Deterministic Crowding, Recombination And Self-Similarity

Bo Yuan
School of Information Technology and Electrical Engineering
The University of Queensland
Brisbane, Queensland 4072
Australia
E-mail: s4002283@student.uq.edu.au

Abstract – This paper proposes a new crossover operation named *asymmetric two-point crossover* (ATC). We show how deterministic crowding can be successful in the HIFF problem and the M7 function with this new crossover. We also point out that self-similarity in the solution plays an important role in the success of ATC.

I. INTRODUCTION

The genetic algorithm (GA) is a probabilistic optimization method widely used for solving various complicated search problems [1][3].

Since GAs always maintain a population with finite size, in some cases, GAs can magnify a small sampling error, causing the problem of *premature convergence* [2][3]. Furthermore, although GAs with strong convergence can find the fittest solution quickly, it is not always appropriate in that it may cause GAs to converge to one region of the search space, even when other regions also contain some good solutions. In order to provide GAs with good ability in both global optimization tasks and multimodal function optimization tasks, the maintenance of diversity has been a crucial issue.

One of the early studies that considered the maintenance of population diversity was by De Jong in 1975 (cited in [4]). He proposed a technique named "crowding factor model". However, because of the stochastic errors in the replacement of population members, this kind of crowding is not very successful at preserving population diversity. Mahfoud presented an improved crowding scheme called deterministic crowding, which virtually eliminates replacement errors [4].

Deterministic crowding (DC) was shown to be effective at multimodal function optimization tasks [4][7]. It was also used to solve the HIFF problem in which there is a need to build complex modules while maintaining diversity in the population [5][6][8]. But DC also has some limitations. Because it tends to maintain individuals on each fitness peak, if there are a large number of local optima in the fitness landscape, DC may also get stuck on them, especially when the population size is relatively small. Since DC without mutation can be viewed as a kind of parallel, crossover hill-climbing algorithm, it turns out that crossover largely determines the performance of DC [7]. So, a question would be: What kind of crossover is good for DC?

Watson and Pollack [8] compared the performance of deterministic crowding using one-point, uniform and disrespectful crossover operations on HIFF. From their results, one-point crossover was shown to be the best.

In one-point crossover, the end biases would make the formation of certain global optima more difficult than others and two-point crossover does not possess any end biases. As a result, using two-point crossover, the distribution among global optima is much closer to uniform [7].

One of the problems of standard one-point and two-point crossover operations is that they cannot create new genes in positions of individual. If genes in some bits become fixed due to the loss of genetic diversity, there is no force to regain those genes (i.e., genes lost will be lost forever). On the other hand, since the standard one or two-point crossover operation only exchanges genes on the same positions between two parents, if the two parents are very similar or even identical, this kind of recombination will be of little value. In other words, if the two parents selected are both from the same region around a local optimum, it is often the case that their offspring will be in the same region, performing local hill climbing. This character can make GAs relatively stable but will also waste search efficiency if those individuals are located in a poor area.

In order to maintain the diversity in a population, one can employ a large population but this would make the evolution very time-consuming. Another way is to use mutation. Due to its randomness, mutation is like a two-edged sword: It can add variation at the cost of destroying good building blocks.

Since the influence of the bias of initial population and other random factors cannot be easily diminished, it is reasonable to consider a question: Can we design a crossover operation that can force DC to explore other regions of a search space even when most of the individuals are located at the same region? In this paper, we propose a non-standard two-point crossover, *asymmetric two-point crossover* (ATC). Actually, it is an extension of standard two-point crossover. The only difference is, in ATC, the crossover points may be different in two parents. That is to say, this kind of crossover can produce offspring through crossover of genes on different parts of the two parents.

The remaining sections of this paper are organized as follows: The next section introduces DC, the two main test functions used in this paper and the asymmetric two-point crossover algorithm. Section three presents some simulations and results of DC on HIFF and M7 using ATC compared with standard two-point crossover. Section four contains in-depth analysis and additional simulations. Section five introduces a variation of ATC (VATC) for HIFF. Section six is the conclusion of simulation results and the relationship between DC, ATC and self-similarity.

