Missouri Estimation of Distribution Algorithms Laboratory

Performance of Network Crossover on NK Landscapes and Spin Glasses

Mark Hauschild and Martin Pelikan

MEDAL Report No. 201006

March 2010

Abstract

This paper describes a network crossover operator based on knowledge gathered from either prior problem-specific knowledge or linkage learning methods such as estimation of distribution algorithms (EDAs). This operator can be used in a genetic algorithm (GA) to incorporate linkage in recombination. The performance of GA with network crossover is compared to that of GA with uniform crossover and the hierarchical Bayesian optimization algorithm (hBOA) on 2D Ising spin glasses, NK landscapes, and SK spin glasses. The results are analyzed and discussed.

Keywords

Genetic Algorithms, Crossover, Replacement, Hierarchical BOA, efficiency enhancement, model structure, model complexity, estimation of distribution algorithms

Missouri Estimation of Distribution Algorithms Laboratory (MEDAL)
Department of Mathematics and Computer Science
University of Missouri–St. Louis
One University Blvd., St. Louis, MO 63121
E-mail: medal@cs.umsl.edu
WWW: http://medal.cs.umsl.edu/

Performance of Network Crossover on NK Landscapes and Spin Glasses

Mark Hauschild
Missouri Estimation of Distribution Algorithms Laboratory (MEDAL)
Dept. of Math and Computer Science, 320 CCB
University of Missouri at St. Louis
One University Blvd., St. Louis, MO 63121
mwh308@umsl.edu

Martin Pelikan
Missouri Estimation of Distribution Algorithms Laboratory (MEDAL)
Dept. of Math and Computer Science, 320 CCB
University of Missouri at St. Louis
One University Blvd., St. Louis, MO 63121
pelikan@cs.umsl.edu

Abstract

This paper describes a network crossover operator based on knowledge gathered from either prior problem-specific knowledge or linkage learning methods such as estimation of distribution algorithms (EDAs). This operator can be used in a genetic algorithm (GA) to incorporate linkage in recombination. The performance of GA with network crossover is compared to that of GA with uniform crossover and the hierarchical Bayesian optimization algorithm (hBOA) on 2D Ising spin glasses, NK landscapes, and SK spin glasses. The results are analyzed and discussed.

Keywords: Genetic Algorithms, Crossover, Replacement, Hierarchical BOA, efficiency enhancement, model structure, model complexity, estimation of distribution algorithms.

1 Introduction

It has been argued that to solve many difficult classes of problems in a robust and scalable manner, variation operators of genetic algorithms (GAs) must respect linkages between variables (Goldberg, 2002). Unfortunately, conventional variation operators of GAs often break up important linkages (Goldberg, 2002). To help remedy this problem, competent linkage learning GAs such as estimation of distribution algorithms (EDA) (Baluja, 1994; Mühlenbein & Paaß, 1996; Larrañaga & Lozano, 2002; Pelikan, Goldberg, & Lobo, 2002) have been developed. EDAs work by building a probabilistic model of promising solutions and sampling new candidate solutions from the build model. While EDAs have many advantages over standard GAs (Larrañaga & Lozano, 2002; Pelikan, Sastry, & Cantú-Paz, 2006), the model building is often computationally intensive, and much of the work in EDAs focused on alleviating this bottleneck through efficiency enhancement techniques (Cantú-Paz, 2000; Schwarz & Ocenasek, 2000; Pelikan, 2005).

Often, though, practitioners have prior information about the problem being solved. For example, in graph-based problems we are implicitly given a guide to the strongest dependencies in a

problem. In addition, linkage learning algorithms can also be used to find the structure of the problem by mining their generated models (Hauschild, Pelikan, Sastry, & Goldberg, 2008; Hauschild & Pelikan, 2009). The key question is, how do we exploit this information in practice?

One way to do this is to modify the crossover operator in a GA to better respect the strongest linkages in the underlying problem structure. This paper discusses a network crossover operator that works with a user-specified network graph to determine which bits are exchanged. We compare the performance of GA with network crossover to that of GA with uniform crossover and the hierarchical Bayesian Optimization Algorithm (Pelikan & Goldberg, 2001; Pelikan, 2005), which is one of the most powerful linkage learning GAs. All algorithms are tested on a broad range of problems of different structure known to be hard for standard evolutionary algorithms.

