Exploiting Synergies of Multiple Crossovers: Initial Studies *

Inki Hong, Andrew B. Kahng and Byung Ro Moon
UCLA Computer Science Department
Los Angeles, CA 90095-1596

Abstract

Genetic algorithms (GAs) are believed to exploit the synergy between different traversals of the solution space that are afforded by crossover and mutation operators. While dozens of different crossovers are known, comparatively little attention has been devoted to improving performance by using multiple crossover operators within a given GA implementation. Here, we examine various aspects of combining different crossovers; we demonstrate that mixtures of crossovers can outperform any single crossover, and that choosing appropriate mixing proportions is critical for good performance. We conjecture that good crossover mixtures are characterized by "balance" in the crossovers' respective influences in the population, and explore three adaptive strategies for mixing crossovers.

1 Introduction

The crossover and mutation operators of genetic algorithms (GAs) navigate the solution space in synergetic ways: crossover generates new solutions by combining traits of already-visited solutions, while mutation generates new solutions by random perturbations. The utility of crossover has been extensively discussed (e.g., [13, 18, 6, 7]), and there are dozens of different crossovers in the literature. Results showing that the various crossovers differ in their traversals of the solution space (e.g., [4, 5]) together hint that using only one crossover – i.e., only one style of traversing the solution space – may not be optimal. However, surprisingly few works use more than one crossover operator in a GA implementation.

*This work was partially supported by NSF YI award MIP-9257982 and grant MIP-9223740. The authors are with the UCLA VLSI CAD Laboratory and the UCLA Commotion Laboratory (NSF CDA-9303148).

Two works are notable for combining more than one crossover operator [14, 17]. Schaffer and Morishima [14] proposed an adaptive crossover scheme in which the number of crossover points varies according to punctuation marks on the chromosomes; it was reported that the punctuated crossover outperformed the traditional (single-point) crossover, but whether this performance improvement resulted from adaptiveness of the crossover was not clear. Spears [17] has suggested an adaptive crossover in which two-point crossover and uniform crossover are used with adaptively varying rates, based on a tag bit for this purpose. For the N-Peak problems [3], the performance of Spears' adaptive crossover is always intermediate between that of the two underlying crossovers (i.e., the better of the two underlying crossovers outperformed the adaptive crossover) [17].

In this paper, we study interactions among multiple crossovers within a single GA implementation. Specifically, we examine (i) whether using mixtures of different crossovers can improve performance over using only one crossover, (ii) how different mixing proportions of crossovers can affect the performance, (iii) how the influences of different crossovers in a mixture will change within the population over time, and (iv) the characteristics of good crossover mixtures. Within a testbed consisting of three basic crossover types and a hybrid GA driver for VLSI cell placement, we demonstrate that crossover mixtures exhibit what appear to be unimodal relationships between the proportion of a given crossover in the mixture and the GA performance. Other experiments suggest that balancing the crossovers' respective influences in the makeup of the population is an important factor for good performance. With this in mind, we examine three adaptive strategies for mixing distinct crossovers: the first strategy gives higher chance to the crossover which contributed more to the current population, the second strategy offers a direct contrast with the first strategy, while the third strategy tries to adaptively balance the crossovers' cumulative influences on the current population. The second and the third strategies significantly outperform the first strategy.

2 An Experimental Protocol

We denote a mixture, or *combination*, of k different crossovers as $C_1X_1 + C_2X_2 + \cdots + C_kX_k$, where X_i is the i^{th} crossover and C_i is the probability of applying crossover X_i ($\sum_{i=1}^{k} C_i = 1$).[1] Intuitively, we say the crossovers in a combination $\{X_1, X_2, \ldots, X_k\}$ are synergetic, or "orthogonal", if they allow the GA search to visit better-quality regions in the solution space which are not as easily reached by applying any single crossover by itself. The coefficients of combination $\{C_1, \ldots, C_k\}$ can either be fixed or vary adaptively. To study the effect of crossover combination on solution quality, we use the following testbed.

