An Effective Method for Evolving Reaction Networks in Synthetic Biochemical Systems

Quang Huy Dinh, Nathanael Aubert, Nasimul Noman, Teruo Fujii, *Senior Member, IEEE,* Yannick Rondelez, and Hitoshi Iba, *Member, IEEE,*

***Abstract*—In this paper we introduce our approach for evolving reaction networks. It is an efficient derivative of the NeuroEvolution of Augmenting Topologies algorithm directed at the evolution of biochemical *systems* or molecular *programs*. Our method addresses the problem of meaningful crossovers between two chemical reaction networks of different topologies. It also builds on features such as speciation to speed up the search, to the point where it can deal with complete, realistic mathematical models of the biochemical processes. We demonstrate this framework by evolving credible biochemical answers to challenging autonomous molecular problems: *in vitro* batch oscillatory networks that match specific oscillation shapes. Our experimental results suggest that the search space is efficiently covered and that, by using crossover and preserving topological innovations, significant improvements in performance can be obtained for the automatic design of molecular programs.**

***Index Terms*—Evolutionary Algorithm, Molecular Programming, NEAT, Biochemical Oscillators, Crossover.**

I. Introduction

THE emerging field of Molecular Programming explores the possibility of using chemical systems to encode complex information processing in a molecular medium. The *in vivo* version of this idea, also known as synthetic biology [1], harnesses the molecular processes of living cells, notably the genetic expression machinery, to drive artificially inserted genetic circuits. For the *in vitro* case, on which we focus here, various forms of simplified biochemistry have been proposed to allow the rational building of scalar [2]–[9] or spatial [10] molecular circuits. These universal (in a computational sense [11]) schemes are based on both activatory and inhibitory processes between chemical compounds. Those processes can be combined in a modular way to achieve the building of arbitrary network structures, and hence arbitrary computations. One advantage of this *in vitro* approach is that the artificial circuits are not subjected to the interferences from the cellular host. Moreover, such systems are based on a limited set of molecular interactions and may be subjected to quantitative modeling [2], [3].

However, as in the *in vivo* case [12], it is not clear how best to harness the available *in vitro* molecular primitives to achieve a given information processing task. In most cases, Boolean logic is used as a framework to guide the engineering of complex operations [6], [13]. However, mass-action molecular kinetics is intrinsically an analog process and therefore an analog design has been proposed as a possible, more potent alternative [12], [14]. Hence, for example, this has led to the description [15], [16] or actual implementation [4] of various neural network architectures using for instance DNA-based reaction networks.

Whatever the framework, there are significant difficulties remaining in the design of molecular circuits (the process that leads from the desired behavior to the actual wet implementation). Generally, trial-and-error procedures are applied in parallel with the use of mathematical models [17], [18], and experimental engineering rules inferred from natural examples [19], [20]. These approaches are not efficient for optimization and implementation of complex systems that may require years and hundreds of experiments to complete. Hence for this reason, the automation of some steps becomes necessary.

Evolutionary Algorithms (EA) have already been applied to the problem of searching the space of possible biochemical networks implementing a given function [21]–[31]. However, due to computational issues, as well as the unavailability of accurate models [32], these approaches have always been applied to highly idealized mathematical representations of the underlying chemistry or biochemistry. While the reduction in the number of variables can maintain some qualitative prediction, quantitative prediction is generally lost. This means that the result of the search process may suggest general design guidelines, but will not provide directly a realistic candidate for the wet implementation [33]. In this work, we build on the existence of quantitative mathematical models describing a particular *in vitro* molecular programming scheme known as the DNA (Dynamic Network Assembly)-toolbox [2], [3], [5], [34]. These quantitative models could theoretically permit the automation of the full design of biochemical circuits, starting from the target function and up to the actual concentration of each chemical compound. However, they impose the use of realistic, hence computationally costly, models. Moreover,

Manuscript received July, 10, 2013.

Q. H. Dinh and H. Iba are with the IBA Laboratory, Department of Information and Communication Engineering, Graduate School of Information Science and Technology, University of Tokyo, Tokyo, Japan (e-mail: huy@iba.t.u-tokyo.ac.jp; iba@iba.t.u-tokyo.ac.jp).

N. Aubert is with the HAGIYA Laboratory, Department of Computer Science, Graduate School of Information Science and Technology, University of Tokyo, Tokyo, Japan (e-mail: naubert@is.s.u-tokyo.ac.jp).

N. Noman is with the School of Electrical Engineering and Computer Science, University of Newcastle, Australia (e-mail: nasimul.noman@newcastle.edu.au).

T. Fujii is with the Center for International Research on Micromechatronics (CIRMM)/Institute of Industrial Science (IIS), University of Tokyo, Tokyo, Japan (e-mail: tfujii@iis.u-tokyo.ac.jp).

Y. Rondelez is with the Laboratory for Integrated Micromechanic Systems - National Scientific Research Center (LIMMS-CNRS), University of Tokyo, Tokyo, Japan (e-mail: rondelez@iis.u-tokyo.ac.jp).

the increased number of variables yields inflation in the size of the search space, making simple optimization strategies unlikely to find a solution within a reasonable time. This calls for the development of an optimized and resource-efficient evolutionary method.

In recent years, new ideas have been put forward to improve the efficiency of EA, e.g. in evolving controllers for robots [35]. For example, NEAT [36] has been proposed to evolve the structure and parameters of Artificial Neural Networks (ANN), and showed better performance than the best fixed-topology methods on challenging problems. The increased efficiency has been ascribed to an effective crossover strategy and structural innovation protection, combined with a parsimonious addition of new nodes. The molecular networking of the DNA-toolbox, outlined in Fig. 1, rests on a combination of activatory or inhibitory chemical processes linking chemical compounds considered as the nodes of a network. In this sense, they bear a structural resemblance to neural networks. Therefore, we apprehend that it might be possible to adapt modern ANN construction strategies to *in silico* evolution of biochemical systems, with a potentially large improvement in search efficiency.

The rest of the paper is organized as follows. In section II we describe the overview of the previous works done to fully or partially automate the design of biochemical circuits and how it is related to the presented work. Section III provides a short primer on the *in vitro* biochemistry we are dealing with. Moving on, in section IV we describe our method in detail. Section V shows the experimental results and finally discussion is given in section VI.

II. Related Work

There have been several previous attempts to automate the design of genetic or chemical circuits by using simple versions of Genetic Algorithms (GAs). Francois and Hakim [21] pioneered this idea with the use of a mutation-only GA to evolve both the structure and the reaction rates of idealized abstractions of genetic regulatory networks. They showed that the resulting circuits provide a variety of functional designs relevant to the organization of known biological networks. Despite the simplicity of the evolutionary scheme, promising results were achieved, encouraging the use of evolutionary computations on such problems. A similar algorithm was used by Fujimoto et al. [22] to evolve gene regulatory networks that produce striped patterns of gene expression. They analyzed the evolved networks and discovered 3 classes, reproducing the various segmentation strategies observed among arthropods. Similarly, Kobayashi et al. [23] applied an algorithm with only structural rewiring mutations to find genetic networks that exhibit stable periodic oscillations with a prescribed temporal period. Deckard and Sauro [24] used a simple genetic algorithm without crossover to search for chemical reaction networks with particular signal processing capabilities. They did try crossover and noticed worse performance. Their work was later successfully extended by Paladugu et al. [25] to evolve oscillators, bistable switches, homeostatic systems and frequency filters. Crossover was not included, as they argued it would be disruptive.

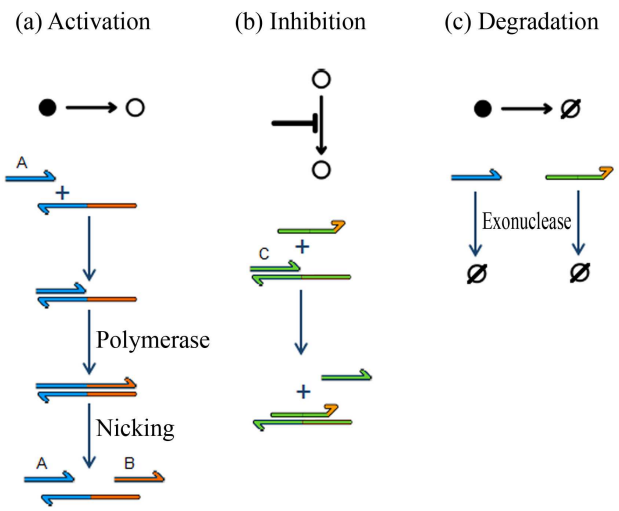


Fig. 1: DNA toolbox's components. DNA strands are represented as harpoons with colors coding for domains. Templates are 22-base long single stranded oligonucleotides whereas signals are their partial complements. In Activation (a), a signal sequence A comes in contact with the template A→B and is elongated by a polymerase. Next, a nickase nicks the newly extended strand, releases the input signal sequence A and an output signal sequence B. In the inhibiting process (b), the inhibiting sequence (yellow head arrow) is designed to bind strongly to the template but cannot be elongated. It prevents the signal sequence C from activating the template, thus inhibits the process encoded by the template (here C→C). In addition, signal sequences and inhibiting sequences can be degraded by the exonuclease. The process is shown in (c). Note that the template strands cannot be degraded by the exonuclease because they are chemically protected.

