A Circuit Representation Hierarchy Technique and Probabilistic Models for Evolutionary Analog Circuit Design

H. Mühlenbein[1], T. Mahnig[1], V. M. Kureichik[2], L. A. Zinchenko[3]

[1]Theoretical Foundation GMD Lab, GMD FZ Informationstechnik
53754, Sankt – Augustin, Germany
{muehlenbein, mahnig}@gmd.de
[2]CAD Department, Taganrog State University of Radioengineering,
l. Nekrasovsky, 44, Taganrog, Russia, 347928,
kur@tsure.ru
[3]EE Department, Taganrog State University of Radioengineering
l. Nekrasovsky, 44, 347928, Taganrog, Russia
toe@tsure.ru

Abstract. The approach to computer aided hierarchical circuit design on the base of univariate algorithms was derived by the analysing the mathematical principles behind recombination. In this paper it is considered how to introduce mutation in this approach. A number of experiments applied to circuit design are discussed. Experiments indicate that mutation and elitism increase the performance of the algorithms and decrease the dependence on the correct choice of the population size.

1 Introduction

The modern electronics market is characterized by a high level of complexity. Many different functions are integrated onto a single device. Together with the increase in circuit complexity the design complexity of these system–on-chip designs has increased as well. Advanced computer aided design tools are required for effective design of these mixed circuits. In the digital part various effective approaches have been considered [1]. Unfortunately, we do not have robust circuit synthesis in the analog domain. The analog blocks are still designed by hand, therefore, long time is required for mixed - analog - digital design although analog portion of mixed system is a small fraction of the overall design size.

Considerable progress has been made in computer aided analog synthesis using evolutionary modelling [1]. Evolutionary design can autonomously reconfigure a circuit for adapting to the design requirements by means of a learning capability that employs evolutionary computation. Some positive results have been published [2-12]. However, long learning time, difficulty to predict when an effective capability will appear and other such problems have hindered progress in diffusion into electrical engineering fields. Some strategies to support efficient analog synthesis have been developed by means of combined genetic/annealing optimisation algorithm [4], differential evolution [5], hybrid genetic algorithm [12] etc.

In this paper, we focus on probabilistic evolutionary models. The Estimation of Distribution algorithm EDA have been proposed by Mühlenbein and Paaß [13] as an extension of genetic algorithms. Instead of performing recombination of strings, EDA generates new points according to the probability distribution defined by the selected points. In [14] Mühlenbein showed that genetic algorithms can be approximated by an algorithm using univariate marginal distribution only (UMDA). In [15] Mühlenbein and Mahnig have extended this algorithm by means of mutation. The correct choice of the size of the population turned out to be difficult in evolutionary circuit design. Some numerical experiments are considered. Furthermore we use our results to determine effective ratio between the population size and the mutation rate.

The remainder of the paper is organized as follows. Section 2 briefly surveys prior analog synthesis work. Section 3 develops our hierarchical construction of design space. Section 4 presents experimental results. It turns out that mutation not only makes search algorithm more efficient in many cases, but also more robust in the sense that choosing the correct population size is of less importance.

2 Related Works

The computer-aided design of analog circuits is not trivial. Each possible solution lies within large search space. Its dimension is defined by amount of different component building blocks presented in the framework, amount of rules used to generate the circuit and the application for which the circuit is being evolved. Evolutionary algorithms are employed in computer aided circuit design as they provide a non-heuristic investigation in such large search space.

Evolutionary approach considers a design as evolutionary process of creation, development and selection. There are two main problems in evolutionary circuit design. Methods of search space construction can be defined as the first problem and methods of its investigation are the second problem. They are mutually connected. The choice of space construction rules defines possibilities for new solution generation. Evolutionary algorithms have no restrictions on the representation form. Therefore, encoding must be chosen only for valid solution generation or repair operators are used for solution correction.

Evolutionary algorithms are widely used for computer-aided design as design automation tools at various level of abstraction. Here, we restrict our discussion to analog circuit design. A number of successful examples have been published by the use of genetic algorithms in filters design [3, 6], transducers design [19] etc. A design variety was restricted by the fixed pre-defined topologies.

