ACS SyntheticBiology

Research Article

pubs.acs.org/synthbio

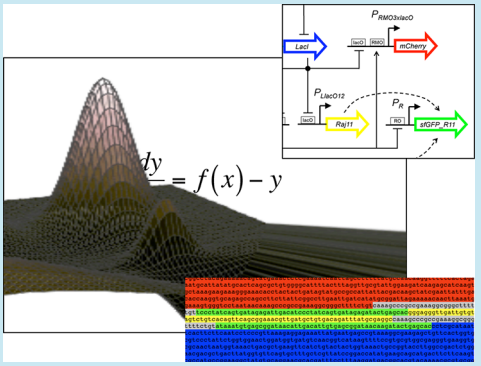
AutoBioCAD: Full Biodesign Automation of Genetic Circuits

Guillermo Rodrigo and Alfonso Jaramillo*

Institute of Systems and Synthetic Biology, CNRS UPS3509, Université d'Évry Val d'Essonne - Genopole, 91030 Évry Cedex, France

Supporting Information

ABSTRACT: Synthetic regulatory networks with prescribed functions are engineered by assembling a reduced set of functional elements. We could also assemble them computationally if the mathematical models of those functional elements were predictive enough in different genetic contexts. Only after achieving this will we have libraries of models of biological parts able to provide predictive dynamical behaviors for most circuits constructed with them. We thus need tools that can automatically explore different genetic contexts, in addition to being able to use such libraries to design novel circuits with targeted dynamics. We have implemented a new tool, AutoBioCAD, aimed at the automated design of gene regulatory circuits. AutoBioCAD loads a library of models of genetic elements and implements evolutionary design strategies to produce (i) nucleotide sequences encoding circuits with targeted dynamics that can then be tested experimentally and (ii) circuit models for testing regulation principles in natural systems, providing a new tool for synthetic biology. AutoBioCAD can be used to model and design genetic circuits with dynamic behavior, thanks to the incorporation of stochastic effects, robustness, qualitative dynamics, multiobjective optimization, or degenerate nucleotide sequences, all facilitating the link with biological part/circuit engineering.



KEYWORDS: *evolutionary computation, combinatorial optimization, regulatory network, design automation*

The engineering of genetic information systems in living cells requires the processing of intricate signals between analog and noisy biological systems. In the long run, we think that it will not be possible to keep the current manual engineering used in synthetic biology. A new engineering paradigm in synthetic biology consists of the design, construction, and characterization of synthetic circuits all carried out, unsupervised, by computers. We can define biological design automation (BDA) analogously to electronic design automation,[1] as the field that develops computational tools for synthetic biology following a standard design pipeline. Alternatively, we could exploit the *automation* terminology to the extreme case, where the computer is not just aiding but doing the full design unsupervised. For instance, it is not the same to use MS Word to write a book as to let your computer write the book. In synthetic biology, we call this approach full biodesign automation (FBDA), and our tool AutoBioCAD implements it by outputting, in standard formats, the mathematical models and nucleotide sequences of gene circuits from an input consisting in a specified behavior (Figure 1).

To implement such FBDA, we use evolutionary computation algorithms.[2,3] They consist of optimization iterations involving a mutation operator and a fitness function for selection. In biological applications, only the FBDA of proteins and nucleic acids has allowed the *de novo* production of a nucleotide sequence.[4−7] However, this has not been possible in the case of regulatory networks, due to the insufficient prediction capability of the effects of arbitrary mutations on function. We propose to resolve this challenge by taking advantage of the development of genetic regulatory elements with characterized behavior[8−10] and of the modularity observed when exchanging such elements.[10−12] This modularity has already been exploited in the rational design of gene networks from a set of standard models selected from a library.[13−18] In this work, we also exploit the modularity of transcriptional and post-transcriptional regulatory elements to design functional circuits,[7] discarding interferences between transcription and translation.[12,19] Importantly, we can also use evolutionary computation to determine the optimal kinetic parameters for given biological parts in such a way that the corresponding assembled circuit best fits experimental data.

Here, we are not restricted to the discovery of the optimal set of parts that match a specified behavior for a fixed topology.[20−22] Rather, we aim to solve the general problem of regulatory network design (Figure 1). Earlier computational design approaches[23−30] were not aimed at producing a nucleotide sequence that could be tested experimentally, nor at exploiting a library of standard models characterized from experimental data. Our approach is also different from the one presented by Beal and colleagues,[31] because we exploit evolutionary computation to design the network and the sequences in one single step. Our methodology is independent of the chosen mathematical model, is scalable, and is easily updated. It also allows us to work with multiple libraries and to optimize network topology. Here, we provide an open-source