Simple GAs with mutation as the only genetic operator are not efficient to solve difficult problems or deal with complex models that require heavy computation, and several work arounds have been suggested. A recent example is the work by Marchisio and Stelling [26]. In their study, the target Boolean function is defined by a truth table, and solutions (structures of the network) could be computed without any optimization procedure. Then, only the most feasible solution undergoes a parameter optimization aiming for reliable performance *in vivo*. Separating the structure design from parameter optimization obviously saves computation time. However, their method relies on the existence of a rational solution, which cannot always be expected, especially in complex tasks. In addition, a correct solution might require a precise combination of both structure and parameters, advocating for their co-evolution. This issue was highlighted by the approach of Cao et al. [27], who used a nested evolutionary algorithm for the sequential structure identification of the target system and parameter optimization of the selected modules. Other works simply freeze the structure, and make use of an efficient Evolution Strategy (ES), to optimize the parameters only. Notable examples are Jin and Sendhoff [28] who used an ES to optimize the parameters of regulatory motifs with a given structure, and Hallinan et al. [29] who combined an ES and stochastic simulation of genetic circuits, to design regulatory systems

based on the Bacillus subtilis sin operon.

A complete GA, including genetic operators of mutation, elitist selection, and crossover, was used by Drennan and Beer [30] successfully to evolve the oscillatory network called repressilator [31]. In their algorithm, pseudo-DNA sequences are used directly as individuals' encoding and genetic operations can be applied in a pseudo-biological way. Although these genetic operators are applied at the sequence level of the genomes, the network graph is described by a connection matrix, leading to a high probability that connection will be broken in the genetic operators. Thus, it is hard to pass on connectivity innovation (or a block of key genes) to later generations, making the algorithm possibly less effective on problems that require precise combination of connections [36].

As can be seen, most GA approaches avoided using crossover, for the reason that meaningful structural crossover strategies are not obvious and worse performance is usually experienced when crossing over non-homologous networks. Moreover, all the previously described systems used highly idealized models of the underlying chemical or biochemical processes. This idealization generally results in a drastic decrease in the computational cost associated with the numerical integration of the temporal behavior, which can compensate for the relative inefficiency of the evolutionary search. Neuroevolution, the evolution of ANN using EAs has similar problems, which were effectively addressed by a method called NEAT [36]. NEAT uses historical marking, facilitating speciation and crossover, and is reported to be very efficient on challenging benchmarks. Indeed, ANNs are very similar to reaction networks and both can be represented as graphs. In addition, NEAT was shown to be able to evolve non-ANN systems efficiently, e.g. in physics [37]. This suggests that NEAT could be a starting point for building an efficient framework for the evolution of realistic molecular circuits targeting a given dynamic function.

III. Modeling of Biochemical Systems

Our work is based on the DNA-toolbox [2], [3], [5], [34], a generic set of DNA-based catalytic activating or repressive reactions that can be cascaded in arbitrary networks. These networks are maintained out of equilibrium and can thus perform complex dynamics. Experimentally demonstrated examples include relaxation oscillators [2], bistable systems and toggle switches [3]. The biochemical functioning has been described in detail and is summarized in Fig. 1. The reactions are based on three thermophilic enzymes, namely an exonuclease, a polymerase, and a nickase. This enzymatic hardware is used to run a circuit whose connectivity is encoded by the sequence of DNA templates (the software). Building on this analogy, the data correspond to unstable DNA oligomers, that mediate the communication between the templates and are constantly flowing to a chemical sink embodied by the exonuclease. These data oligomers can possess one of two roles: signal sequence or inhibiting sequence. Signal sequences provide the activation by acting as primers on templates, whereas inhibiting sequences behave as inhibitors by sequestering a given template (hence blocking the production of signals by this template). These activating or inhibiting modules can be cascaded in arbitrary networks by simply designing templates with the correct input and output sequences. These templates instruct the mass-action kinetics of the system and thus control the collective dynamic behavior of the system.

In the DNA-toolbox each reaction network corresponds to i) an experimentally constructible biochemical system, ii) a graphical representation and iii) a quantitative mathematical model based on mass action kinetic description of the underlying molecular processes. Fig. 2 shows the graphical representation of the Oligator [2], an experimentally validated oscillator based on three sequences and three templates: each node corresponds to a sequence, and each arrow corresponds to a template (positive interaction); bar-headed arrows represent inhibition (negative interaction), and should target a template (an arrow). The weight of these links corresponds, at the chemical level, to the concentration of the templates and the thermodynamic stability of the sequence. The actual temporal behavior of such a circuit results from the combination of the structure (topology of the network) and a set of chemical parameters including kinetic constants and concentrations. Since we focus here on autonomous systems, there is no input node. In other words, every node must have at least one incoming link from another node (or else it would not contribute to the dynamics).

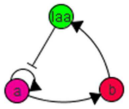
This paper focuses on the *in silico* design of DNA-toolbox network systems and is based on the quantitative modeling proposed by Padirac et al. [3], reviewed in [38]. These models are complete representations where every chemical compound is considered and all the relevant kinetic parameters have been independently tabulated. These models produce realistic predictions of the time traces, at the cost of using large systems of ordinary differential equations (ODE). For example, the 3-node network shown in Fig. 2 is simulated by a 19-variable ODE system. In the present evolution runs, the parts being evolved are the topology, dissociation rates (which are directly linked to the thermodynamic stability of the sequence, i.e. its melting temperature) and template concentrations, while the concentrations of the three enzymes are held fixed. A single arbitrary set of initial concentrations is also used for all the individuals.

IV. Framework for Evolving Reaction Network (ERNe)

The main differences between ANN and reaction networks are the addition of inhibition links and biochemical parameters. Thus, ERNe encoding allows representation of inhibition, and has added parameters. In addition, mutation and crossover operators are also modified from the original NEAT. In this section, the algorithm is described in detail, reviewing the encoding, the genetic operators for mutation and crossover, and the speciation process.

A. Encoding

Our genome consists of sequence genes and template genes. Different from NEAT's node genes, each sequence gene can represent either a signal sequence or an inhibiting sequence



Sequence (Node) Genes

Name	a	b	I_{aa}
Kinetic Parameter	3.873	3.873	0.0129
Initial Concentration	1.0	0.1	0.1

Template (Connection) Genes

From	a	a	b
To	a	b	I_{aa}
Concentration	30.5	10.3	15.4
Enabled	True	True	True
Innovation	1	2	3

Fig. 2: A graph representation and corresponding ERNe encoding of the Oligator. Nodes represent sequences while arrows represent templates; bar-headed arrows represent inhibition. a and b are signal sequences, whereas the green node Iaa is an inhibiting sequence; it inhibits the template a→a (i.e. the self-activation of a). The system has three sequences and three templates. Thus, there are three node genes and three connection genes in its ERNe encoding.

in the system, and consists of a name, a kinetic parameter, and an initial concentration. Each template gene specifies the from-node, the to-node, the template concentration, an enable bit that indicates whether or not the template is enabled, and an innovation number that uniquely identifies each template in the system. This innovation number plays an important role in the implementation of the crossover and speciation and is set for template genes based on the following rule. During the evolution, whenever a template is added to the system, we check if that specific link exists in the evolution history, in which case it takes the original link's innovation number. Otherwise, the next available innovation number will be assigned to the template gene. To make this historical marking effective, a naming mechanism for newly added sequences should be carefully designed. Ideally, in different systems, sequences with same name should carry the same role. It is, however, almost impossible to keep that ideal condition as later mutations might change the role of any sequence. However, a simple naming mechanism can partly deal with this matter. We use a list to map node names with the way they are created, for example, an entry A→A to B shows that whenever a new node is added in the middle of the template A→A, it must be named B. An example of genetic encoding describing the Oligator is shown in Fig. 2.

B. Mutations

Mutations can be applied to change both the parameters and the network structure. We have the following mutation operators: parameter only, disable template, switch template, add sequence, add activation, and add inhibition. Their relationship and effects on the network are shown in Fig. 3.

In parameter mutation, every parameter has a probability to be mutated to a new random value calculated as

$$newValue = oldValue \times (1 + f_1 \times rand_1) + f_2 \times rand_2$$

where f_1 and f_2 are fixed (set to 0.2 and 2.0 respectively in the following experiments), and $rand$ are standard normal deviates. To maintain the biochemical relevance, the available

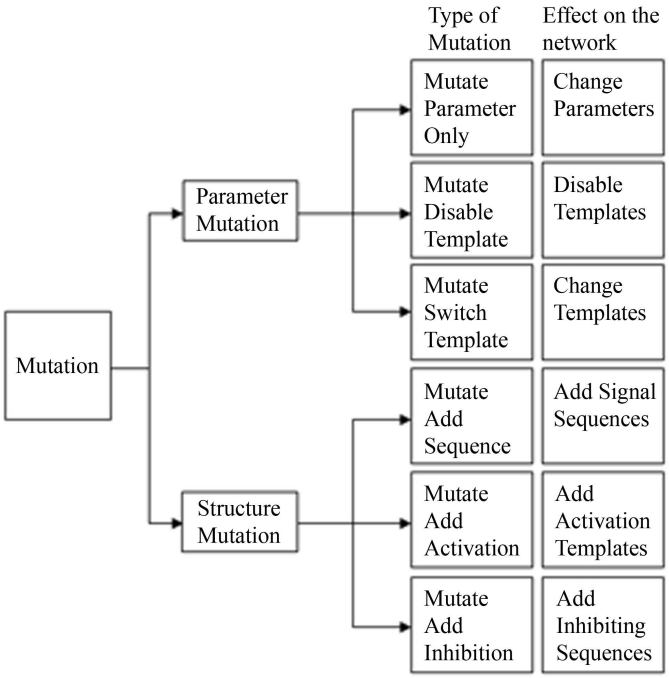


Fig. 3: Different types of mutation used in ERNe. Note that although Mutate Disable Template and Mutate Switch Template belong to Parameter Mutation, they actually change the structure as well.

parameter range is bounded within physically reasonable values, and two further checks are performed on the mutated concentration: first, if that value is below zero, a Switch Template Mutation happens. Indeed, negative concentrations do not exist, so we physically interpret this change as a switch to an inhibitory template. An example of switch template mutation is shown in Fig. 4(ii). Assuming template b→a's concentration is mutated to a negative value, the changes described by Algorithm 1 are applied to the system. Second, if the new value of a template concentration is above zero but below a threshold (here set to 1.0 nM), it has a probability to be disabled through a Disable Template Mutation. These mutations are based on the idea that, once a connection has no meaning (template concentration close to zero), it should be removed (disabled), or even changed to a reverse polarity. Also, because each template brings some variables to the ODE model, these steps help to maintain structure as simple as possible by enabling the removal of templates with a very low concentration, whose relevance to the target function is presumably low.

