# An Examination of Building Block Dynamics in Different Representations

**Annie S. Wu**
Naval Research Laboratory
Code 5514
Washington, DC 20375
aswu@aic.nrl.navy.mil

**Kenneth A. De Jong**
Department of Computer Science
George Mason University
Fairfax, VA 22030
kdejong@cs.gmu.edu

**Abstract- We compare the traditional, fixed genetic algorithm (GA) representation scheme with a floating representation scheme and examine the differences in building block dynamics and how these differences affect a GA's ability to balance exploration and exploitation of building blocks. This study examines both the overall performance of a GA and the detailed events that contribute to overall behavior. Results indicate that the floating representation allows a GA to maintain a higher-level of construction which results in a more diverse population from which to build solutions.**

## 1 Introduction

A number of key characteristics distinguish evolutionary search algorithms such as their use of a population to allow for parallel search, their use of genetic operators to facilitate exploration of the solution space, and their use of selection to exploit known information. There are several types of evolutionary algorithms including evolutionary programming, evolutionary strategies, and genetic algorithms. This paper focuses on examining the building block dynamics in one of these types, the genetic algorithm (GA). The GA's search strategy is commonly thought to be described by the Building Block Hypothesis (Goldberg 1989; Holland 1975): sample the solution space in parallel for small but highly-fit building blocks or partial solutions, modify and combine these highly-fit building blocks to form larger, highly-fit building blocks, and repeat this process recursively until a complete solution is found. Numerous studies have focused on defining building blocks (Goldberg, Korb, and Deb 1989; Haynes 1997), encouraging effective mixing of building blocks (Thierens and Goldberg 1993), and understanding building block dynamics in a GA (Forrest and Mitchell 1992; Spears 1998; van Kemenade 1997; Wu and Lindsay 1996; Wu, Lindsay, and Riolo 1997).

In order for this strategy to work, a fine balance must be maintained between the exploitation of known building blocks of information and exploration for new building blocks. Unfortunately, the traditional GA may not be as finely tuned to this balance as we expect. A detailed study of the traditional GA found that a GA typically spends a significant portion of a run with a relatively homogeneous population (Wu and Lindsay 1996). This observation suggests that a GA does not take full advantage of its abilities to perform parallel search and recombination. Closer examination revealed that while a GA is very effective at retaining building blocks that it has found and propagating these building blocks throughout its population, the genetic operators of crossover and mutation are not effective enough in the construction of new building blocks to maintain a sufficiently diverse collection of building blocks within the population (Wu, Lindsay, and Riolo 1997). The exploitative force of selection appears to be severely overpowering the exploratory force of the genetic operators, resulting in a population of individuals that all contain approximately the same building blocks.

A comparison of the traditional, fixed GA representation scheme with a alternative method, called the floating representation, found that a GA using the floating representation maintains a more diverse population and performs significantly better than a GA using the fixed representation (Wu and Lindsay 1996). In this paper, we examine the differences in the building block dynamics found with these two representations and try to understand whether the floating representation improves the GA's ability to balance exploration and exploitation of building blocks. This empirical study focuses on not only gross population statistics and behavior, but also on detailed events and how they contribute to a run as a whole. Because this work focuses on building block dynamics, all references to problems are assumed to be problems whose solutions are clearly made up of building blocks. For example, in parameter optimization problems with multiple parameters, each parameter value can be considered as a building block. Groups of parameters (building blocks) that interact or influence each other may be interpreted as higher level building blocks. We will show in this paper that the floating representation allows the GA to maintain a higher level of construction (of new building blocks) than the traditional, fixed representation with a slightly lower increase in disruption. This allows the GA to maintain a more diverse population of building blocks from which it can build a solution.

## 2 GA representations

GA representations typically encode problem solutions into linear arrays of characters. The traditional *fixed representation* scheme determines the number of characters required to represent all values of each building block, then assigns the required number of bits to each building block. Each build-

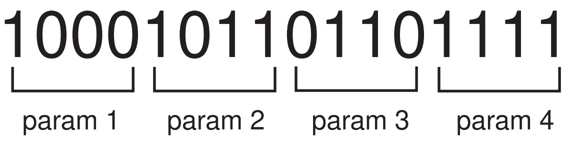


Figure 1: An example of the traditional, fixed GA encoding scheme.

