An Advanced Clonal Selection Algorithm with ad-Hoc Network-based Hypermutation Operators for Synthesis of Topology and Sizing of Analog Electrical Circuits

Angelo Ciccazzo[1], Piero Conca[2]
Giuseppe Nicosia[2] and Giovanni Stracquadanio[2]

[1] ST Microelectronics
Stradale Primosole 50, 95121 Catania, Italy
{angelo.ciccazzo}@st.com
[2] Department of Mathematics and Computer Science
University of Catania
Viale A. Doria 6, 95125 Catania, Italy
{conca,nicosia,stracquadanio}@dmi.unict.it

Abstract. In electronics, there are two major classes of circuits, analog and digital electrical circuits. While digital circuits use discrete voltage levels, analog circuits use a continuous range of voltage. The synthesis of analog circuits is known to be a complex optimization task, due to the continuous behaviour of the output and the lack of automatic design tools; actually, the design process is almost entirely demanded to the engineers. In this research work, we introduce a new clonal selection algorithm, the *elitist Immune Programming*, (EIP) which uses a new class of hypermutation operators and a network-based coding. The EIP algorithm is designed for the synthesis of topology and sizing of analog electrical circuits; in particular, it has been used for the design of passive filters. To assess the effectiveness of the designed algorithm, the obtained results have been compared with the passive filter discovered by Koza and co-authors using the Genetic Programming (GP) algorithm. The circuits obtained by EIP algorithm are better than the one found by GP in terms of frequency response and number of components required to build it.

1 Introduction

The immune system consists of a complex network of process interactions, which cooperates and competes to contrast the antigen attacks. Theory of clonal selection principle hypothesizes that B-cells contrast the infections by means of a series of measures. Every being has a very large population of different B-cells within its body. In case an external entity, such as a virus or a bacterium, trespasses the body barriers, B-cells start trying to match the external body or antigen, by means of the receptors present on their cell surface. When the

receptors of a B-cell totally or partially match the antigen, the B-cell starts to proliferate in a process called *clonal expansion*. Moreover, the cloned B-cells can undergo to somatic mutations, in order to increase the affinity with an antigen: it is a Darwinian process of variation and selection, called *affinity maturation* [1]. This bottom-up behaviour has received a great attention in computer science, and it is the main source of inspiration for the emerging class of *Immune Algorithms* [2–6].

In electronics, the design of analog circuits is an iterative process accomplished by skilled engineers. There is no CAD tool that automatically designs analog circuits starting from a set of requirements [7]. The main idea is to find a general methodology that makes effective this working flow in order to automatically design new analog circuits and speeding up the time-to-market for new devices [8, 9]. In order to tackle this problem, the *elitist Immune Programming* algorithm (EIP) is introduced: it extends the *Immune Programming* (IP) algorithm [10] with the introduction of *elitism* and ad-hoc hypermutation operators for handling analog circuits. The EIP algorithm is adopted for the design of analog circuits belonging to the class of *passive filters*. A Passive filter is an interesting test-bed tackled firstly by the Genetic Programming (GP) algorithm [11–14]. We have conducted several experiments in order to highlight two important aspects: firstly, how the elitism impacts the exploring and exploiting ability of the immune programming algorithm; secondly, the suitability of EIP for the automatic synthesis and sizing of analog electrical circuits. The obtained experimental results confirm that EIP outperforms the standard IP approach in terms of convergence speed and quality of the designed circuits; moreover, the new immune algorithm is able to design passive filters that are clearly better than the one discovered using GP in terms of frequency response and number of components required.

In section two we give an overview on the passive filters; in section three we describe the *elitist Immune Programming* algorithm; in section four, we report our experimental results and in section five we outline conclusions and future works.

