Evolution of Analog Networks using Local String Alignment on Highly Reorganizable Genomes

Claudio Mattiussi
Autonomous Systems Laboratory
ASL-I2S-STI-EPFL
1015 Lausanne, Switzerland
claudio.mattiussi@epfl.ch

Dario Floreano
Autonomous Systems Laboratory
ASL-I2S-STI-EPFL
1015 Lausanne, Switzerland
dario.floreano@epfl.ch

Abstract

We introduce and apply a genetic representation for analog electronic circuits based on the association of character strings extracted from the genome with the terminals and parameters of components, and the use of local string alignment to generate the connection between components. The representation produces a variable genome length structure that tolerates the execution of major genome reorganization operators such as duplication and transposition, along with less disruptive ones such as character insertion, deletion and substitution. The representation can be applied also to other analog networks such as artificial neural networks, control systems, and genetic regulatory networks.

1. Introduction

Many biological and artificial systems are networks, or are controlled by networks. Familiar examples are metabolic, signaling, and genetic regulatory networks; biological and artificial neural networks; electrical and electronic circuits; and any system containing one of these. The functionality of such systems depends on the number and properties of the nodes of the network, and on the characteristics of the links connecting the nodes. The possibility to alter all these aspects is therefore a plus point for an evolutionary process dealing with these systems.

An evolutionary algorithm aimed at the synthesis of networks typically obtains them through the decoding of a genome. If we assume an unchanging decoding process during the course of evolution, the reorganization of the network follows from the reorganization of the genome. It is easy to define genetic operators that radically reorganize a genome, but we must keep in mind that the genome must have a reasonable probability of remaining decodable. Moreover, besides the genome reorganizations that result in major restructuring of the network, it is desirable that the genetic operators possess also a certain degree of smoothness in their effect, to give the evolutionary process the possibility to test slight variations on a given network. For all these reasons, the definition of the structure of the genome, of the genetic operators, and of the decoding process that transforms the genome into a network – in a word, the genetic *representation* of the network – is in itself a challenging task [2, 3, 8, 9]. This is especially true if one wants to favor the open-endedness of the evolutionary process by limiting to a minimum the constraints imposed on the topology of the network and on the properties of its nodes and links.

In this paper, we define a genetic representation that fulfills all the desiderata listed above for a particular kind of networks, which we call generically analog networks. The nodes of this kind of networks can belong to a finite number of basic types, each endowed with a finite collection of evolvable parameters. A node is connected to the other nodes through links characterized by a scalar parameter, which represent the strength of the interaction between the nodes. This strength can vary gradually from the absence of interaction to one with maximal strength.

Analog electronic circuits are a first example of this kind of network, where the nodes are electronic components such as transistors and capacitors, and the value of resistance of the resistors connecting the terminals of the components define the strength of the interaction. Artificial neural networks are another example, where the nodes are the artificial neurons and their input weights define the strength of the interaction. A genetic regulatory network is a biological example of analog network, where the genes act as nodes of the network, and the degree of influence of the biomolecules produced by one gene on the expression of another gene defines the strength of their interaction. Control systems in their functional block representation are another example, and many other examples of analog networks exist. In this paper, all the examples and discussions are based on analog electronic circuits but the adaptation of this representation to other kinds of analog networks is straightforward and is currently under way for neural networks. The evolvability of these other analog networks with the proposed representation, however, remains to be proved.

2. Genetic representation and decoding

In our genetic representation the genome is composed by a finite number of character strings - the *chromosomes*. The characters of the chromosomes are called *nucleotides* and belong to a finite *genetic alphabet*, typically, a subset of the ASCII character set.

2.1. Representing non-resistive components.

The basic idea of the genetic representation for the case of analog electronic circuits is the association of a string extracted from the genome – a *label* – with each terminal of the non-resistive components that can appear in the circuit. Each component can also have a finite number of evolvable parameter values, for example the value of capacitance of a capacitor. With each evolvable parameter, we also associate a string extracted from the genome (Figure 1). These strings are used to define the connections between the terminals and assign the parameter values, as explained below.

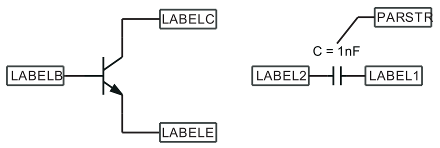


Figure 1. To construct the genetic representation of an analog electronic circuit, we associate a *label* with each terminal of the non-resistive components. A *parameter string* is associated with each evolvable parameter of the components.