We study instances with $k = 2$ or 3, with various combinations of three underlying crossover operators that are expected to afford very different search mechanisms: traditional crossover[2], cycle crossover [12], and two-dimensional geographic crossover [10].[3] While this is just one of many possible crossover combinations, note that promising qualities of each crossover have been shown in the literature. In cycle crossover, the gene at each position of an offspring is guaranteed to be from one of the two parents; cycle crossover is thus believed to have relatively low disruptivity. In experiments on the circuit placement problem [15], cycle crossover performed better than other ordering crossovers such as PMX [8] and order crossover [16]. Geographic crossover [10] is a multi-dimensional crossover that effectively uses genes' geographical linkages. In [10], geographic crossover outperformed both traditional crossover and uniform crossover on instances of graph bisection.

Our test domain is a single instance of the VLSI standard-cell placement problem. A given circuit description may be viewed as a hypergraph $G = (V, E)$ where V corresponds to the set of modules, or *cells*, and E corresponds to the set of signal nets. Each signal net is a hyperedge connecting two or more cells. All cells are assumed to have unit width and unit height; the objective is to place the cells on a given number of equal-width rows such that the total interconnection wire length, denoted *net cost*, is minimized. This problem is well-known to be NP-hard [11]. [4] Throughout this paper, the test case that we use is the *fract* benchmark circuit which contains 149 cells.[5]

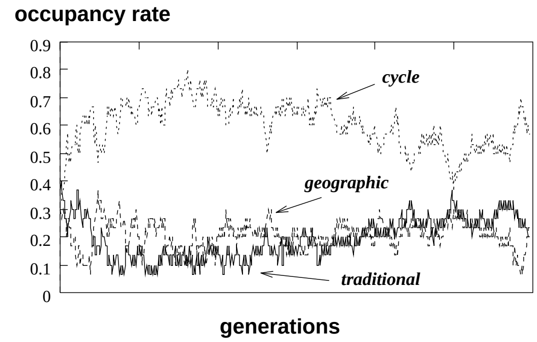

Figure 1: Occupancy rates of three crossovers plotted against time (generation number) for the crossover combination $0.33X_1 + 0.33X_2 + 0.33X_3$.

The underlying optimization engine consists of a steady-state hybrid-type GA, i.e., a GA which applies a local improvement heuristic after mutation. Hybrid-type GAs generally produce much better results than GAs that do not use local improvement, and we use them to reduce running time and enable more extensive experiments. We do not discuss the local improvement heuristic itself except to note that it is a simple iterative local improvement heuristic of our own; further details are beyond our present scope. After generating each offspring, the replacement scheme of [2] is used, i.e., (i) replace the closer parent to the offspring (say, according to Hamming distance) if the offspring is better; (ii) if this fails, replace the other parent if the offspring is better; and (iii) if this also fails, replace the worst-quality solution in the population. Unless otherwise noted, all performance comparisons in this paper are based on averages of solution qualities (total wire lengths) over 100 trials.

3 Observations

3.1 Dynamic Influence of Crossovers

Henceforth, we let X_1, X_2, and X_3 respectively denote traditional crossover, cycle crossover, and geographic crossover. Our first experiments studied the crossover combination $0.33X_1 + 0.33X_2 + 0.33X_3$, i.e., the same usage rate for each of the three crossovers. Over the 100 GA executions, we observed that cycle crossover generates the largest number (approximately 85%) of the "new bests" during a given GA run. However, at the end of a

[1] In our experience, it does not matter whether the probabilities of applying the respective crossovers are enforced by coin-tossing, or by a more deterministic scheme (e.g., cycling through the crossover types according to the proportions C_i). All of our experimental results below are based on a coin-tossing scheme.

[2] We use 5-point crossover for this.

[3] These three crossovers seem likely to afford very different recombinations, but we do not have any formal analysis of this issue.

[4] The optimal interconnection of the terminals in a multi-terminal signal net is given by a rectilinear Steiner tree, which is difficult to compute. Following standard practice, our net cost measure approximates the Steiner tree cost by the half perimeter of the smallest rectangular bounding box that encloses the terminals of the signal net.

[5] This test case is found in the LayoutSynth90 benchmark suite maintained by ACM SIGDA; it and related test cases are available by anonymous ftp to mcnc.org:/pub/benchmark.