0-7803-7282-4/02/$10.00 ©2002 IEEE

II. METHODOLOGY

A. Asymmetric Two-Point Crossover (ATC)

ATC is defined as below:

1. Select two crossover points p_1 and p_2 in parent1.
2. Select a crossover point p_3 in parent2.
3. Create offspring1: Replace genes between p_1 and p_2 in parent1 with those in parent2 starting from p_3.
4. Create offspring2: Replace genes between p_1 and p_2 in parent2 with those between p_1 and p_2 in parent1.

For example:

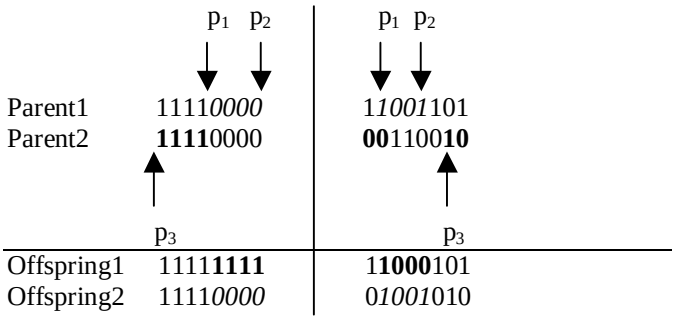

Figure 1: Two examples of ATC

In Figure 1, bold bits in offspring1 are from parent2 and italic bits in offspring2 are from parent1. We can see that offspring2 is the same as that obtained by using standard two-point crossover but offspring1 is different in that it can be created by replacing genes in parent1 with genes in asymmetric part of parent2. It is obviously that ATC can produce many more kinds of offspring than can standard two-point crossover. For example, in the first example, two identical parents can produce one different offspring, which is impossible in standard two-point crossover. Furthermore, each individual is treated as a circle of genes and if the gene sequence reaches the end of individual (e.g., the second example), it will return back to the beginning.

B. Deterministic Crowding (DC)

Deterministic crowding works on the principle of restricted competition. Any individual can be recombined with another but each offspring is only able to replace the parent that it most resembles. By doing so, individuals tend to compete with other individuals that are around the same fitness peaks, performing local hill climbing. This character is useful in preventing premature convergence in that individuals are allocated around different fitness peaks, exploring different search space in parallel. Another advantage is that the mechanism of elitism is inherently built in. As a contrast, due to the bias of initial population and some random factors during evolution, GAs without good genetic diversity maintenance approaches often quickly converge to a local optimum or cannot find and maintain all global optima during evolution.

GA with standard DC works as below:

1. Initialize population randomly.
2. Randomly group individuals in pairs.
3. After recombination, two parents produce two offspring.
4. According to pairing rule, each offspring is paired with one parent.
5. If the offspring's fitness is higher than that of its corresponding parent, the parent gets replaced.
6. Go to step 2 until stop criteria are satisfied.

The distance measurement is the genotypic Hamming distance. In standard two-point crossover, if one offspring is most similar to one parent then the other offspring must be most similar to a different parent. But in ATC, the two offspring can be both similar to one parent. So, in this paper, we changed the definition of DC as below:

*Each offspring is to be paired with the parent to which it is most similar. If the two parents are identical, each offspring is paired with one parent. If the two offspring are both most similar to one parent and their fitness are **all** higher than that of the parent, the offspring that is more similar to the parent wins (i.e., we want to keep the stability of GA).*

C. Test Functions

The two main test functions used in this paper are the HIFF problem [5][6][8] and the M7 function [7].

1) The HIFF problem:

$$f(B)=\begin{cases} 1, & \text{if } |B|=1,\text{else} \\ |B|+f(B_L)+f(B_R), & \text{if}(\forall i\{b_i=0\}\text{or}\forall i\{b_i=1\}) \\ f(B_L)+f(B_R), & \text{otherwise} \end{cases} \quad (1)$$

In equation 1, B is a block of bits, $\{b_1, b_2, \ldots b_n\}$, |B| is the length of the block (n), b_i is the ith element in block. B_L and B_R are the left and right sub-strings respectively. The fitness of a binary string is defined by the recursive function given above. This function regards the string as a binary tree and recursively decomposes it into left and right sub-strings. Each sub-string is a building block in a certain level and if all the bits in the sub-string are of the same value (i.e., all ones or all zeros), it is rewarded as the best block in its level. The fitness of the whole string is the sum of these fitness contributions of all blocks at all level. In the HIFF problem, two different best building blocks at one level can create a second best block at the next level when they are combined together. For example, the local optimum "11110000" can be viewed as the combination of "1111" and "0000", which are two global optima in lower level and this local optimum is maximally distant from two global optima (i.e., "11111111" and "00000000"). This character of the HIFF problem makes it difficult for GAs without good diversity maintenance approaches.