To represent in the genome a component with its associated terminal labels and parameters strings, we first define for each component a particular string or token that will uniquely identify it. For example, assuming as genetic alphabet the set of uppercase letters of the ASCII character set, we associate as token the string "NBJT" to NPN bipolar transistors, and the string "CAPA" with capacitors. Then, we define a token that will act as delimiter for the terminal labels, for example the string "LEAD". Finally, we define a token that will act as a delimiter for the substrings associated with the evolvable parameters of the components, for example the string "VALU". Figure 2 shows an example of a fragment of genome containing all these tokens.

2.2. Decoding the components.

To extract the components and their associated strings from the genome we proceed as follows. We start by locating the tokens that identify the components. Then, in the fragment of genome that follows the identifying token we search for the tokens that delimit the terminal labels and parameter strings. For example, once we have located the "NBJT" token of a transistor, we know that we need three terminal labels, whereas for a capacitor we need two terminal labels and one parameter string. If the required lead and value delimiter tokens cannot be found in the genome, no component is extracted from the genome, even if a legal identifier is present. If all the tokens are found, the corresponding component is created in the circuit (for the moment, unconnected) and the substrings delimited by the tokens are extracted and associated with its terminals and evolvable parameters. If the terminals of the component are not interchangeable, an order is defined for the association, for example (C)→(B)→(E) for the terminals of a BJT, or (+)→(−) for polarized capacitors, and the association follows the order in which the labels appear in the genome. To enlarge the set of decodable genomes, when both terminal labels and parameter strings are required they are left free to mix in any order. Fragments of genome that cannot be decoded meaningfully are just left undecoded as "junk" genome. This decoding technique for components applies to genomes with an arbitrary number of chromosomes.

...TVKUANXQOGIIOCNIZUARXJVFAZKEWJ
NBJT *LSILWUPIEDUNO* **LEAD** *RTCCICWGT* **LEAD** *YK*
RSTZEPTT **LEAD** BKVEYDZGTMJADBVKPVVIFYDU**C**
APA *SMAUSDH* **LEAD** *TRGDNZCZ* **VALU** *VUDDQPAY*
DG **LEAD** ZVBKGGST...

Figure 2. A fragment of genome encoding an NPN BJT and a capacitor with evolvable capacitance value. The "NBJT" token signals the start of the fragment coding for the BJT, and the token "CAPA" the start of that coding for the capacitor. The label *"LSILWUPIEDUNO"* delimited by "NBJT" and "LEAD" is associated with the first terminal of the transistor, which are assumed to be assigned a predefined order, for example (C)→(B)→(E). The label *"RTCCICWGT"* delimited by the first and second "LEAD" tokens is associated with the second terminal, and so on. For the capacitor, the parameter string *"TRGDNZCZ"* terminated by the "VALU" token is associated with the capacitance value.

The procedure of component extraction just described permits the overlapping of strings corresponding to different components. For example, if after the identifier NBJT and before the "CAPA" one in

Figure 2 we delete the last "LEAD" delimiter, the whole string "YKRSTZEPTTBKVEYDZGTMJADB-VKPVVIFDU**CAPA**SMAUSDH" is associated with the third terminal of the transistor. Although in principle this is acceptable, it generates an interaction between components, and this could complicate the evolutionary process. For this reason, in the experiments reported below, we excluded the possibility of component overlapping by considering associated with a component token only the fragment of genome that goes from the end of the current component token to the start of the next one. Note however, that gene overlapping is known to occur in natural genomes [4]. The potentialities of component overlapping will therefore be also explored in future experiments.

2.3. Connecting the components.

At this point, we have obtained from the genome a collection of components having strings associated with their terminals and with their evolvable parameters. To derive a network from these components we must connect them. This we do by inserting a resistor between each pair of terminals. The value of this resistor is obtained by comparing the labels associated with the terminals and calculating for each pair of labels a nonnegative integer value s, their local alignment score (Figure 3).

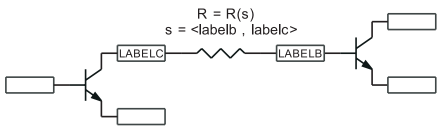


Figure 3. The connection between labeled terminals is established with a resistor whose resistance value is determined by the *local alignment score s* of the two labels.

We will give the definition of s in Section 4. For the time being, it is sufficient to know that is determined by the similarity of the two labels. To define the strength of the connection between terminals, we map the values of s into a set of resistance values $\{R_{min}, \ldots, R_{max}\}$. Figure 4 gives an example of this mapping. To the values of this series, we add the zero-valued resistor $R_0 = 0$, which corresponds to a direct connection between two terminals, and the infinite-valued resistor $R_\infty = +\infty$, which corresponds to the absence of a direct connection between them. The ranges of values of s above that associated with R_{min} and the range below that associated with R_{max}, are associated with R_0 and R_∞.

