A Logical Operator Based Genetic Operator

A Logical Operator Based Genetic Operator

Xiang-Yan Zeng*, Yen-Wei Chen**, Zensho Nakao**

Abstract

In this paper, a new genetic operator designed for function optimization with binary encoding is presented. This operator carries out logical *and* and *or* operation on some corresponding bits of two chromosomes and produces two new children. We proved that this operator combines the features of both crossover and mutation, and that it works in a way consistent with schema theorem. The effectiveness was demonstrated by experimental results.

Keywords: schema theorem, crossover, mutation, logical combination

1. INTRODUCTION

In genetic algorithm research field, people have developed many operators to meet the requirements of different applications. All the operators can be categorized into three classes: reproduction, crossover, and mutation. Fitness based reproduction causes the number of occurrences of each schema to increase or decrease from generation to generation at a near optimal rate, while crossover and mutation are used to introduce new children during the procedure. Among them, mutation is an asexual operator which operates on a single chromosome, and crossover is a sexual operator which recombines the genetic elements of two parent chromosomes to produce children different from their parents. Conventional crossover for binary encoding problems usually swaps some randomly chosen bits of two parents. In this paper, we will introduce another sexual operator called logical combination which instead does logical operations on the corresponding bits of two binary encoding chromosomes.

Logical combination operates on two chromosome as shown in Figure 1, where only the bits starting from the 3rd are combined, which is similar to one point crossover and can be called one point logical combination. We can also design two point and uniform logical combination.

Received on : June 26,2000
*Master student, Department of EEE, Univ. of the Ryukyus
**Department of EEE, Univ. of the Ryukyus

Here *child1* inherits bit values from *parent1* except that the bits from 3rd to 6th are the results of logical *and* operation on the corresponding bits of *parent1* and *parent2*, while *child2* is produced similarly except that *or* is used instead.

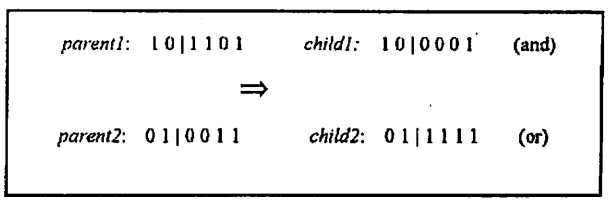

Figure 1 One point logical combination

The features of this operator can be analyzed in an intuitive way. Suppose we do logical combination to get *child1* and *child2* from *parent1* and *parent2*. In the following, we can consider only the substrings *pa1* and *pa2* which will be actually combined during the operation to get two child substrings *ch1* for *child1* and *ch2* for *child2*, and there will be three situations: (1) *ch1* and *ch2* are completely different from their parents as shown in Figure 1: the bit value of *ch1* (0001) is smaller than and that of *ch2* (1111) is larger than both parents (1101 and 0011); (2) *ch1* equals to *pa2* and *ch2* equals to *pa1*, which is exactly same as the only result of crossover; (3) *ch1* equals to *pa1* and *ch2* equals to *pa2*. In addtion, we can calculate the probability of each situation. Suppose the length of the substrings is n, then the probabilities of cases (2) and (3) are equal and are given by:

$$p = \frac{-2^n + 2\sum_{i=0}^{n} 2^i \binom{n}{n-i}}{2^{2n+1}} \qquad (1)$$

On the other hand, the probability of case (1) equals to 1-2p . From the above analysis, we can see that logical combination comprises of conventional crossover and is more general.

Next, we will theoretically prove that it is also in accordance with the Schema Theroem.

2. Analysis by the Schema Theroem

As we know from the Schema Theorem, in case that a genetic operation is used, the expected number $m(H,t)$ of occurrences of a schema H in generation t is approximately

$$m(H,t) \geq \frac{f(H,t-1)}{\overline{f(t-1)}} m(H,t-1)(1-\varepsilon) \qquad (2)$$

where, $f(H,t-1)$ is the average fitness of the observed individuals that belong to the schema, $\overline{f(t-1)}$ is the average fitness of the population at generation $t-1$, and ε is the probability of disruption due to the generic operation. When ε is small, the schema will appear at a near optimal rate in the new population.

In the case of one point crossover and one point mutation, we have

$$\varepsilon_c = p_c \frac{\delta(H)}{L-1} \qquad (3)$$

where the defining length $\delta(H)$ is the distance between the outmost defining bits, and L is the length of the schema, p_c is the crossover rate;

and

$$\varepsilon_m = p_m O(H) \qquad (4)$$

where $O(H)$ is the number of defined bits and p_m is the bit mutation rate.

From the above analysis, it has been concluded that a problem whose solution can be incrementally built up from schemata of relatively short defining length and relatively few defined positions can be handled by genetic algorithms in a near optimal way. This is part of the main idea of schema theorem[1,7].