In Add Sequence Mutation, a new signal sequence is added to the system. There are two ways to perform this type of mutation, as described in Fig. 4. The first is to select an existing template, split it and place the new sequence in the middle (Fig. 4(iii)). The second is to add a new sequence with a self-activatory connection (remember that nodes without incoming link are forbidden) to inhibit an existing template (Fig. 4(iv)) or to activate an existing sequence (Fig. 4(v)).

In Add Activation Mutation, two unconnected sequences are chosen randomly and a new template is added between them (Fig. 4(vi)).

disable template b→a;
if *no existing template that creates a* **then**
  add the template a→a;
  select the template a→a;
else
  select any template that creates a;
end
add the inhibiting sequence Ixa that inhibits the template x→a selected previously;
add the template b→Ixa;

Algorithm 1: Switch Template Mutation

The Add Inhibition Mutation's example is described in Fig. 4(vii). A random template is selected (a→a in the example) and the corresponding inhibition node (Iaa) is created. Then, the mutation selects a random start node (b) and adds the template b→Iaa, completing the module. A special case arises when there is already an activating template between sequences a and b, in which case it is simply disabled (the mutation is then equivalent to a Switch Template Mutation).

C. Crossover

The ERNe encoding and the use of innovation number make crossover straightforward. Using innovation numbers, the template genes in both parents are lined up, then crossover techniques such as one-point and two-point crossover can be easily applied. Fig. 5 shows an example of a two-point crossover operation, which leads to an interesting increase in loop size. We also need to decide how to create node genes for the child. Currently we simply reconstruct it by reading the template genome in order of innovation number, and select the parameters of the corresponding node in the parent that provided the connection.

D. Speciation

When evolving complex systems, it is important to protect topological innovations. Indeed, smaller structures tend to optimize faster than larger ones. Moreover, in many cases, adding new nodes and connections to the system initially decreases the fitness. Thus, newly evolved structures seldom survive more than one generation, even though the innovations they introduce might be crucial towards solving the task in the end [36]. Therefore, we utilize the concept of species in which individuals with similar topologies are sub-grouped in the same species. Consequently, individuals compete primarily within their own niches instead of with the whole population. This way, topological innovations are protected and get a chance to optimize. For each species, we call the first individual that discovers the species its representative. We use a topological compatibility distance δ calculated as

$$\delta = \frac{M}{N}$$

with M the number of mismatched template genes between the two individuals and N the number of total template genes in the larger genome. If δ is below the speciation threshold δ_t we

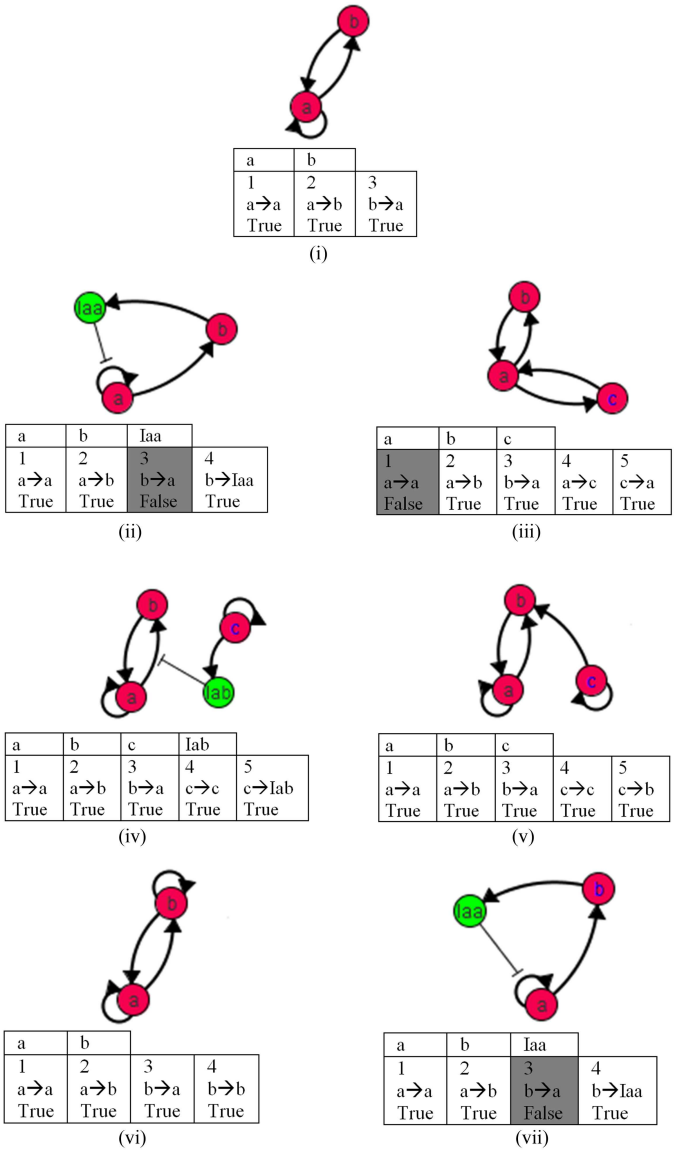


Fig. 4: Mutations from original structure (i). Switch Template Mutation creates the structure (ii) in which the template b→a is disabled, an inhibition node Iaa is added, then a template b→Iaa is added with a new innovation number. Add Sequence Mutation creates the structure (iii) in which a sequence c added in the middle of existing template a→a, the structure (iv) in which a sequence c added to inhibit template a→b, and the structure (v) in which a sequence c is added to activate sequence b. Add Activation Mutation creates the structure (vi) in which the template b→b is added. Add Inhibition Mutation creates the structure (vii) in which the template b→a is disabled, an inhibiting sequence Iaa is added, and a template b→Iaa is added.

say that the two individuals can be placed in the same species. We then use the following method to determine species for a newly created individual. We calculate its distance to all species' best individuals in the previous generation. The first match (having distance below the speciation threshold δ_t) decides its species. If no suitable species are found, all individuals in previous generations are considered. If no suitable species could still be found, then all species from the evolution history will be considered, in which case, the

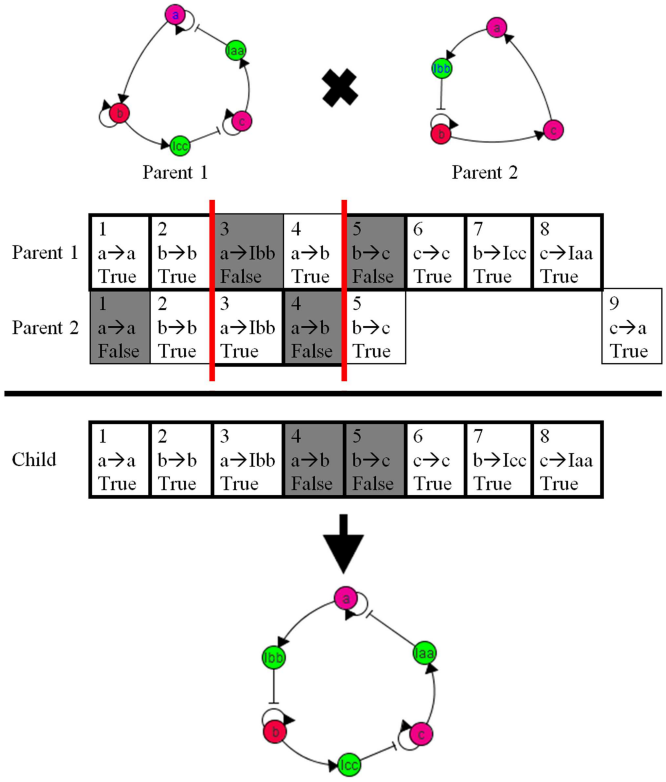


Fig. 5: Two-point crossover of the two individuals Parent 1 and Parent 2. The template genes are lined up in the order of innovation number. The red lines show the two crossover points. The child takes the genes number 1, 2, 5, 6, 7, 8 from Parent 1, and genes number 3, 4 from Parent 2. As a result we have a new topology that has some parts of both parents.

distance is calculated between the individual and the species' representative. If the species cannot be decided after all those steps, then a new species is created, having the individual as the representative.

