ONMCGP: Orthogonal Neighbourhood Mutation Cartesian Genetic Programming for Evolvable Hardware

Fuchuan NI[1,2], Yuanxiang LI[3] and Peng KE[4]

[1,3,4]State Key Lab of Software Engineering, Wuhan, P.R.China

[2]Department of Computer science, Huazhong Agricultural University, Wuhan, P.R.China

E-mail: `fcni_cn@whu.edu.cn`[1], `yxli@whu.edu.cn`[3], `4278072@qq.com`[4]

Abstract. Evolvable Hardware is facing the problems of scalability and stalling effect. This paper proposed a novel Orthogonal Neighbourhood Mutation(ONM) operator in Cartesian genetic programming (CGP), to reduce the stalling effect in CGP and improve the efficiency of the algorithms.The method incorporates with Differential Evolution strategy. Demonstrated by experiments on benchmark, the proposed Orthogonal Neighbourhood Search can jump out of Local optima, reduce the stalling effect in CGP and the algorithm convergence faster.

1. Introduction

Evolvable Hardware(EHW) is a technique to dynamically alter the hardware functionality and physical connections of its circuits using methods inspired by natural evolution[1]. EHW, also briefly is defined as "$EHW = EAs + PLDs$". Among the EAs, Cartesian Genetic Programming(CGP)[2] is the most popular, important algorithm in digital evolvable circuits.

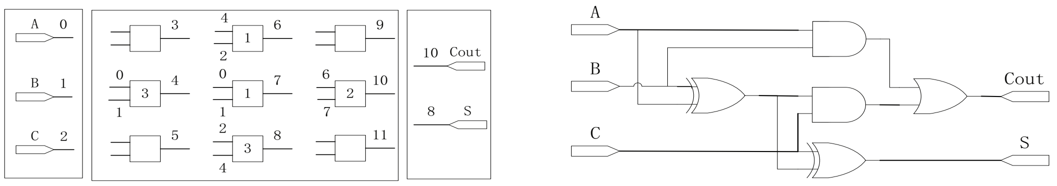
Up to date, because of the limitations in scalability, EHW seldom competes with traditional designs and few real-world applications have been developed[3]. First, The length of the chromosome representation of electronic circuits and the computational complexity grows exponentially with the increasing of inputs and the outputs; Second, for EHW, especially in combination circuits, many-for-one map between genotype and phenotype exits. And The problem of stalling effect in fitness functions is described as the nonimprovement of the fitness values during the evolutionary process. In EAs, the individuals that have the greater fitness have greater possibilities to reproduce. Thus, in the later stage in evolving process, the difference between the individuals becomes very little and "stalling effect " appears[1] [3]. E.Stomeo[4] observed that when the fitness function reaches 82.8% the stalling effect occurs.

Cartesian Genetic Programming (CGP), uses evolution strategy (ES), operating with the population of $\mu + \lambda$ individuals by point mutation. In order to enhance the search ability for the solution space, *The ideal mutation operator* is expected to have the merits of "higher probability of making far long jumps" at the early stage and "much better local fine-tuning ability" at the later stage. Yao[5] proposed a Cauchy mutation operator which jumps longer than Gaussian mutation. Lee[6] proposed levy Mutation. Zhao[7] proposed a Non-Uniform Mutation to control the search step and neighborhood size. Storn R[8]suggested a new strategy, Differential evolution. Leung[9] proposed orthogonal GA, incorporated experimental design methods into the GA. But all these methodologies are devised for function optimization. Thereby this paper proposes the Orthogonal Neighborhood Mutation(ONM) for CGP to evolve the combination circuits efficiently.

This paper is organized as follows. Section 2 considers the principal CGP in EHW. Section 3 proposes ONM operator in CGP to design combinational circuit, as well as the algorithm process. Sections 4 presents the experiment results with benchmark. Finally, section 5 presents the main conclusions.

