LOICA: Logical Operators for Integrated Cell Algorithms

Gonzalo Vidal
Pontificia Universidad Católica de Chile
Santiago, Chile
gsvidal@uc.cl

Carlos Vidal-Céspedes
Pontificia Universidad Católica de Chile
Santiago, Chile
carlos.vidal.c@ug.uchile.cl

Timothy James Rudge
Pontificia Universidad Católica de Chile
Santiago, Chile
trudge@uc.cl

1 ABSTRACT
Mathematical and computational modeling is essential to genetic design automation and for the synthetic biology design-build-test-learn cycle. The construction and analysis of models is enabled by abstraction based on a hierarchy of components, devices, and systems that can be used to compose genetic circuits. These abstract elements must be parameterized from data derived from relevant experiments, and these experiments related to the part composition of the abstract components of the circuits measured. Here we present LOICA (Logical Operators for Integrated Cell Algorithms), a Python package for modeling and characterizing genetic circuits based on a simple object-oriented design abstraction. LOICA uses classes to represent different biological and experimental components, which generate models through their interactions. High-level designs are linked to their part composition via SynBioHub. Furthermore, LOICA communicates with Flapjack, a data management and analysis tool, to link to experimental data, enabling abstracted elements to characterize themselves.

2 INTRODUCTION
Synthetic Biology is an interdisciplinary field that mixes life sciences and engineering. From this perspective living systems are objects to engineer, and a rational way to design them is by modifying their genetic code. This can be done by introducing synthetic DNA that encodes a synthetic regulatory network, also known as a genetic circuit. The design-build-test-learn (DBTL) cycle is central to engineering disciplines and each phase requires appropriate tools, standards and workflows, which are still in development in synthetic biology. The synthetic biology open language (SBOL) is an open standard for the representation of in silico biological designs that covers the DBTL cycle and has attracted a community of developers that have produced an ecosystem of software tools [4].

Modeling is key to the DBTL cycle and is essential to the design and learn stages; a model states a well-defined hypothesis about the system operation. Abstraction enables the construction and analysis of models based on components, devices, and systems that can be used to compose genetic circuits. It is the basis for genetic design automation (GDA), which can accelerate and automate the genetic circuit design process. In order for GDA to proceed in a rational way, the abstract elements of genetic circuits must be accessible to characterization, allowing parameterization of models of their operation and interactions.

Functional abstraction of DNA sequences as parts such as transcriptional promoters (Pro), ribosome binding sites (RBS), coding sequences (CDS), terminators (Ter) and other elements has enabled the assembly of relatively small genetic circuits [1–3]. However, for large-scale genetic circuit design higher-level abstractions are required, as provided by the logic formalism [6]. In this approach circuit compositions are abstracted into genetic logic gates that transition between discrete low and high steady-state gene expression levels according to input signals, either external or internal to the circuit [9]. These genetic logic circuits can be designed automatically, in an analogous way to electronic circuits, based on the required discrete logical truth table [6], however this specification requires knowledge of the domain-specific programming language Verilog.

Despite the discrete logical design formalism, these genetic circuits are dynamical systems and can have autonomous, continuous non-steady-state dynamics, displaying complex and rich behaviors from bi-stability to oscillations and even chaos [2, 3, 11]. Furthermore, typical operating conditions for engineered circuits like colonies, bioreactors or gut microbiomes are time varying, which can lead to complex behaviors from even simple genetic circuits [7].

To design genetic circuit temporal dynamics we therefore require kinetic gene expression data generated at the test phase. This data must be integrated with models to enable characterization of abstracted parts, devices and systems, as well as metadata, including the DNA part composition, to enable automated design. Thus there is a need for software design tools that integrate abstract circuit designs, dynamical models, kinetic gene expression data, and DNA part composition via common exchange standards in a user-friendly and accessible fashion.
3 RESULTS

Logical Operators for Integrated Cell Algorithms (LOICA) provides a high-level genetic design abstraction using a simple and flexible object-oriented programming approach in Python. LOICA integrates models with experimental data via two-way communication with Flapjack, a data management and analysis tool for genetic circuit characterization [12]. This communication not only provides direct access to experimental characterization data, but also to DNA design composition and sequence via SBOL contained in SynBio-Hub [5], enabling characterization and simulation in the same tool, but also facilitating exchange with other tools such as iBioSim [11].

