Evolutionary Design and Optimization of Digital Circuits using Imperialist Competitive Algorithm

Mehdi Anjomshoa
Department of Elec. Eng
Islamic Azad University,
Bousher Branch
Boushehr, Iran

Ali Mahani
Department of Elec. Eng
Shahid Bahonar University
Kerman, Iran

Mostafa Esmaeil beig
Department of Elec. Eng
Islamic Azad University,
Bousher Branch
Boushehr, Iran

ABSTRACT

This paper describes the application of Imperialist Competitive Algorithm (ICA) to design and optimization of combinational logic circuits. Imperialist Competitive Algorithm is a new socio-politically motivated global search strategy that recently has been introduced for dealing with different optimization tasks. We proposed a cost function to evolve circuits at gate level with lower number of transistors. By decreasing the total number of transistors, the area of circuit will be optimized too. The performance of the proposed algorithm is evaluated using different circuits from literature. The simulation results clearly demonstrate the validity of this new technique. We can consider this heuristic algorithm as a search engine in evolutionary hardware applications.

Keywords

Combinational logic Circuit; Evolutionary Hardware;
Imperialist Competitive Algorithm.

1. INTRODUCTION

Evolutionary hardware (EHW) is an interesting field of electronic circuit design and recently many researchers have been focused on it [1, 2, 3]. EHW is an automatic technique to design hardware, based on the evolutionary algorithms.

Due to the fast improvement in digital electronic devices, regarding to their size, it is important to have compact and optimized circuits, in such devices. Logic circuits are implemented as integrated circuits on the chip. In this case, the lower number of transistors which forming logic gates, leads to lower circuit size on chip and production cost will be cheaper.

Actually, many hardware design approaches, such as Boolean algebra, Karnaugh map [4], and Quine–McCluskey [5] have been widely used in solving the digital circuit optimization problem. These methods depend on human knowledge and creativity and in case of huge and complex circuits; they could not effectively have the satisfactory solution.

Many machine learning or artificial intelligent skills have been employed into optimal logic circuit design. The main idea of these studies is to integrate the evolutionary algorithms (EAs) into the hardware encoding. The process of evolutionary circuit design is fundamentally different form traditional design process, because it is not based on designer knowledge and experience The common skills used in these studies include: NGA (Genetic Algorithm with N-cardinality representation) [6, 7], MGA (Multiobjective Genetic Algorithm) [8], MLCEA (Multi-Layer Chromosome Evolutionary Algorithm) [9]. All of these methods try to minimize the number of gates in the circuits, but MLCEA-TC (Multi-Layer Chromosome Evolutionary Algorithm – Transistor Count) [10,11] introduced to find the best hardware design with respect to transistor count.
Recently, a new Evolutionary Algorithm has been proposed by Atashpaz-Gargari and Lucas [12], in 2007 which has inspired from a socio-political phenomenon, called Imperialist Competitive Algorithm (ICA) and it has been introduced for dealing with different optimization tasks [13-15]. In this paper, a new logic circuit design technique based on Imperialist Competition Algorithm is proposed.

The rest of this paper is organized as follows. In section II, ICA algorithm will be introduced. Section III describes structure of circuit and the use of ICA as a new approach for the automatic design of an optimized circuit. Section IV shows experiments and compares them with works which have proposed before. Finally, Section V concludes.

2. IMPERIALIST COMPETITIVE ALGORITHM

Imperialist Competitive Algorithm (ICA) [12] is a new evolutionary algorithm in the Evolutionary Computation field based on the human's socio-political evolution. The algorithm starts with an initial random population called countries. Some of the best countries in the population selected to be the imperialists and the rest form the colonies of these imperialists. In an N dimensional optimization problem, a country is a 1× n array. This array defined as below

$$Country = [p_1, p_2, \dots, p_n] \qquad (1)$$

The cost of a country is found by evaluating the cost function f at the variables $(p_1, p_2, \dots, p_n)$. Then

$$c_i = F(Country_i) = F(p_{i1}, p_{i2}, \dots, p_{in}) \qquad (2)$$

The algorithm starts with N initial countries and the N_{imp} best of them (countries with minimum cost) chosen as the imperialists. The remaining countries are colonies that each belong to an empire. The initial colonies belong to imperialists in convenience with their powers. To distribute the colonies among imperialists proportionally, the normalized cost of an imperialist is defined as follow:

$$C_n = \max\{c_i\} - c_n \tag{3}$$

Where, c_n is the cost of nth imperialist and C_n is its normalized cost. Each imperialist that has more cost value, will have less normalized cost value. Having the normalized cost, the power of each imperialist is calculated as below and based on that the colonies distributed among the imperialist countries.

