Computational design of digital and memory biological devices

Guillermo Rodrigo · Alfonso Jaramillo

Abstract The use of combinatorial optimization techniques with computational design allows the development of automated methods to design biological systems. Automatic design integrates design principles in an unsupervised algorithm to sample a larger region of the biological network space, at the topology and parameter levels. The design of novel synthetic transcriptional networks with targeted behaviors will be key to understand the design principles underlying biological networks. In this work, we evolve transcriptional networks towards a targeted dynamics, by using a library of promoters and coding sequences, to design a complex biological memory device. The designed sequential transcription network implements a JK-Latch, which is fully predictable and richer than other memory devices. Furthermore, we present designs of transcriptional devices behaving as logic gates, and we show how to create digital behavior from analog promoters. Our procedure allows us to propose a scenario for the evolution of multifunctional genetic networks. In addition, we discuss the decomposability of regulatory networks in terms of genetic modules to develop a given cellular function. Summary. We show how to use an automated procedure to design logic and sequential transcription circuits. This methodology will allow advancing the rational design of biological devices to more complex systems, and we propose the first design of a biological JK-latch memory device.

Keywords Computational design · Digital behavior · Synthetic biology · Systems biology · Transcriptional networks

Introduction

In the last years, there has been a tremendous work on inferring the topologies of transcription networks (Babu and Teichmann 2003) with the hope of elucidating their design principles. Studies of recurrent network motifs showed that their dynamics could provide useful functions (Mangan and Alon 2003; Brandman et al. 2005). These reverse-engineering studies have uncovered those principles and they have provided the framework to plan the design of synthetic circuits. The exponential decrease of DNA synthesis cost, together with the new development of standardized genetic parts (Endy 2005), will provide the technological means to design large genetic networks. The easiest way to construct such genetic parts is to use the available genomic knowledge, although there has also been some work on creating synthetic parts such as promoters with modified or added operator sites (Buchler et al. 2003; Bintu et al. 2005a, b), modified ribosome binding sites (Basu et al. 2004), or codon-optimized coding regions (Basu et al. 2005). Protein engineering has also been providing new coding sequences for several decades now, corresponding to new designed proteins with specified functions that sometimes have even no similarity with any known natural protein sequence (Kuhlman et al. 2003; Looger et al. 2003; Jaramillo et al. 2002).
The construction of transcriptional networks could also take advantage from the design of promoter sequences having a combinatorial regulation. In fact, there are already some examples on the design of synthetic promoters regulated by two transcriptional factors (Joung et al. 1994; Mayo et al. 2006). Currently, most of the designed transcriptional networks (Elowitz and Leibler 2000; Gardner et al. 2000; Atkinson et al. 2003) have been engineered using rational design techniques (Hasty et al. 2002), limiting the use of computational modeling to the solving of dynamical equations. The design of such networks would highly benefit from the use of computational methods to automatically design new circuits. For instance, we could evolve our circuits by replacing elementary parts, such as promoters, from a catalogue of these. Furthermore, the design of genetic circuits from modular components allows designing a panoply of transcriptional circuits from a few elementary components. This is exemplified by the combinatorial generation of 125 transcriptional networks, some of them working as logical devices, using only a library of five promoters controlling three transcriptional factors (Guet et al. 2002). They characterized the behavior of their three-gene networks under two inducers and discovered that some of the networks behaved like a logic gate of type NAND, NOR or NOT. Sometimes transcriptional networks having the same topology would give completely different behavior. Each promoter only contained one operator site, so it could only receive one transcription factor. Their results suggested that a given circuit topology could show several behaviors. In this work we perform a computational evolution that will allow us to generate similar transcriptional circuits to those found in that previous work (Guet et al. 2002). An important question in understanding the mechanism of living systems is the design principle of regulatory networks having a given behavior. Analyzing our functional genetic networks, designed following a combinatorial optimization procedure, we can unveil new of such principles.

The combination of computational and combinatorial techniques has already been used in other disciplines to generate complex behaviors. In particular, an essential aspect of networks is the evolution and design or their dynamical behavior. Work on Artificial Life or on hardware evolution (Taubes 1997; Raichman et al. 2003) shows how automated methods could be used to design emergent complexity (Kahtan and Alon 2005). In biology, an evolutionary method to engineer electronic circuits was suggested for the design of genetic circuits with oscillatory behavior (Mason et al. 2004), although the used model was purely Boolean. A more accurate model introduced protein species together with biochemical reactions, and allowed to design networks with a specified function (Francois and Hakim 2004). They obtained switches and oscillators mostly based on protein–protein interactions but without external inputs to control its behavior. Recently, it has been proposed a methodology to evolve circuits with a targeted asymptotic behavior by computing the steady state of the system (Paladugu et al. 2006). Here, we propose to go beyond those results by considering a general time response and addressing transcription regulation to implement genetic networks that could be eventually synthesized. In addition, our networks can be controlled by external signals.