2.4. Assigning parameter values.

We said above that a component successfully extracted from the genome has a string associated with each of its

```
(68)1.00E+00 (67)1.33E+00 (66)1.78E+00 (65)2.37E+00
(64)3.16E+00 (63)4.22E+00 (62)5.62E+00 (61)7.50E+00
(60)1.00E+01 (59)1.33E+01 (58)1.78E+01 (57)2.37E+01
(56)3.16E+01 (55)4.22E+01 (54)5.62E+01 (53)7.50E+01
(52)1.00E+02 (51)1.33E+02 (50)1.78E+02 (49)2.37E+02
(48)3.16E+02 (47)4.22E+02 (46)5.62E+02 (45)7.50E+02
(44)1.00E+03 (43)1.33E+03 (42)1.78E+03 (41)2.37E+03
(40)3.16E+03 (39)4.22E+03 (38)5.62E+03 (37)7.50E+03
(36)1.00E+04 (35)1.33E+04 (34)1.78E+04 (33)2.37E+04
(32)3.16E+04 (31)4.22E+04 (30)5.62E+04 (29)7.50E+04
(28)1.00E+05 (27)1.33E+05 (26)1.78E+05 (25)2.37E+05
(24)3.16E+05 (23)4.22E+05 (22)5.62E+05 (21)7.50E+05
(20)1.00E+06
```

Figure 4. The logarithmically distributed set of resistance values used in the experiments of circuit evolution described at the end of the paper. It covers 6 decades with 8 values per decade, from 1Ω to 1MΩ. The numbers in parentheses are the alignment scores associated with the values. The whole range of scores below 20 is associated with an infinite-valued resistor (no connection) and that above 68 is associated with a zero-valued resistor (direct connection).

evolvable parameters. To assign a value to those parameters we use once again the technique of calculating an alignment score s and mapping it to a value from a discrete set, for example a capacitance $C = C(s)$. Since in this case we have only one string per parameter extracted from the genome, we must define a fixed string that will be used to calculate the alignment score of the parameter string relatively to the fixed string. The fixed string can be the same for all kind of parameter values, or may be specialized for each of them.

Note that we could define parameter values by encoding directly integers in the genome and mapping them into the set of parameter values. However, we opted for the same kind of encoding for connections and parameter values in the effort of obtaining a similar behavior for the connection strengths and the parameter values relatively to the action of the genetic operators.

3. External connections and compartments.

To perform a useful function a circuit must be connected to the external world to provide at least an output signal. More commonly the circuit receives also some input signal and power. This corresponds to the presence of some components that must be considered external to the evolved circuit [2, 9], for example a fixed voltage source as power supply, a signal generator with its source resistance, and a load. To complete the definition of our genetic representation we must specify how these external components are connected to the decoded circuit.

A simple way to extend the decoding of the connectivity defined above to external components consists in associating a fixed, predefined label with each terminal of the external components that must be connected to the decoded circuit. Then, these labels can be added to the collection of those of extracted from the genome, and used to determine the connecting resistors. Although simple, this approach requires some care in the choice of the fixed labels, to assure that all the range of alignment scores that goes from R_0 to R_∞ can be actually obtained.

3.1. I/O ports

A solution for the connection of external components to the decoded circuit that does not require the choice and association of fixed labels to the external terminals, is based on the introduction of new kind components called *I/O Ports*. Each external node that can be connected to the decoded circuit is associated with a particular I/O Port. This I/O Port is identified in the genome by a token. For example, if there are three external nodes to connect we introduce three tokens “IOPTA”, “IOPTB”, and “IOPTC”. Each time one of these tokens followed by a “LEAD” terminated label is found in the genome, it is associated with corresponding external node. These labels are then added to the set of those associated with the components extracted from the genome, to derive the connecting resistors as described above. This is the approach used in the experiment reported below. Figure 5 shows an example of genome and circuit corresponding to the application of this technique.

