Independent Sampling Genetic Algorithms

Chien-Feng Huang
Center for the Study of Complex Systems, 4477 Randall Lab.
University of Michigan
Ann Arbor, MI 48109
cfhuang@engin.umich.edu
(734)763-3323

Abstract

Premature convergence is the loss of diversity in the population that has long been recognized as one crucial factor that hinders the efficacy of crossover. In this paper, a strategy for independent sampling of building blocks is proposed in order to nicely implement implicit parallelism. Based on this methodology, we developed a modified version of GA: independent sampling genetic algorithms (ISGAs). Simply stated, each individual independently samples candidate schemata and creates population diversity in the first phase; subsequently we allow individuals to actively select their mates for reproduction. We will present experimental results on two benchmark problems, "Royal Road" functions of 64-bits and bounded deception of 30-bits, to show how the performance of GAs can be improved through the proposed approach.

1 INTRODUCTION

Genetic algorithms (GAs) have been successfully applied to several difficult search and optimization problems in science and engineering. One major source of the power of GAs is derived from so-called implicit parallelism (Holland, 1975), i.e., the simultaneous allocation of search effort to many regions of the search space. A perfect implementation of implicit parallelism implies that a large number of different short, low-order schemata of high fitness are sampled in parallel, thus conferring enough diversity of fundamental building blocks for crossover operators to combine them to form more highly-fit, complicated building blocks. However, traditional GAs suffer from premature convergence (Goldberg, 1989) where considerable fixation occurs at certain schemata of suboptimal regions before attaining more advancement. Among examples of premature convergence, hitchhiking (Das, & Whitley, 1991; Mitchell, 1996) has been identified as a major hindrance, which limits implicit parallelism by reducing the sampling frequency of various beneficial building blocks. In short, non-relevant alleles hitchhiking on certain schemata could propagate to the next generation and drown out other potentially favorable building blocks, thus preventing independent sampling of building blocks. Consequently, the efficacy of crossover in combining building blocks is restricted by the resulting loss of desired population diversity.

Mitchell, Holland and Forrest (1994) considered a so-called "idealized genetic algorithm" (IGA) that allows each individual to evolve completely independently; thus new samples are given independently to each schema region and hitchhiking is suppressed. Then under the assumption that the IGA has the knowledge of the desired schemata in advance, they derived a lower bound for the number of function evaluations that the IGA will need to find the optimum of Royal Road function R_1 (Mitchell, Forrest, & Holland, 1992).

However, the IGA is impractical because it requires the exact knowledge of desired schemata ahead of time. Partially motivated by the idea of the IGA, we propose a more robust GA that proceeds in two phases: the "independent sampling phase" and the "breeding phase". In the independent sampling phase, we design a core scheme, named the "Building Block Detecting Strategy" (BBDS), to extract relevant building block information of a fitness landscape. In this way, an individual is able to sequentially construct more highly-fit partial solutions. For Royal Road R_1, the global optimum can be attained easily. For other more complicated fitness landscapes, we allow a number of individuals to adopt the BBDS and independently evolve in parallel so that each schema region can be given samples independently. During this phase, the population is expected to be seeded with promising genetic material. Then follows the breeding phase, in which individuals are paired for breeding based on two mate selection schemes (Huang, 2001): individuals being assigned mates by natural selection only and individuals being allowed to actively choose their mates. In the latter case, individuals are able to distinguish candidate mates that have the same fitness yet have different string structures, which may lead to quite different performance after crossover. This is not achievable by natural selection alone since it assigns individuals of the same fitness the same probability for being mates, without explicitly taking into account string structures. In short, in the breeding phase individuals manage to construct even more promising schemata through the recombination of highly-fit building blocks found in the first phase. Due to the characteristic of independent sampling of building blocks that distinguishes the proposed GAs from conventional GAs, we name this type of GA *independent sampling genetic algorithms* (ISGAs).