An interesting step towards the design of complex biological circuits is the ability to design networks with digital behavior, where the output gene concentrations could be described as Boolean functions of the inputs. Moreover, genetic circuits sharing the same topology could behave very differently depending on their kinetic parameters. Therefore, rational design techniques that only take into account network topology considerations are not sufficient to ensure a given dynamics. This would force us to enumerate all possible parameter combinations to design gene networks with a targeted behavior, which is impossible due to the exponential explosion of possible models. To address this issue, we have developed an algorithm that searches the space of artificial transcriptional networks to find the optimal circuits with a targeted temporal behavior. During our simulation, we add or subtract genes, change kinetic constants or the combinatorial regulation logic at promoters. We evolve in time each generated circuit and we use the average deviation to an expected temporal function as a scoring function. We use Monte Carlo Simulated Annealing (Kirkpatrick et al. 1983; Metropolis et al. 1953) method to do an optimization in the space of all possible genetic circuits.

A toggle switch was presented in (Gardner et al. 2000) by implementing a two-gene core repressor–repressor. In fact, this topology implements a RS-Latch. A further step was designing a D-Latch based on the previous circuit with a rational approach (Fritz et al. 2007). In this work, we propose to go beyond those results by designing a more complex memory device: a JK-Latch. This device is richer than a D-Latch because it has two inputs to control its behavior, and it is more predictable than a RS-Latch because all the entries of the truth table are defined. In addition, we have applied our methodology to design some motifs by targeting digital behaviors such as logic gates in order to compare our results with the ones found in Nature.

For simplicity, we have focused on the design of transcription networks, neglecting post-translation regulation. Already those networks can generate an arbitrary complex behavior, as they have recently been suggested to form a programmable computing machine, belonging to the class of Boltzmann machines (Foster 2001). Importantly, one inconvenient of those networks is their slow response,
making difficult to use cascades of many stages (Hooshangi et al. 2005), as it is usually done when designing electrical circuits. Therefore, the implementation of a transcription logic in the promoter region will play an important role in generating complex behavior.

Methods

Mathematical model

Multiple approaches are available to describe biological networks, particularly transcriptional circuits. We consider for this work a deterministic and continuous model to describe the regulatory interactions between genes. Further works will take into account stochastic effects on the system. A whole description of these networks considers all the species involved in the processes of transcription, translation, and regulation, such as DNA, mRNA and proteins (see Fig. 1). In general, a regulation between a transcription factor (repressor or activator) and a gene can be modeled according the following differential equations

\[
\frac{d}{dt}[V] = \rho[U]^n - \sigma[V],
\]

\[
\frac{d}{dt}[D] = \mu[V : D] - \theta[V][D],
\]

\[
\frac{d}{dt}[R] = \varphi[D] + \psi[V : D] - \lambda[R] - \delta[R],
\]

\[
\frac{d}{dt}[Y] = \beta[R] - \beta[Y],
\]

where \( U \) is the non-active form of the regulator and \( V \) its active form (i.e., transcription factor). \( \rho \) and \( \sigma \) are the kinetic coefficients for \( n \)-merization. If a regulator does not \( n \)-merize, this description is also valid with \( U = V, n = 1, \rho = \sigma \). \( D \) is the DNA, \( R \) the mRNA, and \( Y \) the folded protein from that regulated gene. \( \theta \) and \( \mu \) are the kinetic coefficients for binding and unbinding between the transcription factor and the promoter, respectively. \( \varphi \) and \( \psi \) are the transcriptional kinetics from free and occupied DNA, respectively. We assume that DNA is not degraded, mRNA does it with kinetics \( \delta \), and protein with \( \beta \). \( \lambda \) is the translational kinetics. We have considered that the cellular resources, such as RNA-Polymerases, Ribosomes, nucleotides and amino-acids, are sufficient to sustain a synthetic system and could be assumed constant. Thus, the transcription and translation rates only depend on the amounts of DNA and mRNA, respectively. We add an additional mass balance equation,

\[
[D_i] = [D] + [V : D],
\]

as the amount of DNA within the cell is conserved. Those equations can be rewritten for more than one transcription factor.

However, we simplify the model to get a more reduced formulation, despite of introducing more non-linearity. We suppose that the intermediate species have low temporal rates (i.e., \( d[V]/dt = d[D]/dt = d[R]/dt = 0 \)), and we define the following parameters

\[
K_\alpha = \frac{\mu \sigma}{\alpha},
\]

\[
a = \frac{\lambda}{\delta + \lambda} \varphi[D_i],
\]

\[
c = \frac{\lambda}{\delta + \lambda} \psi[D_i].
\]

Thus, we can write the following effective differential equation for the protein dynamics

\[
\frac{d}{dt}[Y] = a \left( \frac{[V]}{K} \right)^\alpha + c \left( \frac{[V]}{K} \right)^\alpha - \beta[Y],
\]

where \( a \) and \( c \) are the rate of protein synthesis at low and high \( U \) concentrations, respectively. Therefore, the temporal evolution of a transcription factor concentration \( [(Y_i)] \) of the network can be written as

\[
\frac{d}{dt}[Y_i] = \alpha_i R_i - \beta_i[Y_i] + \gamma_i,
\]

where \( \alpha_i \) is the synthesis (transcription–translation) rate of gene \( i \), \( \beta_i \) its degradation rate, and \( \gamma_i \) its basal rate. The function \( K_i \) is the regulatory factor that contains the information about the regulation of a given promoter by other transcription factors (see Table 1).