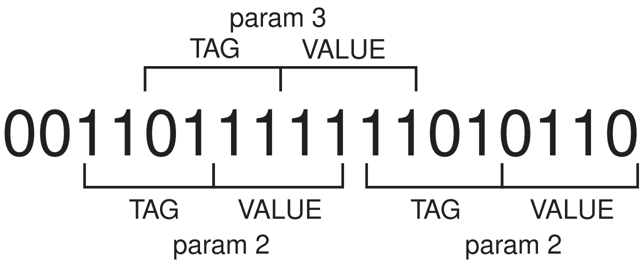


Figure 2: An example of the floating building block encoding scheme.

ing block is placed adjacent to the previous on the individual. The length of an individual is exactly the total number of characters required to encode all of the building blocks of the solution. Figure 1 shows an example of how a problem with four building blocks may be encoded into a binary string. The value or pattern of each building block is obtained from the specified locations. The advantage of this representation is that it is very straightforward. Fixing building blocks at the same location on all individuals simplifies the evaluation of individuals and the application of operators such as crossover. The disadvantage of this representation is that the user must make a number of assumptions in setting up the problem encoding. In particular, the relative positions of building blocks on an individual are decided arbitrarily by the user. Because of the positional bias of the crossover operator, the relative positioning of building blocks can affect GA performance (Eshelman, Caruana, and Schaffer 1989).

The *floating representation* introduces the idea of location independent building blocks to GA representations (Wu and Lindsay 1996). Each building block is assigned a unique tag or pattern of characters. A building block exists on an individual if its tag is found on that individual. The value of a building block is encoded in the characters immediately following the tag. Figure 2 shows an example of a floating building block encoding scheme. Suppose a problem consists of four parameters or building blocks. If we assign the following tags:

| | |
|---|---|
| Building block 1: | `0000` |
| Building block 2: | `1101` |
| Building block 3: | `0111` |
| Building block 4: | `1001` |

Then the example individual in Figure 2 has zero copies of building block 1, two copies for building block 2, one copy of building block 3, and zero copies of building block 4. The advantages of the floating representation include the fact that a GA is now able to evolve the number, placement, and value (encoded information) of the building blocks on its individuals. In addition, building blocks may overlap each other, as seen in Figure 2, providing increased selection pressure on certain regions of an individual. The disadvantages of this representation include the fact that the fitness function must be able to handle both overspecification (multiple copies) and underspecification (missing copies) of building blocks. The ability for building blocks to overlap allows genetic operators to both construct and disrupt multiple building blocks in a single operation. The floating representation also results in the appearance of *noncoding regions* which are ignored by the fitness evaluation function. Though such regions make no apparent direct contribution to fitness, they are thought to encourage the recombination of building blocks as well as provide a buffer against crossover's disruptive tendencies.

## 3 Experimental methods

The goal of this work is to understand how building block dynamics differ between the fixed and floating representations. All experiments were performed on the Royal Road functions (Jones 1994; Mitchell, Forrest, and Holland 1991). The Royal Road functions are a set of functions designed specifically for the study of building block interactions. Each function consists of a set of building blocks whose pattern (or value) and fitness contribution are predefined. If a building block exists on an individual, the fitness contribution of that building block is added to the individual's fitness. The goal of a GA is to find an individual containing one copy of each building block of the function.

A number of different Royal Road functions were tested in these studies. Examples from two specific functions will be discussed in this paper. The RR function, which uses a fixed representation, consists of 16 predefined building blocks. The goal of a GA is to find an individual containing all 16 predefined "optimum" patterns at the assigned locations. A building block exists if all bits match the predefined optimum; it does not exist if one or more bits do not match the predefined optimum. The FRR function, which uses a floating representation, consists of 16 predefined building blocks, each assigned a unique tag. Since a Royal Road building block either exists or does not exist, for the purposes of the discussions in this paper, we assume that a GA solving the FRR function searches for an individual containing 16 unique tags, where the length of each tag is equal to the length of the building blocks in the RR function. Additional details about these functions are available in (Wu and Lindsay 1996).

