Evolutional synthesis with incomplete information

Vitaly Gudilov and Victor Kureichik

Southern Federal University, Russia

Abstract. This article discusses the methods of evolutional synthesis of hardware that can be used for solving problems of hardware design in case of incomplete information about the object being synthesized. Development of a probabilistic evolutional algorithm is shown, and focus is made on the development of genetic operators, coding methods and assessment of circuits, as well as methods of probabilistic transferring hereditary information. Elements of inaccuracy are introduced into the algorithm by means of probabilistic methods for transferring information. They make it possible to escape from the classical concepts of solutions in evolutional algorithms presented in the form of a diagram with preset structure, to dynamically changing structure and data. This extends the scope of evolutional algorithms for designing complex or open systems by modeling open evolution.
[Gudilov V., Kureichik V. **Evolutional synthesis with incomplete information.** *Life Sci J* 2014;11(10s):359-363] (ISSN:1097-8135). http://www.lifesciencesite.com. 68

Keywords: synthesis with incomplete information, probabilistic methods of transferring hereditary information, probabilistic evolutional algorithms

Introduction

This article is based on solving the problem of building a probabilistic genetic algorithm where a fundamentally new for genetic algorithms method of transferring hereditary information is proposed. It is based on determining the probability of transferring to descendants a set of characteristics that determine the genotype of the elite ancestors' domain [1]. In the proposed method for transferring hereditary information, hereditary characteristics are not, unlike in classical genetic algorithms, transferred directly, but with a certain probability, which prevents premature convergence of the algorithm. At the stage of generating a new population this ensures evolution process with a variety of genetic material within the specified reaction norm, which rules out the necessity to use the crossing-over operator and models properties of non-heritable mutations.

Fuzzy methods of transferring hereditary information make it possible to work with arbitrary values of genes and chromosome structure, i.e., they act as the mechanism for maintaining the irregular reaction norm, which, for existing evolutional algorithms, is rigidly defined by the developer of genetic algorithm and remains invariable for the whole duration of evolutional search. Viability of the solutions obtained was assessed not from the point of view of adequacy to the circuit set by the developer (reaction norm), but from the point of view of mathematical model for each solution obtained.

Solving the problem of modeling open evolution using evolutional parameters is an important issue for developing the evolutional modeling theory [2] in the whole. These parameters are used for solving problems of designing complex [3, 4] and multi-level hierarchical systems [5]. Within the framework of this work, results of development and research of hardware evolution design methods have been shown. They are based on the principles of probabilistic transferring of hereditary information combined with abandoning the crossing-over operator.

Evolutional synthesis algorithm

Evolutional synthesis algorithms (ESA) are based on using a population (set, array) of alternative potential solutions. This is its fundamental difference from the methods with one possible solution. Each solution in ESA, gradually gets better, worse, or remains unchanged. The problem here is detecting "bad" solutions and rejecting them. After that, one has to choose those "good" solutions that will influence improving the remaining "bad" solutions and will create new, even better solutions. There is no distinct borderline between the "bad" and the "good" solutions. "Bad" solutions may make it possible to obtain optimal solutions in next generations, and vice versa. The methods where ideas, principles and approaches of population biology, genetics and evolution are used are called Evolutional Calculations (EC).

The main algorithm relating to EC is called Evolution Algorithm (EA)

EA's are divided into Genetic Algorithms (GA) and Evolution Strategies (ES). The first stage of a Generic EA (GEA) is the creation or initialization of a population of alternative solutions for the problem or the task in question. Usually, the population is created using a random, a randomly directed (adaptive), or an integrated method. In the first case, *n* random species (individuals or chromosomes), i.e., alternative solutions are created. In the second case, if the Decision Maker (DM) has information about

search areas where it is highly probable to find alternative solutions with a "good" Target Function, one can generate a population out of these areas. In the third case, alternatives are randomly chosen from the whole task area. In the third case, the first three are integrated.