0-7803-7282-4/02/$10.00 ©2002 IEEE

2) The M7 function:

$$f(x_0,\ldots,x_{29}) = \sum_{i=0}^{4} u\left(\sum_{j=0}^{5} x_{6i+j}\right) \qquad \forall k,\ x_k \in \{0,1\} \tag{2}$$

Function $u(x)$ is specifically built to be deceptive [7]. In our simulations, the values were set to be: 1.0, 0.0, 0.32, 0.64, 0.32, 0.0,1.0 for the integer x from 0 to 6 (see Figure 2). It has two optima of value 1 when x=0 and x=6 as well as a local optimum at x=3. As a result, M7 has 32 global optima with value equal to 5 and millions of local optima.

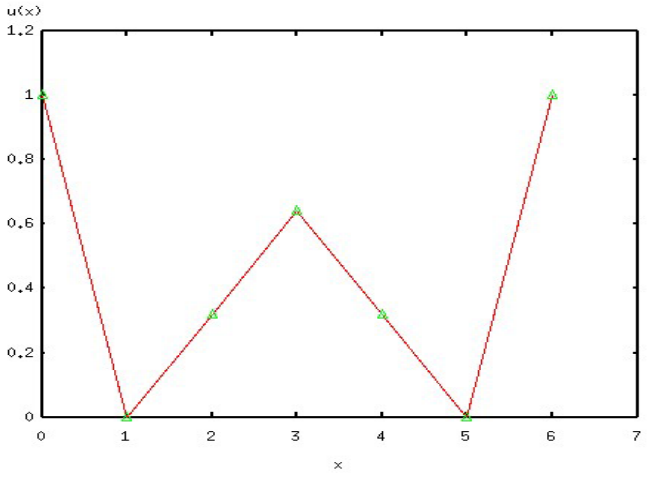

Figure 2: The graph of function $u(x)$. It has two global optima and one local optimum and makes M7 a deceptive multimodal function.

III. SIMULATIONS AND RESULTS

In this section, we applied DC to HIFF and M7 with different recombination approaches: standard two-point crossover and ATC. The mutation rate was set to zero.

A. The HIFF problem

We first conducted some simulations on the 64-bit HIFF problem. All results were averaged over 100 runs (300 generations per run). In Figures 3 and 4, the three categories (from left to right) are the results with population size equal to 100, 300, and 1000 respectively. In Figure 3, the four columns in each category represent (from left to right) the number of runs that find at least one global optimum using ATC (ATC1), the number of runs that find at least one global optimum using standard two-point crossover (TP1), the number of runs that find both global optima using ATC (ATC2) and the number of runs that find both global optima using standard two-point crossover (TP2). In Figure 4, the two columns in each category represent the total number of individuals at global optima at the end of evolution (from left to right: ATC, standard two-point crossover).

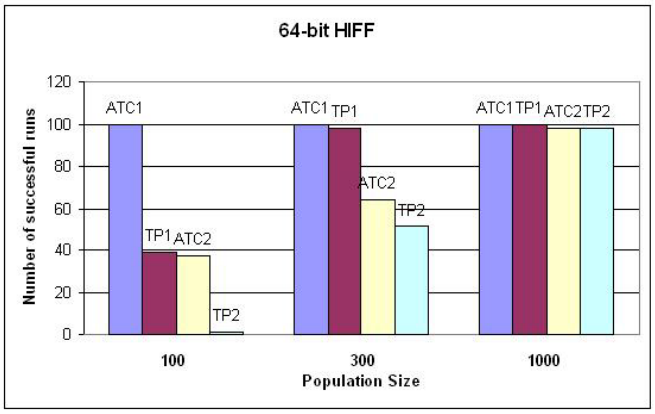

Figure 3: Comparison of the number of successful runs (ATC vs. Standard two-point crossover)