$$p_n = \left| \frac{C_n}{\sum_{i=1}^{N_{imp}} C_i} \right| \tag{4}$$

On the other hand, the normalized power of an imperialist is assessed by its colonies. Then, the initial number of colonies of an empire will be $NC_n = rand\ \{p_n \times N_{col}\}$ where, NC_n is initial number of colonies of nth empire and N_{col} is the number of all colonies.

To distribute the colonies among imperialist, NC_n of the colonies is selected randomly and assigned to their imperialist. The imperialist countries absorb the colonies towards themselves using the absorption policy. The absorption policy shown in Figure 1 makes the main core of this algorithm and causes the countries move towards to their minimum optima. The imperialists absorb these colonies towards themselves with respect to their power that described in (4). The total power of each imperialist is determined by the power of its both parts, the empire power plus percents of its average colonies power.

$$\begin{aligned} T.C_n &= Cost(imperialist_n) \\ &+\xi mean\{Cost(colonies\ of\ impire_n)\} \end{aligned} \tag{5}$$

Where $T.C_n$ is total cost of the nth empire and ξ is a positive number which is considered to be less than one.

In the absorption policy, the colony moves towards the imperialist by x unit. The direction of movement is the vector from colony to imperialist, as shown in Figure 1, the distance between the imperialist and colony shown by d and x is a random variable with uniform distribution. Where β is greater than 1 and is near to 2.

$$x \sim U(0, \beta \times d) \tag{6}$$

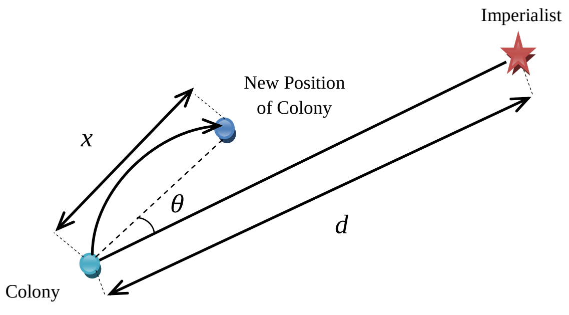


Fig 1. Moving colonies toward their relevant Imperialist [12]

In ICA algorithm, to search different points around the imperialist, a random amount of deviation is added to the direction of colony movement towards the imperialist. In Figure 1, this deflection angle is shown as θ, which is chosen randomly and with a uniform distribution.

$$\theta \sim U(-\gamma, \gamma) \tag{7}$$

As shown in Figure 2 While moving toward the imperialist countries, a colony may reach to a better position, so the colony position changes according to position of the imperialist.

In each iteration we select some of the weakest colonies and replace them with new ones, randomly. The replacement rate is named as the revolution rate.

In this algorithm, the imperialistic competition has an important role. During the imperialistic competition, the weak empires will lose their power and their colonies. To model this competition, firstly we calculate the probability of possessing all the colonies by each empire considering the total cost of empire.

$$NTC_n = \max\{TC_i\} - TC_n \tag{8}$$

Where, TC_n is the total cost of nth empire and NTC_n is the normalized total cost of nth empire. Having the normalized total cost, the possession probability of each empire is calculated as below

$$p_{p_n} = \left| \frac{NTC_n}{\sum_{i=1}^{N_{imp}} NTC_i} \right| \tag{9}$$

After a while all the empires except the most powerful one will collapse and all the colonies will be under the control of this unique empire. Figure 3 shows the flowchart of this algorithm.