The majority of building block activity occurs during reproduction, when building blocks may be constructed, propagated, or disrupted. As a result, our experimental methods will focus on examining what happens to building blocks from one generation to the next. In addition, we observe dif-

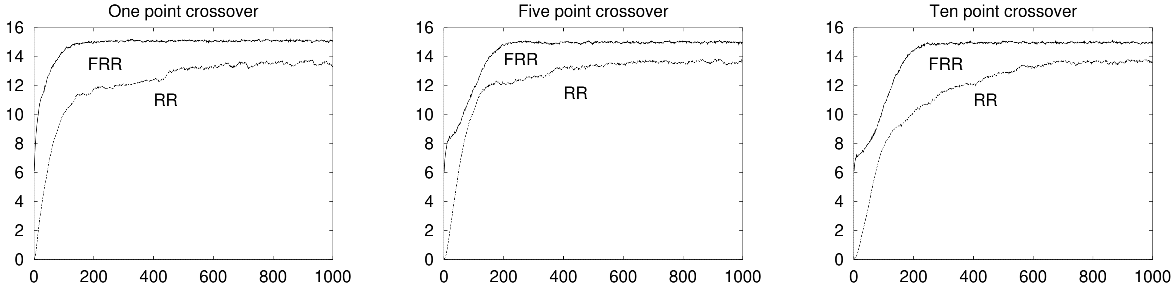


Figure 3: Average fitness of population through time. The $x$-axis indicates generation. The $y$-axis indicates fitness.

ferences across several crossover types, including one-point, two-point, and a probabilistic crossover with settings ranging from an average of two crossovers per individual to an average of 20 crossovers per individual.

A number of preliminary experiments were performed. From these experiments, we settled on the following parameter values. Population size was maintained at 400, crossover rate was set to 0.7, and mutation rate was set to 0.005. The primary variations in the experiments discussed here are problem representation (fixed or floating) and crossover type (one-point, two-point, multipoint). Each experiment lasted for 1000 generations and the data presented here are either from individual runs or averaged over ten runs. Those data from individual runs were checked to ensure that they were representative of all runs from that experiment. The specific areas that we examine include the general behavior of the GA, the survival of building blocks, the source of building blocks, and the activity of crossover.

**General GA performance**

We are studying the floating representation because previous work suggested that it produces better GA performance than the fixed representation. We will first verify previous results through examinations of the amount of time required to find an optimum, the change in population fitness through time, and the population diversity. Population diversity refers to the collection of building blocks in each population.

**Building block survival**

Ideally, we would like the GA to propagate all useful building blocks from one generation to the next. Realistically, the building blocks of a population may or may not survive from one generation to the next. A building block that does not survive to the next generation can die from one of two causes. If the individual on which a building block is found is not selected to reproduce, that building block will not survive to the next generation. If the individual is selected to reproduce but the building block is disrupted by genetic operators, that building block will not survive. We will examine the factors that affect the survival of building blocks.

**Building block source**

While past experience is an invaluable source of information, a GA must also maintain a significant level of construction of new information to continue exploration of the solution space. The building blocks in a generation can originate from one of two sources. A building block may be inherited from the previous generation or a building block may be created by genetic operations. We will examine the factors that affect building block construction.

**Crossover activity**

The expected role of genetic operators is to explore of the solution space of a problem to be solved. This exploration is accomplished through the construction of new building blocks and the disruption of existing building blocks. We measure the effectiveness of crossover by tracking how active it is at constructing and disrupting building blocks.

## 4 Results

The results obtained from these experiments provide an interesting comparison of the two representation schemes and indicate that the floating representation can positively influence a GA.

