Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2013, Article ID 132697, 11 pages
http://dx.doi.org/10.1155/2013/132697

Research Article

Cyber-EDA: Estimation of Distribution Algorithms with Adaptive Memory Programming

Peng-Yeng Yin and Hsi-Li Wu

Department of Information Management, National Chi Nan University, Nantou 545, Taiwan

Correspondence should be addressed to Peng-Yeng Yin; pyyin@ncnu.edu.tw

Received 10 May 2013; Accepted 20 September 2013

Academic Editor: Yang Xu

The estimation of distribution algorithm (EDA) aims to explicitly model the probability distribution of the quality solutions to the underlying problem. By iterative filtering for quality solution from competing ones, the probability model eventually approximates the distribution of global optimum solutions. In contrast to classic evolutionary algorithms (EAs), EDA framework is flexible and is able to handle inter variable dependence, which usually imposes difficulties on classic EAs. The success of EDA relies on effective and efficient building of the probability model. This paper facilitates EDA from the adaptive memory programming (AMP) domain which has developed several improved forms of EAs using the Cyber-EA framework. The experimental result on benchmark TSP instances supports our anticipation that the AMP strategies can enhance the performance of classic EDA by deriving a better approximation for the true distribution of the target solutions.

1. Introduction

Since the 1960s, the idea of bioinspired computation has created fruitful implementations for evolutionary algorithms (EAs). Among them, the genetic algorithm (GA) is one of the most remarkable notions that draw on Darwinian Theory. Researchers have developed successful applications by using GAs in the domains where the classic analytical and numerical methods are not suitable to be applied. A typical example is the optimization problems with the black-box evaluation for the solution quality. This is commonly seen in real world practice that no analytical expression is available for computing the solution quality. Instead, a black-box procedure running a simulation model of the problem is applied to estimate the fitness of the trial solution. The use of evolution metaphor and black-box estimation has promoted the popularity of GA as being a viable approach for tackling complex and unstructured problems.

Holland [1] developed the schema theorem for building the mathematical foundations for GAs. The schema (a pattern of gene alleles) having constantly above-average fitness, shorter defining length, and lower order will be more likely to survive against genetic operations and reproduce copies of themselves at an exponential rate along the evolutionary generations. This ideal assumption leads to a fast convergence in identifying the building blocks for constructing the global optimal solution. However, the simple genetic algorithm (sGA) is plagued by the deceptive problem which indicates that some useful building blocks may be discarded through successive genetic operations. This phenomenon happens when the significance of the evaluated fitness for schema at shorter defining length and lower order contradicts with that measured at longer defining length, and higher order, thus drawing the evolution away from the global optimal solution. Discovering the dependence relationships between alleles in the chromosome is a vivid research direction for preventing sGA from trapping by the deceptive problem.

The estimation of distribution algorithm (EDA) intends to explicitly model the probability distribution of uniformly sampled solutions from those which have a fitness value greater than a threshold. Through selection on elitism to form the next population, the fitness threshold is increasingly strict along the number of generations until the elite individuals in the population cannot be further improved. EDA thus iteratively approximates the probability distribution of the global optima. The probability model can render the

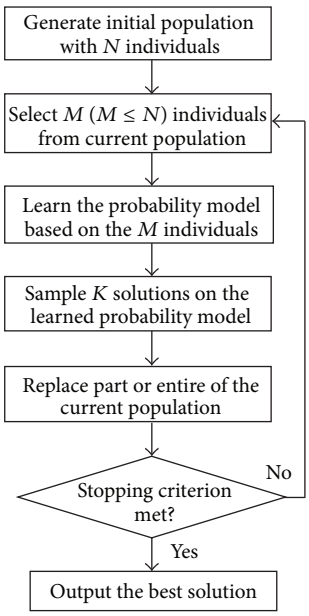

Figure 1: Flow diagram of the EDA framework.

dependence between variables, and the linkage relationship is reserved during sampling on the model. The variable-dependence modeling is performed through the factorization technique. Mutually dependent variables are grouped as a factor. The solution is considered as a set of nonoverlapping factors where the variable is dependent on the remaining variables contained in the same factor but is independent of those variables belonging to the other factors. Then the conditional probabilities for each factor are computed, and their multiplications form the final aggregate probability model. The success of EDA relies on if the probability model is learned efficiently and accurately by the algorithm.