Genetic programming [7, 8] introduces another direction for synthesis, which can be called as developmental synthesis. Unlike previous method the circuit topology, number and types of components, its values are evolved from embryonic circuit (really a set of short circuits) to full circuit. Genetic programming produces circuits that are highly redundant as result of knowledge absence about solved problem. Owing to the nature of such design technique most of circuits generated at the initial steps are faulty circuits. Introduction of transformation rules (cc-bot instructions) allows generate circuit with valid graphs [9]. The application of evolutionary developmental synthesis was limited by blind search problems. Analysis of circuits both useful and faulty results in the increase of computational efforts. Another problem is the development of introns [10]. In electrical circuits they appear as faulty circuits after crossover and mutation operators. It was shown that a selective pressure on the circuit size is the simple effective method. This selective pressure requires the additional computational efforts for elimination of faulty circuits, however. A using of starting points introduced to the initial population [11] supplies less time, which needs to be spent on the search. This approach using a combination of evolutionary adaptation process and construction rules is effective. The similar circuits generation reduces evolutionary adaptation time by means of preliminary knowledge. Its disadvantage is a possibility of stable attractors around a starting point. Only local optimum can be found when landscape has saddle points in initial area.

Perhaps the most important feature in evolution design is a choice of search algorithm. Its characteristics are defined by its speed, flexibility and ability to investigate a solution space effectively. During design most part of steps are iterated. Therefore algorithms should offer the choice of good solutions, which were generated very fast.

One from developing branch of evolutionary algorithms are probabilistic models of promising solutions [13 - 17]. They can be divided in two categories: univariante distribution algorithms, like population-based incremental learning algorithm and univariate marginal distribution algorithm and bivariate distribution algorithms. Several typical examples of probabilistic models are compact genetic algorithm, ant colony algorithm, probabilistic incremental program evolution, factorised distribution algorithm etc. Probabilistic approach defines the direct connection between genotype and fitness. Recently ant colony algorithm has been used to computer - aided design of digital circuits [18].

UMDA [15 - 17] is an evolutionary algorithm that combines mutation and recombination by using of distribution. The distribution is estimated from a set of selected points. It is then used to generate new points for the next generation. Different selection methods have been investigated in [17]. We used only truncation selection, where τ is an amount of selected individuals. In [14] mutation has been introduced into UMDA by a concept called Bayesian prior. In practice, this means that the estimated probability is calculated as $p_i=(m+r)/(N+2r)$, where m is equal to the number of bits with value 1 in the selected parents in i loci, N is the population size. Hyperparameter r is chosen for OneMax function according to [14]. Some experimental results for several test functions have been considered as well [14].

In [20, 21] we introduced the hierarchical technique to evolutionary design of analog cells. By combining hierarchical design at element, topology and parameter levels and UMDA we were able to synthesize a range of analog circuits. Our goal in this paper is to demonstrate that mutation is also within the reach of this technique. Furthermore we consider a role of a several mutation rate to hierarchical circuit design.

3 Evolutionary Design with Hierarchy

The evolutionary design employed in our works is based on the hierarchical construction of design space. An overview of the design process is described in [21]. Our technique differs from the previously mentioned approaches in that we used transition from circuit construction to fitness evaluation without following steps:

- genotype was interpreted in a SPICE netlist representation;
- this netlist is processed by SPICE;
- output from SPICE is then used to compute fitness for the individual.

Our approach reduces computational efforts due to direct decoding of genotype to system of equations. We used modified nodal method likely to SPICE approach.

For reduction of blind search the design process have to be described with the help of separate sets analogously DNA, in which each part defines separate properties of individuals and change in genotype can be performed only between correspondent blocks. Thereby we can describe properties of design object at the most bottom level and

then to change these properties at various levels of hierarchy. Therefore the role of the coding of genotype is important, because at this stage the instructions on selection and placement of initial units are defined.

The construction process is based on the theory of complete graphs. Each solution is represented as a hierarchical set, in which numbers of nodes, types of elements and parameters are defined at different levels. Its first level is amount of complete graph nodes. Second level is defined by sequence of elements from set ELEMENT={S, R, L, C} etc. We used only two-terminal components only. Alleles for components' type are S, R, L, C, which represent switches, resistors, inductors, capacitors etc. Third level is parameter one.

