Xuesong YAN[1,2], Chengyu HU[1], Hong YAO[1], Yuanyuan FAN[1], Qingzhong LIANG[1], Chao LIU[1]

China University of Geosciences (Wuhan) (1), University of Central Arkansas (2)

Electronic Circuit Optimization Design Algorithm based on Particle Swarm Optimization

***Abstract**. A major bottleneck in the evolutionary design of electronic circuits is the problem of scale and the time required to evaluate the individuals, traditional genetic algorithm cannot solve these problems well. Particle Swarm Optimization (PSO) algorithm was developed under the inspiration of behavior laws of bird flocks, fish schools and human communities. In this paper, we use the PSO algorothm to solve the electronic circuit optimization design. The new algorithm keeps not only the fast convergence speed, but effectively improves the capability of global searching as well. The experiment results show that the PSO algorithm is efficient than traditional genetic algorithm.*

***Streszczenie.** W artykule opisano zastosowanie algorytmu optymalizacji rojem cząstek w rozwiązywaniu zagadnienia optymalnego projektowania układów elektronicznych. Proponowane rozwiązanie pozwala na uzyskanie dużej szybkości konwergencji oraz efektywne polepszenie możliwości wyszukiwania globalnego. Wyniki eksperymentalne pokazują, że algorytm PSO jest efektywniejszy niż typowy algorytm genetyczny. (**Algorytm optymalizacji w projektowaniu układów elektronicznych z wykorzystaniem optymalizacji rojem cząstek**).*

Keywords: electronic circuit, particle swarm optimization, circuit optimization design, genetic algorithm
Słowa kluczowe: układ elektroniczny, optymalizacja rojem cząstek, projektowanie optymalizujące układ, algorytm genetyczny.

Introduction

Evolutionary Electronics applies the concepts of genetic algorithms to the evolution of electronic circuits. The main idea behind this research field is that each possible electronic circuit can be represented as an individual or a chromosome of an evolutionary process, which performs standard genetic operations over the circuits. Due to the broad scope of the area, researchers have been focusing on different problems, such as placement, Field Programmable Gate Array (FPGA) mapping, optimization of combinational and sequential digital circuits, synthesis of digital circuits, synthesis of passive and active analog circuits, synthesis of operational amplifiers, and transistor size optimization. Of great relevance are the works focusing on “intrinsic” hardware evolution in which fitness evaluation is performed in silicon, allowing a higher degree of exploration of the physical properties of the medium. This particular area is frequently called Evolvable Hardware [1-3]. A major bottleneck in the evolutionary design of electronic circuits is the problem of scale. This refers to the very fast growth of the number of gates, used in the target circuit, as the number of inputs of the evolved logic function increases. This results in a huge search space that is difficult to explore even with evolutionary techniques. Another related obstacle is the time required to calculate the fitness value of a circuit. Then the traditional genetic algorithm is being trapped easily into a local optimum and the convergence speed is slow.

Particle Swarm Optimization (PSO) algorithm was an intelligent technology first presented in 1995 by Eberhart and Kennedy, and it was developed under the inspiration of behavior laws of bird flocks, fish schools and human communities [4]. If we compare PSO with Genetic Algorithms (GAs), we may find that they are all maneuvered on the basis of population operated. But PSO doesn't rely on genetic operators like selection operators, crossover operators and mutation operators to operate individual, it optimizes the population through information exchange among individuals. PSO achieves its optimum solution by starting from a group of random solution and then searching repeatedly. Once PSO was presented, it invited widespread concerns among scholars in the optimization fields and shortly afterwards it had become a studying focus within only several years. A number of scientific achievements had emerged in these fields [5-7]. PSO was proved to be a sort of high efficient optimization algorithm by numerous research and experiments [8]. PSO is a meta-heuristic as it makes few or no assumptions about the problem being optimized and can search very large spaces of candidate solutions. However, meta-heuristics such as PSO do not guarantee an optimal solution is ever found. More specifically, PSO does not use the gradient of the problem being optimized, which means PSO does not require that the optimization problem be differentiable as is required by classic optimization methods such as gradient descent and quasi-Newton methods. PSO can therefore also be used on optimization problems that are partially irregular, noisy, change over time, etc.

