Dependency Structure Matrix Analysis: Offline Utility of the Dependency Structure Matrix Genetic Algorithm

Tian-Li Yu and David E. Goldberg

Illinois Genetic Algorithms Laboratory (IlliGAL)
Department of General Engineering
University of Illinois at Urbana-Champaign
104 S. Mathews Ave, Urbana, IL 61801
{tianliyu, deg}@illigal.ge.uiuc.edu

Abstract. This paper investigates the off-line use of the dependency structure matrix genetic algorithm (DSMGA). In particular, a problem-specific crossover operator is design by performing dependency structure matrix (DSM) analysis. The advantages and disadvantages of such an off-line use are discussed. Two schemes that helps the off-line usage are proposed. Finally, those off-line schemes are demonstrated by DSMGA on MaxTrap functions.

1 Introduction

Dependency structure matrix genetic algorithm (DSMGA) [1] was proposed under the inspirations of Holland's observation of the importance of linkage [2], Goldberg's genetic algorithm (GA) decomposition [3], and the use of dependency structure matrix (DSM) in organization theory [4,5]. In particular, the DSMGA adopts DSM analysis to identify linkage groups or building blocks (BBs) and then utilizes the BB information to perform a BB-wise crossover which tries to maximize BB mixing and minimize BB disruptions. The DSM analysis utilizes a DSM clustering algorithm [6] which turns pair-wise dependencies into linkage-group information, or BB information. The first proposal of the DSMGA [1] performs the DSM analysis every generation to retrieve BB information, like most probabilistic model building GAs [7] or estimation of distribution algorithms [8] do. However, the DSM analysis can also be performed off-line. In other words, one can simply apply the DSM analysis once to retrieve BB information and then use a simple GA (sGA) with a crossover operator designed for the problem according to the BB information.

The purpose of this paper is to investigate the off-line use of DSM analysis for GAs. The paper first gives an introduction to the DSMGA. The advantages and disadvantages of the off-line DSM analysis are then discussed, and two schemes for off-line usage are proposed. To demonstrate the off-line schemes of DSM analysis, a MaxTrap function is tested. Finally, the paper discusses some possible future work, followed by conclusions.

© Springer-Verlag Berlin Heidelberg 2004

2 An Introduction to DSMGA

This section gives a brief introduction to the DSMGA. Readers who are interested in DSM clustering are referred to [6]. For more details about DSMGA, please refer to [1].

2.1 DSM and DSM Clustering

A DSM is a matrix where each entry d_{ij} represents the dependency between node i and node j [4,5]. Entries d_{ij} can be real numbers or integers. The larger the d_{ij} is, the higher the dependency is between node i and node j. The diagonal entries (d_{ii}) have no significance and are usually set to zero or blacked-out. Figure 1 gives an example of DSM.

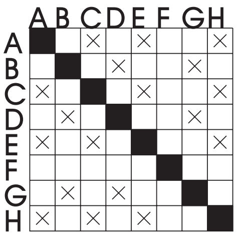

Fig. 1. A DSM. "x" means that dependency exists; the blank squares means no dependency. This figure illustrates, for example, that A and B are independent, and that A and C are dependent. Clustering is not so obviously at the first glance.

The goal of DSM clustering is to find subsets of DSM elements (i.e., clusters) so that nodes within a cluster are maximally dependent and clusters are minimally interacting. DSM clustering is a sophisticated task which requires expertise [9]. For example, not too many people know how to cluster the DSM in Figure 1. However, after we reorder the nodes (Figure 2), it is easily seen that the DSM can be cleanly clustered into three parts: {B, D, G}, {A, C, E, H}, and {F}.

Yu, Yassine, and Goldberg [6] proposed the following objective function by using the minimal description length principle.

$$f_{DSM}(M) = \left(n_c \log(n_c) + \log(n_n) \Sigma_{i=1}^{n_c} cl_i\right) + \left(|S|(2\log(n_n) + 1)\right), \qquad (1)$$

where f measures the description length that a model M needs to describe a given DSM; n_c is the number of clusters in M; n_n is the number of nodes of the DSM; cl_i is the size of the i-th cluster in M; S is the set of mis-described data of M. The above objective function has shown capable to cluster a given DSM, and the clustering results competes with human experts.