Special Issue: IWBDA 2012

Received: September 15, 2012
Published: November 19, 2012

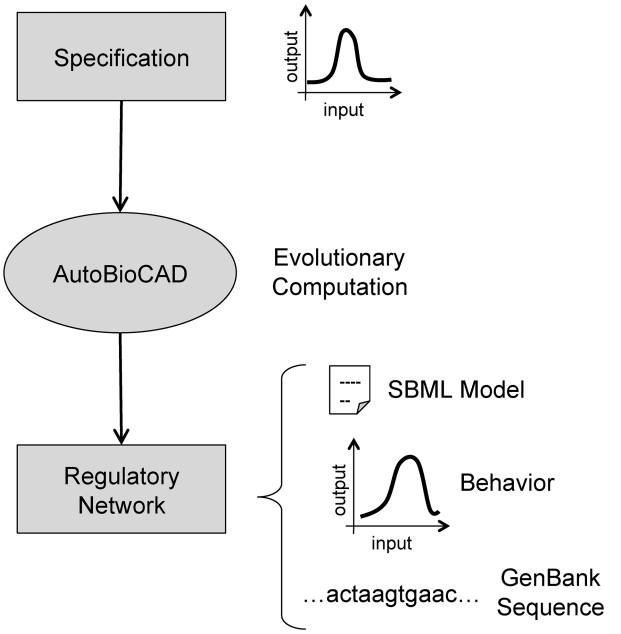


Figure 1. AutoBioCAD designs regulatory networks from the specified behavior of the system and outputs a SBML file containing the mathematical model of that system, its simulated dynamics (in deterministic or stochastic regime), and its nucleotide sequence. The behavior can be stationary (e.g., digital devices) or dynamic (e.g., oscillators). AutoBioCAD follows an evolutionary computation approach to implement the FBDA.

software suite (AutoBioCAD) that implements an automated design process that assembles SBML models[32] of genetic elements, evaluates the performance of the resulting network according to its dynamics, and couples this with appropriate landscape-sampling techniques in order to find the nucleotide sequences of the desired functional networks. AutoBioCAD is a new computational tool for the *de novo* circuit design problem (FBDA) that, by exploring both the topology and parameter spaces, is able to produce testable nucleotide sequences.

RESULTS AND DISCUSSION

We developed AutoBioCAD to design genetic networks with a targeted behavior by using a library of mathematical models of biological parts (Figure 2). By taking advantage of a combinatorial construction of genetic networks,[11] AutoBioCAD is a design platform that harnesses a library of models to uncover the possible dynamical behaviors. AutoBioCAD supports two design strategies, enumeration and optimization, and can account for stochastic effects. In the design by enumeration strategy, several networks are constructed and simulated to explore the design space. Then, the networks that show the dynamical behavior of interest are selected. This approach may be exhaustive if the libraries are small, or heuristic, guided by rational design techniques. In the design by optimization (FBDA) strategy, an evolutionary scheme is applied to iteratively assemble models of existing parts and evaluate the performance of the resulting network according to a dynamic behavior-based fitness function.[33]

Assembly of Part Models. The mathematical model of a network is constructed by assembling models of basic parts. These parts can be promoters, genes (coding or non-coding), and devices. Each part is modeled by transfer functions that relate the output to the input values. Promoter models account only for the reaction of transcription, which we approach with a Hill-like function.[34] Here we assume that all genes downstream of a promoter have the same transcription rate. Gene models account for the reactions of translation (only for coding sequences) and degradation/dilution of mRNA and proteins. We consider first-order and enzymatic degradation kinetics, without considering the coupling between species.[35] Non-coding genes are assumed to be small RNAs that can interact with other elements of the system by exerting regulatory actions.[7] Devices are independent modules with defined function, such as operons, and can be considered by themselves as networks. We only allow the assembly of the following instances: (i) promoter + gene, (ii) gene + gene (polycistronic elements, except for non-coding genes, which form unique transcription units), (iii) gene + promoter, or (iv) add device. To store mathematical models, in addition to the part sequence, we used the SBML format.[32] Hence, we have an SBML file for the model of each biological part, just as crystallographic data is stored in PDB format.[36] Models can represent theoretical parts (for a design process with higher degrees of freedom) or experimental parts (where the SBML includes the nucleotide sequence). The nucleotide sequence of the assembled network is written annotated in GenBank format.[37]

Degenerate Sequences. We introduce the concept of degenerate part, defined as a biological part with a sequence, which contains a degenerate nucleotide known to produce a variation in the associated kinetic parameters when mutated. Solutions containing degenerate parts will produce degenerate sequences suitable for experimental screening. This allows the introduction of a range of variation for the kinetic parameters associated with the models of some parts (evolvable kinetic parameter), allowing the exploration of a larger solution space. For instance, we may know how to mutate the sequence of a given promoter in order to modulate the transcription rate or the dissociation constant with its regulator.[38,39] In another example, we could also know the effects of mutating the 5′ untranslated region (UTR) of an mRNA to change its translation rate,[40] which would be included as a single part with degeneracy. This part re-engineering would be done only after finding an optimal solution, avoiding the need of constructing large libraries. The designer would either construct the required part or, if experimental screening for the desired function is possible, create the corresponding degenerate circuit library.

FBDA I: Optimization Scheme. An initial network is iteratively modified against dynamical constraints toward a solution that satisfies the specified behavior. We used Monte Carlo Simulated Annealing (MCSA) as optimization scheme.[41] A mutation operator modifies the network, either by changing an evolvable kinetic parameter or by changing topology of the network. For degenerate parts, we considered a discretization of 10 intervals taken in a logarithmic scale between the minimal and maximal values of the parameters susceptible to be evolved. Topological changes include part replacement, addition, or deletion, avoiding the elimination of input- and output-related elements. Each mutation event has a probability of occurrence that can be defined by the user. Internally, the algorithm may adjust on the fly the probability of parameter change in order to better explore the landscape. After the mutation operator is applied, a fitness function based on the dynamics of the network is evaluated to decide whether the mutation is accepted or rejected. To avoid the solutions of the dynamics of