The source of the original idea for developing the EDA can be dated back to 1994 [2] though the term EDA first appeared in 1996 [3]. The EDA replaces the implicit biological operations performed in classic EAs by building a probabilistic model to describe the population. The previous research on EDA majorly focuses on three challenging issues. (1) Select/learn a probability model that accurately describes the variable dependence of the addressed problem, (2) develop a procedure that effectively learn the parameters of the probability model, and (3) hybridize the EDA with search heuristics developed in other domains to obtain the result in shorter time. In this paper, we consider the three issues all together because they are interrelated and we believe that the performance of EDA can be significantly enhanced from an integrated system view.

We construct our method primarily by drawing notions from the domain of the adaptive memory programming (AMP) [4] which has shown to be one of the most effective forms of metaheuristics. The AMP is a suite of algorithms which utilize adaptive memory structure to record the temporal changes on the status of the produced solutions, and responsive strategies are executed in accordance with the status changes. The AMP is effective in locating influential solutions (in improving the future solutions) which are usually hardly produced by using classic EAs. Notable AMP algorithms are tabu search (TS) [4], scatter search (SS) [5], path relinking [6], ejection chain [7], and greedy randomized adaptive search procedure (GRASP) [8], to name a few. Recent literature [9–11] has shown that the performance of EAs can be significantly improved by using the AMP techniques. One of the remarkable forms for combining EAs and AMP is the Cyber-EA framework [12] which has created several effective algorithms [9, 13–15]. The contributions of this paper are as follows. (1) In contrast to previous EDA research, our improvements on the EDA algorithm are made from a system point of view. Elaborated learning strategies have been proposed for each EDA component. (2) Under the Cyber-EA framework, various manipulations on adaptive memory and responsive strategies are proposed. (3) Solid statistical and property analyses are conducted for providing insights into the interactions between different strategies.

The remainder of this paper is organized as follows. Section 2 presents a literature review of the EDA methods and the AMP techniques relevant to our work. Section 3 describes the proposed method. Section 4 presents experimental results on performance evaluation. Finally, concluding remarks are given in Section 5.

2. Literature Review

2.1. Learning Issues in EDA. The flow diagram of the EDA framework is illustrated in Figure 1. Spiritually similar to most EAs, the EDA algorithm improves the quality of the current population through generational update. The main difference is the use of the probability model which replaces the reproduction component of classic EAs by sampling solutions on the probability model. The EDA framework consists of four major components, namely, the selection,

learning, sampling, and replacement. The selection component selects a subset (with size M) from the current population (with size N) to form a quality pool of samples for the subsequent update of the probability model. The selection is usually performed on a greedy selection basis; that is, the best M individuals in terms of fitness are retained in the pool. The learning component constructs the probability model by updating the probabilities through learning from the observations on the samples contained in the pool. The sampling component reproduces K new samples from the learned probability model. Finally, the K samples are used to replace part or entire of the current population in the replacement component. The four components are iterated until the program reaches the satisfaction of the specified stopping criterion which is usually set as that the evolution reaches a maximal number of fitness evaluations or the population has not changed for a large number of fitness evaluations.

As previously noted, there are three learning issues in previous EDA research. The first issue investigates the selection of an appropriate probability model that fits the characteristic of the addressed problem. To model the variable (in)dependence relationship in a problem is a difficult task and could be an optimization process itself. Grahl et al. [16] classified the variable (in)dependence relationship into three situations: no interactions between variables, interactions between two variables, and interactions between multiple variables. The factorization technique is a general scheme for modeling the three situations. The variables contained in a solution are divided into disjoint factors. The variable is dependent on the remaining variables contained in the same factor but is independent of those variables belonging to the other factors. Provided that a solution F has m factors F_i, $i = 1, 2, \ldots, m$. The probability distribution of observing F is estimated by the product of density for each factor as follows:

$$P(F) = \prod_{i=1}^{m} P_{\theta_i}(F_i), \tag{1}$$

where θ_i is the vector of parameters for the probability density of factor F_i. The simplest form of the probability model decomposes the multivariate density into a product of univariate densities (i.e., no interaction between variables). Early variants of EDA, such as of PBIL [2], UMDA [3], and cGA [17], stick to this no-interaction model. These methods deal with binary discrete optimization and learn a probability vector whose element value indicates the probability of each binary variable of being 1. This probability model has proved to be able to solve to optimality the unimodal optimization problems, but it is limited to be applied to solve problems involving multiple modals. On the other hand, the probability model accommodating bivariate or multivariate interactions is theoretically able to describe more complex characteristics of the problems, but it is computationally expensive and it may incur spurious variable relationships. The EDA variants, including MIMIC [18], COMIT [19], and BMDA [20], employed bivariate factors to model the interactions between two variables. More recent EDA algorithms, such as ECGA [21], BOA [22], hBOA [23], EBNA [24], and LFDA [25], aim to model interactions between multiple variables.

The second issue relates effective methods for learning the parameters of the adopted probability model. The methods for constructing the probability model can be divided into nonparametric and parametric approaches. Nonparametric approaches are adopted for dealing with combinatorial optimization problems. The approaches, such as the probability vector, marginal histogram model, tree structure, directed acyclic graph, and Bayesian network, consider the relative frequency of each bit value stored in the frequency table as the probability parameters. The complexity of computations and memory storage depends on the cardinality of each variable factor and the learning schemes themselves. For example, PBIL employs competitive learning for shifting the probability vector towards representing those in high evaluation sampled solutions. The BOA uses the previously built Bayesian networks to predict the next network structure. On the other hand, parametric methods are adopted for modeling the continuous optimization problems and these methods usually rely on variants of the normal probability density function (pdf). The extension of some discrete models has been developed for this purpose. PBILc [26] and UMDAc [27] are adaptations of PBIL and UMDA to continuous domains. Following the no-interaction model, they apply the maximum-likelihood-estimate method to derive mean and variance for the normal pdf of each variable. MIMICc [28] is adapted from MIMIC by extending the Bayesian networks to Gaussian networks which can handle conditional normal pdf. The Iterated Density Estimation Evolutionary Algorithm (IDEA) [29] uses mixture distribution of multiple normal pdfs to describe the possible multimodal nature of the search function, so IDEA is able to concentrate on more than a single peak in the continuous domains.

Finally, the third issue of EDA research contemplates that the performance of EDA algorithms can be significantly improved if they are hybridized with appropriate search heuristics in other domains. Handa [30] proposed the niche separation mechanism to split converse schemata into two subpopulations (niches) such that when the incumbent population is converged, the other can be used for substitution. Ahn and Kim [31] developed an extended compact particle swarm optimization (EcPSO) which combines particle swarm optimization and EDA to reserve their unique features. Zhang et al. [32] created an EDA hybrid by embedding two local search heuristics into the EDA framework and called their algorithm EDA/L. The simplex local search is applied to every sampled solution and the UOBDQA local search is used to improve selected solutions obtained by the simplex method.

2.2. AMP Strategies. Glover [33] first coined the Tabu search and developed it to a broader domain called the adaptive memory programming (AMP) [4]. The AMP investigates the usage of adaptive memory to record the solution status variations along the search course and then intrigues the execution of particular search strategies in order to respond to the status dynamics and obtain better solutions. Several useful AMP techniques have been proposed in the scatter

search and path relinking template (SS/PR template) [6]. AMP can handle complex landscape in the objective space, especially when the local search process stagnates by local optimality. There are two major types of strategies proposed in the AMP domain to alleviate this ailing situation. The first method is used to escape from the local optimum by employing complex neighborhood concepts (such methods as path relinking, improvement method, and ejection chain) and resume the exploitation search. The second approach is to restart the search course with a new initial seed solution contained in a region which is not charted yet in the search history (such methods as diversification generation, reference set rebuilding, and restarting). Many evidences reported in the literature have shown that AMP is useful for enhancing evolutionary algorithms. Yin et al. [9] developed the Cyber Swarm Algorithm which is an enhanced form of swarm algorithms by incorporating AMP features into the particle swarm optimization (PSO). Nakano et al. [10] hybridized PSO with tabu search to implement intensification and diversification searches more effectively. Taillard and Gambardella [11] showed that ant systems and genetic algorithms can be improved by AMP in solving the quadratic assignment problem. Kessentini et al. [34] proposed a nonuniform adaptive PSO for rapid resolution of plasmonic biosensor problem and compared favorably to several PSO variants. Maquera et al. [35] presented an application of AMP strategies to the vehicle routing problem with simultaneous delivery and pickup goods. Hassannezhad and Javadian [36] developed a self-adaptive differential evolution (DE) for designing the cellular manufacturing systems and showed the robustness of their algorithm by testing on a set of 25 cell formation instances. Due to the previous success of EDA and AMP, we propose to enhance the performance of EDA by exploiting strategies that have been well established in the AMP domain.