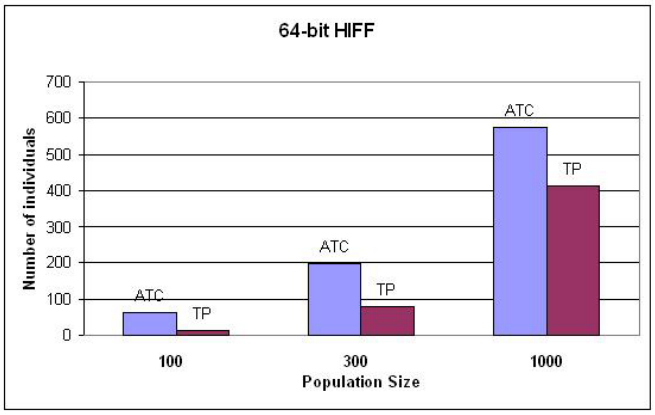

Figure 4: Comparison of the number of individuals at global optima at the end of evolution (TP: Standard two-point crossover)

With a small population, DC with standard two-point crossover often gets stuck in local optima while DC with ATC still has good global optimization ability. With the help of ATC, DC is also more likely to find the two global optima and can maintain more individuals on them meaning genetic diversity maintained by ATC is better than that by standard two-point crossover. Meanwhile, there is a tendency: The bigger the population, the less the difference.

In order to fully investigate ATC's potential, we also conducted some simulations on the 256-bit HIFF problem, each for 30 runs. For this difficult problem, both crossover operations could only find at most one global optimum in one run, even with a large population (e.g., 1000). DC with standard two-point crossover often failed to find the global optimum (only 4 runs out of 30 runs found a global optimum). As a contrast, DC with ATC was still successful with the same population size (i.e., in 30 runs, all runs found one of the global optima). Furthermore, DC with ATC was also capable of solving the 512-bit HIFF problem (i.e., in 30 runs, all runs found one of the global optima), which is very hard for GAs with other standard diversity maintenance approaches.

B .The M7 function

We conducted several simulations on M7 with different population sizes. Results were averaged over 100 runs (500 generations per run). Although both crossover operations could maintain almost all individuals on global optima at the end of evolution, there are still some differences in the distribution of individuals and the number of global optima maintained. To compare the distribution, we calculated the variance of the data set composed of the number of individuals on each optimum. In Figures 5 and 6, the two categories (from left to right) are the results with population size equal to 300 and 800 respectively. In each category, the left column represents the result using ATC and the right column represents that using standard two-point crossover.