dx.doi.org/10.1021/sb300084h | *ACS Synth. Biol.* 2013, 2, 230−236

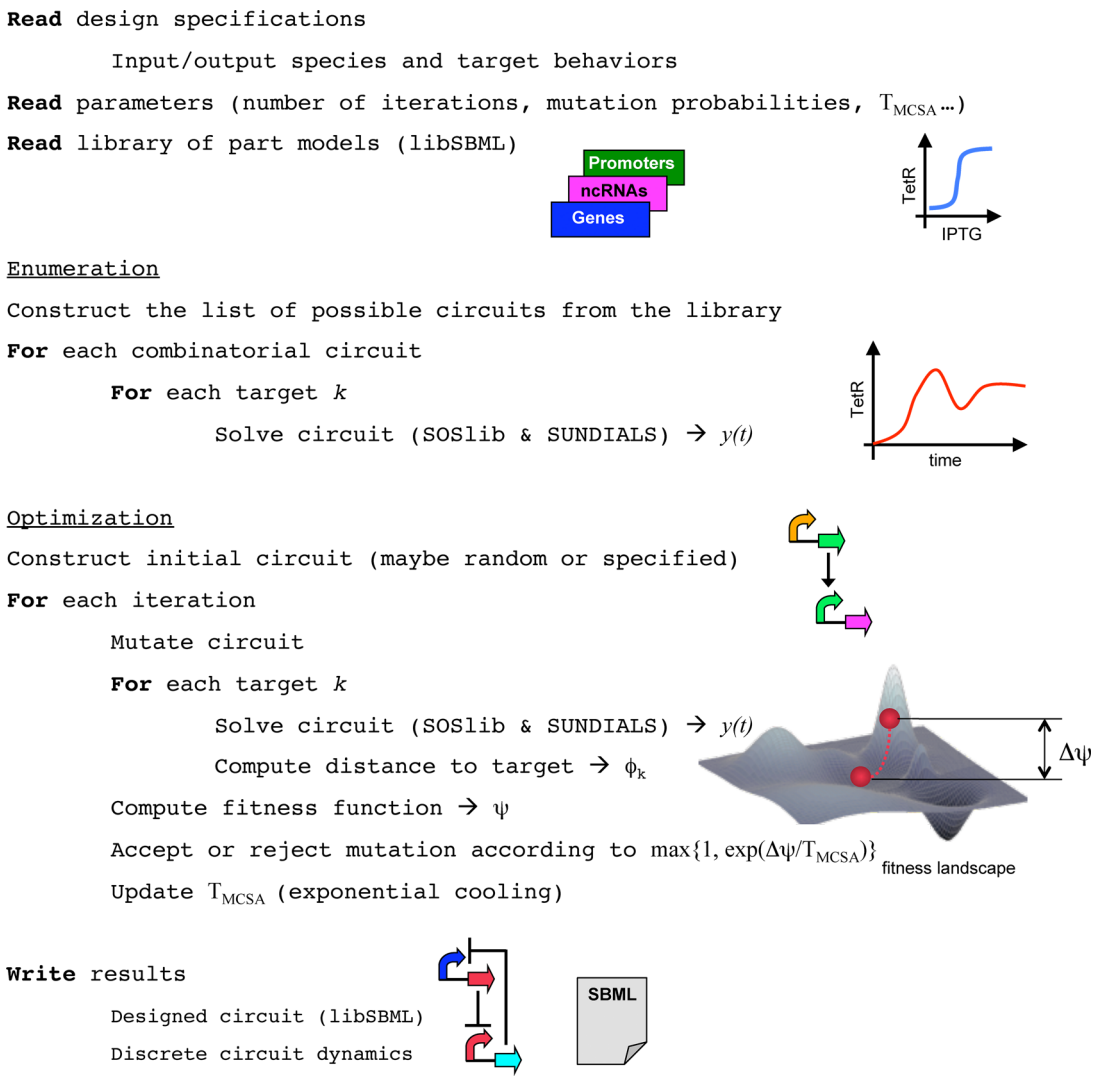


Figure 2. Pseudocode of AutoBioCAD, illustrating two different design strategies: enumeration and optimization of circuits, according to a given library of models.

networks where the output and input are not connected (i.e., systems where the input does not influence the output), we calculate the adjacency matrix and its *n*-th powers at each step. When the fitness gets stalled for several iterations (e.g., 10% of the total), we apply a genetic drift operator to produce a longer jump in the solution space. This jump is random within the first half of the optimization process; otherwise we restart from the best solution found until then.[42] The MCSA temperature is continuously adjusted during the process following an exponential cooling scheme. Accordingly, the initial temperature is usually chosen to be high, hence moving during evolution from random to adaptive walk.

FBDA II: Convergence. A potential limitation of the FBDA approach is the convergence, in this case, of the MCSA, as the time required to find a sufficiently good solution scales with the complexity of the desired behavior.[26,27,29,30] To improve the convergence, AutoBioCAD admits a fitness function rewarding qualitative dynamical behavior, done by implementing a Boolean discretization of the dynamics. Our algorithm exploits the topological properties of the circuit during optimization to avoid unfruitful computation, although for complex functions this could be very time-consuming.

