# Entropy-Based Evaluation Relaxation Strategy for Bayesian Optimization Algorithm

Hoang Ngoc Luong, Hai Thanh Thi Nguyen, and Chang Wook Ahn

School of Information and Communication Engineering, Sungkyunkwan University
300 Cheoncheon-dong, Suwon 440-746, Republic of Korea
{ngochoang,haintt86,cwan}@skku.edu

**Abstract.** Bayesian Optimization Algorithm (BOA) belongs to the advanced evolutionary algorithms (EA) capable of solving problems with multivariate interactions. However, to attain wide applicability in real-world optimization, BOA needs to be coupled with various efficiency enhancement techniques. A BOA incorporated with a novel entropy-based evaluation relaxation method (eBOA) is developed in this regard. Composed of an on-demand evaluation strategy (ODES) and a sporadic evaluation method, eBOA significantly reduces the number of (fitness) evaluations without imposing any larger population-sizing requirement. Experiments adduce the grounds for its significant improvement in the number of evaluations until reliable convergence. Furthermore, the evaluation relaxation does not negatively affect the scalability performance.

**Keywords:** Estimation of Distribution Algorithms, Bayesian Optimization Algorithm, Evaluation Relaxation, On-Demand Evaluation.

## 1 Introduction

Being an outgrowth from classical genetic algorithms, Estimation of Distribution Algorithms (EDAs) [1] have emerged as a robust optimization methodology. Instead of obtaining new solutions by traditional recombination and mutation operators, EDAs gradually develop and maintain a probability distribution of promising candidates. New offspring can be generated by sampling this underlying distribution. Bayesian Optimization Algorithm (BOA) [2] belongs to the class of multivariate EDAs. BOA is capable of tackling multivariate interaction problems decomposable into subproblems of bounded difficulty. Extended Compact Genetic Algorithm (ECGA) [3], Factorized Distribution Algorithm (FDA) [1], and Estimation of Bayesian Network Algorithm (EBNA) [4] are other examples of state-of-the-art EDAs. The strength of BOA is brought into the continuous world by Real-coded Bayesian Optimization Algorithm (rBOA) [5,6].

A standard BOA is composed of the following steps.

**Step 1:** Set $i \leftarrow 0$. Randomly generate first population $\mathcal{P}(0)$. Evaluate $\mathcal{P}(0)$.
**Step 2:** Select a parent set $\mathcal{S}(i)$ of promising solutions from $\mathcal{P}(i)$.
**Step 3:** Learn a Bayesian network $\mathcal{B}(i)$ from $\mathcal{S}(i)$.

N. García-Pedrajas et al. (Eds.): IEA/AIE 2010, Part II, LNAI 6097, pp. 126–135, 2010.

**Step 4:** Generate new solutions population $\mathcal{O}(i)$ by sampling from $\mathcal{B}(i)$.
**Step 5:** Evaluate entire $\mathcal{O}(i)$.
**Step 6:** Create $\mathcal{P}(i+1)$ by replacing some solutions of $\mathcal{P}(i)$ with $\mathcal{O}(i)$.
**Step 7:** Set $i \leftarrow i+1$. If the termination criteria are not met, go to **Step 2**.

BOA uses Bayesian networks [7] to model the regularities of the promising solutions. The process of learning a Bayesian network from a seleted population is concisely presented in Pelikan et al. [2]. A Bayesian network encodes the following joint probability distribution

$$p(X_0, X_1, ..., X_{n-1}) = \prod_{i=0}^{n-1} p(X_i|\Pi_i), \tag{1}$$

where each node $X_i$ is the $i^{\text{th}}$ random variable, $\Pi_i$ is the set of parent nodes of $X_i$ in the network (from each node of the set $\Pi_i$, there exists an edge directing into $X_i$), and $p(X_i|\Pi_i)$ is the conditional probability of $X_i$ given its parents $\Pi_i$. Each variable $X_i$ has a corresponding conditional probability table (CPT) storing its conditional probabilities concerning all possible values of $\Pi_i$.