In looking at overall GA behavior, our results confirmed previous conclusions that, in general, a GA performs better with the floating representation. Figure 3 shows changes in average population fitness through time. These plots are averaged over ten runs. The average population fitness increases faster and levels off at a higher value with the FRR function than the RR function. This behavior is consistent regardless of the type of crossover used. The GA takes approximately ten times longer to find a solution to the RR function (300-500 generations) than to the FRR function (40-60 generations). An examination of population diversity also confirms previous results. Figure 4 plots the average number of copies of each building block on an individual through time. Although each line represents one building block, details are not relevant in these plots. The purpose of these plots are to show that, overall, the GA maintains an average of 1.2 to 1.5 copies of each building block on each individual with the FRR function. Examination of the populations found noticeable vari-

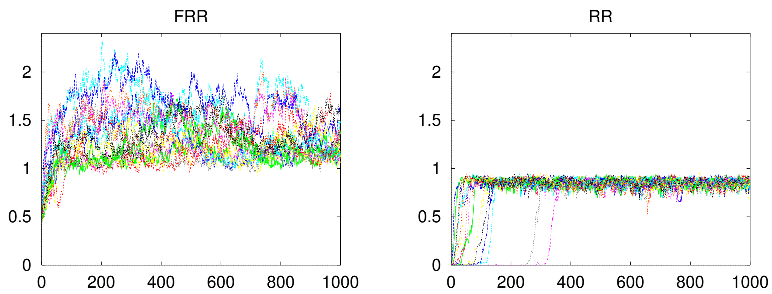


Figure 4: Average number of copies of each building block on an individual. The $x$-axis indicates generation. The $y$-axis indicates number of copies. Each line represents one building block.

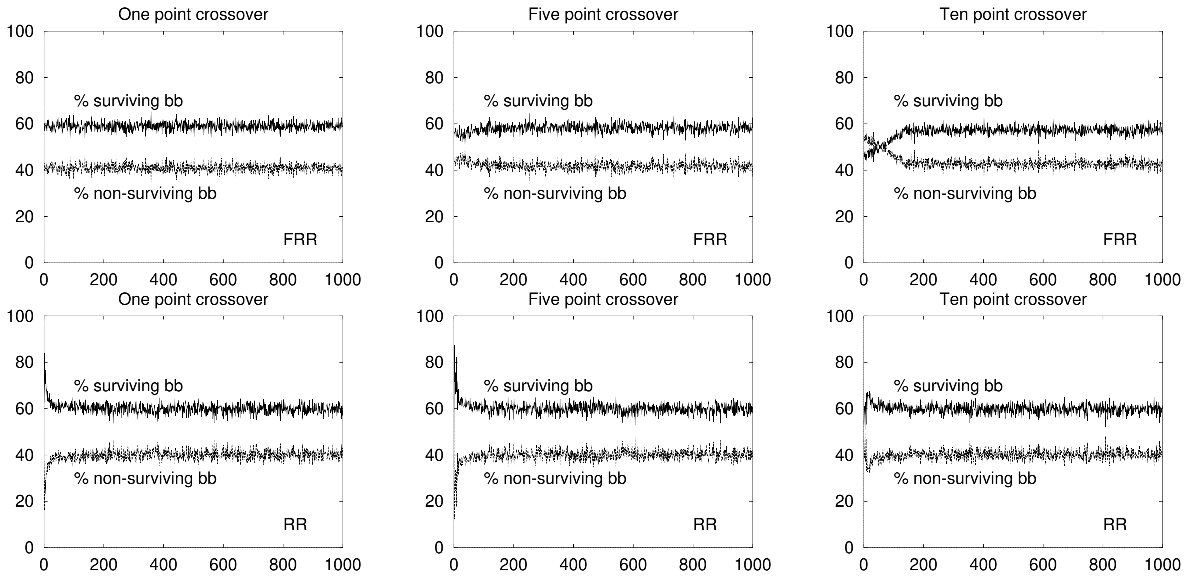


Figure 5: Percentage of building blocks that do and do not survive to next generation. The $x$-axis indicates generation. The $y$-axis indicates percentage of total building blocks in a generation.

ation in genetic make up from individual to individual. Individuals with identical fitness values may have different arrangements of building blocks. With the RR function, the GA searches for one building block at a time, immediately propagating each new building block to all individuals in the population.

Investigation into the survival of building blocks revealed interesting and somewhat surprising results. The plots in Figure 5 show the percentage of building blocks in each generation that do and do not survive to the next generation as a run progresses. Each plot shows a typical run. The plots in the top row involve the FRR function. The plots in the bottom row involve the RR function. Each column shows a different type of crossover. Although there is some difference early in a run, all percentages quickly level off to approximately 60 percent of the building blocks in a generation surviving to the next generation and approximately 40 percent of the building blocks in a generation not surviving. Interestingly, these values remain consistent regardless of function representation or crossover type.