Computational Realization. The program is implemented in C++ and it reads/writes SBML files. It is compiled and executed under Linux and Mac OS X environments. The program reads the design specifications and algorithm parameters and exports the designed network model in a SBML file. The libraries SUNDIALS,[43] libSBML,[44] and SOSlib[45] were linked. AutoBioCAD supports both deterministic and stochastic dynamical simulation, based on ordinary differential or Langevin equations, respectively. A degree of robustness can also be specified to obtain networks insensitive to parameter perturbations,[33] although at a high computational cost. For optimization, the convergence basically depends on the complexity of the targeted behavior, and the dynamics solver is the most time-consuming step for a given iteration. AutoBioCAD can run distributed in high performance computing clusters.

Application. As an advanced example, we applied AutoBioCAD for designing a minimal biological arithmetic logic unit (i.e., half-adder).[46] This device performs the addition with carry of two one-bit binary signals. We specified IPTG and aTc as inputs and sfGFP and mCherry as outputs (sfGFP the adder, and mCherry the carry). The desired behavior consists of sfGFP = IPTG XOR aTc and mCherry = IPTG AND aTc

dx.doi.org/10.1021/sb300084h | *ACS Synth. Biol.* 2013, 2, 230−236

Half-adder truth Table

IPTG	aTc	sfGFP	mCherry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

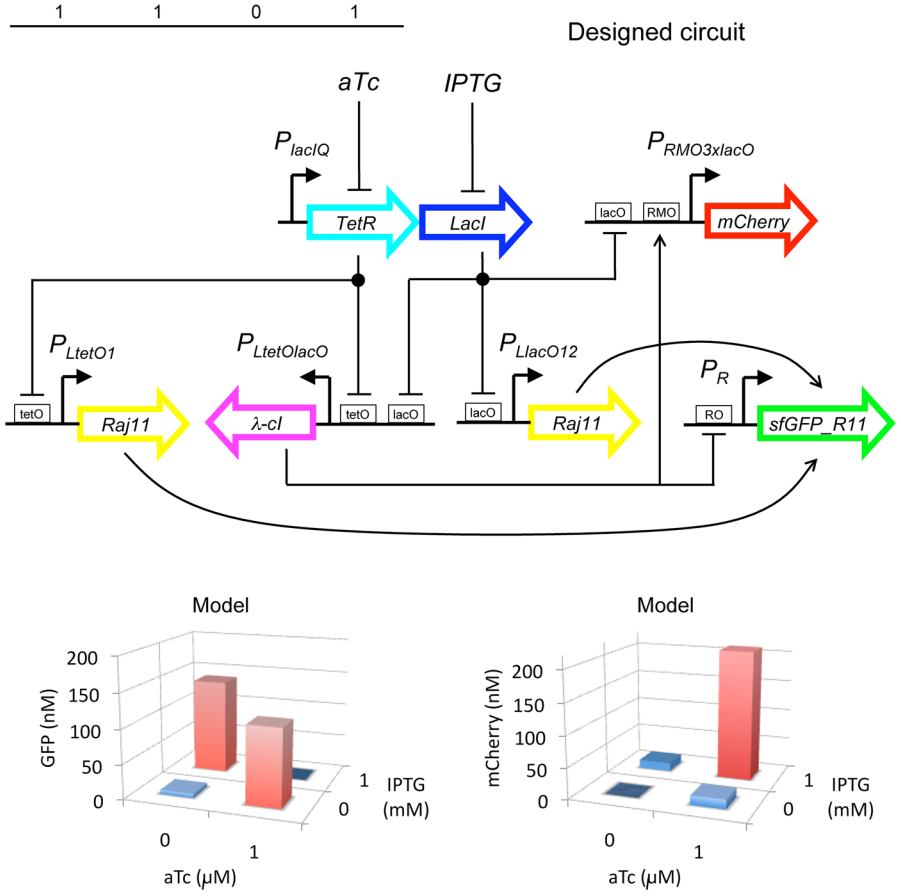


Figure 3. Computational design (by optimization) of a biological arithmetic logic unit. The circuit implements a half-adder (addition of two one-bit binary numbers, with carry). The inputs are IPTG and aTc, and the outputs are sfGFP and mCherry. sfGFP (adder) is the result of IPTG XOR aTc, while mCherry (carry) is the result of IPTG AND aTc. We also show the simulated behavior.

(where XOR and AND represent Boolean operators). Figure 3 shows the circuit we obtained after 10^5 iterations (about 1 h on a 2 GHz Intel, although suboptimal solutions can be obtained with 10^4 iterations in about 10 min). The circuit couples transcription control and riboregulation to implement such logic gates, and it is the result of an FBDA process. In this case, we did not optimize the parameters of the different part models (i.e., no degenerate parts). Without imposition of any initial circuit, we evolved a circuit by replacing, adding, or removing parts, to finally obtain a set of parts that satisfied the truth Table of a half-adder (Figure 3). The designed circuit exploits the constitutive expression of repressors LacI and TetR to couple λ-cI regulation (positive and negative) with riboregulation. Here, Raj11 is a small RNA that binds to a *cis*-repressed 5′ UTR of the GFP mRNA to activate translation by a conformational change.[7] The incorporation of functional modules into the library of models can enhance the optimization of a complex behavior such as this half-adder, taking advantage of a hierarchical design.[47] In addition, its engineering can be simplified when using a strain carrying the cassette Z1[48] or a plasmid with some parts of the circuits. By creating minimal circuits, we can assess the predictability of the assembled models, as is the case in the circuit shown in Figure 4, which we recently engineered in bacteria.[7] Although the designed circuit is composed of experimental parts already characterized and modeled,[7,11,48,49] potential interactions between the regulators may disturb the expected behavior. An experimental characterization of the whole circuit is therefore required.