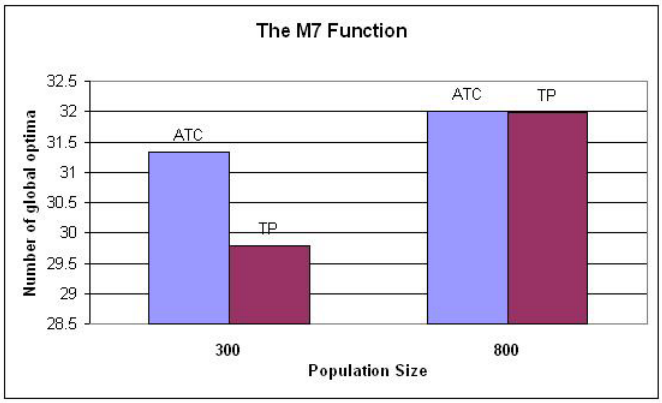

Figure 5: Comparison of the number of global optima maintained

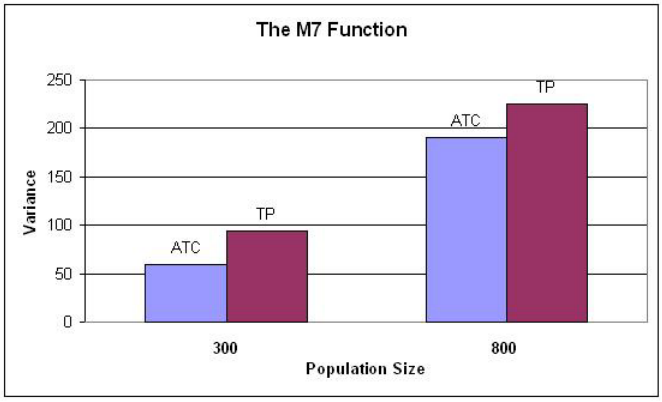

Figure 6: Comparison of distribution of individuals

It is obviously that ATC can find and maintain more global optima during evolution and the distribution of individuals among global optima is closer to uniform. Furthermore, the difference is more distinct when the population size is small than when the population size is big. That is, although it is not a difficult task to find the global optima in M7, ATC still outperforms standard two-point crossover by maintaining better genetic diversity and thus prevents individuals from converging to only some of the optima due to random factors.

IV. ANALYSIS AND DISCUSSION

In our simulations, we found that DC with ATC outperformed DC with standard two-point crossover in HIFF and M7. We also found that when the scale of problem was relatively small (e.g., 64-bit HIFF) and the population size was large (e.g., 1000 in HIFF and 800 in M7), the performance difference was less distinct than that in other situations. The reason is that by using a large population, DC with standard two-point crossover is also capable of maintaining enough genetic diversity, especially for small scale HIFF and M7. So, from Figure 3 we can see that, with population size equal to 1000, there is no distinct difference between ATC and standard two-point crossover. Similarly, in the M7 function, ATC can also find and maintain more global optima than can standard two-point crossover when the population size is small.

Now, it is reasonable to claim that ATC can maintain better genetic diversity during evolution. But why does it work? Given two specific crossover points, standard two-point crossover is only capable of producing two offspring while ATC is, in principle, capable of producing N+1 different offspring (N is the length of individual) because we can have N different crossover points in another parent. This advantage is especially important when many individuals are clustered in a region that is far from global optima. By using ATC, the two offspring produced may be quite different from their parents meaning that these offspring may be located far from the bad area in which their parents reside.

In fact, like standard two-point crossover, ATC also tries to utilize the existing building blocks and create higher order building blocks through recombination. The difference is that ATC can use building blocks in any part of the individuals and in fact the solutions of many problems have some kind of self-similarity (i.e., a good building block in one part of the solution may also be suitable when put in another part). In other words, genes can be shared more efficiently in ATC.

In order to thoroughly investigate the influence of self-similarity on ATC, we conducted additional simulations. The two test functions were variations of the M7 function. In the first test function F1, the values of u (x) were defined to be 0.8, 0.0, 0.32, 0.64, 0.32, 0.0,1.0 for integer x from 0 to 6 respectively. So, F1 is a single-optimum function with all ones as its global optimum. Obviously, the solution of this function has good self-similarity. The second test function F2 was defined similarly except that there were two internal functions $u1$ (x) and $u2$ (x). The first function $u1$ (x) was the same as u (x) but $u2$ (x) was the opposite of u (x) (i.e., the values of $u2$ (x) were set to be 1.0, 0.0, 0.32, 0.64, 0.32, 0.0, 0.8 for integer x from 0 to 6 respectively). In F2, $u1$ (x) and $u2$ (x) alternated (i.e., apply $u1$ (x) on the first six bits and then $u2$ (x) on the next six bits etc.). The reason for doing so is to decrease the level of self-similarity of the solution. Now the optimum of F2 is six ones followed by six zeros followed by another six ones and so on.

From our analysis made before, we can make several conjectures as below:

1. When the population size is small ATC will outperform standard two-point crossover.
2. When the population size is big enough, both crossover operations will work.
3. The performance of ATC will change dramatically on two test functions because they have different levels of self-similarity while standard two-point crossover will remain at the same level.

The performance criterion was the number of generations needed to find the global optimum. All results were averaged over 200 runs (maximum 1000 generations per run).

