasible without the assistance of computer-aided design tools and robotics.

Pareto-based Gene and Pathway Sieve for Biofuel Production

Jole Costanza
Dept. of Maths & Computer Science
University of Catania - Italy
costanza@dmi.unict.it

Claudio Angione
Computer Laboratory
University of Cambridge - UK
claudio.angione@cl.cam.ac.uk

Giovanni Carapezza
Computer Laboratory
University of Cambridge - UK
giovanni.carapezza@gmail.com

Pietro Lió
Computer Laboratory
University of Cambridge - UK
pietro.lio@cl.cam.ac.uk

Giuseppe Nicosia
Dept. of Maths & Computer Science
University of Catania - Italy
nicosia@dmi.unict.it

1. INTRODUCTION

Bacteria are the future efficient plants for a huge variety of products. Plants, algae and bacteria are engineered through genetic interventions in order to obtain biofuels [6]. In many experiments, the Escherichia coli bacterium has been widely studied to obtain substances suitable for producing biofuels by using new synthetic pathways or genetic manipulations [1, 5].

The designing of these organisms includes different steps; the first is the in silico design [2]. We focused our studies on this step and used the genome-scale metabolic network of Escherichia coli K-12 MG1655 [4] to search for genetic strategies and optimize several biological functions, such as succinate. In this case, succinate is a natural product formed by Escherichia coli, so the aim is to increase its production. Succinate is widely used as a diesel fuel additive, solvent and detergent builder [7].

Genetic strategy stands for knockout manipulations, i.e., turning off one or more genes in order to block reactions and specific pathways. Searching for the optimal knockout set is a hard problem, since an organism such as a bacterium contains more than 4000 genes. Additionally, knockout combinations change according to the objective to reach. For instance, when the objective is to maximize succinate production, the set of genes knocked out is different with respect to the genes set knocked out when the objective is to maximize acetate. The manual search for the optimal knockout gene sets requires a strong knowledge on molecular biology. Therefore, the use of computational techniques speeds the search by using mathematical models that mimic the behaviour of metabolic networks.

Importantly, a knockout strategy must also guarantee the survival of the organism. By using Pareto optimality and multi-objective optimisation [3], we ensure the biomass formation and find several solutions that represent trade-offs. Biotechnologists can choose among several *E. coli* strains, characterised by different knocked out genes. We obtained several suitable solutions. For instance, with a knockout-cost 8, in anaerobic conditions succinate production increases of +13659% (from 0.077 to 10.610 mmol h^{-1} gDW^{-1}) (Figure 1-A, in blue). 0.077 mmol h^{-1} gDW^{-1} is the succinate produced by wild type *E. coli*.

IWBDA 2013 July 12-13, London - UK

2. PARETO OPTIMAL GENE SETS

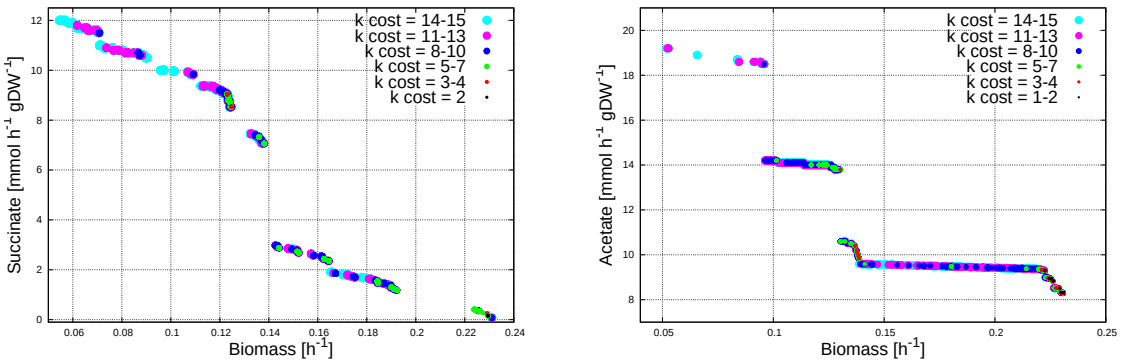
In Figure 1-A, we show the results of the knockout searching for maximising succinate and biomass. Solutions are represented by a two-dimensional Pareto front, and the axes report the two objective functions. The Pareto front is formed by a set of non-dominated (or Pareto) points. A point is non-dominated, if in a maximisation problem there does not exist a solution better in both the objectives. If an object increases, the other decreases. Pareto front methodology provides a powerful gene and pathway sieve towards multi objective biotechnological targets.

The maximisation of succinate production and biomass in *E. coli* provided a set of 2776 Pareto optimal solutions, i.e., 2776 strains designed to maximise succinate. Moreover, we are also able to select the trade-off, constructive, neutral and destructive strategies. A genetic strategy is called *trade-off* if improves at least one objective function, *constructive* if improves all the objective functions, *neutral* if leaves the objective functions unchanged, and *destructive* if worsens the amount of all the objective functions. In genetic strategies searching, constructive strategies do not exist, since turning off a gene carries always on two possible cases: (i) biomass decreases, (ii) biomass does not vary. Instead, destructive genetic strategies include essential genes, i.e., all the genes that are crucial for the biomass formation.

We found that with a knockout cost (*kcost*) equal to 4, succinate increases from 0.077 to 8.527 mmol h^{-1} gDW^{-1}. The kcost is the number of genes turned off in a genetic manipulation. Each gene set can be formed by one or more genes linked by a Boolean relationship. For example, when a reaction can be catalyzed by two isoenzymes, in the model they are represented by an OR relation. In this case, the kcost is equal to 2.

The trade off gene sets are: MhpF or AdhE, YdfG and DeoB. Biomass decreases of 46.03% with respect to wild type. Moreover, for this strain, 39 gene sets have been classified as neutral (e.g. Acs, YicP, CyaA). Instead, HemN is classified as destructive or essential gene.

We also performed the genetic knockout searching for ac-



A) B)

Figure 1: Pareto fronts obtained optimising succinate production in A and acetate production in B. Each point of the front represents a strain with a particular knockout manipulation. Size of signs increases as knockout cost.

etate production in anaerobic condition, with a glucose uptake rate equal to 10 mmol h^{-1} gDW^{-1}. Results are reported in Figure 1-B. By considering the Pareto point (0.1303; 13.7911) of the front in Figure 1-B that has a knockout cost 3 (YdfG; MhpF OR AdhE), we find that 88 gene sets can be considered neutral genetic strategies. Acetate production increases of +66.13% with respect to wild type (from 8.301 to 13.791 mmol h^{-1} gDW^{-1}, biomass decreases of 43.59%). Aas; rffT; AtoB; rfe; Acs; Lpd and SucA and SucB; Amn; and AdiA are just some of the neutral genetic strategies. MraY and murG genes are classified as essential.

Additionally, the gene tnaA is involved in 796 Pareto operations when acetate is maximised in anaerobic conditions. Moreover, this gene is also a most involved genetic manipulations in aerobic conditions (721 over 1000 strategies). tnaA enzyme catalyzes the Tryptophanase (L-tryptophan) reaction of Tyrosine, Tryptophan, and Phenylalanine Metabolism pathway. By cross comparing succinate maximisation and acetate maximisation experiments in anaerobic conditions, to increase simultaneously the production of succinate and acetate, we observed that a common point is the lock of the channel to transport potassium in, via proton symport. Another interesting result is that in anaerobic conditions the genes set "(SapD AND TrkA AND TrkH) OR (Kch) OR (SapD AND TrkA AND TrkG) OR (Kup)" is the most frequent both for maximising acetate and for maximising succinate. In Table 1, we report the most significant results. Our methodology and software provide an effective and efficient way to select best genes for specific biotechnological tasks and filter out those genes increasing both fitness and biotechnological targets.

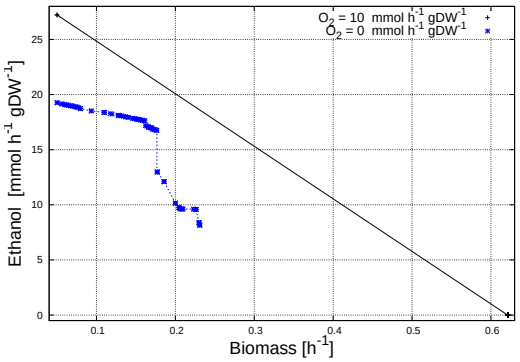


Figure 2: Pareto fronts obtained to maximise ethanol production and biomass formation in anaerobic (blue) and aerobic conditions (black).

Acetate	Anaerobic	Aerobic
tnaA	0.796	0.721
GuaB	0.728	0.310
SurE	0.568	0.077
Succinate	Anaerobic	Aerobic
Apt	0.950	0.510
DeoD	0.610	0.230
DeoD or deoA	0.260	0.320

Table 1: Frequency of some gene sets in genetic strategies for acetate and succinate maximisation in the *E. coli* metabolic network [4]. The values reported in the second and third columns, between 0 and 1, represent the percentage of occurrences of the gene set with respect to the Pareto manipulations obtained with our method.

3. REFERENCES

[1] H. Alper et al. Identifying gene targets for the metabolic engineering of lycopene biosynthesis in escherichia coli. *Metabolic Engineering*, 7(3):155 – 164, 2005.

[2] H. Bernstein et al. Microbial Consortia Engineering for Cellular Factories: in vitro to in silico systems. *Computational and Structural Biotechnology Journal*, Oct. 2012.

[3] J. Costanza, G. Carapezza, C. Angione, P. Lió, and G. Nicosia. Robust design of microbial strains. *Bioinformatics*, 28(23):3097–3104, 2012.

[4] A. M. Feist et al. A genome-scale metabolic reconstruction for escherichia coli k-12 mg1655 that accounts for 1260 orfs and thermodynamic information. *Mol Syst Biol*, 3(121):291–301, June 2007.

[5] T. Liu et al. Genetic engineering of escherichia coli for

biofuel production. *Annual Review of Genetics*, 44(1):53–69, 2010. PMID: 20822440.

[6] Mukhopadhyay et al. Importance of systems biology in engineering microbes for biofuel production. *Current Opinion in Biotechnology*, 19(3):228 – 234, 2008.

[7] J. G. Zeikus et al. Biotechnology of succinic acid production and markets for derived industrial products. *Applied Microbiology and Biotechnology*, 51(5), 1999.

Load Capacity Improvements in DSD Systems Using Partially Open Feedback Control

[Extended Abstract]

Vishwesh V. Kulkarni*
University of Minnesota
Department of Elect. Engg.
Minneapolis, MN 55455
vvk215@gmail.com

Evgeny Kharisov
University of Illinois Urbana Champaign
Department of Aerospace Engineering
Urbana, IL 61801, USA
kharisov@illinois.edu

Naira Hovakimyan
University of Illinois Urbana Champaign
Department of Mechanical Science and Engineering
Urbana, IL 61801, USA
nhovakim@illinois.edu

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

Jongmin Kim
Caltech
Department of Bioengineering
Pasadena, CA 91225
jongmin@dna.caltech.edu

1. INTRODUCTION

DNA-based circuits relying on predictable thermodynamics and kinetics of DNA strand interactions impart flexibility in synthesizing synthetic biological constructs and in coupling these circuits to *in vivo* processes (see [1, 2, 4, 5] and references therein). Here, we focus on the synthetic Kim-Winfree oscillator network, illustrated in Fig. 1(i), which is a simple but effective coupled oscillator system in which two DNA switches SW1 and SW2 are coupled through activator and inhibitor blocks realized by RNA signals and auxiliary DNA species (see [3]). A typical experimental realization is *closed* in the sense that once the operation starts, we do not either add any chemicals, especially NTP fuel, externally into the wet-lab apparatus or remove any chemicals, especially waste products, from the apparatus. Within the closed system, the oscillations are bound to die out sooner or later — diminishing NTP fuel eventually stops supporting the production of RNA signals and accumulating waste products clog down the toeholds and, as a result, adversely affect the signal propagation. Furthermore, the oxidation effects and the pH variations tend to deactivate the enzymes. Loading poses an additional challenge since it increases the order and the uncertainty of the system — indeed, these oscillators have recently been used in [6] to drive conformational changes of a DNA nanomechanical device called DNA tweezers. These tweezers, comprised of two double-helical domains connected by a hinge, have two single-stranded hands that can bind to a targeted oligonucleotide sequences such as RNA signal species and auxiliary DNA species of oscillator, thereby closing the tweezers. As Fig. 1(ii) shows, the oscillator performance degrades sharply under loading. In [6], the challenge of reducing retroactivity from loading was addressed by inserting another DNA switch to amplify the signal to be propagated to the load process. However, this approach may still suffer from uncertain and time-varying loads. So, how should we improve the load capacity of such transcriptional devices? We answer this question by adopting a partially open architecture and by using a sophisticated *in silico* controller, a block diagram of which is illustrated in Fig. 2(i), which is to be coupled to the wet-lab apparatus. A light switching *in silico* controller, implementing a combination of a Kalman filter and a model predictive controller, was recently reported in [8]. Our controller is an $\mathcal{L}_1$ adaptive controller and uses a DNA/RNA strand based actuation that faciliates a much greater control channel bandwidth than the one provided by a light-based actuation.