The paper is organized as follows. Section 2 describes the network crossover operator used in this paper. Section 3 outlines the algorithms tested. Section 4 describes the test problems used in this paper. Section 5 presents the experimental results. Finally, section 6 summarizes and concludes the paper.

2 Network Crossover

While uniform crossover is effective at solving many problems (Goldberg, 1989), it can often become highly ineffective because it processes each bit independently of others and is unable to consider dependencies between problem variables (Thierens, 1999). In many cases it is desirable that crossover preserves combinations of values of variables that depend on each other while it effectively exploits independencies of other variables to ensure effective mixing (Thierens, 1999). The operator in this paper tries to do this by using a fixed network of assumed dependencies between problem variables.

Any two-parent crossover operator starts by creating a mask, which defines what bits to exchange. For example, in uniform crossover, the mask is generated at random by setting each bit to 0 or 1 with equal probabilities. The parents exchange their k-th bit if and only if the mask contains a 1 in position k. In this paper we show how we can create a specialized mask that respects problem structure encoded in a network of dependencies.

The network crossover requires an $n \times n$ incidence matrix G that specifies the strongest dependencies between bits. Specifically, denoting the element in the ith row and jth column of G by $G_{i,j}$, bits in locations i and j depend on each other if $G_{i,j} = 1$, whereas they are independent if $G_{i,j} = 0$. Note that a similar representation of dependencies in a problem is used in the dependency structure matrix GA (DSM-GA) (Yu, Goldberg, Sastry, Lima, & Pelikan, 2009). While the network does require information from the practitioner, it does not require in-depth knowledge of the strength of interactions. For graph problems, such as graph-bipartitioning or graph coloring, this graph G is inherent to the problem definition. For problems like MAXSAT and Ising Spin glasses, it is straightforward to specify such a structure from the additive decomposition of these problems. One may also run an EDA on trial instances of the problem to learn promising network structures as suggested in (Hauschild, Pelikan, Sastry, & Goldberg, 2008). The key point to remember when constructing G is that it is not necessary to specify the entire problem structure, but only the strongest dependencies or linkages.

Given the network G, a crossover mask m is then built as follows. First a random bit i is selected. This bit is then added to the crossover mask by setting $m_i = 1$. Then a randomized breadth-first search is performed on the network G, setting each corresponding bit in m to 1, until m reaches the desired size. If the breadth-first search ends before the desired size is reached, an additional random starting point is selected and the process repeated. This results in a crossover mask that should most often disrupt bits that are not connected in the underlying network defined

by G.

The idea of using a network of dependencies to modify two-parent recombination operators is not new and has been inspired by past work (Drezner & Salhi, 2002; Drezner, 2003; Stonedahl, Rand, & Wilensky, 2008). Our variant of network crossover most closely resembles the work by Stonedahl (Stonedahl, Rand, & Wilensky, 2008) where the mask was built using a random walk through the network structure. On the other hand, we use the breadth-first search to emphasize the short range dependencies, since these dependencies were found to be strongest in prior work (Hauschild, Pelikan, Sastry, & Goldberg, 2008). However, our work is *not* an attempt to improve on this operator, but rather to perform a systematic study of network crossover on a broad range of problems.

3 Tested Algorithms

The genetic algorithm (GA) (Holland, 1975; Goldberg, 1989) evolves a population of candidate solutions typically represented by binary strings of fixed length. The initial population is generated at random according to the uniform distribution over all binary strings. Each iteration starts by selecting promising solutions from the current population; in this work we use binary tournament selection without replacement. New solutions are created by applying variation operators to the population of selected solutions. These new candidate solutions are then incorporated into the population using a replacement operator. The run is terminated when either when the global optimum has been found or after a maximum number of iterations.

Instead of using crossover and mutation to create new candidate solutions, hBOA learns a Bayesian network with local structures (Pelikan & Goldberg, 2001; Pelikan, 2005) as a model of the selected solutions and generates new candidate solutions from the distribution encoded by this model. Using Bayesian networks to generate new candidate solutions ensures effective processing of partial solutions in the class of nearly decomposable and hierarchical problems (Pelikan, 2005), including many problems that cannot be efficiently solved with standard two-parent crossover operators (Pelikan & Goldberg, 2001; Pelikan, 2005).