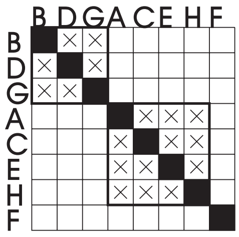

Fig. 2. The same DSM in Figure 1 but after reordered. The DSM can be cleanly clustered as ((B,D,G)(A,C,E,H),(F)).

2.2 Utilizing DSM Clustering to Identify BBs: The DSMGA

The DSM clustering algorithm can be thought as an linkage-identification algorithm which turns pair-wise linkage information into high-order linkage information. DSMGA [1] is proposed based on this thought.

DSMGA utilizes statistical analysis to estimate the fitness of order-2 schemata. Based on a nonlinearity-detection method similar to LINC [10], a DSM where each entry d_{ij} represents the linkage between gene i and gene j is created. By applying the DSM clustering algorithm, the pair-wise linkage information is turned into BB information, which is then utilized to achieve the BB-wise crossover [11]. The DSMGA has shown capable to correctly identify BBs and efficiently solve problems with uniformly-scaled BBs.

3 Why Go for Offline: The Pros and the Cons

There are mainly three types of linkage-identification methods: (1) Perturbation/statistical analysis (*e.g.* mGA [12], gemGA [13], LINC/LIMD GA [10], DSMGA [1]) (2) linkage adaptation (*e.g.* LEGO [14], LLGA [15]), and (3) probabilistic model building [7,8]. Among these three types, the first type of linkage-identification methods is most suitable for off-line usage because they do not require promising individuals; instead, those methods favor a uniformly distributed population over the solution space, or perturbations from several individuals. The remaining of this section discusses the pros and the cons of performing linkage-identification methods off-line.

Pros:

Computational Time Saving. If the linkage-identification algorithm is used off-line, because it is only performed once, obviously there is a speedup obtained from the linkage-identification procedure. The speedup should be in the order of the number of generations of the GA.

The off-line usage does not only save computational time on linkage-identification, it also possibly saves some number of function evaluations. By considering genetic drifting and statistical error of decision-making, Pelikan [16] has shown that to guarantee a high confidence of obtaining correct dependencies, roughly a population size of $O(l)$ is needed, where l is the problem size. By assuming an infinite population size and perfect mixing, the time to convergence has shown to grow proportional to $O(\sqrt{l})$ [17,11] (for a finite population size, see [18]). The above models suggests a lower bound of the number of function evaluations $O(l^{1.5})$ for a GA with linkage-identification techniques.

If the linkage-identification techniques are performed off-line, and every BB is correctly identified, according to the gambler's ruin model [19], a population size $O(l^{0.5}\log l)$ is needed (the $\log l$ is so when the failure rate is set to $O(\frac{1}{l})$). Therefore, the lower bound of the number of function evaluations becomes $O(l)+O(l^{0.5}\log l)\times O(\sqrt{l}) = O(l\log l)$, where the first $O(l)$ comes from the off-line linkage identification, and $O(l^{0.5}\log l)\times O(\sqrt{l})$ comes from the production of the population size and the convergence time. This is surely just a loose lower bound because an infinite population size and perfect mixing is assumed in the time-to-convergence model.

The accuracy of the linkage model should be taken into consideration when we try to estimate a possible speedup. When the linkage-identification method is applied off-line, a fewer number of individuals are investigated and the BB information might be less accurate. As a result, the inaccuracy of the BB information causes a longer convergence time. This will be mentioned later when the disadvantages are discussed.

BB information reusability. Another reason to perform linkage identification off-line is that the BB information retrieved from one problem might be reusable for another problem. Take the car design as an example. The car design usually has similar multiple objectives but the design for different cars weight those objectives differently. For example, the design for one car might put importance on acceleration, while the design for another car might emphasize more on comfortability. However, no matter how different the design weights those objectives, some dependencies are invariant. For instance, one can imagine that the design of the brake system should depend more on the engine and depend less on the windscreen wiper. In such cases, the BBs for the series of the problems are similar, and hence the BB information are reusable. After one of those problems is optimized by the GA with linkage-identification techniques, other problems should be optimized efficiently by a sGA with the same BB information.