The reason for this similarity becomes clear when we examine the reasons why building blocks do not survive. The plots in Figure 6 separate the building blocks that do not survive into two groups: (1) those that do not survive because the individual they were on was not selected to reproduce and (2) those that do not survive because of disruption by genetic operators. These plots show that more than 97 percent of the building blocks that do not survive into the next generation die because the individual they were on was not selected to reproduce. In the RR runs, essentially all building blocks are lost due to no reproduction. While there is a slight increase in the percentage of disrupted building blocks with increasing crossover points with the FRR function, the difference is not significant and this behavior remains relatively consistent over all experiments. The reproduction rate of a population is determined by the selection method and is unaffected by genetic operators or representation. Because, as we see from Figure 6, reproduction can play a significant role in the survival of building blocks, the invariant disruptive behavior seen in Figure 5 can be attributed to the fact that se-

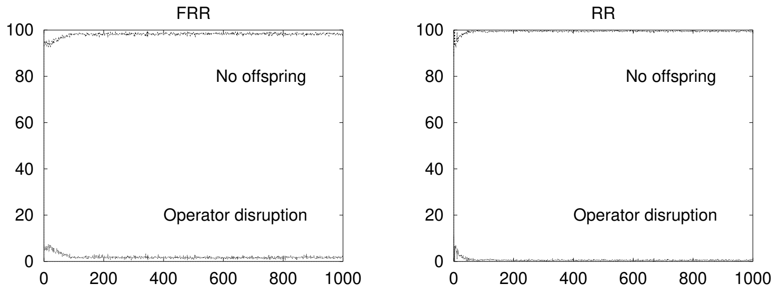


Figure 6: Percentage of building blocks that do not survive to next generation because the individual they were on was not selected to reproduce and the percentage of building blocks that do not survive because of disruption from genetic operators. The $x$-axis indicates generation. The $y$-axis indicates percentage of total building blocks in a generation.

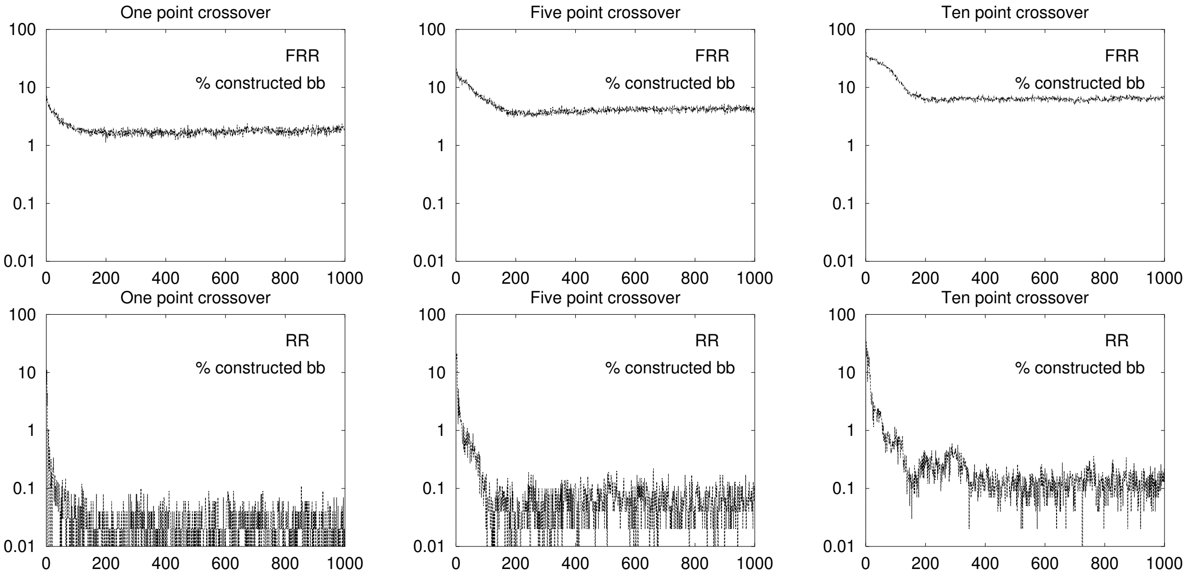


Figure 7: Percentage of building blocks that are constructed in each generation. The $x$-axis indicates generation. The $y$-axis indicates percentage of total building blocks in a generation.

lection was held constant in all of our experiments. In these particular runs, selection pressures are the dominate cause of building block loss. Genetic operators appear to play a minor role.