In both GA and hBOA, a deterministic hill climber (DHC) was incorporated to improve performance. DHC takes a candidate solution represented by an n-bit binary string on input. In each iteration, DHC evaluates all possible one-bit flips and chooses the one that leads to the maximum improvement of solution quality. DHC is terminated when no single-bit flip improves solution quality and the solution is thus locally optimal. Initially, experiments without DHC were considered, but for all settings tested, DHC dramatically improved performance.

4 Test Problems

In concatenated traps of order 5 (trap-5) (Ackley, 1987; Deb & Goldberg, 1991), the input string is partitioned into independent groups of 5 bits. This partitioning is unknown to the algorithm and it does not change during the run. A 5-bit fully deceptive trap function is applied to each group of 5 bits and the contributions of all trap functions are added to form the fitness. The contribution of each group of 5 bits is given by

$$trap_5(u) = \begin{cases} 5 & \text{if } u = 5 \\ 4 - u & \text{otherwise} \end{cases}, \tag{1}$$

where u is the number of 1s in the input string of 5 bits. The task is to maximize the function. An n-bit trap5 function has one global optimum in the string of all 1s and $(2^{n/5} - 1)$ other local optima.

Traps of order 5 necessitate that all bits in each group are treated together, because statistics of lower order are misleading.

An NK fitness landscape (Kauffman, 1989) is fully defined by the following components: (1) The number of bits, n, (2) the number of neighbors per bit, k, (3) a set of k neighbors $\prod(X_i)$ for the i-th bit, X_i for every $i \in \{0, \ldots, n-1\}$, and (4) a subfunction f_i defining a real value for each combination of values of X_i and $\prod(X_i)$ for $i \in \{0, \ldots, n-1\}$, with each subfunction defined as a lookup table.

In this paper we consider two classes of NK landscape instances: (1) Unrestricted NK landscapes and (2) nearest-neighbor NK landscapes. In unrestricted NK landscapes, thet set of k neighbors for each string position X_i is selected at random according to the uniform distribution of all subsets of k bits. Then, a lookup table defining each f_i is generated using the uniform distribution over $[0, 1)$. In this paper we consider unrestricted NK landscapes with $k = 5$; the considered class of NK landscapes is NP-complete (Wright, Thompson, & Zhang, 2000). In nearest-neighbor NK landscapes, the bits are arranged on a circle and the neighbors of each bit are restricted to the k bits that follow this bit in the circle. The bit positions are shuffled randomly in order to eliminate tight linkage. Nearest-neighbor NK landscapes are solvable in polynomial time. The algorithm used to solve nearest-neighbor NK instances is based on refs. (Pelikan, Sastry, Butz, & Goldberg, 2006; Pelikan, Sastry, Goldberg, Butz, & Hauschild, 2009). The branch and bound algorithm used to solve unrestricted NK landscapes is based on ref. (Pelikan, 2008).

Ising spin glasses (Mezard, Parisi, & Virasoro, 1987) are prototypical models for disordered systems. A simple model to describe a finite-dimensional Ising spin glass is typically arranged on a regular 2D or 3D grid where each node i corresponds to a spin s_i and each edge $\langle i, j \rangle$ corresponds to a coupling between two spins s_i and s_j. Each edge has a real value $J_{i,j}$ associated with it that defines the relationship between the two connected spins. To approximate the behavior of the large-scale system, periodic boundary conditions are often used that introduce a coupling between the first and the last elements in each row along each dimension.

For the classical Ising model, each spin s_i can be in one of two states: $s_i = +1$ or $s_i = -1$. Given a set of coupling constants $J_{i,j}$, and a configuration of spins C, the energy can be computed as

$$E(C) = -\sum_{\langle i,j \rangle} s_i J_{i,j} s_j \ , \tag{2}$$

where the sum runs over all couplings $\langle i, j \rangle$. The task is to find a spin configuration for a given set of coupling constants that minimizes the energy of the spin glass. The states with minimum energy are called *ground states*. Here we consider the $\pm J$ spin glass, where each spin-spin coupling is set randomly to either $+1$ or -1 with equal probability. The ground states of the instances were obtained from the Spin Glass Ground State Server at the Univ. of Cologne (Spin Glass Ground State Server, 2004).

The Sherrington-Kirkpatrick (SK) spin glass (Kirkpatrick & Sherrington, 1978) is described by a set of spins s_i and a set of couplings $J_{i,j}$ between all pairs of spins. Thus, unlike the 2D spin glass model, the SK model does not limit the range of spin-spin interactions to only neighbors in the lattice. The goal is to find ground states for the given coupling constants.