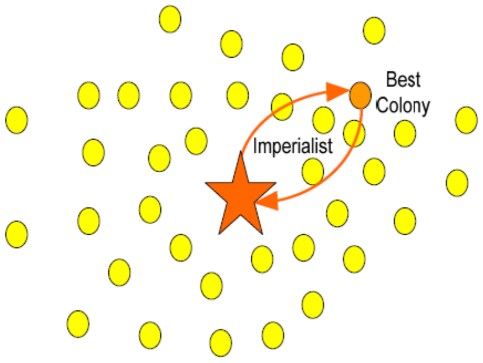


Fig 2. Exchanging the positions of a colony and the Imperialist [12]

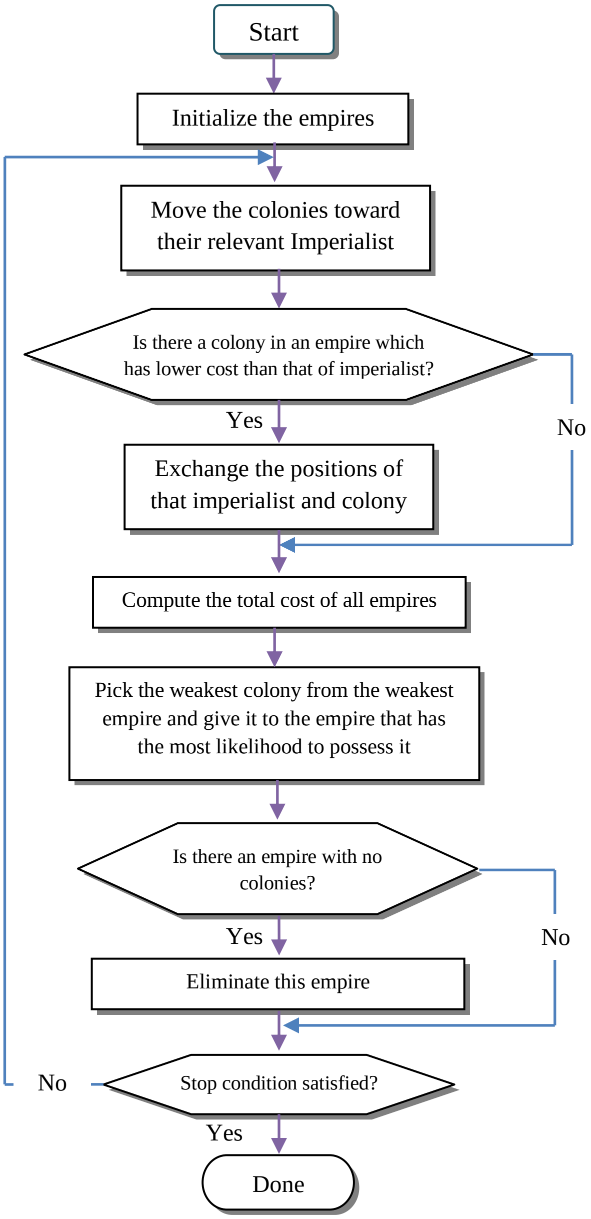

Start
Initialize the empires
Move the colonies toward their relevant Imperialist
Is there a colony in an empire which has lower cost than that of imperialist?
Yes
No
Exchange the positions of that imperialist and colony
Compute the total cost of all empires
Pick the weakest colony from the weakest empire and give it to the empire that has the most likelihood to possess it
Is there an empire with no colonies?
Yes
No
Eliminate this empire
No
Stop condition satisfied?
Yes
Done

Fig 3. Flowchart of the ICA algorithm [12]

3. THE PROPOSED APPROACH

3.1 Circuit Structure

In evolutionary hardware design regarding to digital circuits, usually a random matrix encoding is used as circuit structure. This matrix structure was proposed by Louis and Rawlins in year 1991 [16] and it is shown in Figure 4. Generally the logic circuit can be described by a two-dimensional logic cell matrix. Each cell of the n × m matrix contains the information of the gate type and its corresponding inputs. Unlike the fixed interconnection rules used in [16], the inputs of each unit can be randomly connected by any element output in previous stages. As shown in figure 5, each cell of the circuit matrix is encoded in an array of five integer number. First and second number of array is related to the first input of gate and third and fourth number of array is for second input and finally fifth number indicates gate type.