Despite being a robust global black box optimizer, BOA has two potential bottlenecks involved in its process, the Bayesian networks construction and the fitness evaluations. In real-world application, the latter one costs a considerable amount of time and resources. Thus, various efficiency enhancement techniques for EDAs have been developed to reduce the number of (fitness) evaluations required for discovering the global optimum [8,9,10]. In this paper, a novel fitness evaluation relaxation strategy for BOA is proposed. To this end, the concept of the entropy measurement of (sub)populations is utilized.

The paper is organized as follows. Section 2 briefly describes related works covering various evaluation relaxation approaches for BOA. The concept of the entropy of a certain population and its inherent characteristics are discussed in Sect. 3. Our evaluation relaxation strategy is presented in Sect. 4. Experiments and results are shown in Sect. 5. Section 6 concludes the paper and highlights the future work.

## 2 Related Work

In BOA, the conditional dependencies between random variables are encoded in a directed acyclic graph. Taking advantage of such high level of abstraction, various techniques have been proposed to achieve evaluation relaxation for BOA.

Fitness inheritance in BOA [9] builds surrogate fitness models based on Bayesian networks. At each iteration, only a certain proportion of newly generated solutions are determined using the actual evaluation function. Fitness values of the remaining offspring are estimated by the partial contributions of each variable $X_i = x_i$ with respect to its parents $\Pi_i = \pi_i$ as follows

$$f_{\text{estimation}}(X_0, X_1, ..., X_{n-1}) = \bar{f} + \sum_{i=0}^{n-1} (\bar{f}(X_i|\Pi_i) - \bar{f}(\Pi_i)), \tag{2}$$

where $\bar{f}$ denotes the average value of all individuals used to construct the fitness model, $\bar{f}(X_i|\Pi_i)$ is the average fitness of solutions having a certain configuration of $X_i = x_i$ and $\Pi_i = \pi_i$, and $\bar{f}(\Pi_i)$ is the average fitness of solutions with a particular instance of $\Pi_i = \pi_i$.

While being proved to be a good interpolation for test problems, the construction of the above surrogate fitness model results in a greater population-sizing requirement. Furthermore, the algorithm's performance when enlarging the problem size has not been rigorously tested in the research [9].

BOA with substructural hillclimbing [10] utilizes the above-mentioned surrogate fitness model to estimate the fitness of a mutated individual at each step of the local search until it reaches a local optimum. While a reduction in the number of actual evaluations is achieved, this hillclimbing also requires larger population sizes. Besides, when the problem size is increased (when $l > 80$ as reported in [10]), the speed-up of the algorithm slows down.

In this paper, we measure the entropy value of the (sub)population at each iteration of BOA. Based on this measurement, a novel efficiency enhancement strategy is proposed to accelerate the BOA in terms of the number of evaluations.

## 3 Entropy Measurement of Populations

Based on the dependencies encoded in a Bayesian network, the entropy $H(\mathbf{X})$ of a certain population of BOA is derived as follows:

$$\begin{aligned} H(X_0, X_1, ..., X_{n-1}) &= \sum_{i=0}^{n-1} H(X_i|\Pi_i) = \sum_{i=0}^{n-1} \sum_{\pi_i \in \mathcal{Q}_i} p(\pi_i) H(X_i|\Pi_i = \pi_i) \\ &= -\sum_{i=0}^{n-1} \sum_{\pi_i \in \mathcal{Q}_i} p(\pi_i) \sum_{x_i \in \mathcal{X}_i} p(x_i|\pi_i) \log_2 p(x_i|\pi_i) \\ &= -\sum_{i=0}^{n-1} \sum_{\pi_i \in \mathcal{Q}_i} \sum_{x_i \in \mathcal{X}_i} p(x_i, \pi_i) \log_2 p(x_i|\pi_i) \\ &= -\sum_{i=0}^{n-1} \sum_{\pi_i \in \mathcal{Q}_i} \sum_{x_i \in \mathcal{X}_i} \left( \frac{m(x_i, \pi_i)}{N} \right) \log_2 \left( \frac{m(x_i, \pi_i)}{m(\pi_i)} \right), \quad (3) \end{aligned}$$

where $\mathcal{Q}_i$ is the set of all possible instances of $\Pi_i$ (parent nodes of $X_i$), $\mathcal{X}_i$ denotes the set of all possible values of $X_i$, $N$ is the population size, $m(x_i, \pi_i)$ is the number of individuals having $(X_i, \Pi_i)$ set to $(x_i, \pi_i)$, and $m(\pi_i)$ is the number of individuals having $\Pi_i$ equal to $\pi_i$. Research shows that such an entropy measurement can be used to determine convergence criteria for BOA [11].