We have chosen the following expression for the regulatory factor of gene \( i \):

\[
R_i = c_{00} + c_{10} R_{IA} + c_{01} R_{IB} + c_{11} R_{IA} R_{IB},
\]

where \( c_{pq} \) are a set of coefficients defined to match a given asymptotic behavior (see Table 1) and \( R_{IA} \) and \( R_{IB} \) are given by

\[
\text{Fig. 1 Biological scheme of a transcriptional regulation. Greek letters denote the kinetic constants of the different biochemical reactions}
\[ R_{iA} = \frac{1}{1 + \frac{[A]}{K_{iA}}}^{n_{iA}}, \]  

in case of having a protein that represses transcription and by

\[ R_{iA}^e = \frac{Y_i}{K_{iA}}^{n_{iA}}, \]  

in case of activating transcription. \( K_{iA} \) is the activation or repression coefficient and \( n_{iA} \) is a Hill coefficient. In Table 2 we show the different ranges for the parameter values and their units used this work. We have only considered the \( c_{pq} \) corresponding to the choice of promoters in Table 1, although we will later show the use of another choice of \( c_{pq} \). Those promoters asymptotically implement digital behaviors. The parameters \( c_{pq} \) could also be fitted against experimental data. In addition, we could also have considered promoters regulated by more than two transcription factors.

In addition, we choose to design transcriptional networks composed of promoters regulated by less than three transcriptional factors. We can find promoters with synergistic regulation by two transcriptional factors in \( E. coli \) and in \( S. cerevisiae \): in \( E. coli \), the promoter controlling gene \( fucAO \) is synergistically activated by \( CRP \) and \( fucPIKUR \) (Bronsted and Atlung 1996), the promoter controlling gene \( cirA \) is activated by \( CRP \) and repressed by \( fur \) (Griggs et al. 1990), and the transcription factors \( fnr \) and \( arcA \) repress gene \( sdhCDBA \) (Park et al. 1997). On the other hand, in yeast, the promoter controlling gene \( PDR5 \) is activated by \( PDR1 \) and \( PDR3 \) (Katzmann et al. 1994), and the promoter controlling gene \( GAL1 \) is activated by \( GAL4 \) and repressed by \( MIG1 \) (Nehlin et al. 1991), and the transcription factors \( MOT3 \) and \( ROX1 \) repress gene \( DAN1 \) (Sertil et al. 2003).

### Optimization procedure

We have developed a software to computationally design transcriptional networks with a target behavior (Rodrigo et al. 2007). We design circuits with a targeted dynamics by solving an optimization problem where we generate different circuits that are scored according to a distance from the expected time-behavior. We start from a circuit of random topology and random parameters. We use Simulated Annealing (Kirkpatrick et al. 1983) as an optimization procedure, which is implemented through the use of Monte Carlo algorithm with Metropolis updating and an exponential cooling schedule (Metropolis et al. 1953).

We randomly consider two different types of moves during a Monte Carlo step. In the first type of move, we randomly modify a parameter in the network model by a constant amount (positive or negative). For each parameter, the shift depends on its current value. For simplicity we did not consider the possibility of having different transition probabilities for every parameter. Moreover, this shift is adjusted to have an acceptance probability of 50%, which improves the convergence. For the second type of moves we modify the network topology by randomly choosing among

### Table 1

| TYPE | REGULATORY LOGIC | REGULATORY FUNCTION | PROMOTOR WITH OPERATORS |
|------|------------------|---------------------|-------------------------|
| 0    | -                | 1                   | CONSTITUTIVE            |
| 1    | NOT              | \( R_A^w \)          | 1                       |
| 2    | YES              | \( R_A^w \)          | 1                       |
| 3    | NOR              | \( R_A^w R_B^w \)    | 1                       |
| 4    | XNOR             | \( 1-R_A^w R_B^w + 2R_A^w R_B^w \) | 1 |
| 5    | NAND             | \( R_A^w R_B^w \)    | 1                       |
| 6    | AND              | \( R_A^w R_B^w \)    | 1                       |
| 7    | OR               | \( R_A^w R_B^w \)    | 1                       |
| 8    | XOR              | \( R_A^w R_B^w \)    | 1                       |
| 9    | NNOR             | \( R_A^w R_B^w \)    | 1                       |

The first column shows the promoter’s label together with its name. The second column shows the standard logic symbol for the transcriptional activation as a function of the input concentrations of transcription factors \( A \) and \( B \). The third column shows the corresponding value of the regulatory function \( R_i \) used in Eq. 10. The fourth column schematically shows the operator sites where the repressors (black boxes) or activators (white boxes) bind to the promoter. Bottom: Possible promoter transitions during the optimization procedure.

### Table 2

| Parameter | Units | Max value | Min value |
|-----------|-------|-----------|-----------|
| \( x \)  | \( \mu \text{M/min} \) | 10 | 0.1 |
| \( \beta \) | \( 1/\text{min} \) | 2 | 0.001 |
| \( \gamma \) | \( \mu \text{M/min} \) | 0.01 | 0.0002 |
| \( K \)  | \( \mu \text{M} \) | 5 | 0.01 |
| \( n \)  | – | 3 | 1 |
five possibilities: (i) to replace a promoter (which amounts to replacing the regulatory function in Eq. 11), (ii) to add a new regulation between a gene and a promoter, (iii) to remove a regulation on a given promoter, (iv) to add a new gene and (v) to remove a gene. In all moves, we randomly choose a gene and perform the corresponding modification.