2 LITERATURE REVIEW

In GA research, extensive attention has been paid to how to alleviate premature convergence. An example is the class of parallel GAs (PGAs) that are developed to degrade centralized selection control used in simple GAs in order to accommodate more population diversity. Among these PGAs, the "fine-grained" type (Cantú-Paz, 1997) is an idealized model that allows only one individual to evolve in each deme and thereby implements the decentralization of selection scheme to the maximum degree. Mühlenbein (1991) used a local hillclimbing algorithm to refine the individuals in his fine-grained PGAs along with a mating strategy based on population structure and the empirical results showed that his PGA is an effective optimization tool.

The independent sampling phase of ISGAs is similar to the fine-grained PGAs in (Mühlenbein, 1991) in the sense that each individual evolves autonomously, although ISGAs do not adopt the population structure. The second distinction is that Mühlenbein's fine-grained PGAs process strings in a homogeneous fashion. An initial population is randomly generated. Then in every cycle each individual does local hillclimbing, and creates the next population by mating with a partner in its neighborhood and replacing parents if offspring are better. By contrast, ISGAs partition the genetic processing into two phases: the independent sampling phase and the breeding phase as described in the preceding section. Third, the approach employed by each individual for improvement in ISGAs is different from that of the PGAs. During the independent sampling phase of ISGAs, in each cycle, through the BBDS, each individual attempts to extract relevant information of potential building blocks whenever its fitness increases. Then, based on the schema information accumulated, individuals continue to construct more complicated building blocks. However, the individuals of Mühlenbein's PGAs adopt a local hillclimbing algorithm that does not manage to extract relevant information of potential schemata.

The motivation of the two-phased ISGAs was partially from the "messy genetic algorithms (mGAs)" in (Goldberg, Korb, & Deb, 1989; Goldberg, Deb, Kargupta, & Harik, 1993). The two stages employed in the mGAs are "primordial phase" and "juxtapositional phase", in which the mGAs first emphasize candidate building blocks based on the guess at the order k of small schemata, then juxtaposing them to build up global optima in the second phase by "cut" and "splice" operators. However, in the first phase, the mGAs still adopt centralized selection to emphasize some candidate schemata; this in turn results in the loss of samples of other potentially promising schemata. By contrast, ISGAs manage to postpone the emphasis of candidate building blocks to the latter stage, and highlight the feature of independent sampling of building blocks to suppress hitchhiking in the first phase. As a result, population is more diverse and implicit parallelism can be fulfilled to a larger degree. Thereafter, during the second phase, ISGAs implement population breeding through two mate selection schemes as discussed in the preceding section. In this way, we may examine if the results obtained for the ISGAs are consistent with what has been done for simple serial GAs in (Huang, 2001).

In the following sections, we present the key components of ISGAs in detail and show the comparisons between the experimental results of the ISGAs and those of several other GAs on two benchmark test functions.

3 COMPONENTS OF ISGAS

ISGAs are divided into two phases: the independent sampling phase and the breeding phase. We describe them as follows.

3.1 INDEPENDENT SAMPLING PHASE

To implement independent sampling of various building blocks, a number of strings are allowed to evolve in parallel and each individual searches for a possible evolutionary path entirely independent of others.

In this paper, we develop a new searching strategy, Building Block Detecting Strategy (BBDS), for each individual to evolve based on the accumulated knowledge for potentially useful building blocks. The idea is to allow each individual to probe valuable information concerning beneficial schemata through testing its fitness increase since each time a fitness increase of a string could come from the presence of useful building blocks on it. In short, by systematically testing each bit to examine whether this bit is associated with the fitness increase during each cycle, a cluster of bits constituting potentially beneficial schemata will be uncovered. Iterating this process guarantees the formation of longer and longer candidate building blocks.

The operation of BBDS on a string can be described as follows.

