Substructural Neighborhoods for Local Search in the Bayesian Optimization Algorithm

Claudio F. Lima[1], Martin Pelikan[2], Kumara Sastry[3], Martin Butz[4], David E. Goldberg[3], and Fernando G. Lobo[1]

[1] University of Algarve, Portugal
[2] University of Missouri at St. Louis, USA
[3] University of Illinois at Urbana-Champaign, USA
[4] University of Würzburg, Germany
clima@ualg.pt, pelikan@cs.umsl.edu, ksastry@uiuc.edu, mbutz@psychologie.uni-wuerzburg.de, deg@uiuc.edu, flobo@ualg.pt

Abstract. This paper studies the utility of using substructural neighborhoods for local search in the Bayesian optimization algorithm (BOA). The probabilistic model of BOA, which automatically identifies important problem substructures, is used to define the structure of the neighborhoods used in local search. Additionally, a surrogate fitness model is considered to evaluate the improvement of the local search steps. The results show that performing substructural local search in BOA significatively reduces the number of generations necessary to converge to optimal solutions and thus provides substantial speedups.

1 Introduction

Estimation of distribution algorithms (EDAs) [1,2], a new class of genetic and evolutionary algorithms (GEAs), have frequently been found to be more efficient than traditional GEAs that use fixed, problem-independent variation operators. The conceptual difference is that EDAs replace the traditional variation operators of GEAs by building and sampling a probabilistic model of promising solutions. In essence, this procedure tries to mimic the behavior of an ideal recombination operator that combines subsolutions with minimal disruption.

Although EDAs are effective at exploring the search space to find promising regions, they inherit a common drawback from traditional GEAs: slower convergence to optimal solutions when compared with appropriate local searchers that start the search within the basin of attraction of the optima. This observation has led to the combination of GEAs with local search methods known as hybrid GEAs or memetic algorithms [3,4]. In this context EDAs are no exception and many applications in real-world optimization have been accomplished with the help of some sort of local search. However, systematic methods for hybridizing and designing competent global and local-search methods that automatically identify the problem decomposition and important problem substructures are still scarce. For instance, the probabilistic models of EDAs contain useful information about the underlying problem structure that can be exploited to speedup the convergence of EDAs to optimal solutions.

T.P. Runarsson et al. (Eds.): PPSN IX, LNCS 4193, pp. 232–241, 2006.

In this paper we use substructural neighborhoods to perform local search in the Bayesian optimization algorithm (BOA) [5]. These neighborhoods are defined by the dependency groups learned by the probabilistic model of BOA. Additionally, we use a surrogate fitness model that also makes use of substructural information to evaluate the alternatives while performing hillclimbing in the subsolution search space. The results show that incorporating substructural local search in BOA leads to a significant reduction in the number of generations, providing relevant speedups in terms of number of evaluations.

The next section gives an outline of the Bayesian optimization algorithm and how fitness can be modeled under this framework. In Section 3, we introduce several substructural neighborhoods, followed by the incorporation of a substructural hillclimber in BOA. Section 5 presents and discusses empirical results. The paper ends with a summary and major conclusions.

2 Bayesian Optimization Algorithm

Estimation of distribution algorithms [1,6] replace traditional variation operators of GEAs by building a probabilistic model of promising solutions (that survive selection) and sampling the corresponding probability distribution to generate the offspring population. The Bayesian optimization algorithm [5,6] uses Bayesian networks as the probabilistic model to capture the (in)dependencies between the variables of the problem.

Like traditional GAs, BOA starts with an initial population (usually randomly generated) that is evaluated and submitted to a selection operator that gives preference to high-quality solutions. The set of selected individuals is then used as the *training dataset* to learn the probabilistic model for the present generation. After obtaining the model structure and parameters, the offspring population is generated by sampling from the distribution of modeled individuals. The new solutions are then evaluated and incorporated into the original population. Here, we use a simple replacement scheme where new solutions fully replace the original population.

2.1 Modeling (in)Dependencies Between Variables in BOA

Bayesian networks [7] are powerful graphical models that combine probability theory with graph theory to encode probabilistic relationships between variables of interest. A Bayesian network is defined by a structure and corresponding parameters. The structure is represented by a directed acyclic graph where the nodes correspond to the variables of the data to be modeled and the edges correspond to conditional dependencies. The parameters are represented by the conditional probabilities for each variable given any instance of the variables that this variable depends on. More formally, a Bayesian network encodes the following joint probability distribution,

$$p(X) = \prod_{i=1}^{\ell} p(X_i|\Pi_i), \tag{1}$$

where $X = (X_1, X_2, \ldots, X_\ell)$ is a vector of all the variables of the problem, Π_i is the set of *parents* of X_i (nodes from which there exists an edge to X_i), and $p(X_i|\Pi_i)$ is the conditional probability of X_i given its parents Π_i.