The first case is called "the Blanket Principle" in the theory of Genetic Algorithms [2, 6]. The second one is called the "the Focusing Principle". The third one is called "the Shotgun Principle" The fourth one is called "the Integration Principle".

Genetic Algorithms (GA) are based on the biological evolution model and on random search methods [7-9]. GA work with a population of species (individuals or chromosomes) or of alternative solutions of the problem. Population evolves through using the recombination mechanism (combining two or more parents for obtaining descendants) and mutation (random individual change).

The evolutional synthesis algorithm for combinational circuits is shown in Figure 1. The frame contains the iteration part of the algorithm that includes generation of the population, analysis of synthesized circuits by building a Mathematical Model, assessing the circuits basing on calculation of criterion value, checking of finding the solution. If the criterion for stopping has not been reached (the set number of solutions has not been found or the set number of iteration cycles has not been made), elite area is formed and the probability is calculated. After calculating the probability for all genes of the chromosome, a transfer to the next iteration cycle occurs.

In work [1], authors proposed to use probabilistic methods for transmitting hereditary information for storage and transferring of functionally correct areas of the circuit underlying the developed probabilistic genetic algorithm. Thus, convergence of genetic material to the required parameters is faster, since each generation of synthesized circuits inherits many of the best parameters of circuit design from the previous iteration, and the estimates of probability introduce an element of randomness, which prevents premature convergence of the algorithm to local optimum values.

Let us consider the use of evolutional algorithms for solving the problem of circuit design on the example of combinational circuit synthesis. Let us present the problem of synthesizing combinational circuits as set $R = \{H, cF, P\}$, where element of set H determines the genotype of the synthesized solution, cF determines assessment of its mathematical model, and P determines probability of genetic material inheritance from the current population into the next one. The genotype of the synthesized solution H is determined using the chosen method of coding synthesized circuits, and codes presentation method for representing the architecture of the target device. Then building a mathematical model is none other than transition from architecture building method to representation (mathematical or physical) of the circuit synthesized. Then assessment of cF mathematical model is none other than a selection criterion that is the basis for selecting optimum solutions and forming a symptom of finding the solution.

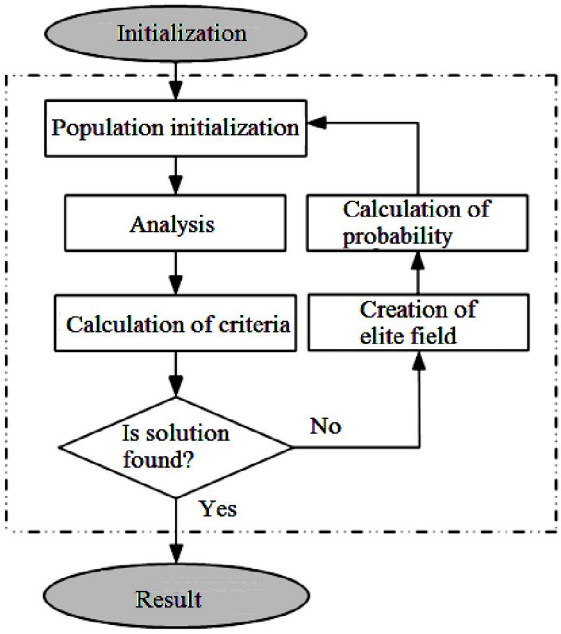

Fig. 1. Algorithm of evolutional synthesis in combinational circuits

For the algorithm of evolutional synthesis in combinational circuits for the basis of logical elements (LE), chromosome coding is determined on the basis of a matrix representation circuit $M_{m,n}$ (Figure 1a) containing abstract logical elements, by transferring in the synthesized circuit from a two-dimensional LE indexing (column m, line n) to the single-dimension one. Number of inputs x_c to the circuit make the zeroth column of element inputs $L_{0,n}$, where $n = c$ defines the number of lines in the $M_{m,n}$ matrix. Number of circuit outputs y_q is determined according to the number of expressions q describing functioning law of the desired circuit. Numbers of elements $L_{r,t}$ in the circuit are numbered continuously, matching the numbering of logic elements within the matrix $M_{m,n}$, and used in circuit synthesis for denoting element output. Indices of elements $L_{r,t}r$ and t are set according to the order of elements in the matrix $M_{m,n}$, ascending from left to right and from top to bottom, where $0 \le t < n$ and $0 \le r \le m$ (zero column $m = 0$ contains the circuit inputs signals, i.e., the number of columns m is complemented by the