2 Passive filters circuits

Passive filters are a particular class of analog circuits, which are made of passive components, such as resistors, capacitors and inductors. Given a signal, a filter leaves it unchanged in a frequency range called *pass band*, instead, in a frequency range called *stop band*, it attenuates the signal below a certain level. In the pass band a *ripple voltage* (V_r) should be achieved; V_r is the maximum acceptable amplitude of the oscillations in pass band. In the range between pass and stop bands, called *transition band*, the filter must reduce the input signal amplitude in order to reach the desired attenuation with a very smooth behaviour. Slight deviations from an ideal behaviour are considered acceptable and they are specified by the two deviation parameters d and h.

The circuit contains a test structure and a circuit core, in this way, the same operating conditions are used for every circuit put into the core structure. The test structure is made of a signal generator (VSOURCE), a series resistance (RSOURCE), a Load Resistance (RLOAD) and a Ground link. This structure supplies three links, the first link provides the power voltage to the circuit core, which is connected to the load resistor via the second link and the third provides the connection to the ground, as shown in Fig. 1.

In our experiments, we synthesize a passive filter with a cut-off frequency of $1KHz$ and a transition band of $1KHz$. The value for d and h were settled respectively at $0.1V$ and $10^{-4}V$ and the V_r parameter was settled to $0.03V$. The set of available values for resistors and capacitors is that of the commercial series E-24. The order of magnitude of resistors values ranges from $10^{8}\Omega$ to $10^{-2}\Omega$, while the order of magnitude of capacitors ranges from $10^{-1}F$ to $10^{-11}F$. For inductors there is not an analogue standardization, so we have chosen values ranging from $1H$ to $10^{-11}H$ with a step size of 0.1 [12].

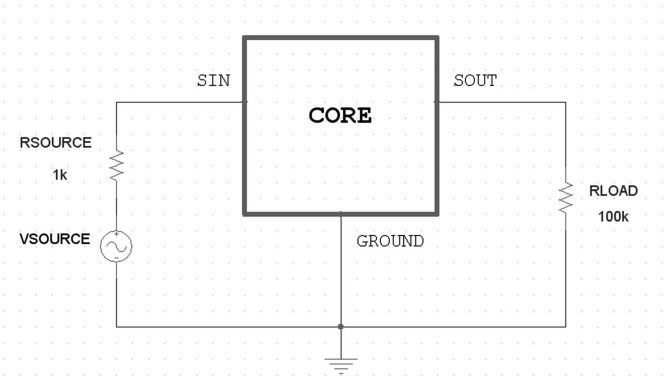


Fig. 1. Passive Filter Circuit. It is possible to note the shunt resistance, RSOURCE, the Load Resistance, RLOAD and the power supply source, VSOURCE.

3 Immune Programming for Analog Circuit Design

In this section we give an overview of the standard IP algorithm and, successively, we give a detailed description of the new EIP algorithm.

3.1 Immune Programming

The *Immune Programming* (IP) is a population-based algorithm inspired by the clonal selection principle. The algorithm starts with a population of randomly generated B-cell. At each generation g, IP builds a new population by considering each B-cell for replacing, cloning or hypermutation. The process is iteratively

performed until the maximum number of generations (or objective function evaluations) is reached. The *Replacement* operator replaces a B-cell of the population with a new random one, it is mainly employed in the early stage of the evolutionary process, and it is one of the major responsible for the diversity of the current population. The *Cloning* operator is used to create multiple copies of the best individuals in the population, this operator gives more chance to explore a promising region of the solution space. The *Hypermutation* operator is used to modify a B-cell according to its fitness value and it is the crucial point for the exploring ability of the algorithm.

In the IP algorithm, these three operators are controlled by three parameters P_r, P_c, P_m. P_r represents the minimum percentage of newly generated B-cell at each iteration, and it is inversely proportional to the average fitness function value of the previous generation. In the early stage of the evolutionary process, the IP algorithm makes a lot of replacements that decrease when a good repertoire of solutions is established. The parameter P_c controls the ratio between number of cloned B-cells and the number of B-cells that will be mutated. The last parameter P_m represents the percentage of receptors of the best B-cell that will be mutated; according to this strategy, the best circuit is less mutated than the worst ones that can undergo a complete mutation of each receptor.

