12th International Workshop on Bio-Design Automation
Worcester Polytechnic Institute, Worcester, MA, USA
Online
August 3rd-5th 2020

Foreword

Welcome to IWBDA 2020!

The IWBDA 2020 Executive Committee welcomes you to the Twelfth International Workshop on Bio-Design Automation (IWBDA). IWBDA 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 synthesis of biological systems.

The field of synthetic biology, still in its early stages, has largely been driven by experimental expertise, and 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. Inspired by these challenges, the field has seen a proliferation of efforts to create computer-aided design tools addressing synthetic biology's specific design needs, many drawing on prior expertise from the electronic design automation (EDA) community. IWBDA offers a forum for cross-disciplinary discussion, with the aim of seeding and fostering collaboration between the biological and the design automation research communities.

The workshop was originally intended to be held at Worcester Polytechnic Institute in Worcester, Massachusetts, but was moved to an online forum in response to the COVID-19 pandemic. In order to adapt the proceedings to this new venue, the program this year differs in a few different respects from previous years. The daily schedule was shortened so that it could accommodate participants across American and European timezones. As a result, fewer talks were accepted this year. In addition, workshops and breakouts have been integrated into the daily schedule to provide opportunities for active engagement to balance the talk sessions. Finally, the traditional poster presentations have been replaced with thematic panel sessions consisting of mini-talks and Q&A.

This year, the program consists of 14 contributed talks organized into 3 sessions: Design Automation, Pipelines, and Circuits & Modeling. The panel sessions consist of 19 presenters organized into 6 topic groups: Design Abstraction, Sequence Design, Microfluidics, Knowledge Engineering, the Synthetic Biology Open Language, and Metabolic Engineering. In addition, we are very pleased to have two distinguished keynote speakers: Dr. Alec Nielsen, Founder and CEO of Asimov, Inc., a synthetic biology pioneer in full-stack biological engineering, and Dr. Nili Ostrov, a post-doctoral fellow at Harvard Medical School, expert in microbial genetics, and leader in the Genome Project Write (GP-write) genome engineering consortium.

IWBDA is proudly organized by the non-profit Bio-Design Automation Consortium (BDAC). BDAC is an officially recognized 501(c)(3) tax-exempt organization.

We would like to thank all the participants for contributing to IWBDA. We would also like to thank the Program Committee for reviewing the abstracts and everyone on the Executive Committee for their time and dedication. Finally, we would like to thank BBN Technologies for their sponsorship.

Sponsors

“Algorithm” Level

Raytheon
BBN Technologies

Organizing Committee

Executive Committee

General Chair - Eric Young, Worcester Polytechnic Insitute

Local Chair - Natalie Farny, Worcester Polytechnic Institute

Program Committee Chair - Bryan Bartley, BBN Technologies

Publication Chair - Bryan Bartley, BBN Technologies

Co-Web Chair - Aaron Adler, BBN Technologies

Co-Web Chair - Prashant Vaidyanathan, Microsoft Research

Finance Chair - Traci Haddock-Angelli, iGEM Foundation

Bio-Design Automation Consortium

President - Aaron Adler, BBN Technologies

Vice-President - Natasa Miskov-Zivanov, University of Pittsburgh

Treasurer - Traci Haddock, iGEM Foundation

Clerk - Prashant Vaidyanathan, Microsoft Research

Program Committee

Aaron Adler	BBN Technologies
Bryan Bartley	BBN Technologies
Jacob Beal	BBN Technologies
Swapnil Bhatia	Boston University
Thomas Gorochowski	University of Bristol
Matthew R. Lakin	University of New Mexico
Curtis Madsen	Sandia National Laboratories
Chris Myers	University of Colorado
Ernst Oberortner	DOE Joint Genome Institute
Luis Ortiz	Boston University
Irene Otero Muras	IIM-CSIC Spanish Council for Scientific Research
Zach Palchick	Zymergen
Dimitris Papamichail	The College of New Jersey
Nicholas Roehner	BBN Technologies
Miles Rogers	BBN Technologies
Howard Salis	The Pennsylvania State University
Khaled Sayed	University of Pittsburgh
Neil Swainston	University of Liverpool
Allison Taggart	BBN Technologies
Jenhan Tao	University of California San Diego
Cheryl Telmer	Carnegie Mellon University
Prashant Vaidyanathan	Microsoft Research
Paolo Zuliani	Newcastle University

Program

Monday, August 3rd

10:30 - 10:45 **Welcome & Opening Remarks** Eric Young (WPI)

10:45 - 10:50 **Introduction to BDAthalon** Prashant Vaidyanathan (Microsoft Research)

10:50 - 12:10 **Session I: Design Automation**, Chair: Marilene Pavan

- 10:50-11:10 *gRNA-SeqRET: Genome-wide guide RNA design and sequence extraction*

  Lisa Simirenko, Ernst Oberortner, Ian K. Blaby, and Jan-Fang Cheng
- 11:10-11:30 *Genetic Circuit Design Automation involving Structural Variants and Parameter Statistics*

  Tobias Schladt, Erik Kubaczka, Nicolai Engelmann, Christian Hochberger, and Heinz Koeppl
- 11:30-11:50 *Laboratory Protocol Automation: A Modular DNA Assembly and Bacterial Transformation Case Study*

  Rita Chen, Nicholas Emery, Marilene Pavan, and Samuel Oliveira
- 11:50-12:10 *SBOLCanvas: A Visual Editor for Genetic Designs*

  Logan Terry, Jared Earl, Sam Thayer, Samuel Bridge, and Chris Myers

12:10 - 13:00 **Meal / Zoom Social Hour**

13:00 - 14:45 **SBOL3 Workshop**

15:00 - 16:00 **Live Keynote I: Machine-Guided Design of Genetic Circuits**, Alec Nielsen, PhD (Asimov, Inc.)

16:00 - 17:00 **Mini-talks and Panel discussions**

- **Design Abstraction**
  - *BioCRNpyler: Compiling Chemical Reaction Networks from Parts in Diverse Contexts with Python*

    William Poole, Ayush Pandey, Andrey Shur, Zoltan Tuza, and Richard Murray
  - *Describing engineered biological systems withSBOL3 and ShortBOL2*

    Matthew Crowther, Lewis Grozinger, James McLaughlin, Goksel Misirli, Jacob Beal, Bryan Bartley, Angel Goni-Moreno, and Anil Wipat
  - *SBModEns: A Modular Toolbox for Model Building, Reduction, Analysis and Simulation in System Biology*

    Fernando Nóbel Santos Navarro, Jesús Picó, and Jose Luis Navarro
- **Sequence Design**
  - *Accurate, Complete, and Contiguous Engineered Yeast Genomes with Prymetime*

    Joseph Collins, Kevin Keating, Tom Mitchell, Bryan Bartley, Nicholas Roehner, and Eric Young

- *Decodon Calculator: Degenerate Codon Set Design for Protein Variant Libraries*

  Dimitris Papamichail, Nicholas Carpino, Tomer Aberbach, and Georgios Papamichail
- *Automation of polycistronic small RNA design through Golden Gate assembly*

  Uriel Urquiza-García, Christoph Wagner, Sascha Ferraro, and Matias Zurbriggen
- *Detecting Co-Occurring Signatures of Engineering in Single Cells with Targeted Sequencing*

  Aaron Adler, Adam Abate, Brian Basnight, Joseph Collins, Benjamin Demaree, Kevin Keating, Xiangpeng Li, Tyler Marshal, Thomas Mitchell, David Ruff, Allison Taggart, Shu Wang, Daniel Weisgerber, Eric Young, and Nicholas Roehner

- **Microfluidics**
  - *Active Learning for Efficient Microfluidic Design Automation*

    David McIntyre, Ali Lashkaripour, and Douglas Densmore
  - *Efficient Large-Scale Microfluidic Design-Space Exploration: From Data to Model to Data*

    Ali Lashkaripour, David McIntyre, and Douglas Densmore
  - *A Droplet-Based Microfluidic Lab Automation for Biosynthetic Pathway Optimization*

    Kosuke Iwai, Megan Garber, Jess Sustarich, Peter W. Kim, William R. Gaillard, Kai Deng, Trent Northen, Hector Garcia-Martin, Paul D. Adams, and Anup K. Singh

Tuesday, August 4th

10:30 - 12:10 **Session II: Pipelines**, Chair: Bryan Bartley

- 10:30-10:50 *Design Automation Workflows for Synthetic Biology and Metabolic Engineering: The Galaxy-SynBioCAD Portal*

  Jean-Loup Faulon, Thomas Duigou, Melchior du Lac, Joan Hérisson, and Pablo Carbonell
- 10:50-11:10 *Automation of a DOE Design Workflow in Synthetic Biology - A Comparative Study*

  Alexis Casas, Charles Motraghi, Matthieu Bultelle, and Richard Kitney
- 11:10-11:30 *Round-Trip: An Automated Pipeline for Experimental Design, Execution, and Analysis*

  Daniel Bryce, Robert P. Goldman, Matthew Dehaven, Jacob Beal, Tramy Nguyen, Nicholas Walczak, Mark Weston, George Zheng, Josh Nowak, Joe Stubbs, Matthew Vaughn, Niall Gaffney, and Chris Myers
- 11:30-11:50 *Integrated Decision-Making to Detect DNA Engineering in Yeast*

  Sancar Adali, Aaron Adler, Joel Bader, Joseph Collins, Yuchen Ge, John Grothendieck, Thomas Mitchell, Anton Persikov, Jonathan Prokos, Richard Schwartz, Mona Singh, Allison Taggart, Benjamin Toll, Stavros Taskalidis, Daniel Wyschogrod, Fusun Yaman, Eric Young, and Nicholas Roehner
- 11:50-12:10 *The Synthetic Biology Knowledge System*

  Jeanet Mante, Chris Myers, Eric Yu, Mai H. Nguyen, Gaurav Nakum, Jiawei Tang, Xuanyu Wu, Eric Young, Kevin Keating, Bridget T. McInnes, Nicholas E. Rodriguez, Jacob Jett, J. Stephen Downie, Brandon Sepulvado, and Logan Terry

12:10 - 13:00 **Meal / Zoom Social Hour**

13:00 - 14:45 **BDAthlon breakouts**

15:00 - 16:00 **Live Keynote II: Reading and Writing Non-canonical Microbial Genomes**, Nili Ostrov, PhD

16:00 - 17:00 **Mini-talks and Panel discussions**

- **Knowledge Engineering**
  - *Intent Parser: a tool for codifying experiment design*

    Tramy Nguyen, Nicholas Walczak, Jacob Beal, Daniel Sumorok, and Mark Weston
  - *Collaborative Terminology: SBOL Project Dictionary*

    Jacob Beal, Daniel Sumorok, Bryan Bartley, and Tramy Nguyen
  - *The Social and Conceptual Organization of Synthetic Biology Ethics*

    Brandon Sepulvado, Jacob Jett, and J. Stephen Downie

- *Discovering Content through Text Mining for a Synthetic Biology Knowledge System*

  Mai Nguyen, Bridget McInnes, Eric Young, Gaurav Nakum, Jiawei Tang, Xuanyu Wu, Nicholas Rodriguez, and Kevin Keating

- **Synthetic Biology Open Language**
  - *VisBOL 2.0 - Improved Synthetic Biology Design Visualization*

    Benjamin Hatch, James McLaughlin, James Scott-Brown, and Chris Myers
  - *Sequence-based Searching For SynBioHub Using VSEARCH*

    Eric Yu and Chris Myers
  - *Analysis of the SBOL iGEM Data Set*

    Jeanet Mante, Chris Myers, and James Mclaughlin
- **Metabolic Engineering**
  - *Dynamic pathway regulation: extended biosensor and controller tuning with multiobjective optimization*

    Yadira Boada, Alejandro Vignoni, Ana Fraile, Jesús Picó, and Pablo Carbonell
  - *Enhanced Microbial Production of Valuable Natural Products Through Computational Metabolic Models*

    Michael Cotner, Zhen Zhang, and Jixun Zhan

Wednesday, August 5th

10:30 - 12:10 **Session III: Circuits & Models**, Chair: Thomas Gorochowski

- 10:30-10:50 *Multistable and dynamic CRISPRi-based synthetic circuits*

  Javier Santos-Moreno, Eve Tasiudi, Joerg Stelling, and Yolanda Schaerli
- 10:50-11:10 *Robust control of biochemical reaction networks via stochastic morphing*

  Tomislav Plesa, Guy-Bart Stan, Thomas Ouldridge, and Wooli Bae
- 11:10-11:30 *Genetic Circuit Hazard Analysis Using STAMINA*

  Lukas Buecherl, Jeanet Mante, Zhen Zhang, Brett Jepsen, Riley Roberts, Pedro Fontanarrosa and Chris J. Myers
- 11:30-11:50 *Minimal model for protein expression accounting for metabolic burden*

  Fernando Nóbel Santos Navarro, and Jesús Picó
- 11:50-12:10 *Bacteria mastering the tic-tac-toe game through synthetic adaptive gene circuits*

  Adrian Racovita, Satya Prakash, Clenira Varela, Mark Walsh, Roberto Galizi, Mark Isalan, and Alfonso Jaramillo

12:10 - 13:00 **Meal / Zoom Social Hour**

13:00 - 14:45 **SBOL Visual workshop**

15:00 - 16:00 **Guided Discussion**, Chair: Prashant Vaidyanathan

16:00 - 17:00 **Awards / Closing**

Keynote Presentation

Machine-guided Design of Genetic Circuits

Alec Nielsen, PhD

Keynote Abstract

Cells use genetically encoded circuits to regulate metabolism, communicate with each other, and generate spatial patterns. Synthetic genetic circuits enable advanced biotechnology applications, but are challenging to design for many reasons: a lack of high-performance genetic parts, inaccurate biophysical simulations, and inefficient algorithms for searching the genetic design space. In this talk, we build upon our previous work in genetic circuit design automation by developing improved biophysical simulators and genetic algorithm-based circuit generators. Using this approach, we achieve state-of-the-art performance in computational circuit design and generalize the method to analog and temporal circuits.

Speaker Biography

Alec Nielsen is co-founder and CEO of Asimov, a Boston-based mammalian synthetic biology company that spun out of MIT in 2017. Alec holds a B.S. in Electrical Engineering and Bioengineering from the University of Washington and a Ph.D. from MIT Biological Engineering. His work focuses on computer-aided design of complex cellular functions, scalable biochemistries for synthetic biology, and machine learning applications in genetic design and biosecurity.

Keynote Presentation

Reading and Writing Non-canonical Microbial Genomes

Nili Ostrov, PhD

Keynote Abstract

The ability to make radical and comprehensive genomic changes opens new avenues for understanding biological principles and for construction of synthetic genomes not found in nature. In this talk, I will discuss development of high-throughput methods for reading and writing entire microbial genomes. I will describe methods for 'bottom up' writing of a virus-resistant E. coli with an altered genetic code. In addition, I will present enabling genetic tools for rapid 'top down' reading of gene function in the non-model marine bacterium Vibrio natriegens, the fastest dividing free-living organism known. These projects demonstrate the need for robust systems for large-scale genome manipulations to accelerate design-build-test of non-canonical organisms.

Speaker Biography

Dr. Nili Ostrov is a postdoctoral fellow in the laboratory of Prof. George Church at Harvard Medical School, where she is constructing synthetic microbial genomes. She is broadly interested in using non-canonical organisms for clinical, agricultural and bioindustrial applications. Nili is a leading member of the Technology Working Group for the Genome Project Write (GP-write) international consortium.

Oral Presentations

Poster Presentations

gRNA-SeqRET: Genome-wide guide RNA design and sequence extraction

Lisa Simirenko, Ernst Oberortner, Ian K. Blaby, Jan-Fang Cheng
DOE Joint Genome Institute (JGI)
{lsimirenko,eoberortner,ikblaby,jfcheng}@lbl.gov

INTRODUCTION

The U.S. Department of Energy (DOE) Joint Genome Institute (JGI) provides DNA sequencing and synthesis services to scientific users via community science programs. The DNA synthesis program covers all aspects of the synthetic biology design-build-test cycle, enabling users to engineer biological systems that are relevant to their research and the DOE mission. One particular product type that the DNA synthesis program offers is the design and construction of libraries with a high degree of variants. Applications of such libraries include, but are not limited to, CRISPR guide RNA (gRNA) libraries for studying the phenotypic changes as a result of changing gene expression, such as demonstrated by Schwartz *et al.* [1].

Here, we present our recently developed *in silico* design workflow for library variants: guide RNA and Sequence Region Extraction Tool (gRNA-SeqRET). The current version enables (i) designing gRNA libraries that target user-specified regions and (ii) extracting any arbitrary sequence region by either keyword or by coordinates within a genome or the whole genome of any prokaryotic organism. A common JGI use case, for example, is the design of gRNA libraries that target the up- and downstream regions of each gene's start codon for respectively activating ("CRISPRa") or interfering ("CRISPRi") gene expression. For such designs, gRNA-SeqRET searches for gRNA variants for each gene in the whole genome and scores and filters the discovered variants based on predicted folding and potential off-target bindings. The gRNA-SeqRET pipeline comprises freely-available software tools and customized Python scripts, and is available at https://grna.jgi.doe.gov under a modified BSD open source license (https://bitbucket.org/berkeleylab/grnadesigner).

RESULTS & DISCUSSION

Due to the wide range of host organisms and type of projects that our users work on, the specification for our tool included the ability to find gRNA sequences for any genome, and automatically select sequence regions both upstream and downstream of the genes of interest. Another requirement was to evaluate ("score") and filter out the top gRNAs based on predicted RNA folding when attached to the scaffold sequence. To address these requirements, we surveyed previously developed CRISPR/gRNA design tools and decided in favor of the CRISPR/Cas9 Target Online Predictor (CCTop) [2] for integration into our gRNA design workflow. CCTop is open-source, easy to integrate and close to meeting our requirements.

To date, we have successfully evaluated gRNA-SeqRET on several user projects. In two projects, for example, we utilized the gRNA library design feature including (i) whole genome gRNA libraries targeted for *Pseudomonas putida* consisting of 12,000 variants and (ii) both CRISPRi and CRISPRa gRNA libraries for 103 transcription factors in *Nannochloropsis oceanica*. In another user project, we utilized gRNA-SeqRET to extract sequences up- and downstream of respectively the start and stop codons of 45 genes in order to design both N-terminal and C-terminal fluorescent tagged knock-in constructs for *Arabidopsis thaliana*.

METHODS

The first step in the design process (see Figure 1) is collecting the user inputs through a web user interface (UI). The user uploads the genome sequence in GenBank format and inputs the name of the targeted genes, if not targeting all the genes in the genome. The region of the gene to be targeted must be specified by indicating the start and stop coordinates relative to the start codon of each gene. For CRISPRa, for example, the promoter region upstream of the start codon will be targeted, represented by a negative number indicating the number of upstream base pairs. For CRISPRi, on the other hand, the region downstream of the start codon is targeted, represented by a positive number. The protospacer adjacent motif (PAM) sequence and the scaffold sequence for the CRISPR associated (Cas) protein must also be provided, as well as the length of the gRNA variants and how many are to be designed for each gene.

Once the user submits these inputs, the application prepares the inputs for CCTop, runs CCTop, and post-processes the CCTop results. The web UI communicates with the backend asynchronously since, depending on the inputs and and the size of the genome, the design process can take several hours. Preparing the CCTop inputs involves converting the user inputs into CCTop inputs. The gRNA-SeqRET backend converts the genome information from the uploaded

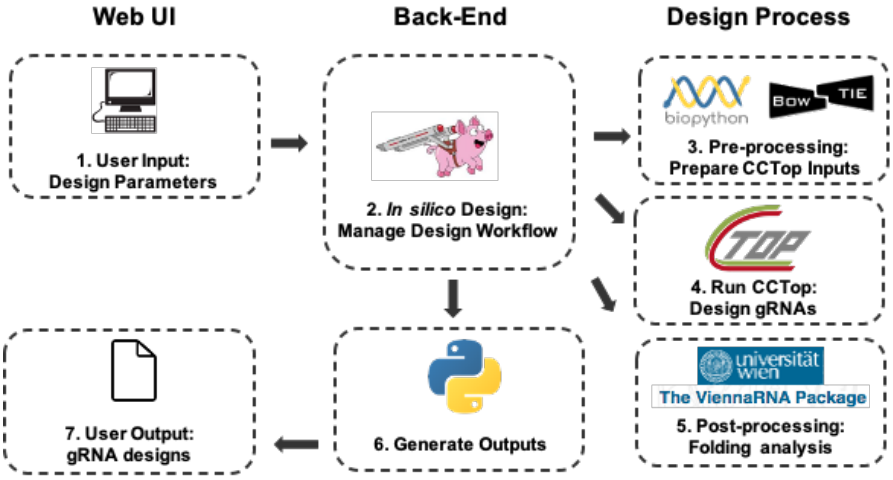


Figure 1: The front-end ("web UI") allows for user input and download of the final outputs. The input is processed asynchronously on the back-end. The pre-processing step converts the web UI inputs into CCTop inputs. After running CCTop, the post-processing step filters and scores each gRNA. Lastly, a custom Python script collects the pipeline outputs and makes them available for download on the web UI.

GenBank file into (i) an indexed version of the genome sequence using bowtie, (ii) a Browser Extensible Data (BED) file containing all the genome annotation and (iii) a FASTA file comprising the sequence regions of the genes that will be targeted. These three files, along with the PAM site, the desired gRNA length, and other CCTop parameters serve as input to CCTop. After a successful CCTop run, a comma separated value (CSV) file is available for each gene of interest with information about each gRNA's scores and offsite target information. CCTop provides two scores for each gRNA. One which takes into account the number of off-target sites in the genome and their quality (i.e. the number of mismatches and position with respect to the PAM), and a CRISPRater score [3] that purports to be a measure of the gRNA efficacy, based on experimental data. In addition to CCTop's scoring algorithms, we evaluate and score each gRNA based on its predicted folding. The goal here is to select gRNAs that (i) will not form hairpins and therefore increase the likelihood of binding to its intended target and (ii) do not interfere with the folding of the scaffold sequence to which it will be fused. The folding score is calculated by counting the number of unpaired bases in the gRNA when fused to the scaffold. The final score of any gRNA is the product of the two CCTop scores and our folding score. The gRNAs are filtered based on the top scores and the desired number of gRNAs for each gene, as specified by the user. The final output of the gRNA-SeqRET design process is written to a CSV file with six columns: (i) a unique gRNA name which incorporates the targeted gene name, (ii) the gRNA sequence, (iii) the PAM sequence, and the location of the gRNA in the genome indicated by (iv) the start coordinate, (v) the end coordinate, and (vi) the strand. On the web UI, the user can download the CSV file. We also provide the download of all input and output files as a gzipped tarball (.tar.gz) file, including all the CCTop input and output files, execution logs and scoring information.

The current version of gRNA-SeqRET supports processing prokaryotic genomes. We are in the process of extending gRNA-SeqRET for handling eukaryotic genomes, taking into account exons and introns within genes. Also, we intend to integrate with genomic databases, such as the Genomes OnLine Database (GOLD, https://gold.jgi.doe.gov/) and the Integrated Microbial Genomes and Microbiomes (IMG/M, https://img.jgi.doe.gov/) system.

ACKNOWLEDGMENTS

This work has been conducted by the U.S. Department of Energy Joint Genome Institute, a DOE Office of Science User Facility, is supported under Contract No. DE-AC02-05CH11231.The gRNA-SeqRET design tool is hosted by DOE National Energy Research Scientific Computing Center (NERSC).

REFERENCES

[1] Schwartz *et al.* Validating genome-wide CRISPR-Cas9 function in the non-conventional yeast *Yarrowia lipolytica*. *Metab Eng.*, 5:102—110, Sep 2019.

[2] Stemmer *et al.* CCTop: An Intuitive, Flexible and Reliable CRISPR/Cas9 Target Prediction Tool. *PLOS ONE*, 10:1—11, 04 2015.

[3] Labuhn *et al.* Refined sgRNA efficacy prediction improves large- and small-scale CRISPR–Cas9 applications. *Nucleic Acids Research*, 46(3):1375—1385, 12 2017.

Genetic Circuit Design Automation involving Structural Variants and Parameter Statistics

Tobias Schladt*, Erik Kubaczka*, Nicolai Engelmann*, Christian Hochberger, Heinz Koeppl
TU Darmstadt, Germany
{schladt,hochberger}@rs.tu-darmstadt.de
erik.kubaczka@stud.tu-darmstadt.de,{nicolai.engelmann,heinz.koeppl}@bcs.tu-darmstadt.de

1 INTRODUCTION

Genetic design automation (GDA) parallels early efforts in electronic design automation (EDA) and recently also got to use state-of-the-art EDA tools to generate gene-regulatory circuits realizing simple combinational logic [4]. While historically EDA quickly ran into unmanageable computational complexity and hence devised clever approximate methods, current GDA problems are yet too small to require such approximations. In contrast to EDA's scalability, GDA suffers from insufficient accuracy of part and device models in their libraries. In particular, the design process does not account for cell-to-cell variability and context effects [1].

The contribution of this work is twofold. First, we demonstrate that better circuit topologies and gate assignments can be found compared to the ones suggested by traditional EDA tools, as for instance used in Cello [4]. Our main approach here is to efficiently enumerate all structural circuit variants. Second, we introduce parametric uncertainty in device models to mimic cell-to-cell variability and extend the circuit scoring function to account for the incurred output variability. Accordingly, two realizations of the same logic circuit showing same medians (or means) across all input assignments and hence leading to identical scores in the traditional setting, could now be scored very differently due to their difference in output variability.

2 SYNTHESIS OF GENETIC CIRCUITS

In contrast to electronic logic circuits, current approaches for combinational genetic circuits are highly limited. They rely on hand-crafted libraries of individual logic gates and their maximum size is limited by energetic aspects and toxic effects on their host cell [4]. Furthermore, today's EDA tools focus on minimizing area and delay, which is not suitable for genetic circuits as they do not consider the complex biological interactions. Thus, we propose a synthesis method to enumerate all circuit structure variants for the small logic problems solvable by genetic circuits.

Combinational logic networks can be represented as Directed Acyclical Graphs (DAG). Therefore, the problem of finding all structurally different implementations of a Boolean logic function is a DAG-enumeration problem, which quickly becomes infeasible due to its highly combinatorial nature. Thus, we intermediately limit the problem to the enumeration of all fanout-free circuit structures (every gate output is connected to exactly one gate input), simplifying enumeration and isomorphism checking. Redundancies on the gate level inherent to fanout-free circuits are then removed by post-processing, thus returning to a general DAG structure.

Table 1: Synthesis results for sample functions with deterministic parameters

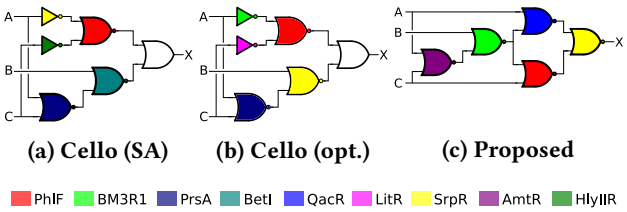
	Cello (SA)		Cello (opt.)		Proposed	
Funct.	**Size**	**Score**	**Size**	**Score**	**Size**	**Score**
0x4D	5	28.82	5	88.62	5	575.25
0x78	5	27.21	5	254.64	5	467.01
0xCD	4	40.86	4	162.07	4	575.72

To measure the benefit of including structural variety in genetic circuit synthesis, we synthesized all functions shown in [4] using Cello's library of genetic logic gates. We then used Cello's circuit score metric to rate the output identifiability of the synthesized circuits. Finally, we compared our results to the circuits synthesized by Cello. To cancel out the influence of non-optimal assignments of biological gates to circuit gates, we simulated all possible assignments exhaustively for both Cello's and our circuit structures.

Using our synthesis approach, we were able to improve the circuit score of 25 of the examined 33 functions, while no circuit performed worse than the corresponding circuit synthesized by Cello. On average, the scores improved by 73 % using approx. one additional gate. Table 1 shows the results for three sample functions synthesized by Cello using its simulated annealing gate assignment algorithm (SA), exhaustive gate assignment (opt.) and our proposed optimal synthesis approach. Figure 1 shows the resulting circuits for function 0x4D. The OR gate shown in figures 1a and 1b is an implicit combination of two output signals and thus not counted as a gate instance.

While the presented enumeration technique is only feasible for small circuits, it can also be used to generate a library of sub-circuits composed of gates of the biological gate library, so called Supergates [2]. These can be used to enable classical technology mapping approaches like structural mapping to produce a nearly complete variety of circuits.

* The authors contributed equally to this research.



(a) Cello (SA) (b) Cello (opt.) (c) Proposed

(d) Repressor legend

Figure 1: Synthesized circuits for function 0x4D

3 STATISTICAL EVALUATION OF GENETIC CIRCUITS

During circuit optimization Cello relies only on median transfer behaviour ignoring computational variability across single instances due to context effects. To remedy this, we incorporate statistical models during circuit optimization. Given statistics of sensor input concentrations, we compute approximate statistics of output concentrations involving quasi-steady state transfer functions based on random ODE models for the kinetics of the gates. To maintain a high degree of flexibility, we use stochastic simulation to evaluate circuits during optimization. To be precise, let a random vector $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ represent the environment a circuit is embedded in. Let thus $\boldsymbol{\theta}$ parameterize a kinetic ODE model $\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\theta})$ for vectors $\boldsymbol{u} \in \mathbb{R}^N_{\geq 0}$ of N input and $\boldsymbol{x} = (\boldsymbol{x}_m, \boldsymbol{x}_o)$ of arbitrarily many intermediate and M output species concentrations $\boldsymbol{x}_o \in \mathbb{R}^M_{\geq 0}$ of the circuit. We then find $\boldsymbol{x}_s = (\boldsymbol{x}_{m,s}, \boldsymbol{x}_{o,s})$ solving $0 = f(\boldsymbol{x}_s, \boldsymbol{u}, \boldsymbol{\theta})$ to obtain a quasi-steady state solution. Using this solution, we can uniquely determine $\boldsymbol{x}_{o,s}$ for particular realizations of $\boldsymbol{u}$ and $\boldsymbol{\theta}$ and find

$$p(\boldsymbol{x}_{o,s}) = \int_{\mathbb{R}^N_{\geq 0}} \mathrm{d}\boldsymbol{u} \int_{\boldsymbol{\Theta}} \mathrm{d}\boldsymbol{\theta}\, p(\boldsymbol{x}_{o,s} \mid \boldsymbol{u}, \boldsymbol{\theta})\, p(\boldsymbol{u}, \boldsymbol{\theta}) \qquad (1)$$

with $p(\boldsymbol{x}_{o,s} \mid \boldsymbol{u}, \boldsymbol{\theta}) = \delta(f(\boldsymbol{x}_s, \boldsymbol{u}, \boldsymbol{\theta}))$ given by a degenerate distribution. To inherit Cello's modularity in gate assignments, we factorize the joints $p(\boldsymbol{u}, \boldsymbol{\theta}) = p(\boldsymbol{u}) \prod_{\mathcal{G}} p(\boldsymbol{\theta}_{\mathcal{G}})$ and $p(\boldsymbol{x}_{o,s} \mid \boldsymbol{u}, \boldsymbol{\theta}) = \prod_{\mathcal{G}} \delta(f_{\mathcal{G}}(\boldsymbol{x}_s, \boldsymbol{u}, \boldsymbol{\theta}_{\mathcal{G}}))$ over gates $\mathcal{G}$ and the circuit's input $\boldsymbol{u}$. Despite of that, the exact calculation of (1) usually remains intractable and we therefore approximate the output statistics using a finite set of particular realizations (particles) representing the presumed true distributions. Since this raises the need of a circuit scoring metric involving probability distributions, we make use of the Wasserstein distance [3] to qualify a particular circuit, which generalizes Cello's scoring metric to non-degenerate distributions. In accordance to Cello, we determine the score by the minimum distance between logarithmic output distributions corresponding to complementary values of the circuit's Boolean function. Fig. 2 shows sample output distributions (standard deviations in table 2) and the score noise-power dependency.

Table 2: Empirical standard deviations of the logarithmic output distributions of circuit 0x1D (lesser values are bold)

$10\,\hat{\sigma}$	-/-/-	-/-/+	-/+/-	-/+/+	+/-/-	+/-/+	+/+/-	+/+/+
Proposed	5.6	5.7	**6.2**	**5.6**	**6.5**	**5.4**	**6.2**	**5.6**
Cello (opt.)	**5.3**	**5.3**	9.2	6.6	7.6	6.7	9.2	6.6

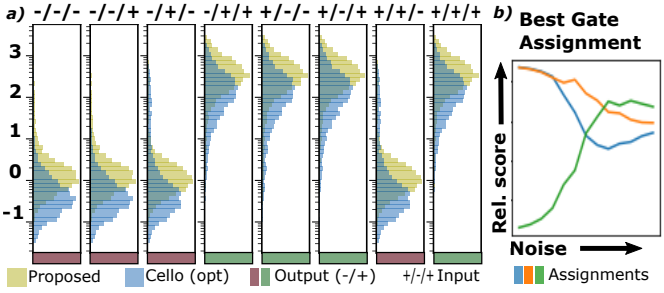


Figure 2: a) Output level histograms for circuit 0x1D. b) The best assignment depends on the variance of the parameters.

Table 3: Synthesis results for sample functions using the proposed statistical scoring incorporating structural variants

	Cello				Proposed			
Funct.	**Size**	$\hat{\sigma}$	$\Delta\mu$	**Score**	**Size**	$\hat{\sigma}$	$\Delta\mu$	**Score**
0x08	4	1.47	1.39	27.86	4	1.29	1.61	159.31
0xC4	3	1.42	3.04	67.5	3	1.17	3.29	219.07
0xCD	4	1.00	1.98	6.74	4	1.46	2.81	98.27

4 CONCLUSION

The new synthesis approach effectively extends the search space for robust circuits by including structural variants. In combination with the new scoring metric, we improve identifiability of the Boolean outputs by not only increasing the distance of complementary values in terms of expression levels but also preferring symmetric, low leakage solutions. This often comes with no cost in terms of increased circuit size. We give average standard deviations and distances between the means of complementary output distributions along with their score and circuit size for some examples in table 3.

REFERENCES

[1] Cardinale, S., and Arkin, A. P. Contextualizing context for synthetic biology – identifying causes of failure of synthetic biological systems. *Biotechnology Journal 7* (2012).

[2] Chatterjee, S., Mishchenko, A., Brayton, R., Wang, X., and Kam, T. Reducing structural bias in technology mapping. In *ICCAD, 2005.* (Nov 2005), pp. 519–526.

[3] Gibbs, A. L., and Su, F. E. On choosing and bounding probability metrics. *International Statistical Review 70* (2002).

[4] Nielsen, A. A. K., Der, B. S., Shin, J., Vaidyanathan, P., Paralanov, V., Strychalski, E. A., Ross, D., Densmore, D., and Voigt, C. A. Genetic circuit design automation. *Science 352*, 6281 (2016).

Laboratory Protocol Automation: A Modular DNA Assembly and Bacterial Transformation Case Study

Rita Chen[1], **Nicholas J. Emery**[2], **Marilene Pavan**[3], **Samuel M.D. Oliveira**[1]
[1]DAMP Lab - Boston University; [2]ASIMOV, Boston; [3]Lanzatech, Chicago

ABSTRACT

Molecular cloning and bacterial transformation are among the most used and essential molecular and cellular protocols in synthetic biology. Biologists make use of such protocols to assemble genetic constructs, one at a time, from a library of various genetic parts. Building multiple, complex constructs, simultaneously, requires the repetitive manual operations when exploring broader design space of genetic combinations. Novel automated frameworks have the potential to save time and resources while retaining standard and reproducible results. Here we developed open-source programming scripts that can auto-generate liquid handling protocols of modular cloning (MoClo), bacterial transformation, and cell plating protocols, which can be implemented automatically into Opentrons OT-2 liquid-handling robots. Once fully functional, these scripts will be part of an ongoing effort to develop software/hardware-based automation workflows for the assembly of larger genetic circuits, bacterial transformation, and performance tests in live cells in a high-throughput manner at the DAMP Lab.

INTRODUCTION

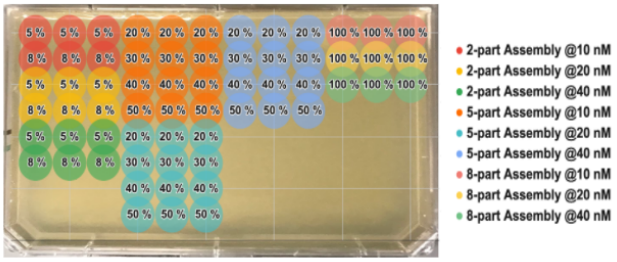
The ability to manipulate recombinant and synthetic DNA through molecular cloning has revolutionized biology over the past few decades [1]. Many application areas, particularly synthetic biology, would benefit from increased throughput in molecular cloning protocols. Therefore it has become an important goal for many academic and industry labs exploring large design spaces. Increased throughput has allowed biologists to explore a broader swath of biological devices with a particular purpose [2]. However, the lack of a standardized, reproducible, and scalable "Build and Test" tools and cycles have created a bottleneck in the experimental space aiming to test and construct various types and sizes of genetic devices. Automating molecular cloning processes using expensive robotic workstations has traditionally been applied to cope with this problem [3]. While automated liquid handling has traditionally been exclusive to industry and a few academic facilities due to the cost and complexity of using such a solution, recently developed platforms, such as the Opentrons OT-2 robot, provide similar capabilities within the financial reach of small research laboratories [4]. Given its affordability and flexibility, the OT-2 robot could be a good starting point for laboratories new to automation. Here, we propose to develop an end-to-end cloning pipeline using the OT-2 robot [5], to automate modular cloning (MoClo) [6], bacterial transformation, and cell plating, which can be easily replicated by other laboratories.

PROTOCOL AND REPRESENTATIVE RESULTS

The end-to-end cloning pipeline is designed to assist users in lacking programming experience while conducting biological experiments in a biology laboratory. It is composed of two layers, a generator script and a template script (Figure 1). Users of the robot can input two input-CSV files to the generator script, one containing all input-DNA part names, and the other describing the combinations of parts desired to be assembled in a final construct. A final protocol script is then generated for users to drag-and-drop into the Opentrons' OT-2 API. Furthermore, the pipeline can be performed in one whole-day tentative step (Figure 1A), or broken down into three separate experimental protocols (Figure 1B). This way, the modular cloning pipeline can be separated based on biological experimental designs, such as modular cloning, cell transformation, and cell plating.

Figure 1: End-to-end cloning pipeline consisted of multiple protocols, Modular Cloning (MoClo), Bacterial Cell Transformation, and Cell Plating in an automated workflow.

We verified the efficiency and accuracy of the pipeline by evaluating the modular cloning protocol results using the OT-2 robot. We first developed a MoClo OT-2 liquid handling protocol script for the assembly of 63 DNA constructs simultaneously, to explore an ample design space of various input DNA concentrations (10, 20, and 40 nM) and three similarly sized genetic constructs (2955, 3152, and 3285 bp), which are composed of two-part, five-part, and eight-part DNA assemblies respectively (Figure 2). Each reaction is cloned in triplicates via the MoClo DNA assembly methodology [6] with a total reaction volume of 20 uL to account for a reduction in the consumables' cost [3,7]. To validate cloning efficiency, we used a benchtop thermocycler. We tested two MoClo protocols with different thermocycling methods, i.e., traditional and isothermal (Figure 2). Optimal transformed cell plating dilution ratios, for 2-part, 5-part, and 8-part assemblies, were incorporated into the scripts with the respective values 3%, 30%, and 100%. Results show that the traditional method yields the optimal cloning efficiency of the highest desired white CFU yields (Figure 2B,C).
We then tested the ability of the pipeline to automate both bacterial transformation and cell plating. We developed one OT-2 liquid handling protocol script to test bacterial transformation and efficiency validation using the DNA constructs, and another to test cell plating and cross-contamination assessment using both normal *E. coli* bacterial cells (white CFU) and *E. coli* cells expressing the violacein



Traditional Thermocycling White CFU Count			Isothermal Thermocycling White CFU Count		
	Mean (±SEM)	Efficiency		Mean (±SEM)	Efficiency
2-part Assembly	42 (±4.04)	70%	2-part Assembly	25 (±8.08)	50%
5-part Assembly	18 (±3.4)	65%	5-part Assembly	6 (±3.4)	21%
8-part Assembly	6 (±1.15)	75%	8-part Assembly	0 (±0)	0%

Figure 2: OT-2 plating map with planned plating concentration and the corresponding locations. The table presents the mean and standard error of the mean of the total number of colonies per parts-assembly class, along with their cloning efficiency (compared to the largest number of white CFU counts), obtained from three biological replicates.