The basic objects in LOICA are Operator and GeneProduct, which may be either a Regulator or Reporter (Figure 1A). A Regulator represents a molecular species that regulates gene expression. A Reporter is a molecular species that provides a measureable signal, such as a fluorescent protein. The Operator maps one or more Regulator concentrations to one or more GeneProduct synthesis rates. An Operator can be implemented in DNA as a combination of Pro and RBS, and the Regulator could be a CDS of a transcription factor or of a regulatory RNA. The interactions between the Operators and the Regulators encode models for genetic circuit temporal dynamics, which are simulated with differential equations. The system is thus:

\[
\frac{dp}{dt} = \Psi(r) - \Gamma p - \mu(t)p, \quad (1)
\]

\[
\Psi(r) = \sum_k \Phi_k(r), \quad (2)
\]

where \(p = (p_0, p_1, \ldots, p_{N-1})^T\) is the vector of GeneProducts, which includes different Regulators \(r = (r_0, r_1, \ldots, r_{M-1})^T\) and Reporters \(s = (s_0, s_1, \ldots, s_{N-M-1})^T\). The non-linear operator \(\Psi\) maps Regulator concentrations to GeneProduct synthesis rates. \(\Gamma\) is a diagonal matrix of GeneProduct degradation rates, and \(\mu(t)\) is the instantaneous growth rate of the cells. Equation 1 shows the overall system where \(\Psi\) encodes the whole circuit, and consists of a sum of individual LOICA Operators \(\Phi_k\) (Equation 2).

Figure 1B shows a mathematical model that results from the interaction of an Operator encoding a simple NOT logic modeled with a Hill equation as transfer function. It can be implemented as a promoter containing repressor binding sites combined with a ribosome binding site (RBS). The logical Operator can thus be instantiated as a genetic device that is repressed by an input Regulator and outputs a GeneProduct synthesis rate. Note that LOICA can be used to define an Operator as any operation that maps from input Regulator concentrations to output synthesis rates.

The repressilator is a useful dynamical system case study because it produces continuous sustained oscillations that escapes ON/OFF logic [2]. To model it, we consider a simple balanced ring oscillator with three NOT Operators connected with three different Regulators. The Operators and Regulators are incorporated into a GeneticNetwork, linked with a Flapjack Vector, which with the Metabolism drives the dynamics of the Sample, also corresponding to a Flapjack Sample (Figure 1A). For the circuit to produce a measurable signal we add three Operators using the same inputs but changing the outputs to three different Reporters, linked with the Flapjack Signal model (Figure 2A). The code to generate this model is in Figure 2B. This approach is used to generate synthetic data (a LOICA Assay) from models that can be uploaded to Flapjack. It is then easy to access Flapjack’s genetic circuit characterization tools, data management and data visualization through a Python package (pyFlapjack) or the web interface shown in Figure 2D.

We have described how to use LOICA to generate and analyze simulation data from models. Another example workflow goes from data to model parameterization and could be as follows for the learn stage of the DBTL cycle. First, a GeneticNetwork is assembled from a collection of Operators, linked by various Regulators and Reporters in some topology. Each Operator is then linked to experimental data contained in Flapjack, which corresponds to measurements of the auxiliary circuits required to parameterize the model encoded. For example, a NOT Operator links to data of a chemical signal receiver and a chemical signal inverter measured in a range of signal concentrations (LOICA and Flapjack Supplement). The Operator provides a function that then extracts the data from Flapjack and uses it to parameterize the corresponding model (Figure 1B). In the example shown that means fitting parameters \(a, b, K, n\) by least squares minimization of the difference between the experimental data and the solution of differential equation models of the auxiliary circuits (Equation 1). Each Operator thus contains the information required to characterize itself. Therefore with LOICA data driven models that include mathematical constraints it is possible to design circuit topologies and look for likely functional designs initiating a new round of the DBTL cycle.

4 CONCLUSION

LOICA integrates the design and characterization of genetic circuit dynamics into Python workspaces, providing an easy-to-understand design abstraction implemented using simple object-oriented programming principles. This programming interface does not require specialist or domain-specific knowledge, but leverages common programming skills, making it accessible but also providing customization capabilities for advanced users. LOICA is able to simulate abstract genetic circuit designs using differential equation models. It abstracts genetic circuit designs into objects which are capable of characterizing themselves via links to data in Flapjack. As
data relating to genetic components is updated in Flapjack, the fitted parameters can be automatically updated upon characterization. Flapjack also provides the connection to SynBioHub which allows design and characterization based on SBOL.