3.2 The Elitist Immune Programming Algorithm

IP was the starting point to develop the new *elitist Immune Programming* (EIP) algorithm; the pseudo-code of the algorithm is provided in Fig.2. EIP differs from IP in several points, the following new features are introduced to effectively tackle the synthesis of topology and the sizing of analog circuits.

Firstly, the algorithm was modified with the introduction of *elitism*. At each generation g, the best solution found so far cannot be erased from the population. This strategy, already introduced in other *immune inspired* algorithms [5, 6, 2], greatly helps the convergence of the algorithm and it overcomes the problem of IP that tends to quickly forget good solutions especially in the initial phase of the search process. The other main difference is the application of the cloning and hypermutation operators. As in IP the chance to be cloned or mutated is driven by a parameter P_c but, in EIP, for each cloning two mutations are performed.

Mutation operators The hypermutation operators operate only on the core structure; in particular, the hypermutation acts on one component, link or node at a time. All the operators take in input and return in output only consistent circuits. This design choice forces the algorithm to search in the feasible region of the solution space, and it helps the algorithm to immediately discard infeasible or meaningless solutions. Ten different mutation operators have been introduced, and each of them makes a specific mutation on the circuit as described below.

`ADD-SERIES`. Given a circuit, it randomly selects a component and it randomly unplugs one of its terminals; successively, a new component is created and the operator connects it in series to the selected component, linking the floating terminal to the new one.

```
1: procedure EIP(D, MaxGen, P_r, P_m)
2:     G ← 1
3:     Population^(0) ← Initialize(D)
4:     Evaluate(Population)
5:     while G < MaxGen do
6:         Population^(G+1) ← empty
7:         Population^(G+1) ← BestCircuit[Population^(G)]
8:         Population^(G+1) ← Hypermutation[BestCircuit[Population^(G)]]
9:         i ← 0
10:        repeat
11:            if rand() < P_r then
12:                NewCircuit ← Initialize()
13:                Population^(G+1) ← NewCircuit()
14:            else
15:                if rand() < P_c(Circuit_i) then
16:                    Population^(G+1) ← Population_i^(G)
17:                end if
18:                for j ← 1 to 2 do
19:                    if rand() < P_m(Circuit_i) then
20:                        Population^(G+1) ← Hypermutation[Population_i^(G)]
21:                    end if
22:                end for
23:                i ← i + 1 mod D
24:            end if
25:        until size[Population^(G+1)] < D
26:    end while
27: end procedure
```

Fig. 2. The pseudo-code of the EIP algorithm.

`ADD-PARALLEL`. It establishes a shunt connection. After a component is selected, the operator randomly creates a new component and then it links its terminals to the same nodes of the selected one.

`ADD-RANDOM-COMPONENT`. It randomly creates a new component that will be connected to two random nodes of the circuit.

`EXPAND-NODE`. This operator randomly selects a circuit node and it randomly generates a new node and a new component. Successively, it connects the new component to the previous selected node.The scope of this procedure is to easily plug in a new component into a highly linked node, or a test structure node.

`DELETE-COMPONENT`. This procedure tries to decrease the size of the circuit by deleting a component. It does not affect the consistency of the circuit; however, if a deletion causes damages, the operator is able to repair the circuit.An inconsistent circuit can arise due to one or more floating terminals, the unplugging of the circuit core from the test structure or the unlinking of a part of the circuit.

`MUTATE-COMPONENT-VALUE.` The operator randomly selects a component and it changes its value by randomly picking a new value from the set of allowed values .

`COPY-COMPONENT-VALUE.` The operator randomly selects a component of the circuit and it copies the value of a randomly chosen component of the same type. If there is no other similar component, it does nothing.

`MUTATE-COMPONENT-KIND.` This operator randomly selects a component, then it modifies the relative type and it assigns a value to the component according to the allowed set of values for the new type.