column of inputs). Outputs of circuit y are only connected to the outputs of elements $L_{r,t}$ in column m ($r = m$), wherein an element of the zeroth line $L_{m,0}$ corresponds to the zeroth output y_0 of the circuit, etc.

As a minimum basis for LE, let us define set $L = \{Li \mid i = 1, 2, ..., nl\}$, elements of which are standard two-input AND, OR, XOR elements and single-input elements NOT and WIRE with one output, where nl is the number of elements in the set. Functional components of AND, OR, XOR and NOT elements are similar to their corresponding logical elements. The WIRE functional element (jumper) transmits to the output the signal coming to the only input of the element, without changing functional component of the input signal. Functional basis elements are encoded by setting the code to their ordinal numbers in ascending order.

Location of ALE in circuit grid points remains unchanged during the synthesis of the circuit, and only their functional component (code) and connections between input and output elements are changed. The logic element $L_{r,t}$ encoded by gene $g_{r,t}$ is populated into chromosome H successively, by columns, from the junior element t in column r to the senior one, where $0 < r \le m$, $0 \le t < n$. Each gene in chromosome H is defined by vector $gn = \{gn_i \mid _i = 1,2,3\}$, which encodes a separate LE $L_{r,t}$, where elements of vector gn_1 and gn_2 specify information about inputs r and t of the LE $L_{r,t}$, and gn_3 sets the code of encoded LE $L_{r,t}$. Gene locus in the chromosome corresponds to set position of the circuit element encoded by it, thus the value obtained at the output of the LE is associated with the gene identified with it. Transition to multi-input circuits is possible by supplementing the functional basis with new elements and modifying the structure of the gene in the chromosome. For analyzing the synthesized circuit, a method of building a mathematical model (MM) of the circuit is shown, with its subsequent analysis for $s = 2^c$ input sets of TI, where c is the number of variable input functions. Building an MM circuit for LE-based circuits is defined as a transition from a one-dimensional LE array that encodes the phenotype as a chromosome, to a number of two-dimensional arrays $Z_{m,n}$ (matrices with dimensions m times n), elements of which reflect the functional component of the object synthesized. Elements $Z_{m,n}$ are presented as matrices $Ze_{m,n}$, $Zin1_{m,n}$, $Zin2_{m,n}$, and $Zout_{m,n}$, where $Ze_{m,n}$ sets the matrix of LE codes, $Zin1_{m,n}$ and $Zin2_{m,n}$ define addresses of LE connected to inputs of the current LE, $Zout_{m,n}$ contains values of LE outputs. MM circuit is built by analyzing genes in the chromosome and building on their basis a $Ze_{m,n}$ LE codes matrix and matrices of inputs addresses $Zin1_{m,n}$ and $Zin2_{m,n}$. Matrices of LE outputs $Zout_{m,n}$ are generated for each set s of input signals in the circuit based on the set of states of input signals in TI on all possible input sets $s = 2^c$.