To improve the convergence, in move (i) not all transitions are possible (see bottom of Table 1). For simplicity, in this work we do not consider promoters regulated by more than two transcription factors. In move (iii) we select randomly a regulation of the circuit to be suppressed. To avoid isolated genes, we also remove genes that do not take part of the circuit after this move. In move (iv) we also add a new regulation between the new gene and another (selected randomly) in the previous circuit. Notice that by adding genes we increase the circuit’s ability to reproduce a given behavior, which will produce an evolution towards large networks. One way to limit the corresponding explosion in network size is to either add a size penalty in the scoring function or reduce the probability to add a new gene. If a gene does not regulate any promoter, we remove it because it does not affect the behavior of the circuit. That gene could be regulated or not. In addition, for convergence purposes we take a much higher probability to do a parameter move (0.99 by default using our algorithm). Like that, for each generated circuit during the optimization, we can explore the possible behaviors according to the fixed range of the parameter values.

For each constructed genetic network, we solve the dynamics for the protein concentrations and then we compute a scoring function. If \( z(t) \) gives the targeted behavior and \( y(t) \) is the dynamics of the corresponding circuit, we define as score

\[
\Phi = \int_{t_i}^{t_f} |y - z| \chi dt,
\]

where \( \chi \) is a weighting factor that takes 0 or 1 values to only score a region of interest.

**Results**

**Design of a memory device**

We have applied our methodology to design a sequential circuit implemented as a transcriptional network. Such network is able to store information resulting in a genetic memory. We have targeted a JK-Latch (Vingron 2003) which is the preferred type of flip-flop for most logic circuit designs because its behavior is very rich and completely predictable under all conditions. For this circuit, there are two inputs (J and K), which are the setters of each stable state. When both inputs are activated for a brief time (producing a pulse) the system switches of state. In fact, if this situation is prolonged the system could behave as an oscillator. Essentially, this circuit works as the toggle switch, already implemented with transcription factors (Gardner et al. 2000), which could be seen as a RS-Latch, except for the case in which both inputs are present.

We have designed two biological implementations of JK-Latches using transcription factors. The first one is guided by the electronic implementation (see Fig. 2a), where we have used as logic gates promoters with combinatorial regulation (see Fig. 2b). This rational approach is applied to engineer a JK-Latch by implementing on a RS-Latch two positive feedbacks from the outputs of the system (\( A \) and \( B \)). In that case, like in the electronic design, these feedbacks are integrated in two synergistically activated promoters. Unfortunately, it is not always possible to perform such rational designs, and in this article we aim to show a new technique to create alternative designs. As described in the previous section, we propose to use an unsupervised computational design technique. We have imposed the desired specifications (as shown in Fig. 3a) to design a novel circuit (see Fig. 3b) that has no resemblance to the previous rational design (see Fig. 2b). We have constructed \( \Phi \) by adding the eight scores constructed by using the four possible entries of the truth table shown in Fig. 3a for each of the two possible initial conditions (\( A = 1, \ B = 0 \) \( \mu \)M and \( A = 0, \ B = 1 \) \( \mu \)M).
We provide as supporting information the corresponding SBML (Hucka et al. 2003) file containing the parameter values of the designed circuit. Our design was obtained by an in silico evolution optimizing both the topology and parameters of the network, without relying on an analogy between biology and electronics.

The work principle of this circuit relies on the fact that there are two stable states and the behavior fluctuates as a flip-flop from one to another according to the external inputs. In the first state (or state A), protein A is expressed activating D, which is in charge of repressing B and C (see Fig. 3c). In the second state (or state B), protein B is expressed repressing A (see Fig. 3d). In this case, C and D remain inactivated. According to the specifications of this circuit, the system remains in its current state when no inputs are present. When K is present the system is set to the state A (i.e., no changes are observed if the system is already in that state, and a flip-flop is produced if the system is in the state B). On the other hand, the transcription factor J is the setter for the state B. A switch of state is always produced when the two inputs pulse simultaneously. In addition, our system has the property to oscillate when both inputs are continuously present. We can see this dynamics in Fig. 4 in which we show different input conditions during a simulation of 1,000 min.

Conditional systems show hysteresis effects. We have computed the steady state of A after a pulse of K for several constant values of J. In Fig. 5a, b we show the hysteresis diagrams for the rationally and computationally designed JK-Latches, respectively. At low levels of J, the system behaves as a memory device. However, for J = 0.5 the system is not appropriate as a memory device because it relaxes back to its previous state when the input disappears. In our computational design, when J = 0 the switch point occurs at K ≈ 0.7 μM. For higher values of J, this point increases and the system loses its ability to store information when we consider a constant J. For J close to 1 μM, the switching dynamics disappears. Similar results are obtained based on the rational design. In this case, for J = 0 the switch point is K ≈ 0.45 μM, and a constant leakage of J avoids reaching A = 1 μM (e.g., A = 0.25 μM for J = 0.5 μM) but still allows the memory function.