5 FUTURE WORK

We aim to complete and automate the DBTL cycle in synthetic biology, proposing a workflow that integrates LOICA, Flapjack [12] and SynBioHub [5] via SBOL3 [4]. In the design stage SBOL/SBML will provide flexibility to use other existing bioCAD tools such as iBioSim [11] as part of the workflow. To make a direct transition to the build stage LOICA will generate a human/machine readable protocols for assembly using Opentrons liquid handling robots [10] and their open Python API. We will expand LOICA’s capabilities to single-cell resolution spatio-temporal systems by connecting it to CellModeller [8] for individual based modeling and stochastic simulations. For the test stage we will develop open science hardware for measuring genetic circuit dynamics that could be used to obtain kinetic data for direct upload to Flapjack.
Figure 1: Diagram of model generation in LOICA. A. Diagram of an Assay encapsulating a Sample which in turn encapsulates Metabolism, Supplement, and GeneticNetwork. In the later, the Operator and Regulator are interacting to generate a model. On the right side the different interactions with the Flapjack and SynBioHub models are shown. B. Mathematical model of a NOT Operator with one input and one output generated through LOICA object interactions. Here $p_{OUT}$ is a GeneProduct (Regulator or Reporter), output of the Operator. In the Operator $a$ is the basal or leaky gene expression, $b$ is the regulated gene expression, $r_{IN}$ is a Regulator concentration, $K$ is the switching concentration, and $n$ is the cooperativity degree of $r_{IN}$ with respect to the Operator. In the GeneProduct $\gamma$ is the degradation rate of $p_{OUT}$. In Metabolism $\mu(t)$ is the instantaneous growth rate which dilutes $p_{OUT}$. Here the Operator is encoded by a Hill equation transfer function that states its regulation by $r_{IN}$. The transfer function could be any mapping from input concentration to synthesis rate, making LOICA Operators flexible and easy to extend.
Figure 2: Example of repressilator design in LOICA. A. LOICA diagram of the modeled repressilator circuit and respective symbols. B. Python code that generates a repressilator in LOICA. GeneticNetwork construction is the first step where the user states all the objects and their interaction. C. Next during Assay setup the user initializes and runs the simulation, and the results can be uploaded to Flapjack. The two-way communication with Flapjack allows data storage and management, enables various analyses to be performed, and allows Operators to characterize themselves. D. Data exploration in Flapjack via the web interface. This interface allows to query, analyze and plot the data, which can also be done through the pyFlapjack Python package.
REFERENCES

[1] Danino, T., Mondragón-Palomino, O., Tsimring, L., and Hasty, J. A synchronized quorum of genetic clocks. *Nature* 463, 7279 (2010), 326–330.

[2] Elowitz, M. B., and Leibler, S. A synthetic oscillatory network of transcriptional regulators. *Nature* 403, 6767 (2000), 335–338.

[3] 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.

[4] McLaughlin, J. A., Beal, J., Misirli, G., Grünberg, R., Bartley, B. A., Scott-Brown, J., Vaidyanathan, P., Fontanarrosa, P., Oberortner, E., Wipat, A., et al. The synthetic biology open language (sboL) version 3: simplified data exchange for bioengineering. *Frontiers in Bioengineering and Biotechnology* 8 (2020), 1009.

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

[6] 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).

[7] Peña, G. A. V., Vidal-Céspedes, C. I., et al. Accurate reconstruction of dynamic gene expression and growth rate profiles from noisy measurements. *bioRxiv* (2021).

[8] Rudge, T. J., Steiner, P. J., Phillips, A., and Haseloff, J. Computational modeling of synthetic microbial biofilms. *ACS synthetic biology* 1, 8 (2012), 345–352.

[9] 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 (2020), e9401.

[10] Storch, M., Haines, M. C., and Baldwin, G. S. Dna-bot: a low-cost, automated dna assembly platform for synthetic biology. *Synthetic Biology* 5, 1 (2020), ysa010.

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

[12] Yáñez Feliú, G., Earle Gómez, B., Codoceo Berrocal, V., Muñoz Silva, M., Nuñez, I. N., Matute, T. F., Arce Medina, A., Vidal, G., Vidal Céspedes, C., Dahlin, J., et al. Flapjack: Data management and analysis for genetic circuit characterization. *ACS Synthetic Biology* 10, 1 (2020), 183–191.