Discussion. Our approach to modularity in gene circuits is explained through an analogy with molecular modeling. Most usual empirical potential energy functions (force fields) used in molecular mechanics include context effects due to electronic polarization in an average way.[50] Although the partial charges of an atom depend on its neighborhood, the use of such averaged values is able to provide accurate molecular dynamics in most cases. The kinetic properties of biological parts also depend on the genetic neighborhood.[51] For instance, the sequences upstream or downstream of a promoter can affect its maximal transcription rate. Could such dependence also be circumvented using averaged values in the kinetic parameters of biological part models? Or are we forced to always include the context? One line of thought positively answers the first question by assuming that it would be possible to engineer parts that minimize the contextual effects. In the previous

dx.doi.org/10.1021/sb300084h | *ACS Synth. Biol.* 2013, 2, 230−236

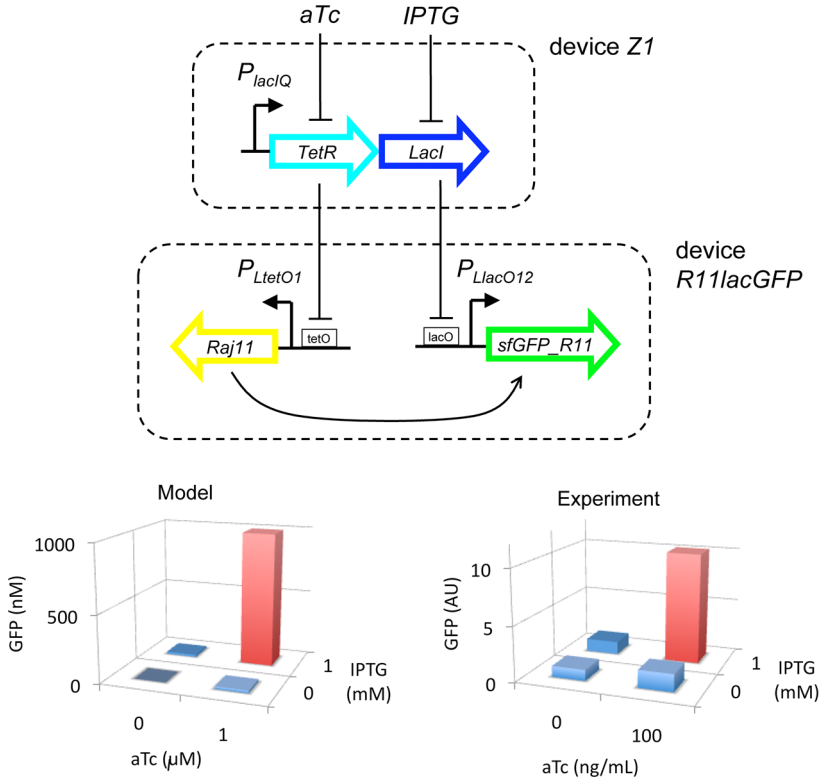


Figure 4. Assembly and simulation of a circuit coupling transcription control and riboregulation. The inputs are IPTG and aTc, and the output is sfGFP, which is the result of IPTG AND aTc. Two modules can be identified. We also show the simulated behavior against the experimental one.

example of promoters, insulator sequences were added to prevent the regulatory effects of neighboring sequences.[19] This way, the inference of biological part models can be tackled following similar procedures as those used in molecular modeling: by using a large set of circuits with characterized single-cell dynamics to compute by optimization the averaged parameters and then develop the part models that better fit the experimental data. Here, we could use AutoBioCAD for either optimizing the parameters or finding appropriate circuit topologies for such a task.

We developed a computational framework to design functional genetic networks, which can also be used to solve SBML models in deterministic or stochastic regimes. This way, our approach allows the contruction of new libraries of networks to analyze their functional diversity.[33] AutoBioCAD merged the core of evolutionary computation[26] with the automated assembly of parts[13] for network construction and optimization. Our previous genetic circuit optimizer (Genetdes), although capable of designing circuits with specified dynamical behaviors, did not use a library of part models; it was restricted to one particular mathematical model and could not run stochastic simulations. We exploited our previous network assembler (Asmparts), which did not allow optimization, to develop AutoBioCAD. Now, when working with models of experimental parts, it is able to deliver a nucleotide sequence (or a diversity of sequences) for the optimal solution, exported in GenBank format. We could also create libraries of theoretical models sampling the regulatory motifs and kinetic parameter values,[34] allowing the analysis of regulation principles in natural systems. We can define functional behaviors such as logic gating, amplitude filtering, or oscillations as design specifications. AutoBioCAD can also be applied to find particular network topologies that allow us to obtain a rapid output response under a step function as input[52] or architectures that confer robustness to input variations or to noise.[53] In addition, AutoBioCAD can be used as a reverse-engineering tool to fit the parameter values of a model of interest against experimental data. In this case, the topology of the network is fixed while we optimize the parameter space. It can be applied to study the emergence of new behaviors after specific mutations, replacements, or gene duplications. Importantly, as we provide the source code, it can be easily extended to more elaborate fitness functions or part models. Moreover, the exported SBML and GenBank files can be uploaded into other software suites[54−56] for postprocessing (Figure 5). To sum up, AutoBioCAD is an accessible and versatile computational design tool that will be helpful in systems and synthetic biology applications.