Population size estimation. Several population-sizing models have been proposed including the decision-making model [20] and the gambler's ruin model [19]. To use those population-sizing models, several parameters are needed including the number of BBs m, the order of BBs k, the variance of fitness contributed by BBs σ_{BB}^2, and the minimal signal magnitude between the correct BB and the most competing BB d_{min}. The first two parameters are straightforward once the BB information is obtained. However, it is not an easy task to estimate the latter two parameters, and we simply leave it as future work for now. As a conclusion, the off-line usage of linkage-identification methods possibly enables us to estimate the population size needed for the GA.

Cons:

Less linkage model error tolerance. In a worst-case scenario, a misidentified BB in the off-line BB information makes the BB difficult to correctly converge for the GA because of BB disruptions. If the signals of BBs are not too small for the linkage-identification method to detect (otherwise, both on-line and off-line linkage identification would fail), when the linkage-identification method is performed every generation, the disruption would not be that severe. When a BB is misidentified in some generation because the current population does not reveal the signal of that BB, as long as the linkage-identification method is not too inaccurate, the BB would be correctly identified in the next generation with a high probability. Yu and Goldberg [21] indicates that as long as the number of misidentified BBs e is smaller than $O(\sqrt{m})$, where m is the number of BBs of the problem, the convergence of $(m-1)$ BBs is highly possible. However, if the linkage-identification method is performed off-line and it misidentifies e BBs, in the worst case, only $(m-e)$ BBs would correctly converge. Therefore, the quality of the retrieved BB information is more important in the off-line usage. In the next section, two schemes are proposed to alleviate the quality problem.

BB-invariance assumption. One of the assumptions of the off-line use of linkage-identification methods is that BBs are invariant. However, it is known that some optimization problems are hierarchical [22], and the dependencies varies with the degree of the GA convergence. In this case, probably the linkage-identification method needs to be performed every several generations, and the scheme becomes a evaluation relaxation version of the on-line usage which performs the linkage-identification algorithm every generation. If on-line estimation of the number of errors of BB information can be obtained, we can perform the linkage-identification algorithm only when the number of errors exceeds some predefined threshold.

4 Identifying Linkage Offline: Two Schemes

In this section, we propose two schemes that help for the off-line usage. Later, in section 5, the strength and weakness of these two schemes are shown empirically.

4.1 Sizing the Population Properly

One important issue when the linkage-identification algorithm is performed off-line is to determine the population size needed for the algorithm. Population sizing is especially important for the off-line usage. A non-sufficient population size will decrease the BB information quality and cause a great loss of solution quality.

Here we propose the following scheme, which is similar to the parameterless GA [23]. First, we choose a reasonable, but small population size N_1. For example, a population size of $N_1 = l$ is a good initial guess, where l is the problem size. In the next trial, N_1 more individuals are investigated, and the linkage-identification algorithm is performed on the doubled population size, $N_2 = 2N_1$. For the next trial, N_2 more individuals are investigated. This procedure is repeated until the BB information does not vary. If the population size actually needed to reveal correct linkages with a high probability is N, this scheme would at most costs $8N$ number of function evaluations (in which case we are unlucky at N, and get correct linkages on $2N$ and $4N$). Depending on the noise of linkage identification and the time constraint, practically one should stop the procedure when the BB information retrieved in the last two runs are similar enough. Note that this population size is only used when performing the linkage-identification algorithm. After that, the GA performs on a population size estimated by one of the population-sizing models [20,19], by some prior knowledge, or by empirical observations. Note that unlike the parameterless GA, the number of function evaluations consumed does not increase that fast, when the doubling-population-size procedure is stopped at a population size N, the number of function evaluations consumed is only N.

4.2 Combining BB Information

As indicated in the previous section, one of the disadvantages of the off-line usage of the linkage-identification method is that the off-line usage is less error-tolerant. If the BB information contains e errors, probably only $(m - e)$ BBs would correctly converge. Suppose that we stop the above doubling-population-size scheme at some point, and the BB information contains some errors. Now we are facing the following two questions: (1) Should we double the population size again to make the BB information more accurate or execute the linkage-identification algorithm on an independent population of the same size and try to combine the BB information, and (2) if the latter scheme is chosen, how to combine BB information? This subsection proposes a method to combine two BB information and leaves the first question to section 5.