Every species at the current generation is then assigned a different number of offspring in proportion to the average fitness of its individuals. Using the average fitness, individuals in the same species must share the fitness of their niche. In other words, a species is considered better if it has higher average fitness from all the individuals. In addition, we implement a capping mechanism that limits the growth of a species to at most 10% of the whole population. This prevents some temporally good species from growing too fast and taking over the population, leaving other species less chance to optimize.

Another problem introduced by speciation is the number of species at each generation. If we have too many species, each might not have enough space to evolve. In contrast, low number of species means low diversity. Actually, the number of species depends on the speciation threshold δ_t. With a high compatibility threshold, we tend to have less species. In our initial attempts, the performance of the run crucially depended on the value of this parameter. To bypass this issue, we implemented a dynamic adjustment of speciation threshold so that the number of species will move toward a specific target N_s:

$$\delta_t = \begin{cases} \delta_t + \epsilon & \text{if number of species} > N_s \\ \delta_t - \epsilon & \text{if number of species} < N_s \end{cases}$$

ϵ is called the modification step.

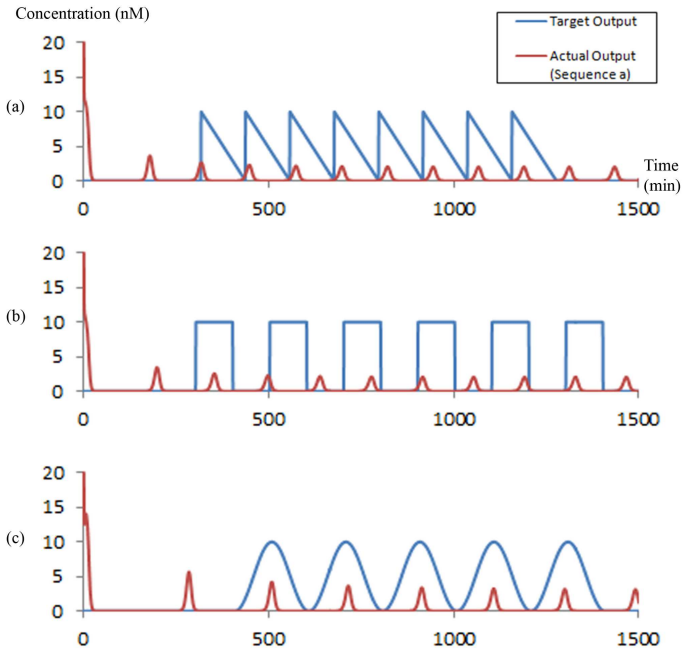


Fig. 6: Best time traces obtained for (a) sine, (b) rectangular, and (c) sawtooth oscillations by applying standard DE on optimizing the Oligator's parameters.

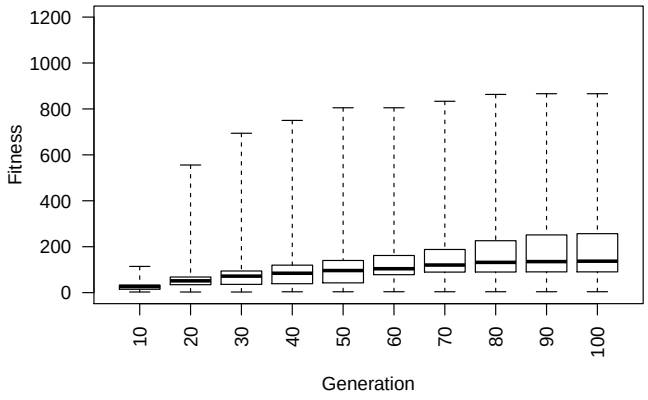
V. Result

Our initial tests confirmed that the algorithm could readily find already known oscillatory [2] or bistable [3] networks. Starting from a single node, it needs only a few generations to find these structures and optimize their parameters. We thus went on to more challenging benchmarks and we tried to find oscillating biochemical systems matching specific periodic functions: sine, rectangular and sawtooth oscillations. The target outputs are generated with reasonable amplitude and period. We use a lexicographic fitness function that first checks the presence of oscillations and then calculates the fitness as

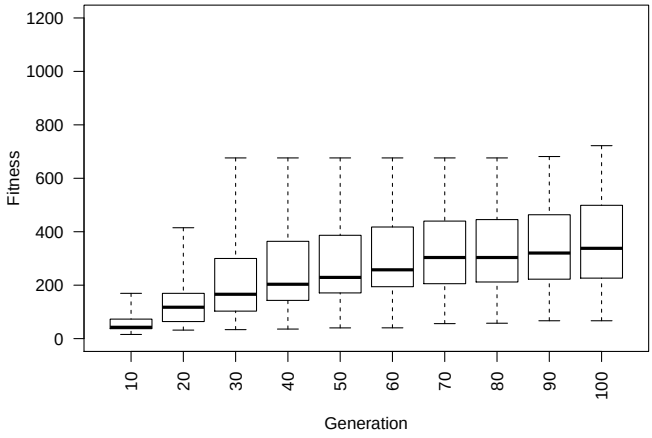
$$fitness = \frac{1000}{MSE} \times limitcycle$$

where MSE is the Mean Square Error between the concentration of signal sequence a and the target function (the geometric MSE was used for the two discontinuous functions) and $limitcycle$ evaluates the proximity a limit cycle by comparing the amplitude of the first and last peaks of the signal sequence a's concentration in the looking interval (set to 300-1500 minutes in our experiments).

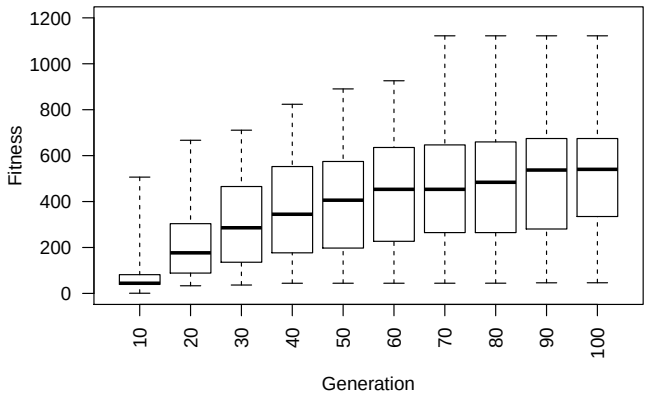
There has been no research showing or giving hints about how such precisely shaped oscillations could be obtained. Thus, our initial attempts were to perform parameter optimization on a recently reported and controllable oscillatory



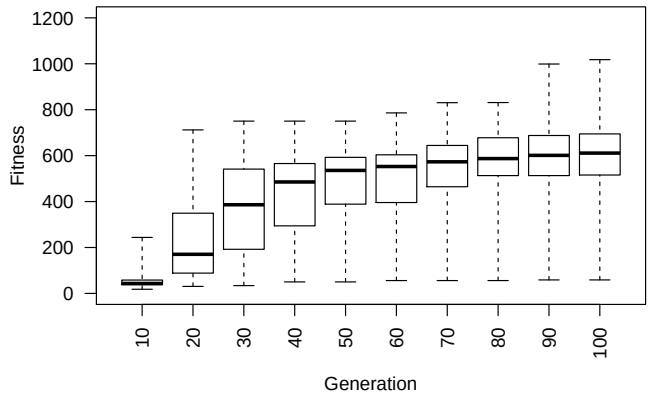
(a) ERNe with crossover, without speciation



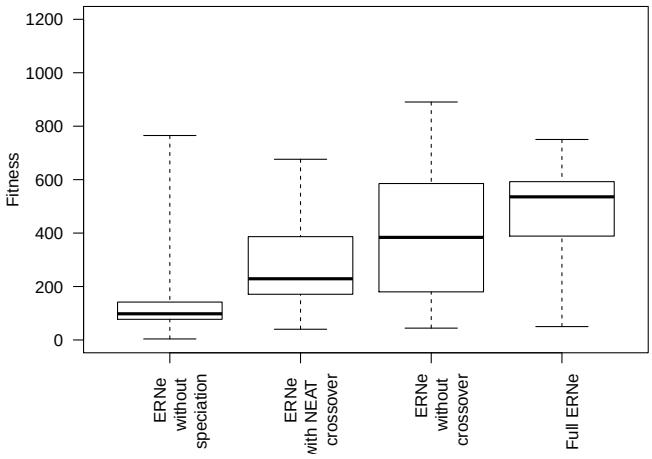
(b) ERNe with original NEAT crossover



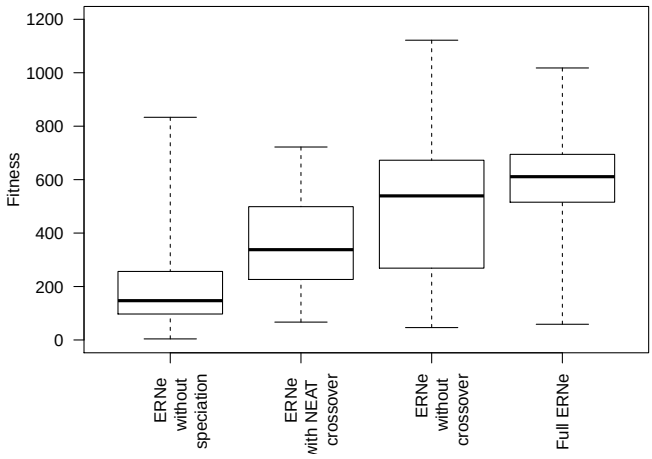
(c) ERNe without crossover



(d) Full ERNe



(e) at generation 50



(f) at generation 100

Fig. 7: Performance results for rectangular oscillation problem. Best fitness values over generations are displayed in box-plot for (a) ERNe without speciation, (b) ERNe with original NEAT crossover, (c) ERNe without crossover, (d) Full ERNe, (e) snapshot of all the runs at generation 50, and (f) snapshot of all the runs at generation 100.

structure built out of the DNA-toolbox, the Oligator. The method used was standard differential evolution [39] with population size of 40 and generation number of 1000. Our best attempts to match the sine, rectangular, and sawtooth oscillations which resulted in MSE of 31.6079, 34.3577, and 25.3381, respectively (time courses shown in Fig. 6), were unsatisfactory. It appears that, without changes in topological structure, the optimizer cannot simultaneously meet the required oscillation in terms of period, amplitude, and shapes. Thus, we believe these targeted time traces could only be generated by more complex structures, that are impossible or at least very difficult to predict in advance. Later in this section, our discovered systems are shown to match these oscillations much better. Their structures are, indeed, much more complex, and in many cases, unrelated to the oscillator.