Using (3), we can compute the entropy of some particular portions of the population with respect to the current network. Figure 1 shows that the entropy of the better half of the population (parents set) decreases after every iteration. When the BOA converges, this entropy value reaches its minimum value 0. This result is naturally logical as the BOA continuously narrows its sampling towards

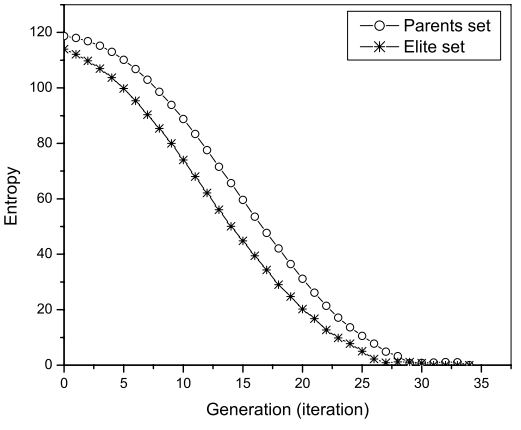


**Fig. 1.** Entropy reduction in a BOA solving 120-bit trap-5 problem

a certain promising region of the search space. The closer the algorithm moves to a specific convergence point, the more predictable it becomes. Thus, the inherent randomness of the probability distribution in the population gradually decreases.

We define an elite set as a set of the most promising individuals selected from the population. In Fig. 1, the top $\tau = 5\%$ of the population (in terms of fitness value) are selected as the elite set. This set also exhibits a tendency to decrease its entropy measurement after every iteration. Obviously, this regional entropy is always smaller than the overall entropy of the entire population, and it reaches the minimum value 0 sooner. From such observation, we conjecture that a promising candidate solution is an individual whose appearance in the elite set causes a reduction in the entropy measurement of that set. The next section describes how this conjecture is used in our evaluation relaxation strategy.

## 4 Entropy-Based Evaluation Relaxation Strategy for Bayesian Optimization Algorithm (eBOA)

### 4.1 The Algorithm

The proposed eBOA is described as follows:

**Step 1:** Initialization: Set $i \leftarrow 0, t \leftarrow 0$.
Randomly generate the initial population $\mathcal{P}(0)$. Evaluate $\mathcal{P}(0)$.
**Step 2:** Selection: Select a set $\mathcal{S}(i)$ of promising solutions from $\mathcal{P}(i)$.
Create an elite set $\mathcal{E}(i)$ of solutions from $\tau\%$ of $\mathcal{P}(i)$ having the highest actual fitness values.
**Step 3:** Model construction: Construct a Bayesian network $\mathcal{B}(i)$ fit for $\mathcal{S}(i)$.
Based on $\mathcal{B}(i)$, compute the entropy $\mathcal{H}(i)$ of $\mathcal{E}(i)$ by (3).
**Step 4:** Create a set of offspring $\mathcal{O}(i)$ by sampling $\mathcal{B}(i)$.
**Step 5:** Perform on-demand evaluation for $\mathcal{O}(i)$.

**Step 6:** Replace some solutions of $\mathcal{P}(i)$ by $\mathcal{O}(i)$ to create $\mathcal{P}(i+1)$.
**Step 7:** Set $i \leftarrow i+1$. If the termination criteria are not met, go to **Step 2**.

The on-demand evaluation strategy used in **Step 5** is defined as follows. Let $t$ be a counter variable, and $\kappa$ be the interval of sporadic evaluations.