Here we consider two types of random instances of the SK model. The first type uses couplings generated from the Gaussian distribution with zero mean and unit variance $N(0, 1)$. The second set of SK instances are one-dimensional spin glasses with power-law interactions (Katzgraber, 2008) where the spins are arranged equidistantly and numbered counterclockwise on a circle with circumference n. While all spins interact with each other, the interactions between spins located further from each other are weaker. The effects of distance are controlled with a parameter σ; in this paper

we examine instances with $\sigma = 2$, which gives the model short ranged behavior. Both types of SK spin glasses are NP-complete (Barahona, 1982).

To find guaranteed ground states, a branch-and-bound algorithm adopted from ref. (Hartwig, Daske, & Kobe, 1984) was used for the smaller instances. For larger systems, the population-doubling approach from ref. (Pelikan, Katzgraber, & Kobe, 2008) was used.

5 Experiments

5.1 Experimental Setup

For trap-5, problem sizes of $n = 100$ to 300 with step 10 were examined and G was constructed by setting $G_{i,j} = 1$ if and only if i and j were in the same trap partition. For nearest-neighbor NK landscapes, problem sizes of $n = 30$ to 210 with step 30 were considered and for unrestricted NK landscapes, problem sizes of $n = 20$ to 38 with step 2 were used. For NK landscapes of all types, $k = 5$ and G was constructed by setting $G_{i,j} = 1$ if and only if i and j were neighbors. For 2D Ising spin glasses, problem sizes of $n \in \{256, 324, 400, 484, 576\}$ were examined and G was constructed by setting $G_{i,j} = 1$ if and only if i and j were neighbors in the underlying spin glass lattice.

For one-dimensional spin glasses with power-law interactions, problem sizes of $n \in \{100, 150, 200, 300\}$ were examined with $\sigma = 2$. For the standard SK spin glass, only two problem sizes were considered, $n = 300$ and 400. For all SK instances, G was constructed by connecting each node i to a fixed number m of spins with the strongest magnitude of couplings with i. For example, for $m = 4$, i was connected to four other spins that have the strongest couplings with i. For the experiments with network crossover, $m \in \{1, \ldots, 7\}$ were examined.

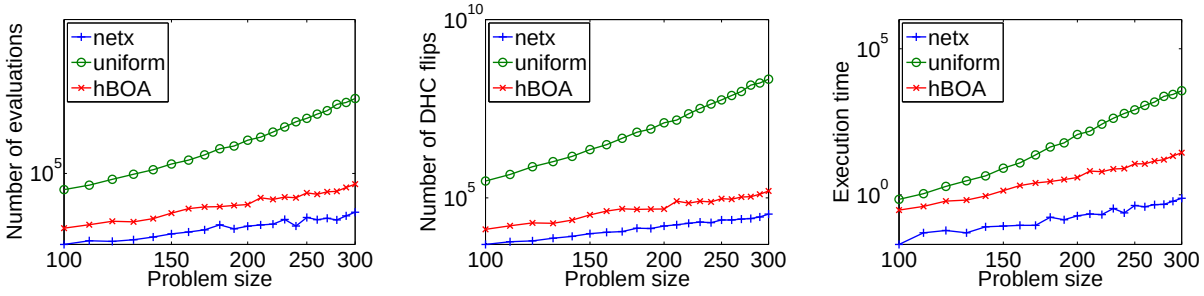
hBOA and GA with both uniform crossover and network crossover were applied to all problem instances. For GA runs, bit-flip mutation was used with a probability $p_m = 1/n$. The probability of crossover was set to $p_c = 0.6$. To effectively maintain population diversity, new solutions were incoroporated into the old population using restricted tournament replacement (RTR) (Harik, 1995). The window size in RTR was set as $w = \min\{n, N/5\}$ where n is the problem size and N is the population size as suggested in ref. (Pelikan, 2005).

For all problem types except for trap-5, 1000 random instances for each problem size were tested. Bisection (Pelikan, 2005) was used to determine the minimum population size to ensure convergence to the global optimum in 10 out of 10 independent runs for each instance. For trap-5, bisection was run 10 times to obtain more reliable results. Each run was terminated when the global optimum had been found or when the maximum number of generations $n \times 4$ had been reached. Three results were compared for all solved instances: the number of evaluations, the number of steps of DHC, and the total execution time in seconds.