Particle Swarm Optimization Algorithm

PSO was presented under the inspiration of bird flock immigration during the course of finding food and then be used in the optimization problems. In PSO, each optimization problem solution is taken as a bird in the searching space and it is called “particle”. Every particle has a fitness value which is determined by target functions and it has also a velocity which determines its destination and distance. All particles search in the solution space for their best positions and the positions of the best particles in the swarm. PSO is initially a group of random particles (random solutions), and then the optimum solutions are found by repeated searching. In the course of every iterations, a particle will follow two bests to renew itself: the best position found for a particle called pbest; the best position found for the whole swarm called gbest. All particles will determine following steps through the best experiences of individuals themselves and their companions. For particle id, its velocity and its position renewal formula are as follows:

(1) $V_{id}^{'} = \omega V_{id} + \eta_1 rand()(P_{idb} - X_{id}) + \eta_2 rand()(P_{gdb} - X_{id})$

(2) $X_{id}^{'} = X_{id} + V_{id}^{'}$

In here: ω is called inertia weight, it is a proportion factor that is concerned with former velocity, $0 < \omega < 1$, η_1 and η_2 are constants and are called accelerating factors, normally η_1 = η_2 =2; $rand()$ are random numbers, X_{id} represents the position of particle id ; V_{id} represents the velocity of particle id ; P_{id} , P_{gd} represent separately the best position particle id has found and the position of the best particles in the whole swarm. In formula(1), the first part represents the former velocity of the particle, it enables the particle to possess expanding tendency in the searching space and thus makes the algorithm be more capable in global searching; the second part is called cognition part, it represents the process of absorbing individual experience knowledge on the part of the particle; the third part is called social part, it represents the process of learning from the experiences of other particles on the part of certain particle, and it also shows the information sharing and social cooperation among particles. The flow of PSO can briefly describe as following: First, to initialize a group of particles, e.g. to give randomly each particle an initial position X_i and an initial velocity V_i, and then to calculate its fitness value f. In every iterations, evaluated a particle's fitness value by analyzing the velocity and positions of renewed particles in formula (1) and (2). When a particle finds a better position than previously, it will mark this coordinate into vector P1, the vector difference between P1 and the present position of the particle will randomly be added to next velocity vector, so that the following renewed particles will search around this point, it's also called in formula (1) cognition component. The weight difference of the present position of the particle swarm and the best position of the swarm Pgd will also be added to velocity vector for adjusting the next population velocity. This is also called in formula (1) social component. These two adjustments will enable particles to search around two bests. The most obvious advantage of PSO is that convergence speed of the swarm is very high.

Experiment Results

Case 1: Function Optimization

In order to verify the PSO algorithm is better than genetic algorithm in the convergence speed, we using four benchmarks function to test.

F1: Schaffer function

$$\min f(x_i) = 0.5 - \frac{(\sin^2\sqrt{x_1^2 + x_2^2} - 0.5)}{[1 + 0.001(x_1^2 + x_2^2)]^2}, -100 \le x_i \le 100$$

In this function the biggest point is that the overall situation (0,0) and the global optimal value is 1.0, the largest in the overall points for the center, to 3.14 for the radius of a circle on the overall situation from numerous major points of the uplift, and, This function has a strong shock, therefore, it is difficult to find a general method of its global optimal solution.

F2: Shubert function

$$\min f(x,y) = \left\{\sum_{i=1}^{5} i\cos\left[(i+1)x + i\right]\right\} \times \left\{\sum_{i=1}^{5} i\cos\left[(i+1)y + i\right]\right\},$$
$$x, y \in [-10, 10]$$

This function has 760 local minimum and 18 global minimum, the global minimum value is -186.7309.

F3: Hansen function

$$\min f(x,y) = \sum_{i=1}^{5} i\cos((i-1)x + i)\sum_{j=1}^{5} j\cos((j+1)y + j),$$
$$x, y \in [-10, 10]$$

This function has a global minimum value -176.541793 , in the following nine point (-7.589893, -7.708314)、(-7.589893, -1.425128)、(-7.589893, 4.858057)、(-1.306708, -7.708314)、(-1.306708, -1.425128)、(-1.306708, 4.858057)、(4.976478, -7.708314)、(4.976478, -7.708314)、(4.976478, 4.858057) can get this global minimum value, the function has 760 local minimum.