`LINK-MODIFY.` The operator randomly disconnects a link of a component and reconnects it to a different circuit node. Like `DELETE-COMPONENT`, this procedure is able to recover from inconsistent circuits.

`SHRINK.` The `SHRINK` operator scans the circuit in order to find a series or parallel connection between two or more components. It reduces the circuit size by replacing a couple of components with one equivalent component which value is as close as possible to the values of the two components.This operator greatly improves the quality of the design since it allows the automatic introduction of standard components and the reduction of the circuit size with only marginal side effects [15].

Fitness function The quality of the circuit is assessed by means of an ad-hoc objective function; it measures the distance between the curve described by a circuit and the one described by a hypothetical ideal circuit according to the following expression:

$$f_{pf}(x) = \sum_{i=100mHz}^{1KHz} [W_p((f_i), f_i) \times d(f_i)] + \sum_{i=2KHz}^{100MHz} [W_s((f_i), f_i) \times d(f_i)] \qquad (1)$$

where x is a consistent circuit, f_i is the $i-th$ frequency, $d(f_i)$ is the signal deviation from an ideal behaviour and $W_p(d(f_i), f_i)$ and $W_s(d(f_i), f_i)$ are weighting factors respectively for the pass and stop band. For each frequency, the corresponding weighting factor for the pass band is determined as follows:

$$W_p = \begin{cases} 0 & d(f_i) \leq V_r \\ c & V_r < d(f_i) \leq d \\ 10 \cdot c & d(f_i) > d \end{cases}$$

where V_r is the ripple voltage and d, c are experimentally obtained constants that were fixed to $d = 0.1V$ and $c = 3$. The weighting factor for the stop band term is obtained as follows:

$$W_s = \begin{cases} 0 & d(f_i) \leq SBA \\ m & SBA < d(f_i) \leq h \\ 10 \cdot m & d(f_i) > h \end{cases}$$

where SBA is the desired *Stop Band Attenuation*, that was fixed to $-60dB$ and d, h, m are experimentally obtained constants fixed to $d = 0.1V$, $h =$

			CLFF		CLNC	
ALGORITHM	d	P_m	f_{pf}	Components	f_{pf}	Components
IP	5×10^3	0.1	1632.04	5	1632.04	5
IP	5×10^3	0.3	1343.03	5	1343.03	5
IP	10^4	0.1	1758.54	3	1758.54	3
IP	10^4	0.3	1742.77	6	1763.77	4
EIP	5×10^3	0.1	20.5486	20	20.948	18
EIP	5×10^3	0.3	10.2221	20	11.3294	16
EIP	10^4	0.1	**0.0**	12	0.29	**10**
EIP	10^4	0.3	8.7778	18	8.78324	16

Table 1. Experimental results, the performances of the two immune algorithms. For each parameters setting, we report the *Circuit with the Lowest Fitness Function value* (CLFF) and *the Circuit with the Lowest Number of Components* (CLNC).

$10E - 5V$ and $m = 50$. It is possible to observe that the co-domain of the distance function is $[0, +\infty[$, where an ideal circuit has $f_{pf}(x) = 0$. This distance function neglects small deviations from an ideal behaviour and it strongly penalizes unacceptable deviations. The fitness of each B-cell is the value of f_{pf} normalized in the range $[0, 1]$ according to the following expression:

$$fitness(x_i^g) = \frac{1 - s_{f_{pf}}(x_i^g) \times m_{f_{pf}}(x_i^g)k}{\alpha} \tag{2}$$

$$s_{f_{pf}}(x_i^g) = \frac{f_{pf}(x_i^g)}{f_{pf}^{MAX}(g)} \tag{3}$$

$$m_{f_{pf}}(x_i^g) = e^{\frac{f_{pf}(x_i^g)}{k}} \tag{4}$$

where x_i^g is the $i-th$ B-cell of the population at generation g, $f_{pf}^{MAX}(G)$ is the max value of the objective function at generation g, instead k is a constant used to constraint the fitness in the range $[0, 1]$. Moreover, the fitness was scaled of $\alpha = 25\%$ in order to prevent that the worst B-cell undergoes to a complete mutation of the circuit.