5.2 Experimental Results

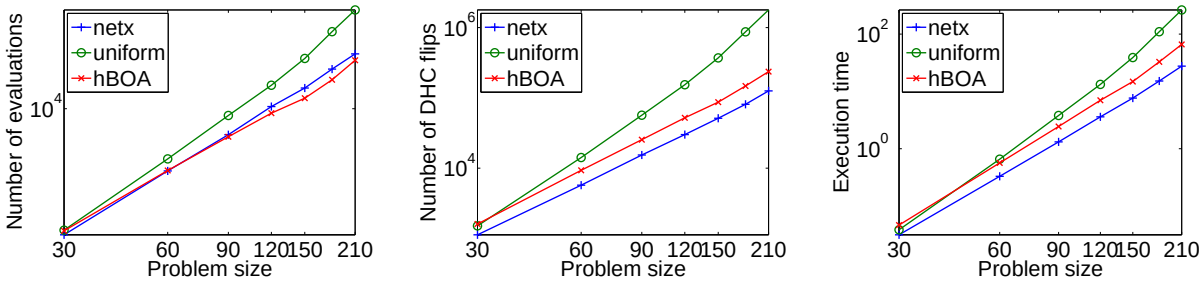
The number of evaluations, the number of DHC flips, and the execution time to solve trap-5 is shown in figure 1. GA with network crossover clearly performed the best. This is to be expected as the network crossover will disrupt much fewer trap partitions than other algorithms and, unlike hBOA, network crossover is given the correct problem decomposition on input. hBOA scales similarly as GA with network crossover, indicating that hBOA can learn the problem decomposition scalably. GA with uniform crossover performs very poorly.

Figure 2 shows the performance of the three compared algorithms on nearest-neighbor NK landscapes. hBOA is the best performing algorithm with respect to the number of evaluations.



(a) Number of evaluations (b) Number of DHC flips (c) Execution time

Figure 1: Results on trap-5.



(a) Number of evaluations (b) Number of DHC flips (c) Execution time

Figure 2: Results on nearest-neighbor NK landscapes with $k = 5$ neighbors.

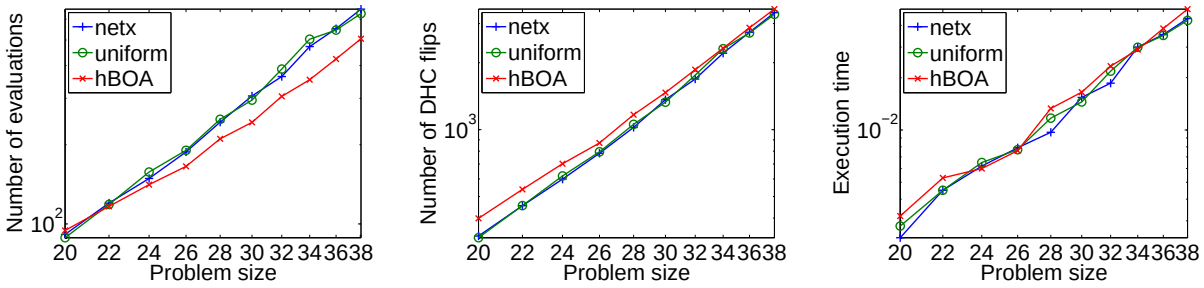
However, GA with network crossover is best with respect to the total execution time and the number of DHC steps. GA with uniform crossover is again the worst performer, scaling much worse than the other algorithms.

The results on unrestricted NK landscapes are shown in figure 3. With respect to the number of evaluations, hBOA is clearly the best performing algorithm and GA performs approximately the same regardless of the crossover operator used. With respect to the number of local search steps, while hBOA starts off performing poorly, it scales better than GA with uniform or network crossover. With respect to the total execution time, the results are mixed and all algorithms seem to scale about the same.

The performance of the three compared algorithms on 2D Ising spin glasses is shown in Figure 4. We see that hBOA strongly outperforms the other algorithms with respect to the number of evaluations and local search steps. However, due to less overhead, GA with network crossover has very similar execution times, although it scales slightly worse compared to hBOA as the problem size increases. GA with uniform crossover is shown to have quite poor performance.