In examining the source of building blocks, we find noticeable differences between the two types of representations. Figure 7 shows the percentage of building blocks that are constructed in each generation (all remaining building blocks are inherited). Each plot shows a typical run. The plots in the top row involve the FRR function; the plots in the bottom row involve the RR function. Each column refers to a different type of crossover. The FRR function shows a noticeable increase in the construction of building blocks as the number of crossover points increases. Though the percentage of constructed building blocks is still low, particularly for one-point crossover, the FRR function consistently shows significantly more construction of building blocks than the RR function. The RR function shows little or no construction and no discernible change with changing crossover type.

Shifting our focus from building blocks to the operators that affect building blocks, we examine crossover activity through time. Figure 8 shows plots of the percentage of crossovers in each generation that construct at least one building block. Figure 9 shows plots of the percentage of crossovers in each generation that disrupt at least one building block. Crossover is clearly more active with the floating representation than the fixed representation, quickly peaking at 50 to 70 percent then leveling off at approximately 20 percent. With the fixed representation, crossover activity never gets above 12 percent and stays mostly level at around one percent. Variations in crossover type appear to affect the amount of time that peak crossover activities are sustained, with increased crossover points resulting in longer periods of higher construction and disruption.

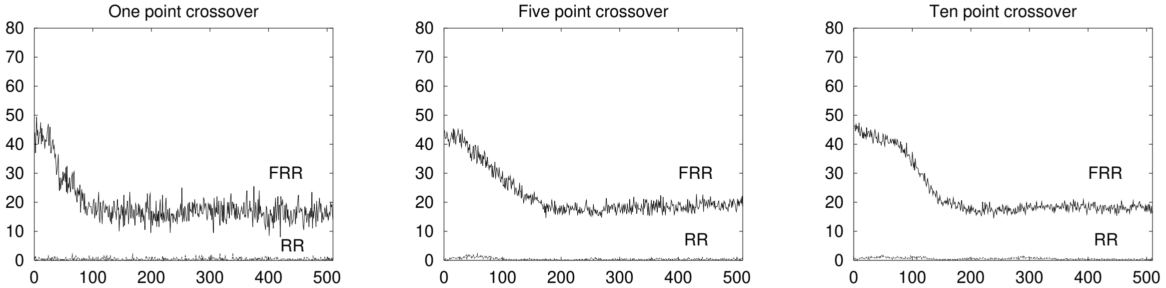


Figure 8: Percentage of crossovers in a generation that construct building blocks. The $x$-axis indicates generation. The $y$-axis indicates percentage of total crossovers in a generation.

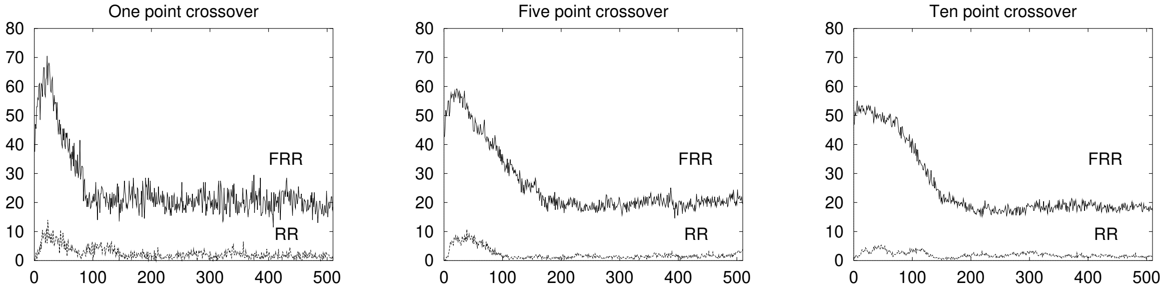


Figure 9: Percentage of crossovers in a generation that disrupt building blocks. The $x$-axis indicates generation. The $y$-axis indicates percentage of total crossovers in a generation.