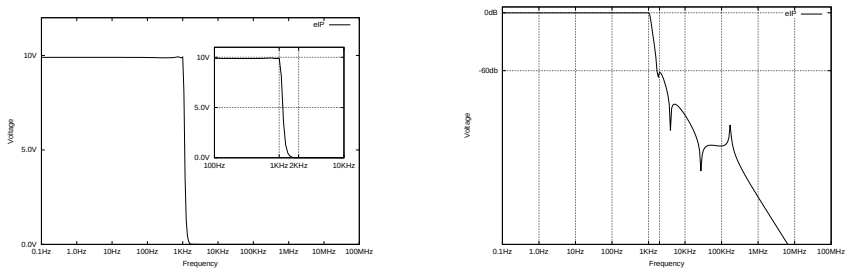
4 Experimental Results

In order to assess the effectiveness of the EIP algorithm, we performed several experiments. Firstly, we compared EIP with the standard IP algorithm. We have tested these two algorithms with a population of size $d \in \{5000, 10000\}$ [16]. The mutation probability parameter was settled to $P_m \in \{0.1, 0.3\}$; since P_m is the percentage of receptor mutated in the best circuit, a larger value of this parameter makes the algorithm acting as a *random search*. Finally, the replacement probability P_r and the cloning probability P_c are fixed to $P_r = 0.01, P_c = 0.2$ [10]. In order to simulate the behaviour of the circuits, the tested algorithms use the NGSPICE circuit simulator. The maximum number of objective function

Run	Algorithm	f_{pf}	Components	Algorithm	f_{pf}	Components
1	IP	1542.72	3	eIP	3.93	20
2	IP	1765.63	6	eIP	16.79	20
3	IP	1658.13	6	eIP	12.62	20
4	IP	1492.22	4	eIP	0.29	10
5	IP	1497.31	3	eIP	0.0	12
	Average	1591.202	4.4	Average	6.726	16.4

Table 2. Experimental results, a comparison of 5 independent runs of IP and eIP using the best parameter setting according to Tab.1.

evaluations was set to 10^7 for all the experiments and the same set of mutation operators were used in both algorithms.

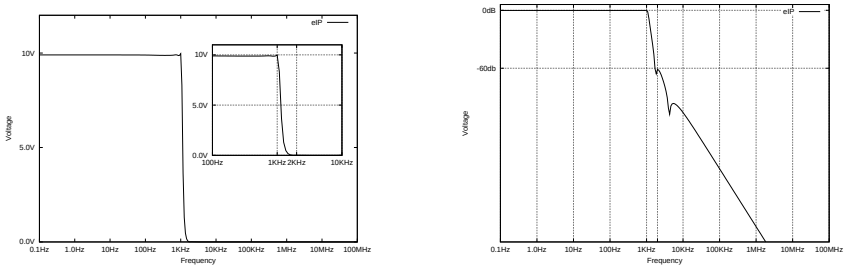


(a) (b)

Fig. 3. The *output voltage frequency response* (a) and the *attenuation plot* (b) of the best circuit found by eIP ($f_{pf} = 0.0$, 12 components)

It is possible to note in Tab.1,2 that eIP clearly outperforms the IP algorithm. For all settings, eIP shows a good convergence to near optimal solutions, instead IP produces only meaningless circuits. The scheme adopted by IP for replacement, cloning and hypermutation is not effective for this problem; at each iteration only replacement are performed and it means that IP works most likely a random search.