Considering one point logical combination, from equation(1), we can get the probability of disruption of a schema H due to this operator.

$$\varepsilon_l \leq p_l \frac{X_1}{L-1}\left(1 - \frac{-2^{O(H)} + 2\sum_{i=0}^{O(H)} 2^i \binom{O(H)}{O(H)-i}}{2^{2O(H)}}\right) + p_l \frac{1}{L-1}\sum_{j=1}^{\delta(H)}\left(1 - \frac{-2^j + 2\sum_{i=0}^{j} 2^i \binom{j}{j-i}}{2^{2j+1}}\right) \qquad (5)$$

where p_l is the logical combination operation rate, X_1 is the number of don't care symbols before the first defining symbol. Eventhough this is not an accurate expression of ε_l, it is obvious that when X_1, $\delta(H)$ and $O(H)$ are small, ε_l is small. This probability contains the disruption features of both one point crossover and mutation.

3. Experimental Results

In the experiment, we adopted a test suite which consists of six functions $f_1 \sim f_6$:

$f_1(x_i) = \sum_{i=1}^{3} x_i$ $\quad -5.12 \leq x_i < 5.12$

$f_2(x_i) = 100(x_1^2 - x_2)^2 + (1-x_1)^2$ $\quad -2.048 \leq x_i < 2.048$

$f_3(x_i) = \sum_{i=1}^{5} integer(x_i)$ $\quad -5.12 \leq x_i < 5.12$

$$\frac{1}{f_4(x_i)} = 0.002 + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2}(x_i - a_{ij})^6}$$

$-65.536 \leq x_i < 65.536$, $[a_{ij}] = \begin{bmatrix} -32 & -16 & 0 & 16 & 32 & -32 & -16 & \cdots & 32 \\ -32 & -32 & -32 & -32 & -32 & -16 & -16 & \cdots & 32 \end{bmatrix}$

$$f_5(x,y) = 0.5 - \frac{(\sin\sqrt{x^2+y^2})^2 - 0.5}{(1.0 + 0.001(x^2+y^2))^2} \qquad -100 \leq x,y < 100$$

$f_6(x_i) = 20 + \sum_{i=1}^{20}(x_i^2 - \cos(2\pi x_i))$ $\quad -5.12 \leq x_i < 5.12$

The goal is to find the values of the arguments to produce the maximum or minimum value for these functions, that is, to optimize the functions.

Encoding method: A chromosome consists of 22*n bits, of which each 22 bits is interpreted into a real number argument; where n is the number of the arguments of a function.

Fitness measurement: the function values are simply used when the object is to maximize the function,

otherwise, the fitness is calculated by :

$$\frac{1}{1+f-m}$$

where f is the function value, and m is the minimum value of the function .

Selection: r elite chromosomes are copied to the next generation, and then n-r parents are selected by roulette wheel selection for genetic operation, where n is the population size and r is decided by a genetic operation rate[7].

Mutation: bit mutation is performed after crossover or logical combination.

Crossover: it has been found in our experiments that two point crossover is superior to one point and uniform crossover, so two point crossover is used to the substrings corresponding to each argument.

Logical combination: two point logical combination is used to the substrings corresponding to each argument.

The population size is 50, and the number of generation is 100 for f_1, and f_3, 400 for f_2, f_4, and f_5, 1000 for f_6. The mutation rate of each bit of all the chromosomes in one generation is 0.008 for f_1, f_2, f_3, f_4, f_5, and 0.005 for f_6. Here we consider the influence of the length of chromosomes. Logical combination and crossover are used separately but not at the same time. For each function, 40 trials were made and the same random seed set was used for the two operators. We evaluated the results by the following formula:

$$diff = \frac{O_g - R}{O_g} * 100\%$$

where O_g is the global optimal fitness, R is the result. When the algorithm gets the global optimum successfully, *diff* equals to 0

The comparison is shown in Figure 2, in which the numbers of the trials (N-O-T) within specified *diff* range are provided for two operators: there are five kinds of *diff* range: 0, (0,0.05], (0.05,0.5], (0.5,2], (2,5] and each range is represented by its maximum.

4. CONCLUSION

In this paper, we introduced a new genetic operator for binary encoding genetic algorithms. First, we analyzed this operator theoretically and got the conclusion that it was more general than crossover and also in accordance with the Schema Theorem. Then the operator was experimentally compared with crossover by a test function suite. During the experiments, it was found that the difference between the two operators was not obvious when the search space was relatively small and simple such as the cases of f_1, and f_3, and logical combination was especially effective in such cases as f_5, and f_6. In case of f_5, there are many local optima to be easily got stuck in, and in case of f_6, the search space is very large. Even though logical combination operator performs better than conventional crossover in the experimental environments, we can not yet say that it is superior to crossover in all problems, nor is it certainly superior to all other genetic operators. Genetic algorithms, however, need as many as possible operators to meet the requirements of different real world application problems. The future work is to do theoretical analysis and do more experiments to find out for which types of problems this operator is best suited.

References

[1]Lawerance Davis, *Handbook of Genetic Algorithms*, Van Nostrand , Reinhold, New York, 1991.

[2]J. R. Koza, *Genetic Programing*, The MIT Press, London, 1993.

[3]Shigeyoshi Tsutsui and Lakhmi C. Jain, "On the Effect of Multi-parents Recombination in Binary Coded Genetic Algorithms", *Proceedings of 1998 Second International Conference on Knowledge_based Intelligent Electronic Systems*, pages 155-160, 1998.