In BOA, both the structure and the parameters of the probabilistic model are searched and optimized to best fit the data (set of promising solutions). To learn the most adequate structure for the Bayesian network a greedy algorithm is usually used for a good compromise between search efficiency and model quality.

The parameters of a Bayesian network are represented by a set of conditional probability tables (CPTs) specifying the conditional probabilities for each variable given all possible instances of the parent variables Π_i. Alternatively, these conditional probabilities can be stored in the form of local structures such as decision trees or decision graphs, allowing a more efficient and flexible representation of local conditional distributions. In this work, decision trees are used to encode the parameters of the Bayesian network.

2.2 Modeling Fitness in BOA

Pelikan and Sastry [8] extended the Bayesian networks used in BOA to encode a surrogate fitness model that is used to estimate the fitness of a proportion of the population, thereby reducing the total number of function evaluations. For each possible value x_i of every variable X_i, an estimate of the marginal fitness contribution of a subsolution with $X_i = x_i$ is stored for each instance π_i of X_i's parents Π_i. Therefore, in the binary case, each row in the CPT is extended by two additional entries. The fitness of an individual can then be estimated as

$$f_{est}(X_1, X_2, \ldots, X_\ell) = \bar{f} + \sum_{i=1}^{\ell} \left(\bar{f}(X_i|\Pi_i) - \bar{f}(\Pi_i) \right), \tag{2}$$

where $\bar{f}$ is the average fitness of all solutions used to learn the surrogate, $\bar{f}(X_i|\Pi_i)$ is the average fitness of solutions with X_i and Π_i, and $\bar{f}(\Pi_i)$ is the average fitness of all solutions with Π_i.

Fitness information can also be incorporated in Bayesian networks with decision trees or graphs in a similar way. In this case, the average fitness of each instance for every variable must be stored in every leaf of the decision tree or graph. The fitness averages in each leaf are now restricted to solutions that satisfy the condition specified by the path from the root of the tree to the leaf.

3 Substructural Neighborhoods

One of the key requirements for designing an efficient mutation operator is to ensure that it searches in the correct neighborhood. This is often accomplished by exploiting and incorporating domain- or problem-specific knowledge in the design of neighborhood operators. While these neighborhood operators are designed for a particular search problem, oftentimes on an ad-hoc basis, they do not generalize their efficiency beyond a small number of applications. On the other

hand, simple bitwise hillclimbers are frequently used as local search methods with more general applicability, providing inferior but still competitive results, especially when combined with population-based search procedures. Clearly, there is a tradeoff between generalization and efficiency for neighborhood operators with fixed structure. Therefore, it is important to study systematic methods for designing neighborhood operators that can solve a broad class of search problems.

The exploration of neighborhoods defined by the probabilistic models of EDAs is an approach that exploits both the underlying problem structure while not loosing the generality of application. The resulting mutation operators explore a more *global*, problem-dependent neighborhood than traditional local, purely representation-dependent search procedures.

Recently, it has been shown that a selectomutative algorithm that performs hillclimbing in the substructural space can successfully solve problems of bounded difficulty with subquadratic scalability [9]. Sastry and Goldberg [10] proposed a building-block-wise mutation algorithm based on the probabilistic model of the extended compact genetic algorithm (eCGA) [11], where linkage information is used to perform local search among competing subsolutions. Lima *et. al.* [12] extended the regular eCGA by incorporating local search in the subsolution search space and concluded that this hybrid approach is more robust than both single-operator-based approaches [11,10].

In this paper we extend the concept of exploring substructural neighborhoods to the Bayesian optimization algorithm. Given the structure of the Bayesian network, several neighborhood topologies can be considered to perform random or improvement-guided mutations. For a given variable X_i, the corresponding set of parent nodes Π_i, and set of child nodes Ω_i (nodes to where an edge arrives from node X_i), we define three different substructural neighborhoods:

Parental neighborhood considers variable X_i together with the parent variables Π_i. This neighborhood is therefore defined by $K = 1 + |\Pi_i|$ different variables, resulting in 2^K possible values in the binary realm.