1. Generate an empty set for collecting genes of candidate schemata and create an initial string with uniform probability for each bit until its fitness exceeds 0. (Record the current fitness as *Fit*.)

2. Except the genes of candidate schemata collected, from left to right, successively flip all the other bits, one at a time, and evaluate the resulting string. If the resulting fitness is less than *Fit*, record this bit's position and original value as a gene of candidate schemata.

3. Except the genes recorded, randomly generate all the other bits of the string until the resulting string's fitness exceeds *Fit*. Replace *Fit* by the new fitness.

4. Go to steps 2 and 3 until some end criterion.

The idea of this strategy is that the cooperation of certain genes (bits) makes for good fitness. Once these genes come in sight simultaneously, they contribute a fitness increase to the string containing them; thus any loss of one of these genes leads to the fitness decrease of the string. This is essentially what step 2 does and after this step we should be able to collect a set of genes of candidate schemata. Then at step 3, we keep the collected genes of candidate schemata fixed and randomly generate other bits, awaiting other building blocks to appear and bring forth another fitness increase.

However, the step 2 in this strategy only emphasizes the fitness drop due to a bit-flip. It ignores the possibility that the same bit-flip leads to a new fitness rise because many loci could interact in an extremely nonlinear fashion. To take this into account, the second version of BBDS is introduced through the change of step 2 as follows.

Step 2. Except the genes of candidate schemata collected, from left to right, successively flip all the other bits, one at a time, and evaluate the resulting string. If the resulting fitness is less than *Fit*, record this bit's position and original value as a gene of candidate schemata. If the resulting fitness exceeds *Fit*, substitute this bit's "new" value for the old value, replace *Fit* by this new fitness, record this bit's position and "new" value as a gene of candidate schemata, and re-execute this step.

Because this version of BBDS takes into consideration the fitness increase resulted from bit-flips, it is expected to take less time for detecting. Several empirical results so far support this reasoning (for example, the experimental results of these two versions on Royal Road functions shown in the next section).

Other versions of BBDS are of course possible. For example, in step 2, if a bit-flip results in a fitness increase, it can be recorded as a gene of candidate schemata, and the procedure continues to test the residual bits yet without completely travelling back to the first bit to re-examine each bit. However, the empirical results obtained thus far indicate that the performance of this alternative is quite similar to that of the second version. More experimental results are needed to distinguish the difference between them. In this paper, we present the results obtained based on the first and second versions of BBDS.

The overall implementation of the independent sampling phase of ISGAs is through the proposed BBDS to get autonomous evolution of each string until all individuals in the population have reached some end criterion.

In section 4, we will present an analysis of the BBDSs on two types of idealized test functions: "Royal Road" functions (non-deceptive) and problems of bounded deception (deceptive).

3.2 BREEDING PHASE

After the independent sampling phase, individuals independently build up their own evolutionary avenues by various building blocks. Hence the population is expected to contain diverse beneficial schemata and premature convergence is alleviated to some degree. However, factors such as deception and incompatible schemata (i.e., two schemata have different bit values at common defining positions) still could lead individuals to arrive at sub-optimal regions of a fitness landscape. Since building blocks for some strings to leave sub-optimal regions may be embedded in other strings, the search for proper mating partners and then exploiting the building blocks on them are critical for overwhelming the difficulty of strings being trapped in undesired regions. In (Huang, 2001) the importance of mate selection has been investigated and the results showed that the GAs are able to improve their performance when the individuals are allowed to select mates to a larger degree.

In this paper, we adopt two mate selection schemes analyzed in (Huang, 2001) to breed the population: individuals being assigned mates by natural selection only and individuals being allowed to actively choose their mates. Since natural selection assigns strings of the same fitness the same probability for being parents, individuals of identical fitness yet distinct string structures are treated equally. This may result in significant loss of performance improvement after crossover. This issue is the major concern in (Huang, 2001) and we continue this research line to ISGAs in this paper.