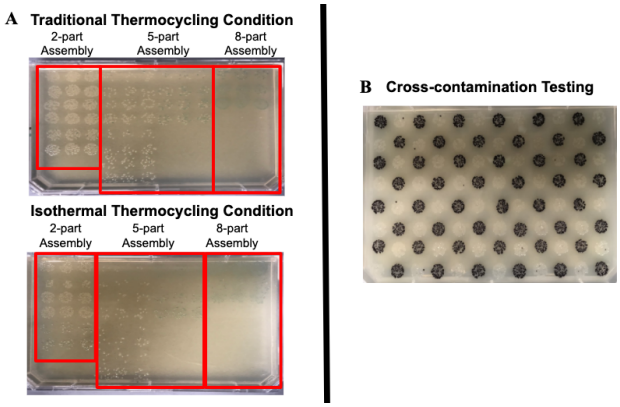


Figure 3: (A) Example plate result of the traditional method (35 cycles at 37°C for 1.5 min. and 16°C for 3 min. followed by one cycle at 50°C for 5 min. and 80°C for 10 min.) with 3 technical replicates per condition; and example plate result of the isothermal method (2 hours at 37°C followed by one cycle at 50°C for 5 min. and 80°C for 10 min.) with 3 technical replicates per condition. (B) Example plate result of the plate cross-contamination test with *E. coli* white colonies and *E. coli* purple colonies.

pathway (purple CFU). Once in the OT-2 robot, cells were heat-shocked, then recovered, diluted, and plated onto single-well plates. The OT-2 liquid handling protocol script for cell plating intercalates the two different cell types onto the same agar plate (Figure 3). By intercalating them, we expected to find potential plating errors, and reveal the robot's plating efficiency, corresponding to cross-contamination. After plating, cells were incubated overnight at 37°C. An example plate in Figure 3 show how the cell cross-contamination test was performed. From counting colonies from three biological replicates, we found the contamination rate to be on average 11.5% (11 single-colony cross-contamination events across 96 colony spots on the plate) using not-filtered OpenTrons' pipette tips. The optimal cell plating volume found was 10 uL.

CONCLUSION AND FUTURE WORK

In this work, we aimed to develop an easy-to-follow prototypical workflow that uses the OT-2 robot to automate the core processes of a typical molecular cloning workflow: assembling and incubating Modular Cloning (MoClo) reactions, transforming *E. coli* cells with MoClo products by heat shock, and the recovery and plating of transformed *E. coli*. We concluded that traditional thermocycling, yielding a higher cloning efficiency, should be adapted for OT-2 modular cloning protocol scripts. The highest percentage of white CFU was observed in the simplest DNA constructs (2-part assemblies), and the three similarly sized constructs were assembled. Thus, we can conclude that increasing the assembly complexity of the DNA construct decreases the percentage of desired white CFU yields. Finally, implementing filtered pipette tips for plating cells may play a role in reducing cross-contamination ratio. The open-sourced scripts developed are available on GitHub (https://github.com/DAMPLAB/OT2-MoClo-Transformation-Ecoli), along with detailed setup instructions for the experimental protocol and the robotic deck setup instructions. In the future, we aim to increase the throughput of such protocols (including internal controls, e.g., intact plasmids) and will provide additional information regarding the sequencing results of constructs obtained from automating colony picking and mini-prep protocols using OT2.

ACKNOWLEDGEMENTS

We thank Will Canine and Kristin Ellis from Opentrons for sponsoring the project with consumables and the OT-2 Robots, Luis Ortiz for proofreading the abstract, and Prof. Douglas Densmore and Prof. Mary Dunlop for supporting this work. This work was funded by NSF LCP Award 1522074.

REFERENCES

[1] Arturo Casini, et al. Bricks and blueprints: methods and standards for DNA assembly. 2015. Nature Reviews Molecular Cell Biology 16, 568-576.
[2] Evan Appleton, et al. Design Automation in Synthetic Biology. 2017. Cold Spring Harb Perspect Biol, 9, a023978.
[3] Luis Ortiz, et al. 2017. Automated Robotic Liquid Handling Assembly of Modular DNA Devices. Journal of Visualized Experiments 130, e54703.
[4] Marko Storch, et al. 2019. DNA-BOT: A low-cost, automated DNA assembly platform for synthetic biology. Oxford University Press Synthetic Biology, SYNBIO-2020-004.
[5] Opentrons 2020. https://opentrons.com (14 April 2020)
[6] Ernst Weber, et al. 2011. A Modular Cloning System for Standardized Assembly of Multigene Constructs. PLoS ONE 6, 2, e16765.
[7] David I. Walsh III, et al. 2019. Standardizing Automated DNA Assembly: Best Practices, Metrics, and Protocols Using Robots. SLAS Technology I-9.

SBOLCanvas: A Visual Editor for Genetic Designs

Logan Terry, Jared Earl, Sam Thayer, Samuel Bridge, Chris J. Myers
University of Utah
Salt Lake City, UT, USA
randoom97@live.com

1 INTRODUCTION

Synthetic biologists often use diagrams to visualize the structure and functionality of genetic designs due to their complicated nature. The *Synthetic Biology Open Language Visual* (SBOLv) [1] is a standard for these diagrams. This standard provides a set of glyphs for synthetic biology components and how they can interact. These visual designs also have a complementary data standard, the *Synthetic Biology Open Language* (SBOL) [3], which represents the structural and functional information for genetic designs.

When a synthetic biology designer is developing a genetic circuit with SBOL and SBOLv, they have three main objectives: 1) an ergonomic way to create and edit visual diagrams, 2) an ability to associate these diagrams with genetic part information, and 3) a means to share their designs with others. One such tool that can assist with these objectives is SBOLDesigner [4], a graphical schematic editor for DNA-level design. This tool has many useful features including the ability to construct a DNA sequence from SBOLv glyphs, import DNA part information from the SynBioHub repository [2], and share resulting designs by uploading them to SynBioHub. However, SBOLDesigner does not support the latest features in SBOLv Version 2. SBOLv2 allows for the inclusion of non-DNA components (RNAs, proteins, small molecules, etc.), as well as a way of representing interactions between them. Furthermore, SBOLDesigner requires local installation to use.

This paper describes SBOLCanvas, an updated web-based genetic design editor that can create visual diagrams using all features of SBOLv2. Specifically, in addition to features supported by SBOLDesigner, SBOLCanvas has the ability to:

- Create designs composed of multiple DNA sequences.
- Add non-DNA components to these designs.
- Link components via interactions such as genetic production, repression, and activation.
- Markup the designs with colors and text annotations.
- Undo and redo designs edits.

Therefore, SBOLCanvas provides a new way for synthetic biologists to specify and visualize the structure and function of their designs.

2 SPECIFIC FEATURES

Multiple Strands & Molecular Species

The SBOL data standard supports more than just single DNA circuits in a design. It allows for multiple circuits as well as denoting interactions between those circuits and other molecular species. Figure 1 demonstrates multiple circuits, interactions, and molecular species.

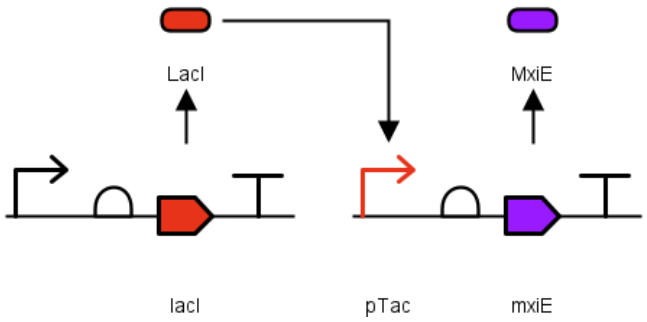


Figure 1: An image exported from SBOLCanvas.

SynBioHub integration

SBOLCanvas allows for direct access to designs stored on SynBioHub. Making it easy to integrate the parts you need into your design. It also allows for saving your own designs to SynBioHub for later access. SynBioHub has a feature that allows you to store commonly used parts in a collection making them easier to access within SBOLCanvas. In the future, we plan to make this even easier by allowing users to import this collection into the part menu, enabling the ability to drag and drop these parts into a design.

Image Exporting

If a researcher wants to build and share a diagram of their biological circuit, they currently have to use some graphical editor, such as Adobe Illustrator. This takes significantly more time than it should. SBOLCanvas lets you export images (.png, .jpeg, .svg, and .gif) of your design, merging visual and informational design into one task.

Ease of Use

SBOLCanvas lowers the barrier to entry by enabling anyone with access to a web browser to visually design synthetic

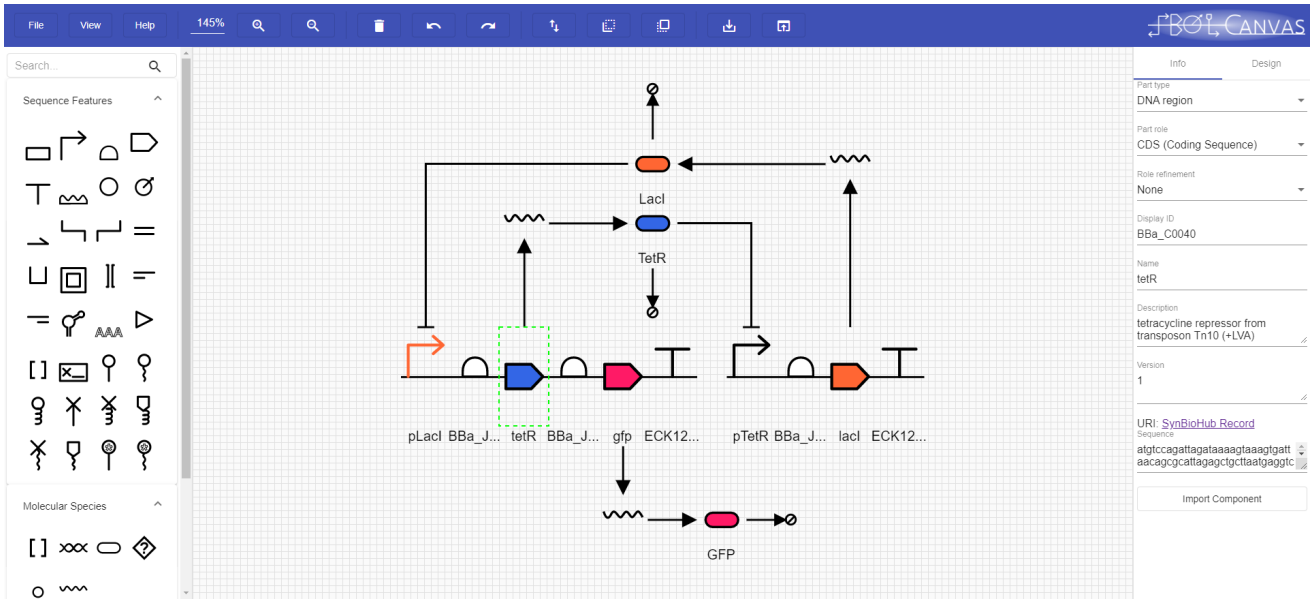


Figure 2: SBOLCanvas' graphical user interface.

genetic circuits. SBOLCanvas' graphical user interface (GUI) is shown in figure 2. The primary way to interact with SBOLCanvas is to drag and drop items from the part menu on the left. If a user has a strand selected, the user can also click on a glyph, and it is added to the end. Interactions can be added by selecting the two items to link, and then clicking the interaction to add from the part menu, or by moving the ends of existing interactions. For graphical work, SBOLCanvas has a visual design menu similar to those found in image editors.

3 DISCUSSION

SBOLCanvas is being built from the ground up with these new features in mind. It is a completely new code base, making it as easy as possible to add new features. It is built with Angular, MxGraph, and uses a stateless back-end that leverages libSBOLJ 2 [5]. SBOLCanvas is under active development, and there are some enhancements that are planned in the near term, including:

- Support for combinatorial design to create many designs from variant libraries of parts and devices.
- Tighter integration with SynBioHub to search for parts and as a means to store designs during their development life cycle.
- Improvements in support for hierarchical design including Modules and Interactions.
- Support for automated addition of sequence tags and scars for a variety of assembly methods.
- Use of parametric SVG to allow glyphs to be added more easily and be easier to modify.
- Development of an SBOL layout Standard for exchange with other tools such as VisBOL.

To request a feature or report an issue, please visit: https://github.com/SynBioDex/SBOLCanvas/issues.

Acknowledgements

SBOLCanvas is supported in part by the SBOL Industrial Consortium.

REFERENCES

[1] Cox, R. S., Madsen, C., McLaughlin, J., Nguyen, T., Roehner, N., Bartley, B., Bhatia, S., Bissell, M., Clancy, K., Gorochowski, T., Grünberg, R., Luna, A., Novère, N. L., Pocock, M., Sauro, H., Sexton, J. T., Stan, G.-B., Tabor, J. J., Voigt, C. A., Zundel, Z., Myers, C., Beal, J., and Wipat, A. Synthetic biology open language visual (SBOL Visual) version 2.0. *Journal of Integrative Bioinformatics 15*, 1 (2018), 20170074.

[2] McLaughlin, J. A., Myers, C. J., Zundel, Z., Misirli, G., Zhang, M., Ofiteru, I. D., Goñi-Moreno, A., and Wipat, A. SynBioHub: A standards-enabled design repository for synthetic biology. *ACS Synthetic Biology 7*, 2 (2018), 682–688. PMID: 29316788.

[3] Roehner, N., Beal, J., Clancy, K., Bartley, B., Misirli, G., Grünberg, R., Oberortner, E., Pocock, M., Bissell, M., Madsen, C., Nguyen, T., Zhang, M., Zhang, Z., Zundel, Z., Densmore, D., Gennari, J. H., Wipat, A., Sauro, H. M., and Myers, C. J. Sharing structure and function in biological design with SBOL 2.0. *ACS Synthetic Biology 5*, 6 (2016), 498–506. PMID: 27111421.

[4] Zhang, M., McLaughlin, J. A., Wipat, A., and Myers, C. J. SBOLDesigner 2: An intuitive tool for structural genetic design. *ACS Synthetic Biology 6*, 7 (2017), 1150–1160. PMID: 28441476.

[5] Zhang, Z., Nguyen, T., Roehner, N., Misirli, G., Pocock, M. R., Oberortner, E., Samineni, M., Zundel, Z., Beal, J., Clancy, K., Wipat, A., and Myers, C. J. libsbolj 2.0: A java library to support sbol 2.0. *IEEE Life Sciences Letters 1* (2015), 34–37.

DESIGN AUTOMATION WORKFLOWS FOR SYNTHETIC BIOLOGY AND METABOLIC ENGINEERING: THE GALAXY-SYNBIOCAD PORTAL

Jean-Loup Faulon*
Melchior du Lac
Thomas Duigou
Jean-Loup.Faulon@inrae.fr
MICALIS Institute, INRAE, AgroParisTech, Université Paris-Saclay
Jouy-en-Josas, France

Joan Hérisson
Génomique Métabolique, Genoscope, Institut François Jacob, CEA, CNRS, Univ Evry, Université Paris-Saclay
Evry, France
Joan.Herisson@univ-evry.fr

Pablo Carbonell
I. U. de Automática e Informática Industrial, Universitat Politècnica de València
València, Spain
pjcarbon@isa.upv.es

1 INTRODUCTION

While many synthetic biology computer-aided design tools are being used throughout the Design-Build-Test-Learn (DBTL) cycle to engineer circuits and produce a variety of chemicals, these tools can be redundant, do not efficiently communicate with one another, making difficult smooth and reproducible runs. We introduce here the Galaxy-SynBioCAD portal, the first Galaxy toolshed for synthetic biology and metabolic engineering. The portal provides a simple design methodology for synthetic biology making use of standardized data exchange and models. It allows one to easily create workflows or use already developed shared workflows on Galaxy, a widely-used web-based scientific analysis platform. The portal is a growing community effort where developers can add new tools and users can evaluate the tools performing design for their specific projects. The tools and workflows currently shared on the Galaxy-SynBioCAD portal cover an end-to-end metabolic pathway design process from the selection of strain and target to the calculation of DNA parts to be assembled to build libraries of strains to be engineered to produce the target (Figure 1).

2 RESULTS

The SynBioCAD portal gathers a set of about 20 published tools under MIT and GPL licenses. All the tools have been dockerized and wrapped into Galaxy nodes. The portal allows a user to perform the design of metabolic pathways in one or more steps as needed. The different steps of the process are as follows:

- The user chooses a chassis strain. Currently all strains with a genome-wide metabolic (GEM) SBML model stored in the MetaNetX database are accessible. The user also provides a molecule of interest in the form of an InChI (International Chemical Identifier).

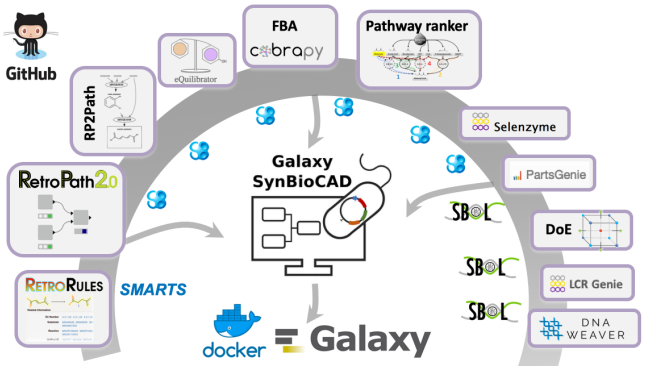


Figure 1: The Galaxy-SynBioCAD Portal.

- A retrosynthesis program (RetroPath node [3]) is used to determine whether the molecule to be produced can be linked by biochemical reactions to the chassis strain molecules. RetroPath creates a retrosynthetic map and the individual metabolic pathways of the retrosynthesic map are listed by the RP2Paths node [3].
- The Gibbs free energy is evaluated for each reaction of each pathway by the Thermodynamics node (based on eQuilibrator [5]). The production flows of each pathway in the chassis strain are calculated by FBA (CobraPy [4]).
- Pathways are ranked according to the results obtained by Retropath (reaction score based on enzyme availability), eQuilibrator (Gibbs free energy for reach reaction), FBA/CobraPy (product flux, thermodynamic feasibility), pathway length, or any other ranking function provided by the user.
- For each ranked metabolic pathway, the Selenzyme node [2] is used to search and classify the enzyme sequences that catalyze the biochemical reactions. The PartsGenie node [6] calculates the ribosome binding

*This research was funded by BioRoBoost and IBISBA H2020 programs and the RCUK's Synthetic Biology programs

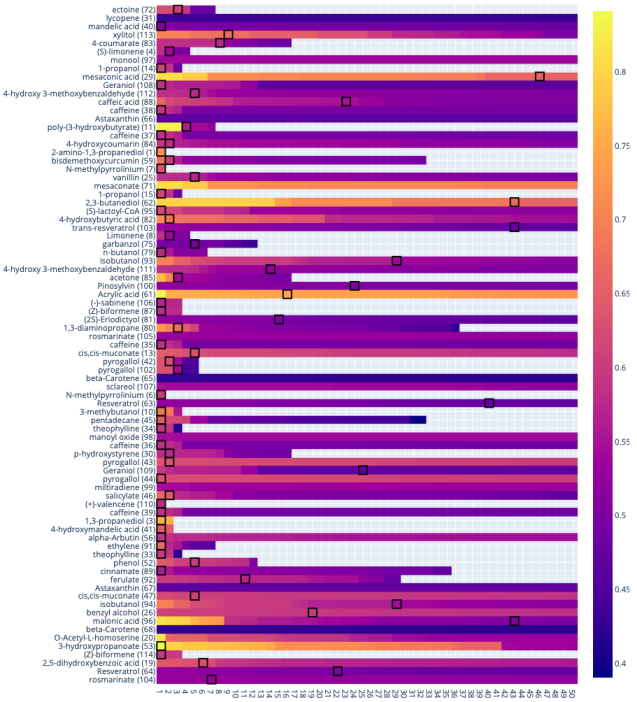


Figure 2: Literature pathways heatmap. Black squares indicate the ranking position of engineered pathways reported in the literature among the solutions generated by workflows on the Galaxy-SynBioCAD Portal.

site (RBS) sequences corresponding to different strength for each enzyme sequence and generates a SBOL file.

At this stage of the process, the system produces a ranked list of pathways, and for each reaction of each pathway a set of enzyme and RBS sequences. These sequences can be placed in different vectors with different replication origins and under the control of promoters of different strengths. The number of potential constructs can therefore be very large.

- The OptDoE (Design of Experiments) [1] node allows one to efficiently sample the space of potential constructs and to produce a user-predetermined number of layouts including replication origins, promoter sequences, RBS sequences (for prokaryotes), enzyme sequences, and terminators. These constructions are all produced in the standard SBOL format and can be stored in databases such as SynBioHub.
- Depending on the assembly protocol chosen by the user, the DNA sequences to be synthesized (or cloned) are generated by the PlasmidGenie node [6] (for LCR assembly) and DNA Weaver node [8] (for Gibson and GoldenGate assembly).

3 BENCHMARKING AND DISCUSSION

To benchmark the pathways produced by the portal, a list of experimentally expressed compounds in engineered organisms (*E. coli*, *S. cerevisiae*, *B. subtilis* and *Y. lipolytica*) reported in the literature was collected. Each target compound and strain within that list was used to run a workflow generating a collection of predicted pathways. The predicted pathways were compared with their corresponding engineered pathway in the literature. In order to find the literature pathway among the top scored predicted pathways we used the ranked-biased overlap algorithm [7] to adjust the weight of the criteria entering the ranking function. Using adjusted weights, Figure 2 shows the results of the ranked-biased overlap optimization schema. Each row is a ranked list of collections of predicted pathways for a given target molecule, where the best ranking pathways are shown on the left-hand side. The color code shows the global score that was used to rank the pathways. The black squares correspond to the predicted pathways that are the most closely similar to the literature pathway. Overall, we find that our workflow has a 65.4% success rate (53 out of a total of 81) in retrieving the literature pathway among the top 10 predicted pathways.

Contact

Jean-Loup Faulon (Jean-Loup.Faulon@inrae.fr). Additional information (videos, tutorials) on the Galaxy-SynBioCAD portal can be found at https://galaxy-synbiocad.org/ and publications on the portal tools at https://www.jfaulon.com/galaxy-synbiocad-portal/.

REFERENCES

[1] Carbonell, P., Faulon, J.-L., and Breitling, R. Efficient learning in metabolic pathway designs through optimal assembling. *IFAC-PapersOnLine 52*, 26 (Jan. 2019), 7–12.

[2] Carbonell, P., Wong, J., Swainston, N., Takano, E., Turner, N. J., Scrutton, N. S., Kell, D. B., Breitling, R., and Faulon, J.-L. Selenzyme: enzyme selection tool for pathway design. *Bioinformatics (Oxford, England) 34*, 12 (2018), 2153–2154.

[3] Delepine, B., Duigou, T., Carbonell, P., and Faulon, J.-L. RetroPath2.0: A retrosynthesis workflow for metabolic engineers. *Metabolic Engineering 45* (2018), 158–170.

[4] Ebrahim, A., Lerman, J. A., Palsson, B. O., and Hyduke, D. R. COBRApy: COnstraints-Based Reconstruction and Analysis for Python. *BMC systems biology 7* (2013), 74.

[5] Flamholz, A., Noor, E., Bar-Even, A., and Milo, R. equilibrator, the biochemical thermodynamics calculator. *Nucleic Acids Research 40*, Database issue (Jan. 2012), D770–D775.

[6] Swainston, N., Dunstan, M., Jervis, A. J., Robinson, C. J., Carbonell, P., Williams, A. R., Faulon, J.-L., Scrutton, N. S., and Kell, D. B. PartsGenie: an integrated tool for optimizing and sharing synthetic biology parts. *Bioinformatics (Oxford, England) 34*, 13 (2018), 2327–2329.

[7] Webber, W., Moffat, A., and Zobel, J. A similarity measure for indefinite rankings. *ACM Transactions on Information Systems 28*, 4 (Nov. 2010), 20:1–20:38.

[8] Zulkower, V., and Rosser, S. DNA Weaver: optimal DNA assembly strategies via supply networks and shortest-path algorithms, IWBDA 2019, Cambridge, UK.

Automation of a DOE Design Workflow in Synthetic Biology - A Comparative Study

Alexis Casas, Charles Motraghi, Matthieu Bultelle, Richard Kitney
{a.casas,charles.motraghi10,m.bultelle97,r.kitney}@imperial.ac.uk
Department of Bioengineering, Imperial College London
London, UK

1 INTRODUCTION

The London DNA Foundry at SynbiCITE (www.synbicite.com) is currently adapting its rapid Design-Build-Test-Learn cycle. The workflow is based on the use/reuse of standard, characterised biological parts, automated robotic assembly of constructs, and high-throughput screening. It incorporates design of experiments (DOE) approaches for the purpose of pathway optimisation.

2 STRATEGIC APPROACH

The Kitney Lab has undertaken a scoping study for the three-gene lycopene pathway based on a simple operon design with 810 promoter/RBS/gene order combinations, with a view to identify the design and variables responsible for the highest production titres. The choice of the pathway and design was motivated not only by the comprehensive background knowledge available for the pathway, but also on the basis that 810 designs are:

- small enough to build all combinations using standard modular plasmid construction methods (such as Golden Gate/MoClo [1] and BASIC assembly [2])
- large enough that DOE may be used (up to 10 possible iterations)
- large enough to compute relevant metrics and draw conclusions from them for any comparative study

Finally, the design space is also large enough that a workflow can be developed that is capable of scaling to larger, more complex problems.

3 METHODS

Within the study, all automation tasks are carried out in Python. Constructs are defined at a higher description level with a data structure that allows the combination and permutation of DNA parts. Parts are queried or retrieved directly from databases such as SynBIS-Lite or SynBioHub [3], or built *de novo*. The biopython [4] and pySBOL [5] modules are used for these steps. DNA assembly of the constructs - both for the BASIC and the CIDAR MoClo assembly methods - are generated with basicsynbio [6] and the MoClo Python framework [7] respectively.

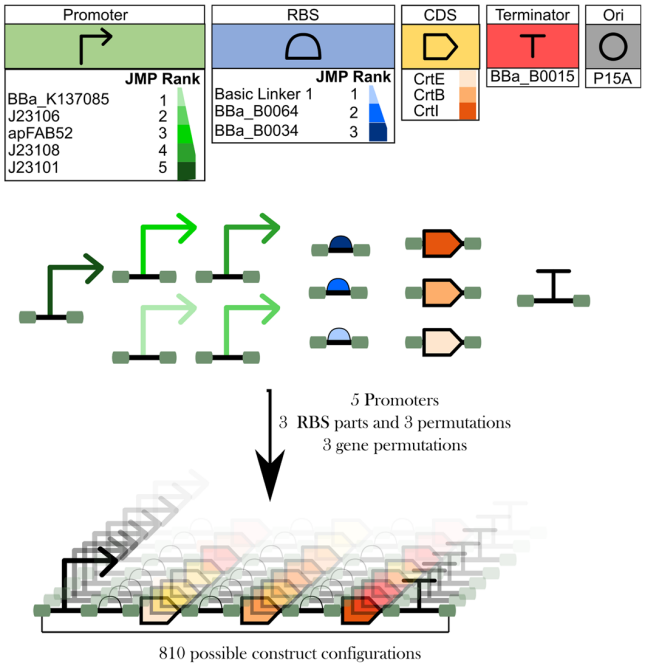


Figure 1: All constructs are based on a single operon with a total of 810 configurations. Promoters and RBSs with a range of strengths are taken from SynBIS-Lite/SynBioHub. The order of CrtE/B/I is also permutated.

The first stage of the study comprises how well-suited two standard DNA assembly techniques (Golden Gate/MoClo [8] [9] and BASIC assembly [2]) are for use in such a DOE project, looking primarily at:

- economic and time efficiency
- ease of use, including flexibility when expanding existing toolkits
- influence on pathway performance, with a focus on the influence of technique-specific scar sequences (as described by [10])

In parallel, the standard DOE models (as offered by the JMP package[11]) are compared, with specific reference to how quickly and how accurately they converge onto the fitness landscape (as defined by all 810 data points gathered experimentally). All comparisons are carried out against

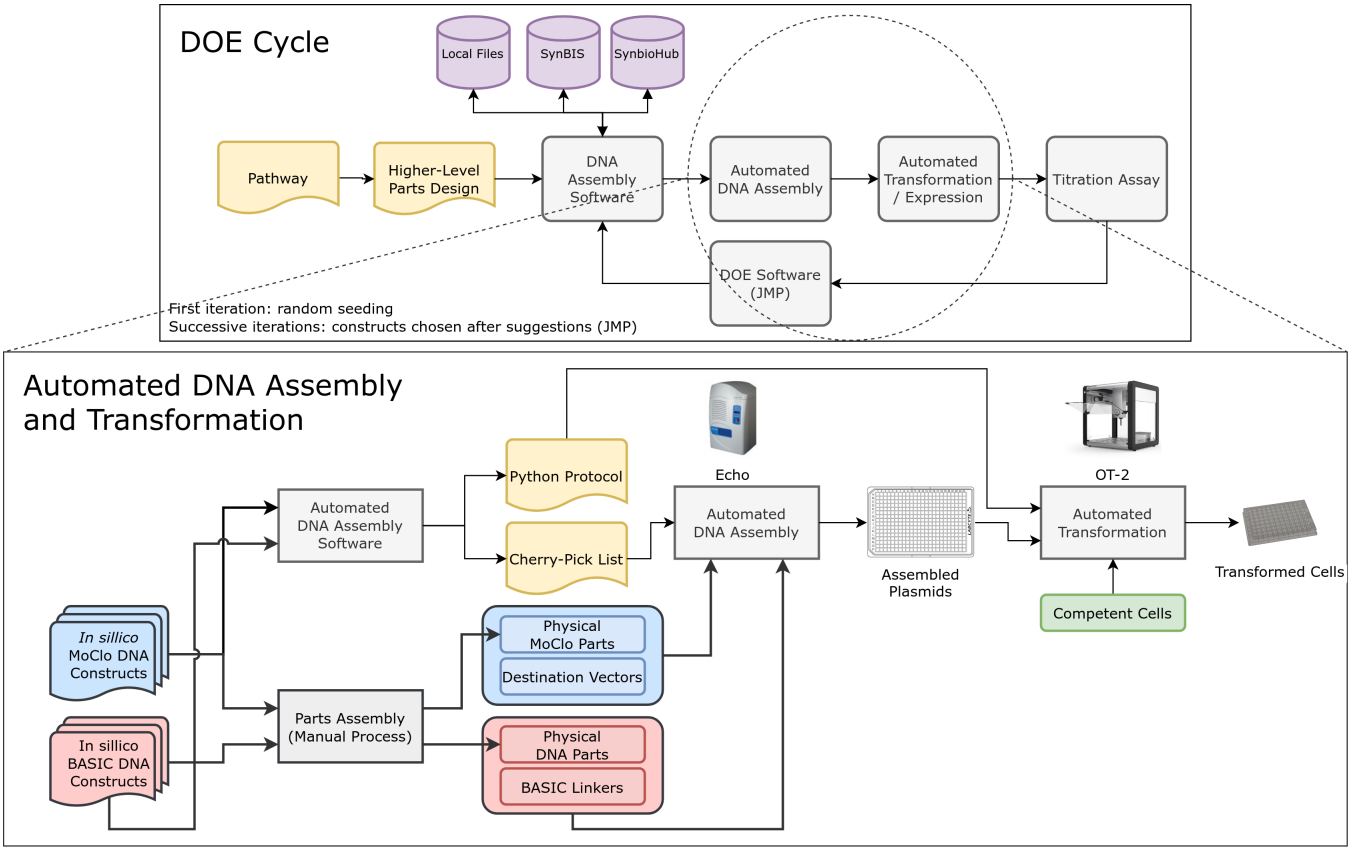


Figure 2: Design of Experiments Cycle and Automated DNA Assembly Workflow

the datasets obtained for both construction methods. The influence of the initial seeding is also assessed.

The second stage of the study will look at ways to mitigate the influence of gene order on synthetic metabolic pathway performance - and, more generally, how to reduce the influence of the genetic context, including (but not limited to) the use of the RiboJ ribozyme, insulated promoters, bicistronic RBSs and combinations thereof.

The final stage of the study investigates whether designing pathways as operons or individual transcription units is preferable.

REFERENCES

[1] Sonya V. Iverson, Traci L. Haddock, Jacob Beal, and Douglas M. Densmore. CIDAR MoClo: Improved MoClo Assembly Standard and New E. coli Part Library Enable Rapid Combinatorial Design for Synthetic and Traditional Biology. *ACS Synthetic Biology*, 5(1):99–103, January 2016.

[2] Marko Storch, Arturo Casini, Ben Mackrow, Toni Fleming, Harry Trewhitt, Tom Ellis, and Geoff S. Baldwin. BASIC: A New Biopart Assembly Standard for Idempotent Cloning Provides Accurate, Single-Tier DNA Assembly for Synthetic Biology. *ACS Synthetic Biology*, 4(7):781–787, July 2015.

[3] James Alastair McLaughlin, Chris J. Myers, Zach Zundel, Göksel Mısırlı, Michael Zhang, Irina Dana Ofiteru, Angel Goñi-Moreno, and Anil Wipat. SynBioHub: A Standards-Enabled Design Repository for Synthetic Biology. *ACS Synthetic Biology*, 7(2):682–688, February 2018. Reporter: ACS Synthetic Biology.

[4] Brad Chapman and Jeffrey Chang. Biopython: Python tools for computational biology, August 2000.

[5] Bryan A. Bartley, Kiri Choi, Meher Samineni, Zach Zundel, Tramy Nguyen, Chris J. Myers, and Herbert M. Sauro. pySBOL: A Python Package for Genetic Design Automation and Standardization. *ACS Synthetic Biology*, November 2018. Reporter: ACS Synthetic Biology.

[6] Matthew C Haines. personal communication, March 2020.

[7] Martin Larralde. Modular cloning simulation with the moclo framework in python. https://moclo.readthedocs.io/en/latest/.

[8] Carola Engler, Romy Kandzia, and Sylvestre Marillonnet. A One Pot, One Step, Precision Cloning Method with High Throughput Capability. *PLoS ONE*, 3(11):e3647, November 2008.

[9] Ernst Weber, Carola Engler, Ramona Gruetzner, Stefan Werner, and Sylvestre Marillonnet. A Modular Cloning System for Standardized Assembly of Multigene Constructs. *PLOS ONE*, 6(2):e16765, February 2011. Reporter: PLOS ONE.

[10] Swati B. Carr, Jacob Beal, and Douglas M. Densmore. Reducing DNA context dependence in bacterial promoters. *PLOS ONE*, 12(4):e0176013, April 2017.

[11] SAS. Jmp®, version 15. sas institute inc., cary, nc, 1989-2019. https://www.jmp.com/.

Round-Trip: An Automated Pipeline for Experimental Design, Execution, and Analysis

Daniel Bryce[1], **Robert P. Goldman**[1], **Matthew DeHaven**[1], **Jacob Beal**[2], **Tramy Nguyen**[2], **Nicholas Walczak**[2], **Mark Weston**[3], **George Zheng**[3], **Josh Nowak**[4], **Joe Stubbs**[5], **Niall Gaffney**[5], **Matthew Vaughn**[5], **Chris Myers**[6]
[1]SIFT, LLC., [2]Raytheon BBN Tech., Inc., [3] Netrias, LLC., [4]Strateos, Inc., [5]Texas Adv. Computing Center, [6]Univ. of Utah
dbryce@sift.net

1 INTRODUCTION

Lack of automated tools for experimental design, execution, and analysis is a key barrier to quickly evaluating biological designs. Experimentalists often face challenges describing experimental plans in a format laboratories can act upon, while labs have difficulty linking experimental data to metadata and plans, making this an expensive manual undertaking [2]. Metadata describing contents, conditions, and context of experimental samples is key to gaining insights from experimental data, but acquiring, tracking, and maintaining it is tedious, expensive, and error-prone.

We alleviate these difficulties with automation. As part of the Synergistic Discovery and Design (SD2) project, we have addressed these needs by developing a *Round-Trip* (RT) architecture simplifying this process by automating several steps where human intervention was previously required. Figure 1 illustrates the major RT components, proceeding left-to-right on top creating an automation-assisted experiment. RT attaches metadata to raw data as it returns from right to left on the bottom. Key steps (cross-referenced in Figure 1, with supporting components detailed in Section 2) include:

(1) **Authoring an Experiment Request**: Experimentalists, starting with a "notional" experiment in mind, author a semi-structured *Experiment Request* document.
(2) **Annotating the Experiment Request**: RT generates an *Annotated Experiment Request* that hyperlinks experimental constructs to SBOL[6] definitions through a simple Data Dictionary.
(3) **Structuring the Experiment Request**: Linked constructs and experiment sample tables are converted into a *Structured Request Template*, formally defining a set of minimal requirements for the experiment.
(4) **Experimental Design**: RT expands these to an *Experiment Design* with metadata for every measurement of every sample, plus a set of *Lab Parameters* configuring a machine-executable laboratory protocol.
(5) **Experiment Execution**: The laboratory (Strateos) conducts the experiment, generating *Experiment Data*.
(6) **Check Expected vs. Actual Data**: RT generates *Expected Experiment Data* and compares it with the Experiment Data, checking that the laboratory fulfills the design and identifying any discrepancies.
(7) **ETL**: The RT then stores experiment data, conducts ETL processing, and annotates the Experimental Request with the final data products.

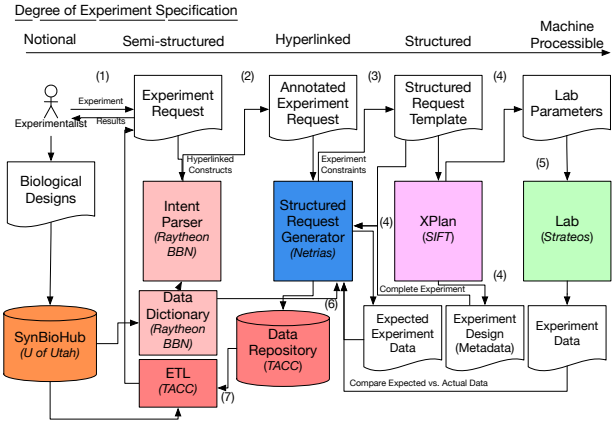


Figure 1: Round-Trip Architecture. (Component developer's affiliation noted in parentheses.)

By adopting "make metadata easy" design principles—providing human affordances, automating tedious metadata design and encoding, and reacting and repairing as deviations arise—this architecture provides many benefits to experimentalists. For example, it connects experimental data and subsequent analyses with deeply-represented experimental constructs by resolving user-friendly construct names. Experiment Requests can be partial, and RT will fill in details in experiment planning. RT also flags mismatches between expected and actual data for follow-up and diagnosis.

2 ROUND-TRIP ELEMENTS

Here we highlight key aspects of software components and data artifacts implementing the major stages of RT. Each component is involved in building up metadata through reducing

This work was supported by Air Force Research Laboratory (AFRL) and DARPA contracts FA875017CO184, HR001117C0092, and HR001117C0094. This document does not contain technology or technical data controlled under either U.S. International Traffic in Arms Regulation or U.S. Export Administration Regulations. Views, opinions, and/or findings expressed are those of the author(s) and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government. Approved for public release, distribution unlimited (DISTAR Case 32707).

Media	Dilution	B-Est. @ 16h	Rep.	Strain
SC Media	50x	0.0, 0.05 uM	10	UWBF_24926, UWBF_24952, UWBF_24959
SC Media	50x	0.0 uM		UWBF_24960, UWBF_25784, UWBF_24962

Table 1: Experiment Request Measurement Table

high-level experiment descriptions to machine representations, and then attaching measurement data to the metadata.

Semi-Structured Experiments: Experiment Requests (ERs) are documents including both prose and tabular descriptions of an experiment. The prose provides context, motivations, and anticipated results, along with lists of strain and reagent descriptions. The ER also includes a measurement table and a parameter table. The measurement table lists experimental factors by column, and constraints on their values by row. Table 1 shows an example measurement table indicating the experiment should contain ten replicates of the strains in the first row, each induced with either 0.0 or 0.05 uM beta-estradiol at 16 hours. It also states that the number of replicates of strains in the second row is still to be determined, and that these are not induced (i.e., 0.0 uM beta-estradiol)

Hyperlinking Experiments: The Intent Parser (IP) [5] processes the ER to identify constructs appearing in the Data Dictionary, then links them to SBOL descriptions in SynBioHub [4]. The Data Dictionary [1] maps each of (potentially many) construct common names to a canonical definition URI in SynBioHub. For example, the term "B-Est" in Table 1 is a common short-hand term that will be linked to the beta-estradiol reagent definition. IP likewise links strain identifiers (e.g., UWBF_24926) and media to their definitions. This provides experimentalists flexibility with common terms and shorthand, while unifying them across experiments (e.g., another ER might use Beta-Est. instead of B-Est.).

Structuring Experiments: Structured Requests (SRs) formally represent the set of samples an experiment will generate. SRs take two forms: templates and expected samples. The SR Generator creates a template capturing constraints from the ER, but this may not map directly to samples (e.g., the second row in Table 1 omits replicate count). After experimental planning (below), the SR Generator expands the SR and an Experiment Design into expected samples, which can be checked against actual samples on experiment completion, also matching lab-specific identifiers (e.g., LIMS inventory IDs) with ER common names via the Data Dictionary.

Machine Processible Experiments: The Experiment Planner (XPlan) [3] uses an SR template to create machine processible experiments that are suitable for laboratory execution. XPlan uses this to constrain its search for an Experiment Design, which in turn describes the expected measurements of each aliquot in the experiment. XPlan dispatches this with a set of Lab Parameters, instructing the lab how to configure and run the experiment. XPlan decides not only how to allocate samples to physical containers, but also which samples to use. For example, XPlan will choose the number of replicates for the strains in the second row of the ER in Table 1 based upon the available containers.