…SKM**NBJT***FGVKS***LEAD***XCZVXYV***LEAD***OGCIF***LEAD**UHX
XCHTVCC**IOPTB***RMMYP***LEAD**DQSKMJVQR**IOPTB***IXQSTA*
*UX***LEAD**FCKVF**CAPACIT***NXERACQX***LEAD***PIAFGRSLU***LE**
AD*AZELON***VALU**SQNYSFXHFD**IOPTC***PQYWAJ***LEAD**ZQE
KGX**IOPTA***QYCGZCLN***LEAD**ZQEKGX**IOPTB***RBUIA***LEAD**P
MCZCZXSDUNUVMYCC**IOPTC***CHBIXUQOO***LEAD**SKRK…

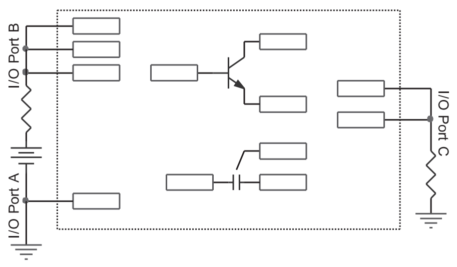


Figure 5. An example of application of the technique of connection of external components to the decoded circuit (bottom) by means of *I/O Ports* associated with the external terminals connected to the decoded circuit. The fragment of genome represented above (top) contains the tokens “IOPTA”, “IOPTB” and “IOPTC” corresponding to the three connected external nodes. The substrings of the genome delimited by these tokens and by the “LEAD” token are associated to the terminals of the external components.

3.2. Compartments

The idea of distinguishing the components of the evolved circuit from the fixed external ones can be extended to give evolution the possibility to generate modular and hierarchical circuits. To this end, it is sufficient to add to the terminals of the components an evolvable marker that identifies the compartment to which each terminal belongs. In this way, the generation of the connecting resistors can be limited to pairs of terminals having the same mark. Alternatively, the whole components can be marked and the connections between compartments can be obtained introducing a new kind of two-terminal component connecting different compartments. This approach, besides giving the possibility to evolve modular and hierarchical networks, limits the rapid growth of the number of comparisons of labels that are required to decode the circuit when the number of components grows.

3.3. Evolving the evolutionary parameters

One of the banes of evolutionary algorithms is the presence of a multitude of parameters whose value the user, often through a process of trial and error, must assign. A possible solution is the encoding of some of those parameters – typically those governing the operation of genome reorganization – in the genome, to let the evolutionary process select those values [1]. There is some biological evidence, however, that at least for unicellular organisms, the extent of genome reorganization during reproduction depends also on the state of the phenotype and is not simply hardwired in the genome [6].

With the genetic representation described above, we can easily implement this kind of dynamic generation of the parameters. For example, we can introduce a fixed resistive voltage divider connected to the power supply for each parameter. The output voltage of the divider defines the value of the parameter, and is included in the set of nodes that can be connected to the evolved circuit. When the output node is not connected to the rest of the circuit, the output voltage gives the default value assigned to the parameter by the user. The evolutionary process, however, can generate a connection to the node and vary this default value.

4. Local alignment of strings

We anticipated above that the value of resistance inserted between the terminals of the decoded components depends on the similarity of the labels associated with the terminals.

These labels are strings that have been extracted from the genome, where in the course of evolution they have been subjected to the action of the genetic operators. Since we are willing to admit major reorganizations of the genome, the length of the labels will in general vary while evolution proceeds. Therefore, we cannot assume that these strings have equal length. This prevents us from basing our definition of similarity on the familiar Hamming distance, which counts the number of positions where two strings of characters of equal length differ.

We can generalize the Hamming distance by supposing that besides *substitutions* of characters, one string can be transformed into another using *insertions* and *deletions* of characters. We can then assign a nonnegative weight $w_s(x, y)$ to each substitution of a generic character x with a character y, and nonnegative weight $w_{di}(x)$ to the deletion or insertion a character x. We can transform one string into another using insertion, deletion and substitution, and associate with the transformation the sum of the weights of the operations performed. The *global alignment distance* between two strings is defined as the minimum value of that sum for all the transformations leading from one string to the other. The correspondence established by each transformation is called a *global alignment* of the two strings [5].

This definition satisfies our need to operate with strings of different length. However, the fact that it puts in correspondence the whole length of the two strings appears as a drawback for our application. This follows from the fact that in general one component terminal can be connected to more than one other terminal. We would like evolution to deal with each connection independently, whereas, by forcing the establishment of a correspondence of the whole length of both strings, the use of the global alignment distance would tend to frustrate this independence.