2. Cartesian genetic programming

In Cartesian Genetic Programming (CGP), which was proposed by Miller[2], a candidate circuit is represented as an array of programmable elements with of u(columns) and v(rows), whose inputs and outputs is fixed. For each node, its' input can be connected either to the output of a node placed in the previous L columns or to some of the circuit inputs. The L parameter is defined as the level of connectivity. Feedback is not allowed. Each node is programmed to perform one of n-input functions defined in the function set. The outputs are selected randomly from the codes in the array. CGP, using evolution strategy (ES), operates with the population of $\mu+\lambda$ individuals. The initial population is randomly generated. Every new population consists of the best individual of the previous population and the offsprings are created by point mutation. The fitness function is the difference between the output and the required truth table. The goal is to minimize the difference.Figure 1(a)describes how one-bit adder representation in CGP, 3 columns and 3 rows.



(a) CGP-Code (b) CGP-DeCode

Figure 1. The CGP representation

The chromosome is arranged as : 2 0 1, $\underline{0\ 1\ 3}$, 1 0 3, $\underline{4\ 2\ 1}$, $\underline{0\ 1\ 1}$, $\underline{2\ 4\ 3}$, 2 8 1, $\underline{6\ 7\ 2}$, 3 1 0, $\underline{10,\ 8}$. The Length is $3 \times 3 \times 3 + 2 = 29$. Node[1][0], i.e. $\underline{0\ 1\ 3}$, 0,1 are the number of connect responding to the input A, B; 3, is the index of the function type, representing XOR. The codes underlined are activated, which would be decoded. Figure 1(b) shows the logic circuit when the the chromosome is decoded.

3. Orthogonal Neighborhood Mutation in CGP

As described in section 1, the CGP often traps into being Stalling. To improve the search efficiency of CGP, Orthogonal Neighborhood Mutation(ONM) is proposed and Differential Evolution strategy is considered. The operator is listed as follow:

(1)For a individual i (i =1,2,3, ..., n; n defined as size of the population), which the fitness is f_i, select other three individuals randomly;

(2)Select four loci as alleles randomly, in gene strings, to compose a Four-factor-three-level orthogonal experiment and local search is conducted;

(3)by means of range analysis, the best combination is obtained;

(4) If the fitness of the best combination(f') is the best solution, the individual i is replaced, else produce a random number R:

if $R < C$ (C, defined as a constant between 0 and 1) the individual i is replaced; else the individual i is reserved.

The algorithm of Construction of Orthogonal Array refers to [9].

In detail, for the example cited in section 2, the genotype consists of thirteen nodes; select the 2nd, 5th, 7th and 8th nodes as 4 alleles randomly from 3 chromosomes, described as $A1, A2, A3; B1, B2, B3; C1, C2, C3$ and the fitnesses are f_1, f_2 and f_3. Compose a Four-factor-three-level orthogonal experiment; the orthogonal array table list as Table1:

Table 1. Orthogonal Arrays

	A	B	C	D	fittness
(1)	A1	B1	C1	D1	f'_1 (=f_1)
(2)	A1	B2	C2	D2	f'_2
(3)	A1	B3	C3	D3	f'_3
(4)	A2	B1	C2	D3	f'_4
(5)	A2	B2	C3	D1	f'_5
(6)	A2	B3	C1	D2	f'_6
(7)	A3	B1	C3	D2	f'_7
(8)	A3	B2	C1	D3	f'_8
(9)	A3	B3	C2	D1	f'_9

Calculate the fitnesses of 9 novel individuals and analyze the range. Presume that the best combination is A', B', C', D' and the fitness is f'.

According to Differential Evolution Strategy[8], the new individual would be accept with probability R. If $f' > max\{f_i, f_1, f_2, f_3, f'_1, f'_2, ..., f'_9\}$, the individual i is substitute by the best combination;If $f' <= max\{f_i, f_1, f_2, f_3, f'_1, f'_2, ..., f'_9\}$, the individual i is substituted by the best combination with probability R.

Compared to local search conducted in each locus for every individual, which deteriorates into exhaustive search, ONM operator selects the best in the neighborhood, and could can jump out of Local optima more easily and faster.