Elements $Zout_i$ in set Z can be represented as a matrix of dimensions n times m as follows:

$$Zout_i = \begin{bmatrix} Z(i)_{1,0} & Z(i)_{1,2} & \cdots & Z(i)_{1,m} \\ Z(i)_{2,0} & Z(i)_{2,2} & \cdots & Z(i)_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ Z(i)_{n,0} & Z(i)_{n,1} & \cdots & Z(i)_{n,m} \end{bmatrix},$$

where $i = 1, 2, ..., 2c$, and elements $Z(i)_{r,t}$, define the values of output signals of logic elements $L_{r,t}$ of the circuit synthesized with the set of signals i at the input of the circuit, with $0 \le r \le m$ and $0 \le t < n$. The zeroth column of $Zout_i$ matrices includes input sequence of signals i applied to inputs of the circuit. Order of evaluation of $Z(i)_{r,t}$ elements in matrix $Zout_i$ is specified as successive formation of $Z(i)_{r,t}$ elements in columns r. Elements $Z(i)_{r,t}$ in current column r take values based on execution of the logical operation. It is determined by code gn_{3c} of this element $L_{r,t}$, above input values coming to inputs of this element from outputs of elements $L_{v,t}$, where addresses of elements $L_{v,t}$ in the synthesized circuit are determined according to the value of elements $gn1r_{r,t}, gn1t_{r,t}, gn2r_{r,t}$ and $gn2t_{r,t}$ of matrices $gn_{1r}, gn_{1t}, gn_{2r}$ and gn_{2t}, respectively, and $0 \le v < r$.

Criterion selection

Degree of compliance with evolving and desired circuit is defined as criterion cF, with its value ranging between zero and one, i.e. with full compliance with evolving and desired circuit cF gets equal to the unity. The desired TI is determined basing on input functions, the evolving TI is calculated by setting s of input values from required TI to inputs of MM of the circuit with tabulating values from circuit outputs. The result of building an MM is an evolving TI THz, elements of which are compared to elements of the required TI $Tiff$, where TI are presented as matrices [1]:

$$TTz = \begin{bmatrix} z_{1,1} & z_{1,2} & \cdots & z_{1,q} \\ z_{2,1} & z_{2,2} & \cdots & z_{2,q} \\ \vdots & \vdots & \ddots & \vdots \\ z_{s,1} & z_{s,2} & \cdots & z_{s,q} \end{bmatrix} \quad TTf = \begin{bmatrix} f_{1,1} & f_{1,2} & \text{L} & f_{1,q} \\ f_{2,1} & f_{2,2} & \text{L} & f_{2,q} \\ \text{M} & \text{M} & \text{O} & \text{M} \\ f_{s,1} & f_{s,2} & \text{L} & f_{s,q} \end{bmatrix}$$

with dimensions s times q elements. Then the value of the hit_i relation is calculated as:

$$hit_i = \begin{cases} 0, \text{если } z_{i,v} \ne f_{i,v} \\ 1, \text{если } z_{i,v} = f_{i,v} \end{cases},$$

where $0 \le i<U$, $0 \le j < s$, $0 \le v < q$ and hit_i defines the result of executing the operation of comparing the i-th element of matrices TTz and TTf. Thus, the criterion for assessing cF for chromosome j is calculated on the basis of the following ratio:

$$cF(j) = \frac{\sum_{i=1}^{U} hit_i}{U}.$$

This value of the criterion assesses the functional component of the circuit and when the value is $cF(j) \to 1$, circuit j encoded by chromosome $H(j)$ tends to reach the desired solution. Thus, the parameter of evolution is the extent to which the circuit corresponds to the desired solution. The taking into account of the specifics of target devices functioning for MM, as well as methods for assessing circuits with memory, synchronization by fronts, etc., makes it possible to modify algorithms of synthesis for various circuits and target devices, thus expanding the scope of application for our synthesis algorithm.