F4: Camel function

$$\min f(x,y) = \left(4 - 2.1x^2 + \frac{x^4}{3}\right)x^2 + xy + \left(-4 + 4y^2\right)y^2,$$
$$x, y \in [-100, 100]$$

Camel function has 6 local minimum (1.607105, 0.568651)、(-1.607105, -0.568651)、(1.703607, -0.796084)、(-1.703607, 0.796084)、(-0.0898,0.7126) and (0.0898,-0.7126), the (-0.0898,0.7126) and (0.0898,-0.7126) are the two global minimums, the value is -1.031628. We run our algorithm and compare the results with traditional genetic algorithm. In the experiment, each case is repeated for 100 times. The convergence times mean the number of times to get the best solution out of the 100 times. Table 1 shows the statistics of our experimental results in terms of accuracy of the best solutions.

Table 1. The experiment results comparison

Function	Algorithm	Convergence times	Optimal solution
F1	GA	72	1.000000
	PSO	80	1.000000
F2	GA	75	-186.730909
	PSO	75	-186.730909
F3	GA	85	-176.541793
	PSO	90	-176.541793
F4	GA	23	-1.031628
	PSO	32	-1.031628

Case 2: One-bit full adder

Evolving the one-bit adder was easier to do on a larger geometry but resulted in a less efficient circuit. That is many genetic algorithm was able to discover 100% functional solutions was intimately related to the size of the geometry, but our algorithm use small geometry to find the fully functional solutions. The circuit design by GA is showed in Fig.1 (with five gates), Fig.2 is our algorithm's results (with three gates).

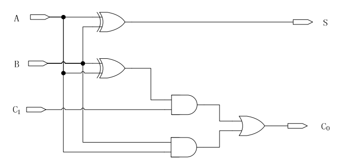

Fig.1. One-bit full adder circuit designed by GA

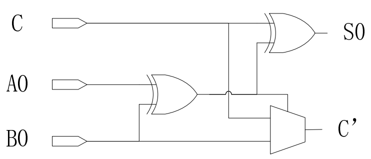

Fig.2. One-bit full adder circuit designed by PSO

Case 3: Two-bit full adder

A two-bit full adder circuit, which with a truth table with 5 inputs and 3 outputs. In this case, the PSO algorithm use small geometry to find the fully functional solutions, the matrix has a size of 3×3. The circuit designed by GA is showed in Fig.3 (with ten gates) and our resulting circuits as shown in Fig.4 (with six gates).

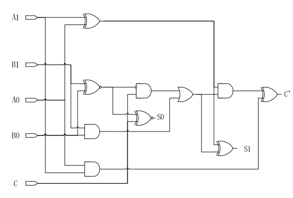

Fig.3. Two-bit full adder circuit designed by GA

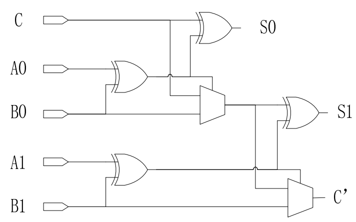

Fig.4. Two-bit full adder circuit designed by PSO

Case 4: Other Circuit

Case 4 is selected from the paper published by Coello in 2000 [9]. In this paper have 4 cases, we use PSO to evolve the results, then compared with Coello in Table.2. From the Table, we can know our results are better than Coello's, especially the case 2 and case 3.

Table 2. Results compared with Coello's algorithm

Case	Coello's results(used gates)	PSO's results(used gates)
Case 1	4	4
Case 2	7	6
Case 3	6	5
Case 4	7	7

Conclusion

A major bottleneck in the evolutionary design of electronic circuits is the problem of scale. This refers to the very fast growth of the number of gates, used in the target circuit, as the number of inputs of the evolved logic function increases. This results in a huge search space that is difficult to explore even with evolutionary techniques. Another related obstacle is the time required to calculate the fitness value of a circuit. Use the traditional genetic algorithm for electronic circuit optimization design is being trapped easily into a local optimum and the convergence speed is slow. In this paper, we use PSO algorithm to overcome the shorcomings of GA. By analyzing the testing results of four Benchmarks optimization, we reach the conclusion: in the optimization speed, the PSO algorithm is efficiency than the GA. We also use our proposed algorithm to solve the circuit optimization design, from the results shown our algorithm is efficiency.