Laboratory Execution: RT submits experiments to the Strateos cloud laboratory for automated execution. Here, RT selects from one of several Strateos experimental protocols, such as growth curves and time series. In these protocols, Strateos measures samples with a plate reader and flow cytometer over several time points, including multiple induction and dilution steps, and returns both raw measurement data and protocol execution traces. In future work, the RT will also interface with laboratories via Aquarium[1].

Metadata Validation and ETL: The SR Generator validates data products by aligning metadata descriptions with expected data. It flags and explains any discrepancies to the experimentalist and lab technicians. If able to successfully match the data, the RT performs a series of ETL steps that summarize results for the experimentalist, organized in terms of the metadata on sample contents, conditions, and context.

3 VALIDATION

Over a four month period, we applied RT to process and execute twenty three ERs, totaling fifty nine 96-well plates of samples and approximately 10 measurements per well. The ERs span three distinct experimental protocols. With RT, we can plan and attach metadata to experimental samples within approximately four hours (not accounting for experiment execution time), whereas before it took approximately three weeks to attach metadata to six 96-well plates worth of data. This has allowed us to reduce laboratory idle time (due to dependent experiments) from several weeks to a few days.

ACKNOWLEDGEMENTS

The authors acknowledge Ben Keller, Peter Lee, and Narendra Maheshri for initial development of ER documents.

REFERENCES

[1] Beal, J., et al. Collaborative terminology: SBOL project dictionary. In *International Workshop on Bio-Design Automation (IWBDA)* (2020).
[2] Jessop-Fabre, M. M., and Sonnenschein, N. Improving reproducibility in synthetic biology. *Frontiers in Bioeng. and Biotech. 7* (2019), 18.
[3] Kuter, U., et al. XPLAN: Experiment planning for synthetic biology. In *ICAPS Workshop on Hierarchical Planning* (2018).
[4] McLaughlin, J. A., et al. SynBioHub: A standards-enabled design repository for synthetic biology. *ACS Syn. Bio. 7*, 2 (2018), 682–688.
[5] Nguyen, T., et al. Intent parser: a tool for codifying experiment design. In *International Workshop on Bio-Design Automation (IWBDA)* (2020).
[6] Roehner, N., et al. Sharing structure and function in biological design with sbol 2.0. *ACS Syn. Bio. 5*, 6 (2016), 498–506.

[1] https://www.aquarium.bio

Integrated Decision-Making to Detect DNA Engineering in Yeast

Sancar Adali[1], Aaron Adler[1], Joel S. Bader[2], Joseph H. Collins[3], Yuchen Ge[2], John Grothendieck[1], Thomas Mitchell[1], Anton Persikov[3], Jonathan Prokos[2], Richard Schwartz[1], Mona Singh[3], Allison Taggart[1], Benjamin Toll[1], Stavros Tsakalidis[1], Daniel Wyschogrod[1], Fusun Yaman[1], Eric M. Young[4], and Nicholas Roehner[1]
[1]Raytheon BBN Technologies, [2]Johns Hopkins University, [3]Princeton University, [4]Worcester Polytechnic Institute
nicholas.roehner@raytheon.com

1 INTRODUCTION

Yeasts are highly industrially relevant organisms[2] and are among the most frequently genetically engineered, making them prime candidates for accidental release or intentional abuse. Despite the large body of literature available on engineering model organisms such as these, significant challenges remain for the detection of DNA engineering in general. These challenges include the large of space of unknown natural DNA, the tendency of engineered biological systems to require unique parts to scale, and the fact that relatively small changes to genotype can produce large changes in phenotype. The first challenge hinders anomaly-based approaches that use models of natural sequences to detect what is unnatural (and possibly engineered), while the second hinders signature-based approaches that use models of engineering signatures to detect engineered sequences (the simplest signature being a foreign part such as a promoter, CDS, or terminator). As a result of the foregoing, it is necessary to employ a variety of different approaches to detect DNA engineering, at which point it also becomes necessary to consider how to best integrate their decisions.

2 THE GUARDIAN SYSTEM

As shown in Figure 1, the GUARDIAN system for detecting DNA engineering can be divided into four subsystems: genome assembly, assembly-first analysis, reads-first analysis, and integrated decision-making. For genome assembly, GUARDIAN uses a customized *de novo* short- and long-read assembly pipeline (PRYMETIME[1] - uses flye and Unicycler) to produce high quality genome assemblies. The resulting assemblies are then input to assembly-first analysis, which can be subdivided into four subsystems: FAST-NA, HMM, DL Models, and N-Gram. FAST-NA, HMM, and N-Gram are anomaly-based approaches that model natural sequences using Bloom filters, hidden Markov models, and N-gram language models, respectively. DL Models is a more signature-based approach that models engineering signatures using Seq2Class DL models with/without Siamese networks.

In parallel with assembly-first analysis, reads-first analysis is performed on the raw sequencing data using another two subsystems: Targeted Search and JHU. Targeted Search

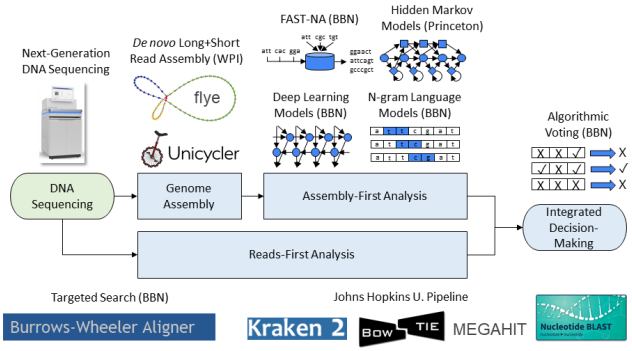


Figure 1: GUARDIAN system architecture.

is a signature-based approach that looks for known engineering parts and scars among short and long reads using the Burrows-Wheeler Aligner. JHU is a more anomaly-based approach that uses Bowtie to filter out natural reads and Megahit to assemble suspicious reads into contigs for BLAST.

Finally, all decisions made by GUARDIAN's subsystems are integrated using an expert-designed voting algorithm in which a sample is called engineered if either FAST-NA or N-Gram says "yes," both JHU and Targeted Search say "yes," or a majority of all subsystems say "yes."

3 RESULTS AND DISCUSSION

We tested GUARDIAN on sequencing datasets for 16 samples of *S. cerevisiae* (13 engineered, 3 not), 25 samples of *Y. lipolytica* (19 engineered, 6 not), and 9 samples of *P. pastoris* (3 engineered, 6 not). As shown in Figure 2, GUARDIAN's anomaly-based approaches had higher sensitivity and specificity than its signature-based approaches when applied to these datasets, with the best (FAST-NA and JHU) having sensitivity 0.8 and specificity 1. This suggests that our available training data for natural sequences are better than our training data for engineered sequences. Approaches in the same class were trained using similar training datasets (some synthetically generated) distinct from the test datasets. Also note that GUARDIAN's overall performance was higher than any of its individual subsystems (sensitivity 0.89 and specificity 1), which suggests the voting algorithm is well designed.

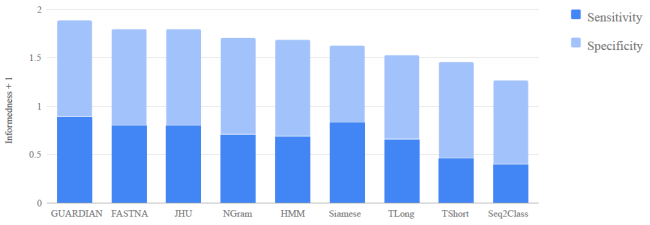


Figure 2: Sensitivity plus specificity for GUARDIAN and its subsystems.

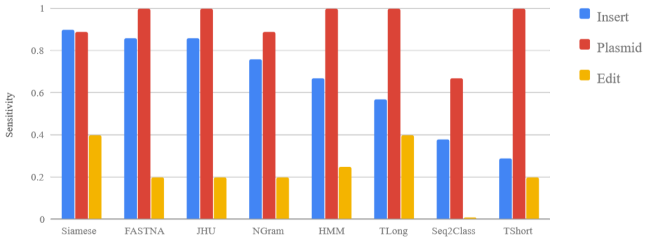


Figure 3: Sensitivity of GUARDIAN's subsystems by type of engineering signature.

Figure 3 breaks down the sensitivity of GUARDIAN's subsystems by the types of engineering signatures found in each sample. Nearly every subsystem had high sensitivity (>80%) when detecting engineering in samples containing non-integrating plasmid vectors. For genomic inserts, however, there is a difference in sensitivity between GUARDIAN's anomaly-based detection subsystems (HMM and all subsystems to its left) and its signature-based detection subsystems (TLong and all subsystems to its right). The former have sensitivity >67% and the latter <57%. A likely explanation for this difference is that there is typically greater diversity among sequences inserted into the genome compared to those found in standard plasmid vectors, which would favor the subsystems trained to recognize unnatural sequences as opposed those trained to recognize engineering based on a specific corpus of engineered sequences. Finally, most subsystems struggled to detect small (e.g. single nucleotide) edits, with no subsystem achieving sensitivity >40%, and most positive detections for small edit samples were likely either spurious or based on contamination (e.g. with a plasmid).

As GUARDIAN continues to evolve, new subsystems may be added or new connections may be made between existing subsystems. In such a scenario, we want to quickly learn a new strategy for decision integration, ideally in a manner that does not require significant subject matter expertise. Figure 4 compares the performance of the current voting algorithm with that of a range of subsystem decision weights learned using two approaches: label-free fusion and linear

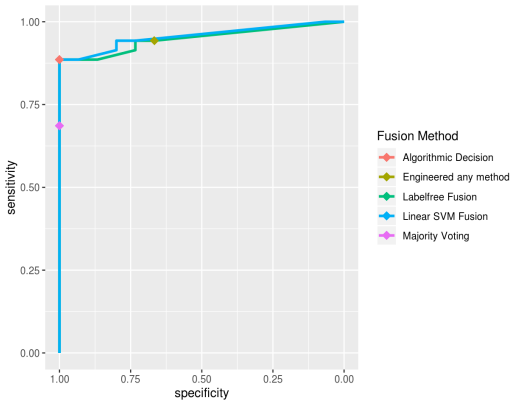


Figure 4: Sensitivity vs. specificity for different voting algorithms (dots) and subsystem decision weights learned with different ML algorithms (lines).

SVM fusion. The former learns weights based on subsystem agreement, while the latter learns weights based on subsystem performance. As shown in Figure 4, both approaches are capable of learning weights with performance as good as the current voting algorithm, and the latter appears to have the highest possible sensitivity with no loss in specificity. This suggests that we may be able to learn new weights for subsystem decisions as GUARDIAN develops rather than design a new voting algorithm. Going forward, we also plan to extend GUARDIAN to better handle deletions, small edits, and metagenomic sequencing datasets.

ACKNOWLEDGMENTS

We thank Pacific Northwest National Laboratory and Lawrence Berkeley National Laboratory for preparing the sequencing data with which we tested GUARDIAN. This research is based upon work supported [in part] by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA) under Finding Engineering Linked Indicators (FELIX) program contract #N6600118C-4507. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

REFERENCES

[1] Collins, J. H., et al. Verification of genetic engineering in yeasts with nanopore whole genome sequencing. *bioRxiv* (2020).

[2] Parapouli, M., Vasileiadis, A., Afendra, A. S., and Hatziloukas, E. Saccharomyces cerevisiae and its industrial applications. *AIMS Microbiology 6* (2020), 1–31.

The Synthetic Biology Knowledge System

Chris Myers
Jeanet Mante
Eric Yu
Logan Terry
University of Utah
Salt Lake City, UT, USA
myers@ece.utah.edu,jv@mante.net
ejyu99@gmail.com,randoom97@live.com

Mai H. Nguyen
Gaurav Nakum
Jiawei Tang
Xuanyu Wu
University of California, San Diego
La Jolla, CA, USA
{mhnguyen,gnakum,jit072,xuw057}@ucsd.edu

Kevin Keating
Eric Young
Worcester Polytechnic Institute
Worcester, MA, USA
kwkeating@wpi.edu,emyoung@wpi.edu

Bridget T. McInnes
Nicholas E. Rodriguez
Virginia Commonwealth University
Richmond, USA
btmcinnes@vcu.edu,rodrigueazne2@vcu.edu

Jacob Jett
J. Stephen Downie
University of Illinois at Urbana-Champaign
Champaign, Illinois, USA
jjett2@illinois.edu,jdownie@illinois.edu

Brandon Sepulvado
NORC at the University of Chicago
Bethesda, MD, USA
sepulvado-brandon@norc.org

1 INTRODUCTION

Synthetic biology has transformative potential in a variety of application areas including agriculture, energy, materials, and health. While much of the research in this field has been in *E. Coli*, many applications require yeast and other bacteria. Researchers often use trial-and-error, since information can be difficult to locate. The goal of the *Synthetic Biology Knowledge System* (SBKS) is to create an an open and integrated resource that harnesses disparate, heterogeneous data sources to accelerate scientific exploration and discovery. This abstract gives an overview of the SBKS project, while several other abstracts submitted to this workshop explain different aspects in more detail.

2 SBKS CURATION PIPELINE

The core of SBKS is a curation pipeline (see Figure 1) that integrates knowledge found in both text and data sources. The knowledge once harvested is encoded into the *Synthetic Biology Open Language* (SBOL) [1], a *rich data format* (RDF) data standard for genetic design. This SBOL representation is then uploaded to the SBKS instance of the SynBioHub [2] data repository. Once deposited, it can be searched and accessed using either a graphical user interface (GUI) or programmatically by its application programmers interface (API).

Text Mining Pipeline: Our initial text data set is all the articles that have been published in ACS Synthetic Biology. These articles are provided in richly annotated JATS XML markup, which includes both a rich set of metadata and the full article text. The metadata and citation elements of the structured article file are harvested and converted into SBOL-compliant RDF/XML with Dublin Core annotations suitable for ingestion into SynBioHub. Among the steps taken during this process is the employment of python scripts to match article DOI's to corresponding PubMed ID's.

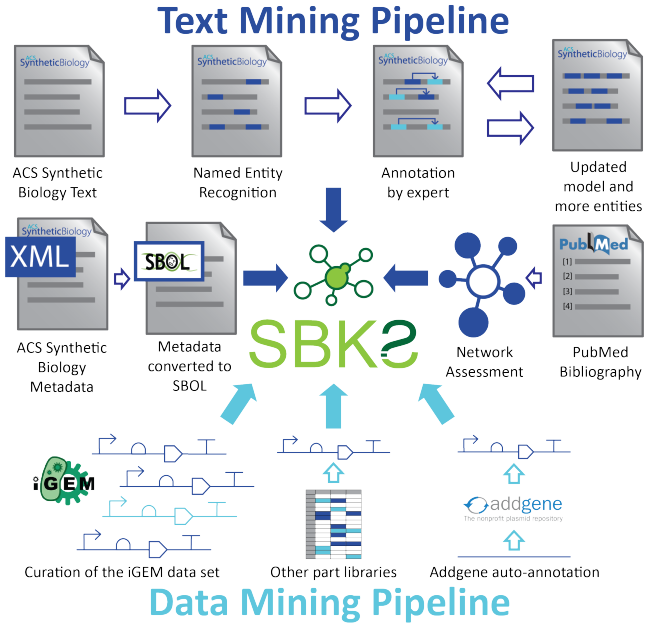


Figure 1: SBKS curation pipeline.

The XML files are also parsed to extract the full article text. The article text is then processed using techniques for *named entity recognition* (NER), which is a sub-task of text mining. The goal of NER is to locate and classify named entities present in text into pre-defined categories. We use deep neural network models to perform NER on these articles. For the initial round of NER, standard biological entity categories (e.g., genes and chemicals) are used since there

is no labeled dataset for synthetic biology entities. Results from this initial round are reviewed and corrected by domain experts to create a more refined dataset with entities more specific to synthetic biology that can be used to fine tune the NER models. Named entities expected to be detected within synthetic biology articles are also added to the articles as suggested annotations, to be confirmed by expert annotators in order to facilitate the creation of gold-standard synthetic biology-specific training data.

Another component of the text mining pipeline is the mapping of the social and conceptual structure of synthetic biology ethics, which is accomplished with network analysis and topic modeling. Building upon established bibliometric techniques to identify the synthetic biology literature, we located all 15,152 publications in the Web of Science pertaining to synthetic biology and then derived from this set of publications a smaller corpus of 562 ethical texts. Although synthetic biology literature began to increase exponentially around 2000, not much attention was devoted to ethics until roughly 2010. Ethical discourse in this field is currently dominated by small set of institutions, and scholars tend to collaborate only with a few others. The next stage of the ethics component will build upon this knowledge in order to return known ethical concerns and relevant literature related to SBKS users' queries.

Data Mining Pipeline: Our initial data set for the data mining pipeline are the synthetic biology parts and designs found in the *International Genetically Engineered Machine* (iGEM) registry of parts. Whilst the iGEM registry is large and expanding, it is not easy to extract information from parts to encourage reuse. As such a more standardized form of the iGEM library was created by converting it into SBOL. As part of the validation process we realised that there are many spurious records either due to improper completion of the record, making the information undecipherable, or as records were simply created as test exercises. Furthermore, many sequences have multiple records—one for each use case, with different use cases sometimes using the sequence in different ways. As such, we propose creating an iGEM library of sequences with different 'experiences' or uses of each sequence being linked to a core sequence. To help in this process, we are using both simple filtering and hand annotations as well as machine learning methods to create 'useful unique entity' records that are fully annotated.

In addition to the iGEM registry, parts are commonly reported in the literature in the form of "toolkit" papers. These papers almost always include part name, type, host organism and characterization data. Sequence information is often included as well, typically in the form of a table in supplemental information. Transferring parts from primary literature into an SBOL database will help to bridge the gap between highly-characterized parts reported in the literature and design tools which require data in a standardized format.

Finally, we plan to link part use to articles using the Addgene data set. Addgene is a company that stores plasmids typically created for published research studies. Once we have a good library of parts, we can use this library to annotate the sequences of the plasmids being stored by Addgene, and thus link parts to their uses in published research papers connecting the data and text pipeline results.

User Interface

One final aspect of the SBKS project is the development of a user interface that can access the information stored in SBKS to assist a designer of a genetic circuit. SBOLCanvas is a web application for creation and editing of genetic constructs using the SBOL data and visual standard. SBOLCanvas allows a user to create a genetic design from start to finish, with the option to incorporate existing SBOL data from a SynBioHub repository, such as SBKS. While SBOLCanvas is currently able to efficiently create genetic designs for parts selected via searches on SynBioHub, the end goal will be to have a design tool that provides a synthetic biology designer a seamless connection to knowledge about the parts that they are or could use in their designs.

3 DISCUSSION

The SBKS project began less than a year ago, and it is being executed by a team that met only a few months before that. While the scope is ambitious, the progress so far is very promising. We look forward to feedback from the community about the needs and potential applications for SBKS.

ACKNOWLEDGEMENTS

This work was funded by the National Science Foundation under Grants No. 1939892, 1939929, 1939885, 1939887, 1939951, and 1939860.

REFERENCES

[1] Beal, J., Nguyen, T., Gorochowski, T. E., Goñi-Moreno, A., Scott-Brown, J., McLaughlin, J. A., Madsen, C., Aleritsch, B., Bartley, B., Bhakta, S., Bissell, M., Castillo Hair, S., Clancy, K., Luna, A., Le Novère, N., Palchick, Z., Pocock, M., Sauro, H., Sexton, J. T., Tabor, J. J., Voigt, C. A., Zundel, Z., Myers, C., and Wipat, A. Communicating structure and function in synthetic biology diagrams. *ACS Synthetic Biology 8*, 8 (2019), 1818–1825. PMID: 31348656.

[2] McLaughlin, J. A., Myers, C. J., Zundel, Z., Misirli, G., Zhang, M., Ofiteru, I. D., Goñi-Moreno, A., and Wipat, A. SynBioHub: A standards-enabled design repository for synthetic biology. *ACS Synthetic Biology 7*, 2 (2018), 682–688. PMID: 29316788.

Multistable and dynamic CRISPRi-based synthetic circuits

Javier Santos-Moreno
Dept. of Fundamental Microbiology
University of Lausanne
Lausanne, Switzerland
javier.santosmoreno@unil.ch

Eve Tasiudi
Dept. of Biosystems Science and Engineering
ETH Zurich
Basel, Switzerland
eve.tasiudi@bsse.ethz.ch

Joerg Stelling
Dept. of Biosystems Science and Engineering
ETH Zurich
Basel, Switzerland
joerg.stelling@bsse.ethz.ch

Yolanda Schaerli
Dept. of Fundamental Microbiology
University of Lausanne
Lausanne, Switzerland
yolanda.schaerli@unil.ch

IWBDA 2020, Aug 2020, Online

Synthetic biology aims to build artificial decision-making circuits that are programmable, predictable and perform a specific function[1]. Since the rise of synthetic biology in the 2000s, most synthetic circuits have been governed by protein-based regulators. Recently, however, there has been growing interest in circuits based on RNA regulators as a means to overcome some of the intrinsic limitations of protein regulators[2].

The prokaryotic adaptive immunity system CRISPR constitutes a powerful platform for the construction of RNA-driven synthetic circuits[3]. CRISPR interference (CRISPRi) offers several advantages over protein regulators for synthetic circuit design. Due to its RNA-guided nature, CRISPRi is highly programmable, allows for easy design of sgRNAs that can be highly orthogonal and whose behavior in different environments can be easily predicted *in silico*. It also imposes low burden on host cells[2] and is encoded in shorter sequences than protein-based repressors, thereby facilitating circuit handling and delivery and reducing cost. A potential drawback of CRISPRi is the lack of cooperativity[4]. Cooperative protein transcription factors function non-linearly, a difference that might prevent the successful implementation of CRISPRi-based dynamic and multistable circuits[4-6].

The last few years have seen a growing interest in developing CRISPRi-based synthetic circuits. However, despite the enormous potential of CRISPRi for synthetic circuit design, the use of CRISPRi circuits in prokaryotes has been largely focused on logic gates and to the best of our knowledge none of the flagship circuits in synthetic biology (namely, the bistable toggle switch[7] and the repressilator[8]) have been re-constructed using CRISPRi. Here, we fill this unaddressed gap by demonstrating that CRISPRi can be used for building some of the most notorious (synthetic) circuit topologies[9]. To this aim, we adopted a design strategy aiming at providing our circuits with high modularity, predictability and orthogonality, and lo metabolic burden on the host, as detailed below.

The main circuit components were all expressed from a single vector to avoid fluctuations in their stoichiometry. Circuit transcriptional units (TUs) were isolated from each other by strong transcriptional terminators and 200 bp spacer sequences. To prevent mRNA context-dependency and provide transcriptional insulation within TUs, a 20 bp Csy4 cleavage sequence[10] was used flanking single guide RNAs (sgRNAs) and upstream of the ribosome binding sites (RBS) of the reporter genes. To avoid cross-talk between fluorescent reporters, orthogonal degradation tags[11] were employed. Sequence repetition at the DNA level was minimized to prevent unwanted recombination events. The levels of dCas9 and Csy4 were kept constant by expressing them from constitutive promoters in a separate vector. All gene circuits were tested in *Escherichia coli* (MK01) incubated in a rich defined media (EZ, Teknova) for maximizing cell fitness while reducing variability. In order to speed up the design-build-test cycle, we adopted a previously described cloning strategy that allows for fast and modular assembly of synthetic networks[12].

Enabled by the intrinsic properties of CRISPRi and the favourable characteristics of our design, we managed to build for the first time a CRISPRi repressilator (named 'CRISPRlator', **Figure 1**), bistable toggle switch, and stripe-pattern-forming incoherent feed-forward loop (IFFL, a.k.a. band-pass filter)[9]. Our mathematical model suggests that unspecific binding in CRISPRi is essential to establish multistability[9]. We also demonstrate that our CRISPRi-based circuits can be easily combined together without cross-reactivity or metabolic impact on the host, as exemplified by a combination of two stripe-forming IFFLs or an IFFL plus a double inverter working in parallel but independently within the same cellular environment[9].

Our work demonstrates the wide applicability of CRISPRi in synthetic circuits and paves the way for future efforts towards engineering more complex synthetic networks, boosted by the advantages of CRISPR technology.

REFERENCES

[1] Xie, M. Q.; Fussenegger, M., *Nat. Rev. Mol. Cell. Bio.* **2018,** *19* (8), 507-525.
[2] Chappell, J.; Watters, K. E.; Takahashi, M. K.; Lucks, J. B., *Curr. Opin. Chem. Biol.* **2015,** *28*, 47-56.
[3] Jusiak, B.; Cleto, S.; Perez-Pinera, P.; Lu, T. K., *Trends Biotechnol.* **2016,** *34* (7), 535-547.
[4] Nielsen, A. A.; Voigt, C. A., *Mol. Syst. Biol.* **2014,** *10* (11), 763.

[5] Clamons, S. E.; Murray, R. M., *bioRxiv* **2017**, 225318.
[6] Lebar, T.; Bezeljak, U.; Golob, A.; Jerala, M.; Kadunc, L.; Pirs, B.; Strazar, M.; Vucko, D.; Zupancic, U.; Bencina, M.; Forstneric, V.; Gaber, R.; Lonzaric, J.; Majerle, A.; Oblak, A.; Smole, A.; Jerala, R., *Nat. Commun.* **2014,** *5*, 5007.
[7] Gardner, T. S.; Cantor, C. R.; Collins, J. J., *Nature* **2000,** *403*, 339.
[8] Elowitz, M. B.; Leibler, S., *Nature* **2000,** *403* (6767), 335-8.
[9] Santos-Moreno, J.; Tasiudi, E.; Stelling, J.; Schaerli, Y., *Nat. Commun.* **2020,** *11* (1), 2746.
[10] Tsai, S. Q.; Wyvekens, N.; Khayter, C.; Foden, J. A.; Thapar, V.; Reyon, D.; Goodwin, M. J.; Aryee, M. J.; Joung, J. K., *Nat. Biotechnol.* **2014,** *32* (6), 569-76.
[11] Butzin, N. C.; Mather, W. H., *ACS Synth Biol* **2018,** *7* (1), 54-62.
[12] Santos-Moreno, J.; Schaerli, Y., *ACS Synth. Biol.* **2019,** *8*, 1691-1697.

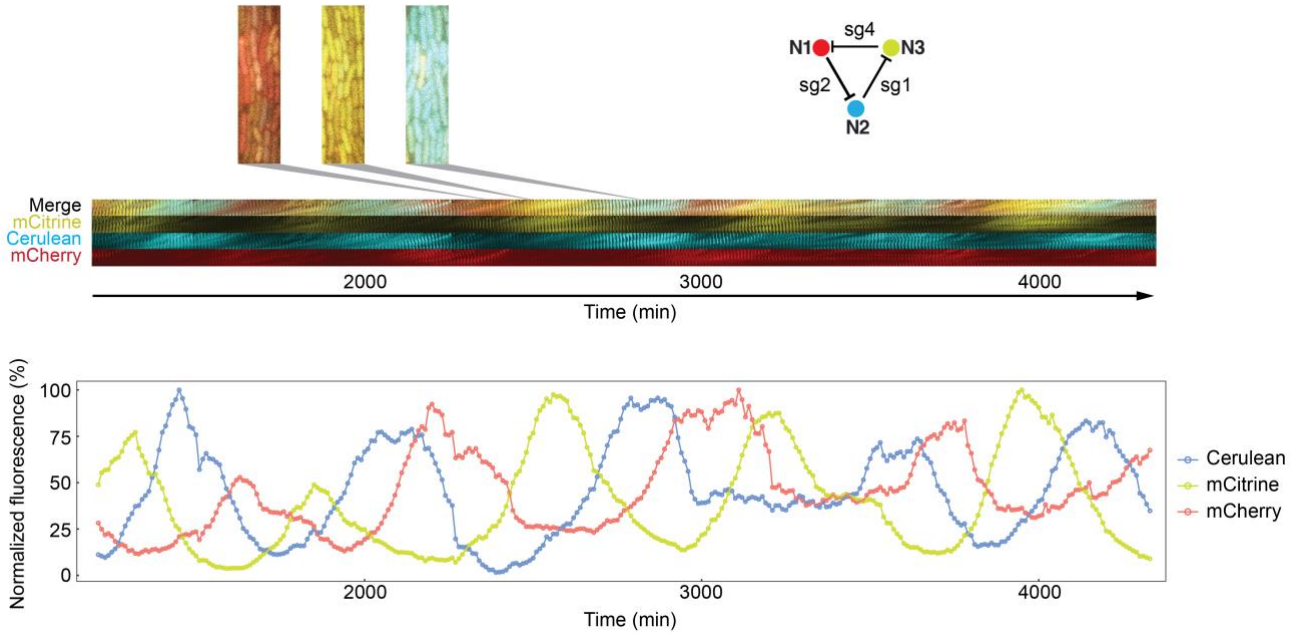


Figure 1: The CRISPRlator, a CRISPRi-based oscillator[9]. The closed-ring repression topology consists of three nodes (N1, N2 and N3), each comprising a single guide RNA (abbreviated here as sg1, sg2 and sg3) and a fluorescent reporter (mCherry, Cerulean and mCitrine) (top right). Shown is a montage displaying the oscillations of the three fluorescent reporters over time. Bacteria were grown in continuous exponential phase in a microfluidic device for 3 days and imaged every 10 min. Microscopy images (as those enlarged in the zoom-ins) are displayed together in a timeline montage (kymograph). Quantification of the population-level fluorescence of ~110 cells over time in shown below. Oscillations display a period of 10-12 h.

Robust control of biochemical reaction networks via stochastic morphing

Tomislav Plesa*
Department of Bioengineering, Imperial College
London, UK
t.plesa@ic.ac.uk

Guy-Bart Stan
Department of Bioengineering, Imperial College
London, UK
g.stan@imperial.ac.uk

Thomas E. Ouldridge
Department of Bioengineering, Imperial College
London, UK
t.ouldridge@imperial.ac.uk

Wooli Bae
Department of Bioengineering, Imperial College
London, UK
w.bae@imperial.ac.uk

1 ABSTRACT

Synthetic biology has experienced multiple breakthroughs in the past two decades, as novel cellular mechanisms are discovered, and cutting-edge theoretical and experimental methods are developed [1–3]. One of the main goals of synthetic biology is the development of molecular controllers that can manipulate the dynamics of a given ambient biochemical network. When integrated into smaller compartments, such as living or synthetic cells, controllers have to be calibrated to factor in the intrinsic noise arising from lower molecular copy-numbers [4–7]. In this context, biochemical controllers put forward in the literature have focused on manipulating the mean (first moment) and reducing the variance (second moment) of the target molecular species [8–10]. However, many critical biochemical processes, such as cellular differentiation and memory, quorum sensing and bacterial chemotaxis, are realized via higher-order moments, particularly the number and configuration of the probability distribution modes (maxima) [6, 7, 11, 12]. Such dynamically exotic and biochemically important phenomena cannot be achieved using controllers that target only the mean and variance.

To bridge the gap, in this talk we put forward the *stochastic morpher* controller [13] that, under suitable timescale separations, morphs the probability distribution of the desired biochemical species into any predefined form. The morphing can be performed at a lower-resolution, allowing one to achieve desired multi-modality/multistability (see Figure 1(a)–(d)), and at a higher-resolution, allowing one to achieve arbitrary probability distributions. Properties of the controller, such as robustness and convergence, are rigorously established using singular perturbation theory, and demonstrated on various examples. Furthermore, we also propose a blueprint for an experimental implementation of the stochastic morpher using DNA strand-displacement nanotechnology [3], allowing one to experimentally design multi-phenotypic synthetic cells (see Figure 1(e)–(f)).

*Corresponding author and lead contact.

REFERENCES

[1] Endy D., 2005. Foundations for engineering biology. *Nature*, **484**: 449–453.

[2] Del Vecchio, D., Dy, A. J., Qian, Y., 2016. Control theory meets synthetic biology. *Journal of the Royal Society Interface*, **13**(**120**): 3–43.

[3] Soloveichik, D., Seeing G., Winfree E., 2010. DNA as a universal substrate for chemical kinetics. *PNAS*, **107**(12): 5393–5398.

[4] Vilar, J. M. G., Kueh, H. Y., Barkai, N., Leibler, S., 2002. Mechanisms of noise-resistance in genetic oscillators. *PNAS*, USA, **99** (**9**): 5988–5992.

[5] Kar S., Baumann W. T., Paul M. R., Tyson J. J., 2009. Exploring the roles of noise in the eukaryotic cell cycle. *PNAS* USA, **106**: 6471–6476.

[6] Plesa, T., Zygalakis, K. C., Anderson, D. F., Erban, R., 2018. Noise control for molecular computing. *Journal of the Royal Society Interface*, **15**(**144**): 20180199.

[7] Plesa, T., Erban, R., Othmer, H. G, 2018. Noise-induced mixing and multimodality in reaction networks. *European Journal of Applied Mathematics*, 1-25. doi:10.1017/S0956792518000517.

[8] Briat, C., Gupta, A., Khammash, M., 2016. Antithetic integral feedback ensures robust perfect adaptation in noisy bimolecular networks. *Cell Systems*, **2**(**1**): 15–26.

[9] Del Vecchio, D., Abdallah, H., Qian, Y., Collins, J. J., 2017. A blueprint for a synthetic genetic feedback controller to reprogram cell fate. *Cell Systems*, **4**(**1**): 109–120.

[10] Briat, C., Gupta, A., Khammash, M., 2016. Antithetic proportional-integral feedback for reduced variance and improved control performance of stochastic reaction networks. *Journal of the Royal Society Interface*, **15**: 20180079.

[11] Ghomi, M. S. , Ciliberto, A., Kar, S., Novak, B., Tyson, J. J. 2008. Antagonism and bistability in protein interaction networks. *Journal of Theoretical Biology*, **250**: 209–218.

[12] Bressloff, P. C., 2017. Stochastic switching in biology: from genotype to phenotype. *Journal of Physics A: Mathematical and Theoretical*, **50**: 133001.

[13] Plesa, T., Stan, G. B., Ouldridge, T. E., and Bae, W., 2020. Robust control of biochemical reaction networks via stochastic morphing. Available as https://arxiv.org/abs/1908.10779.

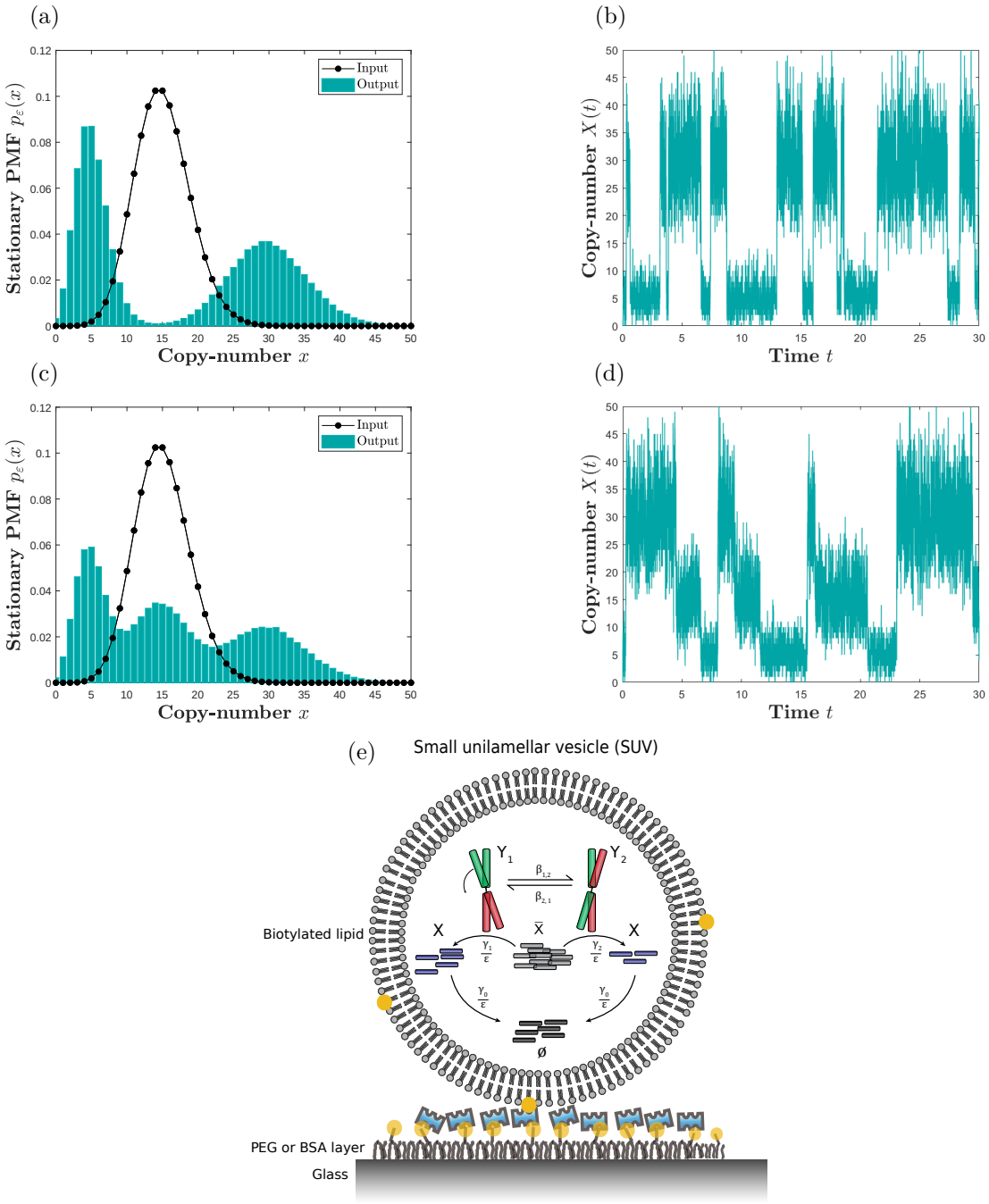


Figure 1: Panel (a) **displays in black the uni-modal stationary probability mass function** (PMF) **of a production-degradation input network, which is transformed into a desired output bi-modal form under the action of a lower-resolution stochastic morpher, shown as the cyan histogram. Panel** (b) **displays the corresponding bi-stable sample path/noisy time-series. Panels** (c) **and** (d) **show analogous plots when the input** PMF **is morphed into a tri-modal/tri-stable form. Panel** (e) **display a proposed experimental scheme for the stochastic morpher, involving an implementation of the stochastic morpher via a** DNA **Holliday junction molecule encapsulated in a vesicle, which switches between two distinct orientations Y_1 and Y_2, and catalytically produces the target species X via suitable** DNA **strand-displacement reactions.**

Genetic Circuit Hazard Analysis Using STAMINA

Lukas Bücherl[1], **Jeanet Mante**[1], **Pedro Fontanarrosa**[1], **Zhen Zhang**[2], **Brett Jepsen**[2], **Riley Roberts**[2], **Chris J. Myers**[1]
[1]University of Utah, [2]Utah State University
u1275346@utah.edu

1 INTRODUCTION

Synthetic Biology describes the approach to apply engineering principles in a biological context. One central concept of the field is the development of genetic parts that can be used in a repeatable manner to build genetic circuits. These genetic circuits are designed to control functions and the behavior of the cells in which they reside. These circuits may, for example, induce or suppress the production of different proteins in the cell in response to environmental signals. The genetic circuits can be used in many applications such as biosensors or drug delivery systems [4].

Inspired by engineering principles, the components of the genetic circuits are viewed in a more abstract way to allow the accessible building of the desired functions and modifications without the need to work directly on the DNA sequence. Similar to electrical circuits, genetic circuits are viewed as the combination of the known logic elements, like AND or OR gates. However, the behavior of genetic circuits is highly unpredictable compared to the binary behavior of electrical circuits since the molecule counts are low, leading to stochastic and noisy behavior [2].

Like electronic circuits, some input changes can cause unwanted switching variations (glitches) in the output of combinational genetic circuits. A *hazard* is the possibility of that glitch occurring. Though glitches are a transient behavior that corrects itself as a steady-state is reached, these unwanted variations can have drastic effects if the output produces irreversible effects, i.e., apoptosis or an early drug release. This means an in-depth analysis of non-avoidable hazards is necessary.

This paper describes the usage of the probabilistic model checker STAMINA [6] for determining the probability of a glitch in a genetic circuit. STAMINA is an infinite state stochastic model checker that reduces vast or infinite state *continuous time Markov chain* (CTMC) models to finite state representations using state space approximation methods. These reduced models are then manageable by existing stochastic model checkers like PRISM [5] and STORM [1].

2 RESULTS

The circuit used for the analysis is presented initially in the Cello paper [7] and shown in Figure 1. This circuit is interesting since a glitching behavior for this circuit was observed experimentally. Further details are in [3].

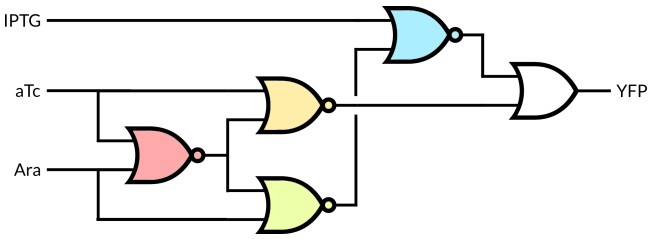


Figure 1: Circuit diagram for circuit 0x8E. The three inducer molecules are *IPTG*, *aTc* and *Ara* and the output is *YFP*. Following the American National Standard Institute the OR gate is represented by  and the NOR gate by  . The circuit was originally presented in the Cello paper [7].