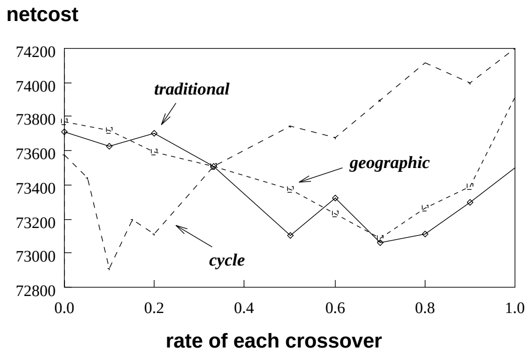

Figure 2: Final solution quality plotted against mixing coefficient of each given crossover (the other two mixing coefficients are assumed equal).

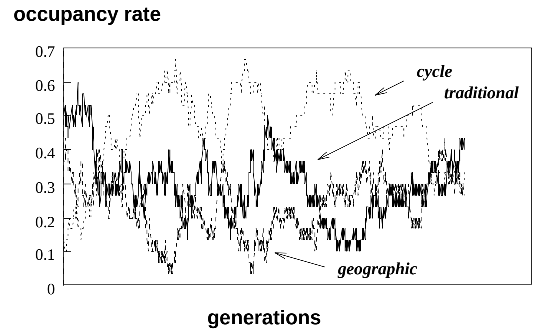

Figure 3: Occupancy rates of three crossovers plotted against time (generation number) for the combination $0.45X_1 + 0.1X_2 + 0.45X_3$.

GA run, all three crossovers were equally likely to have generated the final best (the final solution).

Define the *occupancy rate* of a crossover at a given generation to be the number of solutions in the population that were generated by the crossover, over the total number of solutions in the population. Figure 1 shows typical occupancy rates of the three crossovers, plotted against generation number (i.e., time), with the $0.33X_1 + 0.33X_2 + 0.33X_3$ combination. It is likely that cycle crossover has such a dominant occupancy rate (as well as a high proportion of "new bests") because its low disruptivity makes it likelier to generate higher-quality solutions.

3.2 Performance of Different Combinations

Because we perform at least 100 trials to obtain a single data point, it is difficult to exhaustively analyze all possible combinations. Thus, we examined the restricted set of combinations for which at least two of the three mixing coefficients C_i are the same. In other words, we studied "isosceles" combinations of form $C_1X_1 + \frac{1-C_1}{2}X_1 + \frac{1-C_1}{2}X_3$, or $\frac{1-C_2}{2}X_1 + C_2X_2 + \frac{1-C_2}{2}X_3$, or $\frac{1-C_3}{2}X_1 + \frac{1-C_3}{2}X_2 + C_3X_3$. Figure 2 shows the relationship between each crossover and the average performance obtained for each of its combinations with the other two (evenly split) crossovers. Here, the performance seems roughly unimodal in the relative amount of a given crossover.

Figure 2 also indicates the performance of each *isolated* crossover: the solution qualities plotted above rate = 1.0 represent the performances of the three combinations $1.0X_1$, $1.0X_2$ and $1.0X_3$, with traditional crossover ($1.0X_1$) performing best and cycle crossover ($1.0X_2$) performing worst. (This phenomenon might change if a non-hybrid GA is used.) The best combination we tried ($0.45X_1 + 0.1X_2 + 0.45X_3$) substantially outperforms the best single-crossover strategy, i.e., $1.0X_1$. Figure 3 shows occupancy rates of the three crossovers over time during a typical GA run for the combination $0.45X_1 + 0.1X_2 + 0.45X_3$.

3.3 α: Mean Variance

Given that Figure 3 shows more balanced occupancy rates than Figure 1, it is possible that the balance of occupancy rates affects the overall performance of the multiple-crossover paradigm. One possible intuitive measure of "balance" might be the parameter α, defined as follows.

Given k different crossovers within a steady-state GA framework, let p_j be the occupancy rate of the j^{th} crossover at a given generation. Let

$$\alpha = \frac{\sum_{i=s}^{t} \sigma_i^2}{t - s + 1}$$

where

$$\sigma_i^2 = \frac{\sum_{j=1}^{k} (p_j - \frac{1}{k})^2}{k}.$$

The variable s indicates the index of the first generation at which all individuals in the initial population were replaced by generated offsprings; the variable t is the index of the last generation.