2. MAIN RESULTS

We adopt a partially open architecture analogous to a microchemostat (see [7]) so as to inject control inputs without increasing the reaction volume. In this continuous flow reactor, DNA species, enzymes, and NTP fuel flow in at a low rate, while the outflow removes a portion of reaction mixture, keeping the accumulation of waste products in check. We choose the switch outputs of the main reaction chamber to be the state variables that are to track the desired periodic waveforms. The DNA switches are tagged with fluorophores and the auxiliary DNA activators, A1 and A2, are tagged by quenchers such that the binding of an activator to its target switch reduces fluorescence signal. The two switches, SW1 and SW2, have different fluorophores, allowing for a real-time measurement of the switch outputs as fluorescence signals. The target waveforms are generated internally as the reference signals in an *in silico* controller implemented

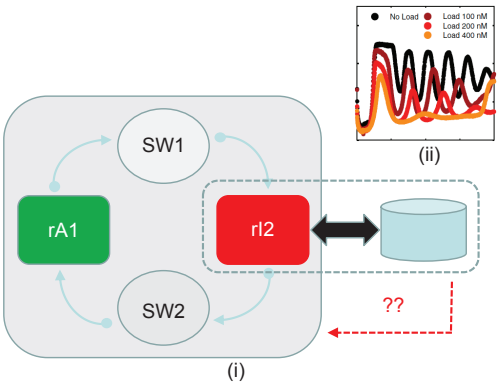


Figure 1: (i) The Kim-Winfree oscillator network comprises two switches (SW1 and SW2) connected through an activator rA_1 and an inhibitor rI_2 block. In [6], it is used to drive DNA tweezers. Due to the loading effects and a closed-system design, this network is unable to drive even moderate loads: the plot (ii), taken from [6], illustrates the loss of oscillations as the load increases from 0 nM to 400 nM. This highlights the need for more sophisticated controllers and for a more open design.

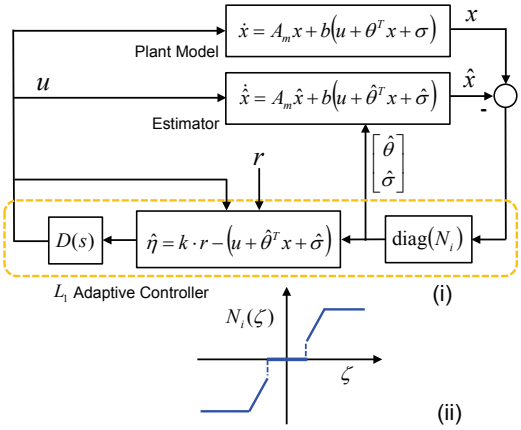


Figure 2: (i) Our $\mathcal{L}_1$ adaptive controller architecture. This controller is implemented *in silico* in a computer outside the wet-lab apparatus and the interfaced with the apparatus. Here, $D(s)$ is a bistable strictly proper low pass filter. (ii) Each of the diagnoal N_i nonlinearities is a combination of dead-zone and saturation nonlinearities.

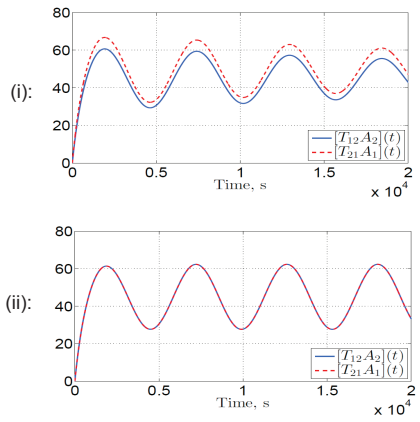


Figure 3: Simulation results for the 100 nM load. (i) When our $\mathcal{L}_1$ adaptive controller is turned off, the oscillations in the switch outputs die out within 10 cycles. (ii) When our $\mathcal{L}_1$ adaptive controller is turned on, the oscillations are sustained.

inside a computer — the commands of *in silico* controller controls the inflow of the continuous flow reactor in which the oscillator and the load are placed. When the discrepancy between the target waveform and switch outputs are detected, the controller calculates the required adjustments for the switching thresholds for the activation and inhibition signals, which in turn are set by the auxiliary DNA species dI1 and A2. Thus, adjusting the concentration of dI1 and A2 strands (and their complements for negative inputs) in the inflow allows the switch outputs to track reference signals. We characterize the expected disturbance and modeling uncertainty and then synthesize an $\mathcal{L}_1$ adaptive feedback controller to ensure that the oscillator network produces robust sustainable oscillations in which the switch states track the reference signals. As Fig. 3 illustrates, a significant improvement in the tunability and loading capacity of oscillator is obtained. This approach can easily be adopted to improve the robustness and the loading capacity of a wide range of wet-lab devices.

3. REFERENCES

[1] Zhang, D.Y. and Seelig, G.: 'Dynamic DNA Nanotechnology using Strand-Displacement Reactions', Nature Chemistry, 2011, 3, (2), pp. 103–113

[2] Padirac, A., Fujii, T., and Rondelez, Y.: 'Nucleic acids for the Rational Design of Reaction Circuits', Current Opinion in Biotechnology, 2012, http://dx.doi.org/10.1016/j.copbio.2012.11.011

[3] Kim, J. and Winfree, E.: 'Synthetic in vitro transcriptional oscillators', Molecular Systems Biology, 2011, 7, 465

[4] Chen, Y., and Smolke, C.: 'From DNA to Targeted Therapeutics: Bringing Synthetic Biology to the Clinic'. Science Translational Medicine, 2011, 3, 106

[5] Montagne, K., Plasson, R., Sakai, Y., Fujii, T., and Rondelez, Y.: 'Programming an *in vitro* DNA Oscillator using a Molecular Networking Strategy', Molecular Systems Biology, 2011, 7, 466

[6] Franco, E., Friedrichs, E., Kim, J., Jungmann, R., Murray, R., Winfree, E., Simmel, F.: 'Timing Molecular Motion and Production with a Synthetic Transcriptional Clock', PNAS, 2011, 108, E784-E793

[7] Balagaddé, F.K., You, L., Hansen, C.L., Arnold, F.H., Quake, S.R.: 'Long-term Monitoring of Bacteria Undergoing Programmed Population Control in a Microchemostat', Science, 2005, 309, 137-140

[8] Milias-Argeitis, A., Summers, S., Stewart-Ornstein, J., Zuleta, I., Pincus, D., El-Samad, H., Khammash, M., and Lygeros, J.: '*In silico* Feedback for *in vivo* Regulation of a Gene Expression Circuit', Nature Biotechnology, 2011, 29, 12, pp. 1114-1116

Species Minimization in Computation with Biochemical Reactions

Ruei-Yang Huang[1], De-An Huang[2], Hui-Ju Katherine Chiang[1,3], Jie-Hong R. Jiang[1], and François Fages[3]

[1]Graduate Institute of Electronics Engineering, National Taiwan University, Taipei 10617, Taiwan
[2]Academia Sinica, Taiwan
[3]EPI Contraintes, INRIA Paris-Rocquencourt, France

{b97901166@ntu.edu.tw, b97901010@ntu.edu.tw, b97901184@ntu.edu.tw, jhjiang@cc.ee.ntu.edu.tw, Francois.Fages@inria.fr}

Engineering biochemical reactions for computational purposes is a common pursue in synthetic biology [1]. In such design tasks, molecular species have to be carefully engineered to ensure modularity and orthogonality [3, 7], and are scarce resources. Minimizing the number of involved molecular species is crucial to accomplish a complex computation within a confined biochemical environment. This report aims at species minimization by reusing modular and regular reactions in an asynchronous time-multiplexed fashion. Our method enhances not only species utility, but also re-programmability and robustness in realizing various logic circuits. A case study demonstrates the ease of design in realizing general logic computation, and simulation confirms the feasibility and robustness of the proposed method.

The quest for deciphering nature's design principle of living organisms has been the primary subject in systems biology. This goal cannot be fully achieved unless human engineers can construct complex biological systems from scratch. A synthetic approach to biology aims to create complex systems bottom up from elementary components [1]. In such construction, a convenient way of modeling biological systems is from a computational aspect regarding system states in terms of molecular concentrations. Despite the intrinsic *hybrid* nature (involving both continuous and discrete state evolutions) of a biological system, the dominating *digital* design methodology of electronic circuits can be systematically carried to designing computation with biochemical reactions [2, 5].

Building upon our prior work [2], we address a fundamental limitation of biological circuit design. Unlike electronic circuit design, where a logic gate can be freely instantiated into several copies to compose a complex logic function, in a biological circuit every signal corresponds to a distinct molecular species, and every instantiation of a logic gate "consumes" molecular species. As molecular species have to be carefully engineered to ensure modularity and orthogonality for proper operation [3, 7], they are expensive resources. It raises an important question, how to minimize molecular species involved in achieving a target computational task.

Methods

We focus on logic computation, which is well studied in circuit synthesis, and without loss of generality assume a given (combinational) logic circuit consists of purely two-input NAND gates. (As a NAND gate is functionally complete, it can be used universally to construct arbitrary Boolean functions.) Moreover, following [2], we consider computation with biochemical reactions under the classical chemical kinetic (CCK) model for simplicity. (Our construct also works under the stochastic model.)

To minimize molecular species in realizing a given logic circuit, we intend to reuse a single NAND processor for all NAND operations using a time multiplexing strategy (a well-known technique in reconfigurable computing). The NAND operations are executed in a topological order according to the circuit structure. In order to fulfill the reuse of the NAND processor, we introduce a three-valued NAND operation with biochemical reactions for asynchronous handshaking without a synchronizing clock signal. The primary inputs and intermediately computed signals are properly placed and shifted in an asynchronous linear pipeline (consisting of asynchronous shift registers) for iterative NAND processing. For each NAND operation, its two inputs are taken from the first two positions of the pipeline. Once the two data tokens are consumed by the NAND processor, an output token is produced and/or placed in proper pipeline position(s) according to the state of a *program counter*, which tracks which NAND gate of the given logic circuit is under processing. The data tokens in the pipeline are asynchronously shifted forwards for the next NAND processing. When all the NAND gates of the given circuit are executed, a done signal signifies its environment to load to the pipeline next input data. The procedure repeats accordingly. The technicalities are detailed as follows.

Three-valued logic abstraction: In the conventional Boolean model of biochemical systems, high and low concentrations of a molecule represent logic values 1 and 0, respectively. We extend the binary domain to the three-valued abstraction $\mathbf{T}_\delta$ with respect to two parameters δ and ϵ for $\delta \gg \epsilon$ with

$$\mathbf{T}_\delta(X) = \begin{cases} 1, & \text{if } |[X] - 2\delta| < \epsilon \\ 0, & \text{if } |[X] - \delta| < \epsilon \\ \text{NULL}, & \text{if } [X] < \epsilon \end{cases}$$

where $[X]$ denotes the concentration of molecule X. In essence, this extension distinguishes NULL (denoting no data, with concentration ≈ 0), logic value 0 (with concentration $\approx \delta$), and logic value 1 (with concentration $\approx 2\delta$). It thus allows signal completion detection for asynchronous operation.

Register transfer: Let the linear pipeline consist of n register pairs in order $\mathbf{r}^0, \mathbf{r}^1, \cdots \mathbf{r}^n$, each corresponds to two inputs to be processed by the NAND processor. A register is essentially a molecule, whose concentration corresponds to the stored data. The NAND processor takes its inputs from $\mathbf{r}^0$. The data content of a register pair $\mathbf{r}^{i+1}$ is shifted asynchronously to $\mathbf{r}^i$ as long as $\mathbf{r}^i$ contains no data token.

The asynchronous register transfer is accomplished in the following steps to ensure correctness. In the following code (see [2] for detailed program constructs), let R_i represent the species of $\mathbf{r}^i$, let C_0 be a flag such that $[C_0] > \delta$ signifies it is safe to start register transfer from each R_{i+1} to its temporary buffer $R_{i+1_{\text{buf}}}$. Once such transfers are done for all i, it is safe to start transfer from $R_{i+1_{\text{buf}}}$ to R_i. The transfer activation condition of C_0 is controlled by the iterative NAND operation to be detailed. (Assume the code is under sequential execu-

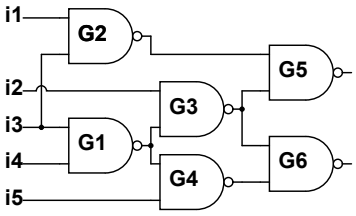


Figure 1: C17 circuit

tion, and reaction rates and *preconditions* [2] are omitted for brevity.)

begin
01 **if** $([C_0] > \epsilon)$
03 **while** ($\exists\ i$, $[R_{i+1}] > \epsilon$)
02 $R_{i+1} \to R_{i+1_{\text{buf}}}$ $\quad i = 0, 1, 2, ..., n$
04 $C_0 \to C_{1_{\text{buf}}}$
05 $C_{1_{\text{buf}}} \to C_1$
07 **while** ($\exists\ i$, $[R_{i+1_{\text{buf}}}] > \epsilon$)
06 $R_{i+1_{\text{buf}}} \to R_i$ $\quad i = 0, 1, 2, ..., n$
08 $C_1 \to C_{0_{\text{buf}}}$
end

Iterative NAND operation: An iteration of NAND operation, $Z = \text{NAND}(X, Y)$, can be described by the following reaction code. (Assume the code is under sequential execution, and reaction rates and preconditions are omitted.) The reactions can be divided into two phases. In the first phase, value of Z is computed; in the second phase, the value of Z is exported to proper locations of the pipeline or to the output, and register transfer is activated.

initial $[S] = \theta$ $(3\delta < \theta < 4\delta)$, $[B] = \delta$, $[A] = 0$, $[C_0] = 2\delta$
begin
01 **while** $[A] < \epsilon$
02 $S + X \to S_{\text{tmp}}$
03 $S + Y \to S_{\text{tmp}}$
04 **if** $([X] + [Y] > \theta)$
05 $B \to Z + A$
06 **else**
07 $B \to 2Z + A$
08 $X \to \emptyset$
09 $Y \to \emptyset$
10 $C_0 \to C_{1_{\text{buf}}}$
11 $C_{1_{\text{buf}}} \to C_1$
12 $Export(Z)$
13 $S_{\text{tmp}} \to S$
14 $C1 \to C_{0_{\text{buf}}}$
15 $C_{0_{\text{buf}}} \to C_0$
16 $A \to B$
end

Pipeline writing: When the output of the i^{th} NAND operation is an input of the later j^{th} NAND operation, it should be stored at register $\mathbf{r}^{j-i}$ because the input of the j^{th} NAND operation should arrive at $\mathbf{r}^0$ after further $j - i$ shiftings. Note that the writing positions may vary depending on the state of the program counter.