We can solve this problem by considering alignments between portions of the two labels. This corresponds to the notion of *local alignment* of strings. Now we can no longer reason in terms of distance and of the minimum sum of weights of the operations that transforms a portion of a string into a portion of the other, because this minimum would be usually attained for very short portions of the strings, for example a single character. We must therefore substitute the nonnegative weights $w_s(x, y)$ and $w_{di}(x)$ defined above, with *matching scores* $s_s(x, y)$, $s_{di}(x)$ for substitutions, deletions and insertions of characters. Contrary to the weights, which penalize dissimilarities, the matching scores reward the similarities with positive values and penalize the dissimilarities with less positive or negative ones. Based on the matching scores we can define the *local alignment score* of two strings as the maximum value of the sum of the scores that can be obtained for a transformation that leads from a substring of one string to a substring of the other [5]. Using this definition to evaluate the similarity of two labels, we obtain the desired result that a label can evolve multiple separate regions that independently match many different other labels (Figure 6).

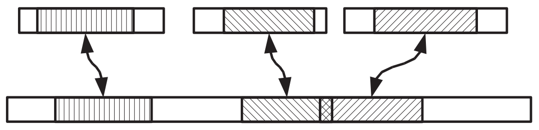


Figure 6. Using local alignment of strings, one label can match independently many other labels with multiple separate matching regions, although overlapping of the matching regions is still possible.

4.1. Calculating the local alignment score

A naïve approach to the calculation of the local alignment score of two strings requires the generation of all possible transformations of all substrings of a string into all substrings of the other. This is computationally unfeasible for even moderate string lengths. There are however algorithms based on dynamic programming whose computation time grows like mn, where m and n are the lengths of the two strings [5]. Using these algorithms the local alignment score of two strings some thousands of characters long takes a few milliseconds on present-day Personal Computers.

5. Genetic operators

The genetic representation for electronic circuits described above permits the execution of many kinds of reorganizations of the genome without compromising its decodability. The only practical limit that must be imposed on the genome concerns the length of the labels and parameter strings, to ensure that the string alignment algorithms are practically executable in reasonable time and memory bounds. Apart from this limit, the fragments of the genome coding for components can be of variable length and be located anywhere within the genome. The genome itself is obviously a variable length genome.

This opens the way to the introduction in our evolutionary algorithms of many genetic reorganization operations that are seldom used in artificial evolution experiments (although they are known to be common in the evolution of biological genomes [4, 6]). In particular, the genetic representation introduced above allows

- Operations on nucleotides, such as insertion, deletion, and substitution of single nucleotides.
- Operations on chromosome fragments, such as duplication, deletion, transposition of fragments of chromosomes, creation of components, and recombination (crossover) of pairs of chromosomes.

- Operations on chromosomes, such as duplication and deletion of whole chromosomes.
- Operations on the whole genome, such as the duplication of the whole genome.

6. Experiments

We report here the results of three series of experiments: string alignment experiments, circuit matching experiments, and circuit evolution experiments.

6.1. String alignment and circuit matching

The goal of the string alignment and circuit matching experiments was the collection of information about the process of local string alignment in an evolutionary context. This information guided the selection of the genetic alphabet, of the matching scores for local string alignment, and of the set of genome reorganization operators for the subsequent experiments. The string alignment experiments started with the random generation of a collection of *target strings* of different lengths. To each target string was assigned a target matching score, and a population of genomes was randomly generated and evolved with the aim of obtaining a genome matching all the target strings with the assigned target scores. The fitness function was the Euclidean distance of the vector of target scores from the vector of evolved matching scores.

The circuit matching experiments are a variant of the sequence alignment experiments where the target strings are not randomly generated. The main objective was the exploration of the evolutionary potentialities of the system and the possibility to generate sets of strings with preassigned alignment properties that reproduce those required by typical analog electronic circuits. We defined the problem for those experiments by choosing a simple circuit, such as an elementary oscillator or amplifier. Then we computed the equivalent resistance between the terminals of the non-resistive components of the circuit. We assigned the mapping $R(s)$ described in Section 2.3 between alignment scores and connecting resistances, and gave to the evolutionary process the goal of generating a collection of strings with alignment scores mapping to the terminal-to-terminal equivalent resistances of the circuit. In other words, the fitness was determined only by the distance between the vector of evolved resistances and the vector of the target circuit resistances; the functionality of the circuit with the evolved set of resistances was not even evaluated.

Without entering into the details of the experiments, the main results obtained are the observation that the presence of certain genome reorganization operators is instrumental to the success of an evolutionary string alignment process. In particular, we observed frequently a phenomenon of overlapping of the regions of the genome devoted to the alignment of two or more target strings (Figure 7, top). An analogous phenomenon of *overlapping genes* is observed in biological genomes [4]. In the absence of operators of duplication of chromosome fragments or of the whole genome, this phenomenon stalls the evolution, since typically at a certain point any improvement in the alignment of one target string is detrimental to that of the other. The duplication of the fragment of genome where the overlap occurs, allows evolution to operate separately on two now independent matching regions, whereas the additional copies of these regions thus generated are no longer under evolutionary pressure and become "pseudogenes" (Figure 7, bottom). Figure 8 shows the progress of an experiment where this phenomenon of overlapping was observed in the genome, and could be overcome by assigning a positive value to the probability of genome duplication.