Truth table is the evaluation criterion of combinational logic circuit in the evolutionary algorithm.

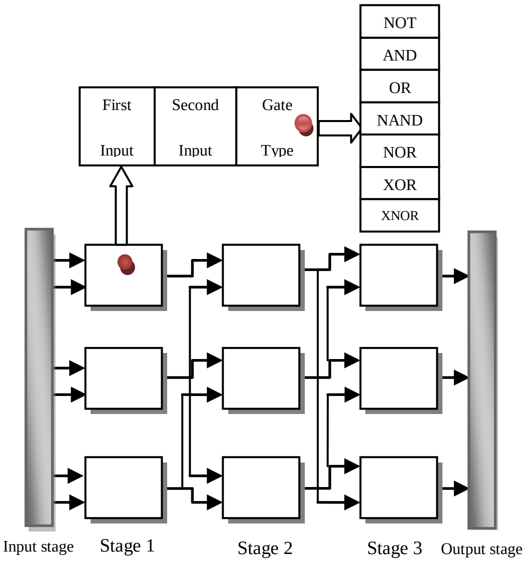


Fig 4. Random circuit matrix [16]

We used different logic cells (NOT, AND, OR, NAND, NOR, XOR, XNOR and wire) in our algorithm. Wire is assumed as a logic gate that transfers data from its input to output without any change. Cells information which was extracted from the open source standard cells vsclib013 [17] in 0.13 micron technology is shown in Table 1.

Figure 5 shows how logic cells are encoded in matrix of circuit structure. This circuit has 3 inputs and one output. Cell (1,3) whose attribute is (1,2,2,2,6) is an XOR gate (according to Table 1).Output of circuit is the output of this gate and the first input of this cell is connected to the output of cell (1,2), which is a WIRE, and the second input is connected to the output of a NOR gate in (2,2). There is an AND gate in cell (1,1) that is connected to primary inputs A, B.

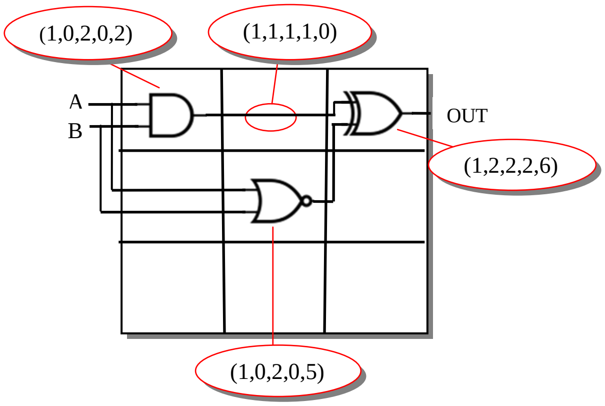
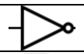
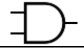
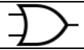
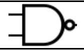
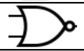
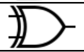
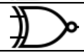


Fig 5. Example of a circuit and its encoding

Table 1. Gate types, Gate ID, corresponding CMOS transistors [17]

Gate Type Code	Gate Symbol	Area (µm)	Number Of Transistors
0	—	0	0
1		1728	2
2		2880	6
3		2880	6
4		2304	4
5		2304	4
6		4608	9
7		5184	9

3.2 Modification Og Algorithm

The original version of Imperialist competitive algorithm [12] operates on real values. As shown in figure 1 distance between colony and imperialist is d and colony moves toward imperialist by x unit.

In digital circuit design with ICA, we deal with matrix structure as countries. Moving these matrix structures towards each other is nonsense. So we propose a modification for assimilation operator.

Assimilating operator shown in Figure 6 is as follows:

1) Select one column of imperialist matrix structure randomly.
2) Replace selected column of imperialist matrix with the same column in colonies matrix structure.

By replacing cells from imperialist to colony, matrix of colonies will approach imperialist matrix.

For revolution operator, we select some colonies (circuits) in each imperialist and replace them with random circuit structures.