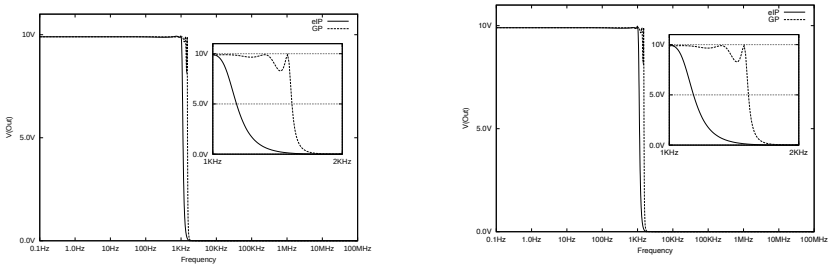
By inspecting the eIP results, it is possible to note that using a population of 10000 B-cells and a mutation probability $P_m = 0.1$, the algorithm found a circuit that perfectly matches the design requirements (Fig.3). By analyzing the circuit structure it is possible to note that is made of only 12 components that is an important aspect for the manufacturability of the filter. Moreover, by inspecting all the circuits designed by eIP, the algorithm has found a circuit of 10 components with $f_{pf} = 0.29$ (Fig.4); despite the value of the fitness function is not optimal, the circuit shows a very regular behaviour and, probably, it can be considered a good trade-off between the requirements and the manufacturability



(a) (b)

Fig. 4. The *output voltage frequency response* (a) and the *attenuation plot* (b) of the circuit with the lowest number of components found by EIP ($f_{pf} = 0.29$, 10 components)

of the filter. By observing the circuits it is possible to note that they show different shapes but common building blocks: this behaviour suggests that EIP is able to find a common regular structure and, at the same time, it is able to arrange them in order to deeply explore the space of solutions. Finally, the *population-based* approach gives to the engineers not only a single solution but a set of circuits that could be inspected in order to find the one that optimally fits the design requirements.



(a) (b)

Fig. 5. A comparison of the output voltage frequency response of the circuit with the optimal fitness function value (a, $f_{pf} = 0.0$, 12 components) and the one with the lowest number of components (b, $f_{pf} = 0.29$, 10 components) found by EIP with the *Campbell filter* [11]. it is possible to note that in the transition band the *Campbell filter* has not a regular behaviour instead the EIP circuits have a regular and smooth curve.

The GP algorithm was able to find a passive filter, known as *the Campbell filter* [11]. This filter shows a very regular structure and a good symmetry, since

it is built using the same building block repeated multiple times in order to form a seven rung ladder structure. The frequency response of the Campbell filter is substantially linear in pass band and the curve inclination is very high. The two best circuits found by EIP are better then the Campbell filter for three important aspects. Firstly, in the transition band, the signal of Campbell filter shows large swings that are an undesirable behaviour instead, the EIP circuits show a very regular and smooth curve as showed in Fig.5. Secondly, the EIP circuits have only 10 and 12 components instead the Koza's circuit has 14 components, and this fact makes the EIP circuits more suitable for a real implementation. Finally, the EIP algorithm requires 10^7 fitness function evaluations to design these circuits instead the GP algorithm requires 1.5×10^7 evaluations; this experimental result proves that the immune algorithm, for this design problem, is more efficient than GP.

5 Conclusions and Future Works

In this research work, we have introduced a new *immune algorithm*, called ELITIST IP, for the synthesis of topology and sizing of analog electrical circuits. The algorithm extends the IMMUNE PROGRAMMING approach with the introduction of *elitism* and *ad-hoc operators* for handling analog circuits.

The experimental results confirms that EIP clearly outperforms the standard IMMUNE PROGRAMMING approach in terms of quality of the circuits and speed of convergence. The analysis of the EIP circuits shows that the algorithm is able to synthesize analog circuits with excellent frequency responses, having small swings, high inclination and a good shape regularity.

The comparison with the *Campbell filter*, a passive filter discovered using GENETIC PROGRAMMING, shows that EIP is able to find a better circuit in terms of regularity in transition band and number of components required.

Starting from these results, there are two major fields that we are investigating. Firstly, we are extending the EIP algorithm in order to use a selection strategy based on the *Pareto Optimality* criterion; using this approach, it is possible to explicitly introduce different design requirements, such as the number of components and the frequency response, and leaving to the algorithm the automatic discovering of optimal trade-off [17]. Finally, we are designing an improved EIP that is able to synthesize the topology and the sizing of active filters [18]; this last task is a visionary research topic since there is not an automatic approach for the design of these analog circuits and it could be an important step to dramatically decrease the time-to-market required for these circuits.