All our runs use the parameter settings shown in Table I. These parameter settings were decided upon after some tuning on different problems and might not be the best set. However, we found it to be efficient for the benchmark runs. For elitism, the best individual of each species is copied to the next generation unchanged. All the runs started from an initial individual that contains only one signal sequence (and hence its self-activatory connection). The rest of this section represents the experiment results in the following order. First, we introduce the performance comparisons between different settings to prove the algorithm's effectiveness. We then show the analysis of biochemical networks evolved relative to the proposed benchmarks. Finally, evolutionary pathways of some interesting runs are given.

TABLE I: Parameters used in experiments

General Parameters	
Population size	200
Number of generations	100
Selection method	Tournament (size 5)
Crossover technique	One-point crossover
Speciation Parameters	
Preferred number of species	10
Starting speciation threshold	0.6
Minimum speciation threshold	0.1
Speciation modification step ϵ	0.03
Crossover & Mutation Parameters	
P(Mutation only)	0.5
P(Interspecies mating)	0.01
P(Mutation after Crossover)	0.75
Mutation Parameters	
P(Parameter Only)	0.9
P(Single Gene Mutation)	0.8
P(Structure-Add Node)	0.2
P(Structure-Add Activation)	0.2

A. Performance Results

To evaluate which features of ERNe are the most important we performed an experiment using the hardest problem, rectangular oscillation. This experiment compared 4 algorithms, each of which was run 50 times. The first two algorithms were ERNe with and without crossover. The third used crossover, but without speciation (like the simple GA that has been used widely in related approaches). The last test is ERNe with the original NEAT crossover technique. The detailed result is given in Fig. 7, showing the performance over generations of all the runs. In addition, we defined a satisfactory fitness of 500, with which outputs of some example systems are shown in Fig. 8. Then, the average generations required to discover a solution with satisfactory fitness are shown in Table II, and their statistical Students t -test results are shown in Table III. These p-values indicate that there are significant differences between full ERNe and others' averages number of generations required to achieve a satisfactory solution. It could be observed from Fig. 7 that the full ERNe with crossover plus speciation is the best. Performance drops significantly in the runs without speciation. Original NEAT crossover performed poorly, even worse than when no crossover is involved. However, with ERNe one-point crossover applied, a solution with satisfactory fitness was found about 50% faster than without crossover on average. These results suggest that crossover, if used properly, plays an important role to speed up the evolution, but those good solutions can also be found without crossover, with the cost of larger runs. Moreover, the fact that the original NEAT crossover - which was efficient in evolving ANN - is disruptive in this problem, suggests that differences between ANN and biochemical reaction networks are significant and should not be overlooked.

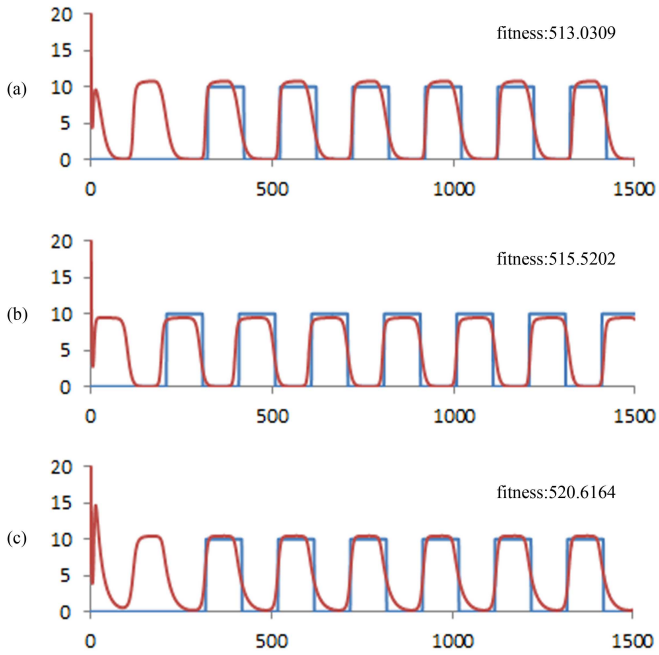


Fig. 8: Example outputs of systems with fitness just above 500, for the rectangular oscillation problem.

B. Analysis of Reaction Networks Evolved

As the performance result suggests, ERNe with crossover showed highest performance. Thus, we use ERNe with crossover for the rest of the experiments. For each of the

TABLE II: Generations required to achieve fitness above 500 for the rectangular oscillation problem

ERNe without speciation	145.6±22.1
ERNe with NEAT crossover	125.2±45.9
ERNe without crossover	84.2±61.1
Full ERNe	59.6±50.7

TABLE III: p-values of t-Distribution calculated from Table II

Full ERNe vs ERNe without speciation	1.48E-09
Full ERNe vs ERNe with NEAT crossover	2.59E-5
Full ERNe vs ERNe without crossover	0.0310

three oscillation shapes, the algorithm was run for 50 times. Importantly, the three problems were addressed with the exact same settings, shown in Table I.

Fig. 9 shows a typical result for the sine oscillation problem. The observed output and the target output are matching almost perfectly. The structure is based on a extended version of the Oligator structure (Fig. 2) with a longer delay line in the negative feedback loop. Side feedforward loops are grafted to this structure and probably serve to fine-tune the shape of the time trace.

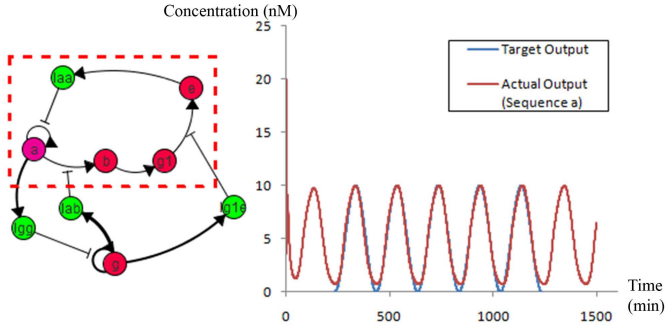


Fig. 9: Typical solution for the sine oscillation problem, with structures based on an elongated Oligator (the structure in the red dotted rectangle), and their corresponding output.

Fig. 10 shows results for the rectangular oscillation. As expected, the fit is not as good as that observed previously for continuous functions. Interestingly, most runs (40/50) converged toward an unreported and unrelated oscillatory topology involving three self-activating sequences linked together by three inhibition reactions. We called this topology the switch oscillator in reference to its tendency to produce fast switching separated by flat plateaus (Fig. 10(i)). Within these 40 runs, 34 were found to contain the extended variant where signal sequence a (the one that we are tracking) is not self-activating but serves as an intermediate compound (Fig. 10(ii)) while the remaining 6 had an additional counter-rotating activation also originating from a (Fig. 10(iii)). This robust convergence to precise topological features suggests that the framework is effective in exploring the search space. The remaining 20% of the runs fell into the long Oligator family (Fig. 10(iv)). Even after intensive tuning, this class of oscillators was unable to accurately match the constant parts of the target function and produced lower fitness.

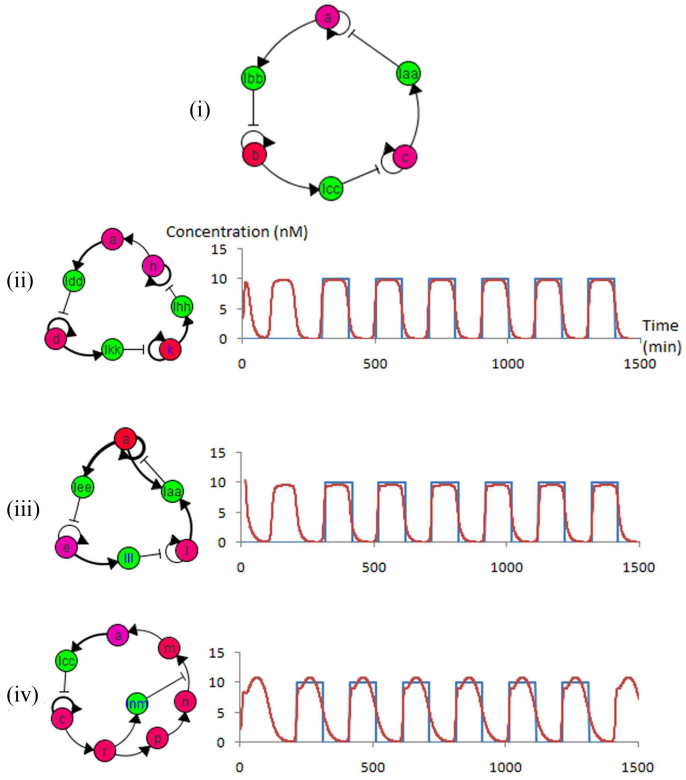


Fig. 10: The new switch oscillator (i) and typical results for the rectangular oscillation problem. 40/50 of the runs converged to a topology derived from (i) where three self-activating sequences linked together by three inhibition reactions and including either an extension (ii, 34 runs) or a self-inhibition (iii, 6 runs) located at the a stage. The rest (10/50 runs) fell into the long Oligator category (iv) and did not match well.