Case Study

We elaborate our method through the benchmark circuit C17, which consists of six NAND gates as shown in Figure 1. The gates $G_1, G_2, \ldots, G_6$ are in a topological order. Let species $P_1, P_2, \ldots, P_6$ serve as the program counter such that $[P_i] < \epsilon$ if and only if G_i is not under current processing. The following reactions are added to control pipeline writing.

$$\begin{cases} P_1 + Z \to P_1 + R_{2Y} + R_{3X}; & P_2 + Z \to P_2 + R_{3X}; \\ P_3 + Z \to P_3 + R_{2Y} + R_{3X}; & P_4 + Z \to P_4 + R_{2Y}; \\ P_5 + Z \to O_1; & P_6 + Z \to O_2; \end{cases}$$

where Z is the output of the NAND processor, R_{iX} (R_{iY}) is the X-end (Y-end) register of $\mathbf{r}^i$, and O_1 and O_2 correspond to the two outputs of C17. The ODE simulation result of SBW simulator [4] (under input assignments $i_1 = 0$, $i_2 = 0$, $i_3 = 0$, $i_4 = 0$, $i_5 = 0$, $i_6 = 0$) is plotted in Figure 2.

Analysis and Discussion

In our realization, a three-valued NAND gate operation mainly consists of 13 reactions and 11 species (excluding absence indicators and their generation reactions); similarly an asynchronous register transfer mainly consists of 12 reactions and 6 species. If a circuit consisting of n two-input NAND gates

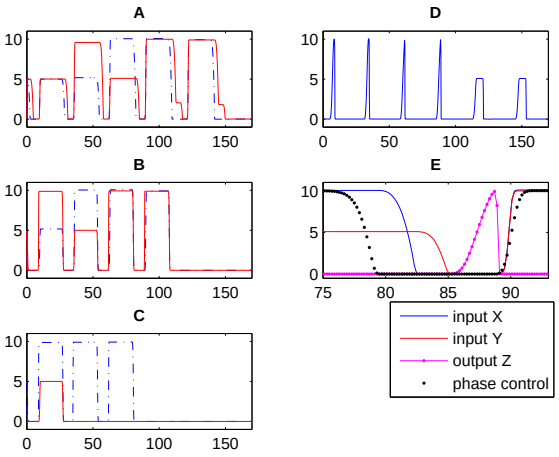


Figure 2: Simulation result: A. Waveforms for the two registers of $\mathbf{r}^0$; B. Waveforms for $\mathbf{r}^1$; C. Waveforms for $\mathbf{r}^2$; D. Waveform for $Z = NAND(X, Y)$; E. Detailed waveforms in the range of 75 ∼ 93 time units.

are "hardwired" directly, $13n$ reactions and $11n$ species are required (excluding absence indicators and their generation reactions). On the other hand, our proposed approach requires $12(n-1)+13$ reactions and $6(n-1)+11$ species. Although our approach may require longer computation time than hardwired implementation, in comparison species limitation is often a more critical issue than delay in biochemical systems.

The advantages of our proposed architecture are three-fold: First, it allows NAND operation to be reused and reduces the number of required molecular species. Second, the NAND processor and the registers always take the same species as input. The reaction structure is very regular, and the corresponding reactions are easy to engineer. Third, expanding the array of shift registers is straightforward, and the computation procedure can be easily re-programmed to implement different circuits. Moreover, as the proposed operations are all asynchronous, our method is robust against delay variations.

Conclusion

For universal logic computation, we proposed a three-valued NAND operation and a linear shift register architecture for asynchronous time multiplexing. Compared to hardwired logic netlists, our method reduces substantially the number of required species, provides insensitivity against delay variations, and admits re-programmability in realizing various logic circuits with minor modifications. For future chemical or biological realization, DNA strand displacement reactions [6] or intracellular genetic reactions [3] could be studied. Computation beyond the logic-gate formulation, e.g., by the dynamics of biochemical reactions, awaits future study.

REFERENCES

[1] E. Andrianantoandro, S. Basu, D. Karig, and R. Weiss. Synthetic biology: New engineering rules for an emerging discipline. *Molecular Systems Biology*, 2006.
[2] D.-A. Huang, J.-H. R. Jiang, R.-Y. Huang, C.-Y. Cheng. Compiling program control flows into biochemical reactions. In *Proc. Int'l Conf. on Computer-Aided Design* (ICCAD), pp. 361-368, 2012.
[3] T. S. Moon, C. Lou, A. Tamsir, B. Stanton, and C. Voigt. Genetic programs constructed from layered logic gates in signle cells. *Nature*, 491(11):249-253, 2012.
[4] Systems Biology Workbench (SBW): `http://sbw.sourceforge.net/`
[5] P. Senum and M. Riedel. Rate-independent constructs for chemical computation. *PLoS ONE*, 6(6), 2011.
[6] D. Soloveichik, G. Seelig, and E. Winfree. DNA as a universal substrate for chemical kinetics. *Proc. of the National Academy of Sciences*, 107(12):5393-5398, 2010.
[7] B. Wang, R. Kitney, N. Joly, and M. Buck. Engineering modular and orthogonal genetic logic gates for robust digital-like synthetic biology. *Nature Communications*, 2011.

Position-specific codon bias within an essential gene of E. coli

Eric Kelsic, Hattie Chung, Harris Wang and Roy Kishony

Variation of synonymous codons within a gene can strongly affect protein expression levels. Therefore, understanding which codons are optimal for a particular gene is of great importance for synthetic biology, for interpreting patterns of evolutionary conservation and for understanding gene regulation. Previous methods of codon optimization have focused on mimicking global patterns such as the average frequencies of codon usage across the genome, but with limited success. At present, we still lack a detailed understanding of which codons are optimal at varying positions within a gene. Here we present a new method for systematically making single-codon variants of an essential gene in E. coli, and measuring the resulting change in growth rates. Mutant libraries were constructed using Multiplex Automated Genome Engineering (MAGE) transformations. Mutants were pooled together in a competition assay, and the change in frequency of all mutants over time was measured using next-generation sequencing. We systematically measured growth rates of more than 4500 single-codon variants across the entire length of a gene that is essential for initiating translation, infA (IF1). We found that many single-codon synonymous changes have a significant negative effect on growth rate. I will present our analysis of this dataset, describing patterns of global and position-specific codon bias at the level of an individual gene.

(notes: I am submitting this abstract for a poster presentation. I will also be presenting this poster at SB6.0, and the abstract above is the same as the one I submitted for SB6.0)

Development and application of CADMAD technology

Tuval Ben Yehezkel
Weizmann institute of science
Dept. of mathematics and computer science
Rehovot, Israel
972528681034
Tbenyeh@weizmann.ac.il

Ehud Shapiro
Weizmann institute of science
Dept. of mathematics and computer science
Rehovot, Israel
97289342125
Ehud.shapiro@weizmann.ac.il

ABSTRACT
In this paper, we describe the development of an enabling technology within the field of synthetic biology. Specifically, we describe the development of a system for Computer Aided Design, Manufacturing and Application of DNA (CADMAD) libraries for synthetic biology and bio-engineering in general.

The system reported here utilizes a DNA specification language we developed that help biologists specify complex DNA libraries and the uses this specification to automatically generate optimized robot control programs that produce these specified DNA libraries using liquid handling robots. These robots execute an enzymatic DNA assembly operation that we developed (the Y operation). Additionally, we will present our results with developing and applying next generation digital microfluidic technology for DNA editing. Finally, we will present biological results from the application of the libraries we built using these technologies in the field of synthetic biology.

Categories and Subject Descriptors
I.2.9 (Robotics, Commercial robots and applications), B.4.5 (Reliability, Testing, and Fault-Tolerance), B.5.2 (Design Aids).

General Terms
Algorithms, Management, Measurement, Documentation, Design, Reliability, Experimentation, Verification.

Keywords
Automated assembly techniques, Synthetic biology, DNA editing, DNA libraries, automation, microfluidics.

1. INTRODUCTION

1.1 Computer Programming and DNA Programming
DNA programming is the DNA-counterpart of computer programming. The basic computer programming cycle is to modify an existing program, test the modified program, and iterate until the desired behavior is obtained. Similarly, the DNA programming cycle is to modify a DNA molecule, test its resulting behavior, and iterate until the goal (which is either understanding the behavior or improving it) is achieved. One key difference between the two is

that unlike computer programming, our understanding of DNA as programming language is very far from being perfect, and therefore trial and error are the norm rather than the exception in DNA-based research and development. Hence DNA programming is more efficient if multiple variants of a DNA program, also called a DNA library, are created and tested in parallel, rather than creating and testing just one program at a time. Hence the basic DNA programming cycle, when operating in full steam, takes the best DNA programs from the previous cycle, uses them as a basis for creating a new set of DNA programs, tests them, and iterates until the goal is achieved.

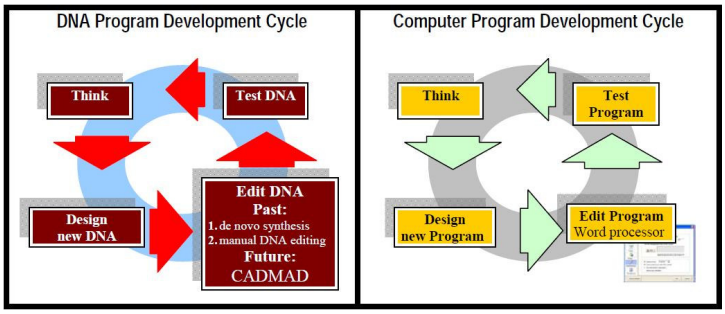


Figure1. The DNA programming cycle is analogous to the computer programming cycle, albeit much more laborious and lengthy.

1.2 Facilitating the Biotechnology R&D cycle
The biotechnology revolution has, to a large extent, been held back by its notoriously prolonged R&D cycle compared to the computer programming cycle. A CAD/CAM technology for DNA which will bring word processor ease to DNA processing and thus support rapid DNA programming will revolutionize biotechnology by shortening the R&D cycle of DNA-based applications. This can only be accomplished by concerting the development of complex, multi layered technologies which integrate expertise from fields as varied as algorithmics, software engineering, biotechnology, robotics, microfluidics and chemistry. These are only now starting to emerge as feasible.

1.3 Editing DNA with Word Processor Ease
What is the DNA-equivalent of the word processor? Word processing was rapidly adopted as a replacement for the typewriter when users had discovered its revolutionary advantages in document creation, editing, formatting and saving. While the electronic representation of text in computers allows the processing of text within a simple unified framework, DNA processing — the creation of variations and combinations of

existing DNA - is performed by biology labs daily using a plethora of unrelated manual labor-intensive methods. As a result, so far no universal method for DNA processing has been proposed and, consequently, no engineering discipline that further utilizes the processed DNA has emerged. We are developing technological platforms that aim to deliver a revolution in DNA processing revolution text editing underwent with the introduction of electronic text editors.

1.4 CADMAD DNA Editing Technology

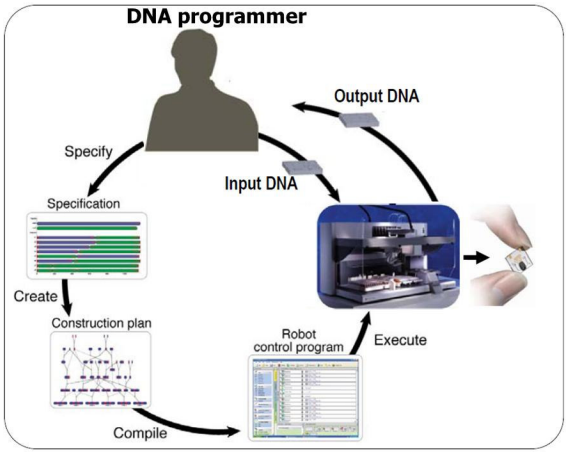


Figure2. Overview of the CADMAD DNA editing technology workflow. A biologist user specifies a library (top left). A construction plan is then computed and a robot control program for its execution is automatically generated (bottom). A robot/microfluidic device executes this plan (right) and its output is tested using one of the verification technologies we developed (1, 3, 4, 5).

To this end we have developed a wide spectrum of technologies ror robotic DNA assembly (1,2) , microfluidic DNA assembly (unpublished), computer aided cloning of assembled DNA (3) , cell-free cloning of assembled DNA (4) and advanced methods for error-correction of assembled DNA (1,5). The development of these technologies and their applications in synthetic biology and biotechnology will be presented and discussed.

1. ACKNOWLEDGMENTS

Our thanks to ACM SIGCHI for allowing us to modify templates they had developed.

2. REFERENCES

[1] Gregory Linshiz, Tuval Ben Yehezkel, Shai Kaplan, Ilan Gronau, Sivan Ravid, Rivka Adar and Ehud Shapiro, 2008. Recursive construction of perfect DNA molecules from imperfect oligonucleotides. *Molecular Systems Biology 4:191* http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2424292/.