**Case 1:** If $\mathcal{H}(i) \leq \frac{\mathcal{H}(0)}{2}$ and $t < \kappa$,
Consider each newly generated offspring $X$ of $\mathcal{O}(i)$,

**1.** Put $X$ into $\mathcal{E}(i)$ to create $\mathcal{E}'(i)$. Compute the entropy $\mathcal{H}'(i)$ for $\mathcal{E}'(i)$.
**2.** If $\mathcal{H}'(i) \leq \mathcal{H}(i)$, estimate

$$\mathrm{f}_{\mathrm{estimation}}(X) = f(Y), \qquad Y \in \mathcal{E}(i), \forall Z \in \mathcal{E}(i), f(Y) \leq f(Z) \ . \tag{4}$$

Otherwise, evaluate $f(X)$.
**3.** $t \leftarrow t+1$.

**Case 2:** If $\mathcal{H}(i) > \frac{\mathcal{H}(0)}{2}$ or $t = \kappa$,
**1.** Evaluate all offspring of $\mathcal{O}(i)$.
**2.** $t \leftarrow 0$.

Next, we explain the two main ideas of our entropy-based evaluation relaxation, *the on-demand evaluation strategy* and *the sporadic evaluation.*

### 4.2 On-Demand Evaluation Strategy (ODES)

Evaluating the fitness of a candidate solution is an expensive operation, thus we should only do so when necessary. As stated in Sect. 3, an individual is a promising candidate solution if its appearance achieves a reduction in the entropy value of the elite set. If an individual causes such an entropy reduction, it has the same characteristics (in terms of building blocks) with other candidate solutions in the elite set. Thus, it can be selected without being evaluated by the actual fitness function. Otherwise, the individual does not belong to the elite set, and it should be evaluated to obtain the correct fitness value.

If an individual is deemed to belong to the elite set, it should be assigned a high fitness value. We have performed various fitness assignment methods for a selected individual, such as assigning the fitness value of the closest individual (in Hamming distance) or using the median value of the elite set. However, their empirical results (which are not shown here) exhibit no significant difference. In this work, we choose to assign the fitness value of the worst candidate solution in the elite set to that individual. The justification for doing so is to minimize the severity of incorrect estimation errors. For the same reason, the elite set should only be selected from the candidates evaluated by the actual fitness function.

The condition $\mathcal{H}(i) \leq \frac{\mathcal{H}(0)}{2}$ denotes that we wait until the entropy of the elite set decreases by half of its original value before applying ODES. Before this juncture, the randomness (or unpredictability) of the elite set still remains high. If ODES is applied earlier, the estimation power is not strong enough to give good approximations. This would result in slower convergence and more

actual evaluations. On the other hand, if we delay the application of ODES until later iterations, the algorithm almost converges and we cannot achieve the maximal reduction in the number of fitness evaluations. Thus, the iteration $i$ when $\mathcal{H}(i) \leq \frac{\mathcal{H}(0)}{2}$ is a rational choice. Experiments supporting this choice are given in Sect. 5.

### 4.3 Sporadic Evaluation

The more our on-demand evaluation strategy is continuously used, the more estimation errors are accumulated. Even though our model improves its accuracy in later iterations, the appearances of incorrectly estimated individuals prevent the algorithm from convergence. One simple method to eliminate such accumulated errors is to sporadically perform complete evaluations of entire offspring populations. Instead of evaluating the whole population at each iteration as in the standard BOA, this should be performed only after every some generations have passed.

Methods with more complicated calculations can be developed to determine appropriate iterations to apply the sporadic evaluation. In this paper, however, we simply choose the criterion that after every $\kappa$ iterations, the actual fitness function should be used to evaluate all the newly generated offspring. The condition $t < \kappa$ in the above-mentioned algorithm denotes this criterion.

## 5 Experiments and Discussion

### 5.1 Test Problems

In this work, the OneMax function and the Concatenated 5-bit trap function are taken as the test problems. OneMax defines the fitness value of an individual as simply its sum of all the bits:

$$f_{\text{onemax}}(X_0, X_1, ..., X_{n-1}) = \sum_{i=0}^{n-1} X_i, \tag{5}$$

where $(X_0, X_1, ..., X_{n-1})$ is an input binary string of $n$ bits. The optimal solution for an $n$-bit OneMax problem is a binary string containing all 1s. Since OneMax is an easy problem for evolutionary algorithms, both BOA and eBOA must be able to work well on this problem.