## 5 Discussion

This paper describes a series of studies on building block dynamics in a GA. We compare observations of a GA using the traditional, fixed representation scheme with observations of a GA using a floating representation scheme. These studies were performed on an artificial function, called the Royal Road function, because it allows detailed monitoring of building block interactions. Further investigations on real world problems would significantly strengthen the conclusions that we draw here.

Comparisons of overall GA behavior confirmed previous conclusions that a GA is able to maintain a more diverse population and deliver better performance with the floating representation. The added diversity is thought to be due in part to the flexibility of the floating representation. With the traditional fixed representation, fitness and genetic make up are closely linked. Individuals with similar fitness have a high probability of having similar genetic makeup. With the floating representation, this connection is much weaker. The flexibility of location independent building blocks expands the pool of individuals with similar fitness to include a much greater variety of genetic make ups. As a result, it is possible for a GA to find high fitness individuals without converging and losing genetic diversity.

Using a floating representation clearly increases crossover activity, both constructive and disruptive. Examinations of the survival of building blocks reveal that both representations propagate approximately the same proportion of building blocks to the next generation. In the function studied here, approximately 60 percent of the building blocks in each generation survive to the next generation. Interestingly, this proportion appears to be more dependent on selection method than on genetic operators or representation. The proportion of building blocks that are inherited from the previous generation is significantly larger indicating that the proportion of newly constructed building blocks in a population is low. Nonetheless, there is a small but significant increase in building block construction with the floating representation.

The somewhat unexpected link between selection and building block survival is interesting and deserves further investigation. Observations indicate that the majority of building blocks that do not survive to the next generation die because the individual they are on is not selected to reproduce. As a result, genetic operators appear to play a minor role in discouraging the propagation of building blocks. A strong connection between selection and building block survival suggests that it may be possible to increase construction of building blocks with increasing genetic operator activity while absorbing some of the disruptive effects with appropriate adjustments to selection pressure.

**Acknowledgments**

This research was conducted at the Naval Research Laboratory with support from the National Research Council and the Office of Naval Research. The authors would like to thank David Fogel for his helpful comments.

## Bibliography


Eshelman, L. J., R. A. Caruana, and J. D. Schaffer (1989). Biases in the crossover landscape. In *Proc. 3rd International Conference on Genetic Algorithms*, pp. 10–19.

Forrest, S. and M. Mitchell (1992). Relative building-block fitness and the building-block hypothesis. In *Foundations of Genetic Algorithms 2*, pp. 109–126.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley.

Goldberg, D. E., B. Korb, and K. Deb (1989). Messy genetic algorithms: Motivation, analysis, and first results. *Complex Systems 3*, 493–530.

Haynes, T. (1997). Phenotypical building blocks for genetic programming. In *Proc. 7th International Conference on Genetic Algorithms*, pp. 26–33.

Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems*. University of Michigan Press.

Jones, T. (1994). A description of Holland's Royal Road function. *Evolutionary Computation 2*(4).

Mitchell, M., S. Forrest, and J. H. Holland (1991). The royal road for genetic algorithms: Fitness landscapes and GA performance. In *Toward a Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life*.

Spears, W. M. (1998). Dining with GAs: operator lunch theorems. In *Foundations of Genetic Algorithms*.

Thierens, D. and D. E. Goldberg (1993). Toward a better understanding of mixing in genetic algorithms. *Journal Society of Control Engineers 32*(1), 10–16.

van Kemenade, C. H. M. (1997). Cross competition between building blocks. In *Proc. 7th International Conference on Genetic Algorithms*, pp. 2–9.

Wu, A. S. and R. K. Lindsay (1996). A comparison of the fixed and floating building block representation in the genetic algorithm. *Evolutionary Computation 4*(2), 169–193.

Wu, A. S., R. K. Lindsay, and R. L. Riolo (1997). Empirical observations on the roles of crossover and mutation. In T. Back (Ed.), *Proceedings of the 7th International Conference on Genetic Algorithms*, pp. 362–269.