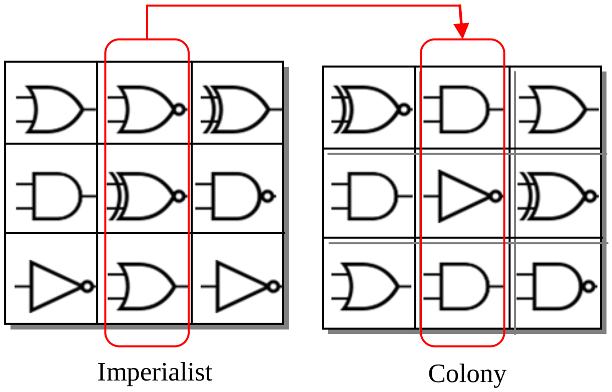


Fig 6. Modification of moving colonies toward Imperialist by replacing one column from imperialist to colony

3.3 Cost Function

.It is important for us to have a circuit with fewer gates and lower number of transistors and also optimized area. In order to achieve this aim we define a weighted cost function as follows:

$$f = \frac{w_{match} \times N_{match} + w_{transistor} \times N_{transistor}}{w_{match} + w_{transistor}} \qquad (10)$$

Where N_{match} is:

$$N_{match} = \frac{\text{the number of correct outputs from circuit}}{\text{the number of outputs from truth table}} \qquad (11)$$

And $N_{transistor}$ is the total number of transistors used in the gates of circuit structure.

Since, it is important to have functional circuit first and then optimized; we assign 10 for w_{match} and 1 for $w_{transistor}$.

4. EXPERIMENTS AND RESULTS

The proposed ICA method has been simulated in Matlab2009. For evaluating the effectiveness of the proposed method, three combinational logic circuit designs were used for experiments. We compare our results with traditional methods and methods based on genetic algorithms [6-9] and MLCEA-TC [10] and GA-TC method described in paper [11].

ICA parameters are listed in Table 2. Total number of countries is 300 and usually 10% of them assigned as colonies.

Truth tables of three test circuits are presented in table 3, 4. In truth tables symbol “In” represent inputs, and symbol ”O” corresponds to circuit outputs.

Table 2. Proposed algorithm (ICA) parameters

Parameter	Value
Number of Countries	300
Number of Imperialists	30
Zeta	0.05
Revolution rate	0.2

Table 3. Truth tables of circuits 1, 2

Circuit No. 1					Circuit No. 2						
In				O	In				O		
A	B	C	D	F	A_1	A_0	B_1	B_0	X_3	X_2	X_1
0	0	0	0	1	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	1	0	0	1
0	0	1	0	0	0	0	1	0	0	1	0
0	0	1	1	0	0	0	1	1	0	1	1
0	1	0	0	1	0	1	0	0	0	0	1
0	1	0	1	1	0	1	0	1	0	1	0
0	1	1	0	1	0	1	1	0	0	1	1
0	1	1	1	1	0	1	1	1	1	0	0
1	0	0	0	1	1	0	0	0	0	1	0
1	0	0	1	1	1	0	0	1	0	1	1
1	0	1	0	1	1	0	1	0	1	0	0
1	0	1	1	0	1	0	1	1	1	0	1
1	1	0	0	0	1	1	0	0	0	1	1
1	1	0	1	1	1	1	0	1	1	0	0
1	1	1	0	0	1	1	1	0	1	0	1
1	1	1	1	1	1	1	1	1	1	1	0

Table 4. Truth table of circuit 3

Circuit No. 3							
In				O			
A_1	A_0	B_1	B_0	C_3	C_2	C_1	C_0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0
0	0	1	0	0	0	0	0
0	0	1	1	0	0	0	0
0	1	0	0	0	0	0	0
0	1	0	1	0	0	0	1
0	1	1	0	0	0	1	0
0	1	1	1	0	0	1	1
1	0	0	0	0	0	0	0
1	0	0	1	0	0	1	0
1	0	1	0	0	1	0	0
1	0	1	1	0	1	1	0
1	1	0	0	0	0	0	0
1	1	0	1	0	0	1	1
1	1	1	0	0	1	1	0
1	1	1	1	1	0	0	1

Our first example has 4 inputs and one output and second example has 4 inputs and 3 outputs. This circuit represents a 2 bit adder and finally our third example is a circuit with 4 inputs and 4 outputs which represents a 2 bit multiplier. Figures 7, 8, 9 show the circuits designed with proposed method (ICA) and comparisons of our results with other methods for 3 circuits are shown in Table 5, 6 and 7. In comparison Tables, NG and NT represents for number of gates and number of transistors respectively.