A concatenated 5-bit trap is composed of several trap functions of order 5. Each trap-5 function is defined as follows:

$$f_{\text{trap5}}(u) = \begin{cases} 5 & \text{if } u = 5 \\ 4 - u & \text{if } u < 5 \end{cases}, \tag{6}$$

where $u$ is the number of bits having value 1 in a trap-5 function. The values of all the traps are added together to form the overall fitness value. An $n$-bit trap-5 function has one global optimum (a string of all 1s) and $(2^{n/5} - 1)$ local optima.

The difficulty in optimizing this function is that in each 5-bit trap function, all 5 bits have to be considered together. A robust EDA has to be able to discover the structure of the problem because all statistics of any lower orders lead the algorithm away from the global optimum.

### 5.2 Experimental Results

Both the OneMax and the Concatenated 5-bit trap problems are employed to test the standard BOA and eBOA. The problem sizes are enlarged from 30, to 60, 90, 120, and 150 bits. For each test problem and each problem size above, 30 independent experiments are performed. For each experiment, the bisection method is used to determine the minimal population size for the algorithm to obtain the optimal solution (with 100% correct bits). The convergence criterion is when the proportion of a certain value on each position reaches 99%. The truncation selection with $\tau = 50\%$ is used to select the better half of population as the parents set. The offspring replace the worse half of the old population. In all runs of eBOA, we select the elite set with the top $\tau = 5\%$ from the old population, and we perform sporadic evaluation after every $\kappa = 5$ iterations.

Figure 2 and Table 1 compare the performance of BOA and eBOA on the OneMax problem. On average, eBOA requires 85% of the number of fitness evaluations of BOA until convergence. Although in some test cases, the average number of evaluations of both algorithms are not significantly different, eBOA is still proved to be competitive with BOA in solving the OneMax problem.

Figure 3 and Table 2 compare the performance of standard BOA and eBOA on the trap-5 problem. The results prove that our eBOA achieves a significant reduction in the number of evaluations until convergence. On average, eBOA only needs 76% of the number of fitness evaluations of BOA. Furthermore, eBOA does

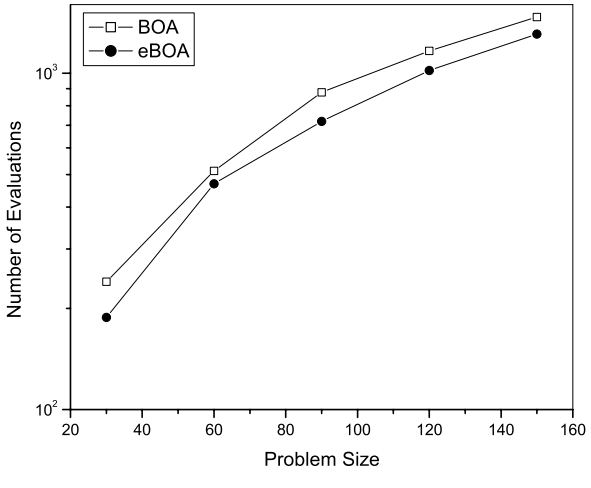


**Fig. 2.** Performance of BOA and eBOA on OneMax problem

**Table 1.** Statistical comparison of algorithms for OneMax problem

| Size | 30 | 60 | 90 | 120 | 150 |
|---|---|---|---|---|---|
| BOA | 240.6 | 512.4 | 877.0 | 1166.4 | 1470.4 |
| $\sigma$ | (55.6) | (75.2) | (191.7) | (204.7) | (235.2) |
| eBOA | 188.6 | 469.0 | 719.2 | 1019.9 | 1308.9 |
| $\sigma$ | (45.6) | (82.8) | (95.3) | (113.7) | (185.1) |
| $p$-value | Statistical $t$-test: (BOA - eBOA) | | | | |
| | 7.32E-4$^\dagger$ | 0.04619 | 8.55E-4$^\dagger$ | 0.00431$^\dagger$ | 1.02E-02 |

$^\dagger$ Significance by a paired, two-tailed test at $\alpha = 0.01$.