We adopt the tournament selection scheme (Mitchell, 1996) as the role of natural selection and the mechanism for choosing mates in the breeding phase is as follows:

During each mating event, a binary tournament selection—with probability 1.0 the fitter of the two randomly sampled individuals is chosen—is run to pick out the first individual, then choosing the mate according to the following two different schemes:

A. Run the binary tournament selection again to choose the partner.

B. Run another two times of the binary tournament selection to choose two highly-fit candidate partners; then the one more dissimilar to the first individual is selected for mating.

The implementation of the breeding phase is through iterating each breeding cycle which consists of 1) Two parents are obtained based on the mate selection schemes above. 2) Two-point crossover operator (crossover rate 1.0) is applied to these parents. 3) Both parents are replaced with both offspring if any of the two offspring is better than them. Then steps 1, 2, and 3 are repeated until the population size is reached and this is a breeding cycle. (To give crossover its stiffest test, we turn off mutation for all the performance tests in this paper.)

In (Huang, 2001), the results showed that the mate selection scheme B outperforms scheme A in general, given the objective of finding the global optimum with minimum time. Since those results were obtained in simple GAs, we are concerned with whether this conclusion can be extended to the ISGAs as well.

Having described the components of ISGAs, we are now on the road to test their performance.

4 EXPERIMENTAL RESULTS

Two types of test functions are used for examining the performance of the ISGAs: “Royal Road” functions (non-deceptive) and problems of bounded deception (deceptive). The performance of some other approaches will be compared with that of the ISGAs as well.

4.1 PERFORMANCE ON ROYAL ROAD FUNCTIONS

The Royal Road functions designed by Mitchell, Forrest, and Holland (1992) were to investigate in more detail the validity of the *Building Block Hypothesis* (Holland, 1975; Goldberg, 1989), which implies that the performance of GAs largely depends on the efficacy of crossover to combine small, highly-fit schemata to form more complex, highly-fit schemata. The fitness landscape of Royal Road functions consists of two characteristics: the presence of short, low-order, highly-fit schemata and hierarchical structure which allows these small schemata to repeatedly construct more and more highly-fit schemata and eventually reach the global optimum. One example of this class is Royal Road R_1 whose fitness landscape is composed of eight consecutive building blocks of eight ones each. It is apparent that Royal Road R_1 is a non-deceptive function and it was expected that GAs perform quite well on such a fitness landscape due to the Building Block Hypothesis. However, Mitchell’s experimental results indicated that the unsatisfactory GA performance on this function is primarily from hitchhiking phenomenon, one of possible causes of premature convergence.

In (Mitchell, 1995), the performance of the GA was further compared with those of three iterated hill-climbing searching algorithms: steepest-ascent hill-climbing (SAHC), next-ascent hill-climbing (NAHC) (Mühlenbein, 1991), and random-mutation hill-climbing (RMHC) (Forrest, & Mitchell, 1993). They performed 500 runs for the GA with population size 128 and 200 runs for each of the three hill-climbing algorithms, and reported that SAHC and NAHC never found the optimum within 256,000 function evaluations but the GA can attain the optimum in an average of 61,334 function evaluations. Moreover, RMHC found the optimum only in an average of 6179 function evaluations, nearly ten times faster than the GA.

We performed 1000 runs of the ISGA on Royal Road

R_1 and the end criterion is the moment for the global optimum being found. It turned out that for such a hierarchical, non-deceptive structure only an individual is needed, based on the building block detecting strategy discussed earlier, to serve for good performance. It actually found the optimum only in an average of 975 function evaluations for the first version of BBDS and 901 for the second version—more than six times faster than RMHC and sixty times faster than the GA.

These experimental results can be summarized in Table 1 in which the standard deviation is shown for each case as well.