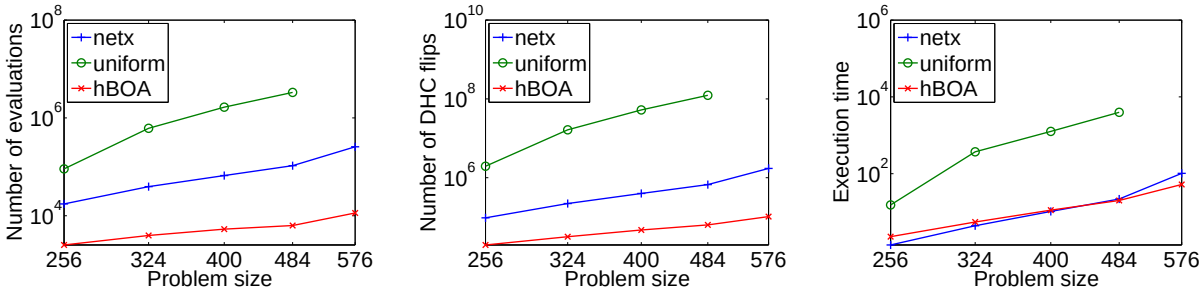
Selected experimental results on the one-dimensional SK spin glasses are shown in figure 5. We see that GA with uniform crossover is the worst performing algorithm. GA with network crossover with $m = 3$ (each node in the network is connected to 3 other bits), performs worse than hBOA. For $m = 5$, while GA with network crossover is the best performer in terms of the execution time for the smaller problem sizes, hBOA shows superior scaling. The results for other values of m were omitted as $m = 5$ was the best performing value found.

Table 1 shows the results of the three compared algorithms on unrestricted SK spin glasses. For network crossover, the best value of m is used for each value of n. In contrast to all previous results, for SK spin glasses hBOA is the worst performing algorithm with respect to the growth of the execution time with problem size. For $n = 300$ GA with network crossover ($m = 3$) performed



(a) Number of evaluations (b) Number of DHC flips (c) Execution time

Figure 3: Results on unrestricted NK landscapes with $k = 5$ neighbors.



(a) Number of evaluations (b) Number of DHC flips (c) Execution time

Figure 4: Results on 2D Ising spin glasses.

best, whereas for $n = 400$ GA with uniform crossover performed best.

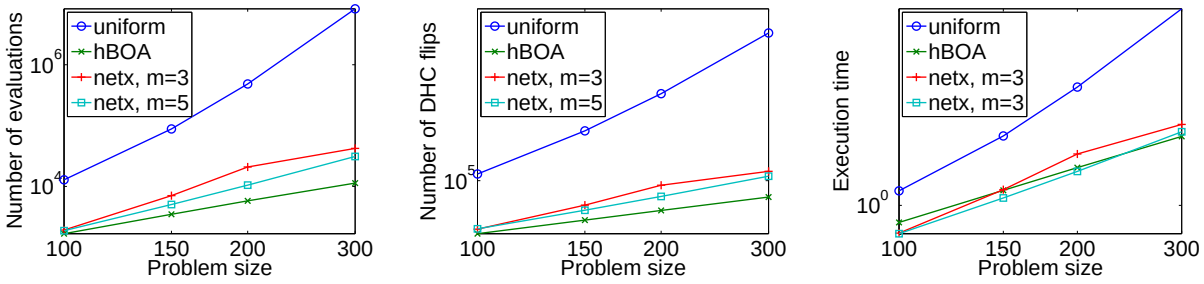
6 Summary and Conclusions

This paper described a network crossover operator which can be used to incorporate problem-specific knowledge about dependencies between problem variables into a GA. Performance of GA with the described crossover operator was then compared to GA with uniform crossover and the hierarchical BOA (hBOA) on a number of challenging problem instances from several problem classes.

On most problems, hBOA and GA with network crossover outperformed GA with uniform crossover. This result is not surprising because all test problems were additively decomposable and it has been argued that to effectively solve additively decomposable problems, recombination must often exploit problem structure to ensure proper mixing and juxtaposition of important partial solutions (Thierens, 1999; Goldberg, 2002). GA with uniform crossover was slightly better than other algorithms only on one problem, which contains little structure. The more regular the structure of the problem was, the better hBOA and GA with network crossover performed.