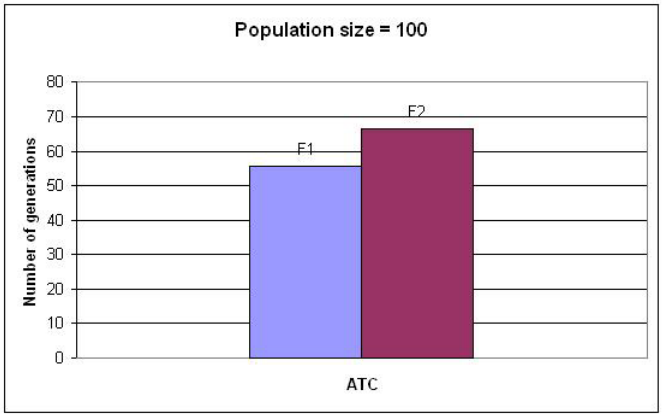

Figure 7: Comparison of the number of generations needed to find global optimum (population size=100)

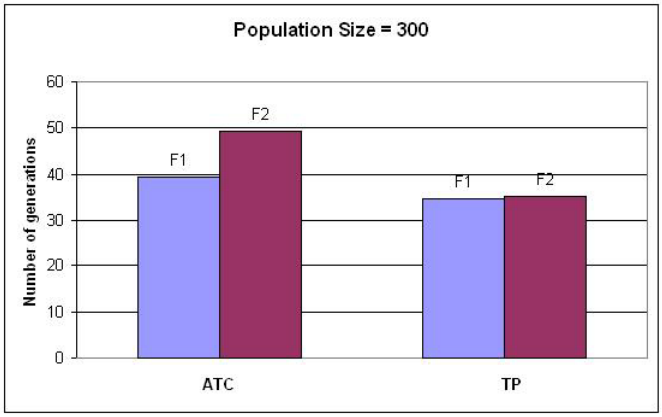

Figure 8: Comparison of the number of generations needed to find global optimum (population size=300)

In Figure 7, the left column is the performance of ATC on F1 and the right column is the performance on F2 (standard two-point crossover couldn't always find the global optimum within 1000 generations). Figure 7 shows that even with a small population, ATC is still capable of maintaining enough genetic diversity and finding the global optimum in each run while standard two-point crossover may fail in some runs. At the same time, ATC's performance changed dramatically in F1 and F2 showing that F2 is more difficult for ATC than F1. In Figure 8, the left category is ATC and the right one is standard two-point crossover. In each category, the left column is the result on F1 and the right one is the result on F2. With a big population (e.g., 300), standard two-point crossover can find the global optimum successfully. Also, it is obvious that the performance of standard two-point crossover remained at the same level on two test functions while ATC still changed greatly.

The reason is that ATC maintains genetic diversity at the cost of adding many variations in the population. If the population size is big enough, standard two-point crossover is also capable of maintaining good diversity. In this situation, the variation caused by ATC may be not necessary or even harmful. In other words, since ATC always tries to find building blocks on different parts of the individuals, if the function (e.g., F2) has poor self-similarity, this kind of effort will be of little value and waste a lot of search efficiency.

V. EXTENSION

A. A Variation of ATC: VATC

In the previous section, we demonstrated that the level of self-similarity has a great influence on the performance of ATC. Since HIFF has perfect self-similarity compared with M7, it is straightforward to consider a question: Can we further improve ATC's performance on HIFF?

For example, in the 8-bit HIFF problem, ATC can produce one global optimum through the recombination of two identical local optima "00001111" and "00001111". So, it is reasonable to think whether we can modify ATC so that it can produce both two global optima at one time. Actually, this can be realized by simply exchanging genes between the two crossover points in parent1 with genes starting from the third crossover point in parent2 (see Figure 9).

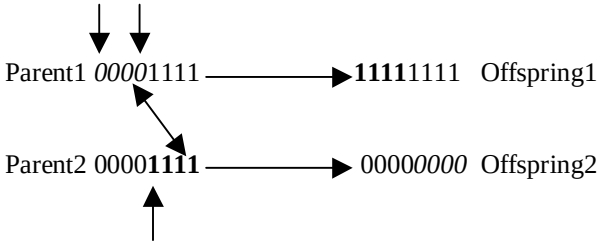

Figure 9: An example of VATC

B. Simulations on 64-bit HIFF

Tables 1 and 2 are comparisons between ATC and VATC on a 64-bit HIFF problem. Results were averaged over 100 runs (300 generations per run).