In our hierarchical coding the circuit is termed a phenotype. The genotype is defined as a linear structure with variable length: $NE_n \ldots E_2E_1E_0\, P_k \ldots P_2P_1P_0$, where N is the amount of complete circuit graph nodes; $E_n \ldots E_2E_1E_0$ is part of the genotype, defining the evolution of elements; $P_k \ldots P_2P_1P_0$ is part of the genotype, describing parameter evolution (changes parameter elements without changes of topology); n is amount of circuit elements; k is amount of circuit element parameters. Note that when we use the complete graph and m bits to parameter description, we get

$$n = \sum_{i=0}^{N-1}(N-i), k = 2^{mn}. \quad (1)$$

For elimination of blind search the design is described by separate sets E, P analogously to DNA. The set E describes a set of components on the most bottom level. Their connection is defined by the sequence of compete graph branches. Set P determines component parameters. Analog circuits are hard to manufacture. Only fixed parameters are possible for circuit manufacturing. Therefore, parameters can be changed only as discrete values. Finally, the chromosome hierarchy structure is defined as discrete set (Fig. 1).

Element	Connection nodes	Parameters
R	(i1, j1), (i2, j2)	{100, …,100M}
L	(i1, j1),(i2, j2)	{1 nH, …, 1H}
C	(i1, j1), (i2,j2)	{10 nF, …, 1 mF}

Fig. 1. A chromosome hierarchy structure with the dynamic parameter and topology coding used for the circuit description

In general, evolutionary circuit design can be divided into two main fields [2]: evolutionary design optimisation and creative evolutionary design. Many researches seem to fall into single category, but some work does attempt to combine evolutionary optimisation and creative evolutionary design into integral evolutionary design systems. Usually several applications of circuit having fixed topology are determined by its parameters. A variation of single parameter can change a bandwidth, gain etc. Therefore, simulation within single parameter supplies only local search. We use some fixed amount of parameter variety. Each element is determined by type and parameter. The first and the second bits of each group define the type. Each element can take on 3 different types, namely, switch (0), capacitance (1) and inductance (2). The following bits define sizing of component. Parameter of switch can take on two fixed values only. We consider that 0 means to a switch turned off and 1 to a switch turned on. Parameters of inductance and capacitance can be chosen from fixed range. The input of the circuit is always node 1 and the output is node 2. The branch 1 is always series connection of a voltage source and the source resistance. Branch 2 is the load resistance. Circuit being evolved contains (n-2) branches. Figure 2 shows a chromosome of length 23 and the circuit created from it.

During evolution adaptation we use a growing technique, where the user can specify the boundary limits on the amount of nodes. Then set E is formed by two subsets {E1, E2}, where subset E1 includes elements describing structure of the initial solution; subset E2 includes elements describing evolutionary process. All elements of subset E2 are set to 0 in the initial population and then the initial set E is described as {E1, 0}. The initial population can be defined randomly, from user's experience or from a textbook. During evolutionary design any element of set E randomly can change. In this case set P automatically change. There are 2 possible cases. If the element, which is in E1, changes to 0, then corresponding elements in set P are set to zero. In the second case including change of an element from 0 to any from set ELEMENT, corresponding elements are defined in set P as element parameters. Restrictions are fixed amount of switches defined by user.

At the following stage of evolution the elements of set P can be changed by a random amount from discrete set. Distinctions on this level are definition of parameters within fixed parameters. Possibilities of different phenotype forms describe P set. It consists of two subsets {P1, P2}. Allowed constants, defining one possible phenotype form are elements of first subset. P2 is subset of constants, defining a few forms of phenotype. Therefore subset P2 contains parameters, which can be changed. The correspondence between constants of this subset and variables in phenotype to be satisfied defines the variable domains. All constraints are defined by heuristic rules.