The circuit reacts to the presence of the three inducer molecules *Ara*, *IPTG* and *aTc*. The inducer molecules have to be present for a prolonged time so that the input states can propagate through the different levels of logic. Four of the eight input combinations produce a high output indicated by production of *yellow fluorescent protein* (*YFP*).

Figure 2 shows the probability of a glitch over time for 15000 Monte Carlo simulation runs using iBioSim [9] for some input transitions with known hazards. The red graph for example, shows the inducer transit from *IPTG*, *aTc*, *Ara* = (1, 1, 1) to (1, 0, 0). The output signal is supposed to remain low during this input change, but after 1000 seconds, over 70 percent of the circuits *YFP* glitches to the high state.

Running a stochastic analysis in STAMINA shows that in 73.3% of the cases, *YFP* exceeds our chosen threshold of 30 molecules of YFP. This information is useful when designing genetic circuits: a designer can set a minimum threshold of cases that are allowed to glitch before input restrictions are needed. Depending on the purpose of this circuit's outcome, the threshold should be higher or lower. In the case at hand, if the output of the circuit results in apoptosis or drug release, input restrictions are necessary to avoid this input change, since in 73.3% of the cases, the output is triggered early.

Table 1 shows the glitch probability of all input transitions with known glitching behavior. The three columns of the Table 1 are the input transition, the probability of the glitch simulated in iBioSim, and the probability of the glitch calculated with STAMINA. As Table 1 shows, the results from iBioSim and STAMINA match in nine out of twelve cases. In two cases, STAMINA is failing to converge, while in a third

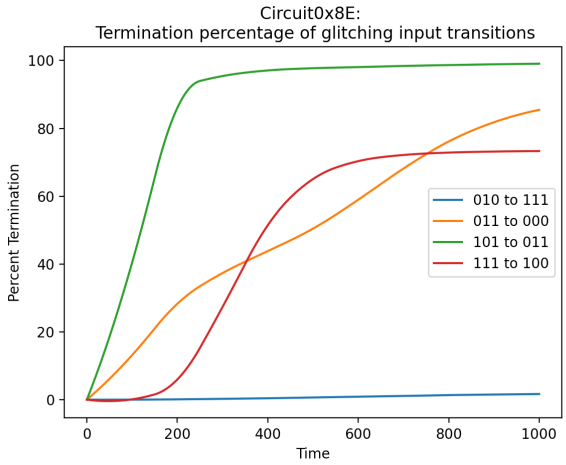


Figure 2: Probability of a glitch occurring over time for input changes known to have hazards calculated from 15000 Monte Carlo simulation runs in iBioSim. The legend shows the state transition of the three inputs *IPTG, aTc, Ara*. In example, for the blue graph the input transits from *IPTG, aTc, Ara* = (0,1,0) to (1,1,1) with a glitch probability of 1.66%.

it is getting too high a result. We are currently investigating the sources of these discrepancies.

Table 1: Glitch probabilities of input transitions simulated with iBioSim and STAMINA. The order of the inputs is *IPTG, aTc, Ara*.

Input Transition	iBioSim	STAMINA
$(0,1,0) \rightarrow (1,1,1)$	0.0166	Incomplete
$(0,1,0) \rightarrow (1,0,0)$	0.4021	0.3951 - 0.3958
$(1,1,1) \rightarrow (1,0,0)$	0.733	0.7351 - 0.7360
$(1,1,1) \rightarrow (0,1,0)$	0.6937	0.6947 - 0.6947
$(1,0,0) \rightarrow (0,1,0)$	0.454	0.4549 - 0.4558
$(1,0,0) \rightarrow (1,1,1)$	0.0168	Incomplete
$(0,1,1) \rightarrow (1,0,1)$	0.9895	1.0
$(0,0,0) \rightarrow (0,1,1)$	0.8256	1.0
$(0,0,0) \rightarrow (1,0,1)$	0.9901	1.0
$(1,0,1) \rightarrow (0,1,1)$	0.9905	1.0
$(0,1,1) \rightarrow (0,0,0)$	0.8545	0.8580 - 0.8585
$(1,0,1) \rightarrow (0,0,0)$	0.8736	0.8650 - 0.8658

3 DISCUSSION

As mentioned in the introduction, the inherent noisy and stochastic behavior of genetic circuits requires a sophisticated analysis. This paper presents some initial results using the infinite state stochastic model checker STAMINA to perform hazard analysis of complex genetic circuits. While iBioSim can be used to analyze static hazards, it is not able to analyze dynamic hazards or give insight into the causes of these hazards. Probabilistic model checkers, such as STAMINA, can provide such analysis.

Stochastic analysis can be used to predict the behavior of the analyzed genetic circuits, which will be critical to produce reliable genetic circuits design. Therefore, we plan to perform a further and more detailed analysis of different hazards for the presented genetic circuit and others.

This work uses a generic model generated within iBioSim using default parameters for the different reactions. However, we are planning to extend this work to use a characterized dynamic model [8] in the future. This would enable the user to predict glitch propensities and, in turn, help in the redesign process to avoid the glitches the user deems critical for the intended purposes of the circuit.

4 METHODS

The genetic circuit design and the generation of its model were achieved using the software tool iBioSim using the default, not on experimental data-based parameters. IBioSim was also used for running stochastic Monte Carlo simulations of the circuit to show its the glitching behavior. The model was exported as an SBML file and converted to a PRISM model using the SBML-to-PRISM translator implemented in PRISM. Finally, a stochastic analysis was run in STAMINA.

REFERENCES

[1] Dehnert, C., Junges, S., Katoen, J.-P., and Volk, M. A storm is coming: A modern probabilistic model checker. In *Computer Aided Verification* (Cham, 2017), R. Majumdar and V. Kunčak, Eds., Springer International Publishing, pp. 592–600.

[2] Elowitz, M. B., Levine, A. J., Siggia, E. D., and Swain, P. S. Stochastic gene expression in a single cell. *Science 297*, 5584 (Aug. 2002), 1183–1186.

[3] Fontanarrosa, P., Doosthosseini, H., Borujeni, A. E., Dorfan, Y., Voigt, C. A., and Myers, C. J. Genetic circuit dynamics: Function hazard and glitch analysis. Forthcoming.

[4] Khalil, A. S., and Collins, J. J. Synthetic biology: Applications come of age. *Nature Reviews Genetics 11*, 5 (May 2010), 367–379.

[5] Kwiatkowska, M., Norman, G., and Parker, D. PRISM 4.0: Verification of probabilistic real-time systems. In *Proc. 23rd International Conference on Computer Aided Verification (CAV'11)* (2011), G. Gopalakrishnan and S. Qadeer, Eds., vol. 6806 of *LNCS*, Springer, pp. 585–591.

[6] Neupane, T., Myers, C. J., Madsen, C., Zheng, H., and Zhang, Z. STAMINA: Stochastic approximate model-checker for infinite-state analysis. In *Computer Aided Verification* (Cham, 2019), I. Dillig and S. Tasiran, Eds., Springer International Publishing, pp. 540–549.

[7] Nielsen, A. A. K., Der, B. S., Shin, J., Vaidyanathan, P., Paralanov, V., Strychalski, E. A., Ross, D., Densmore, D., and Voigt, C. A. Genetic circuit design automation. *Science 352*, 6281 (Apr. 2016), aac7341.

[8] Shin, J., Zhang, S., Der, B. S., Nielsen, A. A., and Voigt, C. A. Programming *Escherichia coli* to function as a digital display. *Molecular Systems Biology 16*, 3 (Mar. 2020), e9401.

[9] Watanabe, L., Nguyen, T., Zhang, M., Zundel, Z., Zhang, Z., Madsen, C., Roehner, N., and Myers, C. ibiosim 3: A tool for model-based genetic circuit design. *ACS Synthetic Biology 8*, 7 (2019), 1560–1563.

Minimal model for protein expression accounting for metabolic burden

Fernando N. Santos and Jesús Picó*
Synthetic Biology and Biosystems Control Lab, Institute ai2, Universitat Politècnica de València (UPV)
{fersann1,jpico}@upv.es

1 INTRODUCTION

Design of synthetic genetic circuits without considering the impact of host circuit interactions results in an inefficient design process and lengthy trial-and-error iterations to appropriately tune protein expression levels. Microorganisms have evolved to reach an optimal use of cellular resources. This balance is perturbed by circuit-host interactions resulting from the interaction among the cell environment from which the cell takes substrates, its metabolism, and the needs of the synthetic genetic circuit introduced in the cell host. The resulting competition for common shared cell resources introduces spurious dynamics leading to problems of malfunctioning of the synthetic circuit due to lack of enough cellular resources. Therefore, there is an increasing interest in development of methods for model-based design of synthetic gene circuits considering host-circuit interactions. Here we present a medium-size model explaining host-circuit interactions caused by competition for shared resources and overcoming some of the drawbacks of either too over-simplified or too complex over-parameterised [4] existing models. We explicitly take into account relevant biological aspects and lab-accessible parameters, like promoter and RBS strengths, degradation of mRNA and proteins, generation of polysomes and ribosomes density, nutrient uptake and metabolization and biogenesis of ribosomes. The model, applied to *E. coli*, is able to predict with great precision the cell growth, the amount of free ribosomes and the effect of competition for them on protein expression and growth.

2 RESULTS

Growth rate depends monotonically on the amount of mature active ribosomes

In agreement with the accepted literature, our model predicts a monotonically relationship between the specific growth rate μ and the total amount of actively translating ribosomes (i.e. those involved in translating complexes at a given time instant) given by:

$$\mu(s_i) = \frac{m_{aa}}{m_c}\frac{\nu s_i}{K_{sc}+s_i}\Phi_r r_a, \tag{1}$$

where $\Phi_r r_a$ is the total amount of ribosomes actively translating, s_i is the amount of intracellular substrate, K_{sc} is a Michaelis-Menten parameter related to cell substrate uptake and catabolic capacity, ν the maximum translation rate per ribosome, m_c the protein cell mass, and m_{aa} the average mass of an amino acid. Figure 1 shows how Eq. 1 accurately predicts the experimental values of μ obtained from [1].

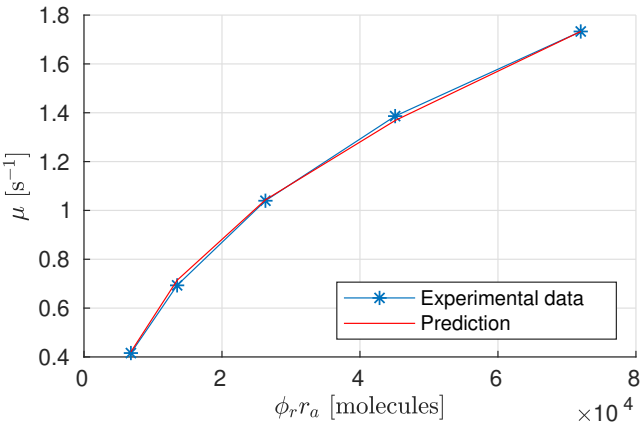


Figure 1: Experimental relationship [1] between the growth rate and the amount of translating ribosomes (blue) and predicition of the growth rate by appling Eq. 1 (red).

Only very small amount of free ribosomes is compatible with truly competition for cell resources

Estimation of the amount of free ribosomes in the cell, r, is key for assessing the competition among the cell circuits for cellular resources. Our model predicts a very low amount of r. If this was not the case, there would be no truly competition to recruit them. In addition, having too many r in excess would imply a superfluous use of energy for the cell. To evaluate the range of typical values of r, we used experimental data from [2] and evaluated the translation efficiency per mRNA $Y_{p/mRNA}$:

$$Y_{p/mRNA} = \frac{\beta_{pk}}{d_{mk}} = \frac{0.62}{l_e}\frac{\nu s_i}{K_{sc}+s_i}\frac{r}{\frac{d_{mk}}{K^k_{C^0}(s_i)}+\mu}, \tag{2}$$

where β_{pk} is the effective translation rate (protein/mRNA/t), d_{mk} is the mRNA degradation rate, l_e is the ribosome occupancy length and $K^k_{C^0}$ is a substrate dependent parameter

*Both authors contributed equally to this research. This research was partially supported by PAID-01-2017 and MINECO/AEI, EU DPI2017-82896-C2-1-R.

essentially related to the RBS strength. For any given protein and s_i, the amount of r will determine the required value of $K^k_{C^0}$ to attain the experimental value of $Y_{p/mRNA}$. All parameters in Eq.2 are known but $K^k_{C^0}$, which we estimated to be in the range $K^k_{C^0} \subset [0.02, 0.2]$ (molec^{-1}). From the results shown in Figure 2, a maximum amount of $r \approx 350$ could confidently explain the translation efficiencies per mRNA $Y_{p/mRNA}$ for almost all proteins in *E. coli*. This amount of r agrees with the estimation r = N(350, 35) in [3].

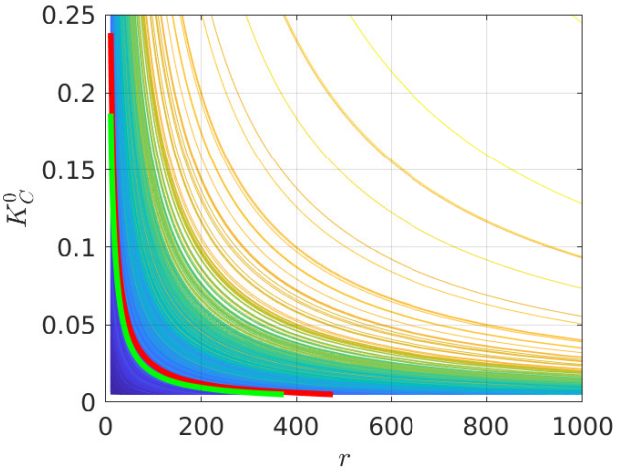


Figure 2: Relationship between the RBS strength-related term $K^k_{C^0}$ and the amount of free ribosomes r obtained using the experimental data in [2] for non-ribosomal proteins in *E. coli*. Thin lines correspond to the experimental value of the translation efficiency per mRNA $Y_{p/mRNA}$ for each protein. The red thick line corresponds to the mean for all proteins and the green thick line corresponds to the approximated mean when the term μr is neglected in Eq. 3.

Protein expression arises from the interaction of the resource recruitment strength of different genes

We defined the dimensionless function $J_k(\mu, r)$ which gives the strength with which the k-th gene recruits cellular resources to get expressed:

$$J_k(\mu, r) = E_{mk}\omega_k(T_f)\frac{1}{d_{mk}/K^k_{C^0}(s_i) + \mu r}, \tag{3}$$

where E_{mk} is related to the ribosomes density and ω_k is the transcription rate dependent on a transcription factor T_f. The resources recruitment strength function $J(\mu, r)$ provides a high level approach to understand competition among genes to be expressed. Figure 2 shows that the term μr could be neglected, this would make the $J(\mu, r)$ a constant value only dependent on T_f. Thus, the dynamics of a protein k (without active degradation) can be expressed as:

$$\dot{p}_k = \mu\left[\frac{m_c}{m_{aa}l_{pk}}\frac{J_k(\mu, r)}{\sum_{i=1}^{n_p} J_i(\mu, r)} - p_k\right], \tag{4}$$

where p_k is the amount of the protein k, l_{pk} its length and n_p the amount of different protein-coding genes. The term $J_k(\mu, r)/\sum_{i=1}^{n_p} J_i(\mu, r)$ shares similarities with Ohm's law, being $J(\mu, r)$ analogous to electrical resistance. Thus, increasing the expression of a protein k (i.e. increasing the value of $J_k(\mu, r)$) will increase the term $\sum_{i=1}^{n_p} J_i(\mu, r)$ for all the proteins and thus reducing the expression of the rest of proteins. Other models in the literature also exploit this analogy with the Ohm's law. However, in our model the $J(\mu, r)$ function explicitly relates the relevant metabolic variables (e.g. growth rate and ribosome availability) with the gene parameters (e.g. promoter and RBS strength). This makes it specially useful for its application to the model-based tuning of parameters in synthetic gene circuits.

3 DISCUSSION

Our model captures the effects of competition for shared cellular resources on protein expression and cell growth while keeping a good balance between simplicity and avoiding over-parametrization. Furthermore, it is able to explicitly relate relevant lab-accessible parameters (e.g. promoter and RBS strength) with growth rate and protein expression. Key in the model is the defined resources recruitment strength function $J(\mu, r)$, which allows to understand easily the protein-host interactions with explicit consideration of relevant lab-accessible parameters. Using basic experimental data available in the literature the model was able to predict the amount of active ribosomes, availability of free ribosomes, growth rate, required RBS strength in very good agreement with existing estimations. This assures its use for model-based tuning of gene synthetic circuits accounting for metabolic burden. In addition, the model can be easily integrated within a multi-scale one considering the extracellular environment at the bioreactor scale.

REFERENCES

[1] Dennis, P. P., and Bremer, H. Modulation of Chemical Composition and Other Parameters of the Cell at Different Exponential Growth Rates. *EcoSal Plus 3*, 1 (2008).

[2] Hausser, J., Mayo, A., Keren, L., and Alon, U. Central dogma rates and the trade-off between precision and economy in gene expression. *Nature Communications 10*, 1 (2019).

[3] Kaczanowska, M., and Rydén-Aulin, M. Ribosome Biogenesis and the Translation Process in Escherichia coli. *Microbiology and Molecular Biology Reviews 71*, 3 (2007), 477–494.

[4] Weisse, A. Y., Oyarzún, D. A., Danos, V., and Swain, P. S. Mechanistic links between cellular trade-offs, gene expression, and growth. *Proceedings of the National Academy of Sciences of the United States of America 112*, 9 (2015), E1038–E1047.

Bacteria Mastering the Tic-Tac-Toe Game Through Synthetic Adaptive Gene Circuits

Adrian Racovita[1],Satya Prakash[1],Clenira Varela[1],Mark Walsh[1],Roberto Galizi[2],Mark Isalan[3],Alfonso Jaramillo[1 4 5]*

[1]Warwick Integrative Synthetic Biology Centre and School of Life Sciences, University of Warwick, Coventry, UK; [2]Keele University, School of Life Sciences (CAEP), Keele, UK; [3]Department of Life Sciences, Imperial College London, UK; [4]CNRS, France; [5]Institute for Integrative Systems Biology (I2SysBio), University of Valencia-CSIC, Paterna, Spain.
*Correspondence to Alfonso.Jaramillo@synth-bio.org

1 INTRODUCTION

In this oral presentation we show how bacteria can be engineered to master simple but non-trivial games by reinforcement learning. We highlight our invention of a novel genetic device we named "memregulon" (from"memory regulon", a biological equivalent to the memristor device in physics). The memregulon combines the CAMERA analog memory [1] with a genetic system that regulates its expression depending on its history (**Fig. 1**). The system relies on an intracellular mixture of two plasmids (e.g. co-transformed) with the same logic gate, a fluorescent protein (GFP and RFP for plasmid A and B respectively) and an antibiotic resistance gene. Plasmid B is made to be indistinguishable from the original plasmid A except for the fluorescent protein and the loss-of-function mutation of the antibiotic resistance. We define a and b as the copy numbers per cell of plasmid A and B respectively. Previous work showed that they remain constant which was proposed as a memory system [1].

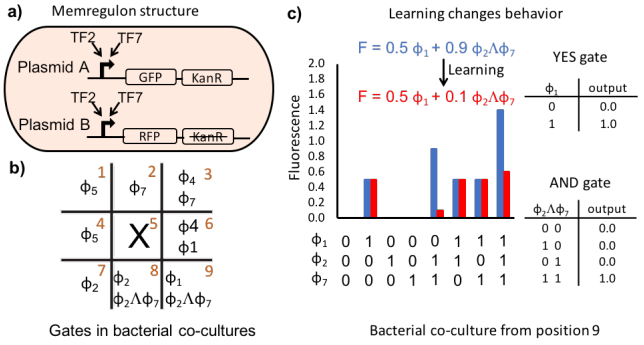


Figure 1: The memregulon. a) Memregulon genetic map. The $b/(a+b)$ ratio is defined as the weight ω (memory). ϕ=chemical input. Selection with Kanamycin changes the weight (learning). [1] b) Tic-Tac-Toe game board with an example of the gates used in one of our experimental implementations. The board is marked with numbers corresponding to suitable chemical inputs of an engineered *Escherichia coli* strain [2]. c) Truth table functionality. We describe a promoter with the combinatorial logic of the gate 2 Λ 7.

2 PLAYING AND LEARNING THE GAME

We implement complex gene circuits by exploiting the distribution of minimal gene circuits across a bacterial population. The additivity of the outputs is similar to the signal integration by a perceptron to provide an output if the signal is above a threshold, where the non-linearity is determined by the sigmoid behaviour of the transcription regulation. The output of several memregulons is combined using a winner-take-all (WTA) algorithm in the software analysis of the fluorescence.

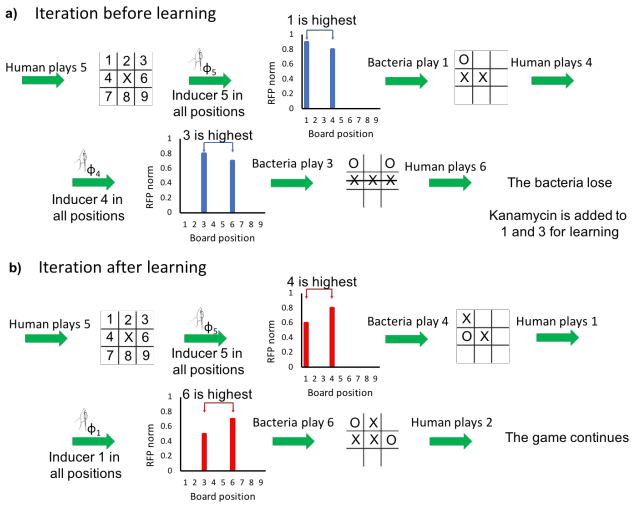


Figure 2: Experimental set-up of bacteria playing the game and learning from losses. a) Human (X) starts in position 5 and thus inducer 5 is added to every well. Because the YES5 memregulon has a higher red fluorescence in position 1 than 4, bacteria (O) is considered to play 1. Next, the human plays in position 4, the inducer 4 is added to every position and the bacteria play again the position with the highest red fluorescence (position 3). The bacteria lose as the human plays 6 which triggers a Kanamycin dose in the previously played positions 1 and 3. b) In the next game, the moves of the bacteria will be 4 instead of 1 and 6 instead of 3.

Theoretically, a library of one YES memregulon (**Fig. 1c**) for each of N independent chemical inputs is complete because any Boolean function of N inputs can be computed by WTA applied to weighted sums of inputs (**Fig. 4**). We have designed an experimental set-up where living bacteria can master the Tic-Tac-Toe game through reinforcement learning (**Fig. 2**). Playing requires the appropriate logic gates in each of the 9 board positions. The move of the first player is assigned to a chemical input which is added to every position. The highest fluorescence value will indicate the move of the second player. This is done iteratively until the game ends. Resigns occur if no fluorescence value is above a threshold. Reinforcement learning changes the memory of the bacteria based on the result. Thus, negative reinforcement decreases the weights of the activated memregulons. However, the increase in expertise (**Fig. 3**) diminishes with iterations as the probability of losing decreases after each learning.

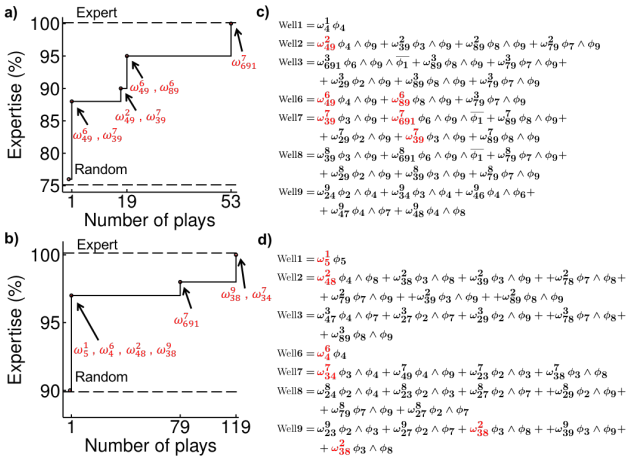


Figure 3: Bacteria learning against random players with only negative reinforcement (simulation data). Bacteria played 1st (a, c) or 2nd (b, d). We limited 1st and 2nd players to always play position 1 and 4 (or 4 and 1) in 2nd and 1st round respectively. a) and b) Gain in expertise through learning. Expertise was defined as percentage of positive results (wins and draws) by testing every possible game played against a random player. c) and d) Memregulon logic gates at each position, in red the weights changed at learnings.

An expert player was previously built (without learning) using catalytic single-stranded DNA molecules [4] (realising in vitro logic gates similar to (**Fig. 3c,d**), and in bacteria there have been computational reports regarding simplified learnings [4]. We have developed a software to simulate the behaviour of the bacteria based on experimental data characterised with our memregulon library (**Fig. 1**). **Fig. 3** shows that bacteria can play against a random player and in only 4 reinforcement learnings they reach 99% expertise.

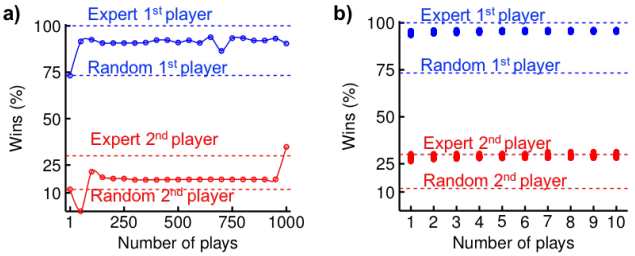


Figure 4: Bacteria learning with negative reinforcement (simulation data). a) Learning in a tournament of bacteria (blue) vs bacteria (red). b) Learning of 100 pairs of bacteria playing against each other using a random network of interactions and weight crossovers in only 10 plays.

We have simulated a memregulon network with an initial uniform set of weights which learned to above 90% wins (**Fig. 4**). **Fig. 4a** shows our results of simulating bacteria against bacteria. **Fig. 4b** depicts a simulation of 100 parallel tournaments of bacteria against bacteria. After each play, all 1st–1st player and 2nd–2nd player pairs are considered for weight crossover (where an offspring gets the addition of parent's weights, equivalent to experimentally mixing the cultures) and the parents are replaced by the offspring if they have more expertise. All 1st–2nd pairs are reshuffled. Importantly in only 10 rounds the bacteria achieved higher expertise than running single in 1,000 rounds. While reinforcement learning depends on the result, the genetic algorithm changes the weights independently of the output, creating diversity while the weight crossover maintains the initial expertise. After re-shuffling, reinforcement learning is again applied.

3 CONCLUSION

Our simulation results show that our memregulon library (**Fig. 3**) and even a simplified library of YES gates (**Fig. 4**) could allow living bacteria to master Tic-Tac-Toe through reinforcement learning if implemented experimentally. We are currently conducting the experiments to validate this.

REFERENCES

[1] Tang W. et al. Rewritable multi-event analog recording in bacterial and mammalian cells. Science. 2018;360(6385).
[2] Meyer AJ. et al. (2019) Escherichia coli "Marionette" strains with 12 highly optimized small-molecule sensors. Nat. Chem. Biol. 15, 196–204.
[3] Pei, R. et al. Training a molecular automaton to play a game. Nature Nanotech 5, 773–777 (2010).
[4] Didovyk A et al. Distributed classifier based on genetically engineered bacterial cell cultures. ACS Synth Biol. 2015;4(1):72-82.

BioCRNpyler: Compiling Chemical Reaction Networks from Parts in Diverse Contexts with Python

William Poole[1], **Ayush Pandey**[1], **Andrey Shur**[1], **Zoltan A. Tuza**[2], **Richard M. Murray**[1]
[1]California Institute of Technology, [2]Imperial College London
{wpoole,apandey,ashur}@caltech.edu,ztuza@imperial.ac.uk,murray@cds.caltech.edu

1 INTRODUCTION

Chemical Reaction Networks (CRNs) are commonly used for modelling in systems and synthetic biology [2]. The power of CRNs lies in their expressivity; CRN models can range from physically realistic descriptions of individual molecules to coarse-grained idealizations of complex multi-step processes [17]. However, this expressivity comes at a cost - frequently choosing the right level of detail in a model is more an art than a science. The modelling process requires careful consideration of the desired use of the model, the available data to parameterize the model, and prioritization of certain aspects of modelling or analysis over others.

The available tools for a CRN modeller are vast and include: extensive software to generate and simulate CRNs, databases of models, model analysis tools, and many more [5, 8, 10, 13, 15]. However, relatively few tools exist to aid in the automated construction of CRN models from simple specifications. For example, even though synthetic biologists have taken a module and part-driven approach to their laboratory work [3], models are still typically built by hand on a case-by-case basis.

The BioCRNpyler* package is a software framework and library designed to aid in the rapid construction of models from common motifs, such as molecular components, biochemical mechanisms and parameter sets. These customizable motifs can be reused and recombined to rapidly generate CRN models in diverse chemical contexts at varying levels of model complexity. Some similar tools exist include [7, 12, 16]. What makes BioCRNpyler unique is an open-source and object-oriented framework written in python which allows complete control over model compilation by developers as well as a large library of easy-to-use parts and models relevant to synthetic biologists and bio-engineers. The BioCRNpyler package is available on GitHub [1].

2 THE BIOCRNPYLER FRAMEWORK

BioCRNpyler is an open-source python framework (Figure 1) that compiles high-level specifications into detailed CRN models saved as SBML [9]. Specifications may include: biomolecular Components, modelling assumptions (Mechanisms), biochemical context (Mixtures), and Parameters. BioCRNpyler is written in python with a flexible object-oriented design, extensive documentation, and detailed examples to allow for easy model construction by modelers as well as customization and extension by developers.

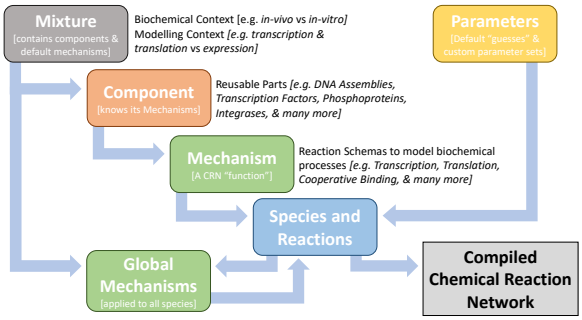


Figure 1: The hierarchical organization of classes in the BioCRNpyler framework. Arrows represent compilation.

Species and Reactions make up a CRN and are the output of BioCRNpyler compilation. Many sub-classes exist such as ComplexSpecies and reactions with different kinds of rate function (e.g. massaction, hill functions, etc).

Mechanisms are reaction schemas, which can be thought of as abstract functions that produce CRN Species and Reactions. They represent a particular molecular process, such as transcription or translation. During compilation, Mechanisms are called by Components. **Global Mechanisms** are called at the end of compilation in order to effect all species of a given type or with given attributes, for example, dilution of all protein Species.

Components are reusable parts; they know what kinds of Mechanisms effect them but are agnostic to the underlying schema. For example, a promoter is a Component which will call a transcription Mechanism; similarly a Ribosome Binding Site (RBS) is a Component which will call a translation Mechanism. However, the same Promoter and RBS can use many different transcription and translation Mechanisms depending on the modelling context and detail desired.

Mixtures are sets of default Mechanisms and Components that represent different molecular and modelling contexts. As an example of molecular context, a cell-extract model requires reactions to consume a finite supply of fuel while a steady-state model of living cells does not have a limited fuel

*Pronounced as bio-compiler

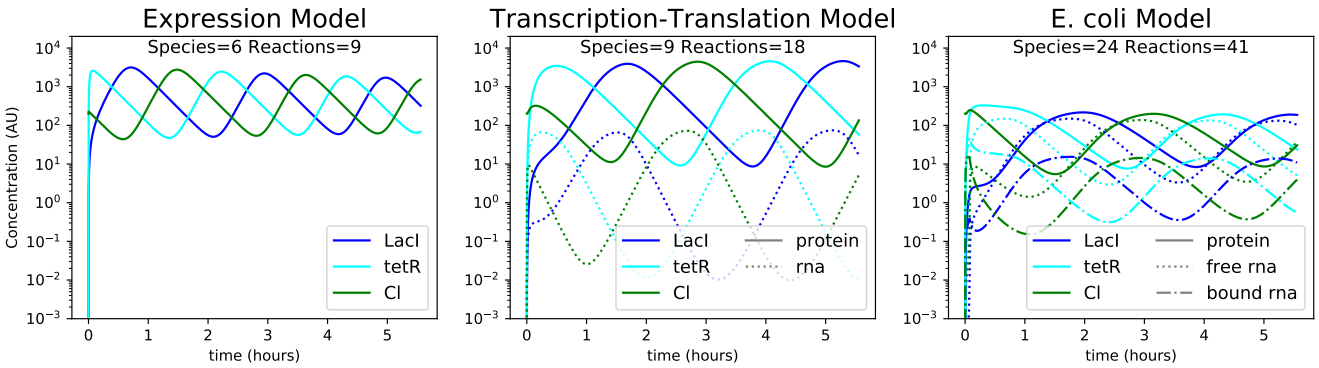


Figure 2: Using BioCRNpyler to compile the repressilator at various levels of detail. Simulation parameters come from the repressilator paper [6] and [4, 11]. Simulations were carried out with Bioscrape [14].

supply. As an example of modelling context, a simple model of gene expression may have a gene catalytically create a protein product while a more complex model might include cellular machinery such as RNA-Polymerase and Ribosomes with Michalis-Menten kinetics.

Parameters are designed for flexibility; they can default to biophysically plausible values (such as a default binding rate), be shared between Components and Mechanisms, or have specific values for Component-Mechanism combinations. This system is designed so that models can be produced quickly without full knowledge of all parameters and then refined with detailed parameter files later.

```
parameter_file = "default_parameters.txt"

expression_params = {("negativehill_transcription", "k"):2.8}
```
Parameters default for rapid model building and can be programmatically set via dictionaries or custom parameter files.

```
lacI = Species(name= "lacI", material_type = "protein")
tetR = Species(name = "tetR", material_type = "protein")
cI = Species(name = "cI", material_type = "protein")

pLac = RepressablePromotor(name = "pLac", transcript = "tetR", repressor = lacI)
pTet = RepressablePromotor(name = "pTet", transcript = "cI", repressor = tetR)
pCI = RepressablePromotor(name = "pCI", transcript = "lacI", repressor = cI)

placI_tetR = DNAassembly(name = "lacI_tetR", promoter = pLac, rbs = "BCD8", protein = tetR)
ptetR_cI = DNAassembly(name = "tetR_cI", promoter = pTet, rbs = "BCD8", protein = cI)
pcI_lacI = DNAassembly(name = "cI_lacI", promoter = pCI, rbs = "BCD8", protein = lacI)
```
Modular Components are combined together to produce diverse biochemical circuits.

```
RepressilatorExp = ExpressionDilutionMixture(name = "expression",
            components = [placI_tetR, ptetR_cI, pcI_lacI],
            parameter_file = parameter_file, #can use multiple parameter sources
            parameters = expression_params)  #custom parameters take precedent
RepressilatorExpCRN = RepressilatorTxTl.compile_crn()
```
Mixtures determine the context and level of detail used to model Components.

Expression CRN models gene expression (with leak) and dilution.

```
RepressilatorTxTl = SimpleTxTlDilutionMixture(name = "txtl",
            components = [placI_tetR, ptetR_cI, pcI_lacI],
            parameter_file = parameter_file)
RepressilatorTxTlCRN = RepressilatorTxTl.compile_crn()
```
Transcription-Translation CRN models transcription (with leak), translation, and dilution.

```
RepressilatorEcoli = TxTlDilutionMixture(name = "e coli",
            components = [placI_tetR, ptetR_cI, pcI_lacI],
            parameter_file = parameter_file)
RepressilatorEcoliCRN = RepressilatorTxTl.compile_crn()
```
E. coli CRN models transcription via RNA-Polymerase, translation via ribosomes, mRNA-degradation via endo-nucleases, background cellular loading, and dilution.

Figure 3: Python code generating three repressilator CRNs.

3 THE BIOCRNPYLER LIBRARY

The BioCRNpyler library contains a growing collection of Mechanisms, Components and Mixtures as well as extensive Jupyter notebooks. Currently, this library is geared towards synthetic biological applications with numerous Mechanisms for transcription, translation, gene regulation, catalysis, molecular binding and many more. Components include common synthetic biological parts such as Promoters, RBSs which can be combined into DNA-assemblies to produce RNA and Proteins, as well as more specific parts such as dCas9. Mixtures include both models of cell-like systems growing at steady state and extract-like systems with finite resources. Importantly, for different modelling contexts BioCRNpyler includes Mixtures with different default levels of complexity. The ease in generating increasingly complex models is illustrated in Figure 3 which shows code to compile a repressilator from a few common Components into multiple CRNs of very different levels of complexity. Simulations from these models are shown in Figure 2.

4 FUTURE WORK

We plan on extensively adding to available parts in the library. We will also add a module to import SBOL files to automatically generate models of DNA constructs commonly used in synthetic biology. Finally, we will connect BioCRNpyler to a parameter inference pipeline to characterize models from experimental data.

Acknowledgements: We would like to thank the Caltech BE240 class and Murray lab for extensive testing of this software and discussions of relevant models, parts, and used. In particular, we would like to thank Matthieu Kratz, Liana Merk, and Ankita Roychoudhury for contributing to the software library.

REFERENCES

[1] *BioCRNplyer.* https://github.com/BuildACell/BioCRNPyler.
[2] Alon, U. *An introduction to systems biology: design principles of biological circuits.* CRC press, 2019.
[3] Benner, S. A., and Sismour, A. M. Synthetic biology. *Nature Reviews Genetics 6*, 7 (2005), 533–543.
[4] Ceroni, F., et al. Quantifying cellular capacity identifies gene expression designs with reduced burden. *Nature methods 12*, 5 (2015), 415.
[5] Choi, K., et al. Tellurium: an extensible python-based modeling environment for systems and synthetic biology. *Biosystems 171* (2018), 74–79.
[6] Elowitz, M. B., et al. A synthetic oscillatory network of transcriptional regulators. *Nature 403*, 6767 (2000), 335–338.
[7] Harris, L. A., et al. Bionetgen 2.2: advances in rule-based modeling. *Bioinformatics 32*, 21 (2016), 3366–3368.
[8] Hoops, S., et al. Copasi-a complex pathway simulator. *Bioinformatics 22*, 24 (2006), 3067–3074.
[9] Hucka, M., et al. The systems biology markup language (sbml): a medium for representation and exchange of biochemical network models. *Bioinformatics 19*, 4 (2003), 524–531.
[10] Le Novere, N., et al. Biomodels database: a free, centralized database of curated, published, quantitative kinetic models of biochemical and cellular systems. *Nucleic acids research 34*, suppl_1 (2006), D689–D691.
[11] Milo, R., et al. *Cell biology by the numbers.* Garland Science, 2015.
[12] Myers, C. J., et al. ibiosim: a tool for the analysis and design of genetic circuits. *Bioinformatics 25*, 21 (2009), 2848–2849.
[13] Somogyi, E. T., et al. libroadrunner: a high performance sbml simulation and analysis library. *Bioinformatics 31*, 20 (2015), 3315–3321.
[14] Swaminathan, A., et al. Fast and flexible simulation and parameter estimation for synthetic biology using bioscrape. *bioRxiv* (2019), 121152.
[15] The MathWorks, Inc. Mathlab simbiology toolbox.
[16] Tuza, Z. A., et al. An in silico modeling toolbox for rapid prototyping of circuits in a biomolecular "breadboard" system. In *52nd IEEE Conference on Decision and Control* (Dec 2013), pp. 1404–1410.
[17] Vecchio, D. D., and Murray, R. M. *Biomolecular Feedback Systems.* Princton University Press, 2014.

Describing engineered biological systems with SBOL3 and ShortBOL2

Matthew Crowther[1], **Lewis Grozinger**[1], **James McLaughlin**[1], **Göksel Mısırlı**[2], **Jacob Beal**[3], **Bryan A. Bartley**[1], **Angel Goñi-Moreno**[1], **Anil Wipat**[1]
[1]Newcastle University, [2]Keele University, [3]Raytheon BBN Technologies
angel.goni-moreno@ncl.ac.uk,anil.wipat@ncl.ac.uk

1 INTRODUCTION

Data standards are essential to exchange information about the engineering of biological systems. The Synthetic Biology Open Language (SBOL) is a community-driven standard that facilitates the exchange of data relating to the design, implementation, testing and refinement of engineered biological systems [4]. Versions 1 and 2 of SBOL have gained widespread adoption, with over 170 developers, 29 SBOL supporting software tools and 42 institutions involved in their development and deployment (as of June 2020). Recently, SBOL was refactored to simplify its data model, resulting in the release of the SBOL3 specification [1].