[2] Uri Shabi, Shai Kaplan, Gregory Linshiz, Tuval BenYehezkel, Hen Buaron, Yair Mazor, Ehud Shapiro 2010. Processing DNA molecules as text. *Syst Synth Biol (2010) 4:227–236*

http://www.ncbi.nlm.nih.gov/pubmed/21189843

[3] Tuval Ben Yehezkel, Shiran Nagar, Danny Mackrants, Zipora Marx, Gregory Linshiz, Uri Shabi and Ehud Shapiro, 2011. Computer-aided high-throughput cloning of bacteria in liquid medium. *BioTechniques* 50:124-127 doi 10.2144/000113514

http://www.ncbi.nlm.nih.gov/pubmed/21486254

[4] Tuval Ben Yehezkel, Gregory Linshiz, Hen Buaron, Shai Kaplan, Uri Shabi and Ehud Shapiro, 2008. De novo DNA synthesis using single molecule PCR. *Nucleic Acids Research*, 2008, Vol. 36, No. 17 e107 doi:10.1093/nar/gkn457

http://www.ncbi.nlm.nih.gov/pmc/articles/pmc2553596/

[5] Tuval Ben Yehezkel., Tamir Biezuner, Gregory Linshiz, Yair Mazor, Ehud Shapiro 2012. Programmable In Vivo Selection of Arbitrary DNA Sequences. *PLoS ONE* 7(11): e47795. doi:10.1371/journal.pone.0047795

http://www.ncbi.nlm.nih.gov/pubmed/23155373

JBEI-ICE: An Open Source Biological Part Registry

Hector Plahar, Timothy Ham, Joanna Chen and Nathan Hillson

The Joint BioEnergy Institute Inventory of Composable Elements (JBEI-ICE) is an open source registry platform for managing information about biological parts such as plasmids, microbial host strains and Arabidopsis seeds. The information deposited in an ICE installation instance is accessible both via a web browser and through the web application programming interfaces, which allows automated access and interaction with the platform via third-party programs. JBEI-ICE includes several useful web browser-based graphical applications for sequence annotation, manipulation and analysis that are also open source.

One distinguishing feature of ICE is the inclusion of a distributed software platform to enable the efficient sharing of composable biological elements across labs in the synthetic biology research community, thus expanding the search space of biological constructs. It offers community collaboration capabilities and opportunities which enable scientists to publish and share their data sets across multiple ICE installations in different geographic locations. Advanced security mechanisms are also in place to restrict access to particular data sets that researchers are not yet ready to share with the wider community or will prefer to keep private or restricted to a smaller group. The software also enables defining and managing members of these groups. This distributed infrastructure is constructed to be compatible with existing and emerging computer-aided design tools for synthetic biological systems.

JBEI-ICE has been adopted as the registry platform in multiple research labs and institutions across the globe. A public instance is available at public-registry.jbei.org, where users can try out features, upload parts or simply use it for their projects. There is also a downloadable installation made available that users can run on their local machines to try out the software.

Vector NTI® Express Designer - a new bioinformatics platform for synthetic biology design

Kevin Clancy, Kok Hien Gan, Brandon Ang, Kin Chong Sam, Yong Ming Chen, Reina Angelica, Bee Im Teo, Wolfgang Strenzl, Michael Muth, Brynne Stanton, Virgil Rhodius, Ying-Ja Chen, David Raab, Lian Seng Loh, Chris Voigt, Douglas Densmore and Todd Peterson

The increasing scale and sophistication of genetic engineering will necessitate a new generation of biology oriented computer-aided design (BioCAD®) software. For large genetic programs, keeping track of the DNA on the level of nucleotides becomes tedious and error prone. To push the size of such projects, it is important to abstract the designer from the process of part selection and optimization. It is also important to provide the designer access to tools to develop new parts, enter and utilize classification and characterization data on existing parts and to use this information in design projects. As projects develop over time, it is important to provide access to tools and computational methods to refactor under-performing designs. As laboratories develop collections of parts, devices and circuits, it is important to manage such collections. Vector NTI ®Express Designer is a new computational platform that integrates best practice bioinformatics platform with BioCAD® computational design tools. a configurable interface permits 1. the specification of designs from abstract, in silico elements to management of libraries of reliable genetic circuits, 2. automated design of devices and circuits through use of the EUGENE algorithm from the Densmore laboratory, 3. biophysical methods to connect and optimize parts, devices and circuits, 4. tools to debug and refactor designs, 5. algorithms for optimizing codon usage and synthesis of gene sequences, custom vectors and genetic circuits, and 6. algorithms for DNA assembly and experimental design. The software supports many of the the Synthetic Biology Open Language community's recommendations for sharing data, including the SBOL data model, the SBOL visualization recommendations and the SBOL extensions for performance, host context, modeling and regulatory interactions. We have used the software to manage the characterized parts collections developed by the Voigt laboratory. This comprehensive software is accessible to scientists ranging from molecular biologists through to synthetic biologists.

Computationally designed DNA linkers to enhance a novel automated DNA assembly standard

Arturo Casini, Tom Ellis and Geoff Baldwin

Cloning, and assembling DNA in general, is one of the main bottlenecks slowing down the progress of synthetic biology: it's a slow and expensive process, and requires a lot of "ad hoc" adjustments. The lack of a standard also complicates exchange and comparison of constructs between different labs, as the same functional parts can be assembled in radically different ways.

We propose a DNA assembly standard compatible with both the "small parts to genes" and the "genes to pathways" level. In order to achieve high throughput and improve reliability, this method is designed to be compatible with automated liquid-handling platforms by using extremely robust reactions such as restriction digestions and ligations. It is also compatible with a wide range of part sizes and features (such as high GC% or presence of secondary structures) and can be used for the combinatorial assembly of libraries of parts.

One of the main defining features of our approach is the use of "linker" DNA sequences to join parts together: we have specifically developed a new software, R20DNA Designer, that employs a series of algorithms to generates sequences that have all the necessary characteristics to achieve maximum efficiency and specificity in the DNA assembly reactions.

PR-PR: Cross-platform standard language for automation of Synthetic Biology research

Gregory Linshiz, Nina Stawski, Sean Poust, Changhao Bi, Jay Keasling and Nathan Hillson

The development of biology-friendly standard cross-platform programming tools is a crucial step towards automation of research laboratories. The current lack of such tools is a major obstacle for modernization of the Life Science. We have developed high-level biology-friendly programming language PR-PR (Programming Robot) that is used for management of liquid-handling robotic and microfluidic automated platforms. This language allows researchers efficient implementation of protocols for automated DNA synthesis and other complex Synthetic Biology processes, permitting plethora of new high-throughput complex experiments.

PR-PR has established the same set of basic commands for any automation platform. Most of the advanced features of the language are also available for all the compatible platforms, while the underlying optimization and translation processes are unique for each platform. We have created PR-PR as an open source project to boost the adoption of the language as a standard and to encourage the scientific community to add translators for various automated platforms.

Performing laboratory operations on small scales and increasing experiments throughput by using miniaturized microfluidic Lab-on-a-Chip (LOC) devices is the next step forward in the laboratory automation. We are working on development of a new integrated multipurpose microfluidics platform, operated by PR-PR. We can perform a broad spectrum of Synthetic Biology applications on our microfluidics platform using the same PR-PR scripts that can be run on the robotic platforms.

SynBio App Selector: An Application Selection Engine for Synthetic Biology

Sayane Shome
Leukippos Institute
Vellore, India
+91 9566422529
sayaneshome.rsg@gmail.com

Pablo Cárdenas
Leukippos Institute
Bogotá, Colombia
057 2115012
pablocarderam@gmail.com

Gerd Moe-Behrens
Leukippos Institute
Berlin, Germany
0049 8501 8621
leukipposinstitute@googlemail.com

Pedro L. Fernandes
Instituto Gulbenkian de Ciência
Oeiras, Portugal
+351 214407900
pfern@igc.gulbenkian.pt

Alioune Ngom
School of Computer Science,
University of Windsor
Windsor, Ontario, Canada
+1 519 253 3000 Ext. 3789
angom@uwindsor.ca

ABSTRACT
We face a growing number of synthetic biology-related software applications. It is difficult to get an easy overview and orientation about them. We present a novel application to solve this problem with a user interface (UI) based on a systemic, holistic concept. The search is done from top to bottom mainly along the line of the central dogma of molecular biology (DNA-transcription-RNA-translation-protein). Moreover, curated crowdsourcing was used to generate a comprehensive database. SynBio App Selector is an easy-to-use UI to find synthetic biology-related software.

Categories and Subject Descriptors
J.3 [Computer Applications]: Life and Medical Sciences– *Biology and genetics*

General Terms
Algorithms, Design, Documentation.

Keywords
Big Data, systemics, search, synbio software, crowd intelligence.

1. INTRODUCTION
We face a so-called "big data" problem in contemporary science. A large data pool results from computers with enormous storage and processing capacity, vast amounts of data from high throughput experiments, an increase in number and/or size of data sets, and a huge output in scientific literature. All together, it has become difficult to read, assimilate, and process such huge amounts of information [4]. This problem is particularly pertinent in synthetic biology. Synthetic biologists must deal with large amounts of information derived from genomes, transcriptomes, proteomes and metabolomes. The large amount of diverse information must be manipulated with computational tools, and a great number of them have been developed to deal with the task. However, the diversity of tools available matches the diversity of information, and it is often very difficult to find and evaluate all the alternatives in order to determine the appropriate tool for a task. The abundance and variety of the available data resources generates disorientation and calls for a systemic way to guide searches and decision-making.

5th International Workshop on Bio-Design Automation'13, July 12–13, 2013, London, UK.

2. SYSTEMICS AND BIG DATA
Systemics provide a theoretical framework to deal with such a big data problem [1, 2, 6]. Systemic thinking can be described as a holistic view from top to bottom. This paradigm can be applied to search in large data pools. In line with this kind of concepts, immersive, hierarchically structured user interfaces have been developed [5]. Here we describe a novel kind of user interface for easy search in large data pools, based on this previous work.

3. THE APP SELECTOR
3.1 Overview
SynBio App Selector is designed to be an easy-to-use, quick, and accurate reference guide to synthetic biology-related software. The app uses HTML5, JavaScript, and other web technologies to run on a web browser, and supports some mobile platforms. Future versions may include a native mobile app available for download.

3.2 Design Architecture
The App Selector is built using the Model-View-Controller system. This allows for an iterative design strategy, and is especially useful in applications needing regular updates such as the App Selector, which will index new synthetic biology software as it is developed.

3.2.1 Model: Database
The App Selector is built on a MySQL database that stores the list of software and their associated information. As of today, the database contains more than 150 different pieces of software, with information about their function, development status, availability, and dependency on other software. The contents of this database were gathered through a crowd effort, with the work of online volunteers at the Leukippos Institute. After creating a preliminary

list, we sifted through the items determining which pieces of software were pertinent and available. This method of crowdsourcing in combination with a curation process ensured a comprehensive yet high-quality database. Moreover, it shows that cloud collaboration in science is possible. The database will be brought up to date regularly using this crowdsourcing-curating cycle.

3.2.2 View: User Interface

The user interface consists of three different menus (figure 1). Apps are classified according to function in the structure, and apps with multiple functions are included in different branches.

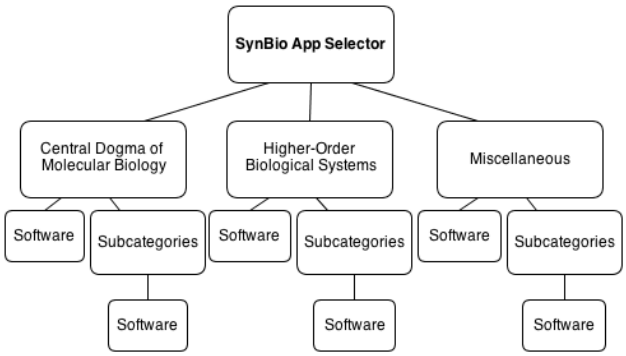


Figure 1: Hierarchical structure used to organize and search for software indexed in SynBio App Selector.

These menus are rendered in the form of icons on a 3D sphere, and the user navigates them by dragging and zooming the sphere. Quick access buttons redirect users to the top menu. The first of these menus displays a schematic representation of the central dogma of molecular biology and leads to software that works with the different molecules and processes involved, such as DNA plasmid design, RNA primer design, or protein analysis (figure 2).

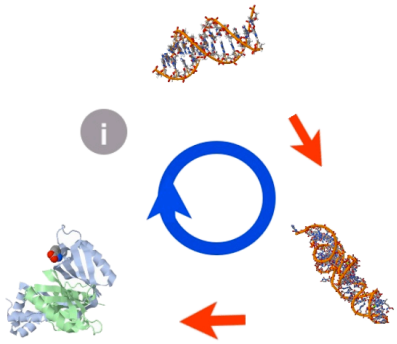


Figure 2: Screenshot of the Central Dogma menu. The icons lie in a 3-D space that can be rotated and zoomed at will.

The second menu contains icons leading to software related to the design, analysis, and modeling of higher order biological systems. The third contains miscellaneous categories, such as educational software, laboratory tools, and information repositories. When the user clicks on one of the entities or processes in the schematic, a screen is presented displaying a list of software and further subcategories to narrow the search. Once the user selects a piece of software, the App Selector displays a screen with information about that particular selection, including a description and links to the developer's web page and related resources. For web-based software, the App Selector may even run the web app internally.

3.2.3 Controller

The controller will be a PHP piece of middleware that addresses queries from the JavaScript UI and fetches the necessary information from the database, using AJAX (Asynchronous JavaScript and XML) technology.

3.3 Development at the Leukippos Institute

The SynBio App Selector is being developed by a team of volunteers at the Leukippos Institute for *in silico* synthetic biology. The Institute, founded in 2010, is an online community that works through cloud collaboration. It is an experiment in open science using web technologies as catalysts for research, discussion, and project development. As of June 2013, a first iteration of the App Selector's database and UI are ready for deployment. Work on the controller is underway, and an alpha version of the application is scheduled for mid July.

4. CONCLUSIONS

The disorganized and vast landscape of available software tools creates a problem in synthetic biology. SynBio App Selector intends to solve this problem, by creating a centralized index of these tools through curated crowdsourcing and providing an organizational structure for them. The hierarchically structured user interface will help people who need software assistance for their synthetic biology designs to find the necessary apps in an easy and intuitive way. In addition, crowdsourcing was shown as an effective way to generate a very comprehensive collection of apps.

5. ACKNOWLEDGMENTS

This project is being developed with the online community at the Leukippos Institute (http://leukippos.org). A full list of members can be found here:

http://www.facebook.com/groups/320000391386916/members/

6. REFERENCES

[1] Ashby WR. 1956. *Introduction to Cybernetics.* Chapman & Hall, London, UK.