As indicated in [21], there are two types of error that can happen for a linkage-identification algorithm. One is that the linkage-identification algorithm indicates that two genes are linked but in reality they are not; the other is that the linkage-identification algorithm indicates no linkage between two genes but in reality they are linked. The first type of error slows down the BB mixing [24] and causes a larger estimation of the population size (because of the higher-order linkage). The second type of error causes BB disruptions. When the linkage-identification method is performed off-line, the second type of error means a lower solution quality as mentioned in section 3. In other words, in the off-line scheme, the second type of error is more destructive than it is in the on-line scheme. Therefore, when combining two copies of BB information, the combined BB information indicates no linkage between two genes only when both copies of BB information indicate so.

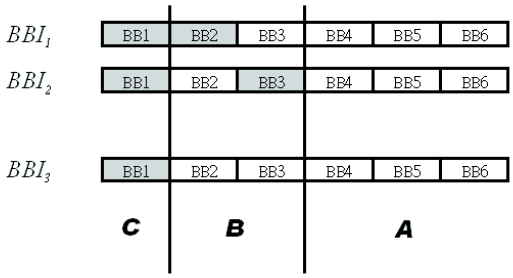

Fig. 3. Combining BB information. BBI_3 is the combination of BBI_1 and BBI_2. The shaded BBs are misidentified. Suppose a proportion $e1$ of BBs are misidentified in BBI_1, and $e2$ proportion in BBI_2. BBI_3 at most misidentifies a proportion e_1e_2 of BBs. The proportion of correctly identified BBs (regions A and B) in BBI_3 is $(1-e_1)(1-e_2)$.

Suppose that we have two independent copies of BB information, BBI_1 and BBI_2; let BBI_3 be the combined BB information according to the above scheme (Figure 3). Suppose that BBI_1 causes e_1 proportion of BB disruptions, and BBI_2 causes e_2 proportion of BB disruptions. On average, a proportion $(1-e_1)(1-e_2)$ of BBs are correctly identified in both BBI_1 and BBI_2, and they are also correctly identified in BBI_3 (region A in Figure 3). A proportion $(e_1)(1-e_2)+(e_2)(1-e_1)$ of BBs will be disrupted by one of BBI_1 and BBI_2, while BBI_3 will correctly identify those BBs according to the combining scheme (region B in Figure 3). A proportion e_1e_2 of BBs will be disrupted by both BBI_1 and BBI_2, and BB disruptions might also occur by using BBI_3 (region C in Figure 3). However, BBI_3 still have a chance not to disrupt them. For instance, if a BB {1, 2, 3, 4, 5} is misidentified by BBI_1 as {1, 2, 3} and {4, 5}, and misidentified by BBI_2 as {1, 2} and {3, 4, 5}, BBI_3 will still correctly identify that BB. To be conservative, we assume that the region C in Figure reffig:BBI will always cause BB disruptions. Then BBI_3 will only cause a proportion of e_1e_2 BB disruptions.

Note that the above calculations assume that BBI_1 and BBI_2 are independent.

4.3 A Little Complexity Analysis

This subsection shows that the above usage of multiple copies of BB information does not increase the lower bound of the number of function evaluations that GAs need.

Suppose that b different copies of BB information were retrieved by performing the linkage-identification method off-line b times, and on average the number of misidentified BBs is e. According to [21] (also see this for the meaning of the misidentified BB), $e < O(\sqrt{m})$ must be satisfied; otherwise, even the on-line version of the GA would not achieve a $(m-1)$-BB convergence. Assuming that the errors are uniformly distributed, the probability for a copies of BB information misidentifying a specific BB is then $\frac{e}{m}$. The probability that all b copies of BB information contains wrong information for a specific BB is then $\left(\frac{e}{m}\right)^b$. The expected number of BBs that all b copies of BB information misidentify is $m\left(\frac{e}{m}\right)^b$. To achieve a $(m-1)$-BB convergence, $m\left(\frac{e}{m}\right)^b \leq 1$ must be satisfied. Given that $e < O(\sqrt{m})$, we can obtain that b is roughly bounded by $O(\log m)$. The estimation of number of function evaluations when we talked about the computational time saving of the off-line usage becomes $O(l \log l) + O(l^{0.5} \log l) \times O(\sqrt{l}) = O(l \log l)$, which means, the use of multiple different copies of BB information does not increase the lower bound of the number of function evaluations.