For our present testbed, the α measure turns out to be strongly correlated with performance, i.e., a combination that yields good performance is likely to have a small α value. Figure 4 plots the average α values of different combinations (averaged over 20 trials for each) and their average solution qualities. The correlation coefficient (i.e., the strength of linear relationship) between α and the final solution quality is 0.876.

4 Adaptive Crossovers

Assume that we are to adaptively combine two different crossovers, e.g., traditional crossover and cycle crossover. In this section, we describe three possible heuristics, the last two of which are motivated by the concept of "balance" in the occupancy rates of the crossovers.

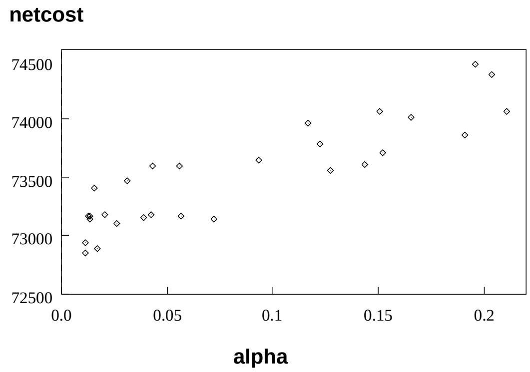

Figure 4: Correlation between solution cost and the value of the α parameter.

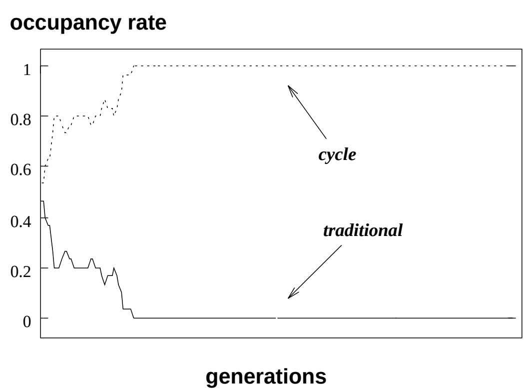

Figure 5: Occupancy rates of two crossovers under Strategy #1.

4.1 Adaptive Strategy #1

Strategy #1 is an adaptive crossover where (for the example of mixing traditional and cycle crossovers): (i) if both parents were generated by traditional crossover, traditional crossover is applied; (ii) if both parents were generated by cycle crossover, cycle crossover is applied; and (iii) if one parent was generated by traditional crossover and the other parent was generated by cycle crossover, either of the two crossovers (randomly selected by a coin toss) is applied. This strategy is very similar to that suggested by Spears in [17], wherein (i) and (ii) still apply, but in the case of (iii) the crossover is selected by a tag bit maintained as an extra gene for this purpose.

This strategy preferentially applies the crossover which has generated better results (on average) so far. In practice, one crossover quickly tends to dominate the population and remains dominant thereafter, as shown in Figure 5. This might be undesirable mechanism, if a crossover that is superior in the initial stages of the GA becomes less useful in the later stages. Note that Figure 5 is consistent with the reported data in [17], but shows a somewhat larger gap in occupancy rates between the two crossovers. Again confirming [17], we find that the

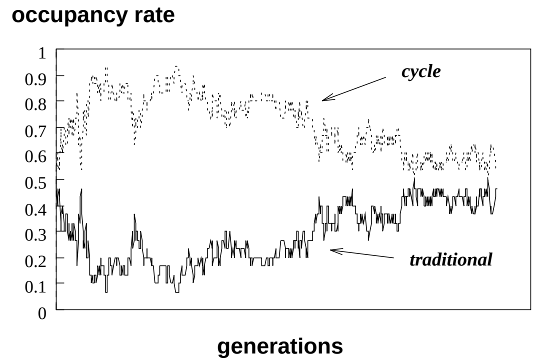

Figure 6: Occupancy rates of two crossovers under Strategy #2.

performance of the adaptive strategy is intermediate between those of the two underlying crossovers. Since this actually indicates a negative effect of this type of adaptive strategy, we next propose two new adaptive crossover strategies that each offer promising empirical performance.