[2] Bertalanffy L. 1968. *General System theory: Foundations, Development, Applications.* George Braziller, New York, NY.

[3] Krasner, G. E., Pope, S. T. 1988. A Description of the Model-View-Controller User Interface Paradigm in the Smalltalk-80 System. *System*, *1*(3), 26-49. Retrieved from http://www.itu.dk/courses/VOP/E2005/VOP2005E/8_mvc_krasner_and_pope.pdf

[4] Moe-Behrens, G. 2012. Tool Tales: Leukippos – Synthetic Biology Lab in the Cloud. *Soapbox Science: Nature Blogs.* Retrieved from: http://blogs.nature.com/soapboxscience/2012/05/01/tool-tales-leukippos-synthetic-biology-lab-in-the-cloud

[5] Widjaja YY, Pang CNI, Li SS, Wilkins MR, Lambert TD. 2009. The Interactorium: visualising proteins, complexes and interaction networks in a virtual 3D cell. *Proteomics*, **9**: 5309-315. DOI=http://onlinelibrary.wiley.com/doi/10.1002/pmic.200900260/abstract

[6] Wiener N. 1948. *Cybernetics: Or Control and Communication in the Animal and the Machine.* MIT Press, Cambridge, MA.

SynBrick: A Crowdsourcing Game for Synthetic Biology

Pablo Cárdenas
Leukippos Institute
Bogotá, Colombia
057 2115012
pablocarderam@gmail.com

Sayane Shome
Leukippos Institute
Vellore, India
+91 9566422529
sayaneshome.rsg@gmail.com

Alioune Ngom
School of Computer Science,
University of Windsor
Windsor, Ontario, Canada
+1 519 253 3000 Ext. 3789
angom@uwindsor.ca

Pedro L. Fernandes
Instituto Gulbenkian de Ciência
Oeiras, Portugal
+351 214407900
pfern@igc.gulbenkian.pt

Gerd Moe-Behrens
Leukippos Institute
Berlin, Germany
0049 8501 8621
leukipposinstitute
@googlemail.com

ABSTRACT
Crowdsourcing games offer a novel design approach for synthetic biology by using an online crowd as a means to solve difficult problems. The strategy has already proven its worth in the area of life sciences. We propose to create SynBrick, a game in which web players provide best solutions to complex synthetic biology problems. SynBrick is intended to be a general-purpose problem solver for synthetic biology, in which different problems are provided in the form of puzzles to be solved by an online crowd composed of anyone with time and interest in the game. SynBrick will be a self-contained game with dynamic access to synthetic biology resources (such as databases, apps, and educative resources) required for solving problems given in the game.

Categories and Subject Descriptors
J.3 [Computer Applications]: Life and Medical Sciences– *Biology and genetics*

General Terms
Design.

Keywords
Big Data, crowd intelligence, game, synthetic biology, problem solving, design methodologies.

1. INTRODUCTION
The digital age presents a challenge to science: Big Data. Computational tools are producing high-throughput data sets larger and more complex than ever before, and networking technologies have made access to the plethora of information easy and fast. Conventional computational techniques often fall short in this context, and new approaches are necessary for analyzing and processing such amounts of data [3]. However, digital technologies also enable new ways of coping with these problems, by tapping into unexploited computing resources like the Cloud or the Crowd [1]. One way of harnessing this new computing potential is crowdsourcing via gamification [9]. Engineering designs that are methodologically generated tend to follow conservative patterns. Crowdsourced designs tend to ignore these patterns in the first place, allowing more creative, unforeseen solutions. The strategy has been used in the biological sciences in games like EteRNA, Phylo, and particularly FoldIt, which has produced a number of publications from its results [2, 4, 6]. The advantages of gamifying research problems are twofold: not only is science advanced, but it is also democratized, as the community is involved in a more active way than what simple divulgation would allow. In this spirit, we propose to develop SynBrick, a web-based game in which players cooperate to solve real-life problems using the tools of synthetic biology. The advantages of gamifying research problems are twofold: not only is science advanced, but it is also democratized, as the community is involved in a more active way than what simple divulgation would allow. SynBrick will be a game in which players cooperate to solve engineering problems using the tools of synthetic biology, such as designing biological systems to produce biofuels or medicines, diagnose diseases and clean hazardous waste, and the boundless other applications available for synthetic biology.

5th International Workshop on Bio-Design Automation'13, July 12–13, 2013, London, UK.

2. SYNBRICK

2.1 Proposal
Synthetic biology is in essence a creative practice. Synthetic biologists aim to solve problems through the construction of biological devices and systems. These problems, however, are far too complex to be solved in one go. They must be decoupled into simpler problems that can be worked on independently [5]. Taking advantage of the modular, standardized structure of BioBricks [5, 7], SynBrick uses gamification to engage a web crowd to provide the best solutions to difficult and complex synthetic biology problems, by providing the tools of creation in simple interface and an engaging context.

2.2 Game Structure
SynBrick hinges on synthetic biology's abstraction hierarchy, consisting of parts, devices and systems [5] (see Figure 1). The main objective of the game is to design a synthetic biology construct that solves a given problem. These challenges are proposed and voted on by the player community. Challenges are then decoupled into separate tasks: designing systems and devices that perform specific functions. Each individual task is a puzzle to be solved. A gaming session or "level" consists of a player

attempting to solve a puzzle by building a genetic circuit that fulfills the given task.

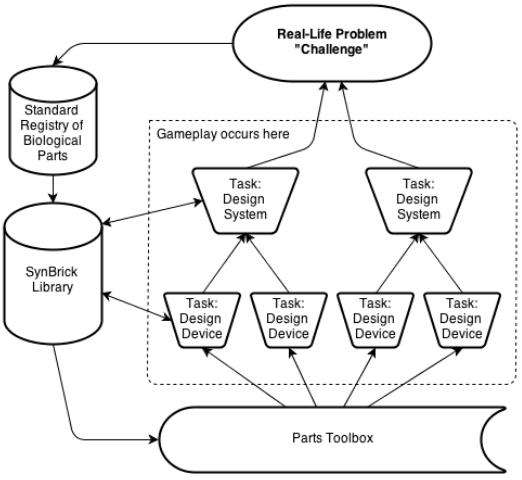


Figure 1: SynBrick's structure and problem-solving design strategy. Arrows denote flow of information.

2.3 Gameplay

Thank to the abstraction hierarchy, players do not have to deal with actual sequences to build their circuits. Players use parts from the Standard Registry to design the devices and systems that carry out the task. Players have at their disposal a "toolbox" of BioBricks from the Registry to choose from. BioBricks are represented by icons with colors and symbols corresponding to their function. Each BioBrick has an attached "data sheet", in which the player can access its name, function, and properties. The player mixes and matches BioBricks in a snap-on, modular fashion. The game simulates the genetic circuit built by the player, displaying interactions among BioBricks, the flow of Polymerases Per Second, and the output of the circuit in a graphical, iconized manner. Each solution is evaluated by simulating the circuit's functioning based on each part's characterization in the Registry, in a way similar to the way BioCAD software like TinkerCell works (http://tinkercell.com). Solutions are judged on predicted efficiency and resource consumption. Biosafety is also taken into account through homology searches with potentially dangerous sequences, concatenating each part's Registry sequence data. Players earn points for their solutions, and the best solutions can be tested in the wet lab.

3. PROSPECTS

3.1 Challenges

A few potential issues may hinder SynBrick's development and effectiveness. Incorrect part characterization, incompatibility, "junk" parts in the Registry, unpredictable circuitry, and variability [8] may all affect the simulation's accuracy and the game's reliability as a crowdsourcing platform. While including only well-characterized and tested BioBricks from the Registry may lessen these issues, they are issues inherent to synthetic biology's state of the art. Nevertheless, the sheer amount of solutions provided by the game will doubtlessly at least point to new potential designs, as do all crowd data strategies.

3.2 Design Strategy

We are making SynBrick using an iterative design strategy. We will start with a simple prototype focused on key game functionality, and work up in iterations adding functionality based on the feedback we get from beta testers. We are currently working on such a tentative prototype. A mock-up demo will be ready for the conference.

4. CONCLUSIONS

Crowdsourcing through gamification is a novel design and optimization methodology worth testing in the context of synthetic biology. In spite of the challenges it faces, it may well produce some new, creative, and potentially useful ideas, as have other crowdsourcing games. Our proposal of a web game, SynBrick, aims to harness this potential. Finally, it is citizen science project that democratizes the field and engages public in research directly. In this way, it improves public perception and knowledge of synthetic biology.

5. ACKNOWLEDGMENTS

This project is being developed with the online community at the Leukippos Institute. A full list of members can be found here:

http://www.facebook.com/groups/320000391386916/members/

6. REFERENCES

[1] Brabham, D. C. 2008. Crowdsourcing as a Model for Problem Solving: An Introduction and Cases. *Convergence: The International Journal of Research into New Media Technologies, 14*(1), 75-90. DOI= http://con.sagepub.com/cgi/doi/10.1177/1354856507084420

[2] Cooper, S., Khatib, F., Treuille, A., Barbero, J., Lee, J., Beenen, M., Leaver-Fay, A., et al. 2010. Predicting protein structures with a multiplayer online game. *Nature, 466*(7307), 756-760. DOI= http://www.nature.com/doifinder/10.1038/nature09304

[3] Dumbill, E. 2013. Making Sense of Big Data. *Big Data 1*(1): 1-2. DOI= http://online.liebertpub.com/doi/full/10.1089/big.2012.1503.

[4] Eiben, C. B., Siegel, J. B., Bale, J. B., Cooper, S., Khatib, F., Shen, B. W., Players, F., et al. 2012. Increased Diels-Alderase activity through backbone remodeling guided by Foldit players. *Nat Biotechnol, 30*(2), 190-2. DOI= http://www.nature.com/doifinder/10.1038/nbt.2109

[5] Endy, D. 2005. Foundations for engineering biology. *Nature*, *438*(7067), 449-53. DOI– http://www.nature.com/doifinder/10.1038/nature04342

[6] Khatib, F., Cooper, S., Tyka, M. D., Xu, K., Makedon, I., Popovic, Z., Baker, D., et al. 2011. Algorithm discovery by protein folding game players. *P Natl Acad Sci USA, 108*(47), 1-5. DOI= http://www.pnas.org/cgi/doi/10.1073/pnas.1115898108

[7] Knight, T. 2003. Idempotent Vector Design for the Standard Assembly of Biobricks. *Artificial Intelligence*, 1-11. DOI= http://dspace.mit.edu/handle/1721.1/21168

[8] Kwok R. 2010. Five hard truths for synthetic biology. *Nature* 463: 288–290. DOI– http://www.nature.com/doifinder/doi:10.1038/463288a

[9] Lakhani K. R., Boudreau, K. J., Loh, P. R., Backstrom, L., Baldwin, C., Lonstein E., Lydon M., et al. 2013. Prize-based contests can provide solutions to computational biology problems. *Nat Biotechnol, 31*(2), 108-111. DOI= http://www.nature.com/doifinder/10.1038/nbt.2495

Cello: A software workflow for modular design and compilation of genetic circuits

Rishi Ganguly[2], Peng Liu[3], Swapnil Bhatia[2], Christopher Voigt[3], Douglas Densmore[1,2]
Departments of Bioinformatics[1], Electrical and Computer Engineering[2],
Boston University, Boston, MA, USA, and Department of Biological Engineering
Massachusetts Institute Of Technology [3], Boston, MA, USA
Email: {rganguly, dougd}@bu.edu

I. Introduction

We present a software workflow called "Cello" (a syllabic abbreviation of "cell logic") which encapsulates genetic devices as independent functional modules and combines them to produce biological systems satisfying a more complex behavior. There are two steps in the workflow. First, a user describes a "cello motif": a specification defining the device's intended function and rules for combining it with other devices. In the second step, Cello uses these motifs to synthesize a device satisfying the given function. We present our approach to this synthesis problem of generating candidate biological circuits satisfying a given specification using the library of cello motifs defined by the user. The specifications we support range from structural Verilog descriptions representing boolean algebraic primitives to behaviorial Verilog descriptions describing complex systems. Our algorithms utilize user-defined constraints such as an upper-bound on the length of the DNA sequence of the circuit and number of parts of a particular type, to produce an optimized set of candidate circuits with components assigned from a database of characterized parts, though we do not describe the details of the assignment algorithms here.

II. Methods and Algorithms

Our workflow comprises two steps: 1) specifying a cello motif using context free grammar rules and 2) synthesizing an abstract genetic circuit from a given specification.

A. Specifying Functional Motifs

A "Cello motif" is an abstract representation of a genetic circuit module with a specified logic function. Cello motifs serve as "building blocks" to be combined with each other, producing more complex logic. Examples of motifs could be abstract genetic devices that implement boolean logic gates[1]. We have developed a formal system for defining these motifs using context-free grammar rules which comprises:[2]

- a set of *terminal* symbols, where each symbol corresponds to a biological part type (examples: Promoters, Terminators, Coding-sequence, etc)
- a set of *non-terminal* symbols, which are place-holders for patterns of terminal symbols that can be generated by the non-terminal symbols. These symbols are enclosed in square brackets as shown in left of figure 1, and the subscript denotes the part-type associated with it.
- a set of *productions*, which are rules for replacing non-terminal symbols (on the left side of the production) in a string with other non-terminal or terminal symbols (on the right side of the production).
- a *start symbol*, which is a special non-terminal symbol that appears in the initial string generated by a particular motif grammar. This initial string represents the overall composition of the motif.

Our approach is sufficiently powerful to describe functionality and composition of devices, while independent of the underlying biological mechanism realizing the function.The following are the basic rules for writing a valid Cello motif:

- Composition of the circuit is defined by using grammar rules over the alphabet of terminals and non-terminals.
- "Inputs" to the motif are symbols that resolve to *blank* non-terminals. These symbols could be replaced by a terminal symbol matching the part-type, or a whole motif, whose start symbol matches the part type.
- Rules of composition should be such that the underlying logic function is preserved.
- Logic functions are defined by equations along with grammar rules,where the equation is defined in terms of the symbols used in the rule.