METHODS

Before running AutoBioCAD, the user has to define the target behavior in the form of dynamics for a given input value or function. For example, for an AND logic gate, four different target dynamics have to be defined. The initial network, if required, is specified as a list of sorted parts to be assembled. Details and examples are provided in Supporting Information. AutoBioCAD mainly outputs two standard files for a designed network: its mathematical model in SBML format[32] and its nucleotide sequence in GenBank format.[37]

The characterization experiment of the circuit shown in Figure 4 was performed in *E. coli* K-12 MG1655-Z1 cells (MG1655 *lacI*$^+$ *tetR*$^+$ *araC*$^+$) for control over the $P_{LlacO12}$ and P_{LtetO1} promoters.[48] The circuit was encoded in a plasmid with pMB1 origin and Kanamycin resistance (50 μg/mL). Overnight cultures were diluted 200 times in 200 μL of M9 within each well of a plate, which was assayed in an Infinite F500 multiwell fluorometer (TECAN) at 37 °C with shaking to obtain the

dx.doi.org/10.1021/sb300084h | *ACS Synth. Biol.* 2013, 2, 230−236

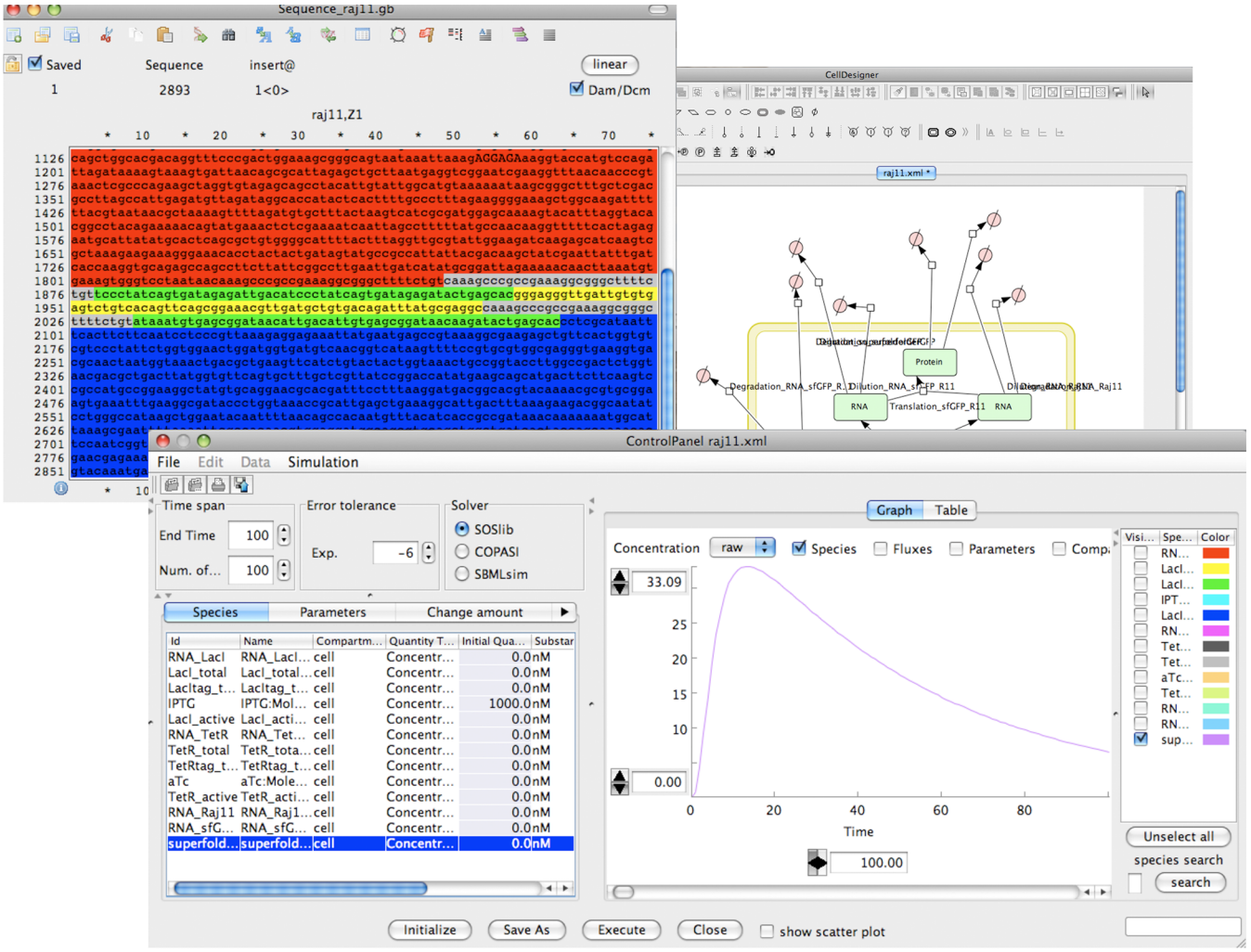


Figure 5. Visualization of a network designed with AutoBioCAD (shown in Figure 4) in CellDesigner,[54] together with its nucleotide sequence in ApE.[56]

measurements of absorbance (600 nm) and fluorescence (480 nm excitation, 530 nm emission for GFP). IPTG and aTc, when needed, were added at time 0. The absolute fluorescence was divided by OD_{600} to have a magnitude per cell, and then stationary GFP expression values were obtained by taking the maximum value of the time series, ensuring exponential growth.[7]

■ ASSOCIATED CONTENT

Supporting Information

A detailed tutorial with practical examples, information about the libraries of models constructed, and the SBML files of the networks presented in this article. This material is available free of charge via the Internet at http://pubs.acs.org.