Table 1: Experimental Results on R_1

	Function Evaluations to Optimum	
	Mean	Standard Deviation
GA	61334	32583
SAHC	>256,000	–
NAHC	>256,000	–
RMHC	6179	2630
BBDS v. 1	975	314
BBDS v. 2	901	309

To see why the structures aggregated by the BBDS are indeed potentially promising schemata, let us turn to the motivation of the BBDS, i.e., the IGA. On the idealized model Royal Road R_1, Mitchell et al. (1994) discussed the expected time for the IGA to construct all the eight building blocks for reaching the optimum and obtained the theoretical result of 696 function evaluations. In the process of BBDS version 1, when the first building block emerges in the string, 64 evaluations are required to detect it since we need to flip all the 64 bits, one at a time, and evaluate the resulting string. Similarly, as the second building block comes in sight, another 56 evaluations is required to detect it. By this reasoning, the total evaluations required for detecting the building blocks are 64+56+48+40+32+24+16=280. (It is not necessary to detect the final single block because the appearance of the final building block is at the same moment of the optimum being attained.) If two or more building blocks appear simultaneously, the evaluations for detecting will be less than 280, but this occurs with a rather small probability. Therefore, the sum of 696 function evaluations (the theoretic result obtained by Mitchell et al.) for constructing all the eight building blocks and 280 function evaluations for detecting these building blocks is 976. This is almost perfectly consistent with the result obtained for BBDS version 1 shown in Table 1. Thus, we can conclude that BBDS implements nearly the idealized GA on Royal Road R_1 in the sense that extra function evaluations are required to detect the building blocks. As for the performance of the second version of BBDS, since it adopts a more greedy method to detect the building blocks, it takes less time to attain the optimum than the first version does.

Another idealized model to test the power of BBDS is Royal Road R_2 (Forrest, & Mitchell, 1993). This function was designed to verify if the presence of intermediate "stepping stones" (intermediate-order higher-fitness schemata that result from combinations of the lower-order schemata, and that in turn can combine to form even higher-fitness schemata) can speed up GAs' searching process. Forrest et al. (1993) found that if some intermediate stepping stones are much fitter than the primitive components, then hitchhiking problem becomes more severe and thus premature convergence slows down the discovery of some necessary schemata.

We summarize the results from two versions of the BBDS and those reported in (Forrest, & Mitchell, 1993) in Table 2.

Table 2: Experimental Results on R_2

	Function Evaluations to Optimum	
	Mean	Standard Deviation
GA	73563	40115
SAHC	>256,000	–
NAHC	>256,000	–
RMHC	6551	2998
BBDS v. 1	975	314
BBDS v. 2	901	309

This table shows that the GA indeed performed worse on R_2 than on R_1. However, under the same random seed, the BBDS has the exact performance on R_1 and R_2, indicating that stepping stones do not have any negative impact on the search power of the BBDS. This is because the BBDS essentially takes into account only fitness increase or decrease, not the amount of relative fitness difference. Thus any extra fitness difference contributed by the stepping stones of R_2 does not affect the performance of the BBDS.

Since Royal Road functions are non-deceptive, such landscapes allow BBDS to exhibit the maximum capability to extract information concerning the building blocks whenever they come in sight on the string; thus the global optimum can be reached very quickly. Several empirical results obtained so far indeed show that BBDS significantly outperforms RMHC and traditional GAs on such non-deceptive fitness landscapes.

Although the ISGA needs to employ only a string to attain the optimum of Royal Road functions, this single individual can be fooled by any deceptive schemata. If this is the case, the ISGA with population size one is certainly not enough for attaining gratifying performance. In the next subsection, we present the experimental results of the ISGAs with larger population size on another benchmark test function which bears this fitness landscape feature.

4.2 PERFORMANCE ON 30-BIT BOUNDED DECEPTION PROBLEM

The problems of bounded deception designed by Goldberg et al. (1989) were to investigate the performance of GAs on deceptive functions in which low-order, highly-fit schemata mislead GAs away from global optima and toward the complement of the global optimum. One example of this class is an order-3 fully deceptive function as defined in Table 3.