5 Empirical Results

This section demonstrates the off-line scheme by performing off-line DSM analysis on a 5×10 MaxTrap function (a concatenated trap function of 10 5-bit traps). A GA with $(\lambda + \mu)$ selection, where $\lambda = 100$ and $\mu = 1000$, and uniform crossover is used to cluster the DSM of genetic dependencies (for the linkage identification). All experiment results are averaged over 50 independent runs. The reason of choosing MaxTrap as the test base is that if a BB is misidentified, most likely the crossover operator would disrupt the BB. For more details about deceptions, see [25].

The doubling-population-size scheme on average needed 11712 function evaluations (6400 for 17 times; 12800 for 83 times), and correctly identifies 99.8% BBs (100% for 98 times; 90% for 2 times). As predicted, the number function evaluations is much larger than needed (Figure 4).

Figure 4 illustrates the use of the combining BB information scheme. The dashed line represents the combination of the BB information given by DSM analysis on two populations of the same size. For instance, for number of function evaluation 4000, the direct scheme (solid line) simply applies the DSM analysis on a population of size 4000. The combining scheme (dashed line) applies the DSM analysis on two independent populations of size 2000 and then uses the combined BB information. It can be seen that when the BB information is highly erroneous, combining BB information is not worthwhile, and simply doubling the population would be better. However, when the BB information is more accurate (roughly when the BB information is 92% accurate in experiments), the combining scheme outperforms the direct scheme.

The reason might be that when the population is too small, doubling the population size will greatly increase the accuracy of pair-wise dependency detection. However, when the population is large enough to reveal pair-wise dependencies, the errors mainly come from the the GA failure on DSM clustering. In the experiments, the proportion of correctly identified BBs roughly saturates at 0.98. The combining scheme outperforms the direct scheme since the combination of two 0.9 BB information will correctly identify more than $1 - 0.1 \times 0.1 = 0.99$ BBs.

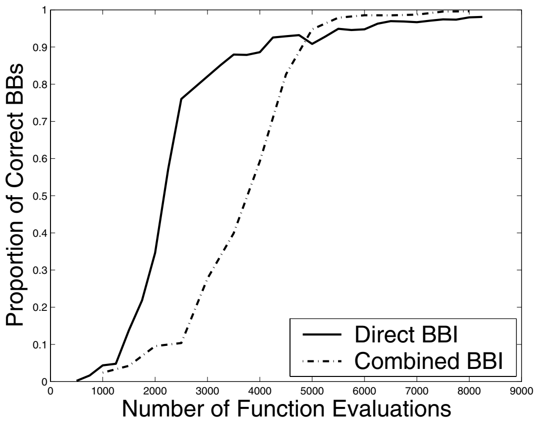

Fig. 4. The number of function evaluations versus the proportion of correctly identified BBs for two different schemes. The solid line represents directly applying the DSM analysis on individuals. The dashed line represents the combination of the BB information given by DSM analysis on two populations of the same size.

We can do a little modeling for the two schemes. Assume that the probability that one chromosome reveals a particular pair-wise dependency is p. The probability that all N independent chromosome fail to reveal a particular pair-wise dependency can be modeled as $q(N) = (1-p)^N$. Suppose that for a BB to be identified, s pair-wise dependencies need to be detected. Assuming pair-wise dependencies are uniformly important, the probability that the signal of a BB is too weak to identify can be approximated as $P_1(N) = 1-(1-q(N))^s$. In addition, even if the signal of a BB is strong enough for the linkage-identification method to identify, the linkage-identification method might still fail with a small probability P_2 because of stochastic behaviors. The probability that the linkage-identification method fails to identify a particular BB is then $error_{Scheme_1}(N) = 1-(1-P_1(N))(1-P_2)$. The value of function $error_{Scheme_1}$ also varies with p and s, but here we emphasize that it is a function of N. If we combine two copies of BB information, the error of the combining scheme can be modeled as $error_{Scheme_2}(N) = error^2_{Scheme_1}(N/2)$. Figure 5 plots the models with $p = 0.1$, $s = 10$, and $P_2 = 0.05$. The models basically agree with the empirical observations in Figure 4.