Also, we have performed a stochastic simulation (Gillespie 1977) of this circuit to see its robustness under molecular noise (see Fig. 6). We have simulated several input conditions from different initial conditions. To perform such simulation we define a set of effective reactions involving the whole process of transcription and translation. We consider two possible reactions: protein synthesis and protein degradation, neglecting the fluctuations due to the mRNA dynamics. Hence, their fluxes are Hill functions of the transcription factors. We provide a MatLab file to perform this simulation as supporting information. We can see how the average behavior of the system is maintained as in Fig. 4. Further simulations could be performed, by detailing the reaction map and proposing typical parameter values to complete it, to obtain more accurate stochastic dynamics as the molecular noise can induce errors in the dynamics (Kepler and Elston 2001).

Design of digital devices

We have applied our methodology to design genetic devices behaving as logic gates. Our devices consist on
genetic circuits having the concentration of two and one transcription factors as input and output, respectively. We have targeted AND, OR, NAND and NOR gates, and in Fig. 7 we show the resulting circuits. To compute the objective function we have averaged the score obtained with each transfer function corresponding to every entry of the truth table. $u_1$ and $u_2$ are the input transcription factors and $y$ is the output corresponding to the concentration of gene $a$’s product. We have evaluated the score during 100 min. However, to better appreciate the behavior under different input conditions, we have chosen to plot a temporal dynamics where the input transcription factors concentrations $u_1$ and $u_2$ take all possible Boolean values of a two-input truth table. Inputs can be activators or repressors according to the chosen promoter during the simulation. In the inset of Fig. 7 we also show the equivalent digital circuits according to the interaction of each transcription factor with its corresponding promoters. We provide the parameters of those circuits in the supporting information.

For the AND device (see Fig. 7, circuit I), the algorithm selected inducible promoters at the input genes, which implies that the $u_1$ and $u_2$ would correspond to the concentrations of two activator transcription factors. At $t = 200$ min we appreciate a little peak due to the simultaneous change of the inputs, which is not unexpected as the device was evolved using the scores of steady states. This device could be easily understood in terms of its regulations. Initially the output gene $a$ is off as it is controlled by a synergistic AND promoter (type 6 in Table 1) that requires $b$ and $c$ to be at high concentrations to start transcription. When the two inputs increase their concentration $b$ and $c$ start to build up and then $a$ is produced. If either $u_1$ or $u_2$ are at low concentrations then $b$ or $c$ cannot be produced in enough quantities to induce the transcription of $a$. For the OR circuit (see Fig. 7, circuit II) we notice that the output almost fits the targeted behavior, being insensitive to the input changes at $t = 200$ min and at $t = 300$ min. This device has more latency when the output concentration shifts down because there is a self-activation on gene $a$. For the NAND network (see Fig. 7, circuit III), we obtain two repressors as inputs and the circuit has symmetry. In the NOR circuit (see Fig. 7, circuit IV) we have got a repressor and an activator as inputs. The latencies at $t = 0$ min and at $t = 400$ min are different. In the first case, the system starts from the initial condition and follows a spontaneous evolution with no inputs actions. However, the state of the system with the two inputs takes more time to grow up when their effects stop. We could reduce this latency, but compromising the steady state value. Actually, we have seen that we could get gene networks with small latencies as a result of our optimization, although sometimes we could not decrease the latency (circuits II and IV in Fig. 7) without compromising the score. In fact, we could lower the latency of circuit IV when we decreased the activation coefficient corresponding to

![Fig. 4 Dynamics for our biological memory device implementing a JK-Latch. In (a) dynamics of $A$, in (b) dynamics of $B$, in (c) dynamics of $J$, and in (d) dynamics of $K$.](image)
the regulation of gene $d$ by gene $c$ ($K_{dc}$), but the height of maximum output expression also decreased.

Sensitivity analysis

To study the robustness of our designed memory devices, we have performed a sensitivity analysis for the transcription ($\alpha$) and degradation ($\beta$) rates. For that, we plot the relative score variation ($\left(\Phi - \Phi_0\right)/\Phi_0$) versus the relative parameter value ($\alpha/\alpha_0$ and $\beta/\beta_0$) for the different circuit genes. In Fig. 8a, b we show the robustness of the computationally designed JK-Latch under perturbations on $C$ and $D$ at the transcription and degradation level. Oppositely, in Fig. 8c, d we show the robustness of the rationally designed JK-Latch under perturbations on $R$ and $S$ at the transcription level and on $R$ at the degradation level. We note that as our designs were optimized by Simulated Annealing, they are located in local minima being robust in a neighborhood of the optimized value.

On the other hand, as it is difficult to map a given network topology to a function, we have studied the parameter sensitivities for the circuit II from Fig. 7. Figure 9 allows us to show this fact by performing a functional evolution just by changing one kinetic parameter. In Fig. 9a we show how varying the transcription–translation rate of gene $c$ we could modify the circuit behavior between OR and AND.