Children neighborhood considers variable X_i together with the child variables Ω_i. Thus this neighborhood is defined by $K = 1 + |\Omega_i|$ variables.

Parental+Children neighborhood considers variable X_i together with both parent variables Π_i and child variables Ω_i. This neighborhood is composed by $K = 1 + |\Pi_i| + |\Omega_i|$ variables.

These three neighborhoods explore the structure captured by the Bayesian network to different extends. In this paper, we focus on the parental neighborhood to define the neighborhood topology to be used by local search.

A somewhat related approach has been recently proposed by Handa [13], where the traditional bitwise mutation operator is employed in the estimation of Bayesian networks algorithm (EBNA) [14] and consequently variables that depend on the mutated node are resampled according to the conditional probabilities for the new instance. Although this mutation operator takes into account the dependencies between variables, it is specifically designed to perturb solutions in order to maintain diversity in the population. Our approach is to

interpret the structure of the Bayesian network as a set of linkage groups that are used to define neighborhoods to be explored by local search.

4 BOA with Substructural Hillclimbing

This section introduces a hillclimber that uses the parental neighborhood defined in the previous section to perform hillclimbing in the substructural space of an individual. This hillclimbing is performed for a proportion of the population in BOA to speedup convergence to good solutions, as in traditional hybrid GEAs. After the offspring population is sampled from the probabilistic model and evaluated, each individual is submitted to substructural hillclimbing with probability p_{ls}. The substructural hillclimber can be described as follows:

1. Consider the first variable X_i according with the ancestral reverse ordering of variables in the Bayesian network.
2. Choose the values (x_i, π_i) associated with the maximal substructural fitness $\bar{f}(X_i|\Pi_i)$.
3. Set variables (X_i, Π_i) of the considered individual to values (x_i, π_i) if the overall fitness of the individual is improved by doing so, otherwise leave the individual unchanged.
4. Repeat steps 2-3 for all remaining variables following the ancestral reverse order of variables.

Some details need further explanation. First, we use the reverse order of that used to sample the variables of new solutions, where each node is preceded by its parents. By doing so, higher-order dependencies within the same linkage group are *optimized* first. This procedure aims to reduce the possibility of doing incorrect decisions when considering problems whose lower-order statistics lead the search away from global optima.

Also, we consider two different versions of the substructural hillclimber in our study, that only differ in step 3. The first version uses the estimated fitness of the individual (Equation 2) to decide if the best substructure (according to $\bar{f}(X_i|\Pi_i)$) for a given neighborhood should be accepted, while the second version uses the actual fitness function to make the decision. After performing substructural hillclimbing for all variables, the resulting individual is evaluated with the fitness function before it is inserted back into the population. This avoids the propagation of error possibly introduced by using surrogate fitness. Thus, the surrogate is only used to perform local search in the substructural neighborhoods.

We also note that searching within the same substructural neighborhoods for different individuals yields results whose similarity increase with the accuracy of the linkage model. However, in practice, performing local search on different individuals helps to overcome incorrect biases from the errors in the substructural models.

5 Experiments

This section describes the test problems used, presents the results obtained for varying proportions of local search p_{ls}, and empirically analyzes the scalability of the proposed method with increasing problem size.

5.1 Test Problems and Experimental Setup

Two different problems are used to test the proposed method: OneMax and Trap functions. These problems represent two important bounds on a class of additively decomposable problems with bounded difficulty. In OneMax the fitness is simply given by the sum of ones in a binary string. This is a simple linear function with the optimum in the solution with all ones. Therefore, there is no need of linkage learning to be able to solve this problem. While the optimization of the OneMax problem is easy, the probabilistic models build by EDAs such as eCGA and BOA, however, are known to be only partially correct and include spurious linkages. Therefore, the results on this function will indicate if the effect of using partially correct linkage mapping on the accuracy of the surrogate is significant, and consequently if performing substructural local search under these conditions is still advantageous. This paper considers a OneMax function with size $\ell = 50$.