Summing up, using a genetic representation based on string alignment, operators of major genome reorganization are not only a possibility: they are a necessity.

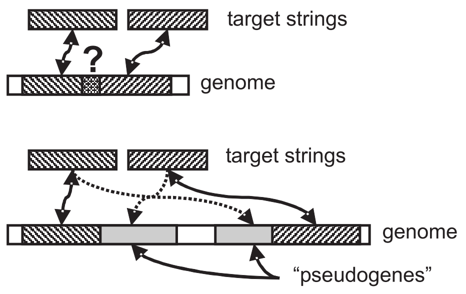


Figure 7. The presence of suitable genetic operators is instrumental to the disentanglement of situations that can bring evolution to a standstill. Here, an overlap of genes (top) is solved by the duplication of a fragment of genome (bottom). The copies of the original genes are then liberated from evolutionary constraints and are probably bound to become pseudogenes.

6.2. Circuit evolution experiments

To implement circuit evolution, each genome was decoded into a preliminary circuit, following the steps described above. The components having all their terminals coincident where removed from this preliminary circuit, and the passive components in parallel where coalesced. The resulting circuit was simulated with SPICE [7].

At the time of this writing, the results of only one kind of circuit evolution experiment were available, aimed at the evolution of a voltage reference analogous to the one described in [2]. The external circuit is the one represented in Figure 5, where the voltage of the power supply (left) can

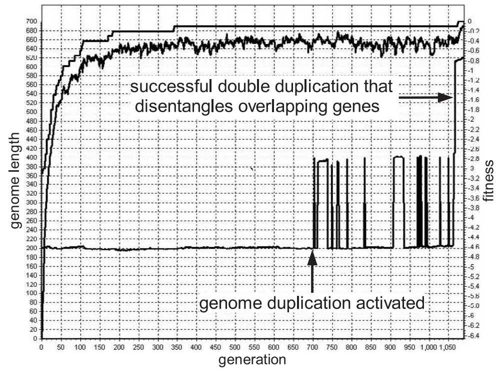
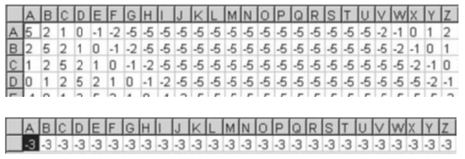


Figure 8. An evolutionary string alignment brought to a standstill by gene overlap, which was observed inspecting the genome and is witnessed here by the stagnation of the fitness (top curve) relatively to its maximum achievable value of zero. Following the activation of the genome duplication operator at generation 700, the maximum genome length (bottom curve) shows a series of short-lived episodes of genome duplication. Eventually, a successful double episode of genome duplication around generation 1050 eliminates the overlap and lets the fitness attain its maximum value.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	5	2	1	0	-1	-2	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-2	-1	0	1	2
B	2	5	2	1	0	-1	-2	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-2	-1	0	1
C	1	2	5	2	1	0	-1	-2	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-2	-1	0
D	0	1	2	5	2	1	0	-1	-2	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-2	-1

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
	-3	-3	-3	-3	-3	-3	-3	-3	-3	-3	-3	-3	-3	-3	-3	-3	-3	-3	-3	-3	-3	-3	-3	-3	-3	-3

Figure 9. A partial view of the matrix of matching scores for nucleotide substitution (top) and the complete vector of nucleotide deletion and insertion scores (bottom). The hidden lines of the matching scores matrix repeat cyclically the pattern of those shown.

vary from 4V to 6V, the source resistance is 1kΩ, and the load resistance (right) is 10kΩ. The goal is the generation of a constant 2V voltage across the load when the temperature varies from 0°C to 100°C. To this end, the decoded circuits where simulated with a power supply voltage varying from 4V to 6V in steps of 0.1V, with simulation temperatures of 0°C, 25°C, 50°C, 75°C, and 100°C. For each power supply voltage and circuit temperature, we computed the square of the difference between the actual voltage on the load and the required output voltage. The fitness was defined as the opposite of the sum of all these squares, so that the goal was the maximization of the fitness, with optimal value zero. The genome of all the individuals of the initial population was constituted by one chromosome containing 10 NPN BJT descriptors and two I/O Ports for each of the three connected external nodes. The terminal labels for all the components had an initial length of five characters, randomly filled with elements of the genetic alphabet, which corresponds to the set of uppercase alphabetic ASCII chars. The matching scores were those shown in Figure 9, and the mapping of the alignment scores to the connecting resistors were those shown in Figure 4.