In this case, we observe that the OR state is more robust. In Fig. 9b, the evolution is between OR and NOR playing with the self-activation coefficient of gene $a$. In this case, we observe that the NOR state is more robust.
We could also perform a functional evolution between AND and NOR by modifying the two considered parameters at the same time. Notice that we could then engineer an AND function from the OR device by simply doing directed mutagenesis on the ribosome binding site of gene $c$, decreasing then its expression rate. Or engineer an NOR function by decreasing the transcription factor affinity of $a$ on its promoter region.

We could understand the working principle of our computational JK-Latch by considering the relevant subcircuits. Firstly, we find a toggle switch between $D$ and $B$ (as $B$ represses $D$ by means of $A$), which will become the core of our biological memory. Nevertheless, the ability to switch when both inputs are present, but independently of the previous state, requires a more complex topology. In fact, the oscillatory behavior when both inputs are present is a non-trivial consequence of such specification, since we targeted a switch that changed state when pulsing with both inputs at the same time. This behavior can be reached in a small region of the parameter space though. In addition, we note that, instead of using two positive feedbacks as the rational design proposes, our design relies on the feedback of $A$ on $D$. Furthermore, our circuit just requires the use of three two-regulated promoters, instead of four as the rational design proposes, which is convenient for the experimental implementation.

However, the in vivo implementation of a circuit which has been designed in silico usually requires many experiments in order to tune the desired behavior (Weiss 2001) and to avoid possible cross talks (Buchler et al. 2003).
Moreover, a constant fine-tuning of the model by using experimental feedbacks will introduce more accuracy even with in vitro environments (Kim et al. 2006). It could be possible to incorporate this experimental feedback into our automatic design procedure by developing a new algorithm that would design circuits by assembling models of biological parts (Rodrigo et al. 2008). This will require a repository of already characterized synthetic or natural parts. Fortunately, recent combinatorial work has shown the easiness of creating large amounts of promoter parts by generating a set of promoters together with their experimental characterization (Cox III et al. 2007). Our computational design of a JK-Latch could be implemented by using araC as $K$, and $\lambda$-cI as $J$ for the external controlling signals. Subsequently, luxR-luxI could work as $A$, using luxP$_{\mu}$cI-O$_{\mu}$1 as promoter of $D$ (Basu et al. 2004), and $\lambda$-cro as $B$ with the $P_{RM}$ as promoter of $A$ (Hasty et al. 2002). Finally, lacI could be $C$ and tetR be $D$, implementing the lacI and tetR operator sites on the promoter region of $B$, and the araC and tetR sites on the promoter of $C$ (Cox III et al. 2007). Nevertheless, the use of such biological parts will require experimental fine-tuning to obtain the specified behavior. In fact, previous works (Weiss 2001) needed to use site-directed mutagenesis in the ribosome binding sites or in the operator sites to evolve the whole system. In addition, directed evolution techniques can evolve the behavior of the system by fine-tuning the kinetic parameters (Yokobayashi et al. 2002; Nomura and Yokobayashi 2007).

Could natural genetic networks be understood as systems of devices?

Could natural circuits have a selective pressure for a given network motif or for a given dynamical function? We have found several circuits with targeted AND behavior. If a selective pressure existed for a given network module behavior, then some circuits within a module would get rewired by evolution while maintaining their functionality. For instance, it could be that some AND circuits would occasionally appear in evolution substituted by another AND circuit. We notice that several of our results in Fig. 7 showing digital behaviors appeared very often in natural gene networks. It is interesting that two of them ($I$ and $V$ in Fig. 7b) were found very often in *E. coli* and *S. cerevisiae* as network motifs. Circuit $I$ was found 28 and 26 times in
and S. cerevisiae, respectively, as a coherent feed-forward loop (FFL) system (Mangan and Alon 2003) where it was suggested this Circuit could function as an AND gate able to filter out pulses in the input. For instance, the topology of circuit I appears in the genes (corresponding to our genes a, b and c) appCBA, appY and arcA in E. coli, and in the genes PDR5, PDR3, and PDR1 of S. cerevisiae. Likewise the topology of the circuit V is found in the genes CYC7, ROX1 and TUP1 in S. cerevisiae, but it is not present in E. coli (Mangan and Alon 2003).

On the other hand, it could happen that natural gene circuits would not rely on functional modules (Hartwell et al. 1999) but on a complex intertwined network of interactions, as it would usually happen with evolutionary design (Bosl and Li 2005). In this later case, maybe the only way to design a system of devices would be by using an evolutionary design procedure. We could then use directed evolution of gene circuits (Yokobayashi et al. 2002) or in a combination with a computational procedure. More work is needed to elucidate this point.

Functional evolution of genetic networks

Our optimization procedure also allows us to analyze the evolvability of genetic devices. Here we consider the circuit II of Fig. 7, where we have modified a single parameter, $z$, in order to change the device function to behave as an AND gate (see Fig. 9a). This parameter was related to the expression of gene $c$ and this sensitivity provides an evolvable mechanism for device function, as mutations affecting its expression will have a dramatic effect in the device function. Therefore, the transcription–translation efficiency of that gene would be under a strong selection pressure, which could be the general rule in biology, according to the experimental evidence from recent work where protein expression levels were found to be shaped by evolution (Dekel and Alon 2005). Moreover, by changing a few parameters we could get a NOR gate as well, inverting completely the initial behavior (see Fig. 9b).