We provide transcriptional and invertase based motif examples in the left of Figure 1. The following is the alphabet set used in the grammar rules for the motifs in the figure.
{ **p** : *promoter*, **cds** : *coding sequence/gene*, **r** : *ribosome binding site*, **t** : *terminator* }

B. Compilation from a high level specification to Abstract Genetic Circuits

As mentioned earlier, the functional specification can be a structural Verilog description representing boolean algebraic primitives to a behaviorial Verilog description. Our custom compiler parses the Verilog code to produce an equivalent truth table specification, which is further reduced to a minimum sum-of-product expression, using the two level Quine-McCluskey logic minimization algorithm [5]. We then parse

[1]The exact granularity of the "building blocks" is left for the user to determine

[2]A context-free grammar (CFG) is a set of recursive rewriting rules (or productions) used to generate patterns of strings [3].

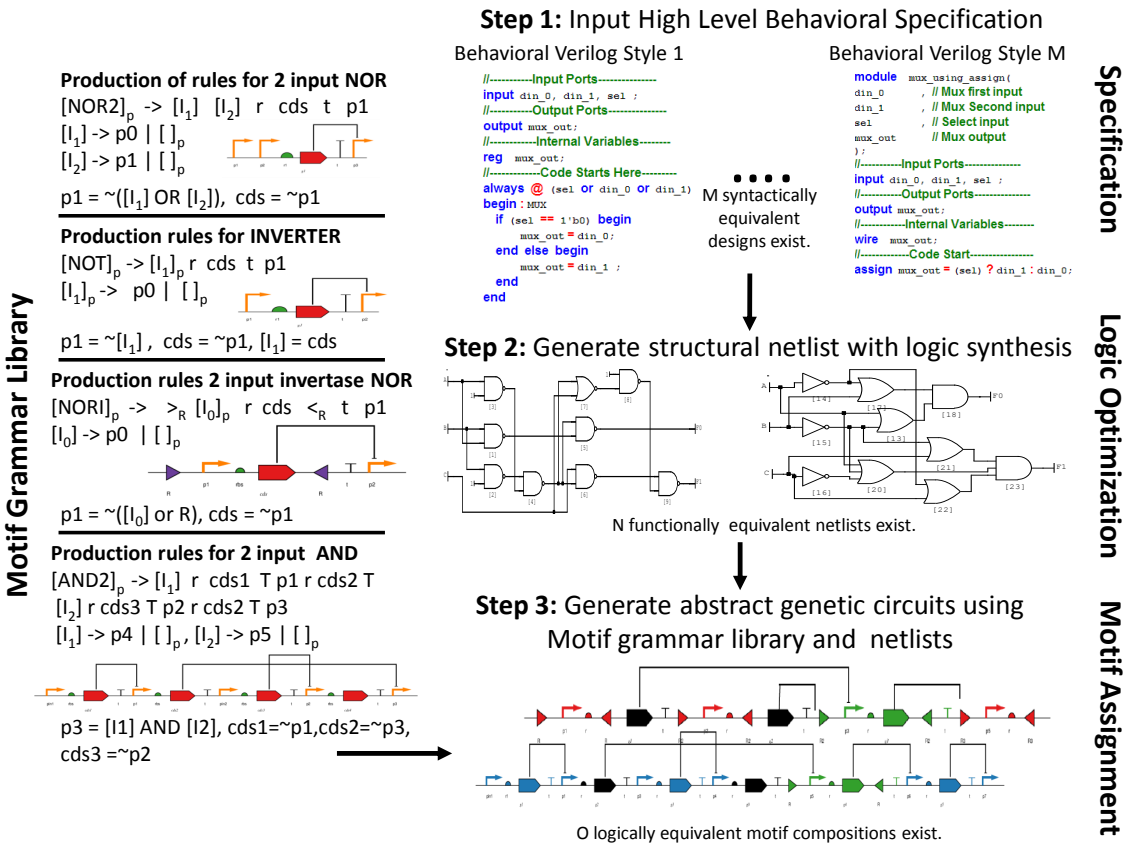


Fig. 1. An overview of the Cello workflow: Step 1, consists of defining the motifs using a Context Free Grammar formalism. Non-terminal symbols are enclosed in square brackets and empty square brackets refer to a blank non-terminal. Each symbol is also associated with a part type, where the subscript of the non-terminals, and prefixes of terminal symbols represents their part type. The logic equations associated with the start rule, inform the boolean logic of the circuit, and the repressing or inducing relation with the output promoter and gene pair respectively. In step 2, a given Verilog program is compiled to produce a functional logic circuit in the form of a *netlist*. The same functional specification could be written in multiple ways in Verilog, which can produce different netlists based on structural optimizations using constraints specified by user. In step 3, the resultant netlists are mapped to corresponding genetic motifs matching the logic function of the corresponding module. A single netlist can again produce multiple genetic circuits depending on user defined motif constraints such as those on the maximum number of parts of a type or on the total allowable length of the genetic circuit.

the resultant expression, and form a logic circuit in the form of a "netlist" using the motifs already provided.

C. Motif search using constraints

Each of the functional modules in the netlist corresponds to a set of Cello motifs. Using the user defined constraints, such as an upper-bound on number of parts of a type to be used or maximum length of the DNA sequence of the resultant circuit, we assign the functional modules with multiple matching genetic motifs, producing multiple abstract genetic circuits. We formulate an optimization problem with the constraints as parameters to perform a search in the "motif space" to find circuit topologies that will meet the functional specification.

III. Conclusions and Future Work

We have outlined a formalism for defining biological motifs encapsulating functionality and composition, while independent of the underlying biological mechanism realizing the function. To the best of our knowledge, this approach is unique, among existing bio-design automation approaches such as [1], [2] and [4]. We reuse and reconfigure existing Electronic Design Automation tools and methodologies to produce genetic circuits of arbitrary complex behavior, using the motifs. We have also outlined approaches to utilize given constraints to produce multiple abstract genetic circuits adhering to the specifications. Immediate future work entails integration with a characterized part registry database (with transfer functions) to simulate dynamic behavior of the resultant circuits and using existing biophysical modeling algorithms such as [6] to fine tune the circuits to match specifications.

References

[1] L. Huynh, A. Tsoukalas, M. Kppe, and I. Tagkopoulos. SBROME: A scalable optimization and module matching framework for automated biosystems design. *ACS Synthetic Biology*, 0(0):null, 0.

[2] R. W. Jacob Beal mail, Ting Lu. Automatic compilation from high-level biologically-oriented programming language to genetic regulatory networks. *plos one*, 2011.

[3] H. Lewis and C. Papadimitriou. *Elements of the Theory of Computation*. Prentice Hall, 1998.

[4] M. A. Marchisio and J. Stelling. Automatic design of digital synthetic gene circuits. *PLoS Comput Biol*, 7(2):e1001083, 02 2011.

[5] R. L. Rudell. Multiple-valued logic minimization for pla synthesis. *Memorandum No. UCB/ERL M86-65 (Berkeley)*, 1986.

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

Online Tools for Characterization, Design, and Debugging

Aaron Adler
Raytheon BBN Technologies
10 Moulton Street
Cambridge, MA, USA 02138
aadler@bbn.com

Fusun Yaman
Raytheon BBN Technologies
10 Moulton Street
Cambridge, MA, USA 02138
fusun@bbn.com

Jacob Beal
Raytheon BBN Technologies
10 Moulton Street
Cambridge, MA, USA 02138
jakebeal@bbn.com

1. MOTIVATION

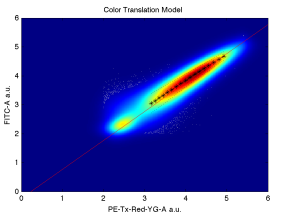
The engineering of biological systems can be greatly aided by better models, derived from high-quality characterization data, and by better means for designing and debugging new genetic circuits. Web-based tools and repositories have proven a successful approach to distributing such techniques, particularly because the centralization of infrastructure greatly decreases adoption cost for new users. Notable examples include the Parts Registry [8], the RBS calculator [10], GeneDesign [9], GenoCAD [4], BioFab [7], and JBEI ICE [6].

No prior web-based tools, however, have supported either analysis of characterization data or high-level design that can take advantage of such data. Previously, we have constructed a number of such tools during the course of building the TASBE end-to-end tool-chain for biological design [2]. We have now improved these tools to be more user friendly and broadly applicable, and placed them online as free web-based tools, embedded in a secure architecture to preserve data privacy.

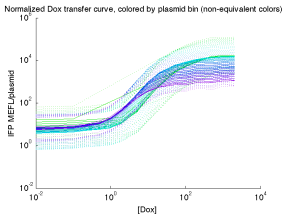
2. TASBE WEB-BASED TOOLS

At the present time, the TASBE Tools website provides access to a suite of three tools: Color Models, Characterization Experiment, and BioCompiler. These tools are accessible online at `https://synbiotools.bbn.com`. Access is free, and may be done either anonymously or with a registered account that allows data to be kept private, as described below in Section 4.

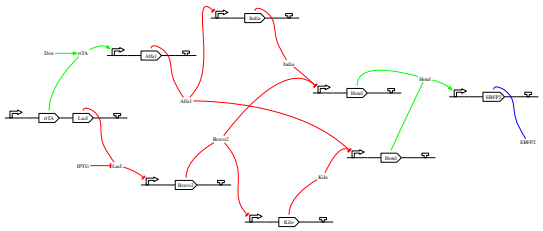
The Color Models tool uses flow cytometry data from control samples to create a calibrated model of single-cell fluorescent expression. This tool implements the fluorescent calibration methodology described in [3], in which the arbitrary units produced by a flow cytometer are mapped into standardized FITC units. This requires four controls: two standard controls: (1) blank cells for estimating autofluorescence, and (2) single constitutive expression of each fluorescent color, and two non-standard controls: (3) the fluorescent beads typically used for calibration and maintenance of flow cytometers (used for mapping the cytometer's FITC channel to absolute units), and (4) co-expression controls with two or three colors independently expressed using identical promoters (used for determining equivalent fluorescent expression levels in context). Given these inputs, the tool produces a color model that can be used to translate



(a) Red→Yellow Map



(b) Dox. Induction Curve



(c) Generated Exclusive-Or Circuit

Figure 1: Example results from the three web services currently available on the TASBE tools site: (a) calibration of unit translation for a Color Model, (b) transfer curve (mean (solid lines) and standard deviations (dashed lines)) binned and normalized by constitutive expression from a Characterization Experiment, and (c) optimized transcriptional exclusive-or designed and visualized by BioCompiler.

flow cytometer data into reproducible absolute units of measurement, as well as compensate for autofluorescence and spectral overlap. Figure 1(a) shows an example of the color model's collateral outputs: a translation model for mapping between red and yellow fluorescence.

The Characterization Experiment tool provides detailed analysis of single-variable flow cytometry experiments, again following the methodology presented in [3]. In particular, the tool computes the relationship between a controlled variable (e.g., time, inducer) and the statistical distributions of a constitutive marker and up to two other fluorescent proteins. Using a color model produced by the prior tool, the results are given in standardized MEFL units that are replicable between labs and experiments. This approach is particularly useful for transient transfections, where the number of

IWBDA '13 London, UK

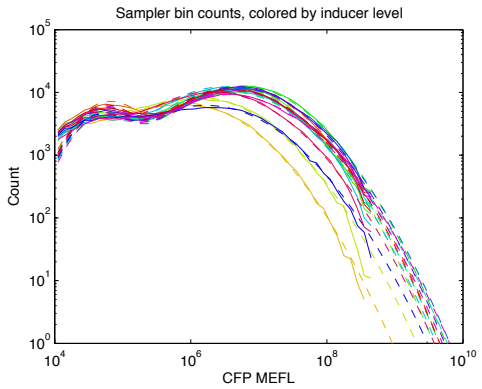


Figure 2: TASBE tools reveal normally hidden problems in experimental data (solid lines; model fits shown as dashed lines), such as the three low-quality transfections (orange, light green, and dark blue) in this data set.

plasmids per cell may vary by orders of magnitude, but has also been applied to tighter distributions, such as replicating plasmids or genomic integrations. Figure 1(b) shows an example of its outputs: an input/output curve for induction of rtTA with doxycycline, normalized by estimated plasmid count.

Finally, the Proto BioCompiler [1] takes high-level computational specifications and creates optimized genetic regulatory networks that implement those specifications. The newly designed network is then exported for the user in two standards-based formats—an SBOL [11] XML file specifying the design, and a visualization in GraphViz [5] using its built-in SBOLv symbols—plus Matlab files specifying an ODE-based simulation. Figure 1(c) shows an example of its outputs: a GraphViz visualization of a genetic regulatory network generated from a specification for a transcriptional exclusive-or circuit.

3. BIOLOGIST-FOCUSED DEBUGGING

A key challenge in biological experiments is differentiating between erroneous results that should be discarded and odd or unexpected results that bear further investigation. The TASBE tools encode knowledge to assist biologists in making this determination, by automatically examining color model and characterization experiment data for known classes of problems. These are then reported to the user in the form of warnings (e.g., of a more than 5% mismatch to the expected fluorescent bead distribution) or errors (e.g., such poor control fluorescence that color translation is not possible).

There are many ways that experiments can go wrong, and any human is likely to overlook some potential problems, especially subtle ones. A software tool reliably checks all its encoded failure cases, where a hurried experimenter may skip or overlook something. We provide an online biologist-focused manual that describes how the messages from the tools typically relate to issues in experimental protocols, and provides recommended strategies for addressing these issues.