Network crossover received information about the structure of the problem on input. That is why it is expected that GA with network crossover would typically outperform hBOA, which must learn the problem structure on its own. Nonetheless, the results were somewhat surprising because in most cases, hBOA showed superior scaling of execution times with problem size. This is in spite of the fact that hBOA is given no information about the structure of the problem on input and it is required to obtain such information itself.

While there is no doubt that using information about the structure of the problem can help in solving many difficult optimization problems more efficiently, learning the problem structure



(a) Number of evaluations (b) Number of DHC flips (c) Execution time

Figure 5: Results on the one-dimensional SK spin glass with power-law interactions, $\sigma = 2$.

Table 1: Results on the general SK spin glasses. The table includes the best found value of m for network crossover.

Size	Algorithm	Evaluations	Execution time	Number of DHC flips
300	hBOA	6,615	27.1	284,482
300	GA with uniform crossover	11,579	13.5	53,0150
300	GA with netx, $m = 3$	9,028	10.8	45,0834
400	hBOA	30,021	215.5	1,236,547
400	GA with uniform crossover	26,601	56.8	1,767,551
400	GA with netx, $m = 3$	32,672	66.7	2,077,639

is a computationally intensive task. That is why it should certainly be beneficial if one could incorporate prior problem-specific knowledge into the GA if such knowledge is available, and use specialized crossover operators such as the network crossover discussed here. However, the mixed results presented in this paper show that this is certainly not a straightforward task, and that there are several questions that must be addressed in future research in this area.

Most importantly, future research in this area should examine whether the mixed results are a consequence of using two-parent recombination as opposed to gene-pool recombination, or whether the main reason for these results was in the specific dependencies used for constructing masks in network crossover. Variations of network crossover should also be explored, using different approaches for constructing network masks. Finally, GA with network crossover should be tested on additional classes of problems.

Acknowledgments

This project was sponsored by the National Science Foundation under CAREER grant ECS-0547013, by the Air Force Office of Scientific Research, Air Force Materiel Command, USAF, under grant FA9550-06-1-0096, and by the University of Missouri in St. Louis through the High Performance Computing Collaboratory sponsored by Information Technology Services, and the Research Award and Research Board programs.

The U.S. Government is authorized to reproduce and distribute reprints for government purposes notwithstanding any copyright notation thereon. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation, the Air Force Office of Scientific Research, or the U.S. Government. Some experiments were done using the hBOA software developed by Martin Pelikan and David E. Goldberg at the University of Illinois at Urbana-Champaign and most experiments were performed on the Beowulf cluster maintained by ITS at the University of Missouri in

St. Louis.

References

Ackley, D. H. (1987). An empirical study of bit vector function optimization. *Genetic Algorithms and Simulated Annealing*, 170–204.

Baluja, S. (1994). *Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning* (Tech. Rep. No. CMU-CS-94-163). Pittsburgh, PA: Carnegie Mellon University.

Barahona, F. (1982). On the computational complexity of Ising spin glass models. *Journal of Physics A: Mathematical, Nuclear and General*, *15*(10), 3241–3253.

Cantú-Paz, E. (2000). *Efficient and accurate parallel genetic algorithms*. Boston, MA: Kluwer.

Deb, K., & Goldberg, D. E. (1991). *Analyzing deception in trap functions* (IlliGAL Report No. 91009). Urbana, IL: University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory.

Drezner, Z. (2003). A new genetic algorithm for the quadratic assignment problem. *INFORMS Journal on Computing*, *15*(3), 320–330.

Drezner, Z., & Salhi, S. (2002). Using hybrid metaheuristics for the one-day and two-way network design problem. *Naval Research Logistics*, *49*(5), 449–463.

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization, and machine learning*. Reading, MA: Addison-Wesley.

Goldberg, D. E. (2002). *The design of innovation: Lessons from and for competent genetic algorithms*. Kluwer.

Harik, G. R. (1995). Finding multimodal solutions using restricted tournament selection. *International Conf. on Genetic Algorithms (ICGA-95)*, 24–31.

Hartwig, A., Daske, F., & Kobe, S. (1984). A recursive branch-and-bound algorithm for the exact ground state of Ising spin-glass models. *Computer Physics Communications*, *32*(2), 133 – 138.

Hauschild, M., Pelikan, M., Sastry, K., & Goldberg, D. E. (2008). Using previous models to bias structural learning in the hierarchical BOA. *Genetic and Evolutionary Comp. Conf. (GECCO-2008)*, 415–422.