4.2 Adaptive Strategy #2

Strategy #2 is simply the *opposite* of Strategy #1, i.e., (i) if both parents were generated by cycle crossover, apply traditional crossover; (ii) if both parents were generated by traditional crossover, apply cycle crossover; and (iii) select the crossover via a random coin toss otherwise. Thus, for any pair of crossovers, Strategy #2 balances the occupancy rate by probabilistically giving lower priority to the crossover which has affected more members of the current population. Figure 6 shows that the occupancy rates of the cycle and traditional crossovers are more balanced under Strategy #2.

4.3 Adaptive Strategy #3

Assuming that reducing the mean variance of occupancy rates results in improved performance (recall the study of the α parameter above), we propose yet another adaptive strategy which uses a greedy balancing approach. For a given set of k crossovers, Strategy #3 tries to maintain an occupancy rate for each crossover as close as possible to $1/k$ for all generations (a very strong constraint). Initially, random numbers ranging from 1 to k are assigned to the solutions in the population; this randomly relates each solution to a specific crossover. At each generation, the crossover with least occupancy rate is applied. We easily have

Fact 1 *(Assume without loss of generality that the population size p is a multiple of k.) Under Strategy #3, once a balance of occupancy rates is achieved, all occupancy rates remain within the range* $[1/k - 1/p,\ 1/k + 1/p]$.

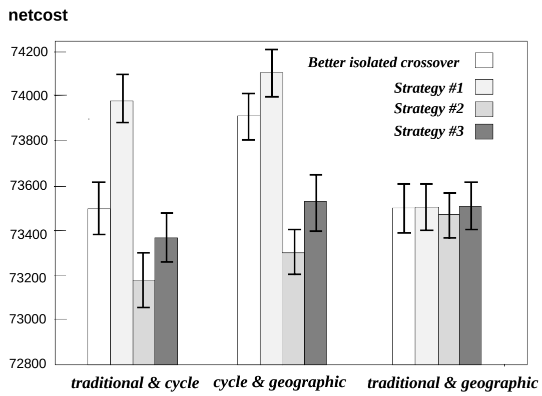

Figure 7: Performances of three strategies for adaptive combination of crossovers, relative to performance of the better of the two underlying crossovers. Error bars indicate +/- one sample standard deviation.

Hence, the occupancy rate of each crossover under Strategy #3 will essentially remain constant at $1/k$. Figure 7 shows that for the VLSI standard-cell placement instance, Strategies #2 and #3 substantially outperform Strategy #1. Observe that the performances of Strategies #2 and #3 are comparable, which may imply that maintaining $\alpha \simeq 0$ is too strong an objective (e.g., flexibility in maintaining small α values may be beneficial). Note also that both Strategy #2 and Strategy #3 performed better than any of the isolated underlying crossovers; this is in contrast to the results observed for Strategy #1 (and in [17]).

5 Discussion

The exact mechanism by which using two or more different crossovers improves GA performance is still a difficult open issue. In this paper, we report initial studies which examine various aspects of the synergetic combination of different crossovers within a single GA implementation. We also propose two new adaptive crossover combination strategies, which for the present testbed outperform the previous method of [17]. Our ongoing work is aimed at a number of open questions and gaps in our experimental methodology, including the following.

- Our experiments use hybrid-type GAs for the efficiency reasons discussed above, but the local search hybridization may mask aspects of the orthogonal combination of crossovers. More computationally demanding studies can isolate the effects of crossover combination using non-hybrid type GAs.
- We have used the notion of "orthogonality" to intuitively convey a synergetic relationship of different crossovers. It would be useful to achieve more formal definitions and measures for the orthogonality between two or more crossovers. Existing analyses of structure in fitness landscapes (e.g., [1, 9]) might be helpful here, since the concept of fitness landscape is directly associated with the concept of solution space traversal.
- Maintaining strict balance of occupancy rates ($\alpha \simeq 0$) did not significantly improve solution quality. Indeed, Strategy #2 outperforms Strategy #3 on the circuit placement problem despite having higher α values. Thus, it is likely that other factors must be considered, as we note below. For example, in our present experimental testbed solutions associated with cycle crossover can have higher probability of being selected as parents under proportional selection, since they are likely to be of higher quality than solutions generated by other crossovers. In evaluating the influence of each crossover in the population, one might consider such parameters as the sum of solution fitnesses associated with each crossover, extended genealogical histories, etc.
- In our experiments, better-performing combinations of crossovers usually run longer than inferior combinations. While it is unlikely that the inferior combinations can find significantly better solutions just by tightening the convergence criteria (i.e., extending the number of generations), a more careful comparison must vary and normalize resource usage (e.g., by tuning the number of generations and the population size).