SBOL data is created using both bespoke software tools and through the use of the SBOL specific software libraries such as pySBOL [2] and libSBOLj [6]. However, the creation of SBOL data using these libraries is often limited to tool developers with programming skills. To address this issue, numerous graphical computer aided design (CAD) tools have been developed to allow non-programmers to capture data from the design, build, test and learn (DBTL) cycle in SBOL format. However, visual design tools often support only a subset of features of the SBOL data model. Visual editing can be a slow and rather manual process that does not scale well for large designs such as genomes.

Previously, we have described a language, ShortBOL, which provides a human readable/writable short-hand for describing biological designs in SBOL [3]. ShortBOL was developed for synthetic biologists familiar with the SBOL data model who wish to rapidly describe synthetic biology designs using a text based scripting language instead of using a traditional programming language. Here, we describe a new release of ShortBOL, version 2.0. This new version sees the introduction of two new modes of use, as well as support for SBOL 3.0. The new modes of use of the language alter the levels of abstraction to consider two major categories of synthetic biologists with different expertise and backgrounds: (i) User mode for synthetic biologists with little to no understanding of the SBOL data model, and (ii) Developer mode for synthetic biology developers, familiar with programming and the SBOL data model. The new ShortBOL user-mode syntax simplifies the generation of SBOL data for the first category of users. The latter category of users are supported through full access to the terms of the data model. SBOL 3.0 support is provided for both these different levels of abstraction. We illustrate these two modes using examples developed in SBOL3. ShortBOL as a service can be accessed from http://shortbol.org and the code is available at https://github.com/intbio-ncl/shortbol.

2 EXPLORING SBOL3 USING SHORTBOL2

ShortBOL was originally designed to be easy to use by synthetic biologists who understand the fundamentals of the SBOL data model. A standard template library is incorporated within ShortBOL, allowing the introduction of new ShortBOL language terms and different aspects of genetic designs to be generated. SBOL3 support is provided via additional templates within the ShortBOL2 application.

We have also developed a tutorial and set of examples that illustrate the use of SBOL3, and its comparison to SBOL2, to aid developers in transitioning to the new version of the standard. For example, Figure 1 shows a side by side comparison of the TetR inverter module of the classical genetic toggle switch example [5] written using ShortBOL2 in developer mode for both SBOL2 and SBOL3 (Fig.1.). The ShortBOL2 application features support for generating both SBOL2.0 and SBOL3, the choice of which is selected by the user from the application menu. Depicting SBOL as ShortBOL allows the structure of SBOL3 to be viewed and also compared to the equivalent representation in SBOL2.0. SBOL is a data exchange format and SBOL data produced by different tools are all compatible. However, with the introduction of SBOL3 there is not currently compatibility between the SBOL2 and SBOL3 data models. As tooling for the data model is developed converters will become available.

3 ABSTRACTING SBOL3 USING SHORTBOL - USER MODE

The SBOL3 data model can still be unwieldy for designers unfamiliar with computational data representations. ShortBOL2 also introduces further templates that provide a more abstract version of the ShortBOL language, aimed at the average user who does not wish to work with the SBOL3 data model at a detailed level. This mode makes designs shorter and easier to understand. As an example, Figure 2 shows a

SBOL 2.0

```
pTetR is a Promoter()
lacI_RBS is a RBS()
lacI_CDS is a CDS()
lacI_term is a Terminator()

tetRInverter is a DNAComponent()
(
  precedes(pTetR,lacI_RBS)
  precedes(lacI_RBS,lacI_CDS)
  precedes(lacI_CDS,lacI_term)
)

tetRInverter_mod is a ModuleDefinition()
(
  stimulates(pTetR,lacI_CDS)
)
```

SBOL 3

```
pTetR is a Promoter()
lacI_RBS is a RBS()
lacI_CDS is a CDS()
lacI_term is a Terminator()

tetRInverter is a FunctionalEntity()
(
  precedes(pTetR,lacI_RBS)
  precedes(lacI_RBS,lacI_CDS)
  precedes(lacI_CDS,lacI_term)

  stimulates(pTetR,lacI_CDS)
)
```

Figure 1: A TetRInverter device shown in ShortBOL for SBOL 2.0 and SBOL3. The change from the use of ModuleDefinition to a single FunctionalEntity, introduced in SBOL3, is illustrated by a side-by-side comparison of the designs.

ShortBOL2 representation of the SBOL3 approach to defining the LacI inverter module of the genetic toggle switch. These ShortBOL templates allow for a more succinct representation of the design, which is then expanded out into the full representation on conversion into SBOL. Developers are still free to use the more expansive representation, if they wish, which is closer to the SBOL3 data model and allows expression of details that are not part of typical use patterns. The ShortBOL user mode has slight constrictions and is achieved using composition to create some SBOL3 objects behind the scenes. However, these constrictions are only present when working with very niche aspects of the data model.

ShortBOL 2.0 - Developer mode

```
TetR is a Protein()
LacI is a Protein()
TetR_sc is a SubComponent(TetR)
LacI_sc is a SubComponent(LacI)


LacI_lacinv_part is a Inhibitor(TetR_sc)
TetR_lacinv_part is a Inhibited(LacI_sc)


LacITetR_int is a Inhibition()
(
    hasParticipation = TetR_lacinv_part
    hasParticipation = LacI_lacinv_part
)

# The LacI inverter module
LacI_inverter is a FunctionalEntity()
(
  description = "LacI inverter"
  hasFeature = TetR_sc
  hasFeature = LacI_sc
  hasInteraction = LacITetR_int
)
```

ShortBOL 2.0 - User mode

```
TetR is a Protein()
LacI is a Protein()

# The LacI inverter module
LacI_inverter is a FunctionalEntity()
(
  description = "LacI inverter"
  inhibition(TetR,LacI)
)
```

Figure 2: A ShortBOL2 design for the LacI inverter module demonstrating the inhibition of TetR protein production by LacI protein in Developer mode and the more abstract version in User mode.

4 CONCLUSIONS AND FUTURE WORK

We have expanded ShortBOL1.0 to support SBOL3. ShortBOL2 aims to provide an easy to use tool for the composition of SBOL3 related data. This latest version introduces a user mode providing a more abstract version of ShortBOL where the SBOL data model can be used in a less granular fashion. ShortBOL2 allows the structure of SBOL3 to be explored succinctly by developers wishing to become familiar with the SBOL data model, who can gain exposure to the terminology and approach without having to work with the SBOL code libraries. Furthermore, users can also produce SBOL data without being exposed to the full details of the SBOL3 data model. ShortBOL makes it easier to prototype SBOL3 designs, and in the future it may be possible to simplify the ShortBOL syntax even further to provide a language that shields a user entirely from the SBOL3 data model.

5 ACKNOWLEDGEMENTS

This document does not contain technology or technical data controlled under either U.S. International Traffic in Arms Regulation or U.S. Export Administration Regulations. The authors of this work are supported by The Engineering and Physical Sciences Research Council grants EP/J02175X/1, EP/R003629/1, EP/N031962/1 (J.M. and A.W.), and EP/R019002/1 (A.G.-M.), EPSRC studentship 34000024085 (M.C.), and the European CSA 820699 (A.G.-M. and A.W.). M.C. is supported by Doulix Ltd. J.B. and B.B. are supported in part by the Air Force Research Laboratory (AFRL) and DARPA under contract FA875017CO184.

REFERENCES

[1] Baig, H., Fontanarrosa, P., Kulkarni, V., McLaughlin, J., Vaidyanathan, P., Myers, C., Bartley, B., Beal, J., Crowther, M., Gorochowski, T., et al. Synthetic biology open language (sbol) version 3.0.0. *Journal of Integrative Bioinformatics 17*, 2 (2020), 17.

[2] Bartley, B. A., Choi, K., Samineni, M., Zundel, Z., Nguyen, T., Myers, C. J., and Sauro, H. M. pysbol: a python package for genetic design automation and standardization. *ACS synthetic biology 8*, 7 (2018), 1515–1518.

[3] Crowther, M., Grozinger, L., Pocock, M., Taylor, C. P., McLaughlin, J. A., Misirli, G., Bartley, B. A., Beal, J., Goni-Moreno, A., and Wipat, A. Shortbol: A language for scripting designs for engineered biological systems using synthetic biology open language (sbol). *ACS Synthetic Biology 9*, 4 (2020), 962–966.

[4] Galdzicki, M., Clancy, K. P., Oberortner, E., Pocock, M., Quinn, J. Y., Rodriguez, C. A., Roehner, N., Wilson, M. L., Adam, L., Anderson, J. C., et al. The synthetic biology open language (sbol) provides a community standard for communicating designs in synthetic biology. *Nature biotechnology 32*, 6 (2014), 545–550.

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

[6] Zhang, Z., Nguyen, T., Roehner, N., Misirli, G., Pocock, M., Oberortner, E., Samineni, M., Zundel, Z., Beal, J., et al. libsbolj 2.0: a java library to support sbol 2.0. *IEEE life sciences letters 1*, 4 (2015), 34–37.

SBModEns: A Modular Toolbox for Model Building, Reduction, Analysis and Simulation in System Biology

Fernando N. Santos, Jose Luis Navarro and Jesús Picó*
Synthetic Biology and Biosystems Control Lab, Institute ai2, Universitat Politècnica de València (UPV)
{fersann1,jlnavarr,jpico}@upv.es

1 INTRODUCTION

The software consists of a set of tools that are used to define symbolic models in Matlab and to work easily and effectively with them. This is aimed to both inexperienced users, who need a simple environment to be able to simulate and model; and advanced users, who seek to automate (or reduce) much of the modeling, simulation, and analysis process.

2 DESCRIPTION OF THE TOOL

The initial work has been directed to the development of a common framework for model abstraction and tools integration. Then, other tools have been programmed in order to facilitate the construction of reduced-order models [4].

The tools available so far are:

- Modelling: Model construction with modular approach for reusing model parts or the whole system model. Also, a simple syntax is used for reactions definition. This tool transform reactions to internal model object. Furthermore, it generates a symbolic equation model in Matlab that can be used for simulation or system analysis.
- Invariant equations: From model connection graph, invariant species equations are found [3]. User must select the independent species that will be used in reduced models.
- Quasi Steady State (QSS) analysis: system dynamics not only depends on kinetic parameter. Some variables can exhibit negligible dynamics because they have higher order magnitude than others. For catching all QSS alternatives, dynamic simulations are drawn for system parameter guesses and initial concentration values. Then time response characteristics are calculated for confirming or rejecting QSS hypothesis.
- Flow analysis simplification: another approach for model reduction is to simplify small fluxes that are not relevant in the total balance of every species. So, flow trends are plotted and user can choose to make it zero or fixed in its steady state and compare if the simulation with this assumption is inside tolerance response.

Integration with existing workflows

The software follows the SOLID principles of object-oriented design: it is made up of a set of tools and each one performs a specific task, achieving the so-called open/closed principle that allows any user to expand the functionality but keeping the original one intact. This is a crucial advantage because it will allow the tool to be integrated into existing Matlab workflows effortlessly, and the user community could also extend the original functionality.

Model construction

It is the first and the basic tool. We could use the standard SBML language for system definition [2]. It is used broadly and it is very powerful, but in the first version of SBModEns we don't want to exploit all model functionality of SBML. So, we have developed a simple syntax (inspired in Open Modelica) to define biological models from chemical reactions or symbolic mathematical equations. In future releases we want to extend our system for importing SBML models.

Balanced equations for every reaction are drawn with kinetic constant (law of mass actions are assumed). The Variables (species names) and parameters (kinetic constants) are extracted and initial concentration values and parameter guesses are fixed. For example, an implementation of the antithetic controller [1] would be the following with our syntax:

```
0 -> z1, mu=1.0.
x2 -> x2 + z2, theta=1.0
z1 + z2 -> 0, nu=1.0
z1 -> z1 + x1, k1=1.0
x1 -> x1 + x2, k2=1.0
x1 -> 0, gamma1=1.0
x2 -> 0, gamma2=1.0
```

Listing 1: Definition of the model of the antithetic controller with our syntax.

And with just this code we have defined everything we need to use the rest of the tools that will be presented from now on.

*All authors contributed equally to this research. This research was partially supported by PAID-01-2017 and MINECO/AEI, EU DPI2017-82896- C2-1-R.

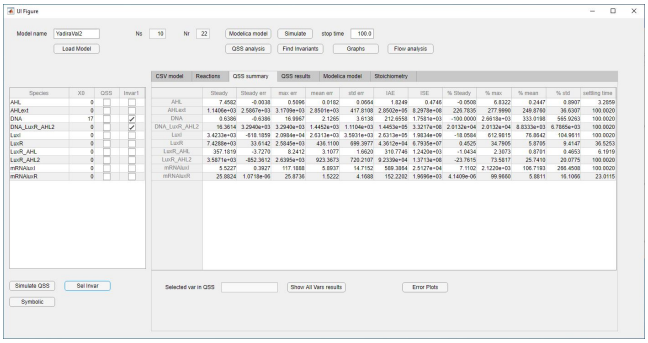


Figure 1: QSS analysys table: it shows model response error statistics for one extra species in QSS assumption. In left list check boxes one can select the species in QSS

Tools simplify and automate working with models

First, there is a parser tool that automatically converts the model defined in our syntax to a symbolic equations model in Matlab by applying the law of mass actions. Then, it is possible to simulate the model (even if it has algebraic loops, which is a common case when using the quasi-steady-state approximation). Furthermore, it is even possible to do stochastic simulations because we have information on the chemical reactions that make up the model.

Many of the existing tools in Matlab require an explicit ode function of the model (e.g., optimization tools, simulink, etc.), so there is another parser that automatically generates the code of these functions without dependencies to our tool and it can even generate S-Functions. As a result, this set of tools can be easily integrated into different workflows in Matlab.

Thanks to having the symbolic model in Matlab, there is a mathematical analysis tool that can calculate equilibrium points, linearize the model, find eigenvalues, calculate the Jacobian matrix, etc. This allows to standardize and automate this type of analysis, which democratizes the use of these techniques and allows users not to waste time programming them.

The open/closed principle allows us to extend the original functionality easily. For example, we have developed a tool that applies a contractivity test. Contractivity is a technique for analyzing the stability of a system, and the advantage it has over another approach like Lyapunov is that the test is a list of mechanical steps that can be programmed, however, its results are usually quite conservative. Once we have programmed the test, it is virtually free to apply the test in each of the models that we have already defined and this could lead to valuable insights.

Finally, there is a tool that converts symbolic models into code for LaTeX documents. In this way, one can automate the process of writing the models on the paper.

REFERENCES

[1] Briat, C., Gupta, A., and Khammash, M. Antithetic Integral Feedback Ensures Robust Perfect Adaptation in Noisy Bimolecular Networks. *Cell Systems 2*, 1 (2016), 15–26.

[2] Hucka M, Bergmann FT, D. A. e. a. The systems biology markup language (sbml): Language specification for level 3 version 2 core. *Journal of Integrative Bioinformatics 15*, 1 (Mar. 2018).

[3] Mirzaev, I., and Gunawardena, J. Laplacian Dynamics on General Graphs. *Bulletin of Mathematical Biology 75*, 11 (Nov. 2013), 2118–2149.

[4] Snowden, T. J., van der Graaf, P. H., and Tindall, M. J. Methods of Model Reduction for Large-Scale Biological Systems: A Survey of Current Methods and Trends. *Bulletin of Mathematical Biology 79*, 7 (July 2017), 1449–1486.

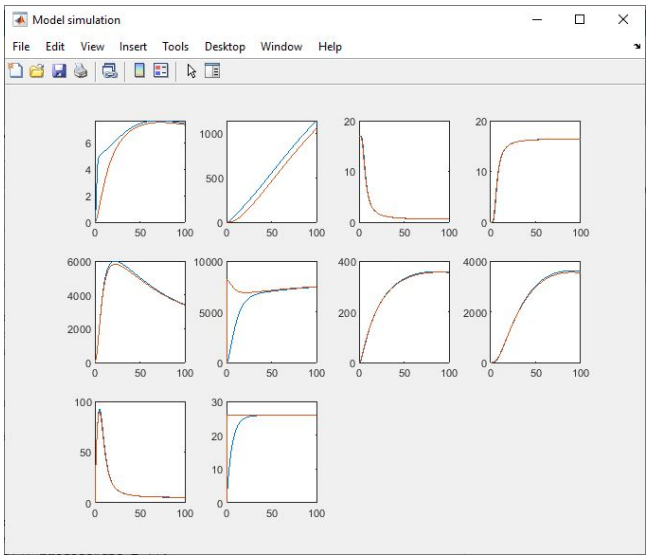


Figure 2: QSS trends: it compare full system and system under QSS hypothesis.

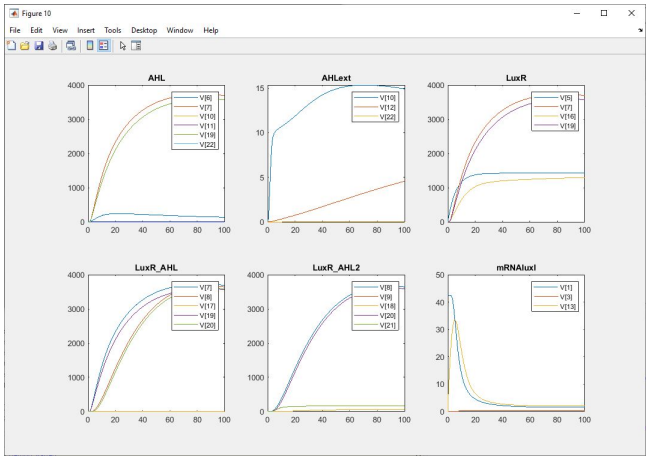


Figure 3: Flow trends: it can show negligible or near-constant flows for user assistance in model reduction.

Accurate, Complete, and Contiguous Engineered Yeast Genomes with Prymetime

Eric M. Young
Joseph H. Collins
Kevin W. Keating
Worcester Polytechnic Institute
Worcester, MA, USA
emyoung@wpi.edu,jhcollins@wpi.edu
kwkeating@wpi.edu

Nicholas Roehner
Aaron Adler
Tom Mitchell
Bryan Bartley
Raytheon BBN Technologies
Cambridge, MA, USA
fourth@name.org

1 INTRODUCTION

Whole genome sequencing (WGS) is an attractive method for evaluating genetic engineering. Yet, WGS is not often used, despite increasing evidence of unintended results of genetic engineering detected with WGS [1, 5, 7, 10–13], and many unpublished accounts of WGS revealing unintended edits in engineered industrial strains. Clearly, WGS is needed to detect and validate genetic engineering.

WGS is particularly needed for engineered yeast strains with deletions[3], plasmids[4], insertions[9], and SCRaMbLEd chromosomes[2, 6]. For WGS results to be useful, it must be possible to accurately resolve yeast genome integrations and episomal plasmids. Such ability may also enable detection of synthetic parts within mixed or unknown samples. This is not possible with current methods. Assembly of genetic engineering lacks accuracy, completeness, and contiguity, and synthetic parts are not labeled.

Here, we report an integrated workflow, named Prymetime, that uses sequencing reads from inexpensive NGS platforms, assembly and error correction software, and a list of genetic parts to achieve accurate whole genome sequences of yeasts with genetic parts annotated. We show it can accurately reproduce many edits in one genome and detect engineering signatures in a complex metagenomic sample.

2 DEVELOPING THE PIPELINE

We set a standard that our genome assembly workflow must be able to resolve chromosomal integrations and multiple plasmids used in yeast engineering. To achieve this standard, we needed to integrate both short and long read sequencing, along with assembly algorithms for chromosomes and plasmids. This hybrid approach is necessary to achieve accuracy, contiguity, and contigs with both chromosomes and closed plasmids. The final workflow, called Prymetime, is depicted in Figure 1. Prymetime is an acronym for "Pipeline for Recombinant Yeast genoMEs That Identifies Markers of Engineering."

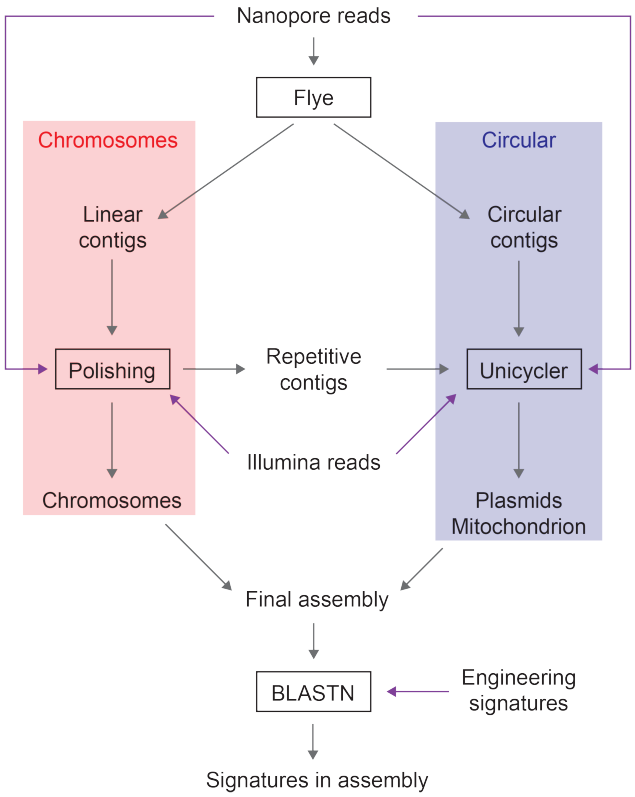


Figure 1: Prymetime software pipeline depicting the hybrid assembly approach.

Validation with a heavily engineered yeast strain

We built a *S. cerevisiae* CEN.PK113 strain containing an integrated carotenoid pathway, the native 2μ plasmid, a dCas9 plasmid, and a gRNA plasmid, shown in Figure 2a. We named this strain "FEY_2." This strain captures a number of engineering features common in metabolic engineering and synthetic biology. The particular challenges in this strain for genome assembly are two plasmids that share a great deal of sequence, and parts repeated from the genome. These

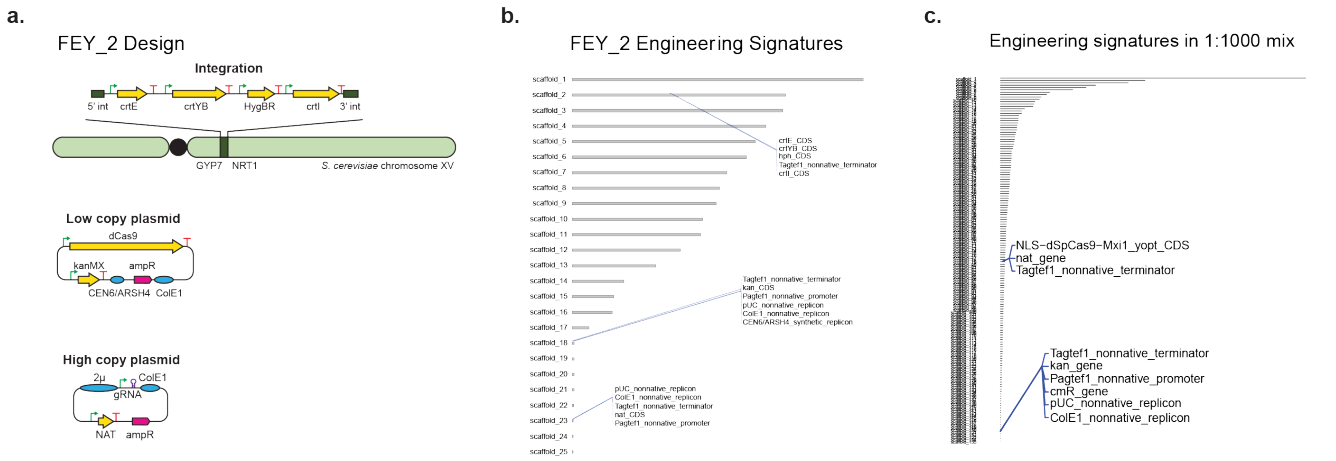


Figure 2: Prymetime software pipeline depicting the hybrid assembly approach.

high-identity features can easily confuse assembly software.

We sequenced FEY_2 with an Oxford Nanopore MinION and Illumina iSeq 100 to obtain long and short reads, respectively, at a read depth of 40X. We observed that in this step it is key to use tagmentation-based sequencing library preparations to obtain sufficient reads from plasmids. The reads were then passed through the Prymetime software package, resulting in a complete *S. cerevisiae* genome with the exact engineering signatures intended. A visualization of this assembly is shown in Figure 2b. This plot is produced automatically in a Prymetime run for facile identification of engineering signatures.

Detecting signatures in metagenomic samples

We next attempted to resolve signatures of engineering in a metagenome assembly with Prymetime. Publicly available reads from the Zymo mock metagenome[8] were combined with reads from another heavily engineered yeast strain to simulate detection of an engineered strain in a mixed sample. Integrations and plasmids were completely resolved even at a dilution of 1:1000 (Figure 2c). This shows that synthetic biology parts can be resolved in mixed samples, even when the strain containing the part is present at only one cell per thousand. Therefore, Prymetime may be used to detect known engineering signatures in unknown, mixed samples.

3 DISCUSSION

Our approach permits rapid, on-site acquisition of reference quality yeast genome sequences and annotation of genetic parts, overcoming barriers in strain validation and detection of parts in mixed samples. To do this, we used inexpensive sequencing platforms and the Prymetime software package. With read depth of 40X, we estimate that up to 30 *S. cerevisiae* genomes can be sequenced on one MinION flow cell and up to 4 genomes can be sequenced on one Illumina iSeq flow cell and still achieve useful results. Thus, it is now possible to accurately detect and validate genetic engineering in yeasts with whole genome sequencing. The Prymetime script is avialable at https://github.com/emyounglab/prymetime.

4 ACKNOWLEDGEMENTS

The authors thank James Kingsley at WPI for his help implementing Prymetime on WPI's server. This research is based upon work supported in part by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA) under Finding Engineering Linked Indicators (FELIX) program contract #N66001-18-C-4507. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein. This work is also supported by Worcester Polytechnic Institute startup funds.

REFERENCES

[1] Anton, B. P., Fomenkov, A., Raleigh, E. A., and Berkmen, M. Complete genome sequence of the engineered escherichia coli shuffle strains and their wild-type parents. *Genome Announcements 4*, 2 (2016), e00230–16.

[2] Blount, B. A., Gowers, G.-O. F., Ho, J. C. H., Ledesma-Amaro, R., Jovicevic, D., McKiernan, R. M., Xie, Z. X., Li, B. Z., Yuan, Y. J., and Ellis, T. Rapid host strain improvement by in vivo rearrangement of a synthetic yeast chromosome. 1932.

[3] Brachmann, C. B., Davies, A., Cost, G. J., Caputo, E., Li, J., Hieter, P., and Boeke, J. D. Designer deletion strains derived from saccharomyces cerevisiae s288c: A useful set of strains and plasmids for pcr-mediated gene disruption and other applications. *Yeast 14*, 2 (1 1998), 115–132.

[4] DiCarlo, J. E., Conley, A. J., Penttilä, M., Jäntti, J., Wang, H. H.,

and Church, G. M. Yeast oligo-mediated genome engineering (yoge). *ACS Synthetic Biology 2*, 12 (Dec 2013), 741–749.

[5] Gallegos, J. E., Hayrynen, S., Adames, N., and Peccoud, J. Challenges and opportunities for strain verification by whole-genome sequencing. *bioRxiv* (2019).

[6] Gowers, G.-O. F., Chee, S. M., Bell, D., Suckling, L., Kern, M., Tew, D., McClymont, D. W., and Ellis, T. Improved betulinic acid biosynthesis using synthetic yeast chromosome recombination and semi-automated rapid LC-MS screening. 868.

[7] Li, J., Manghwar, H., Sun, L., Wang, P., Wang, G., Sheng, H., Zhang, J., Liu, H., Qin, L., Rui, H., Li, B., Lindsey, K., Daniell, H., Jin, S., and Zhang, X. Whole genome sequencing reveals rare off-target mutations and considerable inherent genetic or/and somaclonal variations in crispr/cas9-edited cotton plants. *Plant Biotechnology Journal 17*, 5 (2019), 858–868.

[8] Nicholls, S. M., Quick, J. C., Tang, S., and Loman, N. J. Ultra-deep, long-read nanopore sequencing of mock microbial community standards. *GigaScience 8*, 5 (05 2019). giz043.

[9] Ronda, C., Maury, J., Jakočiunas, T., Baallal Jacobsen, S. A., Germann, S. M., Harrison, S. J., Borodina, I., Keasling, J. D., Jensen, M. K., and Nielsen, A. T. CrEdit: CRISPR mediated multi-loci gene integration in saccharomyces cerevisiae. 97.

[10] Schwarzhans, J.-P., Wibberg, D., Winkler, A., Luttermann, T., Kalinowski, J., and Friehs, K. Non-canonical integration events in pichia pastoris encountered during standard transformation analysed with genome sequencing. *Scientific Reports 6* (Dec 2016), 38952 EP –. Article.

[11] Solis-Escalante, D., van den Broek, M., Kuijpers, N. G., Pronk, J. T., Boles, E., Daran, J. M., and Daran-Lapujade, P. The genome sequence of the popular hexose-transport-deficient saccharomyces cerevisiae strain eby.vw4000 reveals loxp/cre-induced translocations and gene loss. *FEMS Yeast Res 15*, 2 (2015).

[12] Veres, A., Gosis, B., Ding, Q., Collins, R., Ragavendran, A., Brand, H., Erdin, S., Cowan, C. A., Talkowski, M., and Musunuru, K. Low incidence of off-target mutations in individual crispr-cas9 and talen targeted human stem cell clones detected by whole-genome sequencing. *Cell Stem Cell 15*, 1 (2014), 27 – 30.

[13] Young, A. E., Mansour, T. A., McNabb, B. R., Owen, J. R., Trott, J. F., Brown, C. T., and Van Eenennaam, A. L. Genomic and phenotypic analyses of six offspring of a genome-edited hornless bull. *Nature Biotechnology* (2019).

Decodon Calculator: Degenerate Codon Set Design for Protein Variant Libraries

Dimitris Papamichail*
papamicd@tcnj.edu
The College of New Jersey
Ewing, New Jersey, USA

Nicholas Carpino
The College of New Jersey
Ewing, New Jersey, USA

Tomer Aberbach
The College of New Jersey
Ewing, New Jersey, USA

Georgios Papamichail
New York College
Athens, Greece

1 INTRODUCTION

Mutant libraries representing protein variants are increasingly used to optimize protein function. Protein Engineering involves screening mutant libraries for novel proteins that show enhanced expression levels, solubility, stability, or enzymatic activity. To reach such objectives, it is often necessary to modify extant proteins, developing variants with improved properties [3, 4]. However, there exists a massive space of potential variants to consider.

Computational design of combinatorial libraries [1, 2, 6, 7] provides a reasonable approach in the development of improved variants. Library-design strategies seek to experimentally evaluate a diverse but focused region of sequence space in order to improve the likelihood of finding a beneficial variant. Such an approach is based on the premise that prior knowledge can inform generalized predictions of protein properties, but may not be sufficient to specify individual, optimal variants. Libraries are particularly appropriate when the prior knowledge does not admit detailed, robust modeling of the desired properties, but when experimental techniques are available to rapidly assay a pool of variants.

The design of mutant protein libraries typically involves a manual process in which required sites for mutation are selected and ambiguous *degenerate* codons (those containing mixtures of nucleotides) are designed to introduce controlled variation in these positions. This is particularly useful in cases where definitive decisions regarding specific amino acid substitutions are non-obvious [4]. The design of the protein variant library is complemented by use of synthesized degenerate oligonucleotides which enable annealing based recombination. Custom oligonucleotide overlaps enable the targeted introduction of crossovers at only desired positions, in turn enabling the desired level and type of diversity in a combinatorial library.

*Corresponding author

Table 1: Degenerate Bases and their codings

Degenerate Base	Actual Bases Coded
N	A or C or G or T
B	C or G or T
D	A or G or T
H	A or C or T
V	A or C or G
K	G or T
M	A or C
R	A or G
S	C or G
W	A or T
Y	C or T

2 THE PROBLEM: TARGETED MUTANT PROTEIN LIBRARIES

Traditional mutant protein library design methods involve the incorporation of a single degenerate codon (thereafter referred to as *decodon*) at each position where amino acid substitutions are explored. Decodons contain ambiguous bases (*degenerate* bases), as shown in Table 1.

An online tool called CodonGenie [5] was created to aid the effort of designing decodons that code for any provided set of amino acids. The CodonGenie tool ranks candidate decodons by specificity, attempting to minimize coding of undesirable amino acids and/or STOP codons. Even so, when using a single decodon to code for a set of amino acids, it is often unavoidable to code for additional unwanted amino acids. Using an example from [5], when coding the non-polar residues A, F, G, I, L, M and V, CodonGenie picks decodon DBK ([AGT][CGT][GT]) as its top choice, which, in addition to the desired set, codes also for amino acids C, R, S, T, and W. In total, the decodon DBK codes 26 total DNA variants, 18 DNA variants coding for desired amino acids, and 8 DNA variants for undesirable ones.

In our work we explored the coding of a set of amino acids by potentially multiple decodons. The usage of annealing based recombination of degenerate oligos containing the decodons can produce libraries on the productive portion of the space by eliminating unwanted mutations, therefore improving the yield of beneficial variants and the overall quality of the library. In turn, this method can significantly reduce labor costs assaying the pool of variants, at the expense of additional oligo synthesis, whose comparative cost is modest and continuously dropping.

The Decodon Calculator Tool

We have designed and implemented an algorithm that, given any set of amino acids, produces the minimum number of decodons necessary to code for exactly this set, i.e. without coding for extraneous amino acids or STOP codons. There are 15 nucleotide codes ("letters"), ranging from the completely unambiguous A, C, G and T representing a single nucleotide, to the completely ambiguous N representing all 4 nucleotides. There are $15^3 = 3,375$ decodons that can be assembled from this 15-letter alphabet of ambiguous codes, compared to the $43 = 64$ codons that can be constructed from the standard 4-letter alphabet of unambiguous nucleotides.

Using our algorithm we calculated minimum cardinality decodon sets for all $2^{20} - 1 = 1,048,575$ possible amino acid subsets. Our results indicate that 6 decodons are always sufficient to code for any amino acid subset, where at most 4 decodons are sufficient to encode more than 90% of all amino acid subsets. Our algorithm also produces an example of a decodon set of minimum cardinality for each amino acid subset.

We also built a web tool called *Decodon Calculator* that allows the calculation of the minimum number of decodons needed to code any amino acid subset. Once a set of amino acids is selected and the Submit button is pressed, results are displayed on the bottom of the screen, as shown in in Figure 1. In this particular example, we can observe that the non-polar residues A, F, G, I, L, M and V can be coded by the two degenerate codons DTB and GBA, which code for 12 desirable DNA variants, in contrast to the 26 variants of the single best decodon generated by CodonGenie, 8 of which are undesirable.

The Decodon Calculator can be accessed at http://algo.tcnj.edu/decodoncalc/.

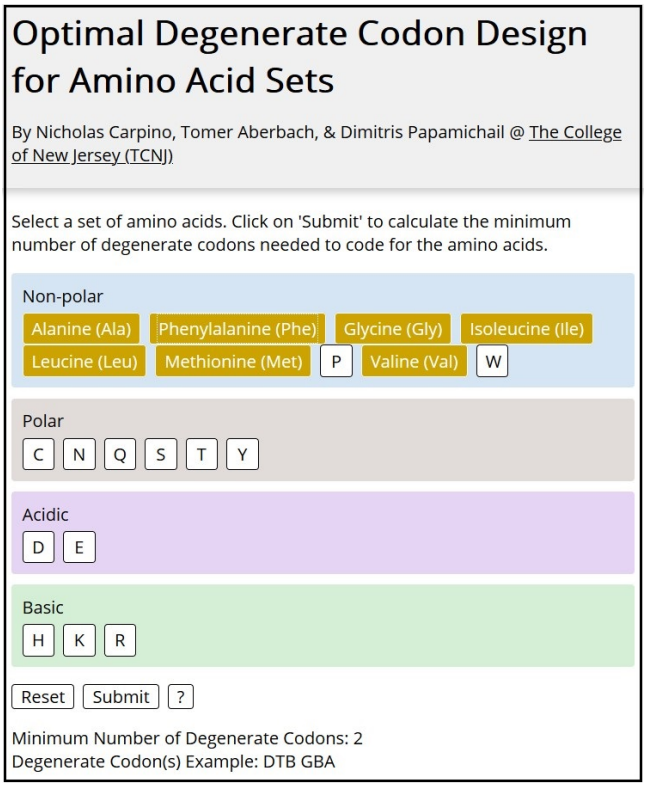


Figure 1: Calculating the minimum number of decodons necessary to encode the amino acid set { }

3 ACKNOWLEDGEMENTS

This work has been supported in part by NSF Grant CCF-1418874. The authors also acknowledge use of the ELSA high performance computing cluster at The College of New Jersey for hosting the web service reported in this paper. This cluster is funded by the NSF grant OAC-1828163.

REFERENCES

[1] Meyer, M. M., Silberg, J. J., Voigt, C. A., Endelman, J. B., Mayo, S. L., Wang, Z.-G., and Arnold, F. H. Library analysis of SCHEMA-guided protein recombination. *Protein Science* (2003).

[2] Pantazes, R. J., Saraf, M. C., and Maranas, C. D. Optimal protein library design using recombination or point mutations based on sequence-based scoring functions. *Protein Engineering, Design and Selection* (2007).

[3] Parker, A. S., Zheng, W., Griswold, K. E., and Bailey-Kellogg, C. Optimization algorithms for functional deimmunization of therapeutic proteins. *BMC Bioinformatics* (2010).

[4] Reetz, M. T., and Carballeira, J. D. Iterative saturation mutagenesis (ISM) for rapid directed evolution of functional enzymes. *Nature Protocols* (2007).

[5] Swainston, N., Currin, A., Green, L., Breitling, R., Day, P. J., and Kell, D. B. CodonGenie: Optimised ambiguous codon design tools. *PeerJ Computer Science* (2017).

[6] Treynor, T. P., Vizcarra, C. L., Nedelcu, D., and Mayo, S. L. Computationally designed libraries of fluorescent proteins evaluated by preservation and diversity of function. *Proceedings of the National Academy of Sciences of the United States of America* (2007).

[7] Voigt, C. A., Martinez, C., Wang, Z. G., Mayo, S. L., and Arnold, F. H. Protein building blocks preserved by recombination. *Nature Structural Biology* (2002).

Automation of polycistronic small RNA design through Golden Gate assembly

Uriel Urquiza-García[1,2], Christoph Wagner[1], Sascha Ferraro[1] and Matias Zurbriggen[1,2]
[1]Institute for Synthetic Biology, Heinrich-Heine-University, Duesseldorf.
[2]CEPLAS, Cluster of Excellence on Plant Sciences.

1 ABSTRACT

The tRNA processing system is highly conserved in animals and plants. The processing capabilities have been recently harnessed for producing small guide RNAs used in CRISPR/CAS9 applications [5]. The RNAs of interest can be flanked by tRNAs and incorporated in a synthetic intron. The architecture can be represented as (tRNA-sgRNA$_1$-tRNA-sgRNA$_2$-tRNA-sgRNA$_n$-tRNA) named (Polycistronic tRNA-sgRNA, PTG). The PTG is transcribed by a Pol-II polymerase, after which the endogenous tRNA processing RNAses free the sgRNAs that can then be used in different CRISPR/CAS9 applications. The latest and most exciting being Prime Editors [1]. Nonetheless, the PTG assembly can only be performed by Golden-Gate-like (GGL) approaches. Optimal GGL overhangs can be computed on the variable sgRNA regions, however manual design is error prone especially for large multiplexed designs. Furthermore, Prime Editors (PE) are significantly more complex to design than standard sgRNAs which further complicates the design of synthesis strategies. Therefore, we developed and experimentally validated a python CAD tool (PolyGEN, Polycistron Generator, Fig. 1B) capable of automating the design of oligonucleotides for synthesising PTGs through GG assembly.