Probabilistic methods of transferring genetic material

Genetic material is transferred basing on the best solutions. They are selected by a selection operator that performs transfer from population P_{cs} with size Ps, a set number Es of chromosomes $el(i)$ with the maximum value of criterion $cF(j)$ in descending order of $cF(j)$, where $0 < i \le Es$, $0 < j \le Ps$ and $i \le j$. The developed method of transferring hereditary information is defined by the probability vector $P = \{pi \mid i = 1, 2, ..., hL\}$ for inheritance of genetic material from a variety of elite chromosomes H_{el} from current P_{cs} population onto the next Π_{cs+1} population where hL is the length of the chromosome. The p_i value determines how much diversity of genetic material for gene g_i represented in locus i of chromosomes in elite area relates to value Es of the elite area. In case of binary chromosome encoding, value p_i defines probability of inheriting single genes:

$$p_i = \frac{\sum_{i=0}^{eS} g_i}{Es}.$$

With decimal encoding circuit, when the gene of a chromosome defines LE, elements p_i of vector P define probability of inheriting all possible states of gene g_i, and have length determined by the number of states that the gene g_i can take. Values of probability vectors p_i may range between zero and unity, and unity value means that genes in locus i of chromosomes H of next population Π_{cs+1} inherit dominant meaning of this gene's genetic material.

Initial population of a set number Ps of chromosomes H is generated randomly. In the process of evolution, i-th chromosome gene is generated on the basis of selecting probability values that determine probability of inheriting combinations for the i-th gene of the chromosome as a result of comparing a number generated randomly to values of the probability vector represented by the appropriate vector p_i.

Segregation method is proposed as an additional method for transferring genetic material, being based on the method of elitism (dominance of elite chromosomes), making it possible to accumulate genetic material, which has significant impact on the algorithm convergence [1]. Directed mutation method serves as a mechanism for preventing early convergence of algorithms. It does not destroy the circuit. It should be used only if the algorithm gets into a local optimum. In order to expand variety of topologies of the synthesized solutions, a procedure is proposed for withdrawing the population from the area of global extremum, based on reduction method that excludes chromosomes that represent the desired solution [10-13] from the population.

Results of the experiment

As comparison object, we will use results of works of experts [12] and results of the synthesis algorithm for combinational logic circuits based on an evolutionary approach, as presented in work [12]. In the algorithm of synthesis of combinational logic circuits based on probabilistic genetic algorithm [1], the following synthesis parameters were set: probability of crossing-over operator: 35%; probability of mutation operator: 50%; population size: 1,200 chromosomes; number of generations: 200; circuit search space is limited to a 4 times 4 elements matrix. At the 50th iteration, genetic search resulted in finding the best solution. The obtained circuit consisted of 9 elements: 3 OR, 3 XOR, 1 NOT and 2 WIRE elements. Search time for the solution was 102 seconds.

In the shown algorithm of evolutionary synthesis of combinational circuits, the following synthesis parameters had been set: population size: 90 chromosomes; elite area: 20 chromosomes; dominance of elite chromosomes: 6; mutation coefficient: 0.235%, number of iteration cycles: 300; circuit size was limited to 4 times 5 elements matrix. At the 65th iteration (chromosome No.1), evolutional synthesis resulted in finding the best solution. The obtained circuit consisted of 7 elements: 1 NOT, 3 OR, and 3 XOR elements. Search time for the solution was 1,360 milliseconds. Compared results are shown in Table. 1.

Comparing synthesis results shown in Table 1, we can conclude that the algorithm for automated synthesis of combinational circuits ensures synthesis of similar quality circuits in shorter time.

Table 1. Comparison of results of combinational circuits synthesis based on work of experts, genetic algorithm and combinational circuits evolutionary synthesis algorithm

Algorithm	Result	Search time for the solution.
Algorithm for evolutional synthesis of combinational circuits	7 gates: F=((A□D)+B)'+((A+D) □(A□C)) 1 – NOT, 3 – OR, 3 – XOR.	1.36 sec.
Genetic search [14]	7 gates: F=(C+(A□D))'+((A+D) □(B□D)) 3 – OR, 3 – XOR, 1 – NOT	102 sec.
Expert 1	9 gates: F=((A□B)□((AD)(B+C)))+((A+C)+D) 2 – AND, 2 – OR, 2 – XOR, 1 – NOT	Undefined
Expert 2	10 gates: F=A'B+A(B'D'+C'D) 4 – AND, 2 – OR, 4 – NOT	Undefined

Results of studying the algorithm of evolution synthesis of combinational circuits showed better performance of this algorithm compared to its analogues in case of software implementation. In work [1, 4], a detailed description of hardware implementation of evolutional synthesis algorithm is given that makes it possible to increase operating speed of this algorithm more than 4 times compared to software implementation.