Finally, we again emphasize that the crossovers used in this paper represent only several tentative combinations, and that the choice of testbed (VLSI placement, with hybrid-type GA) introduces many experimental variables. It is not yet possible to conclude that the phenomena observed in this paper hold more generally for other combinations of different crossovers. We have recently combined 2-point crossover and uniform crossover for another combinatorial optimization problem (the planar Euclidean traveling salesman problem), and found that the combination always had performance intermediate between those of the two underlying crossovers no matter which of our adaptive strategies was used. This is more consistent with the results reported in [17], and clearly points out the need for more detailed future investigations.

Acknowledgments

We thank William Spears for providing careful comments which helped improve this paper.

References

[1] K. D. Boese, A. B. Kahng, and S. Muddu. A new adaptive multi-start technique for combi-

natorial global optimizations. *Operations Research Letters*, 15:101–113, 1994.

[2] T. N. Bui and B. R. Moon. Graph partitioning and genetic algorithms. 1994. To appear *IEEE Trans. on Computers*.

[3] K. DeJong and W. Spears. An analysis of the interacting roles of population size and crossover in genetic algorithms. In *Parallel Problem Solving from Nature I*, pages 38–47. Springer-Verlag, 1990.

[4] K. DeJong and W. Spears. A formal analysis of the role of multi-point crossover in genetic algorithms. *Annals of Math. AI Journal*, 5:1–26, 1992.

[5] L. Eshelman, R. Caruana, and D. Schaffer. Biases in the crossover landscape. In *Third International Conference on Genetic Algorithms*, pages 10–19, 1989.

[6] L. Eshelman and D. Schaffer. Crossover's niche. In *Fifth International Conference on Genetic Algorithms*, pages 9–14, 1993.

[7] D. Fogel and L. Stayton. On the effectiveness of crossover in simulated evolutionary optimization. *Biological Cybernetics*, 32:171–182, 1994.

[8] D. Goldberg and R. Lingle. Alleles, loci, and the traveling salesman problem. In *First International Conference on Genetic Algorithms and Their Applications*, pages 154–159, 1985.

[9] T. Jones and S. Forrest. Fitness distance correlation as a measure of problem difficulty for genetic algorithms. In *Sixth International Conference on Genetic Algorithms*, pages 184–192, 1995.

[10] A. B. Kahng and B. R. Moon. Toward more powerful recombinations. In *Sixth International Conference on Genetic Algorithms*, pages 96–103, 1995.

[11] F. T. Leighton. *Complexity Issues in VLSI*. MIT Press, 1983.

[12] I. Oliver, D. Smith, and J. Holland. A study of permutation crossover operators on the traveling salesman problem. In *Second International Conference on Genetic Algorithms*, pages 224–230, July 1987.

[13] J. D. Schaffer and L. J. Eshelman. On crossover as an evolutionarily viable strategy. In *Fourth International Conference on Genetic Algorithms*, pages 61–68, 1991.

[14] J. D. Schaffer and A. Morishima. An adaptive crossover distribution mechanism for genetic algorithms. In *Second International Conference on Genetic Algorithms*, pages 36–40, 1987.

[15] K. Shahookar and P. Mazumder. A genetic approach to standard cell placement using meta-genetic parameter optimization. *IEEE Trans. on Computer-Aided Design*, 9(5):500–511, 1990.

[16] D. Smith. Bin packing with adaptive search. In *First International Conference on Genetic Algorithms*, pages 202–206, 1985.

[17] W. M. Spears. Adapting crossover in evolutionary algorithms. In *Proc. Fifth Annual Conference on Evolutionary Programming*, 1995.

[18] W. M. Spears and V. A. Anand. A study of crossover operators in genetic programming. In *International Symposium on Methodologies for Intelligent Systems*, pages 409–418, 1991.