If the proportion of correctly identified BBs can be estimated, it is possible to combine the strength of the two schemes. Suppose that the population size is N in the current iteration, the BB information is BBI_1, and the proportion of correctly identified BBs is $Q(BBI_1)$. In the next generation, when N_1 more individuals are being investigated, the DSM analysis is performed on both the new N individuals and the combined $2N$ individuals. Suppose the BB information are BBI_2 and BBI_3 respectively. Let BBI_4 be the combined BB information of BBI_1 and BBI_2. Then we switch to the combining-BB-information scheme

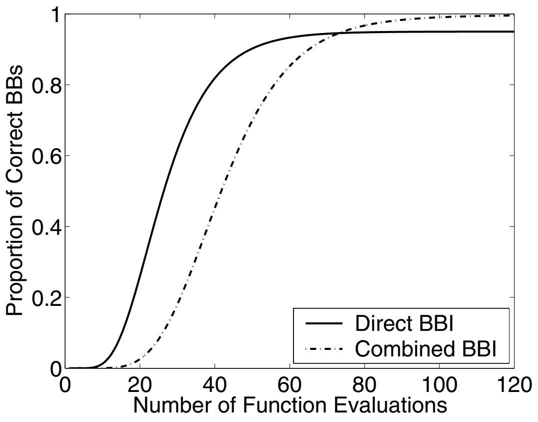

Fig. 5. Models for the two schemes in Figure 4.

when $Q(BBI_4) > Q(BBI_3)$; otherwise, we keep doubling the population size until the combined BB information is better.

6 Future Work and Conclusions

This paper leaves a number of possibilities for future work. The empirical results indicated that the combining-BB-information scheme works better when the BB information is accurate. We would like to see if it is possible to suggest an optimal switching time between the two schemes. The discussions in this paper are not limited to separable problems, but more experiments need to be done to verify that, and some modifications of the discussions can be expected. In section 3, we mentioned about utilizing the BB information to estimate the needed population size, and we are currently investigating the possibility of doing so.

To conclude, this paper investigated the off-line use of DSM analysis. The advantages and disadvantages of the off-line use of linkage-identification methods were discussed. Two schemes that helps the off-line usage were proposed. The number of function evaluations needed for the two schemes were analyzed. To demonstrate those schemes, experiments of a MaxTrap function were performed. The results showed that the off-line usage of DSMGA is applicable, and DSM analysis is suitable for designing a problem-specific crossover operator.

Acknowledgment. This work was sponsored by the Air Force Office of Scientific Research, Air Force Materiel Command, USAF, under grant F49620-03-1-0129, and by the Technology Research, Education, and Commercialization Center (TRECC), at University of Illinois at Urbana-Champaign, administered by the National Center for Supercomputing Applications (NCSA) and funded by the Office of Naval Research under grant N00014-01-1-0175. The U.S. Government is authorized to reproduce and distribute reprints for government purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Office of Scientific Research, the Technology Research, Education, and Commercialization Center, the Office of Naval Research, or the U.S. Government.