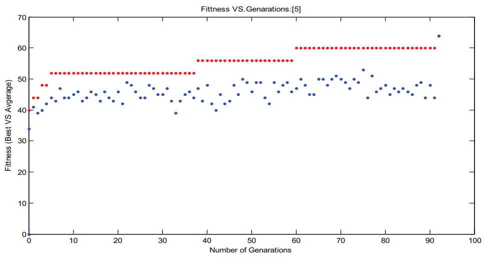
4. Experiment result

The compared experiment between traditional CGP and the ONMCGP is complimented. The benchmark is one-bit full adder circuit which has 3 inputs and 2 output. The two algorithms are same except Mutation operator. The CPU is Intel Core i5-3230 @2.6GHZ, RAM 8G based on Windows 8 64bits. The population size is 10;The number of generation is 60000; The probability of mutation is 0.6. The Code Matrix is 3 rows with 10 columns and the level-back is set to 10. The two algorithms all run 30 times.The succeed rates reach 100%. The average First Hit generation of CGP is 4543 and that of ONMCGP is 500.

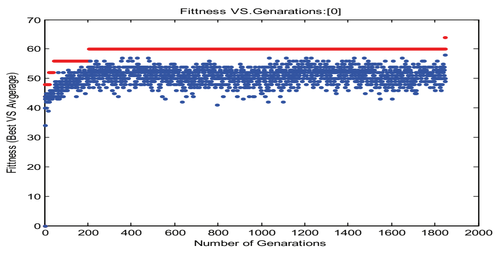
Figure 2 shows the evolving process of the ONMCGP. Figure 2(a) is the best, which converged at 93 generation; Figure 2(b) is the worst, which costs 1900 iterations to get the satisfied solution. Figure 3 shows the evolving process of the CGP. Figure 3(a) is the best, which converged at 174 generation; Figure 3(b) is the worst, which costs about 12500 iterations to get the satisfied solution.

5. Conclusion

The results of the compared experiments demonstrate that the ONM operator could reduce the stalling effect and convergence more quickly. Incorporated with *orthogonal experimental design method* , the proposed operator can scan the feasible solution space once to locate the

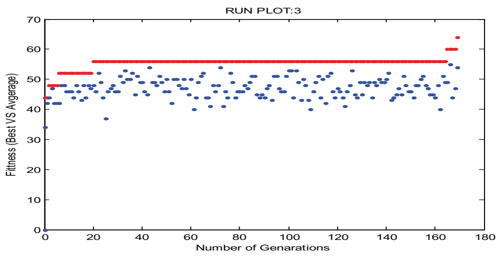


(a) ONMCGP-Best

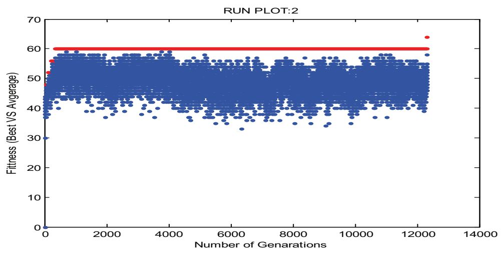


(b) ONMCGP-Worst

Figure 2. The ONMCGP Evolving Process



(a) CGP-Best



(b) CGP-Worst

Figure 3. The CGP Evolving Process

better points for further exploration in subsequent iterations. The ONM operator can reduce the stalling effect, and thus be beneficial to the Scalability of the CGP.

6. Acknowledgements

This paper is supported by Natural Science Foundation of China (NO.60442001, 61070009) and Project 2013JC007 Granted by the Fundamental Research Funds for the Central Universities of China.

7. References

[1] Yao X and Higuchi T 1999 *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on* **29** 87–97
[2] Miller J F 1999 An empirical study of the efficiency of learning boolean functions using a cartesian genetic programming approach *Proceedings of the Genetic and Evolutionary Computation Conference* vol 2 pp 1135–1142
[3] Haddow P C and Tyrrell A M 2011 *Genetic Programming and Evolvable Machines* **12** 183–215
[4] Stomeo E, Kalganova T and Lambert C 2006 *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on* **36** 1024–1043
[5] Yao X, Liu Y and Lin G 1999 *Evolutionary Computation, IEEE Transactions on* **3** 82–102
[6] Lee C Y and Yao X 2004 *Evolutionary Computation, IEEE Transactions on* **8** 1–13
[7] Zhao X, Gao X S and Hu Z C 2007 *Applied Mathematics and Computation* **192** 1–11
[8] Storn R and Price K 1997 *Journal of global optimization* **11** 341–359
[9] Leung Y W and Wang Y 2001 *Evolutionary Computation, IEEE Transactions on* **5** 41–53