The second problem considered is a concatenated 5-bit Trap function [15]. This problem consists in concatenating a number of copies of the Trap function with size $k = 5$. The Trap function used is defined as follows

$$f_{Trap}(u) = \begin{cases} 5 & \text{if } u = 5 \\ 4 - u & \text{otherwise,} \end{cases} \tag{3}$$

where u is the number of ones in the substring of 5 bits. In this problem the accurate identification and exchange of the building-blocks is critical to achieve success, because processing substructures of lower order will lead to exponential scalability. Ten concatenated copies of the 5-bit Trap are used, which makes the total problem size also $\ell = 50$.

For each problem, we perform experiments for different proportions of local search p_{ls}. The proportions tested are $0.001, 0.005, 0.01, 0.03, 0.05, 0.1, 0.15, 0.2$. For the 10x5-bit Trap function, an additional value of 0.0005 is also considered. The minimal number of function evaluations required to obtain the optimal solution is empirically determined using a bisection method over the population size. For each experiment, 10 independent bisection runs are performed. Each bisection run searches for the minimal population size required to find the optimum in 10 out of 10 independent runs. Therefore, the results for the minimal sufficient population size are averaged over 10 bisection runs, while the results for the number of function evaluations and the number of generations spent are averaged over 100 (10×10) independent runs. For all experiments, binary tournament selection without replacement is used.

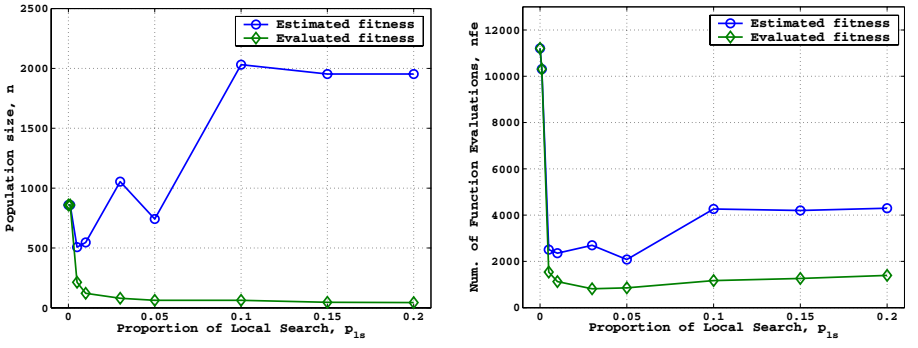

Fig. 1. Population size and number of function evaluations required to solve the 50-bit OneMax problem

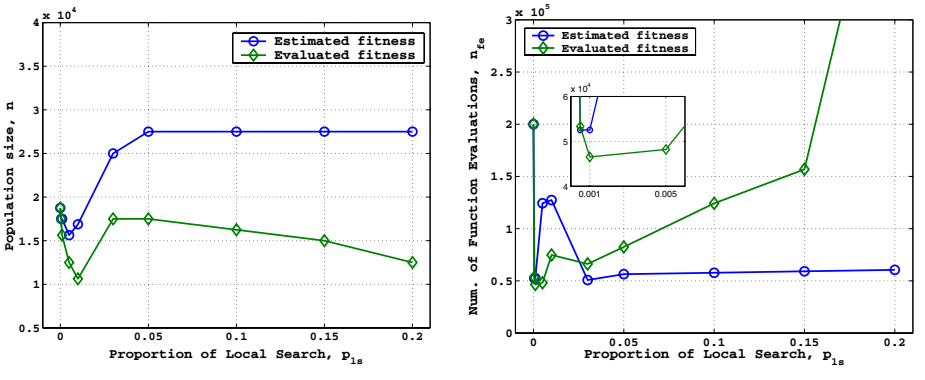

Fig. 2. Population size and number of function evaluations required to solve the 10x5-bit Trap problem

5.2 Results and Discussion

The results obtained are shown in figures 1 and 2. For both problems, the number of evaluations is significatively reduced when using local search that explores substructural neighborhoods. Also, both versions of the acceptance criteria in the substructural hillclimber reduce the cost to solve the problem. However, different dynamics can be observed for each problem.

For OneMax, using the actual fitness function when deciding if substructures should be accepted or not provides slightly better results than using estimated fitness, while the population size required is significatively smaller, in particular for higher proportions of local search. Note that the correctness of the substructural neighborhoods is not crucial when solving OneMax using local search because there is no linkage. However, the choice of the best alternative in each neighborhood is based on the substructural fitness contribution that is estimated by the surrogate whose correctness relies on the accuracy of the linkage model.