Fig. 11 shows the results for sawtooth oscillations. As expected given the relaxation form of the target function, all runs converged to a topology based again on the long Oligator (which is a relaxation oscillator). We could again observe the repetitive convergence to precise topological details, such as the target sequence being either self-activating (7/50 runs with average MSE of 5.2004±1.3651, Fig. 11(i)) or the last one of the negative feedback loop (43/50, producing significantly better individuals with average MSE of 1.9422±0.5343, Fig. 11(ii)). In this case, also, side branches serve to fine-tune the shape of the temporal evolution.

C. Evolutionary Pathway

One example of the evolution to match the rectangular oscillation is shown in Fig. 12. The initial network (i, generation -1) mutated to (ii, generation 0) by adding a new node d in the middle of the self-activation a→a, with no improvement in fitness (no oscillation means zero fitness). Later, the activation a→d was switched to inhibition, creating the Oligator motif (iii, generation 1). This structure is compatible with oscillation, however the parameters are not yet optimized. Later, it mutated to a long Oligator (iv, generation 3) - which is a more robust oscillator - by adding a new node h. The system (iv) optimized its parameter for several generations and could reach

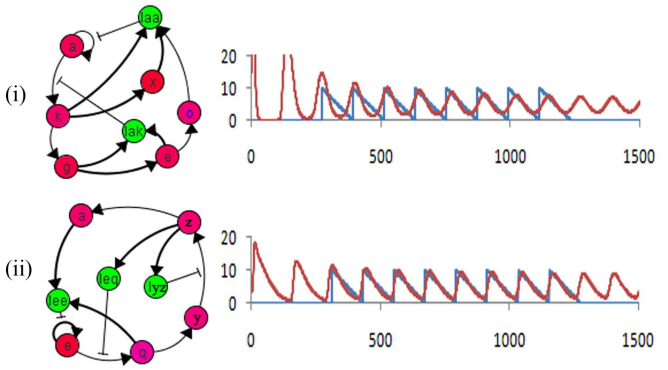


Fig. 11: Typical structures and their sample outputs for the sawtooth oscillation problem. All the runs converged to the topology of elongated Oligator, in which 7/50 topologies have sequence a as self-activating sequence (i), and 43/50 have sequence a as the last of the negative feedback loop (ii). The latter provides better solutions.

a strong oscillation with a fitness of 40.8673 at generation 8. At generation 9, another new node k is added to the system to elongate the feedback loop of the system and results in system (v). This mutation decreased the fitness slightly. The final structure - which is a derivative of the switch oscillator (Fig. 10(i)) could be obtained at generation 10, when two switch mutations happened at the same time to the activation d→k, and k→h. Interestingly, a huge fitness drop (to 12.2585) was experienced with the new structure. Although the oscillation it produces are more similar to rectangular oscillation, the mismatch in both amplitude and period led to high MSE. Thanks to the speciation, this structure could be kept in the population and optimized and finally became the best solution with fitness of 668.1365. Here we only focus on the mutations that changed the structure. There were, in fact, many parameter mutations coming along that also played important roles.

For this specific run, also, the speciation process is depicted in Fig. 13, where the generations are shown from the left to right, with the species depicted vertically for each generation. The height of each species is proportional to its population in that generation. It can be clearly seen that the switch oscillator species appeared early with a small population, then became dominating later. The long Oligator species, on the other hand, always performed well and occupied a stable portion of the population. It is also interesting that most initial species disappeared before generation 10, indicating that simple structures are not suitable to solve this complex problem.

VI. Discussion

The results for the three oscillation problems show the efficiency of the proposed framework. In each case, we could discover not only the structure but also the set of parameters to provide excellent matching. This proves that the framework can be used to optimize simultaneously the structure and parameters of realistic models of biochemical networks. The algorithm showed some signs of generalization as all the three problems were successfully solved using the same algorithmic setup. The repetitive convergence to the

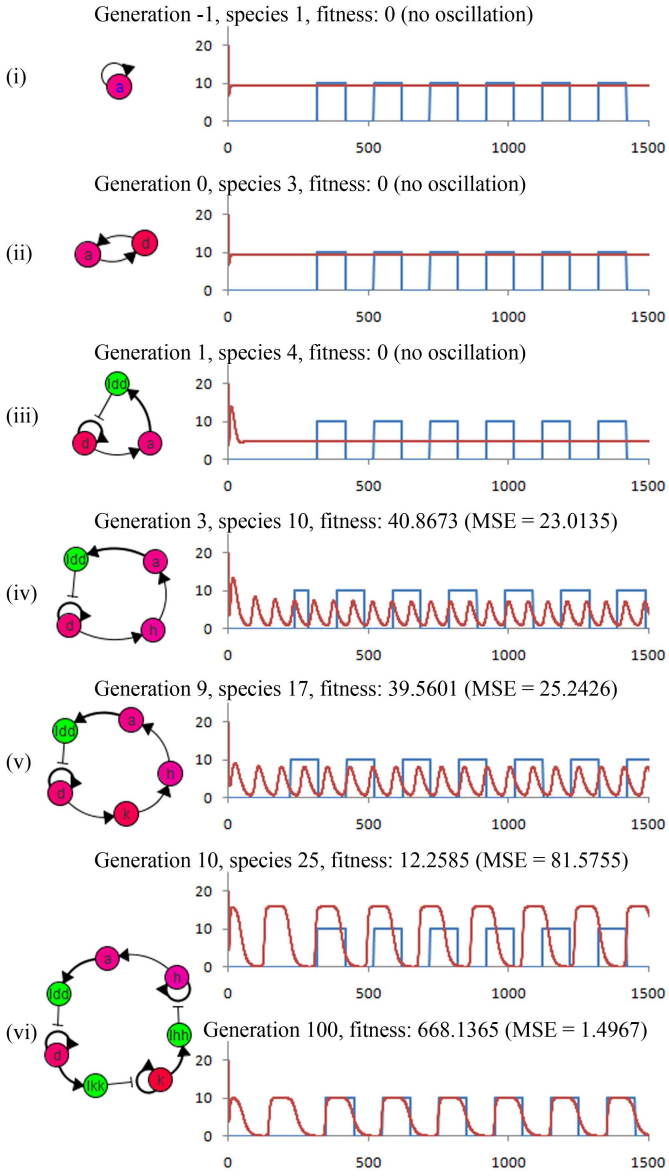


Fig. 12: Evolution of a network for the rectangular oscillation. Only the changes to the topology are shown. There were, in fact, many parameter optimizations during the evolution.

same problem-dependent topological features indicates that the search space was effectively explored even in these relatively small runs (about 2 hours each on a desktop machine). Finally the convergence rate was much faster than that of a basic GA without crossover or speciation (Fig. 7). These efficiency and robustness result from some important algorithmic features.

First, biochemical reaction networks normally require several synergistic topological mutations to progress to a better solution, and a single change may initially decrease the fitness. We have shown that speciation based on topology allows structural innovations to be preserved and improved during later generations. Template innovation numbers are instrumental in this regard. However, a specific issue emerged from the use of the DNA-toolbox where nonlinear behaviors are stabilized primarily by delays created by long activation chains [40].

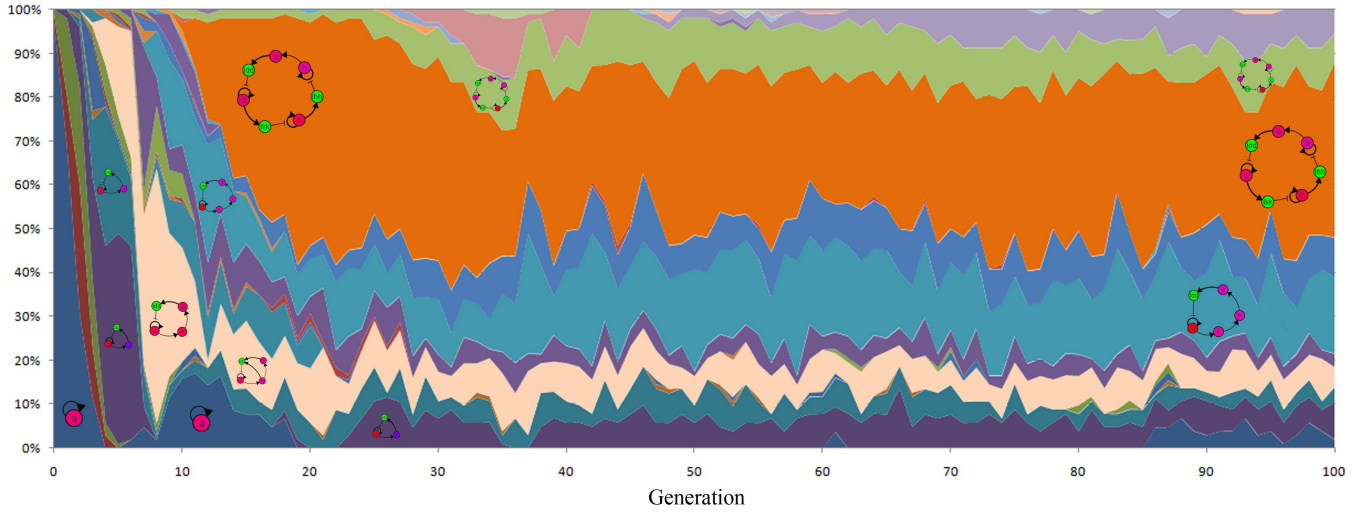


Fig. 13: Species visualization for a sample run of the rectangular oscillation problem.