We could propose a scenario for the evolution of device function by considering the situation $d$ in Fig. 9. This circuit shows an intermediate behavior between AND and OR digital devices (when $z$ is around 0.15 $\mu$M/min), but rapidly converged to OR (when $z$ is higher) or AND (when $z$ is lower). Then, if we consider the case where the input concentrations always remain below a threshold (such as 0.7 $\mu$M) then the device would work as a perfect AND gate. On the other hand, if the concentrations were always above another threshold (such as 0.5 $\mu$M) then the device would work as a perfect OR gate. Hence, by shifting the expression level of the input transcriptions factors, either
dynamically or by evolution (Dekel and Alon 2005), we could change the function of the device while maintaining all the circuit properties. This promiscuity in system function could play a mayor role in the evolution of biological systems. In fact, a device could simultaneously have several functions and a change in the environment could switch between them. In this way, if during evolution the environment would not change enough then the alternative function could be lost.

Further applications

Our methodology could also be adapted to the inference of regulatory networks provided a specific device function is known. Another application is to train parameters for given regulatory networks provided a specific device function is. Our methodology could also be adapted to the inference of logical systems. In fact, a device could simultaneously have several functions and a change in the environment could switch between them. In this way, if during evolution the environment would not change enough then the alternative function could be lost.

Acknowledgments

We thank J. Carrera and O. Martin for fruitful discussions. G.R. acknowledges the excelencia fellowship from the Universitat Politècnica de València and the FPI fellowship from Generalitat Valenciana. A.J. acknowledges support from the Universitat Politècnica de València invited prestigious researcher award in 2006 and from the EU grant FP6-NEST BiomodularH2 (contract num. 043340).

Open Access

This article is distributed under the terms of the Creative Commons Attribution NonCommercial License which permits any noncommercial use, distribution, and reproduction in any medium, provided the original author(s) and source are credited.

References

Atkinson MR, Savageau MA, Myers JT, Ninfa AJ (2003) Development of genetic circuit exhibiting toggle switch or oscillatory behavior in Escherichia coli. Cell 113:597–607.

Babu MM, Teichmann SA (2003) Evolution of transcription factors and the gene regulatory network in Escherichia coli. Nucleic Acids Res 31:1234–1244.

Basu S, Mehrjua R, Thibeurge S, Cheng MT, Weiss R (2004) Spatiotemporal control of gene expression with pulse-generating networks. Proc Natl Acad Sci USA 101:6355–6360.

Basu S, Gerchman Y, Collins CH, Arnold FH, Weiss R (2005) A synthetic multicellular system for programmed pattern formation. Nature 434:1130–1134.

Bintu L, Buchler NE, Garcia HG, Gerland U, Hwa T, Kondve J, Phillips R (2005a) Transcriptional regulation by the numbers: models. Curr Opin Genet Dev 15:116–124.

Bintu, Buchler NE, Garcia HG, Gerland U, Hwa T, Kondve J, Kulman T, Phillips R (2005b) Transcriptional regulation by the numbers: applications. Curr Opin Genet Dev 15:125–135.

Bodis WJ, Li R (2005) Mitotic–exit control as an evolved complex system. Cell 121:325–333.

Brandman O, Ferrell JE Jr, Li R, Meyer T (2005) Interlinked fast and slow positive feedback loops drive reliable cell decisions. Science 310:496–498.

Bronskill L, Altshuler T (1996) Effect of growth conditions on expression of the acid phosphatase (cxy-appA) operon and the appY gene, which encodes a transcriptional activator of Escherichia coli. J Bacteriol 178:1556–1564.

Buchler NE, Gerland U, Hwa T (2003) On schemes of combinatorial transcription logic. Proc Natl Acad Sci USA 100:5136–5141.

Cox III RS, Surette MG, Elowitz MB (2007) Programming gene expression with combinatorial promoters. Mol Syst Biol 3:145.

Dekel E, Alon U (2005) Optimality and evolutionary tuning of the expression level of a protein. Nature 436:588–592.

Elowitz MB, Levine S (2000) A synthetic oscillatory network of transcriptional regulators. Nature 403:335–338.

El-Samad H, Kurita H, Doyle JC, Gross CA, Khammash M (2005) Surviving heat shock: Control strategies for robustness and performance. Proc Natl Acad Sci USA 102:2736–2741.

Endy D (2005) Foundations for engineering biology. Nature 438:449–453.

Feng XI, Hooshanghi S, Chen D, Li G, Weiss R, Rabitz H (2004) Optimizing genetic circuits by global sensitivity analysis. Biophys J 87:2195–2202.

Foster JA (2001) Evolutionary computation. Nat Rev Genet 2:428–436.

Francois P, Hakim V (2004) Design of genetic networks with specified functions by evolution in silico. Proc Natl Acad Sci USA 101:580–585.

Fritz G, Buchler NE, Hwa T, Gerland U (2007) Designing sequential transcription logic: a simple genetic circuit for conditional memory. Syst Synth Biol. doi: 10.1007/s11699-007-9006-8.

Gardner TS, Cantor CR, Collins JJ (2000) Construction of a genetic toggle switch in E. coli. Nature 403:339–342.