References

1. Abbas, A., Lichtman, A., Pober, J., et al.: Cellular and molecular immunology. WB Saunders (2000)

2. Cutello, V., Nicosia, G., Pavone, M.: Real coded clonal selection algorithm for unconstrained global optimization using a hybrid inversely proportional hypermutation operator. Proceedings of the 2006 ACM symposium on Applied computing (2006) 950–954
3. Cutello, V., Nicosia, G., Pavone, M., Timmis, J.: An Immune Algorithm for Protein Structure Prediction on Lattice Models. Evolutionary Computation, IEEE Transactions on **11**(1) (2007) 101–117
4. Freitas, A., Timmis, J.: Revisiting the Foundations of Artificial Immune Systems: A Problem-Oriented Perspective. Artificial Immune Systems: Second International Conference, Icaris 2003, Edinburgh, Uk, September 1-3, 2003: Proceedings (2003)
5. Cutello, V., Nicosia, G., M., P.: A hybrid immune algorithm with information gain for the graph coloring problem. In: GECCO. Volume 2723. (2003) 171–182
6. Cutello, V., Morelli, G., Nicosia, G., M., P.: Immune algorithms with aging operators for the string folding problem and the protein folding problem. In: EVOCOP. Volume 3448. (2005) 80–90
7. Streeter, M., Keane, M., Koza, J.: Iterative Refinement Of Computational Circuits Using Genetic Programming. Proceedings of the Genetic and Evolutionary Computation Conference table of contents (2002) 877–884
8. Koza, J., Bennett III, F., Andre, D., Keane, M., Dunlap, F.: Automated synthesis of analog electrical circuits by means of genetic programming. Evolutionary Computation, IEEE Transactions on **1**(2) (1997) 109–128
9. Kashtan, N., Alon, U.: Spontaneous evolution of modularity and network motifs. Proceedings of the National Academy of Sciences **102**(39) (2005) 13773–13778
10. Musilek, P., Lau, A., Reformat, M., Wyard-Scott, L.: Immune programming. Information Sciences **176**(8) (2006) 972–1002
11. Koza, J., Bennett III, F., Andre, D., Keane, M.: Synthesis of topology and sizing of analog electrical circuits by means of genetic programming. Computer Methods in Applied Mechanics and Engineering **186**(2-4) (2000) 459–482
12. Koza, J., Jones, L., Keane, M., Streeter, M.: Towards industrial strength automated design of analog electrical circuits by means of genetic programming. Genetic Programming Theory and Practice II (2004)
13. Grimbleby, J.: Automatic analogue circuit synthesis using genetic algorithms. Circuits, Devices and Systems, IEE Proceedings [see also IEE Proceedings G-Circuits, Devices and Systems] **147**(6) (2000) 319–323
14. Alpaydin, G., Balkir, S., Dundar, G.: An evolutionary approach to automatic synthesis of high-performance analog integrated circuits. Evolutionary Computation, IEEE Transactions on **7**(3) (2003) 240–252
15. Dastidar, T., Chakrabarti, P., Ray, P.: A Synthesis System for Analog Circuits Based on Evolutionary Search and Topological Reuse. Evolutionary Computation, IEEE Transactions on **9**(2) (2005) 211–224
16. Koza, J.: Genetic Programming III: Darwinian Invention and Problem Solving. Morgan Kaufmann (1999)
17. Subramanian, A., Sayed, A.: Multiobjective filter design for uncertain stochastic time-delay systems. Automatic Control, IEEE Transactions on **49**(1) (2004) 149–154
18. El-Habrouk, M., Darwish, M., Mehta, P.: Active power filters: a review. Electric Power Applications, IEE Proceedings **147**(5) (2000) 403–413