Amount of nodes	Branch 1	Branch 2	Branch 3	Branch 4
0 1 1	1 0 (Element) 0 1 1 (parameter)	0 1 1 0 1	0 1 1 0 1	0 0 0 0 1

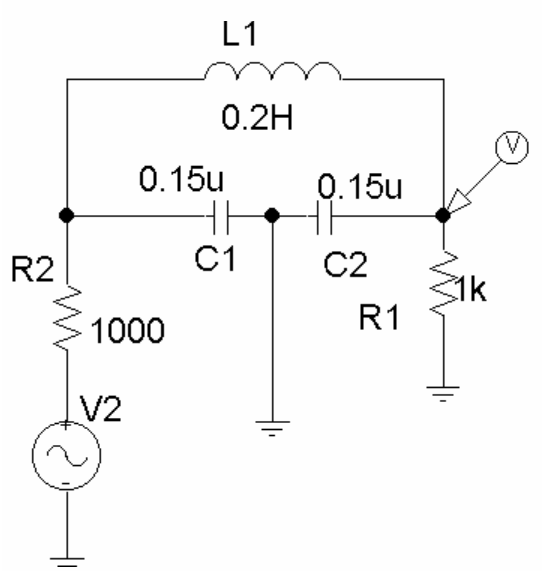


Fig. 2. Chromosome and the circuit, created from it

Then the best solution of topological design level is used as initial circuit to the following parametrical optimisation. At this step all bits correspond to element parameter, what supplies more flexible search within fixed topology. This type of connection between genotype and phenotype is the dynamic coding, allowing to use the adaptation of genotype and phenotype simultaneously. Various forms of connection between genotype and phenotype are possible by the use of P1 and P2. Subset P1 defines only one possible form, while subset P2 establishes the dynamic relation between genotype and phenotype. One form of genotype defines a few forms of phenotypes. Therefore evolution on phenotype level is possible for dynamic coding. The use of such hierarchy to chromosome coding eliminates blind search by applications of heuristic rules and the reduction of search space dimension.

Fitness is measured in terms of the sum of the absolute weighted difference between the actual output voltage and target one. These error values were summed across evaluation points *Nc* for definition of fitness value

$$F = \sum_{i=0}^{Nc} W[d(f_i), f_i)d(f_i)], \qquad (2)$$

where *f* is the frequency of fitness calculation, *d(f)* is the absolute value of difference between output voltage and the target and *W (d(f), f)* is the weighting for difference *d(f)* at frequency *f*.

Fitness landscape has many local optimums, which can be explained by using of a set of particular topologies therefore there is a set of different particular fitness landscape for each solution.

EXPERIMENTAL RESULTS

This section applies our hierarchical technique developed in the previous section and distribution algorithms to lowpass filter design. The starting point for the design of a filter is the specification by the user of the frequencies for the filter's passband and stopband. The design of filters with high stopband attenuation and low passband attenuation is difficult. It is more difficult to design a lowpass filter with small transitional region between two bands. The experiment described here is based on "synthesis of lowpass filter" in [8]. We considered a filter design with a passband below 1000 Hz and a stopband above 2000 Hz. It is acceptable if the output voltage in the passband is between 970 mV and 1 V. It is acceptable if the output voltage in the stopband is between 0 V and 1 mV. The ratio in the passband (1 Volt) to the maximum acceptable signal in the stopband (1 millivolt) is 60 decibels, but there is a 2-to-1 ratio between the stopband frequency and the passband frequency. The known approaches for filter design find the required circuit, but there are different fluctuations in the stopband and the passband.

The first negative result was the increase of the ratio between the genotype length and the population size supplying the finding of optimal solution. In [20, 21] we provided the effective design with the ratio within range from one to two. We emphasize that effective integral design can be supported with the ratio above two only. It can be explained that fitness landscape has many local optimums. The search can to stop in local hill between two large valleys with near heights. Unfortunately it is impossible to use a theoretical framework to determine an optimal mutation rate. In order to make a fair comparison, we have to compare actual simulation results. In addition we investigate UMDA with elitism by simulation. We use a set of circuits, which can be generated within topology of complete graph with 3 nodes as the simplest test function. The best circuit is given in Fig. 2. Amount of hits is equal to the 86. Our results have been obtained under assumption that amount if iteration is less that genotype length.