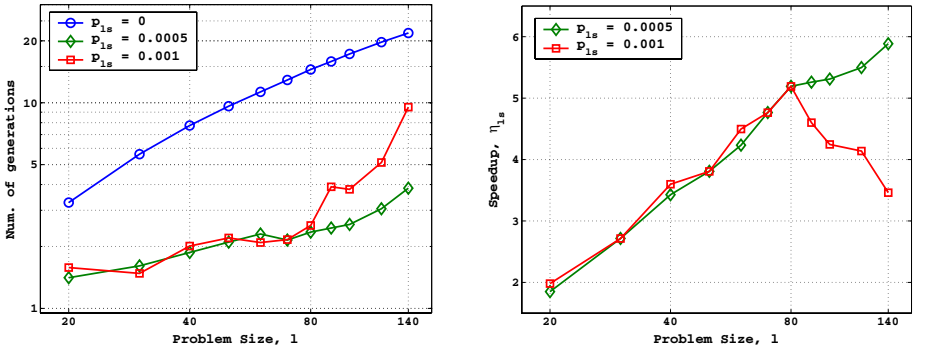

Fig. 3. Number of generations required to get the optimum and the speedup obtained by performing substructural local search on a number of concatenated 5-bit Trap functions. The speedup scales as $\mathcal{O}(\ell^{0.45})$ for $\ell \leq 80$. For $\ell > 80$ the speedup grow is more moderate for the *optimal* value of $p_{ls} = 0.0005$, while for higher proportions of local search the speedup starts to decrease due to diversity reduction in the population.

But even more important is the acceptance (or not) of the substructures. By using real fitness evaluation in this decision, only those building-blocks that really improve the fitness of the individual are accepted, which drastically reduces the need of having an accurate surrogate fitness model (and consequently a larger population size). For the hillclimber that uses only estimated fitness, the population size required grows even more for higher proportions of local search because the diversity in the population is quickly reduced, which requires the surrogate to be accurate enough to solve the problem in the first generation.

In the 10x5-bit Trap, the identification of the correct substructures is crucial to solve the problem, requiring the accuracy of the probabilistic model of BOA to be high. Therefore, both hillclimbers perform similar for small proportions of local search. Here, however, the cost of using fitness function calls at each step of the substructural hillclimber shows to be an expensive overhead for higher values of p_{ls}. Similar to OneMax, there is a transition phase in the population size required for the hillclimber that uses surrogate fitness. For $p_{ls} \geq 0.05$, the population size stagnates at a value where the model is accurate enough to solve the problem in the first generation by performing substructural local search.

Figure 3 presents the results obtained for increasing number of concatenated 5-bit Trap functions for BOA with the hillclimber that uses estimated fitness. The number of generations required to reach optima and the speedup of performing local search are shown. Note that the speedup is simply the ratio of the number of evaluations required by BOA without and with local search. Several proportions of local search were tested between 0.0001 and 0.005, but for clarity only two illustrative cases are plotted: 0.0005 and 0.001. The population size required (not plotted) scales similarly for all tested p_{ls} values.

The results show that while obtaining a significant reduction in the number of generations, substantial speedups are provided by using substructural local search

in BOA. The speedup grows approximately as $\mathcal{O}(\ell^{0.45})$ for $\ell \leq 80$. For larger problem sizes the increase in speedup becomes more moderate for $p_{ls} = 0.0005$, while for higher proportions of local search the speedup decreases. This is due to the population size required for larger problems, increasing the number of individuals that undergo local search for the same value of p_{ls}, and thereby reducing diversity in the population. Note that the resulting individuals from substructural hillclimbing are very similar. On the other hand, smaller proportions of local search (not plotted) lead to a curve with similar slope to that obtained for the best proportion but with inferior speedups. As a final remark, while $p_{ls} = 0.0005$ was found to be the most adequate value the spectrum of problem sizes tested, the optimal proportion should decrease for larger problems than considered here.

The reduction of the slope in the speedup curve for larger problem sizes is also related to the structure of the model learned by BOA. Analyzing the dependency groups captured by the Bayesian network with decision trees, it can be observed that the number and size of spurious linkages increases with problem size. By spurious linkage we mean additional variables that are considered together with a correct linkage group. Although the structure of the Bayesian network captures such spurious dependencies, the conditional probabilities nearly express independency between the spurious variables and the correct linkage, therefore not affecting the capability of sampling such variables as if they were independent. In fact, this capability of decision trees to detect more complex dependencies is one of the keys in hierarchical BOA [6] to solve more complex decomposable problems such as hierarchical problems.