The tools also take advantage of the internal cross-validation in high-resolution characterization data, presenting results in such a way as to reveal problems that are obscured by

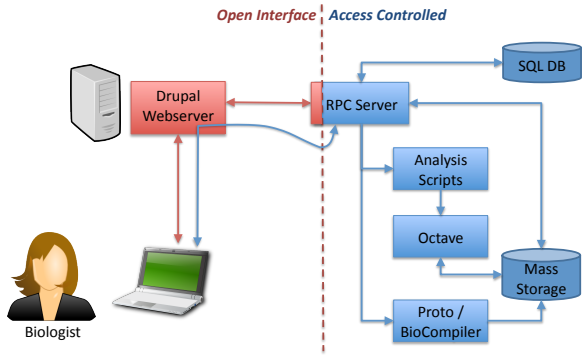


Figure 3: The architecture of the TASBE tools web serivce is designed to ensure data security and scalability.

typical analyses. For example, Figure 2 shows constitutive distribution curves for a characterization experiment (experimental data shown as solid lines, model fits as dashed lines): three are clearly indicated by their relatively much lower curves as poor quality transfections that should be discarded, even though their data covers the approximately same overall range.

4. WEB SERVICE ARCHITECTURE

Two key requirements drove our architectural decisions for these web services. First and foremost is the necessity to ensure the biological data remains secure and private. Not only are there potential ethical, legal, and intellectual property considerations, but also biologists typically consider secrecy of data vital in order to avoid being scooped. Second is the need for scalability, which comes from the large requirements for data and computation when processing flow cytometry data; data sets are frequently more than a gigabyte in size and can take more than ten minutes to process.

We thus designed the architecture shown in Figure 3, executing on an SE Linux platform. Critically, no data is ever placed in a location accessible to the web server: instead, the web server is simply used for routing data on a secure and authenticated connection between the RPC server (which is not externally exposed) and the user's browser. This means that no third party accessing the web server can obtain a user's data without being able to authenticate as that user.

5. FUTURE DIRECTIONS

The TASBE tools are already being used by several laboratories, and we aim to continue improving the tools to better serve the needs of an expanding user community and according to priorities based on user feedback. Key expected improvements include batch file upload, laboratory and project data sharing, and incremental addition of new knowledge-driven debugging assistance. We also plan to continue adding more tools from the TASBE suite; near-term candidates include MatchMaker [12], control experiment planning, color selection tools, and multi-variable characterization.

6. ACKNOWLEDGMENTS

Example data produced by Noah Davidsohn and Ron Weiss.

7. REFERENCES

[1] J. Beal, T. Lu, and R. Weiss. Automatic compilation from high-level biologically-oriented programming language to genetic regulatory networks. *PLoS ONE*, 6(8):e22490, August 2011.

[2] J. Beal, R. Weiss, D. Densmore, A. Adler, E. Appleton, J. Babb, S. Bhatia, N. Davidsohn, T. Haddock, J. Loyall, et al. An end-to-end workflow for engineering of biological networks from high-level specifications. *ACS Synthetic Biology*, 1(8):317–331, 2012.

[3] J. Beal, R. Weiss, F. Yaman, N. Davidsohn, and A. Adler. A method for fast, high-precision characterization of synthetic biology devices. Technical Report MIT-CSAIL-TR-2012-008, MIT, April 2012. Technical Report: MIT-CSAIL-TR-2012-008 http://hdl.handle.net/1721.1/69973.

[4] M. J. Czar, Y. Cai, and J. Peccoud. Writing dna with genocad. *Nucleic Acids Research*, 37(Web Server issue):W40–W47, 2009.

[5] J. Ellson, E. R. Gansner, E. Koutsofios, S. C. North, and G. Woodhull. Graphviz and dynagraph: static and dynamic graph drawing tools. In *GRAPH DRAWING SOFTWARE*, pages 127–148. Springer-Verlag, 2003.

[6] T. S. Ham, Z. Dmytriv, H. Plahar, J. Chen, N. J. Hillson, and J. D. Keasling. Design, implementation and practice of jbei-ice: an open source biological part registry platform and tools. *Nucleic acids research*, 40(18):e141–e141, 2012.

[7] V. K. Mutalik, J. C. Guimaraes, G. Cambray, C. Lam, M. J. Christoffersen, Q.-A. Mai, A. B. Tran, M. Paull, J. D. Keasling, A. P. Arkin, and D. Endy. Precise and reliable gene expression via standard transcription and translation initiation elements. *Nature Methods*, 10:354–360, April 2013.

[8] Registry of standard biological parts, Retrieved May, 2013.

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

[11] SBOL. Synthetic Biology Open Language. http://www.sbolstandard.org, January 2013.

[12] F. Yaman, S. Bhatia, A. Adler, D. Densmore, and J. Beal. Automated selection of synthetic biology parts for genetic regulatory networks. *ACS Synthetic Biology*, 1(8):332–344, 2012.

A NoSQL Approach to a Synthetic Biology Database

Eduardo Abeliuk
TeselaGen Biotechnology
San Francisco, CA
eduardo@teselagen.com

Rodrigo Pavez
TeselaGen Biotechnology
San Francisco, CA
rpavez@teselagen.com

Yuri Bendana
TeselaGen Biotechnology
San Francisco, CA
ybendana@teselagen.com

Nick Elsbree
TeselaGen Biotechnology
San Francisco, CA
nelsbree@teselagen.com

Tim Thimmaiah
TeselaGen Biotechnology
San Francisco, CA
tim@teselagen.com

Nathan J. Hillson
TeselaGen Biotechnology
San Francisco, CA
njhillson@teselagen.com

Michael Fero
TeselaGen Biotechnology
San Francisco, CA
mike.fero@teselagen.com

ABSTRACT
Traditional relational databases have been applied to almost every data archival and retrieval problem imaginable. Why not choose a standard SQL-driven relational database management system (RDMS) for synthetic biology? Here we discuss our selection of MongoDB, a NoSQL document driven database, to support our BioCAD/CAM software platform. Advances in NoSQL databases, like MongoDB, have made it easier and more practical to build dynamic web based applications on top of document driven databases. Here, we explain how we are exploiting these advances to build a highly interactive system for building and testing synthetic biological constructs.

Keywords
NoSQL, CAD, CAM, synthetic biology, DNA assembly.

1. INTRODUCTION
Recent advances in multiple restriction site free (MCS-free) cloning technologies have opened new opportunities for basic research and industrial biotechnology. These advances have been made in response to an increasing demand for large-scale DNA assemblies, up to and including entire genomes. There is a need for platforms that can empower the synthetic biologist to easily and cost-effectively design and assemble genetic designs. At TeselaGen Biotechnology, we are developing a BioCAD/CAM rapid prototyping system for biology that can provide a design-build-test-evolve platform that includes automated DNA assembly, with a particular strength in the design of combinatorial DNA libraries. Here, we describe a portion of our approach and discuss its benefits compared with alternatives.

2. PLATFORM OVERVIEW
In the following, we focus on the "design" layer of our platform architecture.

2.1 BioCAD for combinatorial DNA design
Our BioCAD tool is a visual canvas, inspired by DeviceEditor [1] and VectorEditor [2] and built with ExtJS and NodeJS, which facilities the creation of DNA designs ranging from simple DNA parts and devices to more complex combinatorial DNA designs (see Figures 1 and 2). While there are several formats used to specify DNA components (FASTA, GenBank, SBOL Core), there has been a lack of standards to specify, store, and share combinatorial DNA designs. Combinatorial DNA designs are important because biological systems are not as easily characterized or simulated as their electronic or mechanical counterparts. Real progress in the field frequently involves building and testing many variations of a genetic construct, and then performing functional tests to find the optimal DNA construct that encodes the best performing biological system.

2.2 Cloud-based DNA assembly optimizer
Cloud computing is currently being used by thousands of institutions, from universities, to small startups, to big companies, and is emerging as a viable solution for storing and processing data outside of institutional settings. Cloud computing provides several advantages over traditional in-house storage and computing. Perhaps most importantly, institutions can avoid making big infrastructure investments and alternatively route their resources to projects that fully exploit their expertise.

Our BioCAD tool is supported by a cloud-based DNA assembler, j5 [3]. j5 cost-optimizes DNA assembly, leveraging outsourced DNA synthesis where cost-effective. TeselaGen's optimizer identifies putative assembly incompatibilities and suggests workarounds, as it automatically generates instructions to assemble scar-less combinatorial DNA libraries. During this optimization process, our optimizer can apply Eugene specification design rules [4] to automatically eliminate undesirable constructs. Optimization of the assembly instructions for a DNA library can become computationally intensive when working with combinatorial libraries with millions of constructs. Depending on the size and complexity of the DNA library, in our experience the optimization of DNA assembly processes can take from a few seconds to several hours, using a single server.

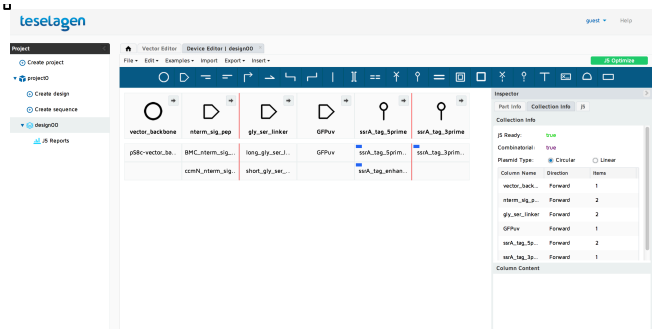


Figure 1. Example of a combinatorial DNA design rendered on TeselaGen's BioCAD

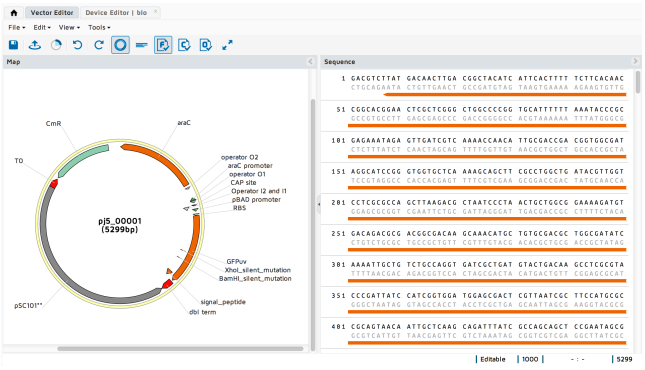


Figure 2. Example of a DNA vector rendered on TeselaGen's bioCAD.

3. IMPLEMENTATION IN A NOSQL DB

TeselaGen's platform greatly benefits from a NoSQL database system, which is very well suited to cloud-based storage and computing, as well as to agile development.

3.1 Categories of NoSQL databases

Various NoSQL databases have been developed for different purposes and applications. Some categories of NoSQL databases include:

i) Key-value and row-oriented databases, such as Redis and Cassandra. These databases are typically optimized for the fast recovery of string, integer, or array objects distributed across multiple machines.

ii) Graph databases, such as Neo4j and InfiniteGraph. These databases are loaded with powerful methods to compute graph metrics and perform topological analysis on Social Network or Semantic Web data.

iii) Document-based databases, such a MongoDB and BaseX. These databases typically recognize structure within objects and provide functionality to search for objects with specific attribute values. These databases are ideal for storing and querying objects, such as DNA combinatorial designs, that contain some data structures that need to be efficiently indexed and retrieved, but that can't be easily defined in a structured schema.

3.2 MongoDB for BioCAD

TeselaGen's backend relies on MongoDB [5], which is highly optimized for storing and indexing schema-less objects. Each DNA design is stored in MongoDB as a single BSON object (a binary form of JSON), which can be efficiently queried, easily visualized in our BioCAD tool, and exported as an XML document for cross-compatibility with other BioCAD tools. Since BSON objects stored in MongoDB are easily expressed in JSON, they integrate very well with our web-development platform, which is based on HTML5 and JavaScript.

3.3 Evolving standards for bioDesign

Developing BioCAD software tools for the synthetic biology community is challenging because data models are often rapidly changing or poorly defined. For example, storing and sharing experimental performance specifications have yet to be fully standardized. This is to be expected, since synthetic biology is an emerging and fast-growing field, where new data exchange standards such as those developed by the Synthetic Biology Open Language (SBOL) [6] community are still in their infancy.

In contrast with relational databases that require well-defined schemas, it is possible to store and query objects in a NoSQL database such as MongoDB without a pre-defined schema. Querying such a database can also be accomplished without the need of complex joins. This translates into faster development cycles and better maintainability, which are key for the successful development of a fast-paced evolving software platform for synthetic biology.

4. CONCLUSIONS

We have created a cloud-based solution for designing and automating the assembly of complex combinatorial DNA libraries. We store complex combinatorial DNA designs as objects in a schema-less NoSQL MongoDB database, which is very well suited to cloud-based storage and computing and agile development, supports the rapidly changing or poorly defined data models frequently encountered in synthetic biology applications, and integrates very well with our HTML5 and JavaScript web-development platform.

5. ACKNOWLEDGMENTS

The authors would like to acknowledge support from the National Science Foundation (SBIR awards 1215583 and 1330944).

6. REFERENCES

[1] Chen, J., Densmore, D., Ham, T.S., Keasling, J.D., and Hillson, N.J. (2012) DeviceEditor visual biological CAD canvas. *J. Biol. Eng. 6 (1).*

[2] Ham, T.S., Dmytriv, Z., Plahar, H., Chen, J., Hillson, N.J., and Keasling, J.D. (2012) Design, Implementation and Practice of JBEI-ICE: An Open Source Biological Part Registry Platform and Tools. *Nucleic Acids Research 40(18)*:e141.

[3] Hillson, N.J., Rosengarten, R.D., and Keasling, J.D. (2012) j5 DNA assembly design automation software. *ACS Synthetic Biology 1(1)*, 14-21.