Table 1. Performance comparison with different mutation rate for population size Np=300 and truncation selection with τ=0.02 – 0.04. Succ is the number of times, when the best solution was found in 100 runs, Gen is the number of generations and Count is the number of evaluation after 10% of the population consists of the best solution

Mutation rate r	τ=0.02			Mutation rate r	τ=0.04		
	Iteration	Count	Success		Iteration	Count	Success
0	4.8	1440	29.7	0	6.07	1820	29.7
0.068	5.68	1703	55.4	0.136	6.49	1946	36.6
0.136	5.85	1755	59.4	0.272	6.61	1983	53.5
0.204	5.83	1748	57.4	0.409	7.02	2105	50.5
0.272	5.81	1744	58.4	0.545	6.6	1981	47.5
0.341	6.2	1860	64.3	0.682	7	2100	45.5
0.409	5.87	1763	64.4	0.818	7.6	2279	46.5
0.477	6.63	1989	61.4	0.954	7.6	2279	43.6
0.545	6.7	2010	66.3	1.09	7.86	2360	37.6
0.614	7.06	2119	61.4	1.23	8.32	2495	43.6
0.682	7.36	2210	70.3	1.36	8.43	2528	46.5
0.75	7.49	2248	70.3	1.5	8.92	2677	52.4
0.818	7.17	2151	69.3	1.63	9.1	2729	41.6
0.886	8.54	2562	71.3	1.77	10.02	3006	46.5
0.954	9.06	2719	63.4	1.91	11.92	3575	47.5

In tables 1, 2, we examine the behaviour for a fixed number of bits 23 and a truncation threshold τ between 0.02 to 0.04. Population size changes from Np=300 to Np=500 respectively. We considered the successful runs after 10% of the population consisted of the best solution only. It can be seen that for all mutation rate and the truncation threshold performance with prior was better than performance without a prior. When the mutation rate is high, the number of function evaluations increases, as the probabilities are shifted towards ½ too much. When the mutation rate decreases, the success rate decreases also, because it becomes too improbable to flip the remaining bits that are wrong. It is remarkable that for all runs increasing selection pressure (using smaller values of τ) is more effective. Reducing the population size from 500 to 300 reduced the number of successful runs from (42.6 – 86.1) to (29.7 – 71.3). The number of function evaluations decreased in this process, but because the success rate was too low. For the runs without mutation (r=0), the population size had to be carefully chosen in order to have a good result. When using the high mutation rate, this choice is much more easy.

Table 2. Performance comparison with different mutation rate for population size Np=500 and truncation selection with τ=0.02 – 0.04. Succ is the number of times, when the best solution was found in 100 runs, Gen is the number of generations and Count is the number of evaluation after 10% of the population consists of the best solution

Mutation rate r	τ=0.02			Mutation rate r	τ=0.04		
	Iteration	Count	Success		Iteration	Count	Success
0	5	2483	59.4	0	6.09	3046	42.6
0.114	5.63	2818	73.3	0.2272	6.54	3270	49.5
0.227	5.5	2746	70.3	0.455	6.94	3471	52.4
0.341	5.57	2789	70.3	0.682	6.42	3213	46
0.455	6.06	3029	69.3	0.909	7.04	3520	47.5
0.568	5.92	2959	72.3	1.136	7.42	3713	46.5
0.682	6.15	3076	84.1	1.364	7.06	3531	47.5
0.795	6.09	3046	75.2	1.59	7.5	3750	47.5
0.909	7	3500	74	1.818	7.9	3949	48.5
1.023	6.76	3379	86.1	2.045	8.22	4111	44.5
1.136	7.05	3525	79.2	2.27	8.39	4193	43.6

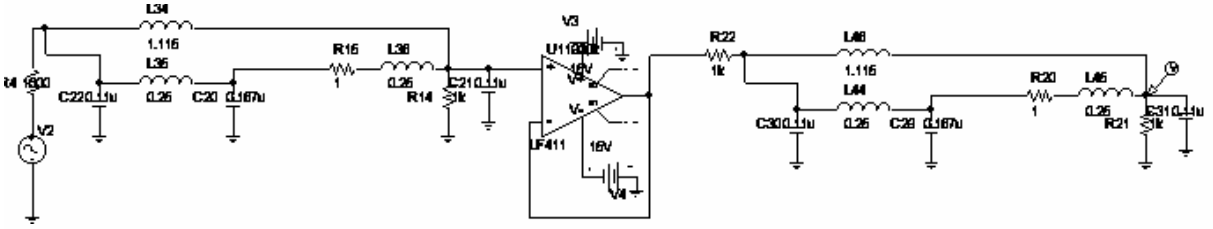
Still higher rate of success rate can be reach if elitism is used. In this case the best string is kept in the population. In table 3, the case with elitism is compared to a run without elitism. It can be seen, that for the most runs except the 3 runs performance with mutation was better than performance without mutation.