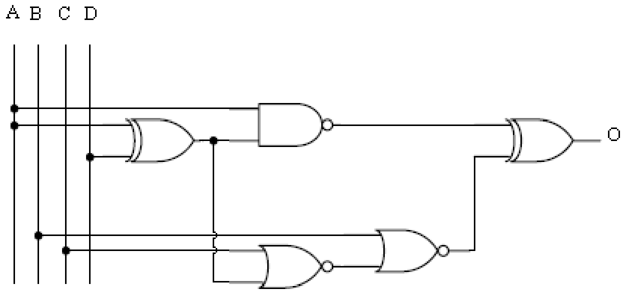


Fig 7. Circuit schematic designed by ICA for Example 1

Logic function obtained by proposed method for first example is:

$F= ((A.\ (A \oplus D))' \oplus (B+(C+ (A \oplus D))')')$

Table 5. Comparison results for first circuit

Method	NG	NT	Area(μm)
KM	9	56	28224
NGA,MGA	7	47	24192
GA-TC	8	42	23040
MLCEA-TC	7	38	20736
ICA	5	30	16128

The percentage of improvement regarding to the best previous method (MLCEA-TC) for gate number is 28.5%, number of transistors 21% and for area 22%.

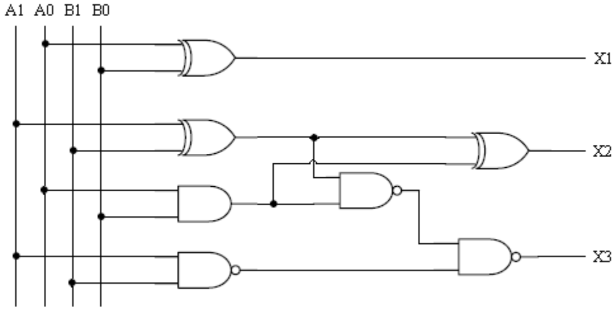


Fig 8. Circuit schematic designed by ICA for Example 2

Logic function obtained by proposed method for second example is:

$X1= (A0 \oplus B0)$

$X2= ((A1 \oplus B1) \oplus (A0.B0))$

$X3= (((A1 \oplus B1).(A0.B0))'.(A1.B1)')'$

Table 6. Comparison results for second circuit

Method	NG	NT	Area(μm)
KM	12	77	38592
MGA,NGA, MLCEA	7	51	25344
MLCEA-TC	9	47	26496
ICA	7	45	23616

The percentage of improvement regarding to the best previous method for number of transistors is 4.2% (compare to MLCEA-TC) and comparing to NGA, MGA and MLCEA, area is 6.8% and gate number is the same.

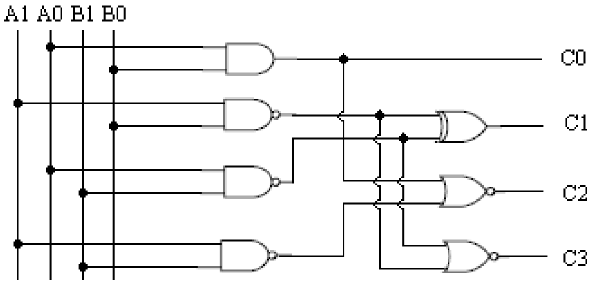


Fig 9. Circuit schematic designed by ICA for Example 3

Logic function obtained by proposed method for third example is:

$C0= (A0.B0)$

$C1= ((A1.B0)' \oplus (A0.B1)')$

$C2= ((A0.B0) + (A1.B0)')'$

$C3= ((A1.B0)' + (A0.B1)')'$

Table 7. Comparison results for third circuit

Method	NG	NT	Area(μm)
KM	12	71	35136
BGA	8	54	26496
NGA	7	48	23616
GA-TC, MLCEA-TC	-	-	-
ICA	7	35	19008

The percentage of improvement regarding to the best previous method for number of transistors is 27% and for area is 19.5% and gate number is the same as previous method (NGA).