2 OUTLINE

The problem with building intron-codified tRNA-sgRNA-tRNA polycistrons is their repetitive nature, which complicates the assembly in conventional DNA synthesis. This issue can be solved by using Golden Gate assembly [3, 5]. Type-II restriction enzymes allow scarless assembly as these cut several bases downstream of their recognition site. Sites can be added using custom primers with PCR. The short length of sgRNAs allows the amplification of tRNA and gRNA-tRNA scaffolds that will share the RNA of interested added on the 3' and 5' side respectively. Therefore, the sgRNA is divided in two fragments and is assembled by Golden Gate cloning. A 4 bp overlap region inside the sgRNA should be chosen that will provide the overhangs for Golden Gate assembly. This selection is not a trivial process and tools have been generated for easing this process [4]. Furthermore, in the case of

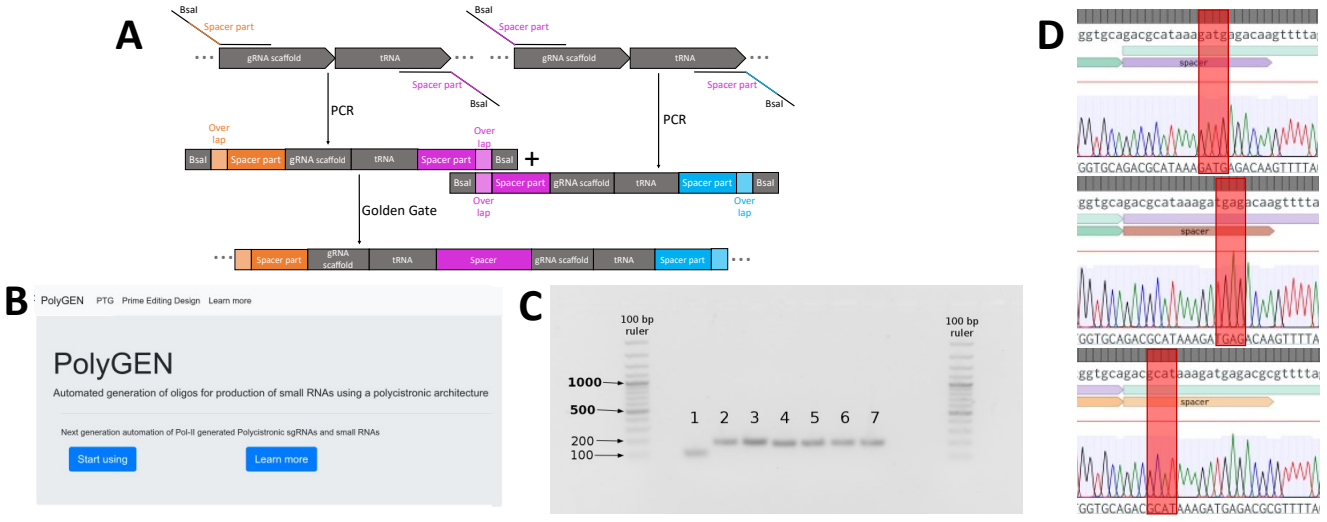


Figure 1: (A) Experimental workflow using the computationally designed primers. Based on a template sequence, computed primers can bind to the unchanging gRNA-tRNA backbone and produce inserts capped off by unique overlaps. During Golden Gate assembly, the overlaps are exposed by BsaI and the entire construct is allowed to assemble in a unique manner. (B) Homepage of the PolyGEN webapp. (C) PCR amplification of unique inserts for pUU105-pUU107. (D) Sequencing of the generated scarless, PTG exemplified by plasmid pUU107. Highlighted in red are the overlap regions.

Prime Editing the design of an extended sgRNA required by the Murine Leukaemia Virus (MLV) retrotranscriptase fused to nCAS9 is also not trivial.

Boosting multiplexed Prime Editing requires the automation of primer design for assembly. We designed and implemented a python engine for automated generation of primers for multiplexed Prime Editing. We use python3, Biopython, iBioCAD [3] and packed them in a docker image for development of a Webapp using Flask assuring portability. We hope that PolyGEN will lower the burden of associated with designing PTGs for Prime Editing. Our tool provides the foundations for automating synthesis of CRISPR tools that will help in the design of complex phenotypic outputs generated by foreseeable CRISPR applications.

3 ALGORITHM

The proposed algorithm requires a user to provide the desired RNAs for the PTG together with their type (sgRNA, pegRNA, smRNA etc.) and two BsaI restriction sites from a plasmid, which will take up the generated PTG, as inputs. Precomputed sets of BsaI 4 bp overlaps, with experimentally validated optimized efficiency for correct assembly [3], were adopted for the algorithm. Compatibility of used overlaps is critical for correct assembly, as certain overlap pairs are prone to mismatching and causing incorrect assembly [3, 4]. The sets are increasing in size from 10 to 50 overlaps to enable one-pot assemblies of varying complexity. For designing the PTG, the smallest possible set with overlaps in all necessary linker regions is selected. The linker regions are the variable regions within each custom RNA (e.g. the spacer in a sgRNA). These overlaps will then become the borders of the final parts used in Golden Gate assembly, effectively shifting the ends of each part to facilitate optimal assembly (Fig. 1A,D). Using the new ends and desired melting temperatures, the algorithm will compute primers for producing the newly designed parts from a plasmid containing a tRNA-gRNA template via PCR. Since the primers will be designed with a BsaI recognition site and optimal overlap, the PCR products (Fig. 1C) will then also include these features and can therefore be used in a subsequent Golden Gate assembly reaction (Fig. 1A). The computed primers together with the generated PTG are put out by the algorithm to be visualized and verified in arbitrary further software. In this way, PolyGEN combines several existing ideas on assembly and design automation [3, 5] to create a tool for simple and user-friendly work with polycistrons. In addition, by working with existing gRNA-tRNA templates, this workflow allows a quick and cheap production of parts which only requires ordering short primer sequences to include each part's variable sequence. PolyGEN largely extends iBioCAD for generating multiplexed RNA production that can be used in the context of CRISPR, Prime Editing or microRNA synthetic biology.

4 EXPERIMENTAL VALIDATION

We experimentally tested the PolyGEN tool by automatically generating primers for a three-parts PTG that contains a sgRNA targeting Boyle's protospacer [2]. We created three different plasmids that carry the protospacer in position 1 (pUU105), 2 (pUU106) and 3 (pUU107). The oligos were synthesized by a commercial provider and pUU017 was used as template for generating each PTG fragment (Fig. 1C). After amplification the fragments were cleaned with a PCR purification kit. Then a BsaI assembly was performed using pUU086 as acceptor vector as described by New England Biolabs protocol. The reactions were transformed in *E. coli* TOP10 and a candidate for each PTG version was sequenced by Sanger's method (Microsynth, Germany). The sequencing results showed that two out of three PTGs were assembled successfully without mutations in the overhangs (Fig. 1D). We sent a second candidate for the failed assembly with successful results.

5 DISCUSSION

Our experimental validation provides evidence for the power of PolyGEN with an estimated assembly time of 4 days from primer arrival to sequencing results and minimal experimental complexity. The clean amplification of the PCR products provides evidence for further lab automation which is currently under investigation in our research group.

6 DATA AND CODE AVAILABILITY

PolyGEN git repository: https://github.com/jurquiza/polygen. We deposited the sequences for pUU017, pUU086, pUU105, pUU106 and pUU107 in https://public-registry.jbei.org/folders/598.

Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germanys Excellence Strategy – EXC-2048/1 – project ID 390686111

REFERENCES

[1] Anzalone, Randolph, D. S. K. e. a. *Search-and-replace genome editing without double-strand breaks or donor DNA.* Nature, 2019.

[2] Boyle, Andreasson, C. S. W. G. D. G. High-throughput biochemical profiling reveals sequence determinants of dcas9 off-target binding and unbinding.

[3] HamediRad, Weisberg, C. L. Z. *Highly Efficient Single-Pot Scarless Golden Gate Assembly.* ACS Synthetic Biology, 2019.

[4] Potapov, Ong, K. L. B. e. a. *Comprehensive Profiling of Four Base Overhang Ligation Fidelity by T4 DNA Ligase and Application to DNA Assembly.* ACS Synthetic Biology, 2018.

[5] Xie, Minkenberg, Y. *Boosting CRISPR/Cas9 multiplex editing capability with the endogenous tRNA-processing system.* PNAS, 2015.

Detecting Co-Occurring Signatures of Engineering in Single Cells with Targeted Sequencing

Aaron Adler[1], Adam Abate[2], Brian Basnight[1], Joseph H. Collins[3], Ben Demaree[2], Kevin W. Keating[3], Xiangpeng Li[2], Tyler Marshal[1], Thomas Mitchell[1], David Ruff[4], Allison Taggart[1], Shu Wang[4], Daniel Weisgerber[2], Fusun Yaman[1], Eric M. Young[3], and Nicholas Roehner[1]

[1]Raytheon BBN Technologies, [2]University of California San Francisco, [3]Worcester Polytechnic Institute, [4]Mission Bio

nicholas.roehner@raytheon.com

1 INTRODUCTION

As engineered organisms such as yeasts and other fungi are increasingly used to combat pests[2] and manufacture chemicals at an industrial scale, the risk of their accidental release or nefarious use increases as well. Given that real-world samples contain many different cell types and that engineered microorganisms can be diluted and effectively masked by their metagenomic context, there is an increasing need for automated methods to detect DNA engineering at the level of individual cells. Even when bulk sequencing methods can be used to detect rare sequences, they are still unable to determine with confidence whether two sequences came from the same cell, which is often a necessary condition for detecting DNA engineering. Lastly, while single-cell sequencing datasets permit this degree of resolution, they are also highly multi-dimensional and can present a significant challenge for a human analyst to evaluate.

To address the need for single-cell sequencing to detect DNA engineering, we have developed a targeted sequencing pipeline for yeast as part of the Guard for Uncovering Accidental Release, Detecting Intentional Alterations, and Nefariousness (GUARDIAN) project. The pipeline leverages Mission Bio's Tapestri, a microfluidic instrument that enables users to encapsulate cells in droplets and mix them with reagents for purposes such as cell lysis, target amplification, and DNA barcoding. In addition, we have developed a single-cell sequencing analysis pipeline that produces customized visualizations to summarize and de-dimensionalize targeted single-cell sequencing data to permit their analysis.

2 GUARDIAN SINGLE-CELL SEQ. PIPELINE

As shown in Figure 1, the GUARDIAN single-cell sequencing pipeline is comprised of five main stages, beginning with single-cell library prep using Mission Bio's Tapestri instrument and library amplification using a panel of 100 amplicons designed to detect engineered yeast (in particular, *S. cerevisiae*, *Y. lipolytica*, and *P. pastoris*). Next, the single-cell libraries are sequenced in bulk using typical next-generation protocols, and the resulting data are processed bioinformatically to map reads to amplicons and group them by cell barcodes. Finally, the processed single-cell sequencing data

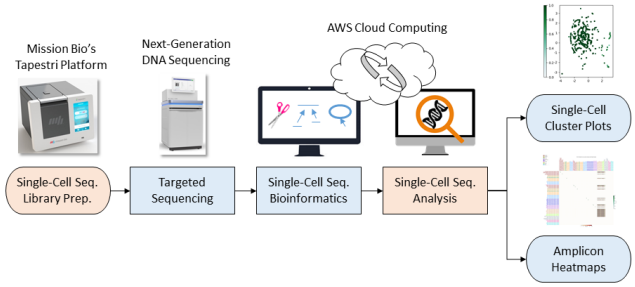


Figure 1: GUARDIAN single-cell sequencing pipeline.

are analyzed and visualized using using amplicon heatmaps and cell cluster plots.

Figure 2 is an example of the amplicon heatmaps generated by the analysis portion of the pipeline. In evaluating these heatmaps to determine whether a sample is engineered, we generally look for co-occurrence of (1) reads for amplicons targeting natural sequence features for a particular yeast species with (2) amplicons targeting non-natural features of engineering for said species. For the heatmap in Figure 2, we conclude that it most likely represents a sample of the yeast knockout strain BY4742 based on the following observations: first, most of the amplicons for *S. cerevisiae* features are present in >95% of cells and co-occur with each other (see Box A). In addition, reads are present for amplicons targeting the expected knockout junctions in BY4742 for the auxotrophic markers URA3 and LEU2, and reads are absent for amplicons targeting the LYS2 marker sequence. Second, we conclude that this sample likely has a small subpopulation of engineered cells (0.4%) containing a plasmid with the Pagtef1 promoter and Tagtef1 terminator, since amplicons for these sequence features co-occur with those for *S. cerevisiae* and plasmid features such as the F1 origin and LacZ reporter (see Box B). The percent of cells in which these amplicons occur also exemplifies the typical limit of detection for our pipeline: 1 in 500 engineered cells following analysis of less than 10,000 cells. Lastly, there are also reads for *Y. lipolytica* amplicons present (see Box C), but they occur in only 0.1% of cells and most importantly do not co-occur

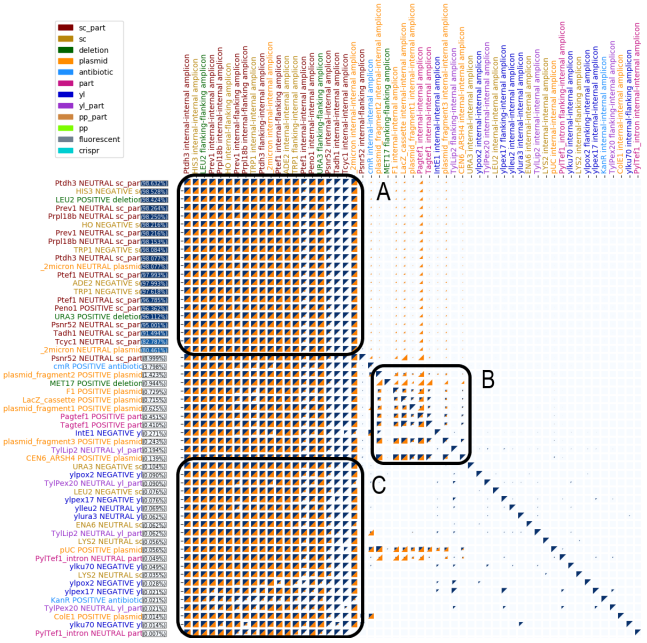


Figure 2: Amplicon heatmap for sample of BY4742 containing 1 in 500 engineered cells. Blue triangles quantify by size the co-occurrence of the column amplicon in cells containing the row amplicon (0% to 100%). Orange triangles quantify by size the average number of reads for the column amplicon in cells containing the row amplicon (0 to max threshold of 100 reads). Row and column amplicons are identical and are ordered by the overall percent of cells in which they occur. NEUTRAL amplicons are those primarily used to genotype a sample, while POSITIVE and NEGATIVE amplicons are used to call sample engineered based on the presence and absence of their reads, respectively. The internal-internal, internal-flanking, and flanking-flanking labels on the column amplicons refer to whether their primers occur within or adjacent to the sequence features that they detect.

with each other, indicating that they are likely the result of non-specific amplification.

3 RESULTS AND DISCUSSION

We tested the GUARDIAN pipeline on 50 samples (tubes of lab-grown cells) in total, including 12 samples of *S. cerevisiae* (6 engineered, 6 not), 31 samples of *Y. lipolytica* (19 engineered, 12 not), and 7 samples of *P. pastoris* (1 engineered, 6 not). As shown in Figure 3, human-in-the-loop analysis of amplicon heatmaps generated with the pipeline yielded an overall sensitivity of 0.88 and specificity of 0.92. The best performance was achieved for *P. pastoris*, followed by *Y. lipolytica* and *S. cerevisiae*. The differences in sensitivity between these organisms are largely explained by differences in the types of engineering features found in each sample. In particular, *S. cerevisiae* was the only organism tested in this

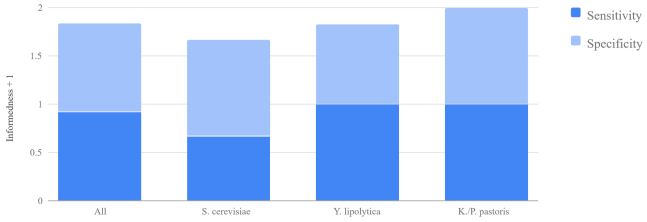


Figure 3: Sensitivity plus specificity for the GUARDIAN single-cell pipeline when applied to three target yeast organisms (overall and by organism). Sensitivity plus specificity equal to 2 or 0 indicates perfectly informed or misinformed performance, respectively, while a sum equal to 1 indicates performance no better than random.

batch that had some samples containing small nucleotide edits and no other signature of engineering, making them more difficult to call engineered (and impossible to call using amplicon heatmaps alone). Moving forward, we plan to incorporate variant calling into our analysis pipeline to enable detection of smaller edits. We also plan to automate some analysis of the amplicon heatmaps and other visualizations to better highlight portions that are indicative of engineering or that are possibly the result of experimental defects such as non-specific amplification. Finally, we are also working to apply Tapestri to single-cell whole-genome sequencing (WGS)[1] to enable application of a separate WGS analysis pipeline that is being developed as part of the GUARDIAN project.

ACKNOWLEDGMENTS

We thank Pacific Northwest National Laboratory and Lawrence Berkeley National Laboratory for preparing the samples with which we tested GUARDIAN. This research is based upon work supported [in part] by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA) under Finding Engineering Linked Indicators (FELIX) program contract #N6600118C-4507. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

REFERENCES

[1] Lan, F., et al. Single-cell genome sequencing at ultra-high-throughput with microfluidic droplet barcoding. *Nature Biotechnology 35* (2017), 640–646.

[2] Lovett, B., and Leger, R. J. S. Genetically engineering better fungal biopesticides. *Pest Management Science 74* (2017).

Active Learning for Efficient Microfluidic Design Automation

David McIntyre
dpmc@bu.edu
Department of Biomedical Engineering
Boston University
Boston, MA

Ali Lashkaripour
lashkari@bu.edu
Department of Biomedical Engineering
Boston University
Boston, MA

Douglas Densmore
dougd@bu.edu
Department of Computer and Electrical Engineering
Boston University
Boston, MA

1 INTRODUCTION

Droplet microfluidics has the potential to eliminate the testing bottleneck in synthetic biology by screening biological samples encapsulated in water-in-oil emulsions at unprecedented throughput [2]. Sophisticated screens require functional and complex devices that perform exactly as designed. Effective performance characterization and predictive design of droplet microfluidic components has been hampered due to low-throughput and expensive fabrication with standard soft lithography techniques. This has limited droplet microfluidics to proof-of-concept devices. Even when some of these barriers are removed through rapid prototyping, developing a robust dataset to effectively represent all parameters as a "lookup table" is near impossible.

One solution to explore how design parameters affect performance in microfluidics is through machine learning. Although machine learning can make accurate microfluidic design automation tools, standard development pipelines require a large, naively-generated training set **(Figure 1, left)**. These approaches become intractable in cases where generating labeled data is particularly time or money-intensive.

Training data-restricted models can benefit from active learning algorithms, in which the model queries an "oracle" (the user) during the training process to only generate or label the data it predicts would best improve model performance **(Figure 1, right)** [6]. Through structured data generation, the amount of training data needed for an accurate model can be significantly reduced, speeding up the time to predictive design and eliminating unproductive user efforts.

Here, we present a novel experimental paradigm to rapidly generate microfluidic design automation tools. Efficacy of this method was tested against a previously generated dataset for a droplet generator design tool (DAFD) [3, 4]. This method can be extended to additional microfluidic components or fabrication methods, provided a method for data generation is high-throughput enough.

2 RESULTS

Efficient data generation for active learning algorithms necessitates evaluation of the quality of unlabeled data (informativeness and/or diversity) used in each round of model training [6]. Informativeness is the predicted amount that a specific datapoint can improve the model, whereas diversity is the spread of the data used across the design space. Here, previously generated data is pooled as "chips" (i.e., all datapoints generated using the same microfluidic device), which includes $\tilde{1}000$ datapoints pooled as 43 chips that were fabricated with previously developed rapid prototyping workflows [5]. Data was pooled in this way to minimized future microfluidic devices that need to be made, the most resource-intensive step in the data generation process.

To initially explore the advantages of active learning, three data quality metrics were implemented: (1) random choice; (2) greedy sampling (GS), which chooses the most different chip to the training set [7]; and (3) query by committee (QBC), which chooses the most informative chip [1]. In all cases, the model is seeded with one chip randomly picked from the training set.

In greedy sampling, optimal candidates are chosen by the maximum average distance of the geometric features of the chip from the existing labeled training set **(Equation 1)**. All features of each datapoint is normalized to avoid bias.

$$d_{dp} = \frac{1}{N_{train}} \sum_{i=1}^{N_{train}} ||\mathbf{x_{dp}} - \mathbf{x_i}|| \tag{1}$$

Alternatively, the potential "information" gained through adding a specific datapoint can be evaluated with QBC **(Equation 2)**.

$$I(x) = \frac{1}{P} \sum_{p=1}^{P} \frac{(\hat{y}_p(x) - \bar{y}(x))^2}{\bar{y}(x)} \tag{2}$$

In QBC, the quality of each unlabeled point is evaluated by the variance of each prediction across P regressors. Each regressor is trained on a bootstrapped collection of the training set. Points with high information are estimate to be those with a large variance in predicted value. In this study, results were normalized by the mean prediction to avoid bias for larger values. Each iteration, the chip with the max average variance was chosen as the next datapoint.

These methods were implemented into the DAFD framework, consisting of 4 neural networks (NN) predicting the droplet size and generation rate in the dripping and jetting

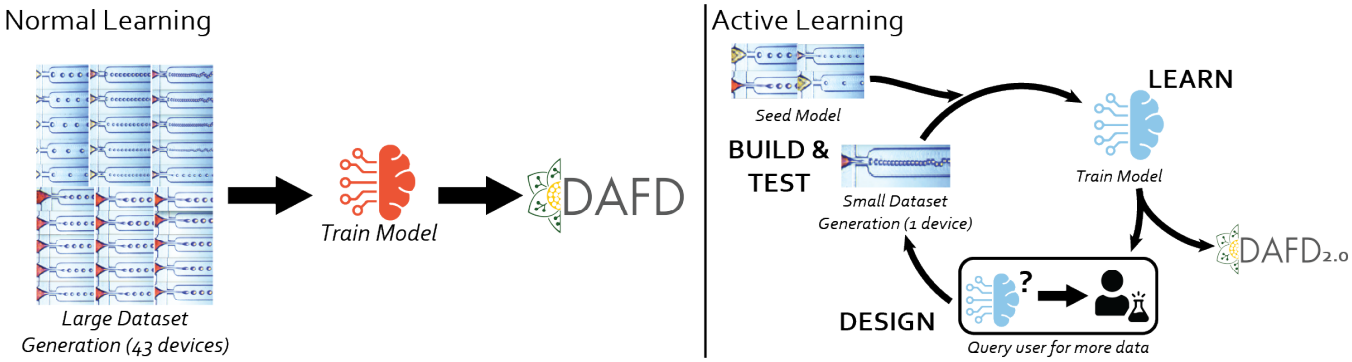


Figure 1: Comparison between a normal machine learning pipeline (left), in which a large training set it fed into the model and active learning (right), in which the model queries a user for more data after "seeding" with a small initial dataset.

regimes **(Figure 2)**. Regressor accuracy was tested on a randomly partitioned 20% of the total dataset and evaluated using root-mean-square error (RMSE). Across all NNs, GS performed better than or equivalent to random choice. This was distinct in regime 2: an RMSE of 0.9 was achieved with 100 and 150 fewer datapoints for size and generation rate, respectively. This indicates that diversity of data is the most important characteristic of the training set. QBC had improved performance than random choice in some cases, however, performed worse when predicting droplet size in regime 2. Poor performance by QBC could be from poor initialization or balancing data selection over the 4 regressors.

3 CONCLUSION & FUTURE WORK

Here, we have shown that active learning can provide a design framework to streamline the experimental workflow for developing design automation tools. While model improvement was variable while simultaneously training four regressors, this framework can be improved through development of a more sophisticated algorithm accounting for both diversity and informativeness. Model seeding could also be improved through formal Design of Experiments (DoE), giving a high-quality base model for further data generation and model evaluation cycles.

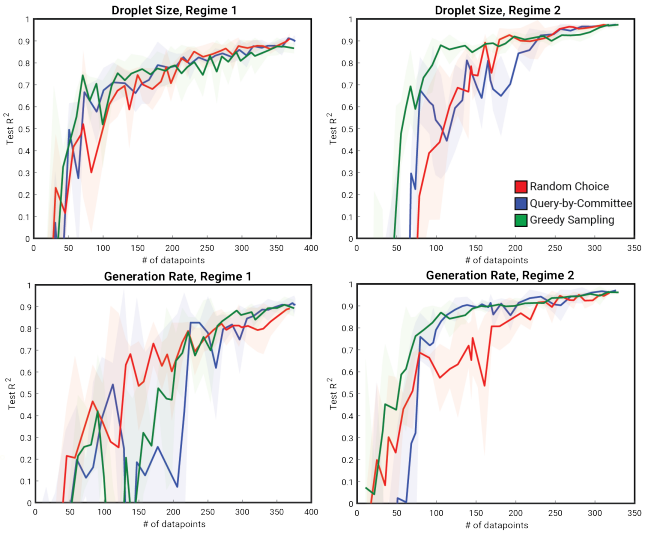


Figure 2: RMSE error across all four regressors with different active learning algorithms. Curves and shaded regions are the mean and standard deviation, respectively (N=3)

While this first study has used an existing dataset exploring how microfluidic device parameters affect droplet generation, we can extend this approach to *de-novo* models of different components (droplet sorter, merger, etc.). This method can also be used to rapidly perform transfer learning for using a device with custom fluid classes or different fabrication methods. Development of a streamlined pipeline for design automation is a necessary step for the standardization of microfluidics, and further spread its adoption by non-experts.

REFERENCES

[1] Burbidge, R., Rowland, J. J., and King, R. D. Active Learning for Regression based on Query by Committee. Tech. rep.

[2] Guo, M. T., Rotem, A., Heyman, J. A., and Weitz, D. A. Droplet microfluidics for high-throughput biological assays. *Lab on a Chip 12*, 12 (may 2012), 2146.

[3] Lashkaripour, A., Rodriguez, C., Mehdipour, N., McIntyre, D., and Densmore, D. Modular microfluidic design automation using machine learning. 11th International Workshop on Bio-Design Automation (IWBDA-19).

[4] Lashkaripour, A., Rodriguez, C., Ortiz, L., and Densmore, D. Performance tuning of microfluidic flow-focusing droplet generators. *Lab on a Chip 19*, 6 (2019), 1041–1053.

[5] Lashkaripour, A., Silva, R., and Densmore, D. Desktop micromilled microfluidics. *Microfluidics and Nanofluidics 22*, 3 (mar 2018), 31.

[6] Wu, D. Pool-Based Sequential Active Learning for Regression. Tech. rep., 2018.

[7] Yu, H., and Kim, S. Passive sampling for regression. In *2010 IEEE International Conference on Data Mining* (2010), IEEE, pp. 1151–1156.

Efficient Large-Scale Microfluidic Design-Space Exploration: From Data to Model to Data

Ali Lashkaripour
Biomedical Engineering Department, Boston University
lashkari@bu.edu

David McIntyre
Biomedical Engineering Department, Boston University
dpmc@bu.edu

Douglas Densmore*
Department of Electrical and Computer Engineering, Boston University
dougd@bu.edu

1 INTRODUCTION

Droplet microfluidics is well poised to improve the gold standard in many fields such as synthetic biology [2]. However, the lack of available design automation tools that can create a microfluidic droplet generator based on a desired performance, forces the design process to be iterative, inefficient, and costly, thus, hampering the wide-spread adoption of droplet microfluidics in the life sciences. Machine learning and design automation tools have advanced many fields with new capabilities, such as genetic circuit design, cell pattern synthesis, and multi-cellular mass formation design [1, 8, 9]. The recent introduction of low-cost rapid microfabrication techniques enables generation of large-scale experimental datasets which was previously not viable in a realistic cost and time-frame [7]. We previously developed an open-source machine learning based design automation tool, DAFD (dafdcad.org) [5], which can utilize the available data to provide both performance prediction and design automation. However, to achieve accurate performance prediction and design automation a full-factorial search in flow conditions (25) and an orthogonal design of experiments for geometry search (25 devices) were used resulting in a total 625 experiments. By analyzing the the contribution of each device to the exploration of the design-space, we identify a more efficient method to map approximately the same design-space with just 5 chips. Therefore, by utilizing a low-cost fabrication method droplet generation design-space was explored, analyzed, and understood, which in turn enables design automation of high-performance and high-end droplet generators in a viable and realistic cost-frame.

2 CONFIDENCE ELLIPSES

The observed performance of a microfluidic droplet generator can be summed up in two parameters: droplet diameter and generation rate. Since, all the 25 microfluidic devices were tested at the same 25 flow conditions, the devices that show a larger variation in the observed performance induced by changing the flow condition, is more efficient in exploring the design-space. A confidence ellipse can be drawn by using the variance of the 25 observed data per device in both directions (droplet diameter and generation rate) as given in Eq. (1):

$$\left(\frac{x}{\sigma_F}\right)^2 + \left(\frac{y}{\sigma_D}\right)^2 = s \tag{1}$$

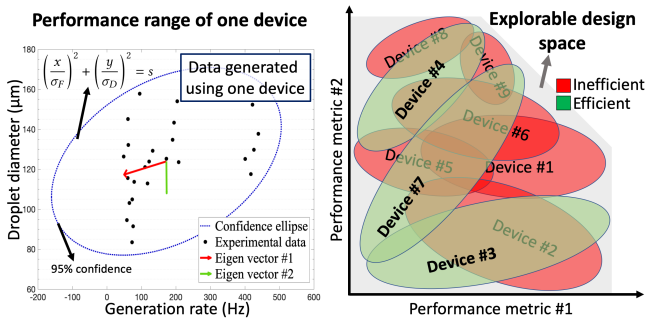


Figure 1: The performance range of a given microfluidic droplet generator design can be estimated by confidence ellipses. Using a dataset generated through a low-cost rapid prototyping method and a low-cost fluid combination, performance range of each design and the amount of performance overlap are approximated. This analysis reduces the necessary number of designs that should be fabricated and tested for extending design automation tools to cover new high-end fluid combinations and fabrication techniques.

where σ_F is the variance in generation rate, σ_D is the variance in droplet diameter, and s defines the size of the ellipse (confidence value). Since droplet diameter and generation rate are independent and by assuming a Gaussian distribution, Eq. (1) becomes a Chi-Square distribution [4], therefore, for a 95% confidence, $s = 5.991$. Using the covariance matrix of the data of each device, the eigenvectors are calculated to determine the angle that the ellipse takes.

3 EFFICIENT DESIGN-SPACE EXPLORATION

The workflow consists of three phases. **Phase 1** starts with cost and time-efficient exploration of the entire design-space using low-cost desktop micromilling to generate the initial large-scale dataset as we previously reported [6]. In **phase 2**, machine learning models are fitted to the data and using metrics such as coefficient of determination the accuracy of the predictive models and the sufficiency and diversity of the dataset are verified. Afterwards, iso-contours of the Gaussian distribution (confidence ellipses) [3] for the data points generated with a single device are used to determine the contribution of each device in exploring the design-space. The devices with a confidence ellipse that shows a lot of overlap with other confidence ellipses signifies an inefficient exploration. On the other hand, the devices with a confidence

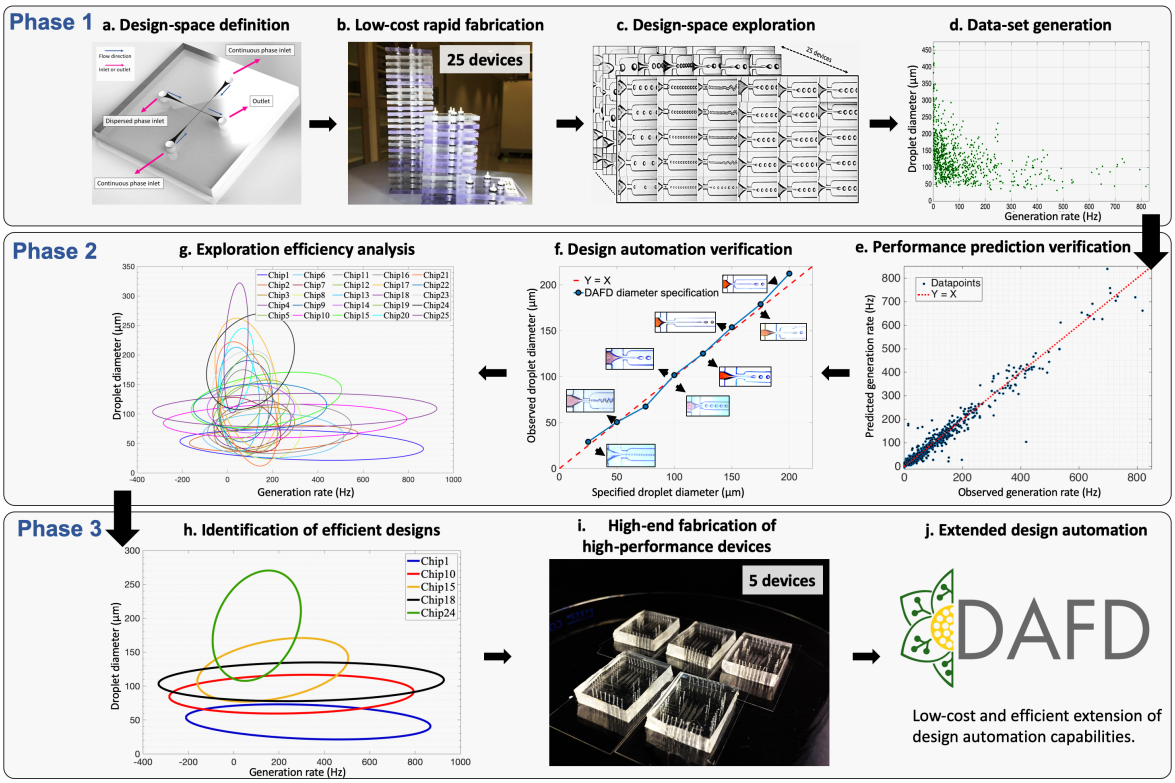


Figure 2: Phase 1: Low-cost rapid prototyping and design of experiments methods are used to generate a large-scale dataset. Phase 2: Machine learning based performance prediction and design automation accuracy and data sufficiency are verified. The dataset is analyzed to find a reduced number of devices that cover a similar design-space. Phase 3: The identified designs can be used for extending the capabilities of design automation tools to support high-end fluids and fabrication methods.

ellipse that has a minimum overlap and encompasses a larger design-space demonstrates a more efficient search. Therefore, in **phase 3**, we can remove the inefficient devices and determine the minimum number of devices that can be used to explore the design-space. Consequently, by significantly reducing the number of microfluidic devices necessary to explore a design-space, high-end fabrication methods and fluid combinations could be used to extend the design automation tool to support high-performance microfluidics in a time- and cost-efficient manner.

4 CONCLUSION AND FUTURE WORK

Machine learning algorithms enable accurate microfluidic design automation. However, generating large-scale datasets required for training these algorithms are resource intensive. Therefore, efficient frameworks are required for extending design automation tools. In here, we used information inferred from a dataset generated using low-cost material to efficient create a dataset for high-end material.

REFERENCES

[1] Appleton, E., Mehdipour, N., Daifuku, T., Briers, D., Haghighi, I., Moret, M., Chao, G., Wannier, T., Chiappino-Pepe, A., Belta, C., et al. Genetic design automation for autonomous formation of multicellular shapes from a single cell progenitor. *bioRxiv* (2019), 807107.

[2] Ferry, M. S., Razinkov, I. A., and Hasty, J. Microfluidics for synthetic biology: from design to execution. In *Methods in enzymology*, vol. 497. Elsevier, 2011, pp. 295–372.

[3] Ferstl, F., Kanzler, M., Rautenhaus, M., and Westermann, R. Visual analysis of spatial variability and global correlations in ensembles of iso-contours. In *Computer Graphics Forum* (2016), vol. 35, Wiley Online Library, pp. 221–230.

[4] Lancaster, H. O., and Seneta, E. Chi-square distribution. *Encyclopedia of biostatistics 2* (2005).

[5] Lashkaripour, A., Rodriguez, C., Mehdipour, N., McIntyre, D., and Densmore, D. Modular microfluidic design automation using machine learning. 11th International Workshop on Bio-Design Automation (IWBDA-19).

[6] Lashkaripour, A., Rodriguez, C., Ortiz, L., and Densmore, D. Performance tuning of microfluidic flow-focusing droplet generators. *Lab on a Chip 19*, 6 (2019), 1041–1053.

[7] Lashkaripour, A., Silva, R., and Densmore, D. Desktop micromilled microfluidics. *Microfluidics and Nanofluidics 22*, 3 (2018), 31.

[8] Mehdipour, N., Briers, D., Haghighi, I., Glen, C. M., Kemp, M. L., and Belta, C. Spatial-temporal pattern synthesis in a network of locally interacting cells. In *2018 IEEE Conference on Decision and Control (CDC)* (2018), IEEE, pp. 3516–3521.

[9] Nielsen, A. A., Der, B. S., Shin, J., Vaidyanathan, P., Paralanov, V., Strychalski, E. A., Ross, D., Densmore, D., and Voigt, C. A. Genetic circuit design automation. *Science 352*, 6281 (2016), aac7341.

A Droplet-Based Microfluidic Lab Automation for Biosynthetic Pathway Optimization

Kosuke Iwai*
KosukeIwai@lbl.gov
Joint BioEnergy Institute
Emeryville, CA, USA
Sandia National Laboratories
Livermore, CA, USA

Megan Garber
Joint BioEnergy Institute
Emeryville, CA, USA
Lawrence Berkeley National Laboratory
Berkeley, CA, USA
megarber@lbl.gov

Jess Sustarich
Joint BioEnergy Institute
Emeryville, CA, USA
Sandia National Laboratories
Livermore, CA, USA
JSustarich@lbl.gov

Peter W. Kim
Joint BioEnergy Institute
Emeryville, CA, USA
Sandia National Laboratories
Livermore, CA, USA
Lawrence Berkeley National Laboratory
Berkeley, CA, USA
pkim@lbl.gov

William R. Gaillard
Joint BioEnergy Institute
Emeryville, CA, USA
Sandia National Laboratories
Livermore, CA, USA
RGaillard@lbl.gov

Kai Deng
Joint BioEnergy Institute
Emeryville, CA, USA
Sandia National Laboratories
Livermore, CA, USA
kdeng@lbl.gov

Trent Northen
Joint BioEnergy Institute
Emeryville, CA, USA
Lawrence Berkeley National Laboratory
Berkeley, CA, USA
trnorthen@lbl.gov

Hector Garcia-Martin
Joint BioEnergy Institute
Emeryville, CA, USA
Lawrence Berkeley National Laboratory
Berkeley, CA, USA
hgmartin@lbl.gov

Paul D. Adams
Joint BioEnergy Institute
Emeryville, CA, USA
Lawrence Berkeley National Laboratory
Berkeley, CA, USA
pdadams@lbl.gov

Anup K. Singh*
Joint BioEnergy Institute
Emeryville, CA, USA
Sandia National Laboratories
Livermore, CA, USA
aksingh@sandia.gov

1 INTRODUCTION

In recent years, synthetic biology has drawn significant interest for both scientific research and industrial applications such as biofuel and pharmaceutical production. One good example is indigoidine, bacterial natural dye with antioxidant and antimicrobial activities [4]. Synthetic biology process, however, requires multiple iterations of Design-Build-Test-Learn (DBTL) cycles for optimal production of target biomolecules, which is time-consuming and labor-intensive due to low availability of advanced tools and high-throughput workflows. Here we propose a versatile and robust droplet-based microfluidic platform that enables high-throughput iterations of DBTL cycles (Figure 1a). The heart of our system is a digital microfluidic (DMF) chip that enables reactions in parallel using nL droplets as reaction vessels as described in our earlier publications [1]. In addition to DMF manipulations for mixing and transporting droplets containing biological parts, proposed platform with 100 discrete chambers is capable of parallel electroporation with different conditions, and additional reservoirs allow recovery incubation and screening on chip.

*This work was part of the DOE Joint BioEnergy Institute (http://www.jbei.org) supported by the U. S. Department of Energy, Office of Science, Office of Biological and Environmental Research, through contract DE-AC02-05CH11231 between Lawrence Berkeley National Laboratory and the U. S. Department of Energy.

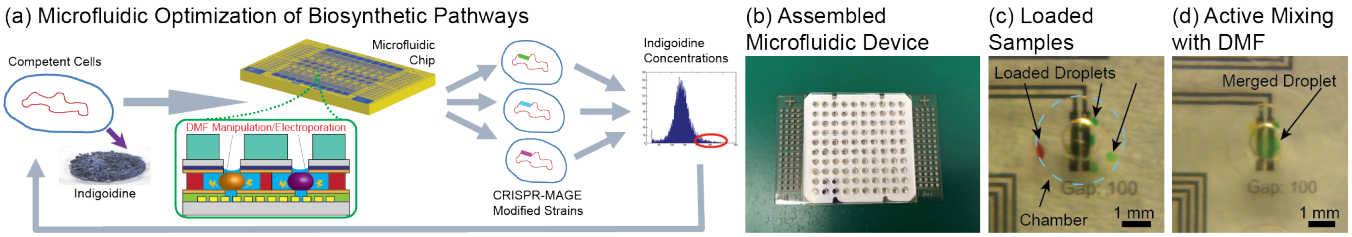


Figure 1: Microfluidic optimization of biosynthetic pathways. (a) concept of the workflows. (b) Fabricated microfluidic devices with 100 reaction chambers in 384 well format. (c) Multiple samples loaded on chip. (d) Demonstration of active mixing of samples in droplets with DMF.

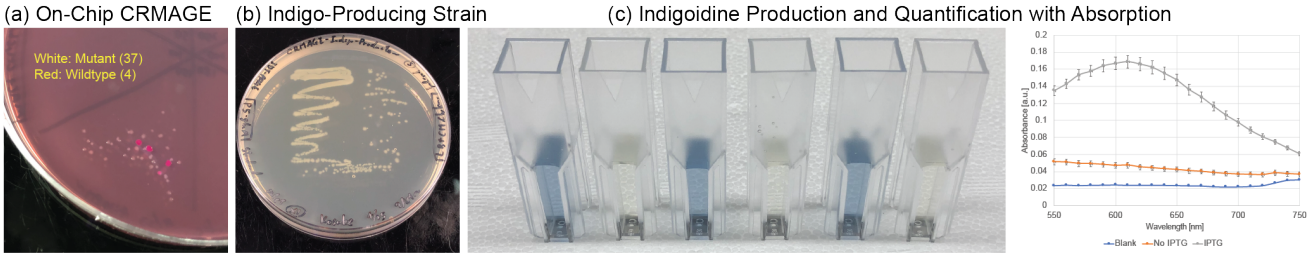


Figure 2: Experimental results of on-chip transformation and indigoidine production. (a) CRMAGE results to knockout GalK. On-chip CRMAGE achieved over 90% of mutation. (b) Modified indigo-producing strain. (c) Successful indigoidine production quantified at OD612. Error bars denote standard deviation of biological triplicate.