[4]J. Craig Potts, Terri D. Giddens, and Surya B.Yadav, "The Development and Evaluation of an Improved Genetic Algorithm Based on Migration and Artificial Selection ", *IEEE Trans. on Sys., Man, and Cybern*, Vol. 24, pp.73-86, Jan. 1994.

[5]*Kenneth Alan De Jong*, " Analysis of the Behavior of a Class of

Genetic Adaptive Systems ", *Technical Report No.185, Department of Computer and Communication Sciences, University of Michigan,1975.*

[6] Masafumi Hagiwara, "Pseudo-Hill Climbing Genetic Algorithm (PHGA) for Function Optimization," *Proceedings of 1993 International Joint Coference on Neural Networks*, Vol.1, pp.713-716, 1993.

[7]H. Muhlenbein, M. Schomisch, J. Born, " The Parallel Genetic Algorithm as Function Optimizer", *Proceedings of the fourth International Conference on Genetic Algorithms*, pp.271-278, 1991.

[8] Zbigniew Michalewicz, *Genetic Algorithms + Data Structure = Evolution Programs*, Springer-Verlag, Berlin, 1994.

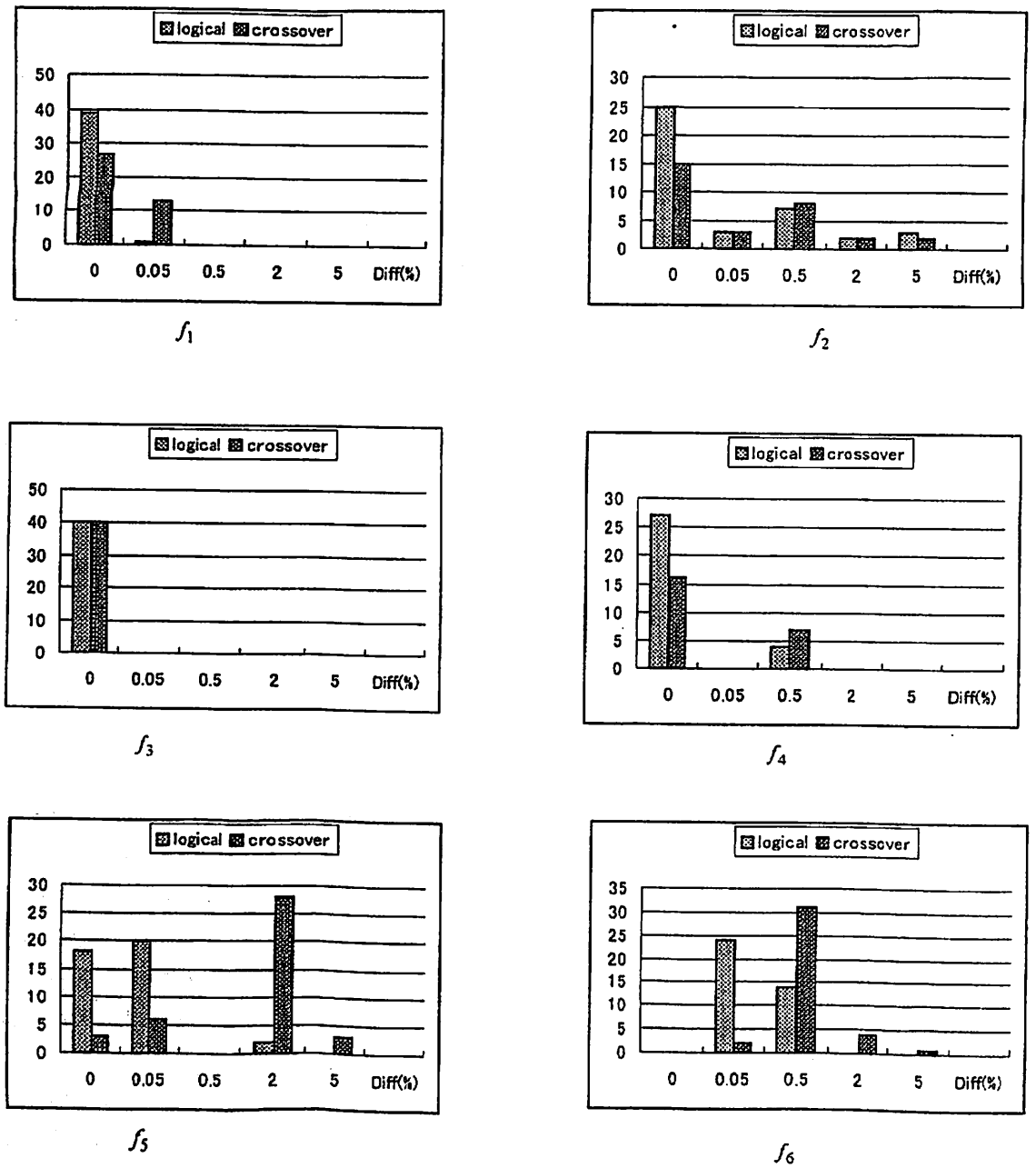

Figure 2 Experimental results fo test functions