Table 3. Performance comparison with different mutation rate for population size Np=500 and truncation selection with and without elitism, τ=0.02. Succ is the number of times, when the best solution was found in 100 runs, Gen is the number of generations and Count is the number of evaluation after 10% of the population consists of the best solution

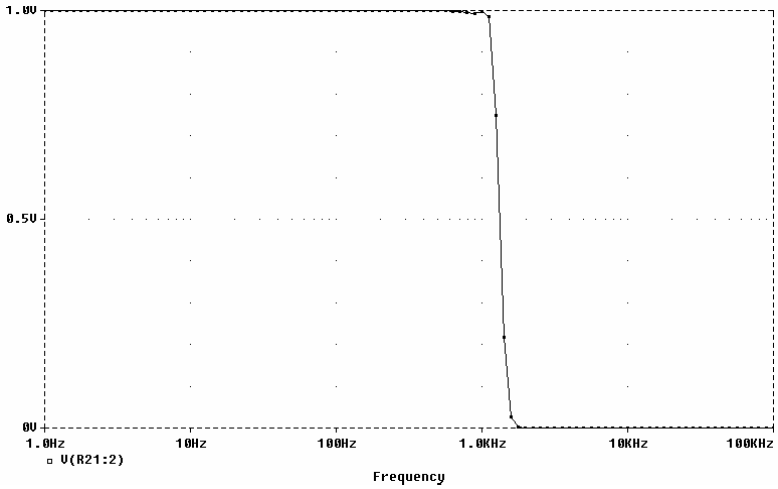
Mutation rate r	Without elitism, τ=0.02			With elitism, τ=0.02		
	Iteration	Count	Success	Iteration	Count	Success
0	5	2483	59.4	4.95	2473	54.4
0.114	5.63	2818	73.3	5.81	2905	78.2
0.2272	5.5	2746	70.3	5.8	2897	72.3
0.341	5.57	2789	70.3	5.87	2934	75.2
0.455	6.06	3029	69.3	5.79	2894	79.2
0.568	5.92	2959	72.3	6.69	3344	79.2
0.682	6.15	3076	84.1	7.375	3688	79.2
0.795	6.09	3046	75.2	6.49	3244	81.2
0.909	7	3500	74	6.54	3271	82.1
1.023	6.76	3379	86.1	7.24	3620	78
1.136	7.05	3525	79.2	7.5	3747	88

Without mutation choosing a too small population size leads to a small success rate, while choosing a too big population size is a waste of function evaluations. Both effects are overcome by using a high mutation rate. Thus, we used the mutation rate between 0.15τN to 0.5τN depending on the size of design space.

Then we consider a design of analog cells to Field Programmable Gate Arrays. The best circuit, having minimal amount of elements is given in Fig. 3, a. Each cell contains 6 elements only. From fig. 3, b we can see that the frequency response falls within the target specifications. PSPICE was used to evaluate the best circuit characteristics.



a



b

Fig. 3. The best circuit (a) and its characteristics (b)

CONCLUSIONS

In this paper we have developed hierarchy circuit representation and distribution-based technique to integral evolutionary analog circuit design. The results obtained have shown that distribution algorithms with mutation are both fast and requiring small computational efforts. Proposed algorithms require a smaller population size with comparison with [7 - 9]. Truncation selection with smaller τ has shown to be better in computational complexity. A using of elitism allows reserve best solutions during evolutionary search and to increase the success rate. In addition, by using mutation, the dependence on the right choice of the population size could be reduced.

ACKNOWLEDGMENTS

This work was supported by INTAS (International Association for the promotion of co-operation with scientists from the New Independent States of the former Soviet Union) under grant № YSF 00 - 190. We thank all participants from FhG in this cooperative project for interesting and useful discussions related to the topic of the paper.

The second and the last authors acknowledge support from RFBR under grant 01-01-00044.

REFERENCES

[1] Carley, L. R., Gielen, G. G. E., Rutenbar, R.A., Sansen, W.M.C.: Synthesis Tools for Mixed-Signal ICs: Progress on Frontend and Backend Strategies. Proceeding ACM/IEEE DAC (1996).