[4] Bilitchenko L, Liu A, Cheung S, Weeding E, Xia B, Leguia M, Anderson JC, Densmore D. (2011) Eugene--a domain specific language for specifying and constraining synthetic biological parts, devices, and systems. *PLoS One 6(4).*

[5] MongoDB. http://www.mongodb.org

[6] SBOL XML/RDF file format: http://www.sbolstandard.org/initiatives/serialization

Biomolecular Implementation of Linear Dynamical Systems

[Extended Abstract]

Vishwesh V. Kulkarni*
University of Minnesota
Department of Elect. Engg.
Minneapolis, MN 55455
vvk215@gmail.com

Boyan Yordanov
Microsoft Research Center
Bio Computation Group
Cambridge CB1 2FB, U. K.
yordanov@microsoft.com

Andrew Phillips
Microsoft Research Center
Bio Computation Group
Cambridge CB1 2FB, U.K.
aphillip@microsoft.com

Jongmin Kim
Caltech
Department of Bioengineering
Pasadena, CA 91225
jongmin@dna.caltech.edu

1. INTRODUCTION

Whereas no generic rules are available today to synthesize programmable *in vivo* devices in a modular manner, it is possible to attempt such a modular design of programmable *in vitro* devices since well-understood models of hybridization and strand displacement are available to predict and synthesize molecular interactions. Furthermore, it was recently shown that *any* physically realistic abstract chemical reaction can be well-approximated by an appropriately designed DNA strand-displacement reaction (see [1]). Indeed, Oishi-Klavins have recently observed that any arbitrary linear dynamical systems can be built as a composition of 3 idealized chemical reactions, namely, catalysis, degradation, and annihilation (see [2]). As a result, Oishi-Klavins have elegantly facilitated a modular architecture to synthesize linear dynamical systems using these 3 types of abstract idealized chemical reactions.

A practical implementation of these reactions using biomolecular components remains a challenge though. For example, suppose we were to rely on the *DNA strand displacement* (DSD) technique to implement these reactions — today, toehold-mediated branch migration and strand displacement reactions are quantitatively well-understood and many complex devices have been constructed around this design principle (see [3]). In the chemical realization proposed in [2], both catalysis and annihilation reaction requires 5 DNA strands (including the fuel species), and the degradation reaction requires 2 DNA strands. Thus, even relatively simple circuits require hundreds of distinct DNA strands for its implementation. Despite successful examples in the laboratory (see [4] and references therein), the wet-lab implementation of large-scale strand-displacment circuits still remains a challange. Furthermore, these experimental strand-displacement cascades are driven by the approach to the thermodynamic equilibrium, limiting the evolutions of the concentrations of system components to simple, unidirectional trajectories [5]. As illustrated in Fig. 1, we show that a pure DSD-based implementation is too slow to be of any use in most practical applications. For example, it turns out that an addition of a pole in the system dynamics increases this time scale by a factor of 1000. We propose solutions to some of the above difficulties by using a combination of DNA and enzymes to realize the dynamical systems.

2. MAIN RESULTS

Our solution relies on the enzymatic reactions used in the *DNA toolbox* (see [5]). Briefly, in the DNA toolbox framework, short DNA activators activate the DNA templates that consequently produce other DNA activators or inhibitors and establish the communication between templates. The *activation* is realized when a DNA activator hybridizes to a DNA template and primes the polymerization; the *inhibition* is realized when a DNA inhibitor hybridizes to a template, thus blocking the activators. An exonuclease degrades DNA activators and inhibitors into dNMPs.

Within the DNA-toolbox, one DNA strand is required to provide template for replication, i.e., to implement the catalysis. In a typical experimental realization, the activator input serves as the primer for DNA replication, and therefore input itself is consumed at first. Nicking enzyme can be used to separate the activator input DNA from the output DNA. Input DNAs are not immune to DNase activity and hence the reference input for the controller should be protected by chemical modification; template DNAs are similarly protected from exonuclease. Because exonuclease acts on any

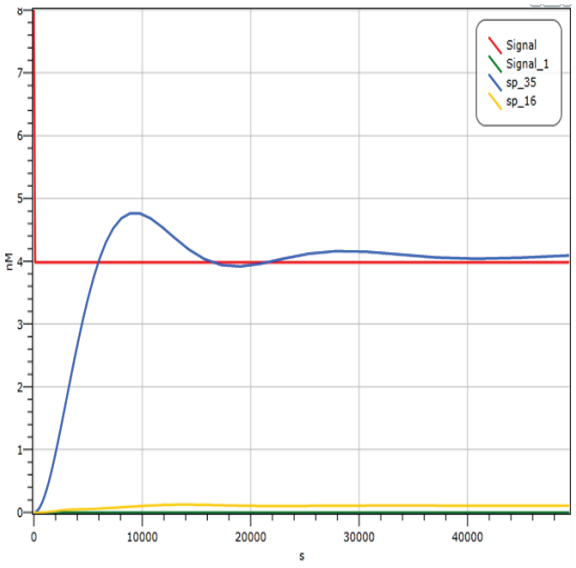


Figure 1: Step response of a constant gain system under a so-called PI controller; this system is explained in [2]. Output of this idealized DSD implementation using Visual DSD software ([6]) is structurally similar to the output of an in silico implementation of PI controller, e.g., using MATLAB, except for the fact that the time-scale is quite large. The reason why the time scale is so large is due to the fact that the DSD reactions cannot proceed arbitrarily fast.

unprotected DNA strands, no DNA strands are required to implement the degradation. Therefore, by introducing a few enzymes and monomeric fuel species to replace DNA gates for catalysis, degradation, and fuel species, the number of DNA strands required is greatly reduced in these hybrid systems.

To outline the improvement, consider the number of DNA strands required to implement the catalysis reaction $x_i \xrightarrow{\gamma} x_i + x_j$ via a 4-domain **DSD implementation**:

$$x_i + G_k \xrightarrow{q_k} O_k, \quad O_k + T_k \xrightarrow{q_{max}} x_i + x_j.$$

In the DSD implementation, G_k consists of two strands: a fuel species O_k and a bottom strand complementary to x_i; T_k consists of three strands: the input species x_i, the output species x_j, and a bottom strand complementary to O_k. Together, three DNA strands are spent for each catalysis reaction: the fuel species O_k and the two bottom strands for G_k and T_k. The output strand x_j needs to be provided as part of T_k, increasing the total number of DNA strands to 4 unless x_j is accounted for in the realization of other chemical reactions. Finally, if the so-called history domain for the original input x_i and x_i that was trapped in T_k are different, then the total number of DNA strands for catalysis reaction increases to 5. For a 2-domain DSD implementation even more strands are required but their structure is simplified.

Now, consider the **DNA toolbox implementation**:

$$x_i + T_{ij} \xrightarrow{q_{max}} x_i : T_{ij}, \; x_i : T_{ij} \xrightarrow{k_{pol}} x_i x_j : T_{ij},$$
$$x_i x_j : T_{ij} \xrightarrow{k_{nick}} x_i : x_j : T_{ij}, \quad x_i : x_j : T_{ij} \xrightarrow{k_{pol}} x_i x_j : T_{ij} + x_j.$$

By introducing the template T_{ij} consisting of domains $x_j^* x_i^*$, x_i serves as the primer for DNA polymerization. Once the polymerization is complete forming a duplex $x_i x_j : T_{ij}$, which also contains recognition site for nicking enzyme, the top strand $x_i x_j$ is broken into two parts x_i and x_j. Subsequent polymerization reactions from the free $3'$ end of x_i can displace x_j from the complex [7]. Therefore, only one DNA strand is required to implement catalysis reaction in the DNA toolbox, greatly reducing the number of DNA strands as compared with DSD implementation. Likewise, the degradation reaction can be implemented using exonuclease without any increase in the number of DNA strands required. The DSD mechanism to implement the annihilation reaction can be used in DNA toolbox as well so long as the $3'$ ends of signal species x_i^+ and x_i^- are designed to be not complementary to the gate species, thereby preventing a non-specific polymerization.

Such programmable interactions of DSD combined with catalytic power of enzymes allows for a simpler implementation of linear dynamical systems. To illustrate the improvement, consider the PI controller: here, a total of 10 catalysis reactions, 4 degradation reactions, and 4 annihilation reactions are used [2]. Using our approach, the number of DNA strands needed to realize it can be reduced by at least 27; this significantly reduces the expected crosstalk. Simulations in the Visual DSD software, extended to include enzyme reactions, show that the combination of DSD and enzyme reactions speeds up the response time of PI controller.

3. REFERENCES

[1] Soloveichik, D., Seelig, G. and Winfree, E.: 'DNA as a universal substrate for chemical kinetics', Proceedings of the National Academy of Sciences, 2010, 107, (12), pp. 5393–5398

[2] Oishi, K. and Klavins, E.: 'Biomolecular implementation of linear I/O systems', IET Systems Biology, 2011, 5, (4), pp. 252-260.

[3] Zhang, D. and Seelig, G.: 'Dynamic DNA nanotechnology using strand-displacement reactions', Nature Chemistry, 2011, 3, (2), pp. 103–113

[4] Qian, L., Winfree, E.: 'Scaling up digital circuit computation with DNA strand displacement cascades', Science, 2011, 332, (6034), pp. 1196–1201

[5] Padirac, A., Fujii, T., and Rondelez, Y.: 'Nucleic acids for the rational design of reaction circuits', Current Opinion in Biotechnology, 2012, http://dx.doi.org/10.1016/j.copbio.2012.11.011

[6] Lakin, M. R., Youssef, S., Cardelli, L., and Phillips, A.: 'Abstractions for DNA circuit design', J. R. Soc. Interface, 2011, 9, (68), pp. 470–486

[7] Montagne, K., Plasson, R., Sakai, Y., Fujii, T., and Rondelez, Y.: 'Programming an in vitro DNA oscillator using a molecular networking strategy', Molecular Systems Biology, 2011, 7, 466

ACM Word Template for SIG Site

1st Author
Name:
Ehud Magal
Affiliation:
Weizmann Institute Of Science
Prof. Ehud Shapiro's Lab.
Software Developer
Work address:
Hertzel St. 234
Rehovot PIN code 76100, Israel
Phone number:
00972543142214
Email:
ehudmagal@gmail.com

ABSTRACT
In this paper I will describe quality control methodologies

for liquid handling robots developed by me during

my work at Weitzman Institute Of Science.

Categories and Subject Descriptors
Category: automation, quality control.

Subject: monitoring and analyzing performance

of Liquid handling operations executed by robots.

General Terms

Measurement of accuracy of liquid handling robots.

Keywords

Liquid handling robots, automation, QC

1.INTRODUCTION
During my work at the Shapiro lab with robotics, I came across a

fundamental problem:

robots are programmed to execute various tasks. There is no simple methodology for monitoring weather the robot has succeeded or failed in its operations.

As a result it is very hard to debug biological experiments performed on robots, since there is an elements of robotic error that is not accounted for.

1. Robot QC methodologies:
1.1 Our options:
When we thought of the ways to solve the problem, we thought of

2 ways measuring pipetting accuracy: weight and light absorbance. Weight is the old traditional way which has many limitations. For example: the smaller volume you deal with, weight becomes harder.

The second way is light absorbance. This enables us to use small volumes , easily compare our work to manual work and mimic our everyday operations in accuracy tests..

1.2 What we basically do:
Basically, we test the robot operation as it runs in our regular experiments with one difference: we use food color instead of biochemical materials. After that,

We do the same process manually.

Both manual and robot operations are then measured by a Tecan Plate reader.

The reports are uploaded to a server that produces graphic reports.

According to these reports we can tell if we succeeded or failed

and use these reports for tuning the robot and keeping it in top shape, which is critical for biological applications..

1.3 How do we save our data and generate graphic reports
I have built a web server on google app engine: http://robotqcapp.appspot.com/.

User: udilab

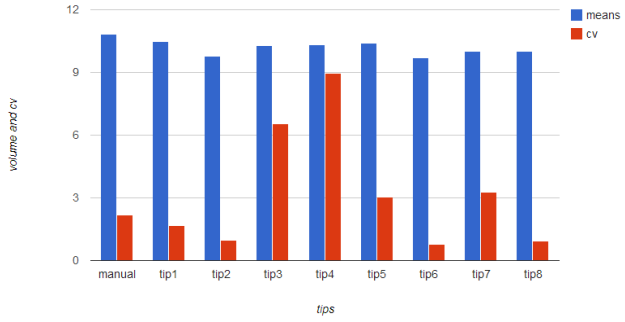
Password: udilab

The server is written in Django. It stores all the experiments we do. For each experiment, the server generates a battery of graphic reports by Google charts. All files connected to the experiment could be downloaded from the web server.

1.4 Example

Assume we want to test an operation where we aspirate 10 ul from eppendorf tube and dispense it into a pcr-biorad plate.

We run the test as described above. Some of the results:



1.5 Explanation:

Each graph has 9 columns. Left most column is performed manually to test against the robot. Each column has a blue part for volume and red part for CV. CV stands for coefficient of variation which is the standard defined by Tecan for measuring accuracy:

http://www.tecan.com/platform/apps/product/index.asp?MenuID=2696&ID=5273&Menu=1&Item=21.1.8.

1.6 additional features:

Just recently, I have added a new important feature to my application.

Problem:

assume a robot technician runs 10 tests to tune the robot for a single operation. Assume 5th test has the best result. Now the technician wants to return to the configuration his robot had in the 5th test.

Solution:

Now, for each test, the configuration of the liquid class* is saved on the server and the user can roll back to the liquid class configuration on the best test without affecting any other robot configurations.

This feature works only on Tecan robots.

2. Conclusions:

The web application I have built provides us with a powerful, easy to use, cheap tool to tune and maintain liquid handling robots.

I would like to present this abstract orally and by poster at the IWBDA since, with the increasing prevalence of robots in modern biological research, liquid handling robot accuracy and automated monitoring is becoming a major issue with widespread application.

3. ACKNOWLEDGMENTS

my thanks go Google app engine to host my application for free.

My thanks also go to the Weitzman Institute of science to give me the opportunity To develop my application in their labs.

*a liquid class is a software object that defines liquid handling on Tecan robot.