Fig. 14 shows an example of such chains and the naming problem that they create: although the topology of (ii) and (iii) are the same, they have different node names that might lead to the creation of two species exploring the same area - hence a decrease in search efficiency. We implemented a mechanism to detect whether or not the template receiving the added sequence is in the middle of an unbranched activation chain (in Fig. 14 system (i) has the unbranched activation chain of a→b→c), in which case the template in the beginning of the chain will be selected instead. In the example, if a new node is to be inserted to the chain a→b→c, the template a→b is always chosen. This mechanism only partly solved the issue, so encouraging behavioral diversity [41] might be a more general and possible future work.

In evolving computationally heavy models, ERNe also has some advantages over general GAs. Indeed ERNe starts from minimal individuals and evolves by adding nodes and connections, so that the search direction is from simpler to more complex topologies. Thus, less computation is needed in comparisons with other methods starting e.g. from random populations [36]. In addition, we tend to find minimal structures without the need of implementing any penalty on structural complexity. However, in some cases, we observed that the best individual found was not the simplest structure able to produce an equivalent fitness. This minimal structure, however, always existed in the run at some stage, but could not optimize as efficiently as other more complex structures. Because the ultimate goal of this work is to actually implement the results of the *in silico* process into wet experiments, we are primarily interested in the simplest possible networks. Therefore the correct allocation of resources between the local (parameter) and the global (structure) search remains an important goal [42]. We note, however, that the model used in the present proof of concept did not include competitive inhibition of the enzymes [16], [43], [44]. We anticipate that the inclusion of this mechanism will naturally contribute to limit the total concentration of templates, hence the complexity of the network.

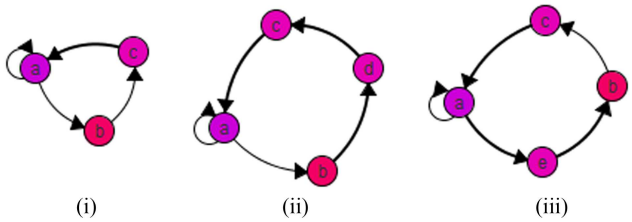


Fig. 14: Chaining problem. From the system (i), system (ii) was created by inserting a new sequence in the middle of the link b→c, whereas system (iii) was created by adding a new sequence in the middle of the link a→b. System (ii) and system (iii) have the same topological structure but their different naming may lead to erroneous speciation.

Having correct and relevant mutation rules is also crucial, and the set of mutation operators should ensure evolvability, i.e. the genome's ability to organize adaptively how mutations affect the phenotype. Disabling the Switch Template Mutation, for example, lowers the probability that the Switch Oscillator could be found, to roughly 10%. We assume that this mutation provides important bridges in the search space, because in the biochemical implementation, activation and inhibition are represented by two very different processes, whereas in the original neural framework they correspond to a continuous drift from positive to negative values. The large performance improvement brought by this mutation shows that the physical nature of the information processing medium should be taken into account in the definition of relevant mutation operators.

VII. Conclusion

We have successfully adapted recent algorithmic innovations from neuro-evolution to the search for realistic biochemical systems. The differences between ANN and reaction networks, e.g. the additional connection type, could be efficiently encoded in ERNe. The experimental results showed that speciation, and an efficient crossover strategy, provide a dramatic increase in search effectiveness and hence

reduction in search time. In our future research, we will apply this work to solve more challenging problems including the possibility for the network to interact with an external, time varying outside environment (i.e., non autonomous systems). We will also explore concepts such as robustness [45] or non-regulatory couplings [34]. Moreover, other possible directions are to embed the ideas of effective local search [39], multi-objective optimization [46], [47], and automatic parameter tuning [48], [49] to improve the framework's efficiency. We hope that this work will encourage further research into the *in silico* design of molecular programs of increasing complexity.

REFERENCES

[1] E. Andrianantoandro, S. Basu, D. K. Karig, and R. Weiss, "Synthetic biology: new engineering rules for an emerging discipline," *Mol. Syst. Biol.*, vol. 2, no. 28, 2006. [Online]. Available: http://dx.doi.org/10.1038/msb4100073

[2] K. Montagne, R. Plasson, Y. Sakai, T. Fujii, and Y. Rondelez, "Programming an in vitro DNA oscillator using a molecular networking strategy," *Mol. Syst. Biol.*, vol. 7, no. 466, 2011. [Online]. Available: http://dx.doi.org/10.1038/msb.2010.120

[3] A. Padirac, T. Fujii, and Y. Rondelez, "Bottom-up construction of in vitro switchable memories," *Proc. Natl. Acad. Sci. USA*, vol. 109, no. 47, pp. E3212–E3220, 2012.

[4] L. Qian, E. Winfree, and J. Bruck, "Neural network computation with DNA strand displacement cascades," *Nature*, vol. 475, pp. 368–372, 2011.

[5] A. Padirac, T. Fujii, and Y.Rondelez, "Nucleic acids for the rational design of reaction circuits," *Curr. Opin. Biotechnol.*, vol. 24, no. 4, pp. 575–580, 2012.

[6] M. N. Stojanovic and D. Stefanovic, "A deoxyribozyme-based molecular automaton," *Nat. Biotechnol.*, vol. 21, pp. 1069–1074, 2003.

[7] J. Kim, K. S. White, and E. Winfree, "Construction of an in vitro bistable circuit from synthetic transcriptional switches," *Mol. Syst. Biol.*, vol. 2, no. 68, 2006. [Online]. Available: http://dx.doi.org/10.1038/msb4100099

[8] G. Seelig, D. Soloveichik, D. Y. Zhang, and E. Winfree, "Enzyme-free nucleic acid logic circuits," *Science*, vol. 314, no. 5805, pp. 1585–1588, 2006.

[9] T. Fujii and Y. Rondelez, "Predator-prey molecular ecosystems," *ACS Nano*, vol. 7, no. 1, pp. 27–34, 2013.

[10] A. Padirac, T. Fujii, A. E. Torres, and Y. Rondelez, "Spatial waves in synthetic biochemical networks," *J. Am. Chem. Soc.*, vol. 135, no. 39, pp. 14 586–14 592, 2013.

[11] D. Soloveichik, G. Seelig, and E. Winfree, "DNA as a universal substrate for chemical kinetics," *Proc. Natl. Acad. Sci. USA*, vol. 107, no. 12, pp. 5393–5398, 2010.

[12] R. Daniel, J. R. Rubens, R. Sarpeshkar, and T. K. Lu, "Synthetic analog computation in living cells," *Nature*, vol. 497, pp. 619–623, 2013.

[13] L. Qian and E. Winfree, "Scaling up digital circuit computation with DNA strand displacement cascades," *Science*, vol. 332, no. 6034, pp. 1196–1201, 2011.

[14] N. E. Buchler, U. Gerland, and T. Hwa, "On schemes of combinatorial transcription logic," *Proc. Natl. Acad. Sci. USA*, vol. 100, no. 9, pp. 5136–5141, 2003.

[15] A. Hjelmfelt, E. D. Weinberger, and J. Ross, "Chemical implementation of neural networks and turing machines," *Proc. Natl. Acad. Sci. USA*, vol. 88, no. 24, pp. 10 983–10 987, 1991.

[16] A. J. Genot, T. Fujii, and Y. Rondelez, "Scaling down DNA circuits with competitive neural networks," *J. R. Soc. Interface*, vol. 10, no. 85, 2013. [Online]. Available: http://dx.doi.org/10.1098/rsif.2013.0212

[17] H. deJong, "Modeling and simulation of genetic regulatory systems: a literature review," *J. Comput. Biol.*, vol. 9, no. 1, pp. 67–103, 2002.

[18] L. Bintu, N. E. Buchler, H. Garcia, U. Gerland, T. Hwa, J. Kondev, and R. Philips, "Transcriptional regulation by the numbers: models," *Curr. Opin. Genet. Dev.*, vol. 15, no. 2, pp. 116–124, 2005.

[19] M. E. Wall, W. S. Hlavacek, and M. A. Savageau, "Design of gene circuits: lesson from bacteria," *Nat. Rev. Genet.*, vol. 5, pp. 34–42, 2004.

[20] J. Hasty, D. McMillen, and J. J. Collins, "Engineered gene circuits," *Nature*, vol. 420, no. 6912, pp. 224–230, 2002.

[21] P. Francois and V. Hakim, "Design of genetic networks with specified functions by evolution in silico," *Proc. Natl. Acad. Sci. USA*, vol. 101, no. 2, pp. 580–585, 2004.

[22] K. Fujimoto, S. Ishihara, and K. Kaneko, "Network evolution of body plans," *PLOS ONE*, vol. 3, no. 7, 2008. [Online]. Available: http://dx.doi.org/10.1371/journal.pone.0002772

[23] Y. Kobayashi, T. Shibata, Y. Kuramoto, and A. Mikhailov, "Evolutionary design of oscillatory genetic networks," *Eur. Phys. J. B.*, vol. 76, pp. 167–178, 2010.