[2] Bentley, P. (ed.): Evolutionary design by computers. Morgan Kaufmann (1999).

[3] Kruiskamp, W., Leenaetrs, D.: Darwin: CMOS OpAmp Synthesis by means of a Genetic Algorithms. Proceeding ACM/IEEE DAC (1995).

[4] Krasnicki, M., Phelps, R., Rutenbar, R. A., Carley, L.R.: Maelstrom" Efficient Simulation-Based Synthesis for Custom Analog Cells. Proceeding ACM/IEEE DAC (1999).

[5] Vancorenland, P., De Ranter, C., Steyaert, M., Gielen, G.: Optimal RF Design Using Smart Evolutionary Algorithms. Proceeding ACM/IEEE DAC (2000).

[6] Zebulum, R.S., Pacheho, M.A., Vellasco, M.: Comparison of Different Evolutionary Methodologies Applied to Electronic Filter Design. Proc. 1998 IEEE Int. Conf. Evolutionary Computation. IEEE Press (1998) 434-439.

[7] Koza, J.R., Bennett, F. H., Andre, D., Keane, M. A., Dunlap, F.: Automated Synthesis of Analog Electrical Circuits by means of Genetic Programming. IEEE Trans. Evol. Comp., vol. 1, July (1997) 109-128.

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

[9] Lohn, J.D., Colombano, S.P.: A Circuit Representation Technique for Automated Circuit Design. IEEE Trans. on Evolutionary Computation, Vol.3, No. 3 (1999) 205 –219

[10] Ando, S., Iba, H.: Analog Circuit Design with a Variable Length Chromosome. Proc. IEEE Int. Conf. Evol. Computation, IEEE Press (2000) 994 - 1001

[11] Goh, C., Li, Y.: GA Automated Design and Synthesis of Analog Circuits with Practical Constraints. Proc. IEEE Int. Conf. Evol. Computation, IEEE Press (2001) 170 -177

[12] Grimbleby J.B.: Hybrid Genetic Algorithms for Analogue Network Synthesis. Proc. IEEE Int. Conf. Evol. Computation, IEEE Press (1999) 1781 -1787

[13] Mühlenbein, H., Paaß, G.: From Recombination of Genes to the Estimation of Distributions. Lecture Notes in Computer Science 1141: Parallel Problem Solving from Nature, PPSN IV (1996) 178-187

[14] Mühlenbein, H.: The Equation for Response to Selection and its Use for Prediction. Evolutionary Computation, 5 (1998) 303-346

[15] Mühlenbein, H., Mahnig, Th.: Optimal Mutation Rate Using Bayesian Priors for Estimation of Distribution Algorithms (in press)

[16] Mühlenbein, H., Mahnig, Th.: Evolutionary Algorithms: from Recombination to Search Distributions. In Kallel, L., Naudts, B., Rogers, A. (eds.): Theoretical Aspects of Evolutionary Computing. Natural Computing Springer Verlag (2000) 137-176

[17] Mühlenbein, H., Mahnig, Th.: Comparing the Adaptive Boltzmann Selection Schedule SDS to Truncation Selection. Proc. of the 3d International Symposium on Adaptive Systems. (2001) 121-128

[18] Coello, C.A., Christiansen, A.D., Aguirre, A.H., Use of Evolutionary Techniques to Automate the Design of Combinational Circuits. International Journal of Smart Engineering System design (1999)

[19] Kureichik, V.M., Zinchenko, L.A.: Evolution Design of Electronic Devices. Proc. ICECS 2000, 7 International conference on electronics, circuits and systems, v.2. (2000) 879-882

[20] Mühlenbein, H., Kureichik, V.M., Mahnig, Th., Zinchenko, L.A.: Evolutionary Algorithms with Hierarchy and Dynamic Coding in Computer Aided Design. Proceedings EUROGEN 2001, CIMNE. (2001)

[21] Mühlenbein, H., Kureichik, V.M., Mahnig, Th., Zinchenko, L.A.: Adaptive Algorithms of Evolutionary Modeling with knowledge for multi-agent CAD system. Proceedings International symposium NOLTA, Japan (2001) 299-302