Gillespie DT (1977) Exact stochastic simulation of coupled chemical reactions. J Phys Chem 81:2340–2361.

Griggs DW, Kafka K, Nau CD, Konisky J (1990) Activation of expression of the Escherichia coli ccr gene by an iron-independent regulatory mechanism involving cyclic AMP-cyclic AMP receptor protein complex. J Bacteriol 172:3529–3533.

Guet CC, Elowitz MB, Hsing W, Leibler S (2002) Combinatorial synthesis of genetic networks. Science 296:1466–1470.

Hartwell LH, Hopfield JJ, Leibler S, Murray AW (1999) From molecular to modular cell biology. Nature 402:C47–C52.

Hasty J, McMillen D, Collins JJ (2002) Engineered gene circuits. Nature 420:224–230.

Hooshanghi S, Thibeurge S, Weiss R (2005) Ultraviolet sensitivity and noise propagation in a synthetic transcriptional cascade. Proc Natl Acad Sci USA 102:3581–3586.

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.

Jaramillo A, Wernisch L, Herr S, Wodak SJ (2002) Folding free energy function selects native-like protein sequences in the core but not on the surface. Proc Natl Acad Sci USA 99:13554–13559.

Joung JK, Koepp DM, Hochschild A (1994) Synergistic activation of transcription by bacteriophage λ cl protein and E. coli cAMP receptor protein. Science 295:1863–1866.
Kashtan N, Alon U (2005) Spontaneous evolution of modularity and network motifs. Proc Natl Acad Sci USA 102:13773–13778
Katzmann DJ, Burnett PE, Golin J, Mahe Y, Moye-Rowley WS (1994) Transcriptional control of the yeast PDR5 gene by the PDR3 gene product. Mol Cell Biol 14:4653–4661
Kepler TB, Elston TC (2001) Stochasticity in transcriptional regulation: origins, consequences, and mathematical representations. Biophys J 81:3116–3136
Kim J, White KS, Winfree E (2006) Construction of an in vitro bistable circuit from synthetic transcriptional switches. Mol Syst Biol 2:68
Kirkpatrick S, Gelatt CD, Vecchi MP (1983) Optimization by simulated annealing. Science 220:671–680
Kuhlman B, Dantas G, Ireton GC, Varani G, Stoddard BL, Baker D (2003) Design of a novel globular protein fold with atomic-level accuracy. Science 302:1364–1368
Looger LL, Dwyer MA, Smith JJ, Hellinga HW (2003) Computational design of receptor and sensor proteins with novel functions. Nature 423:185–190
Mangan S, Alon U (2003) Structure and function of the feed-forward loop network motif. Proc Natl Acad Sci USA 100:11980–11985
Mason J, Linsay PS, Collins JJ, Glass L (2004) Evolving complex dynamics in electronic models of genetic networks. Chaos 14:707–715
Mayo AE, Setty Y, Shavit S, Zaslaver A, Alon U (2006) Plasticity of the cis-regulatory input function of a gene. Plos Biol 4:e45
Metropolis N, Rosenbluth A, Rosenbluth M, Teller A, Teller E (1953) Equation of state calculations by fast computing machines. J Chem Phys 21:1087–1090
Nehlin JO, Carlberg M, Ronne H (1991) Control of yeast GAL genes by MIG 1 repressor: a transcriptional cascade in the glucose response. EMBO J 10:3373–3377
Nomura Y, Yokobayashi Y (2007) Dual selection of a genetic switch by a single selection marker. BioSystems 90:115–120
Paladugu SR, Chickarmane V, Deckard A, Frumkin JP, McCormack M, Sauro HM (2006) In silico evolution of functional modules in biochemical networks. IEE Proc Syst Biol 153:223–235
Park SJ, Chao G, Gunsalus RP (1997) Aerobic regulation of the sucABCD genes of Escherichia coli, which encode a-ketoglutarate dehydrogenase and succinyl coenzyme A synthetase: roles of ArcA, Fnr, and the upstream sdhCDAB promoter. J Bacteriol 179:4138–4142
Raichman N, Segev R, Ben-Jacob E (2003) Evolvable hardware: genetic search in a physical realm. Physica A Stat Mech Appl 326:265–285
Rodrigo G, Carrera J, Jaramillo A (2007) Genetdes: automatic design of transcriptional networks. Bioinformatics 23:1857–1858
Rodrigo G, Carrera J, Jaramillo A (2008) Asmparts: assembly of biological model parts. Syst Synth Biol. doi: 10.1007/s11693-008-9013-4
Sertil O, Kapoor R, Cohen BD, Abramova N, Lowry CV (2003) Synergistic repression of anaerobic genes by Mot3 and Rox1 in Saccharomyces cerevisiae. Nucleic Acids Res 31:5831–5837
Taubes G (1997) Computer design meets Darwin. Science 277:1931–1932
Vingron SP (2003) Switching theory. Insight through predicate logic. Springer Verlag
Weiss R (2001) Cellular computation and communications using engineered genetic regulatory networks. PhD thesis, Massachusetts Institute of Technology, Cambridge MA, USA
Yokobayashi Y, Weiss R, Arnold FH (2002) Directed evolution of a genetic circuit. Proc Natl Acad Sci USA 99:16587–16591