TABLE 1
COMPARISON OF ATC AND VATC (POPULATION SIZE=100)

Name	A	B	C	D
ATC	100	37	63	54
VATC	100	28	86	39

TABLE 2
COMPARISON OF ATC AND VATC (POPULATION SIZE=300)

Name	A	B	C	D
ATC	100	64	196	48
VATC	100	64	256	37

Notes:

A: The number of runs finding at least one global optimum.
B: The number of runs finding both global optima.
C: The number of individuals maintained on global optima.
D: The number of generations needed to first find global optimum.

VATC can find global optima more quickly and maintain more individuals on global optima. But when the population size is small, VATC is less likely to find both global optima. That is, VATC achieves a strong convergence at the cost of losing some multimodal optimization ability. Furthermore, VATC was still successful in the 2048-bit HIFF problem meaning it is much less likely to get stuck than ATC due to the better genetic diversity maintained.

VI. CONCLUSION

The major motivation of this paper was to investigate DC's performance under different situations, especially when there are a large number of local optima in the fitness landscape. What we tried to make clear is the kind of crossover operation that is good for DC.

We first proposed a new crossover operation named ATC and then demonstrated DC's ability of successfully solving the HIFF problem and the M7 function with ATC. The major character of ATC is that it can thoroughly exploit the building blocks in the current population and even local competition of individuals in a bad area can still produce some good individuals located far from their parents. By doing so, ATC could be expected to be effective in genetic diversity maintenance.

Through the comparison with the results obtained by using standard two-point crossover, we found that ATC outperformed standard two-point crossover in both test functions. Because DC with standard two-point crossover tends to maintain each fitness peak, when the population size is small or the number of local optima is large, it is also possible to get stuck. In our simulations, DC with standard two-point crossover could only be successful in the 64-bit HIFF problem. By contrast, DC with ATC was still successful in the 512-bit HIFF problem. In the M7 function, ATC could maintain more fitness peaks and produce a more uniform distribution of individuals among global optima.

It is interesting to find that crossover operations in GAs do not necessarily need to be symmetric. Through in-depth analysis and further simulations, we pointed out that the reason lies in the self-similarity of solutions and ATC is successful in problems having good self-similarity in their solutions. Furthermore, we proposed a variation of ATC, VATC, which has much better global optimization ability.

Now, it is time to draw the conclusion. First, although DC is a good diversity maintenance approach, sometimes it may also make some mistakes in identifying search space, getting stuck in local optima. Fortunately, some specifically designed crossover operations such as ATC and VATC may be helpful. Second, self-similarity in solutions should be taken into account sufficiently. Goldberg [3] claimed that important similarities among good individuals can help guide a search but we found good self-similarity can also greatly improve search efficiency. Certainly, it is not always the case but if we can benefit from it, why not?

ACKNOWLEDGEMENTS

Thanks to Janet Wiles for the motivating discussion of this work and some suggestions in improving it. The author is also grateful to Paul J. Darwen for his kind help.

REFERENCES

[1] Melanie Mitchell, *An Introduction to Genetic Algorithms*, MIT Press, 1996.

[2] Stephanie Forrest and Melanie Mitchell, "What Makes a Problem Hard for a Genetic Algorithm? Some Anomalous Results and Their Explanation", *Machine Learning* 13, pp. 285-319, 1993.

[3] David E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Reading, Mass., 1989.

[4] Samir W. Mahfoud, "Crowding and Preselection Revisited", In *Parallel Problem Solving from Nature*, 2, North-Holland, pp. 27-36, 1992.

[5] Richard A. Watson and Jordan B. Pollack, "Hierarchically-Consistent Test Problems for Genetic Algorithms", In *Proceedings of 1999 CEC*, Angeline, et al. eds. IEEE Press, pp.1406-1413, 1999.

[6] Richard A. Watson, Gregory S. Hornby and Jordan B. Pollack, "Modeling Building-Block Interdependency", In *Proceedings of the 5th Conference on Parallel Problem Solving from Nature* (pp. 97-106). Springer-Verlag, LNCS 1498, 1998.

[7] Samir W. Mahfoud, "Niching Methods for Genetic Algorithms", Doctoral dissertation, University of Illinois at Urbana-Champaign, 1995.

[8] Richard A. Watson and Jordan B. Pollack, "Recombination Without Respect: Schema Combination and Disruption in Genetic Algorithm Crossover", In *Proceedings of GECCO 2000*, Whitley, D, et al (eds.), Morgan Kaufmann, 2000.