■ AUTHOR INFORMATION

Corresponding Author

*Phone: +33 169474444. Fax: +33 169474437. E-mail: Alfonso.Jaramillo@issb.genopole.fr.

Notes

The authors declare no competing financial interest.

Downloadable Material: Software, with examples and libraries of models, for MAC OS X and Linux is open source and freely available at http://synth-bio.org.

■ ACKNOWLEDGMENTS

We thank W. Rostain for critical reading of the manuscript and J. Carrera for his contribution to AutoBioCAD. This work was supported by the FP7-ICT-043338 grant (BACTOCOM) to A.J., and G.R. acknowledges an EMBO long-term fellowship cofunded by Marie Curie actions (ALTF-1177-2011).

■ REFERENCES

(1) Chen, W-K. (2009) *Computer Aided Design and Design Automation*, CRC Press, Boca Raton, FL.

(2) Koza, J. R. (1992) *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press, Cambridge MA.

(3) Rodrigo, G., Carrera, J., Landrain, T. E., and Jaramillo, A. (2012) Perspectives on the automatic design of regulatory systems for synthetic biology. *FEBS Lett. 586*, 2037−2042.

(4) Dahiyat, B. I., and Mayo, S. L. (1997) De novo protein design: fully automated sequence selection. *Science 278*, 82−87.

(5) Kuhlman, B., et al. (2003) Design of a novel globular protein fold with atomic-level accuracy. *Science 302*, 1364−1368.

(6) Seelig, G., Soloveichik, D., Zhang, D. Y., and Winfree, E. (2006) Enzyme-free nucleic acid logic circuits. *Science 314*, 1585−1588.

(7) Rodrigo, G., Landrain, T. E., and Jaramillo, A. (2012) De novo automated design of small RNA circuits for engineering synthetic riboregulation in living cells. *Proc. Natl. Acad. Sci. U.S.A. 109*, 15271−15276.