Table 3: A fully deceptive, order-3 problem

bit	value	bit	value
111	30	100	14
101	0	010	22
110	0	001	26
011	0	000	28

On this 3-bit, deceptive problem, calculations of the average fitness of schema show that GAs are likely to be led toward the complement of the global optimum, i.e., 000, instead of toward the global optimum, 111. To demonstrate the effect of this deception on the search power of GAs, Goldberg et al. (1989) designed a 30-bit deceptive function, **E10**, which is composed of ten consecutive blocks of this 3-bit deceptive function.

In contrast to the non-deceptive feature of Royal Road functions, it is apparent that this 30-bit deceptive function imposes enough difficulty for GAs to arrive at the global maximum (1,1,...,1).

We performed 50 runs of the ISGAs with mate selection schemes A and B on **E10**, based on the second version of BBDS, for population size 40 and 80. The end criterion of the BBDS in this case is the moment that the length of candidate schemata reaches the length of the string. After all the strings reach the end criterion of the BBDS, the independent sampling phase stops and the breeding phase gets started. We then measure the number of function evaluations required to find the global optimum and the results are shown in Table 4 (the standard deviation is given in the parentheses).
Notice that the ISGA with mate selection scheme B requires fewer function evaluations than that with scheme A. These results indicate that the ISGA with mate selection B indeed outperforms that with scheme A, which is consistent with the results obtained in (Huang, 2001).

Table 4: Experimental Results on **E10**

	Function Evaluations to Optimum	
	Population size 40	Population size 80
Scheme A	11786 (11034)	16794 (12175)
Scheme B	9582 (7889)	11122 (5614)

To see how the BBDS version 2 searches this fitness landscape, we show that after the independent sampling phase, only (111) or (000) will emerge at each block, and the probabilities are $\frac{1}{4}$, and $\frac{3}{4}$, respectively (please see Appendix). Thus for a population of 40 individuals, the probability that the population contains no building block (111) at a building-block location is only $(\frac{3}{4})^{40} \approx 1.0 \times 10^{-5}$; and for a population size 80, the probability is $(\frac{3}{4})^{80} \approx 1.0 \times 10^{-10}$. Therefore these two population sizes serve for enough underlying building blocks to construct the global optimum.

To compare total function evaluations used by the two phases in the ISGAs, we show the results in Table 5, where the first element corresponds to the evaluations spent in the independent sampling phase and the second corresponds to that in the breeding phase. In this table, it is clear that scheme B has higher efficiency of exploiting the building blocks found in the independent sampling phase to construct the global optimum.

Table 5: Total Function Evaluations in Two Phases

	Population Size 40	Population Size 80
Scheme A	(1159,10627)	(2322,14472)
Scheme B	(1160,8422)	(2320,8802)

To demonstrate the capability of the ISGAs, we compare their performance (based on population size 40) with that of several different types of GAs: a mGA (Goldberg, Korb, & Deb, 1989), a modified mGA (Goldberg, Deb, Kargupta, & Harik, 1993), a Breeder GA (BGA) (Mühlenbein & Schlierkamp-Voosen, 1993), and two versions of PGA (2pc-wohc, two-point cyclic crossover without hill-climbing, and 2pc-nahc, two-point cyclic crossover with next ascent hill-climbing) (Mühlenbein, 1991). We also ran a simple serial GA over 50 runs (based on a binary tournament selection–with probability 1.0 the fitter of the two randomly sampled individuals is chosen, mutation rate 0.005, two-point crossover rate 0.7, population size 80, and maximum function evaluations 50000 for each run). The experimental results of the ISGAs and other GAs reported can be summarized in Table 6 from which we can see that the ISGAs significantly outperform other GAs.