2 EXPERIMENTAL

Assembled microfluidic chip is shown in Figure 1b, and its 384-well based configuration is compatible with state-of-the-art liquid handling robots and eliminates the complex processes of formation and loading of droplets to the chip. Loaded samples in droplets are kept inside the oil preventing the evaporations (Fig. 1c), and DMF manipulation enables active mixing and electroporation on-chip (Fig. 1d). In addition, our microfluidic chip is manufacturable with commonly used processes, which is suitable for cost-competitive target molecules such as indigoidine.

3 RESULTS AND DISCUSSION

We adapt CRISPR-based MAGE (CRMAGE) recombineering with our platform for efficient gene-editing processes [3], first targeting an enzyme galactokinase (*galK*). *E. coli* colonies on MacConkey plates with red color indicate the wildtypes and white colonies indicate the mutants with *galK* knockout (Fig. 2a), confirmed by sequencing the genome. With optimized assay protocols (e.g., concentrations of gRNA and Cas9 inducer, anhydrotetracycline), wildtype killing rate achieved over 90% with on-chip transformation.

We then targeted glutamine synthetase (*glnA*) and blue-pigment synthetase (*SFP/bpsA*) enzyme related with indigoidine production (Fig. 2b). For the initial round, we designed oligos and gRNA sequence for CRMAGE plasmids targeting T7 promoters [2]. These mutations were screened by antibiotics (kanamycin), and indigoidine production rate with IPTG induction was quantified by measuring absorption spectra at 612nm (Fig. 2c). Results clearly indicates the successful production of blue pigments. After quantification, CRMAGE plasmid can be self- eliminated by inducing rhamnose for the next round targeting different loci.

4 CONCLUSION

We have developed a droplet-based microfluidic pathway optimization system, which successfully demonstrated on-chip CRISPR-based gene editing and quantification of indigoidine production rate. With the results shown above, we believe our fully-automated system with capable of 100 reactions at a time would dramatically accelerate the DBTL cycle of biosynthetic pathways for emerging synthetic biology applications.

REFERENCES

[1] Gach, P. C., Iwai, K., Kim, P. W., and Singh, A. K. Droplet microfluidics for synthetic biology. *Lab Chip* (2017).
[2] Komura, R., Aoki, W., Motone, K., Satomura, A., and Ueda, M. High-throughput evaluation of t7 promoter variants using biased randomization and dna barcoding. *PLoS One* (2018).
[3] Ronda, C., Pedersen, L., Sommer, M., and Nielsen, A. Crmage: Crispr optimized mage recombineering. *Sci. Rep.* (2016).
[4] Xu, F., Gage, D., and Zhan, J. Efficient production of indigoidine in escherichia coli. *J. Ind. Microbiol. Biotechnol.* (2015).

Intent Parser: a tool for codifying experiment design

Tramy Nguyen[1], **Nicholas Walczak**[1], **Jacob Beal**[1], **Daniel Sumorok**[1], **Mark Weston**[2*]
[1]Raytheon BBN Technologies, [2]Netrias, Cambridge MA
tramy.t.nguyen@raytheon.com,nicholas.walczak@raytheon.com

1 INTRODUCTION

Many biological experiments are described in text documents, such as laboratory notebooks, capturing information such as the purpose, execution, and results of an experiment. In such descriptions, however, authors typically present information in a highly personal and idiosyncratic manner, at varying levels of detail and often omitting critical information. Consequently, this lack of consistency lead to a variety of issues commonly encountered when attempting to compare experimental reports created by different authors or to build upon those results in new work. Humans can sometimes infer sufficient information to interpret such informal documentation of experiment designs, but this is typically an ad-hoc, challenging, and error-prone process, not particularly susceptible to automation. At the same time, precise and unambiguous specifications of both elements and their combinations can be expressed in machine-readable representations such as SBOL [2], but making use of these tools is difficult for many investigators.

A "middle-ground" approach combining both accessibility and representational precision, however, has been known at least as far back as Winograd's SHRDLU system [4], using machine feedback and prompting to shape human input into a semi-structured form that can be readily interpreted and checked by machines. We have applied this approach to develop Intent Parser, a tool that combines a word-processing interface with structured tables and assisted linking to definitions to provide a simple interface for incremental codification of experiment designs. Use of this tool can help synthetic biology collaborations by reducing the time and skills required to produce precise experiment designs, enabling automatic checking for errors and ambiguities, and simplifying interpretation of experimental data.

2 INTENT PARSER ARCHITECTURE

Fundamentally, the Intent Parser acts as a link between a user-friendly document editor (in this case Google Docs) and a repository of formal definitions (in this case SynBioHub [3]). By linking these and adhering to certain document conventions, the tool allows users to generate unambiguous machine-readable descriptions of experiment design. In our implementation, we have chosen to use Google Docs for the editor, SynBioHub [3] as the repository (in combination with the SBOL Project Dictionary interface [1]), and to output experiment design specifications in JSON.

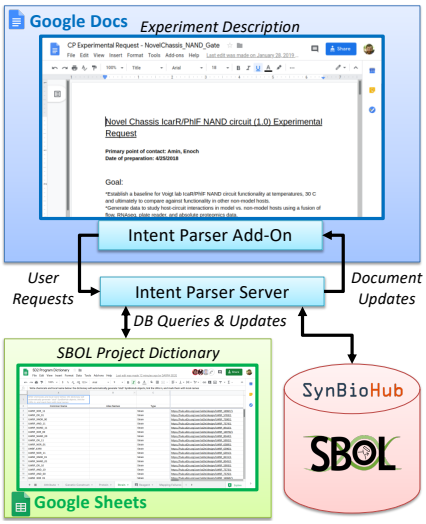


Figure 1: Architecture of Intent Parser: a Google Docs add-on sends user requests to a "back-end" server that interacts with databases of definitions (SynBioHub) and terminology (SBOL Project Dictionary). Results return to the add-on, which offers the user choices and alters the document.

Intent Parser is implemented as an "add-on" to Google Docs using the Google Docs API to implement the architecture shown in Figure 1. Once installed, this add-on provides a file menu of operations that can be performed on any Google Doc. Users conceiving of an experiment write up an experiment description in a Google Doc and invoke requests around two workflows: 1) grounding document text with links to definitions in SynBioHub [3], and 2) defining, validating, and exporting experiment requests that make use of those definitions. These requests can be made at any time, supporting an incremental and collaborative approach to experiment design.

When the user makes a request in the add-on, an HTTP request is sent to the Intent Parser Server, which then parses the document and returns an HTTP response with the result back to the add-on. The server is a backend that carries out requests by interacting with SynBioHub [3], which provides

*Supported by AFRL and DARPA under contract FA875017CO184. This document does not contain technology or technical data controlled under either U.S. International Traffic in Arms Regulation or U.S. Export Administration Regulations. 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 AFRL and DARPA.

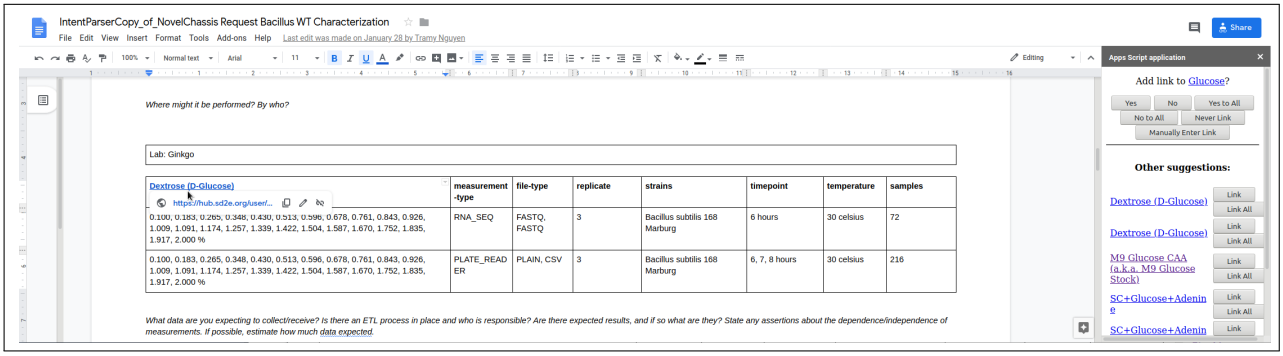


Figure 2: Screenshot of Intent Parser in action on a document from the DARPA SD2 program, showing a measurement table with reagents linked to definitions in SynBioHub [3]. The navigation panel on the right suggests links to add, in this case a link for the term "Glucose" (document location not shown), providing both a best guess and a number of potential alternatives.

information about referenced elements in SBOL format [2], and with the SBOL Project Dictionary [1], which provides a spreadsheet interface that tracks shorthand and lab-specific names. Figure 2 shows an example of linking information found on SynBioHub to names and terminologies on the Google Doc.

Experiment designs are based on tables following a specific format, for which templates can be generated from an add-on menu item. In this abstract, these tables are referred as Intent Parser tables. Validation and export requests are sent to the Intent Parser Server to validate the contents of Intent Parser tables parsed from the Google Doc. Validation follows a predefined data schema that checks for the required information, using SynBioHub and the SBOL Project Dictionary to validate that all terms extracted from the table are properly grounded. From these, the server generates both reports on validity and JSON representations of experiment design.

3 CASE STUDY: DARPA SD2 PROGRAM

The DARPA Synergistic Discovery and Design (SD2) program aims to accelerate scientific discovery by machine-assisted integration of experimental design, build, test, and learning, and is testing these aims via a collaboration of over 100 researchers across over a dozen organizations. In SD2, Intent Parser is used by both data scientists and experimentalists to define and request experiments via Google Docs.

Stakeholders including data scientists, subject matter experts, and experimental labs were consulted to help determine a format that was sufficiently general to specify experiment designs spanning across multiple challenge problems, protocols, laboratories, and experiment designs. The information recorded in these experiment requests includes the name of the lab to execute the experiment, which measurements are to be taken and at what time points, amounts of reagents, strains, and media to be used in each sample, as well as experimental conditions and parameters such as culturing temperature. As users describe the experiment, they also check its validity and required number of samples with Intent Parser. Finally, when when the experiment design is validated and all parties are satisfied, the users can request that the experiment be executed.

The generality of the approach is demonstrated by the breadth of usage in this program: during a four month period, 19 SD2 users from various organizations generated 34 experimental requests in multiple different areas of investigation, resulting in a total of 16,876 experimental samples executed using three different protocols and collecting data from a variety of instruments. Because these experiments are generated systematically with grounded definitions, metadata assignment and analysis has been greatly simplified and accelerated, helping enable faster analysis and more effective sharing of data and analyses across the SD2 program.

4 CONTRIBUTIONS

Intent Parser provides a user-friendly process for describing experiments, grounding narrative design descriptions in links to a SynBioHub repository, and generating and validating specifications for wet-lab experiments. The positive experiences of users in the SD2 program suggest this approach has value, and should continue to be elaborated. Potential future directions include improving integration and UI, increasing scope of descriptions, and using natural language processing to extract additional semantic content from prose.

This tool is actively developed at SD2E's GitHub repository https://github.com/SD2E/experimental-intent-parser.

REFERENCES

[1] Beal, J., et al. Collaborative terminology: SBOL project dictionary. Submitted to IWBDA 2020.

[2] Cox, R. S., et al. Synthetic Biology Open Language (SBOL) version 2.2.0. *Journal of integrative bioinformatics 15*, 1 (2018).

[3] McLaughlin, J. A., et al. SynBioHub: A standards-enabled design repository for synthetic biology. *ACS Synthetic Biology 7*, 2 (2018), 682–688. PMID: 29316788.

[4] Winograd, T. Procedures as a representation for data in a computer program for understanding natural language. Tech. rep., Massachusetts Institute of Technology, Project MAC, 1971.

Collaborative Terminology: SBOL Project Dictionary

Jacob Beal, Daniel Sumorok, Bryan Bartley, Tramy Nguyen*
Raytheon BBN Technologies, Cambridge MA
jakebeal@ieee.org

1 INTRODUCTION

Sharing information about biological experiments between researchers is often challenging. Reagents, strains, and genetic constructs are often given "shorthand" names that are ambiguous (e.g., "ara" for L-arabinose), differ between researchers (e.g., "L-arab" vs. "Arabinose") or are unknown outside of a particular group (e.g., "plasmid 37"). Likewise, the particular combinations used in each sample of an experiment are often expressed in variable personal shorthands, often accidentally omitting important details.

Humans can sometimes infer sufficient information to interpret such informal documentation of experiment designs, but this is typically an ad-hoc, challenging, and error-prone process, not particularly susceptible to automation. At the same time, precise and unambiguous specifications of both elements and their combinations can be expressed in machine-readable representations such as SBOL [2], but making use of these tools is difficult for many researchers. Common machine-readable terminology also typically needs to be agreed upon in advance, which is often onerous or impossible given the ongoing evolution of terms in research projects—indeed, some studying the philosophy of science argue that the ability to define terminology is a good marker for the conclusion of a scientific investigation! [3]

The SBOL Project Dictionary helps bridge this gap with a simple spreadsheet-based interface that allows researchers to collaboratively and incrementally construct a shared terminology. This interface provides a structured format of tabs and columns for researchers to link lab-specific names to shared names, aliases, and canonical definitions using SBOL. Software tooling can then access this set of relations at any time in order to translate metadata terms into an evolving common vocabulary, thereby supporting simple post-hoc integration and debugging across collaborators.

2 ARCHITECTURE

Figure 1 shows the architecture used to implement the SBOL Project Dictionary. This implementation is based on two key

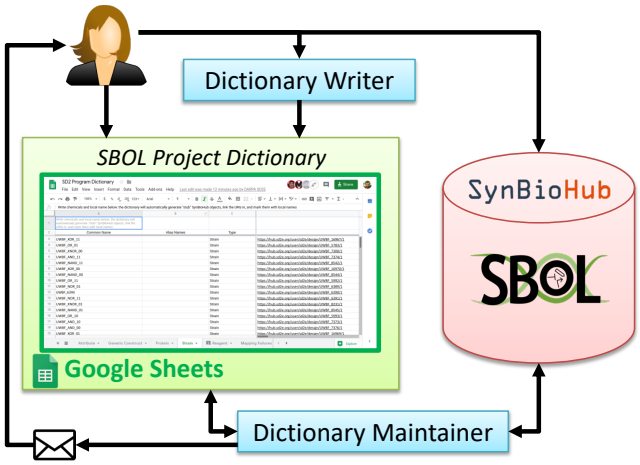


Figure 1: Architecture of SBOL Project Dictionary: the dictionary is held in a Google Sheet, which is maintained and synchronized with canonical storage of names, aliases, and definitions in SynBioHub. Users interact either directly or through tools using a Dictionary Writer API, and receive email notification of integration problems to resolve.

pre-existing tools: Google Sheets, which provides a collaborative spreadsheet editing interface and API for software tools, and SynBioHub [4], a database server for sharing synthetic biology information encoded in SBOL [2]. These are linked by configuring a Dictionary Maintainer service to connect to a particular Google Sheet and SBOL collection in SynBioHub. Periodically, the Dictionary Maintainer updates and synchronizes. First, it ensures the Google Sheet is configured to follow a specified format, including protecting regions that should not be user-editable. It then validates all of the information in the SBOL Project Dictionary and synchronizes with SynBioHub, creating SBOL objects as needed to store new dictionary entries, importing SBOL links when matched by a new entry in the dictionary, and reporting errors to the user via a status column in the spreadsheet.

In particular, the SBOL Project Dictionary provides and maintains spreadsheet columns for the following:

- A single common name, which is the researchers' current consensus on a human-friendly term for each entry, e.g., "Synthetic Complete Media"
- Aliases, which are alternative terms shared between researchers, e.g., "SC Media", "Synthetic Complete". There may be many such aliases per entry.

*This work was supported by the Air Force Research Laboratory (AFRL) and DARPA under contract FA875017CO184. This document does not contain technology or technical data controlled under either U.S. International Traffic in Arms Regulation or U.S. Export Administration Regulations. 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 AFRL and DARPA.

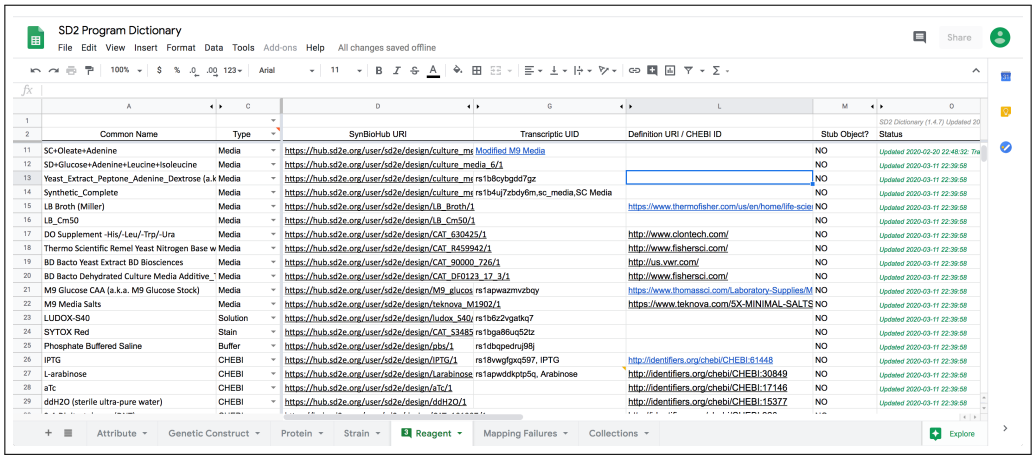


Figure 2: Screenshot of the SBOL Project Dictionary as deployed in the DARPA SD2 program showing key entry columns, including name, type, canonical SynBioHub URI, laboratory UIDs, grounding definition, and status.

- A URL for an entry in SynBioHub, which is guaranteed never to change once an entry has been created.
- Type information (e.g., RNA, DNA, cell strain, media).
- Lab-specific identifiers for each distinct set of collaborators, such as personal shorthands, obsolete terms, or laboratory information management systems (LIMS) unique identifiers (which are generally not human-interpretable). Each lab gets its own column, and there may be many identifiers for each item in each lab.
- Links to a canonical definition, e.g., to curated databases such as ChEBI, NCBI taxonomy, or UniProt, or to the suppliers of complex reagents.
- Status, time last updated, and errors such as type mismatches or detection of duplicate identifiers.

All of these except for the SynBioHub URL and status can be freely edited by researchers at any time, reflecting the ongoing evolution of their private and shared vocabularies (note that ensuring a term's definition remains coherent over time is not generally machine-checkable, and is thus left to the researchers sharing the document). The SBOL Project Dictionary provides different tabs for six main types of vocabulary items—genetic constructs, strains, proteins, reagents, collections, and attributes—with drop-down menus supporting sub-types on each tab as needed.

Users can create and edit entries directly by browsing to the Google Sheet or via software tools they set up to interact with a provided Dictionary Writer API, written in Python. Once entries have been established, other tools can then use the definitions in SynBioHub for integration and display of metadata, mapping between private names, common names, and detailed canonical definitions as needed. When such mappings fail, the SBOL Project Dictionary also provides a "Mapping Failures" tab into which such problems can be reported. Periodically, the dictionary scans this tab and emails the provided contacts for a laboratory to let them know when there are missing entries or errors in entry columns that they are responsible for, such that they can update those entries and resolve mapping failures. This feedback closes the loop to enable not only incremental and post-hoc integration but also incremental and post-hoc error resolution

3 CASE STUDY: DARPA SD2 PROGRAM

In DARPA's Synergistic Discovery and Design (SD2) program, the SBOL Project Dictionary forms a key component in integrating a design-build-test-learn round trip between experimentalists, laboratory automation, and data analysts [1]. Over a period of approximately a year and a half, the SD2 deployment of the SBOL Project Dictionary has been used to curate collaborative terminology for more than 1000 terms: 521 genetic constructs, 304 strains, 54 proteins, 89 reagents, 7 collections, and 62 attributes, a sampling of which are shown in the screenshot in Figure 2. A large fraction of these have been entered and updated incrementally by hand by many different participating researchers, while others have been uploaded using automation—particularly entries for laboratory LIMS identifiers. The entries in the SD2 dictionary have been used to support the integration of data and metadata for dozens of experiments spanning four performing laboratories and five different working groups of researchers, each involving different organisms, goals, and technologies.

REFERENCES

[1] Bryce, D., et al. Round-trip: An automated pipeline for experimental design, execution, and analysis. In *IWBDA* (2020).
[2] Cox, R. S., et al. Synthetic Biology Open Language (SBOL) version 2.2.0. *Journal of integrative bioinformatics 15*, 1 (2018).
[3] Latour, B., and Woolgar, S. *Laboratory Life: The Construction of Scientific Facts.* Princeton University Press, 2013.
[4] McLaughlin, J. A., et al. SynBioHub: A standards-enabled design repository for synthetic biology. *ACS Syn. Bio. 7*, 2 (2018), 682–688.

The Social and Conceptual Organization of Synthetic Biology Ethics

Brandon Sepulvado
sepulvado-brandon@norc.org
NORC at the University of Chicago, Department of Methodology and Quantitative Social Sciences
Bethesda, Maryland

Jacob Jett
jjett2@illinois.edu
University of Illinois at Urbana-Champaign, School of Information Sciences
Champaign, Illinois

J. Stephen Downie
jdownie@illinois.edu
University of Illinois at Urbana-Champaign, School of Information Sciences
Champaign, Illinois

1 INTRODUCTION

The field of synthetic biology, still in its early stages, has largely been driven by experimental expertise, and 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 [3, 7]; however, creating and integrating synthetic components remains an ad hoc process. Additionally, many of the ethical issues concerning the re-purposing of living organisms and their biological processes for processes in the chemical engineering, pharmaceutical, and allied industries remain open areas of philosophical research and social discourse.

The Synthetic Biology Knowledge System (SBKS) is a repository system designed to aid synthetic biologists by interlinking the synthetic biology literature through the application of ontologies. While many of these linking tasks are concerned with DNA/protein sequences, the organisms they are cultivated in, or chemical products of cellular processes, it is equally important to interlink the various ethical issues that concern the tailoring biological life in the form of bacteria and yeasts for participation in industrial processes to consider. Thus, an additional linking task for the SBKS is to link pertinent bioethics articles to biological entities, processes, and products that are central to the research of synthetic biologists. In this paper, we make a preliminary analysis of the state of the ethical discourse in the burgeoning synthetic biology discipline.

2 DATA AND METHODS

The data come from the Web of Science, and we focus in particular two corpora. The first consists of literature from synthetic biology as a field in general, and the second consists of a subset that discusses ethics and ethical implications of synthetic biology work. There exists a well-established focus on iterative strategies for defining interdisciplinary fields [4], and a burgeoning literature applies established approaches to synthetic biology [3, 6, 7]. Shapira et al. [7] conducted an extensive review of search strategies to extract synthetic biology publications, detailing the consequences of all query inclusions and exclusions made by previous literature; we employ their query, which was verified by multiple practicing synthetic biologists.

3 RESULTS

Synthetic Biology

We quickly review the findings of the entire synthetic biology field. The field delineation strategy mentioned above produced 15,152 publications between 1900 and 2019.[1] The earliest publication meeting the selection criteria was in 1913. After 1990, one observes an abrupt increase in texts produced and an exponential increase starting around 2000. Looking at the keywords listed by authors, the top two are by far "synthetic biology" and "metabolic engineering"; other top keywords include *e. coli*, "systems biology", "protein engineering", and "dna". The top three fields represented in synthetic biology publications are biochemistry and molecular biology, biotechnology and applied microbiology, and biochemical research methods; the top three publication outlets are *ACS Synthetic Biology*, *Nucleic Acids Research*, and *PNAS*. Keywords, fields, and journals related primarily to ethics and the social implications of research are notably absent from even the top 20 keywords, sub-fields, and publication outlets.

Synthetic Biology Ethics

The ethics corpus contains 572 publications between 1993 and 2019. Figure 1 visualizes the number of publications per year in this period. A key finding is that the growth starting in 2010 resembles that of synthetic biology more generally starting at 2000; there appears to be a ten-year lag before ethics became a focal topic within synthetic biology.

A correlated topic model (CTM) using stemmed terms [1] suggests that there are nine distinct topics within synthetic biology ethics. Figure 2 plots the proportion of the corpus devoted to each of these nine topics and provides the five most distinctive keywords for each topic. The clustering of the five keywords results from the explicit modeling of correlations among them. One notable finding is that no single topic dominates synthetic biology ethical discourse:

[1] We exclude 2020 because the year is not yet complete.

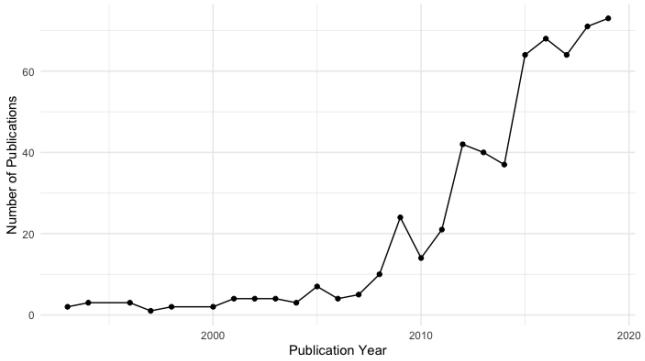


Figure 1: Yearly number of ethics publications

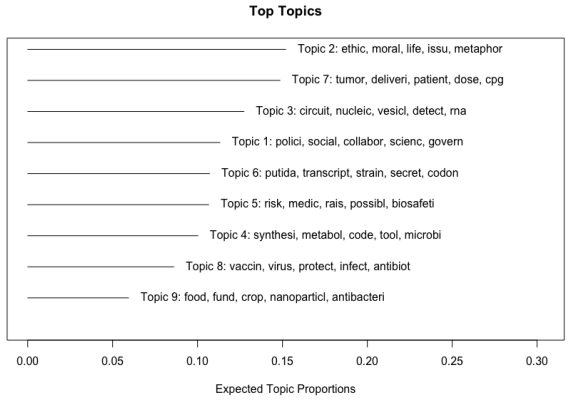


Figure 2: Proportion of topics in ethics corpus

the most prevalent topics each constitute just above 15% of the corpus, and there is a relatively steady decline in prevalence to the least popular topic, which constitutes 6-7% of the corpus. One may see abstract discussions, for example, in Topic 2 that is more philosophical and in Topic 1 that is more political. Topics 5 and 9 appear to occupy a middle ground (re: abstractness), as they discuss medical and food safety, respectively. Other topics, such as 3 and 4, seem to be more focused on concrete synthetic biology practices.

Our final point pertains to the social structure underpinning synthetic biology ethics. Figure 3 plots the number of unique authors and addresses per year. Following the trend from Figure 1, there is a sharp increase in both the unique number of authors and institutions in the synthetic biology publications. One sees in a similar dramatic increase in the fields producing these publications.

These trends suggest that ethical reflection within synthetic biology is at a unique point in its development. "Invisible colleges" [2, 5] describe a form of social organization in new fields where a core set of intellectuals organizes the activities underlying the new field. We conducted a detailed

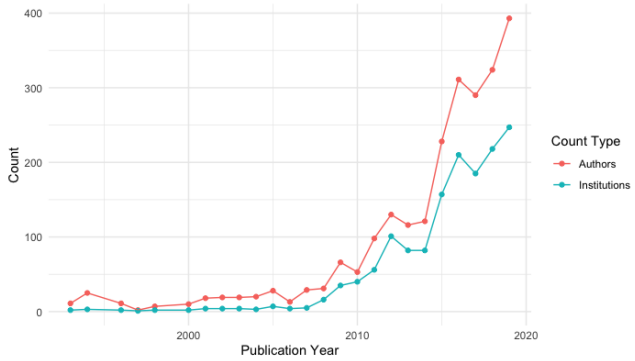


Figure 3: Unique authors and addresses producing ethics publications each year

network analysis of relationships between authors and institutions (to be shown in presentation/poster). Synthetic biology tends to resemble a field transitioning out of an invisible college structure. A core community between collaborators still dominates the field, but this community is rather large and is supplemented by several other communities. Institutions—connected when researchers from each collaborate—are organized into four rather distinct groups: one consisting mostly of U.S. institutions, another consisting mostly of European institutions, a third of largely Chinese institutions, and a four of primarily Japanese institutions. However, synthetic biology ethics remains dominated by a central set of institutions, and authors within this ethics network tend to collaborate only with a small number of other authors. The SBKS and related knowledge systems should incorporate the diverse topics we have identified and the texts from the distinct social and institutional communities.

4 ACKNOWLEDGEMENTS

This material is based upon work supported by the National Science Foundation under Grant Nos. 1939929 and 1939887.

REFERENCES

[1] Blei, D. M., and Lafferty, J. D. A correlated topic model of Science. *The Annals of Applied Statistics 1*, 1 (2007), 17–35.
[2] Crane, D. *Invisible Colleges: Diffusion of Knowledge in Scientific Communities.* University of Chicago Press, Chicago, 1975.
[3] Hu, X., and Rousseau, R. From a word to a world: the current situation in the interdisciplinary field of synthetic biology. *PeerJ 3*, 2008 (2015), e728.
[4] Mogoutov, A., and Kahane, B. Data search strategy for science and technology emergence: A scalable and evolutionary query for nanotechnology tracking. *Research Policy 36*, 6 (2007), 893–903.
[5] Price, D. d. S. *Little Science, Big Science.* Columbia University Press, New York, 1965.
[6] Raimbault, B., Cointet, J. P., and Joly, P. B. Mapping the emergence of synthetic biology. *PLoS ONE 11*, 9 (2016), 1–19.
[7] Shapira, P., Kwon, S., and Youtie, J. Tracking the emergence of synthetic biology. *Scientometrics 112* (2017), 1439–1469.

Discovering Content through Text Mining for a Synthetic Biology Knowledge System

Mai H. Nguyen
Gaurav Nakum
Jiawei Tang
Xuanyu Wu
University of California, San Diego
La Jolla, CA, USA
{mhnguyen,gnakum,jit072,xuw057}@ucsd.edu

Bridget T. McInnes
Nicholas E. Rodriguez
Virginia Commonwealth University
Richmond, VA, USA
{btmcinnes,rodriguezne2}@vcu.edu

Eric Young
Kevin Keating
Worcester Polytechnic Institute
Worcester, MA, USA
{emyoung,kwkeating}@wpi.edu

1 INTRODUCTION

The field of synthetic biology has seen exciting growth in the last few years. Though the amount of data and publications has increased tremendously, the numerous available data sources are fragmented, and locating relevant data for genetic design is a challenging task. For example, finding biological part performance and sequence data remains a manual process of sifting through articles and supplemental material. To address this, we are developing a synthetic biology knowledge system that integrates disparate data and publication repositories to deliver effective and efficient access to available information.

Scientific articles contain a wealth of information about experimental methods and results on biological designs. Due to its unstructured nature and multiple sources of ambiguity and variability, however, extracting information from text is a difficult task. We are exploring various text mining approaches to identify concepts and entities in published articles in order to link each article to other elements in our knowledge system.

This paper describes our work using named entity recognition (NER), a sub-field of text mining, to mine existing literature. The goal of NER is to locate and classify named entities present in text into pre-defined categories. For synthetic biology, examples of such categories are names of genes, vectors, and regulatory elements. NER in biology domains has additional challenges due to the pace of new named entities being added, lack of naming convention, lengthy names, presence of special characters, and frequent and variable use of abbreviations.

2 METHODS

Deep neural network approaches have been applied to NER on biomedical texts. Specifically, state-of-the-art approaches use Long Short-Term Memory (LSTM) [2] with Conditional Random Field (CRF) [3] models and Transformers [6]. In our experiments, we use two deep neural network models developed for biomedical NER, namely HUNER [7] and BioBERT [4].

HUNER (Humboldt-Universität Named Entity Recognition) uses a combination of bidirectional LSTMs and CRFs. LSTMs are a type of deep learning model known as recurrent neural networks that can learn sequential data. Recurrent neural networks have an internal state to retain context from previous inputs, enabling them to learn sequences of data such as speech and text. In a bidirectional LSTM, each input sequence is presented both forwards and backwards so that context before as well as after the word being modeled are captured. A CRF is a discriminative probabilistic graphical model that models the conditional distribution of output variables given observed values. CRFs can also take context into account in making predictions for a data sample, making it ideal for predicting sequential data. In HUNER, a bidirectional LSTM is used to encode forward and backward contexts for the input word, which are then concatenated and fed to a CRF to predict the NER tag for the word.

BioBERT (Bidirectional Encoder Representations from Transformers for Biomedical Text Mining) uses another type of deep neural network model called a Transformer. Transformers are also designed to learn sequence data. Instead of relying on recurrent connections, however, Transformers use an attention mechanism to weigh the relevance of each input in producing the output. BioBERT is pre-trained on top of BERT [1], a general-purpose language representation model. This pre-training was conducted over PubMed abstracts and PubMed Central full-text articles to adapt to biomedical text mining tasks. In our work, the BioBERT output is fed into a simple feed forward neural network for the final NER prediction.

3 DATA

HUNER Data. The HUNER dataset consists of 34 different corpora covering five entity types: Chemicals, Cell Lines, Genes/Proteins, Species, and Diseases. The data was partitioned into 60% training, 10% validation, and 30% testing.

ACS Data. The American Chemical Society (ACS) Data comprises of full text articles from the Synthetic Biology Journal. The data set contains 1,545 articles between the years 2011 and 2019.

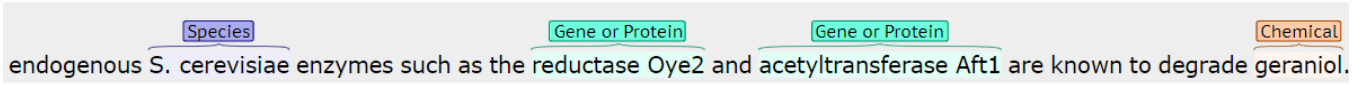


Figure 1: Annotations discovered by our NER model in an ACS article

4 RESULTS

Results on HUNER Test Data. Table1 shows the F-1 scores for HUNER and BioBERT on a subset of the HUNER dataset. F-1 is the harmonic mean between the precision and recall, where precision calculates how many instances are predicted correct out of all instances and and recall calculates how many instances are correctly predicted out of all the correct instances that should have been predicted. The results show that BioBERT obtained a higher F-1 score for all of the datasets except for OSIRIS.

Table 1: HUNER and BioBERT F-1 scores on HUNER Data

Entity Type	Corpus	BioBERT	HUNER
cellline	Gellus	0.924	0.714
	JNLPBA	0.818	0.649
	CLL	0.883	0.730
chemical	SCAI Chemicals	0.931	0.778
	CHEBI	0.877	0.804
	CHEMDNER patent	0.915	0.855
	CHEMDNER	0.920	0.889
	CDR	0.937	0.929
disease	miRNA	0.894	0.823
	NCBI Disease	0.923	0.854
	CDR	0.903	0.837
	SCAI Disease	0.855	0.801
gene	BioCreative II GM	0.898	0.779
	miRNA	0.807	0.697
	Variome Gene	0.928	0.823
	IEPA	0.913	0.824
	BioInfer	0.932	0.846
	DECA	0.731	0.688
	OSIRIS	0.811	0.874
species	Variome Species	0.823	0.701
	s800	0.834	0.725
	miRNA	0.955	0.909

Preliminary results on ACS Data. Our ultimate goal is to extract this type of information from the ACS dataset. To determine the efficacy of applying NER on biology-specific corpora to identify synthetic biology entities, we used our NER model to predict the entity types on 100 randomly selected ACS articles. Figure 1 shows entity mentions found by NER in an article. Each mention is associated with a single entity type. The BRAT annotation software [5] was used to create and view annotations. Figure 2 shows a word cloud that illustrates how often the Chemical types were mentioned in the ACS dataset. The larger the term is in the word cloud, the more often it was identified by NER in the set of ACS articles. Annotations found by the NER model will be reviewed and enhanced by domain experts to create a more refined dataset for fine tuning the model.

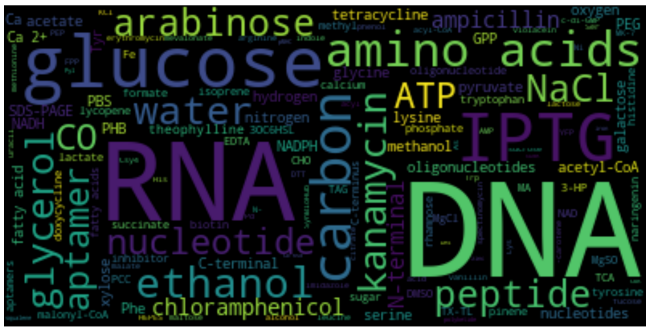


Figure 2: Word cloud of Chemical entities found by NER

5 DISCUSSION

This work presents the application of deep neural network models to identify entities mentioned in scientific articles. The approach described here to extract information about the contents of an article can be used to link publications to data in our knowledge system. The integration of disparate data sources will allow researchers to effectively and efficiently locate related work, enabling maximal leverage of previous research, and has the potential to greatly accelerate exploration and discovery of synthetic biology research.

ACKNOWLEDGEMENTS

This work was funded by the National Science Foundation under Grants No. 1939885, 1939951, and 1939860.

REFERENCES

[1] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[2] Hochreiter, S., and Schmidhuber, J. Long short-term memory. *Neural computation 9*, 8 (1997), 1735–1780.

[3] Lafferty, J., McCallum, A., and Pereira, F. C. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

[4] Lee, J., Yoon, W., Kim, S., Kim, D., Kim, S., So, C. H., and Kang, J. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics 36*, 4 (2020), 1234–1240.

[5] Stenetorp, P., Pyysalo, S., Topić, G., Ohta, T., Ananiadou, S., and Tsujii, J. Brat: a web-based tool for nlp-assisted text annotation. In *Proceedings of the 13th EACL Conference (Demo)* (2012), pp. 102–107.

[6] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. In *Advances in neural information processing systems* (2017), pp. 5998–6008.

[7] Weber, L., Münchmeyer, J., Rocktäschel, T., Habibi, M., and Leser, U. Huner: improving biomedical ner with pretraining. *Bioinformatics 36*, 1 (2020), 295–302.

VisBOL 2.0 - Improved Synthetic Biology Design Visualization

Benjamin Hatch[1], **James A. McLaughlin**[2], **James Scott-Brown**[3], **Chris J. Myers**[1]
[1]University Of Utah, [2]Newcastle University, [3]University of Oxford
benjamin.hat4@gmail.com

1 INTRODUCTION

Visual depiction is an essential tool for both the development and sharing of engineered designs, including those in synthetic biology. The field's determination to develop a well-defined standardized approach for design engineering naturally necessitates a well-defined standard for design visualization. To achieve this, the *Synthetic Biology Open Language Visual* (SBOLv) [1] was created, providing a standardized graphical notation for visualization of biological systems. SBOLv is a complementary standard to the *Synthetic Biology Open Language* (SBOL) [3] standard, which provides a data format that stores both functional and structural information for a given synthetic biology design.

VisBOL was the first visualization tool to directly integrate both SBOLv and SBOL. Implemented in JavaScript, VisBOL was developed for rendering SBOL files on the Web. Currently, it is utilized by web applications such as SynBioHub [2]. Although VisBOL has undoubtedly been useful for visualizing DNA circuits, it has become increasingly outdated as new versions of SBOLv have been released. This paper presents VisBOL 2.0, a heavily redesigned version of the original VisBOL that extends support to more general genetic circuits, further improving web-based visual depiction of synthetic biology designs.

2 RESULTS

By redesigning the architecture of the VisBOL code base, VisBOL 2.0 fixes many of the issues that were causing the VisBOL visualization tool to slowly become obsolete.

Interaction Rendering

The main issue that prompted a redesign of VisBOL was inadequate interaction visualization. SBOL version 2 introduced the capability to represent interactions between different parts of the design. However, the original VisBOL was built to render only simple DNA circuits. The architecture of the VisBOL rendering pipeline made mapping interaction participants to glyphs in the depiction difficult, making consistently accurate interaction rendering placement problematic. In an effort to work around this problem, interactions were rendered separately from the circuit. Figure 1 shows an example of VisBOL attempting to render a genetic design with interactions.