Conclusion

The Evolutional synthesis algorithm for combinational circuits proposed in the article can be generalized for solving more complex tasks used for searching successive circuits within the framework of defined functional basis of elements that correspond to required time and topology limitations for the case where the task is not formulated clearly, or with loss of some information that does not have first-rank importance. The obtained experimental results let us state that the developed methods and algorithms not only can be used as evolutional search strategies, but also show the advantage of the solutions proposed in comparison to existing methods of practical problem solving. Consequently, solving the problem of structure synthesis using evolutional algorithms is required for further research in the area of building algorithms for evolutional design of complex systems where either separate parts, or overall requirements to the system are often incorrectly or incompletely described [13,14]. In this case, structural synthesis is viewed together with parametric and functional synthesis to find optimum design and technological solutions.

The study was performed by the grant from the Russian Science Foundation (project № 14-11-00242) in the Southern Federal University.

Corresponding Author:
Dr. Gudilov Vitaly, Southern Federal University, Russia.

References

1. Gudilov, V.V. and V.M. Kureichik, 2010. Combinational circuits evolutional synthesis algorithms. Monograph. Taganrog: Editing House of SFU TSI, pp: 160.
2. Kureichik, V.V. and Y.A. Kravchenko, 2013. Bioinspired algorithm applied to solve the travelling salesman problem. World Applied Sciences Journal, 22(12): 1789-1797.
3. Zebulum, R.S., M.A. Pacheco and M.M. Vellasco, 2002. Evolutionary Electronics: Automatic Design of Electronic Circuits and Systems by Genetic Algorithms. USA, CRC Press LLC, pp: 304.
4. Knyazeva, E.N. and S.P. Kurdyumov, 1994. Laws of evolution and self-organization of complex systems. Moscow: Nauka, pp: 238.
5. Mesarovich, M.,D. Mako and I. Takahara, 1973. Theory of hierarchical multilevel systems.Moscow: Mir, pp: 344.
6. Gladkov, L.A., V.V. Kureichik, Y.A. Kravchenko, 2013.Evolutionary Algorithm for Extremal Subsets Comprehension in Graphs [Text]. World Applied Sciences Journal, 27(9): 1212-1217.
7. Goldberg, D.E., 1989. Genetic Algorithms in Search, Optimization and Machine Learning. USA: Addison-Wesley Publishing Company, Inc., pp: 372.
8. Practical Handbook of Genetic Algorithms, Vol.3, 1999. Eds., Chambers, I. Washington: CRC Press,pp: 592.
9. Davis, L., 1987. Genetic Algorithms and Simulated Annealing. San Mateo,USA: Morgan Kaufman Publisher,pp: 193-207.
10. Yarushkina, N.G., 2007. Fuzzy hybrid systems. Moscow: Physics and Math Literature,pp: 208.
11. De Jong, K., 1996. Evolutionary Computation: Recent Development and Open Issues. In the Proceedings of the 1st International Conference "Evolutionary Computation and Its Application, EvCA' 96", Moscow, pp: 7-18.
12. Luke, S., 2011. Essentials of Metaheuristics. A set of Undergraduate Lectures Notes. Zaroth Edition,pp: 230.
13. Cherun, S.V., 2002. Evolutional design of logical circuits, Promising information technologies and intellectual systems, 4: 30-34.
14. Zaporozhets, D.Y., D.V.Zaruba and V.V.Kureichik, 2013. Hybrid bionic algorithms for solving problems of parametric optimization. World Applied Sciences Journal, 23(8): 1032-1036.

6/24/2014