Table 6: Performance of Several Types of GAs

Mean Function Evaluations to Optimum	
ISGA (Scheme A)	11786
ISGA (Scheme B)	9582
mGA	40600
Modified mGA	26650
BGA	16000
2pc-wohc PGA	21398
2pc-nahc PGA	40500
Serial GA	0 runs reached optimum

5 DISCUSSIONS

One issue that also concerns us is the effect of population size. In Table 4, we see that the ISGA with larger population size has worse performance than with smaller population size. This can be more clearly seen in Table 5. In this table, for the same population size, the function evaluations required in the independent sampling phase for two schemes are almost the same, yet in the breeding phase the difference between two schemes for population size 80 is larger than that for population size 40.

This is the opposite of what has been obtained for simple serial GAs in (Huang, 2001), in which larger population size reduces the performance difference between these two mate selection schemes. So far, the answer for this seeming paradox has not yet been obtained, but we can conjecture that since the ISGAs implement independent sampling of building blocks in the first phase to a maximum degree, they may generate too diverse a population if the population size is large enough. This in turn slows down the evolution of the population in the breeding phase.

How does population diversity affect the searching process for different goals, such as finding a global optimum or forming speciation? From the discussion above, it is clear that larger population size is not always advantageous and we will manage to investigate the relationship between diversity and finding a global optimum in the near future.

6 CONCLUSIONS

In this paper we first present an exploratory method (BBDS) to show how the searching speed of individuals can be improved. Through explicitly acquiring relevant knowledge of candidate building blocks, BBDS outperformed several representative hill-climbing algorithms on non-deceptive Royal Road functions. Then a new class of GAs based on BBDS, i.e., ISGAs, is proposed. In the first phase of ISGAs, implicit parallelism is nicely realized by allowing each individual to accomplish independent building-block sampling to suppress hitchhiking; thus the population is expected to carry diverse promising schemata. Afterwards, with one mate selection scheme that allows individuals to actively choose their mating partners, the efficacy of crossover is enhanced and the ISGAs have been shown to outperform several different GAs on a benchmark test function that is full of deception.

7 FUTURE WORK

Much work remains to be done. The author is now testing the capability of ISGAs on more complicated fitness landscapes, such as the hyperplane defined functions (HDFs) designed by Holland (2000), which parameterize fitness landscapes to encompass features such as hierarchy, poor-linkage, potholes, hills, badlands, ridges, etc. Other research lines are to examine in more detail the impact of mutation and the difference between two-phased and traditional (one-phased) GAs. Afterwards, our hope is to extend the single two-phased procedure to successive two-phased procedures over the course of evolution, i.e., iterating the two phases during the whole run. In addition, other versions of BBDS are worth investigating so that building-block detecting is more effectively implemented. Moreover, a theoretical foundation is needed to explain the whys and wherefores of the excellent performance of ISGAs, and finally our goal is to extend the application of ISGAs to real problems.

Acknowledgments

The author would like to thank Rick Riolo, John Holland for their advice, and Bob Lindsay, Ted Belding, Leeann Fu, and Tom Bersano-Begey for their comments and suggestions.

Appendix

For BBDS version 2, let us start with an initial 3-bit sub-string, for example, at (101) (fitness = 0). Then the first bit is flipped to 0, which causes the fitness to increase to 26; thus this bit's value must be replaced by 0 and its bit-position and new bit-value (i.e., 0) are recorded as the first gene of the candidate schema. After this function evaluation the candidate schema is

(0xx) ("x" represents a not-yet-tested gene) and this sub-string is now (001).

Then under the instruction of step 2, we have to go back to the first bit again and flip it. But since this gene has been already recorded, we do not flip this bit; instead the next bit is temporarily flipped to 1 for test, leading the fitness to decrease to 0. Thus the original value of the sub-string's second bit is kept (i.e., 0) and we record this bit's position and original value as the second gene of the candidate schema, and this candidate schema is now (00x).