We used a genetic algorithm with tournament selection, tournament size of 5, and elitism. The size of the population was 100. The probabilities of nucleotide insertion, deletion, and substitution, those of chromosome duplication and deletion and the probability of genome duplication were set to 0.001. The probabilities of chromosome fragment duplication, deletion, transposition, and that of chromosome single point crossover were set to 0.01. Chromosome fragment reorganization was performed by selecting two random points in the source chromosome to define the fragment, and one random point in the target chromosome, when required. Figure 10 shows the best result obtained in four runs of evolution of 10000 generations each. This run achieved a fitness of -0.0062. The other runs achieved a fitness of -0.018 and -0.019 – which correspond to good regulation –, and -2.2 – which does not. The opportunistic nature of the evolutionary process leads to the regulation being based on the particular values of the parameters of the transistor model used in the simulations. Thus, the circuit cannot be expected to work with non-selected components. This problem can be addressed using Monte Carlo runs to evaluate the fitness. Figure 11 shows the performance of the best evolved circuit and Figure 12 shows the SPICE input file (with a fictitious obsolete TEMP option including the five actual simulation temperatures).

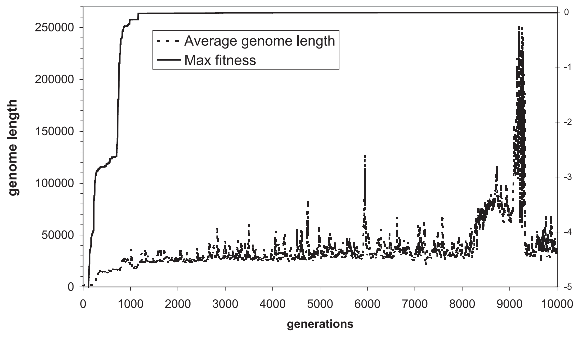


Figure 10. The result of the best run aimed at the evolution of a voltage reference circuit.

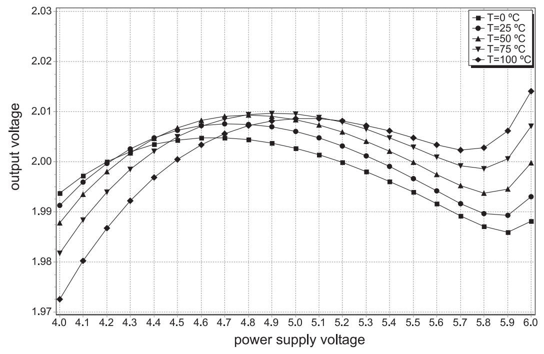


Figure 11. The performance of the evolved voltage reference circuit.

7. Discussion

Let us finally comment on the properties of the representation proposed and compare them with those of other approaches to the same problem. The generation of component connections via string alignment allows the description of almost arbitrary circuit topologies. The only limitation derives from the problem of interference between labels. Given a finite genetic alphabet and a limit for the length of acceptable terminal labels, there is obviously a limit to the number of strings whose alignment scores stay all below a given threshold. Even before that limit is attained, the evolution of new independent strings can become difficult. This problem is in part alleviated by the freedom in the choice of the genetic alphabet and matching scores, and can be fully solved by the compartmentalization technique.

Compared with existing genetic representations for analog electronic circuits [2, 3, 9], the main novelty of the one presented here is the fact of defining implicitly the connecting resistors of the circuit instead of explicitly coding them in the genome. Besides the interference problem mentioned above, this has a cost in terms of the complexity of the decoding. When the number of components grows, the number of comparisons grows quadratically and so does the number of connecting resistors that can potentially appear in the decoded circuit. This problem also can be tackled with the compartmentalization technique.