Acknowledgments: This paper is supported by National Civil Aerospace Pre-research Project of China, supported by the Fundamental Research Founds for National University, China University of Geosciences (Wuhan) and supported by the Provincial Natural Science Foundation of Hubei (No. 2011CDB334 and 2011CDB346),.

REFERENCES

[1] Zebulum, R. S., Pacheco, M. A. and Vellasco, M. M., Evolutionary Electronics: Automatic Design of Electronic Circuits and Systems by Genetic Algorithms, CRC Press, 2001
[2] Thompson, A. and Layzell, P, Analysis of unconventional evolved electronics, Communications of the ACM(1997), Vol. 42, 71-79
[3] Louis, S.J. and Rawlins, G. J., Designer Genetic Algorithms: Genetic Algorithms in Structure Design, in Proceedings of the Fourth International Conference on Genetic Algorithms, 1991
[4] J. Kennedy and R. C.Eberhart, Particle Swarm Optimization, IEEE International Conference on Neural Networks, 1995,1942-1948
[5] Clare M, Kennedy J, The Particle Swarm - Explosion, Stability, and Convergence in a Multidimensional Complex Space, IEEE Trans. on Evolution2ary Computation, 2002, vol.6(1), 58-73
[6] C.A.Coello and M.S.Lechuga, Mopso, A proposal for multiple objective particle swarm optimization, In IEEE Proceedings World Congress on Computational Intelligence, 2002, 1051-1056
[7] J.Kennedy, The particle swarm: social adaptation of knowledge, In Proc. IEEE Conf. on evolutionary computation, 1997, 3003-3008
[8] E. Oscan and C. K.Mohan, Analysis of A Simple Particle Swarm Optimization System, Intelligence Engineering Systems Through Artificial Neural Networks, 1998, 253-258
[9] Coello C, Aguirre A, Buckles B. Evolutionary Multiobjective Design of Combinational Logic Circuits, In Proc. of the Second NASA/DOD Workshop on Evolvable Hardware, 2000, 161-172
[10] Xuesong Yan, Qinghua Wu and Hanmin Liu, Orthogonal Evolutionary Algorithm and its Application in Circuit Design, Przeglad Elektrotechniczny (Electrical Review), Issue 05b/2012, 7-10
[11] Xuesong Yan, Qinghua Wu, Chengyu Hu, Qingzhong Liang, Circuit Design Based on Particle Swarm Optimization Algorithms, Key Engineering Materials, 2011, Vol. 474-476, 1093-1098
[12] Xuesong Yan, Electronic Circuit Automatic Design Based on Genetic Algorithms, Procedia Engineering, 2011, Vol.15, 2948-2954
[13] Xuesong Yan, Qinghua Wu and Hanmin Liu; Orthogonal Evolutionary Algorithm and its Application in Circuit Design. Przeglad Elektrotechniczny(Electrical Review), 2012, Issue 05b/2012, 7-10.

***Authors**: Xuesong Yan, associate prof., School of Computer Science, China University of Geosciences, Wuhan 430074, China and now is the visiting scholar of Department of Computer Science, University of Central Arkansas, Conway 72034, USA, E-mail: yanxs1999@126.com; Chengyu Hu, associate prof., School of Computer Science, China University of Geosciences, Wuhan 430074, China, E-mail: huchengyucug@gmail.com; Hong Yao, associate prof., School of Computer Science, China University of Geosciences, Wuhan 430074, China , E-mail: yaohong@cug.edu.cn. Yuanyuan Fan, Lecturer, School of Computer Science, China University of Geosciences, Wuhan 430074, China , E-mail: yyfan@cug.edu.cn.*

Qingzhong Liang, Lecturer, School of Computer Science, China University of Geosciences, Wuhan 430074, China , E-mail: qzliang@cug.edu.cn.

Chao Liu, Lecturer, School of Computer Science, China University of Geosciences, Wuhan 430074, China , E-mail: chaoliu@cug.edu.cn.