Then the BBDS goes to the third bit and flip it. We thus obtain (000), which leads to the fitness of 28. Thus we record this bit's position and new value as the third gene of the candidate schema, which is now (000).

Since the length of this candidate schema reaches the length of this 3-bit sub-string, we stop here.

The process can be symbolized in the following:

$(101) \longrightarrow (0xx) \longrightarrow (00x) \longrightarrow (000).$

Analogously, the collected candidate schemata for the starting points at (100), (110), (010), (001), and (000) will be (000); and those for (011) and (111) will be (111).

We summarize the results as follows:

$(000) \longrightarrow (0xx) \longrightarrow (00x) \longrightarrow (000);$

$(001) \longrightarrow (0xx) \longrightarrow (00x) \longrightarrow (000);$

$(010) \longrightarrow (0xx) \longrightarrow (00x) \longrightarrow (000);$

$(100) \longrightarrow (0xx) \longrightarrow (00x) \longrightarrow (000);$

$(101) \longrightarrow (0xx) \longrightarrow (00x) \longrightarrow (000);$

$(110) \longrightarrow (0xx) \longrightarrow (00x) \longrightarrow (000);$

$(011) \longrightarrow (1xx) \longrightarrow (11x) \longrightarrow (111);$

$(111) \longrightarrow (1xx) \longrightarrow (11x) \longrightarrow (111).$

References

Cantú-Paz, E. (1997). *A survey of parallel genetic algorithms* (IlliGAL Report No. 97003). Urbana, IL: University of Illinois at Urbana-Champaign.

Das, R., & Whitley, L. D. (1991). The only challenging problems are deceptive: Global search by solving order 1 hyperplanes. *Proceedings of the Fourth International Conference on Genetic Algorithms*, 166-173.

Forrest, S., & Mitchell, M. (1993). Relative building block fitness and the building block hypothesis. *Foundations of Genetic Algorithms 2*, 109–126.

Goldberg, D. E. (1989). *Genetic Algorithms in search, Optimization, and Machine Learning.* Reading, MA: Addison Wesley.

Goldberg, D. E., Deb, K., & Korb, B. (1991). Don't worry, be messy. *Proceedings of the Fourth International Conference on Genetic Algorithms*, 24-30.

Goldberg, D. E., Deb, K., Kargupta, H., & Harik, G. (1993). Rapid, Accurate optimization of difficult problems using fast messy genetic algorithms. *Proceedings of the Fifth International Conference on Genetic Algorithms*, 56-64.

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

Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems.* University of Michigan Press. (Second edition: MIT Press, 1992.)

Holland, J. H. (2000). Building blocks, cohort genetic algorithms, and hyperplane-defined functions. *Evolutionary Computation* 8(4):373-391.

Huang, C.-F. (2001). An Analysis of Mate Selection in Genetic Algorithms. *Proc. of 2001 Genetic and Evolutionary Computation Conference (GECCO-2001)*, submitted.

Mitchell, M., Forrest, S., & Holland, J. H. (1992). The royal road for genetic algorithms: Fitness landscapes and GA performance. *Toward a Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life*, 245–254.

Mitchell, M., Holland, J. H., & Forrest, S. (1994). When will a genetic algorithm outperform hill climbing? *Advances in Neural Information Processing Systems 6*, 51–58.

Mitchell M.(1996). *An introduction to Genetic Algorithms.* Cambridge, MA: MIT Press.

Mühlenbein, H. (1991). Evolution in time and space – The parallel genetic algorithm. *Foundations of Genetic Algorithms*, 316–337.

Mühlenbein, H. (1992). How genetic algorithms really work I: Mutation and hillclimbing. *Proceedings of Parallel Problem Solving from Nature 2*, 15–26.

Mühlenbein, H., & Schlierkamp-Voosen, D. (1993). *Optimal interaction of mutation and crossover in the breeder genetic algorithm.* Technical Report 93-042, GMD.