6 Summary and Conclusions

In this paper, we have introduced the use of substructural neighborhoods to perform local search in BOA. Three different substructural neighborhoods—based on the structure of the learned Bayesian network—were proposed. A hillclimber that effectively searches in the subsolution search space was incorporated in BOA, using a surrogate fitness model to evaluate competing substructures. The results showed that incorporating substructural local search in BOA leads to a significant reduction in the number of generations necessary to solve the problem, while providing substantial speedups in terms of number of evaluations. More importantly, the relevance of designing and hybridizing competent operators that automatically identify the problem decomposition and important problem substructures have been empirically highlighted.

Acknowledgements. This work was sponsored by the Air Force Office of Scientific Research, Air Force Materiel Command, USAF, under grant FA9550-06-1-0096, the National Science Foundation under CAREER grant ECS-0547013, ITR grant DMR-03-25939 at Material Computation Center, UIUC, and Portuguese Foundation for Science and Technology (FCT/MCES) under grant SFRH-BD-16980-2004. The work was also supported by the High Performance Computing Collaboratory sponsored by Information Technology Services, the Research Award and the Research Board at the University of Missouri in St. Louis.

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, or the U.S. Government.

References

1. Larrañaga, P., Lozano, J.A., eds.: Estimation of distribution algorithms: a new tool for Evolutionary Computation. Kluwer Academic Publishers, Boston, MA (2002)
2. Pelikan, M., Goldberg, D.E., Lobo, F.: A survey of optimization by building and using probabilistic models. Computational Optimization and Applications **21**(1) (2002) 5–20 Also IlliGAL Report No. 99018.
3. Moscato, P.: On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms. Technical Report C3P 826, Caltech Concurrent Computation Program, California Institute of Technology, Pasadena, CA (1989)
4. Hart, W.E.: Adaptive global optimization with local search. PhD thesis, University of California, San Diego, San Diego, CA (1994)
5. Pelikan, M., Goldberg, D.E., Cantú-Paz, E.: BOA: The Bayesian Optimization Algorithm. In Banzhaf, W., et al., eds.: Proceedings of the Genetic and Evolutionary Computation Conference GECCO-99, San Francisco, CA, Morgan Kaufmann (1999) 525–532 Also IlliGAL Report No. 99003.
6. Pelikan, M.: Hierarchical Bayesian Optimization Algorithm: Toward a New Generation of Evolutionary Algorithms. Springer (2005)
7. Pearl, J.: Probabilistic reasoning in intelligent systems: Networks of plausible inference. Morgan Kaufmann, San Mateo, CA (1988)
8. Pelikan, M., Sastry, K.: Fitness inheritance in the Bayesian optimization algorithm. In Deb, K.e.a., ed.: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2004), Part II, LNCS 3103, Springer (2004) 48–59
9. Sastry, K., Goldberg, D.E.: Let's get ready to rumble: Crossover versus mutation head to head. In Deb, K., et al., eds.: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2004), Part II, LNCS 3103, Springer (2004) 126–137 Also IlliGAL Report No. 2004005.
10. Sastry, K., Goldberg, D.E.: Designing competent mutation operators via probabilistic model building of neighborhoods. In Deb, K., et al., eds.: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2004), Part II, LNCS 3103, Springer (2004) 114–125 Also IlliGAL Report No. 2004006.
11. Harik, G.R.: Linkage learning via probabilistic modeling in the ECGA. IlliGAL Report No. 99010, Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign, Urbana, IL (1999)
12. Lima, C.F., Sastry, K., Goldberg, D.E., Lobo, F.G.: Combining competent crossover and mutation operators: A probabilistic model building approach. In Beyer, H., et al., eds.: Proceedings of the ACM SIGEVO Genetic and Evolutionary Computation Conference (GECCO-2005), ACM Press (2005)
13. Handa, H.: The effectiveness of mutation operation in the case of estimation of distribution algorithms. Journal of Biosystems (2006) (to appear).
14. Etxeberria, R., Larrañaga, P.: Global optimization using Bayesian networks. In Rodriguez, A., et al., eds.: Second Symposium on Artificial Intelligence (CIMAF-99), Habana, Cuba (1999) 332–339
15. Deb, K., Goldberg, D.E.: Analyzing deception in trap functions. Foundations of Genetic Algorithms 2 (1993) 93–108