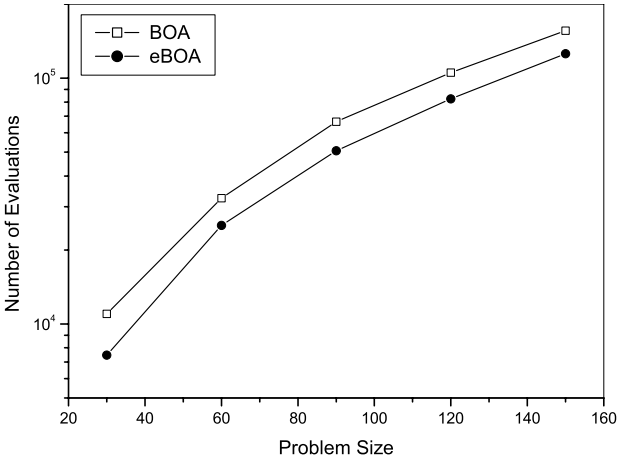


**Fig. 3.** Performance of BOA and eBOA on trap-5 problem

**Table 2.** Statistical comparison of algorithms for trap-5 problem

| Size | 30 | 60 | 90 | 120 | 150 |
|---|---|---|---|---|---|
| BOA | 11005.3 | 32496.4 | 66453.1 | 105245.1 | 155950.3 |
| $\sigma$ | (1519.4) | (3771.1) | (5118.9) | (13198.5) | (15729.4) |
| eBOA | 7465.6 | 25156.7 | 50683.8 | 82321.7 | 125699.6 |
| $\sigma$ | (1167.5) | (3984.8) | (5477.1) | (6297.1) | (14035.6) |
| $p$-value | Statistical $t$-test: (BOA - eBOA) | | | | |
| | 1.39E-11$^\dagger$ | 5.35E-07$^\dagger$ | 1.17E-12$^\dagger$ | 1.67E-09$^\dagger$ | 1.32E-08$^\dagger$ |

$^\dagger$ Significance by a paired, two-tailed test at $\alpha = 0.01$.

not compromise on scalability as the problem size increases. This experiment clearly supports the claim that eBOA outperforms the standard BOA.

Table 3 shows another interesting result obtained by eBOA in solving the trap-5 problem. While being able to reduce the number of evaluations, eBOA does not introduce any larger population-sizing requirements. Note that statistical tests demonstrate no significant difference between the population sizes of BOA and eBOA.

**Table 3.** Statistical comparison of population sizes for solving trap-5 problem

| Size | 30 | 60 | 90 | 120 | 150 |
|---|---|---|---|---|---|
| BOA | 999.1 | 2392.8 | 4404.6 | 6257.5 | 8489.6 |
| $\sigma$ | (163.4) | (278.9) | (461.1) | (1020.2) | (1016.4) |
| eBOA | 992.3 | 2632.4 | 4397.1 | 6146.5 | 8632.2 |
| $\sigma$ | (152.0) | (530.7) | (567.4) | (413.3) | (1160.8) |
| $p$-value | Statistical $t$-test: (BOA - eBOA) | | | | |
| | 0.85544 | 0.05682 | 0.95358 | 0.57186 | 0.60253 |

† Significance by a paired, two-tailed test at $\alpha = 0.01$.

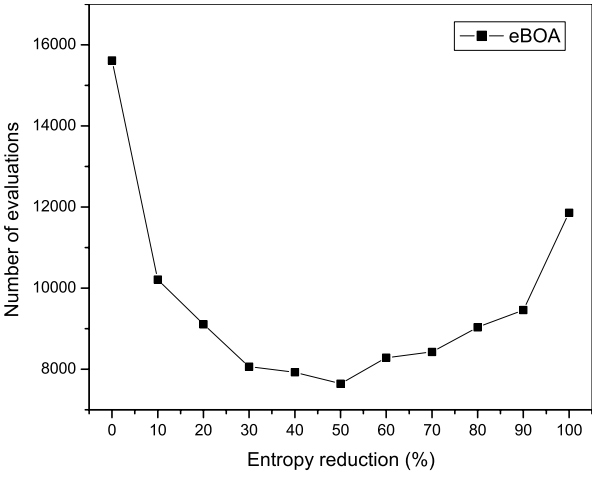


**Fig. 4.** Performance of eBOA with different starting points of ODES

Figure 4 provides experimental results to support our choice of the starting point to apply ODES as stated in Sect. 4.2. We perform experiments for eBOA solving the 30-bit trap-5 problem with different starting points. In terms of entropy reduction, 0% means to start ODES from the beginning of the optimization process, 100% indicates the standard BOA, and 50% comes under eBOA. When the entropy of the elite set decreases as a half of its original value, starting ODES achieves the minimal number of evaluations.