(8) Cox, R. S., III, Surette, M. G., and Elowitz, M. B. (2007) Programming gene expression with combinatorial promoters. *Mol. Syst. Biol. 3*, 145.
(9) Canton, B., Labno, A., and Endy, D. (2008) Refinement and standardization of synthetic biological parts and devices. *Nat. Biotechnol. 26*, 787−793.
(10) Ellis, T., Wang, X., and Collins, J. J. (2009) Diversity-based, model-guided construction of synthetic gene networks with predicted functions. *Nat. Biotechnol. 27*, 465−471.
(11) Guet, C. C., Elowitz, M. B., Hsing, W., and Leibler, S. (2002) Combinatorial synthesis of genetic networks. *Science 296*, 1466−1470.
(12) Wang, B., et al. (2011) Engineering modular and orthogonal genetic logic gates for robust digital-like synthetic biology. *Nat. Commun. 2*, 508.
(13) Rodrigo, G., Carrera, J., and Jaramillo, A. (2007) Asmparts: assembly of biological model parts. *Syst. Synth. Biol. 1*, 167−170.
(14) Cai, Y., Hartnett, B., Gustafsson, C., and Peccoud, J. (2007) A syntactic model to design and verify synthetic genetic constructs derived from standard biological parts. *Bioinformatics 23*, 2760−2767.
(15) Marchisio, M. A., and Stelling, J. (2008) Computational design of synthetic gene circuits with composable parts. *Bioinformatics 24*, 1903−1910.
(16) Chandran, D., Bergmann, F. T., and Sauro, H. M. (2009) TinkerCell: modular CAD tool for synthetic biology. *J. Biol. Eng. 3*, 19.
(17) Cooling, M. T., et al. (2010) Standard virtual biological parts: a repository of modular modeling components for synthetic biology. *Bioinformatics 26*, 925−931.
(18) Bilitchenko, L., et al. (2011) Eugene: A domain specific language for specifying and constraining synthetic biological parts, devices, and systems. *PLoS One 6*, e18882.
(19) Davis, J. H., Rubin, A. J., and Sauer, R. T. (2011) Design, construction and characterization of a set of insulated bacterial promoters. *Nucleic Acids Res. 39*, 1131−1141.
(20) Yaman, F., et al. (2012) Automated selection of synthetic biology parts for genetic regulatory networks. *ACS Synth. Biol. 1*, 332−344.
(21) Wu, C.-H., Lee, H.-C., and Chen, B.-S. (2011) Robust synthetic gene network design via library-based search method. *Bioinformatics 27*, 2700−2706.
(22) Huynh, L., et al. (2012) Automatic design of synthetic gene circuits through mixed integer non-linear programming. *PLoS One 7*, e35529.
(23) François, P., and Hakim, V. (2004) Design of genetic networks with specified functions by evolution *in silico*. *Proc. Natl. Acad. Sci. U.S.A. 101*, 580−585.
(24) Kashtan, N., and Alon, U. (2005) Spontaneous evolution of modularity and network motifs. *Proc. Natl. Acad. Sci. U.S.A. 102*, 13773−13778.
(25) Paladugu, S. R., et al. (2006) *In silico* evolution of functional modules in biochemical networks. *IEE Proc. Syst. Biol. 153*, 223−235.
(26) Rodrigo, G., Carrera, J., and Jaramillo, A. (2007) Genetdes: automatic design of transcriptional networks. *Bioinformatics 23*, 1857−1858.
(27) Tagkopoulos, I., Liu, Y., and Tavazoie, S. (2008) Predictive behavior within microbial genetic networks. *Science 320*, 1313−1317.
(28) Marchisio, M. A., and Stelling, J. (2011) Automatic design of digital synthetic gene circuits. *PLoS Comput. Biol. 7*, e1001083.
(29) Dasika, M. S., and Maranas, C. D. (2008) OptCircuit: An optimization based method for computational design of genetic circuits. *BMC Syst. Biol. 2*, 24.
(30) Cao, H., et al. (2010) Evolving cell models for systems and synthetic biology. *Syst. Synth. Biol. 4*, 55−84.
(31) Beal, J., et al. (2012) An end-to-end workflow for engineering of biological networks from high-level specifications. *ACS Synth. Biol. 1*, 317−331.
(32) Hucka, M., et al. (2003) The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models. *Bioinformatics 19*, 524−531.
(33) Rodrigo, G., Carrera, J., and Jaramillo, A. (2011) Computational design of synthetic regulatory networks from a genetic library to characterize the designability of dynamical behaviors. *Nucleic Acids Res. 39*, e138.
(34) Bintu, L., et al. (2005) Transcriptional regulation by the numbers: models. *Curr. Opin. Genet. Dev. 15*, 116−124.
(35) Cookson, N. A., et al. (2011) Queueing up for enzymatic processing: correlated signaling through coupled degradation. *Mol. Syst. Biol. 7*, 561.
(36) Berman, H. M., et al. (2000) The Protein Data Bank. *Nucleic Acids Res. 28*, 235−242.
(37) Benson, D., et al. (2008) GenBank. *Nucleic Acids Res. 36*, D25−D30.
(38) Rhodius, V. A., Mutalik, V. K., and Gross, C. A. (2011) Predicting the strength of UP-elements and full-length *E. coli* sigma^E promoters. *Nucleic Acids Res. 40*, 2907−2924.
(39) Mayo, A. E., et al. (2006) Plasticity of the cis-regulatory input function of a gene. *PLoS Biol. 4*, e45.
(40) Salis, H. M., Mirsky, E. A., and Voigt, C. A. (2009) Automated design of synthetic ribosome binding sites to control protein expression. *Nat. Biotechnol. 27*, 946−950.
(41) Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. (1983) Optimization by simulated annealing. *Science 220*, 671−680.
(42) Suman, B. (2004) Study of simulated annealing based algorithms for multiobjective optimization of a constrained problem. *Comput. Chem. Eng. 28*, 1849−1871.
(43) Hindmarsh, A. C., et al. (2005) SUNDIALS: Suite of nonlinear and differential/algebraic equation solvers. *ACM Trans. Math. Softw. 31*, 363−396.
(44) Bornstein, B. J., Keating, S. M., Jouraku, A., and Hucka, M. (2008) LibSBML: An API Library for SBML. *Bioinformatics 24*, 880−881.
(45) Machné, R., et al. (2006) The SBML ODE Solver Library: a native API for symbolic and fast numerical analysis of reaction networks. *Bioinformatics 22*, 1406−1407.
(46) Ausländer, S., et al. (2012) Programmable single-cell mammalian biocomputers. *Nature 487*, 123−127.
(47) Chandran, D., and Sauro, H. M. (2012) Hierarchical modeling for synthetic biology. *ACS Synth. Biol. 1*, 353−364.
(48) Lutz, R., and Bujard, H. (1997) Independent and tight regulation of transcriptional units in *Escherichia coli* via the LacR/O, the TetR/O and AraC/I1-I2 regulatory elements. *Nucleic Acids Res. 25*, 1203−1210.
(49) Guido, N. J., et al. (2006) A bottom-up approach to gene regulation. *Nature 439*, 856−860.
(50) Ponder, J. W., and Case, D. A. (2003) Force fields for protein simulations. *Adv. Protein Chem. 66*, 27−85.
(51) Kittleson, J. T., Wu, G. C., and Anderson, J. C. (2012) Successes and failures in modular genetic engineering. *Curr. Opin. Chem. Biol. 16*, 329−336.
(52) Rosenfeld, N., Elowitz, M. B., and Alon, U. (2002) Negative autoregulation speeds the response times of transcription networks. *J. Mol. Biol. 323*, 785−793.
(53) Rodrigo, G., and Elena, S. F. (2011) Structural discrimination of robustness in transcriptional feedforward loops for pattern formation. *PLoS One 6*, e16904.
(54) Funahashi, A., Tanimura, N., Morohashi, M., and Kitano, H. (2003) CellDesigner: a process diagram editor for gene-regulatory and biochemical networks. *BIOSILICO 1*, 159−162.
(55) Sauro, H. M., et al. (2003) Next generation simulation tools: the Systems Biology Workbench and BioSPICE integration. *OMICS 7*, 355−372.
(56) http://biologylabs.utah.edu/jorgensen/wayned/ape