[24] A. Deckard and H. M. Sauro, "Preliminary studies on the in silico evolution of biochemical networks," *ChemBioChem*, vol. 5, no. 10, pp. 1423–1431, 2004.

[25] S. R. Paladugu, V. Chickarmane, A. Deckard, J. P. Frumkin, M. McCormack, and H. M. Sauro, "In silico evolution of functional modules in biochemical networks," *Syst Biol (Stevenage)*, vol. 153, no. 4, pp. 223–235, 2006.

[26] M. A. Marchisio and J. Stelling, "Automatic design of digital synthetic gene circuits," *PLoS Comput. Biol.*, vol. 7, no. 2, 2011. [Online]. Available: http://dx.doi.org/10.1371/journal.pcbi.1001083

[27] H. Cao, F. J. Remoro-Campero, S. Heeb, M. Cmara, and N. Krasnogor, "Evolving cell models for systems and synthetic biology," *Syst Synth Biol.*, vol. 4, no. 1, pp. 55–84, 2010.

[28] Y. Jin and B. Sendhoff, "Evolving in silico bistable and oscillatory dynamics for gene regulatory network motifs," in *IEEE Congr. Evolutionary Computation*, Hong Kong, China, Jun. 2008, pp. 386–391.

[29] J. S. Hallinan, G. Misirli, and A. Wipat, "Evolutionary computation for the design of a stochastic switch for synthetic genetic circuits," in *Conf. Proc. IEEE Eng. Med. Biol. Soc.*, Buenos Aires, Argentina, Sep. 2010, pp. 768–774.

[30] B. Drennan and R. Beer, "Evolution of repressilators using a biologically-motivated model of gene expression," in *Artificial Life X: Proc. Int. Conf. Simulation and Synthesis of Living Systems*. The MIT Press, Aug. 2006, pp. 22–27.

[31] M. B. Elowitz and S. A. Leibler, "Synthetic gene oscillatory network of transcriptional regulators," *Nature*, vol. 403, pp. 335–338, 2000.

[32] B. D. Ventura, C. Lemerle, K. Michalodimitrakis, and L. Serrano, "From in vivo to in silico biology and back," *Nature*, vol. 443, pp. 527–533, 2006.

[33] A. H. Chau, J. M. Walter, J. Gerardin, C. Tang, and W. A. Lim, "Designing synthetic regulatory networks capable of self-organizing cell polarization," *Cell*, vol. 151, no. 2, pp. 320–332, 2012.

[34] A. J. Genot, T. Fujii, and Y. Rondelez, "In vitro regulatory models for systems biology," *Biotech Adv*, vol. 31, no. 6, pp. 789–796, 2013.

[35] S. Doncieux, J. B. Mouret, N. Bredeche, and V. Padois, "Evolutionary robotics: Exploring new horizons," in *New Horizons in Evolutionary Robotics, Studies in Computational Intelligence*. Springer, 2011, pp. 3–25.

[36] K. O. Stanley and R. Miikkulainen, "Evolving neural networks through augmenting topologies," *Evol. Comput.*, vol. 10, no. 2, pp. 99–127, 2002.

[37] T. Aaltonen *et al.*, "Observation of electroweak single top-quark production," *Phys. Rev. Lett.*, vol. 103, no. 9, 2009. [Online]. Available: http://dx.doi.org/10.1103/PhysRevLett.103.092001

[38] N. Aubert, C. Mosca, T. Fujii, M. Hagiya, and Y. Rondelez, "Computer assisted design for scaling up systems based on DNA reaction networks," *J. R. Soc. Interface*, vol. 11, no. 93, 2014. [Online]. Available: http://dx.doi.org/10.1098/rsif.2013.1167

[39] R. Storn and K. Price, "Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optim.*, vol. 11, no. 4, pp. 341–359, 1997.

[40] B. Novak and J. J. Tyson, "Design principles of biochemical oscillators," *Nat. Rev. Mol. Cell Biol.*, vol. 9, pp. 981–991, 2008.

[41] J. B. Mouret and S. Doncieux, "Encouraging behavioral diversity in evolutionary robotics: An empirical study," *Evol. Comput.*, vol. 20, no. 1, pp. 91–133, 2012.

[42] N. Noman and H. Iba, "Accelerating differential evolution using an adaptive local search," *IEEE Trans. Evol. Comput.*, vol. 12, no. 1, pp. 107–125, 2008.

[43] Y. Rondelez, "Competition for catalytic resources alters biological network dynamics," *Phys. Rev. Lett.*, vol. 108, no. 1, 2012. [Online]. Available: http://dx.doi.org/10.1103/PhysRevLett.108.018102

[44] A. J. Genot, T. Fujii, and Y. Rondelez, "Computing with competition in biochemical networks," *Phys. Rev. Lett.*, vol. 109, no. 20, 2012. [Online]. Available: http://dx.doi.org/10.1103/PhysRevLett.109.208102

[45] K. H. ten Tusscher and P. Hogeweg, "Evolution of networks for body plan patterning; interplay of modularity, robustness and evolvability," *PLoS Comput. Biol.*, vol. 7, no. 10, 2011. [Online]. Available: http://dx.doi.org/10.1371/journal.pcbi.1002208

[46] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*. NJ: Wiley, May 2001.

[47] A. Warmflash, P. Francois, and E. D. Siggia, "Pareto evolution of gene networks: an algorithm to optimize multiple fitness objectives," *Phys. Biol.*, vol. 9, no. 5, 2012. [Online]. Available: http://dx.doi.org/10.1088/1478-3975/9/5/056001

[48] A. E. Eiben, R. Hinterding, and Z. Michalewicz, "Parameter control in evolutionary algorithms," *IEEE Trans. Evol. Comput.*, vol. 3, no. 2, pp. 124–141, 1999.

[49] A. E. Eiben and S. K. Smit, "Parameter tuning for configuring and analyzing evolutionary algorithms," *Swarm and Evolutionary Computation*, vol. 1, no. 1, pp. 19–31, 2011.

Quang Huy Dinh received the B.S. (2008) and M.S. (2011) in Computer Science from the University of Engineering and Technology, Vietnam National University, Hanoi. He is currently working towards the Ph.D. degree in Computer Science at the Graduate School of Information Science and Technology, University of Tokyo, Japan. His research interests include evolutionary computation, synthetic biology, and bioinformatics.

Nathanael Aubert received the B.S. (2008) and M.S. (2011) in Computer Science from the University of Rennes 1, France. He is currently pursuing the Ph.D. degree in Computer Science from the University of Tokyo, Japan. His research interests includes the development of computer assisted design tools for DNA computing, evolutionary algorithms applied to biochemical systems and agent-based modeling of artificial life systems.

Nasimul Noman , PhD: He received the PhD degree in frontier informatics from the Graduate School of Frontier Sciences, University of Tokyo. He served as a faculty member in the Department of Computer Science and Engineering, University of Dhaka. He also worked as a research fellow in the Graduate School of Information Science and Technology in University of Tokyo. Now he is working as a lecturer in the School of Electrical Engineering and Computer Science, University of Newcastle, Australia. He is a member of the editorial board in BioMed Research International journal. His research interests include Evolutionary Computation, Computational Biology, Systems Biology and Synthetic Biology.

Teruo Fujii (S'89-M'93-SM'06) received his Ph.D. degree in engineering from the University of Tokyo in 1993. From 1993 to 1995, he was Associate Professor (TOYOTA Endowed Chair) in globe engineering at the Institute of Industrial Science (IIS), University of Tokyo. He was with the RIKEN Institute from 1995 to 1999. In 1999, he was appointed as Associate Professor at Underwater Technology Research Center of IIS. In 2006 he moved to the Center for International Research on MicroMechatronics (CIRMM-IIS) and was promoted to Full Professor in 2007. In 2014, he joined the newly founded center named Center for International Research on Integrative Biomedical Systems (CIBiS-IIS). Since 2007, he has also been serving as Japanese Co-director of the Laboratory for Integrated MicroMechatronic Systems (LIMMS), which is a joint research laboratory between CNRS, France and IIS, University of Tokyo. Since 2013, he has been a deputy director of IIS, University of Tokyo. His research interests include microfluidics and its applications.

Yannick Rondelez received his Ph.D. degree in chemistry from the University of Paris XI in 2002, for his study of supra-molecular models of metalloenzymes. He was then a post-doctoral fellow at the Institute of industrial Science (IIS), University of Tokyo, focusing on he biophysics of natural molecular motors. He worked for a private company in the consulting sector for a few years and entered the French CNRS in 2008. He also holds a position of Associate Professor of the University of Tokyo, where he is currently located. His current research involves the programming of *in vitro* molecular systems to perform complex function and/or information processing. He also has an activity in applying concepts from molecular programming back to biological questions.

Hitoshi Iba , Ph.D.: He received Ph.D. degree from Dept. Engineering of University of Tokyo in 1990. Since then, he had been a researcher in ElectroTechnical Lab. He became an associate professor of Department of Electronial Engineering at the University of Tokyo in April, 1998. He is currently a professor of Department of Information and Communication Engineering, Graduate School of Information Science and Technology at the University of Tokyo. He is an associate editor of IEEE tr. on EC and Journal of Genetic Programming and Evolvable Machines (GPEM). His research interests include Evolutionary Computation, Genetic Programming, Bio-informatics, Foundation of Artificial Intelligence, Machine Learning, Robotics, Vision.