Hauschild, M. W., & Pelikan, M. (2009). Intelligent bias of network structures in the hierarchical boa. (pp. 413–420). New York, NY, USA: ACM.

Holland, J. H. (1975). *Adaptation in natural and artificial systems*. Ann Arbor, MI: University of Michigan Press.

Katzgraber, H. G. (2008). Spin glasses and algorithm benchmarks: A one-dimensional view. *Journal of Physics: Conf. Series*, *95*(012004).

Kauffman, S. (1989). Adaptation on rugged fitness landscapes. In Stein, D. L. (Ed.), *Lecture Notes in the Sciences of Complexity* (pp. 527–618). Addison Wesley.

Kirkpatrick, S., & Sherrington, D. (1978, Jun). Infinite-ranged models of spin-glasses. *Phys. Rev. B*, *17*(11), 4384–4403.

Larrañaga, P., & Lozano, J. A. (Eds.) (2002). *Estimation of distribution algorithms: A new tool for evolutionary computation*. Boston, MA: Kluwer.

Mezard, M., Parisi, G., & Virasoro, M. (1987). *Spin glass theory and beyond*. Singapore: World Scientific.

Mühlenbein, H., & Paaß, G. (1996). From recombination of genes to the estimation of distributions I. Binary parameters. *Parallel Problem Solving from Nature*, 178–187.

Pelikan, M. (2005). *Hierarchical Bayesian optimization algorithm: Toward a new generation of evolutionary algorithms*. Springer-Verlag.

Pelikan, M. (2008). Analysis of estimation of distribution algorithms and genetic algorithms on NK landscapes. (pp. 1033–1040).

Pelikan, M., & Goldberg, D. E. (2001). Escaping hierarchical traps with competent genetic algorithms. *Genetic and Evolutionary Comp. Conf. (GECCO-2001)*, 511–518.

Pelikan, M., Goldberg, D. E., & Lobo, F. (2002). A survey of optimization by building and using probabilistic models. *Computational Optimization and Applications*, *21*(1), 5–20.

Pelikan, M., Katzgraber, H. G., & Kobe, S. (2008). Finding ground states of Sherrington-Kirkpatrick spin glasses with hierarchical BOA and genetic algorithms. In *Genetic and Evolutionary Comp. Conf. (GECCO-2008)* (pp. 447–454). ACM.

Pelikan, M., Sastry, K., Butz, M. V., & Goldberg, D. E. (2006). Performance of evolutionary algorithms on random decomposable problems. In *PPSN* (pp. 788–797).

Pelikan, M., Sastry, K., & Cantú-Paz, E. (Eds.) (2006). *Scalable optimization via probabilistic modeling: From algorithms to applications*. Springer-Verlag.

Pelikan, M., Sastry, K., Goldberg, D. E., Butz, M. V., & Hauschild, M. (2009). *Performance of evolutionary algorithms on NK landscapes with nearest neighbor interactions and tunable overlap* (MEDAL Report No. 2009002). Missouri Estimation of Distribution Algorithms Laboratory, University of Missour–St. Louis, St. Louis, MO.

Schwarz, J., & Ocenasek, J. (2000). A problem-knowledge based evolutionary algorithm KBOA for hypergraph partitioning. Personal communication.

Spin Glass Ground State Server (2004). `http://www.informatik.uni-koeln.de/ls\_juenger/research/sgs/sgs.html`. University of Köln, Germany.

Stonedahl, F., Rand, W., & Wilensky, U. (2008). CrossNet: a framework for crossover with network-based chromosomal representations. In *Genetic and Evolutionary Comp. Conf. (GECCO-2008)* (pp. 1057–1064). New York, NY, USA: ACM.

Thierens, D. (1999). Scalability problems of simple genetic algorithms. *Evolutionary Computation*, *7*(4), 331–352.

Wright, A. H., Thompson, R. K., & Zhang, J. (2000). The computational complexity of n-k fitness functions. *IEEE Trans. Evolutionary Computation*, *4*(4), 373–379.

Yu, T.-L., Goldberg, D. E., Sastry, K., Lima, C. F., & Pelikan, M. (2009). Dependency structure matrix, genetic algorithms, and effective recombination. *Evolutionary Computation*, *17*(4), 595–626.