References

1. Yu, T.-L., Goldberg, D.E., Yassine, A., Chen, Y.-p.: Genetic algorithm design inspired by organizational theory: Pilot study of a dependency structure matrix driven genetic algorithm. Proceedings of Artificial Neural Networks in Engineering 2003 (ANNIE 2003) (2003) 327–332 (Also IlliGAL Report No. 2003007).
2. Holland, J.H.: Adaptation in natural and artificial systems. University of Michigan Press, Ann Arbor, MI (1975)
3. Goldberg, D.E.: The design of innovation: Lessons from and for competent genetic algorithms. Kluwer Academic Publishers, Boston, MA (2002)
4. Steward, D.V.: The design structure system: A method for managing the design of complex systems. IEEE Transactions on Engineering Management **28** (1981) 77–74
5. Yassine, A., Falkenburg, D.R., Chelst, K.: Engineering design management: An information structure approach. International Journal of production research **37** (1999) 2957–2975
6. Yu, T.-L., Yassine, A., Goldberg, D.E.: A genetic algorithm for developing modular product architectures. Proceedings of the ASME 2003 International Design Engineering Technical Conferences, 15th International Conference on Design Theory and Methodology (DETC 2003) (2003) DTM–48657 (Also IlliGAL Report No. 2003024).
7. Pelikan, M., Goldberg, D.E., Lobo, F.G.: A survey of optimization by building and using probabilistic models. IlliGAL Report No. 99018, University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory, Urbana, IL (1999)
8. Larrañaga, P., Lozano, J.A., eds.: Estimation of Distribution Algorithms. Kluwer Academic Publishers (2002)
9. Sharman, D., Yassine, A., Carlile, P.: Characterizing modular architectures. ASME 14th International Conference, DTM-34024 (2002)
10. Munetomo, M., Goldberg, D.E.: Identifying linkage groups by nonlinearity/nonmonotonicity detection. Proceedings of the Genetic and Evolutionary Computation Conference 1999: Volume 1 (1999) 433–440
11. Thierens, D., Goldberg, D.E.: Convergence models of genetic algorithm selection schemes. In: Parallel Problem Solving from Nature, PPSN III. (1994) 119–129
12. Goldberg, D.E., Korb, B., Deb, K.: Messy genetic algorithms: Motivation, analysis, and first results. Complex Systems **3** (1989) 493–530
13. Kargupta, H.: The performance of the gene expression messy genetic algorithm on real test functions. Proceedings of 1996 IEEE International Conference on Evolutionary Computation (1996) 631–636

14. Smith, J., Fogarty, T.C.: Recombination strategy adaptation via evolution of gene linkage. Proceedings of the 1996 IEEE International Conference on Evolutionary Computation (1996) 826–831
15. Harik, G.R., Goldberg, D.E.: Learning linkage. Foundations of Genetic Algorithms **4** (1996) 247–262
16. Pelikan, M.: Bayesian optimization algorithm: From single level to hierarchy. Ph.d. dissertation, University of Illinois at Urbana-Champaign (2002) (Also IlliGAL Report No. 2002023).
17. Mühlenbein, H., Schlierkamp-Voosen, D.: Predictive models for the breeder genetic algorithm: I. Continuous parameter optimization. Evolutionary Computation **1** (1993) 25–49
18. Ceroni, A., Pelikan, M., Goldberg, D.E.: Convergence-time models for the simple genetic algorithm with finite population. IlliGAL Report No. 2001028, Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign, Urbana, IL (2001)
19. Harik, G., Cantú-Paz, E., Goldberg, D.E., Miller, B.L.: The gambler's ruin problem, genetic algorithms, and the sizing of populations. Proceedings of 1997 IEEE International Conference on Evolutionary Computation (1997) 7–12
20. Goldberg, D.E., Deb, K., Clark, J.H.: Genetic algorithms, noise, and the sizing of populations. Complex Systems **6** (1992) 333–362
21. Yu, T.-L., Goldberg, D.E.: Toward an understanding of the quality and efficiency of model building for genetic algorithms. Proceedings of the Genetic and Evolutionary Computation Conference 2004 (2004) (To appear)(Also IlliGAL Report No. 2004004).
22. Simon, H.A.: The Sciences of the Artificial. The MIT Press, Cambridge, MA (1968)
23. Harik, G.R., Lobo, F.G.: A parameter-less genetic algorithm. Proceedings of the Genetic and Evolutionary Computation Conference 1999: Volume 1 (1999) 258–265
24. Sastry, K., Goldberg, D.E.: Analysis of mixing in genetic algorithms: A survey. IlliGAL Report No. 2002012, Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign, Urbana, IL (2002)
25. Deb, K., Goldberg, D.E.: Analyzing deception in trap functions. Foundations of Genetic Algorithms 2 (1993) 93–108