## 6 Conclusion

In this paper, we have presented eBOA with an entropy-based evaluation relaxation strategy. As we have observed in Sect. 3, the entropy value of the elite set gradually decreases as the algorithm moves towards the convergence point. Taking advantage of this inherent characteristic of the entropy value, we have proposed a recognition method to decide whether an individual needs to be evaluated by the actual fitness function or not. Experimental results have proved that

eBOA achieves a significant reduction in the number of fitness evaluations in comparison with the standard BOA. Moreover, it does not impose any larger population-sizing requirements. With a similar population size, the on-demand evaluation strategy can considerably accelerate the optimization process

In future work, we will investigate the effects of incorporating other estimation models into eBOA. Besides, we would like to bring the strength of on-demand evaluation strategy to other evolutionary algorithms. Such an entropy-based efficiency enhancement technique can contribute to a new line of research for EDAs in terms of evaluation relaxation.

**Acknowledgments.** This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MEST) (No. 2009-0066229). Dr. Ahn is the corresponding author.

## References

1. Pelikan, M., Goldberg, D.E., Lobo, F.G.: A Survey of Optimization by Building and Using Probabilistic Models. In: Computational Optimization and Applications, vol. 21, pp. 5–20. Kluwer Academic Publishers, The Netherlands (2002)
2. Pelikan, M., Goldberg, D.E., Cantu-Paz, E.: BOA: The Bayesian optimization algorithm. In: Proceedings of GECCO 1999, pp. 525–532. Morgan Kaufmann Publishers, San Francisco (1999)
3. Harik, G.: Linkage learning via probabilistic modeling in the ECGA. Technical Report No. 99010. IlliGAL (1999)
4. Larrañaga, P., Lozano, J.A.: Estimation of distribution algorithms: A new tool for evolutionary computation. Kluwer Academic Publishers, Boston (2002)
5. Ahn, C.W., Goldberg, D.E., Ramakrishna, R.S.: Real-coded Bayesian Optimization Algorithm: Bringing the Strength of BOA into the Continuous World. In: Deb, K., et al. (eds.) GECCO 2004. LNCS, vol. 3102, pp. 840–851. Springer, Heidelberg (2004)
6. Ahn, C.W., Ramakrishna, R.S.: On the Scalability of Real-Coded Bayesian Optimization Algorithm. IEEE Transactions on Evolutionary Computation 12(3), 307–322 (2008)
7. Pearl, J.: Probabilistic reasoning in intelligent systems: Networks of plausible inference. Morgan Kaufmann, San Mateo (1988)
8. Sastry, K., Lima, C.F., Goldberg, D.E.: Evaluation Relaxation Using Substructural Information and Linear Estimation. In: Proceedings of GECCO 2006, pp. 419–426. ACM Press, New York (2006)
9. Pelikan, M., Sastry, K.: Fitness Inheritance in Bayesian Optimization Algorithm. In: Deb, K., et al. (eds.) GECCO 2004. LNCS, vol. 3103, pp. 48–59. Springer, Heidelberg (2004)
10. Lima, C.F., Pelikan, M., Sastry, K., Butz, M., Goldberg, D.E., Lobo, F.G.: Substructural Neighborhoods for Local Search in the Bayesian Optimization Algorithm. In: Runarsson, T.P., Beyer, H.-G., Burke, E.K., Merelo-Guervós, J.J., Whitley, L.D., Yao, X. (eds.) PPSN 2006. LNCS, vol. 4193, pp. 232–241. Springer, Heidelberg (2006)
11. Ocenasek, J.: Entropy-Based Convergence Measurement in Discrete Estimation of Distribution Algorithms. Studies in Fuzziness and Soft Computing, vol. 192, pp. 39–50. Springer, Heidelberg (2006)