The required presence of operators of genome reorganization entails the risk of rapid growth of the genome size. The first observations of this representation at work, show that most of the resulting genome does not code for components, but is just "junk" genome, which takes memory space but weighs only marginally on the complexity of decoding. Finally, there seem to be no natural way to implement within the string alignment philosophy, the evolution of parameterized component values and circuit topologies that was successfully demonstrated with the Genetic Programming approach [3].

```
* Voltage Reference Problem
*
.dc Vcc_ext 4 6 0.1
.save dc V(1) V(3)
.TEMP 0 25 50 75 100
*
.options  ABSTOL=10.0E-09
+VNTOL=100.E-06 RELTOL=1.00E-04
+GMIN=1.00E-10 ITL1=500 ITL2=200
+ITL4=50 ITL6=0 METHOD=TRAP
+MAXORD=2
*
Vcc_ext 1 0 10
Rsource_ext 1 2 1k
Rload_ext 0 3 10k
*
Q1 4 5 6   BC846B
Q2 4 5 6   BC846B
Q3 10 11 12   BC846B
Q4 13 14 15   BC846B
Q5 16 17 18   BC846B
Q6 19 11 12   BC846B
Q7 22 23 24   BC846B
R1 0 16 2.371E+05
R2 0 17 1.778E+02
R3 0 24 7.499E+05
R4 2 3 1.909E+03
R5 2 5 3.750E+05
R6 2 6 5.623E+05
R7 2 15 2.699E+05
R8 2 18 7.366E+04
R9 3 6 5.623E+05
R10 3 14 9.955E+02
R11 3 15 5.623E+05
R12 3 18 1.724E+02
R13 4 14 6.817E+02
R14 4 15 3.065E+04
R15 4 16 1.000E+06
R16 4 22 5.620E+01
R17 5 11 9.529E+03
R18 5 23 4.217E+05
R19 6 12 1.302E+03
R20 6 15 2.812E+05
R21 6 19 4.217E+05
R22 6 23 7.499E+05
R23 6 24 4.800E+02
R24 11 16 1.000E+06
R25 11 23 2.699E+03
R26 12 14 5.623E+05
R27 12 15 1.000E+06
R28 12 17 7.499E+05
R29 12 22 1.000E+06
R30 12 23 2.812E+05
R31 12 24 1.581E+05
R32 14 15 7.499E+05
R33 14 17 4.217E+05
R34 14 18 1.778E+02
R35 14 22 7.499E+05
R36 15 16 7.499E+04
R37 15 22 1.000E+05
R38 16 22 7.499E+05
R39 16 24 5.623E+05
R40 17 24 2.371E+02
*
.MODEL BC846B NPN IS=1.822e-14
+NF=0.9932 ISE=2.894e-16 NE=1.4
+BF=324.4 IKF=0.109 VAF=82
+NR=0.9931 ISC=9.982e-12 NC=1.763
+BR=8.29 IKR=0.09 VAR=17.9 RB=10
+IRB=5e-06 RBM=5 RE=0.649
+RC=0.7014 XTB=0 EG=1.11 XTI=3
+CJE=1.244e-11 VJE=0.7579
+MJE=0.3656 TF=4.908e-10 XTF=9.51
+VTF=2.927 ITF=0.3131 PTF=0
+CJC=3.347e-12 VJC=0.5463
+MJC=0.391 XCJC=0.6193 TR=9e-08
+CJS=0 VJS=0.75 MJS=0.333
+FC=0.979
*
.end
```

Figure 12. The SPICE description of the evolved voltage reference circuit.

Acknowledgments

This work was supported by the Swiss National Science Foundation, grant no. 620-58049.

References

[1] T. Bäck, Self-adaptation in genetic algorithms. In F.J. Varela, and P. Bourgine, editors, *Proceedings of the first European Conference on Artificial Life*, 263-271, MIT Press, Cambridge, MA, 1992.

[2] J.R. Koza, F.H. Bennet III, D. Andre, and M.A. Keane, *Genetic Programming III: Darwinian Invention and Problem Solving*, Morgan Kaufmann, San Francisco, CA, 1999.

[3] J.R. Koza, M.A. Keane, M.J. Streeter, W. Mydlowec, J. Yu, and G. Lanza, *Genetic Programming IV: Routine Human-Competitive Machine Intelligence*, Kluwer, Norwell, MA, 2003.

[4] D. Graur and W.-H. Li, *Fundamentals of Molecular Evolution*, Sinauer, Sunderland, MA, 2000.

[5] D. Sankoff and J.B. Kruskal (eds.), *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison*, Addison-Wesley, Reading, MA, 1983.

[6] J.A. Shapiro, A 21st century view of evolution, *Journal of Biological Physics*, 28 (4): 745-764, 2002.

[7] A. Vladimirescu, *The SPICE Book*, Wiley, New York, 1994.

[8] X. Yao, Evolving Artificial Neural Networks, *Proceedings of the IEEE*, 87(9): 1423-1447, Sep. 1999.

[9] R.S. Zebulum, M.A.C Pacheco, and M.M.B.R Vellasco, *Evolutionary Electronics: Automatic Design of Electronic Circuits and Systems by Genetic Algorithms*, CRC Press, Boca Raton, FL, 2002.