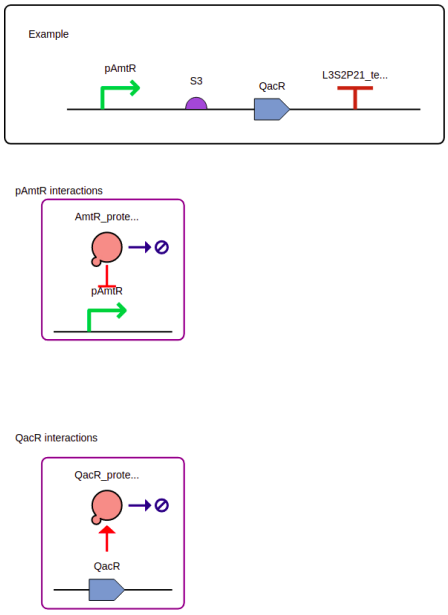


Figure 1: Original VisBOL's rendering of a genetic design with interactions. Note that interactions are rendered separately.

VisBOL 2.0 overcomes this issue by redesigning the rendering process and utilizing a different backing data structure. VisBOL 2.0 constructs the display using a graph data structure, regarding part glyphs as nodes and interactions as edges. Each glyph is assigned its own unique identifier, enabling seamless mapping of interaction participants to glyphs in the rendering. The glyphs are then rendered recursively, consistently placing each glyph and its interactions in the correct location. Figure 2 demonstrates VisBOL 2.0's rendering of the same genetic design the original VisBOL attempts to render in Figure 1.

Viewing Tools

Another important issue with the original VisBOL tool was its lack of visual customization. VisBOL did not support resizing/scaling of its depictions. This limitation became a problem when rendering genetic designs that included hundreds of glyphs; viewing the entire design at once was impossible as it was too large. An inability to view glyphs on a range

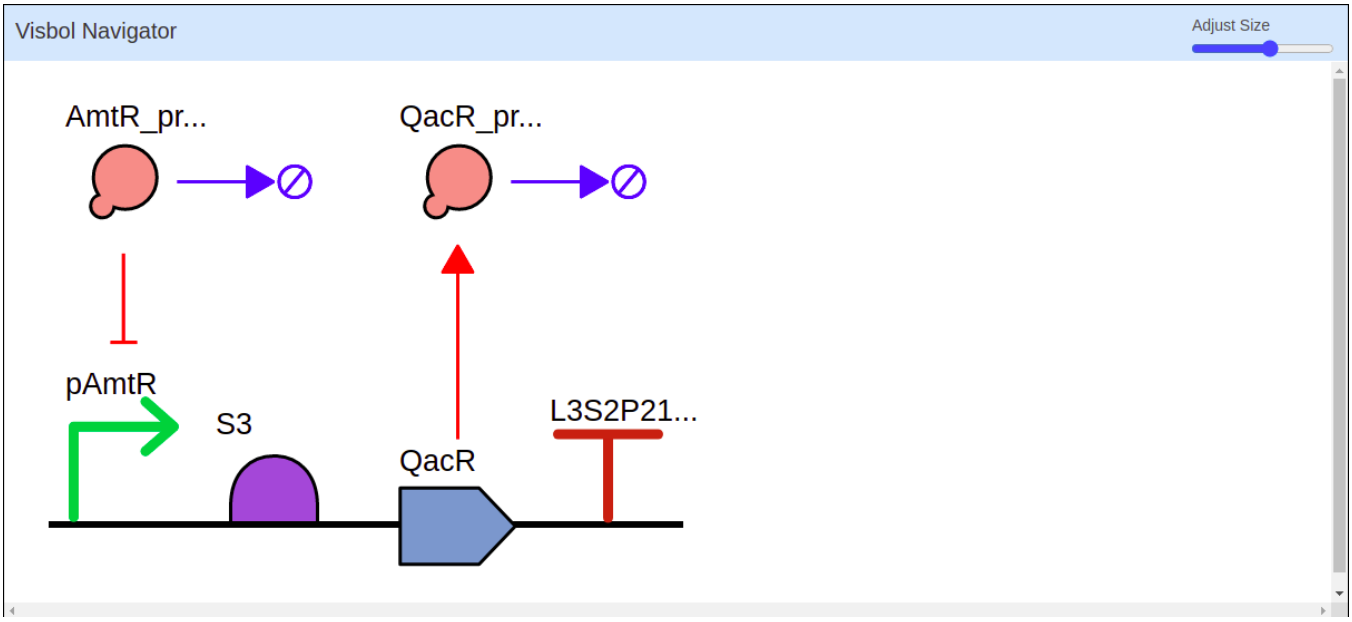


Figure 2: VisBOL 2.0's rendering of the genetic design seen in figure 1. Note that interactions are rendered in their correct locations rather than being rendered separately.

made it frustratingly difficult for users to track what section of the design they were viewing, thereby compounding the effect of not being able to resize large renderings.

VisBOL 2.0 addresses these problems by utilizing parametric SVG and tracking the index of each glyph in the display during rendering. Parametric SVG enables VisBOL 2.0 to resize glyphs in real time without restarting the entire rendering process, making resizing the display efficient and seamless. Keeping track of the index of each glyph in the display allows VisBOL 2.0 to either display glyphs within a certain range or by pages of custom length, making it easier to keep track of what section of the design is being viewed.

Software Scalability

As the original VisBOL attempted to adapt to added capabilities of new SBOL versions, the open-source code base became increasingly difficult to maintain. Since there is not a strict separation of different concerns in the original code, such as gathering relevant information and rendering, implementing significant changes or adding new features to the software, such as interaction rendering, became increasingly difficult.

VisBOL 2.0 implements a separation of concerns, strictly dividing the tasks of parsing design information, creating the backing data structure, and rendering the display. This allows significant changes and new features to be implemented seamlessly. It also enables VisBOL 2.0 to quickly add support for all future SBOL versions.

3 DISCUSSION

VisBOL 2.0 is currently under active development. Listed below are enhancements we plan to begin working on soon:

- Subscribing to a parametric SVG library rather than manually adding new SBOLv glyphs.
- Saving layouts in a standard format enabling exchange with other tools such as SBOLCanvas.
- Improving handling of interaction collisions.
- Enabling users to create constraints for visualization.

VisBOL 2.0's source code can be found on GitHub at https://github.com/VisBOL/visbol-js/tree/visbol_redesign.

REFERENCES

[1] Beal, J., Nguyen, T., Gorochowski, T. E., Goñi-Moreno, A., Scott-Brown, J., McLaughlin, J. A., Madsen, C., Aleritsch, B., Bartley, B., Bhakta, S., Bissell, M., Castillo Hair, S., Clancy, K., Luna, A., Le Novère, N., Palchick, Z., Pocock, M., Sauro, H., Sexton, J. T., Tabor, J. J., Voigt, C. A., Zundel, Z., Myers, C., and Wipat, A. Communicating structure and function in synthetic biology diagrams. *ACS Synthetic Biology 8*, 8 (2019), 1818–1825. PMID: 31348656.

[2] McLaughlin, J. A., Myers, C. J., Zundel, Z., Misirli, G., Zhang, M., Ofiteru, I. D., Goñi-Moreno, A., and Wipat, A. SynBioHub: A standards-enabled design repository for synthetic biology. *ACS Synthetic Biology 7*, 2 (2018), 682–688. PMID: 29316788.

[3] Roehner, N., Beal, J., Clancy, K., Bartley, B., Misirli, G., Grünberg, R., Oberortner, E., Pocock, M., Bissell, M., Madsen, C., Nguyen, T., Zhang, M., Zhang, Z., Zundel, Z., Densmore, D., Gennari, J. H., Wipat, A., Sauro, H. M., and Myers, C. J. Sharing structure and function in biological design with SBOL 2.0. *ACS Synthetic Biology 5*, 6 (2016), 498–506. PMID: 27111421.

Sequence-based Searching For SynBioHub Using VSEARCH

Eric Yu
Chris Myers
University of Utah
Salt Lake City, UT, USA
eric.j.yu@outlook.com,myers@ece.utah.edu

1 INTRODUCTION

With the massive growth of community designed parts, open-source repositories such as SynBioHub [2] have become increasingly popular among synthetic biologists as a convenient way to store and share their genetic designs online. However, the sheer size of such repositories make it difficult to simply browse for the desired parts. The reference instance (https://synbiohub.org) contains over 100,000 publicly available parts in its repository, not counting the various private repositories added by users. Currently, users can only find a part based on a keyword in the part's description, or through various filters such as date of creation, creator, or collection. However, prior to this work, it was not possible in SynBioHub to search for similar sequences.

Well-known tools such as *BLAST* already exist, but are not well-suited for sequence-based searching, as its use of a local alignment algorithm (which aligns a substring of the query sequence to a substring of the target sequence) is more suited for finding patterns between divergent sequences within other domains, which can lead to false positives. *VSEARCH* [3], an open-source alternative to the *USEARCH* [1] tool, uses a more suitable global alignment algorithm, which is more effective at comparing similarities over entire sequences.

VSEARCH was implemented in SynBioHub through *SBOLExplorer* [4], a tool that enhances search by applying PageRank, clustering analysis, and other techniques. Users can either use sequence-based searching through SynBioHub's web interface (Figure 1), or through SynBioHub's API by sending GET requests using *curl* or other languages to get a JSON formatted output instead of a page of results. Various options allow the user to tweak their their search results, which may affect the runtime of *VSEARCH*.

2 RESULTS

With the addition of a new sequence search page at (https://synbiohub.org/sbsearch), users can search by either entering their sequence into a text box or uploading a FASTA or FASTQ file. Table 1 shows the various options that users can adjust when sequence searching.

For users who prefer the command line, SynBioHub's API documentation (https://synbiohub.github.io/api-docs/#search-endpoints) provides instructions to write a GET request using either curl, Python, or JavaScript. An example of a Python script querying for exact matches of a sequence via file upload is shown below:

```
import requests
response = requests.get(
    'http://localhost:7777/search/file_search='
    +'%2FUsers%2Fericyu%2FDownloads%2Fseq.fsa&'
    +'search_exact=true&',
    params={'X-authorization': '<token>'},
    headers={'Accept': 'text/plain'},
)
print(response.status_code)
print(response.content)
```

After submitting the GET request to SynBioHub, users will receive a JSON-formatted output similar to the result below:

```
[{"type":"http://sbols.org/v2#ComponentDefiniti
on","uri":"https://synbiohub.org/public/igem/
BBa_J06480/1","name":"BBa_J06480",
"description":"R0079.B0015",
"displayId":"BBa_J06480","version":"1"}]
```

3 DISCUSSION

Further work is being continually done to add support for more options when using sequence searching through SynBioHub. Additionally, sequence searching will be used as part of the Synthetic Biology Knowledge System (SBKS) that leverages existing data repositories and publications to create a single interface in order to deliver effective and efficient access to collectively available information.

4 METHODS

VSEARCH was made accessible in *SBOLExplorer* through an endpoint implemented in Python using the Flask package, allowing SynBioHub's NodeJS backend to query *SBOLExplorer*.

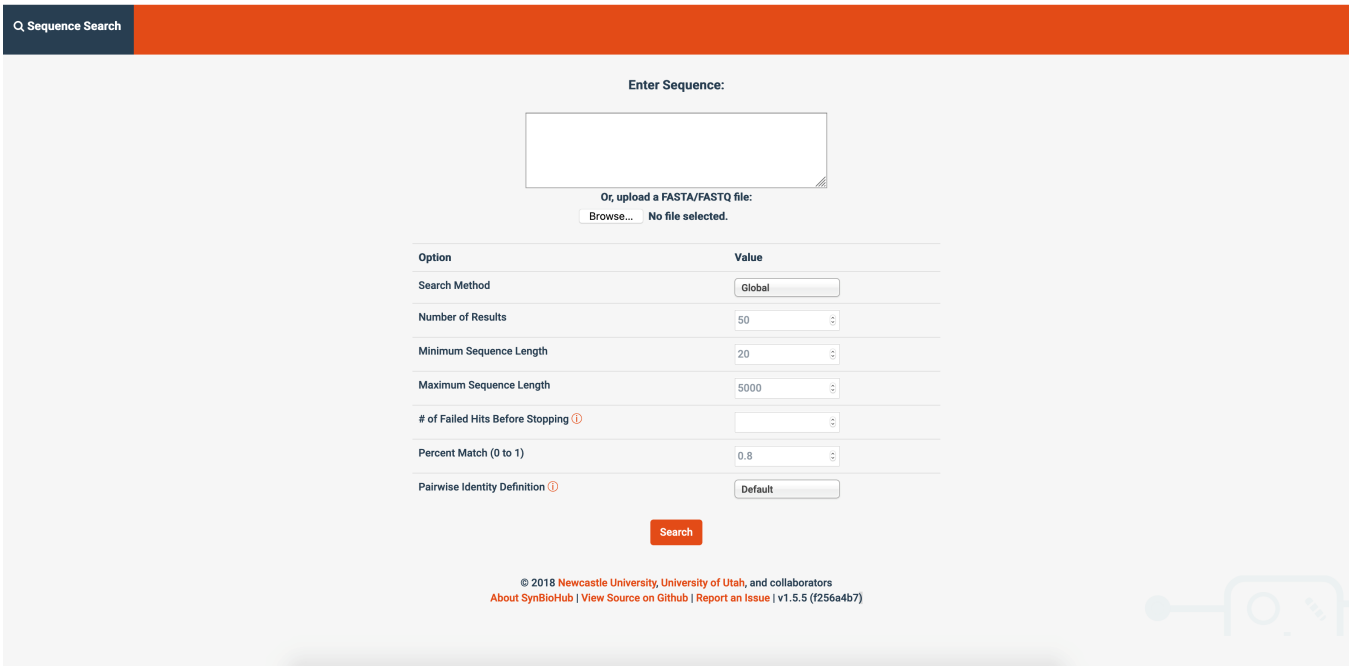


Figure 1: Sequence search tool on SynBioHub

Table 1: Description of search options available to users.

Option Name	Description	Default Value
Search Method	Ability to search by exact match or by some identity threshold (see "Pairwise Identity Definition")	Global
Number of Results	Number of hits before stopping search. Note that a higher number of results will lead to an increase in search time.	50 results
Minimum/Maximum Sequence Length	All sequences below or above the base pair threshold specified will be excluded from the database for comparison.	Min: 20bp Max: 5000bp
# of Failed Hits Before Stopping	Number of false matches before stopping search.	N/A
Percent Match	Float between 0 and 1 specifying percentage identity to query sequence. Anything below the threshold will not be included in the search results.	0.8
Pairwise Identity Definition	Formula to calculate percentage match between query and target sequence.	Edit distance excluding terminal gaps

REFERENCES

[1] Edgar, R. Search and clustering orders of magnitude faster than blast. *Bioinformatics (Oxford, England) 26* (10 2010), 2460–1.

[2] McLaughlin, J. A., Myers, C. J., Zundel, Z., Misirli, G., Zhang, M., Ofiteru, I. D., Goñi-Moreno, A., and Wipat, A. Synbiohub: A standards-enabled design repository for synthetic biology. *ACS Synthetic Biology 7*, 2 (2018), 682–688. PMID: 29316788.

[3] Rognes T, Flouri T, Nichols B, Quince C, Mahé F. VSEARCH: a versatile open source tool for metagenomics. *PeerJ* (2016).

[4] Zhang, Michael and Zundel, Zach and Myers, Chris J. SBOLExplorer: Data infrastructure and data mining for genetic design repositories. *ACS Synthetic Biology 8*, 10 (2019), 2287–2294. PMID: 31532640.

Analysis of the SBOL iGEM Data Set

Jeanet Mante
jv@mante.net
University of Utah
Salt Lake City, Utah

James Mclaughlin
j.a.mclaughlin@ncl.ac.uk
Newcastle University
Newcastle-upon-Tyne, UK

Chris J. Myers
myers@ece.utah.edu
University of Utah
Salt Lake City, Utah

1 INTRODUCTION

Synthetic biology is a movement to standardize genetic engineering and make it more repeatable. An important advancement was the development of standardized genetic parts known as *BioBricks*, which can be composed using restriction enzyme assembly [3]. The iGEM (international Genetically Engineered Machines) competition is an important synthetic biology outreach activity which is run by the iGEM foundation in keeping with their principles of the advancement of synthetic biology via education, competition, and development of an open and collaborative community. As part of the iGEM competition students submit records of any 'parts' they create to the iGEM registry (http://parts.igem.org/Main_Page). The iGEM registry was converted to the *Synthetic Biology Open Language* (SBOL) data format, a standard language for describing genetic designs, [2] and a preliminary analysis of the data was carried out to predict the size of a potential library as well as quantify current problems with the registry data set.

2 RESULTS

The ultimate goal of our analysis is to develop a library of parts (basic and composite) that are well annotated enough to be easily reused and computationally modelled. To do this, this paper proposes thinking of two separate types of data, the innate versus the experimental. The innate data of a part would be the sequence and factors that relate solely to the sequence (such as the fluorescence of GFP). Any data related to the context of the part is considered 'experience' data. For example, the strength of a promoter is experience data as it relates to the organism in which it is used. We suggest a library that contains core parts and their innate data and links in each 'experience' of part use and the data related to that via a provenance annotation. This library model facilitates the reuse of parts as it highlights the 'popularity' (a useful heuristic for confidence) of a part via the number of experiences it has. This also facilitates inter-organism work as it more clearly separates the data linked to a particular model organism and encourages the collection of experimental data (such as strain and growth medium) for every measured property of a part.

As a step towards the development of a library, we analyzed the iGEM SBOL Data set created in 2017. The initial analysis provides an estimate of the number of unique sequences that are useful in future genetic engineering designs.

A detailed analysis is shown in Table 1. This table is based on the sequential application of filters based on the type of part represented using the Sequence Ontology (SO) [1]. The first filter applied was the minimum length filter ("Sequences Over Minimum Length" column). The minimum length used is shown in the column "Minimum Length Parameter". Initially, the minimum length for many parts is set to 6 base pairs (bp) or the equivalent of 2 codons/amino acids (aa). However, for CDS 40 bp (about 13 amino acids) was used as the shortest human enzyme found in UniProt (Cytochrome P450 2A7) is 20aa (60bp). As plasmid and plasmid vectors generally contain a CDS their minimum length was also increased to 40bp. This simple filter removes almost 2,000 components which had no sequence associated with it or a very short sequence. The next filter that was applied looked at unique sequences per SO Type. It removed any exact sequence copies. However, as it worked by SO type the same sequence may, for example, still be repeated as both a terminator and CDS. This can be seen as 33,113 is the total number of unique sequences over a minimum length whereas by role the total is 33,588. Thus 475 sequences are repeated exactly but given different SO types. The final filter of which considers looking for basic parts. Components may be 'basic' or they may be 'composite'. Basic parts do not contain any sub-parts whilst compound parts have one or more sub-parts.

3 DISCUSSION

The initial analysis has indicated that there are probably fewer than **18,000** unique, non-composite sequences which might lead to well described parts with complete records. However, the filtering is not exact. For example, the exclusion of all composite parts is perhaps too strict as there are composite parts (such as the double terminator BBa_B0015) which are useful, and others such as Engineered Regions which would be expected to have sub-parts. As a rough pass, removing parts such as terminators that contain promoters as sub-parts and thus are likely to be mis-annotated, useless, or both, it is a simple and effective heuristic.

The initial filtering removed roughly 45% of the data. Whilst some of this may be too conservative and worth revisiting, there are also parts contained in the final 17,851

Table 1: iGEM Conversion. A sequential filtering was carried out to try and determine the 'useful unique entities' in the converted iGEM dataset. Analysis was carried out per SO type as expectations for different types are different (e.g. RBSs would be expected to be shorter than chromosomes). Sequence count provides a simple count of the number of sequences, sequences over the minimum length is the number of sequences with a length greater than that specified by the minimum length parameter, unique sequences over a minimum length takes the previous count but removes any duplicate sequences within the category. Finally, an analysis was carried out to filter out composite parts; NB: this assumption may be too stringent for types such as Engineered Region.

SO Type	iGEM Types	Minimum Length Parameter	Sequence Count	Sequences Over Minimum Length	Unique Sequences Over Minimum Length	Unique Over Minimum Length Basic Parts
CDS	Basic, Coding	40	7,689	7,198	6,788	6,188
Chromosome	Cell	6	73	13	13	10
Engineered Region	Composite, Device, Generator, Intermediate, Inverter, Measurement, Project, Reporter, Signalling, Translational_Unit	6	20,171	19,477	17,664	3,700
Mature Transcript Region	RNA	6	595	556	538	485
oriT	Conjugation	6	41	39	39	20
Plasmid	Plasmid	40	609	526	484	398
Plasmid Vector	Plasmid_Backbone	40	404	379	369	353
Polypeptide Domain	Protein_Domain	6	769	718	700	665
Primer	Primer	6	582	574	567	567
Promoter	Regulatory	6	3,106	2,965	2,770	2,495
Restriction Enzyme Assembly Scar	Scar	6	40	26	25	24
Ribosome Entry Site	RBS	6	525	494	454	448
Sequence feature	DNA, Other, Terminator	6	3,149	2,734	2,581	1,923
T7 RNA Polymerase Promoter	T7	6	35	32	28	24
Tag	Tag	6	288	263	233	222
Terminator	Terminator	6	388	381	335	329
Total			**38,464**	**36,375**	**33,588**	**17,851**

that still need to be filtered out to create the final part library. We suggest further work on refining part sets based on SO type considering sequence similarity clustering, automated sequence annotation, and machine learning based on the part descriptions provided. We hope to be ready to present the resulting library in the expanded paper arising from this abstract in January. Additionally, we note that whilst the iGEM registry is a large repository of synthetic biology information it does have several drawbacks: 1) The un-standardised nature of fields used, 2) the lack of part verification, 3) the lack of part removal, and 4) part duplication. We suggest that to create a useful library of parts from the iGEM library these four concerns must be addressed, either during a conversion process to SBOL or in the registry itself.

REFERENCES

[1] Eilbeck, K., Lewis, S. E., Mungall, C. J., Yandell, M., Stein, L., Durbin, R., and Ashburner, M. The sequence ontology: a tool for the unification of genome annotations. *Genome biology 6*, 5 (2005), R44.

[2] Galdzicki, M., Clancy, K. P., Oberortner, E., Pocock, M., Quinn, J. Y., Rodriguez, C. A., Roehner, N., Wilson, M. L., Adam, L., Anderson, J. C., and et al. The synthetic biology open language (sbol) provides a community standard for communicating designs in synthetic biology. *Nature Biotechnology 32*, 6 (Jun 2014), 545–550.

[3] Shetty, R. P., Endy, D., and Knight, T. F. Engineering biobrick vectors from biobrick parts. *Journal of Biological Engineering 2*, 1 (Apr 2008), 5.

Dynamic pathway regulation: extended biosensor and controller tuning with multi-objective optimization

Yadira Boada[1], Alejandro Vignoni[1], Ana Fraile[2], Jesús Picó[1], Pablo Carbonell[1]
[1]Synthetic Biology and Biosystems Control Lab, Instituto de Automática e Informática Industrial, Universitat Politécnica de València, [2]Escuela Téc. Sup. de Ingeniería Agronómica y del Medio Natural, Universitat Politécnica de València
{yaboa,alvig2,anfrlo,jpico,pjcarbon}@upv.es

1 BACKGROUND

Natural cells preserve robust growth and endure environmental changes by dynamically adapting cell metabolism by means of complex regulatory networks [6]. However, these complex regulation strategies are the result of years of evolution and they are not compatible with production levels demanded by the industry. Major improvements in yield, titer, and productivity can be achieved by balancing pathway gene expression. There are two different ways of doing this balancing: static control, and dynamic pathway regulation.

Static pathway regulation strategies (Figure 1A) are optimized for a particular situation, and therefore they are incapable of responding to growth and environmental changes that occur during fermentation in a bioreactor [10]. Dynamic balancing addresses the robustness pitfalls of static control through the application of feedback and feedforward regulation (Figure 1B). This makes it possible to obtain higher titers as compared to static regulation [9]. Despite a growing number of success stories, engineering dynamic control remains extremely challenging [5]. Moreover, the performance specifications for synthetic gene circuits and components change significantly with variations in parameters such as temperature, host organism, growth media formulation, and position of the genes in the genome [8]. Model-based design, relying on the principles of control engineering, can provide a powerful formalism to engineer dynamic control circuits and address these challenges. These, together with the tools of synthetic biology, can lead to robust and efficient microbial production at industrial levels [6, 8].

2 METHODS

In this work, we propose using a multi-objective optimization (Figure 1C) approach to optimally tune a recently developed dynamic pathway regulation strategy [4]. The metabolic pathway we used is a phenylpropaniod pathway to produce the metabolite Naringenin (Figure 1A). The controller used to regulate the pathway is the novel antithetic controller [1] in combination with our recently proposed extended metabolic biosensor [4] (Figure 1B). These two pieces together imply a complexity that needs several objectives to be fulfilled simultaneously (i.e. low titer error, fast response to perturbations, parametric robustness, and closed-loop stability among others). In general, these objectives are in conflict and a trade-off must be reached. Multi-objective optimization has shown to be a valuable tool in these situations [3]. With the dynamic system model at hand (developed in [4]), the following steps are necessary for a successful multi-objective optimization: i) define the multi-objective problem (objectives to be optimized), ii) perform the optimization to obtain the solutions (Pareto Front and Pareto Set), and iii) select among the resulting solutions the ones that fulfill the requirements of the design. The result of this optimization is a set of guidelines for the implementation of the biosensor and the controller *in vivo*. Then, when this approach is combined with a collection of parts previously characterized, the results can be interpreted as suggestions about how to select parts like RBS, promoter, plasmid, or enzyme (gene variant) from the collection.

3 DISCUSSION

Several authors have recently explored approaches to help in the tuning of the antithetic controller [2, 7]. Nevertheless, these models and their level of detail are not enough to assist in the *in vivo* implementation of the system.

REFERENCES

[1] Aoki, S. K., Lillacci, G., Gupta, A., Baumschlager, A., Schweingruber, D., and Khammash, M. A universal biomolecular integral feedback controller for robust perfect adaptation. *Nature 570*, 7762 (jun 2019), 533–537.

[2] Baetica, A.-A., Leong, Y. P., and Murray, R. M. Guidelines for designing the antithetic feedback motif. *Physical Biology* (2020).

[3] Boada, Y., Reynoso-Meza, G., Picó, J., and Vignoni, A. Multi-objective optimization framework to obtain model-based guidelines for tuning biological synthetic devices: an adaptive network case. *BMC systems biology 10*, 1 (2016), 27.

[4] Boada, Y., Vignoni, A., Picó, J., and Carbonell, P. Extended metabolic biosensor design for dynamic pathway regulation of cell factories. *iScience In press* (2020).

[5] Gao, C., Xu, P., Ye, C., Chen, X., and Liu, L. Genetic Circuit-Assisted Smart Microbial Engineering. *Trends in Microbiology* (aug 2019).

[6] Liu, D., Mannan, A. A., Han, Y., Oyarzún, D. A., and Zhang, F. Dynamic metabolic control: towards precision engineering of metabolism. *Journal of Industrial Microbiology & Biotechnology 45*, 7 (jan 2018), 535–543.

[7] Olsman, N., Xiao, F., and Doyle, J. C. Architectural principles for

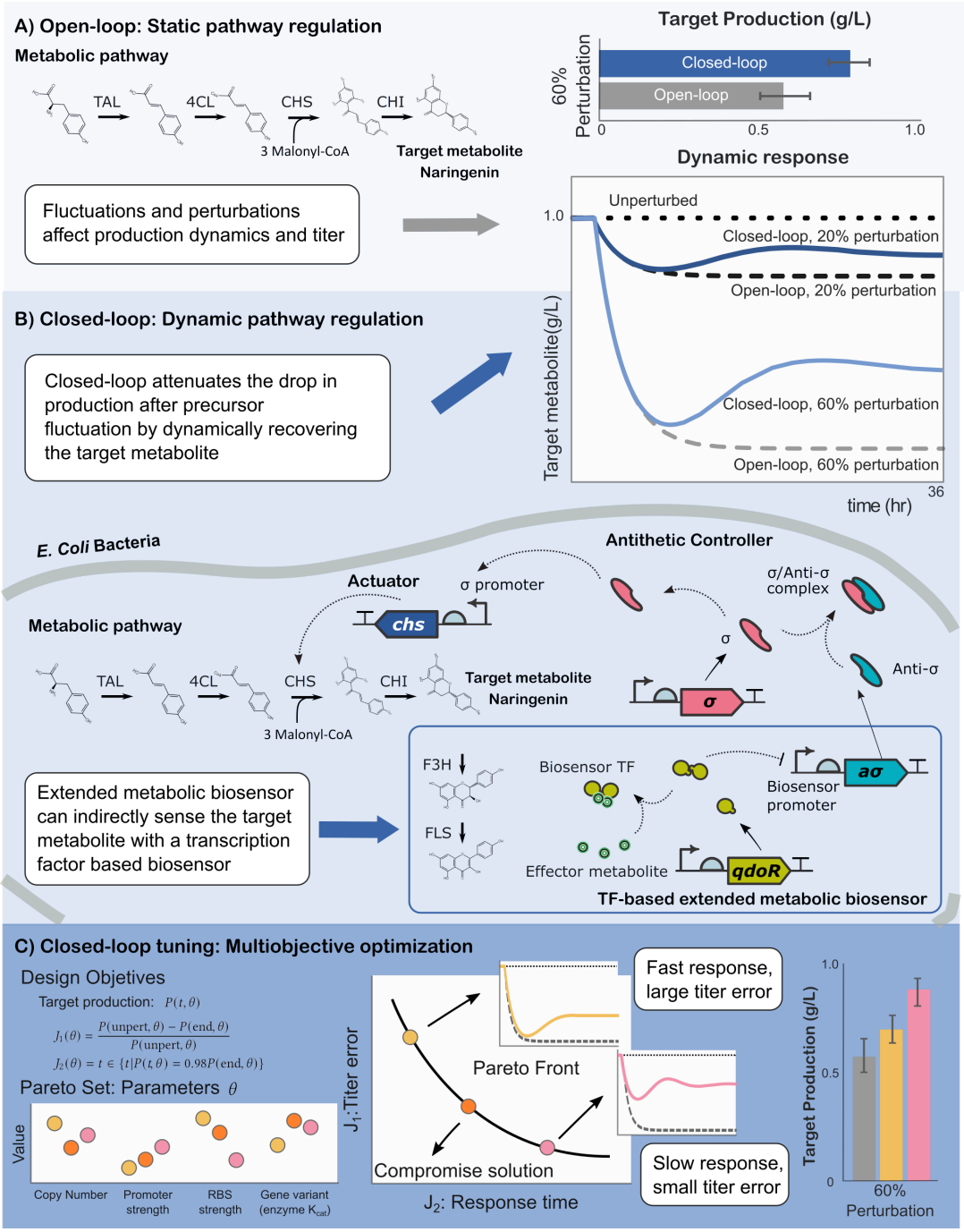


Figure 1: Dynamic pathway regulation: Biosensor and controller parameter tuning.

characterizing the performance of antithetic integral feedback networks. *iScience 14* (2019), 277–291.

[8] Segall-Shapiro, T. H., Sontag, E. D., and Voigt, C. A. Engineered promoters enable constant gene expression at any copy number in bacteria. *Nature Biotechnology 36*, 4 (mar 2018), 352–358.

[9] Stevens, J. T., and Carothers, J. M. Designing RNA-Based Genetic Control Systems for Efficient Production from Engineered Metabolic Pathways. *ACS Synthetic Biology 4*, 2 (feb 2015), 107–115.

[10] Wehrs, M., Tanjore, D., Eng, T., Lievense, J., Pray, T. R., and Mukhopadhyay, A. Engineering Robust Production Microbes for Large-Scale Cultivation. *Trends in Microbiology 27*, 6 (jun 2019), 524–537.

Enhanced Microbial Production of Valuable Natural Products Through Computational Metabolic Models

Michael Cotner
Biological Engineering
Utah State University
Logan, Utah
mike.cotner@aggiemail.usu.edu

Zhen Zhang
Electrical and Computer Engineering
Utah State University
Logan, Utah
zhen.zhang@usu.edu

Jixun Zhan
Biological Engineering
Utah State University
Logan, Utah
jixun.zhan@usu.edu

1 BACKGROUND

This research is based on our previous work on construction of various plant natural product biosynthetic pathways in microorganisms. The engineered metabolic network contains six different enzymes, and by using different combinations of these enzymes, various plant natural products can be formed. For instance, the coexpression of three enzymes can lead to the production of the well-known cardioprotective molecule resveratrol (Figure 1)[7]. Plasmids containing selected genes are transferred into *Escherichia coli*. The engineered strains can be grown in a bioreactor to produce the desired products, which diffuse into the growth media. The products can then be isolated from the media and purified.

As these products are valuable as pharmaceuticals or health-benefiting compounds, it is desirable to optimize their production process within the cell. We present a probabilistic computational model to enable efficient product yield estimation and optimization through stochastic simulation. Optimal yield can be found by tuning parameters such as enzyme and substrate concentration in the computational model,which offers significant efficiency as a computer simulation takes a few seconds where one laboratory experiment can take days or months, along with being more costly.

2 COMPUTATIONAL MODEL

The computational model is built as a system of variables that represent each intermediate in the pathway. The system functions on enzymatic rates that determine when and how frequently one molecule is converted into its successor. The rate of catalysis of an enzyme, v, is given by the Michaelis-Menten kinetic equation shown in Equation 1 below:

$$v = \frac{k_{cat}[E][S]}{K_m[S]} \tag{1}$$

where $[E]$ is the enzyme concentration, $[S]$ is the substrate concentration, and K_m and k_{cat} are enzyme-specific kinetic constants. The rate of catalysis changes as the concentration of substrate changes at each step in the pathway.

Our metabolic model is a *probabilistic* model. Each enzymatic rate in our model represents the probability of the occurrence of the corresponding enzymatic reaction. The model moves stepwise, and at each step evaluates every rate and updates the corresponding reactant and product variables. The higher the rate of an enzyme at a step, the more likely it is to update at that step, in comparison to other reactions with lower rates. This probabilistic functionality means the model varies with each simulation, just as a biological system would vary with each trial. Figure 2 shows a high-level view of the probabilistic model describing the path (colored in blue) that consists of three concurrent steps for producing resveratrol in Figure 2. For each enzyme, the model checks if there is substrate available to be converted. It then calculates the rate, v, which is a function of the substrate and enzyme properties, as shown in Equation 1. The model then decrease the concentration of the substrate and decrease the concentration of product by $1\mu M$ based on the probability represented by the evaluated rate v.

3 RESULTS AND DISCUSSION

The model can successfully simulate the production of any of the products with a reasonable resulting concentration. As of now, experimental data and optimization efforts have been focused on the production of one of the most useful products of the pathway, resveratrol. The production of resveratrol is shown in blue arrows in Figure 1. Table 1 contains the kinetic data for each of the enzymes utilized in resveratrol production, obtained from the literature. The catalytic efficiency of these enzymes, calculated as k_{cat}/K_m, gives a comparable rate for each enzyme. This resveratrol synthesis path model was constructed using the PRISM modeling language [4]. Its discrete-event stochastic simulator was used to generate a large number of simulation traces, which were averaged to give the estimate of product yield of resveratrol.

Using different combinations of rates from different enzymes for TAL and 4CL made a negligible difference on the yield of resveratrol. The efficiency of STS is significantly lower than that of TAL and 4CL and thus, is the rate limiting enzyme. All three kinetic values for TAL from Table 1 used in the simulation gave a similar resveratrol yield of around 170 mg/L, even though they have vastly different rates. In contrast, changing the concentration of STS within the system has a direct effect on the yield of resveratrol. At

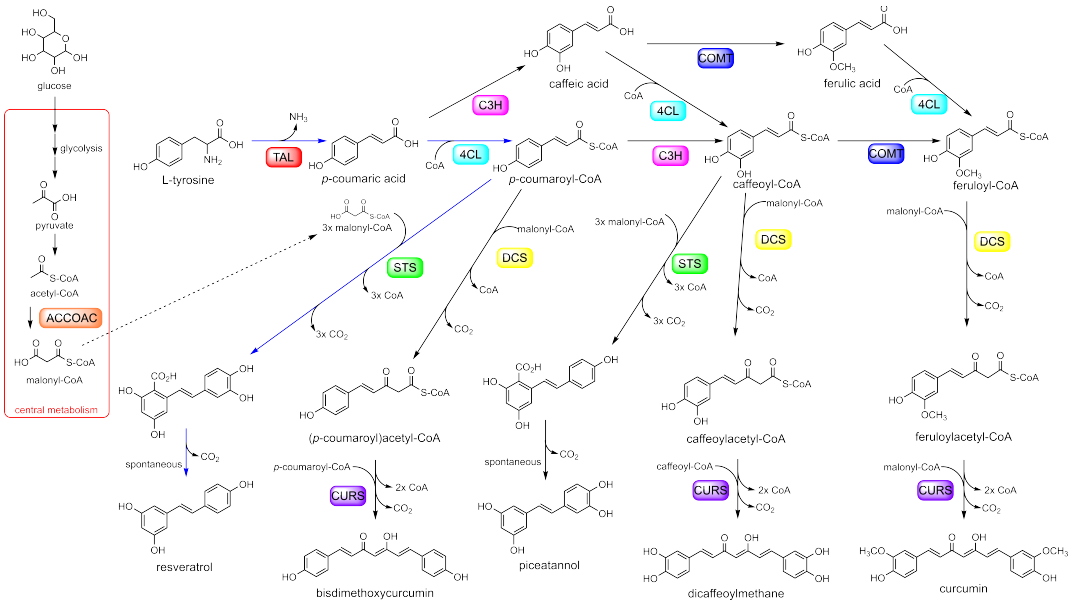


Figure 1: Plant natural product biosynthetic pathways previously established by Wang, *et al.* in *E. coli* [7].

Table 1: Enzyme Kinetics Data.

Enzyme	Genus	$K_m(\mu M)$	$k_{cat}(1/s)$	$k_{cat}/K_m(\frac{1}{s \cdot \mu M})$	Source
TAL	*Saccharothrix*	15.5	0.015	0.967	[1]
TAL	*Saccharothrix*	1492.2	155	103.87	[2]
TAL	*Streptomyces*	433.3	336.5	1287.7	[2]
4CL	*Arabidopsis*	25.1	16.33	650.51	[3]
STS	*Arachis*	6.6	1.8×10^{-3}	0.287	[5] [6]

module resveratrol_pathway
[] tyr>0 → *v*(Kcat_TAL, Km_TAL, E_TAL, tyr):(pca'=pca+1);
[] pca>0 → *v*(Kcat_4CL, Km_4CL, E_4CL,pca)
:(pcoa'=pcoa+1)&(pca'=pca-1);
[] pcoa>0 → *v*(Kcat_STS, Km_STS, E_STS, pcoa)
:(resveratrol'=resveratrol+1)&(pcoa'=pcoa); **endmodule**

Figure 2: The PRISM model for resveratrol biosynthesis.

a concentration of 25, 50, and 100 mg/L of STS in the culture, resveratrol yield was around 44, 85, and 173 mg/L, respectively, indicting that improvement of the STS expression level in *E. coli* could be an effective approach to enhance the production of resveratrol. The concentration of STS in the culture may be improved through genetic manipulation (codon optimization, strong promoter, etc) and high-density fermentation, which will be tested in our future work.

In summary, we have constructed a computational model that can simulate a plant natural product biosynthetic pathway to produce different industrially viable products. We have used this simulation to analyze the production of resveratrol, and identified STS as the limiting enzyme in the pathway. Future directions include experimenting with different model parameters to further improve the production of resveratrol, along with applying these techniques to the other related primary metabolic pathways and their products. Another goal is to include part of the central metabolism of *E. coli*, highlighted in red in Figure 1, to model how the availability of malonyl-CoA affects system.

REFERENCES

[1] Berner, M., Krug, D., Bihlmaier, C., Vente, A., Rolf, M., and Bechthold, A. Genes and enzymes involved in caffeic acid biosynthesis in the actinomycete saccharothrix espanaensis. *Journal of Bacteriology 188*, 7 (Apr 2006), 2666–2673.

[2] Cui, P., Zhong, W., Qin, Y., Tao, F., Wang, W., and Zhan, J. Characterization of two new aromatic amino acid lyases from actinomycetes for highly efficient production of p-coumaric acid. *Bioprocess and Biosystems Engineering 43*, 7 (2020), 1287–1298.

[3] Ehlting, J., Shin, J. J. K., and Douglas, C. J. Identification of 4-coumarate:coenzyme a ligase (4cl) substrate recognition domains. *The Plant Journal 27*, 5 (Jun 2001), 455–465.

[4] Kwiatkowska, M., Norman, G., and Parker, D. PRISM 4.0: Verification of probabilistic real-time systems. In *Proc. 23rd International Conference on Computer Aided Verification (CAV'11)* (2011), G. Gopalakrishnan and S. Qadeer, Eds., vol. 6806 of *LNCS*, Springer, pp. 585–591.

[5] Morita, H., Noguchi, H., Schröder, J., and Abe, I. Novel polyketides synthesized with a higher plant stilbene synthase. *European Journal of Biochemistry 268*, 13 (Dec 2001), 3759–3766.

[6] Schoppner, A., and Kindl, H. Purification and properties of a stilbene synthase from induced cell suspension cultures of peanut*. *Journal of Biological Chemistry 259*, 11 (Jun 1984), 6806–6811.

[7] Wang, S., Zhang, S., Xiao, A., Rasmussen, M., Skidmore, C., and Zhan, J. Metabolic engineering of escherichia coli for the biosynthesis of various phenylpropanoid derivatives. *Metabolic Engineering 29* (May 2015), 153–159.