5. CONCLUSION

In this paper, a new logic circuit design technique by Imperialistic Competitive Algorithm presented. Original version of Imperialist competitive algorithm operates on real values. We modified assimilation operator to deal with digital circuit structure. Also we proposed a weighted cost function to optimize circuit with respect to gate number, number of transistors and circuit area. The results show that the proposed method is able to optimize circuits better than previous methods.

6. REFERENCES

[1] D. Chen, T. Aoki, N. Homma, T. Terasaki & T. Higuchi, “Graphbased evolutionary design of arithmetic circuits. IEEE Transaction on Evolution and Computation, Vol. 6, No. 1, pp. 86–100, 2002.

[2] Shuguang Zhao, Study of the Evolutionary Design Methods of Electronic Circuits, Ph.D. dissertation (in Chinese), Xidian University, 2003.

[3] C. Y. Chen & R. C. Hwang, "A new variable topology for evolutionary hardware design", Expert Systems With Applications, Vol. 36, No. 1, pp. 634-642, 2009.

[4] M. Karnaugh, A map method for synthesis of combinational logic circuits, Transactions of the AIEE, Communications and Electronics, 72(I):593- 599, November 1953.

[5] E.J. McCluskey, Minimisation of Boolean functions, Bell Systems Technical Journal, 35(5):1417-1444, November 1956.

[6] C. A. Coello, A. D. Christiansen, and A. H. Aguirre, “Use of evolutionary techniques to automate the design of combinational circuits”, Int. J. Smart Engineering System Design 2 (4), 299–314 (2001).

[7] C. A. Coello, A. D. Christiansen, and A. H. Aguirre, “Automated design of combinational logic circuits using genetic algorithms”, Proc. Int. Conf. on Artificial Neural Nets and Genetic Algorithms, 335–338 (1997).

[8] C. A. Coello, A. H. Aguirre, and B. P. Buckles, “Evolutionary multiobjective design of combinational logic circuits”, Proc. 2nd NASA/DoD Workshop on Evolvable Hardware, 161–170 (2000).

[9] A. Słowik and M. Białko, “Design and optimization of combinational digital circuits using modified evolutionary algorithm”, Proc. 7th Int. Conf. on Artifficial Intelligence and Soft Computing 3070, 468–473 (2004).

[10] A. Słowik and M. Białko., “Evolutionary design and optimization of combinational digital circuits with respect to transistor count”, Bulletin of the Polish Academy of Sciences,Technical Sciences Vol. 54, No. 4, 2006.

[11] P. Nilagupta and N. Ou-thong, “Logic function minimization base on transistor count using genetic algorithm”, Proc. 3rd Information and Computer Engineering Postgraduate Workshop, Songkla, Thailand, 2003.

[12] E. Atashpaz-Gargari and C. Lucas, “Imperialist Competitive Algorithm: An Algorithm for Optimization Inspired by Imperialistic Competition,” IEEE Congress on Evolutionary Computation (CEC 2007). pp 4661-6447, 2007.

[13] Lucas, C., Nasiri-Gheidari, Z., Tootoonchian, F. "Application of an imperialist competitive algorithm to the design of a linear induction motor”. Energy Conversion and Management 51(7): 1407–1411, 2010.

[14] Mozafari, Hamid; Abdi, Behzad; Ayob, Amran. "Optimization of Transmission Conditions for Thin Interphase Layer Based on Imperialist Competitive Algorithm". (IJCSE) International Journal on Computer Science and Engineering 2 (7): 2486–2490, 2010.

[15] H. Sayadnavard, Monireh; T. Haghighat, Abolfazl; Abdechiri, Marjan "Wireless sensor network localization using Imperialist Competitive Algorithm". 3rd IEEE International Conference on Computer Science and Information Technology, 2010.

[16] S. J. Louis, G. J. E. & Rawlins, G. J. E., “Designer genetic algorithms: Genetic algorithms in structure design”, Fourth international conference on genetic algorithms. San Mateo, CA: Morgan Kauffman, pp. 53–60, 1991.

[17] G. Petley. ASIC standard cell library design. http://www.vlsitechnology.org/.