In addition to responsive strategies, another feature of AMP is the use of adaptive memory mechanism which is further classified as short-term and long-term memory. The short-term memory stores the values found in recent status such as the current population and the forbidden solutions in the tabu list. The long-term memory monitors the status changes in a longer term of time duration. Previous researches have suggested, for example, the frequency of dominant variable values, the reference solutions with both quality and diversity properties, and the duration in which the best solution has not improved. The reference set stores long-term elite solutions with respect to a threshold for the minimal mutual distance between these elite solutions. The reference set has a fixed size. The worst solution x contained in the reference set is replaced by a new elite solution y if the quality of x is worse than that of y and y satisfies the minimal mutual distance constraint. The size of reference set is much smaller than that of the population size. This is to avoid the massive computations due to the systematic, instead of stochastic, way of utilizing the members in the reference set.

3. Proposed Method

In contrast to the development conception of previous EDA variants, our proposed method enhances EDA from a system point of view. Recall the flow diagram of the EDA framework (Figure 1), there are four major components, namely, the selection, learning, sampling, and replacement. Each of the four components can be contemplated for improvement by intriguing strategies from the AMP optimization domains. In this paper, we propose the Cyber-EDA of which the new features for constructing each EDA component are summarized in Table 1 and will be articulated in the following subsections.

Table 1: The Cyber-EDA strategies for enhancing EDA components.

EDA component	Cyber-EDA strategy
Selection	Diversification generation
	Improvement method
Learning	Thresholding with short-term memory
	Incremental learning with long-term memory
	Probability model rebuilding with long-term memory
Sampling	Weighting sampling scheme
Replacement	Generational population update with short-term memory
	Restart population rebuilding with long-term memory
	Partial population rebuilding with long-term memory

3.1. Cyber-EDA Selection Component

3.1.1. Diversification Generation. In the original SS/PR template, the diversification generation method is used for constructing the initial or rebuilt populations and the update of the reference set. The aim is to preserve contrasting solutions in a hope to produce promising solutions that are unobtainable by using greedy selection. Analogously, we apply the diversification generation method for the EDA selection component to obtain M diverse individuals from the initial or rebuilt populations. It is worth noting that the diversification generation method is not applied to the generational populations such that the evolutionary information is preserved. The diversification control can be facilitated by different means. We can directly generate a diversified population by applying the Taguchi method which is broadly used in the experimental design field. The Taguchi method provides orthogonal arrays that can generate representative and mutually diversified samples. Another alternative is to first generate a larger population with N random individual solutions. Then a smaller subset with M diversified solutions can be produced from the population by the max-min distance principle as follows. For each of the N solutions in the population, the minimum distance from it to each of the remaining solutions in the population is computed. Then, the solutions are sorted in descending order of the minimum distance. The top ranked M solutions in the order are thus selected.

3.1.2. Improvement Method. Improvement method is a local search heuristic that has been included as a mandatory component in several effective metaheuristic algorithms such as the Scatter Search, GRASP, and Memetic Algorithms. The improvement method pushes the trial solutions towards nearby local optima, and the schema contained in the local optima is helpful in constructing promising solutions. We anticipate that the improvement method can be also used to enhance the performance of EDA. The learned probability model is inevitably biased due to the limit of the population size and the process of the sampling. The classic EDA intends to reduce the bias through successive generations of greedy competition among individuals. However, this may lead to a lengthy process. The improvement method can expedite this competition process by making it start with a population of diverse local optima and also by maintaining a reference set which contains quality and diverse solutions. Consequently, in the proposed Cyber-EDA, the improvement method is performed at two places. First, it is executed in succession of the diversification generation method; thus, the members in the initial or rebuilt populations are drawn towards local optima. Second, the improvement method is applied to the new member(s) of the reference set during the reference set update process. The reference solutions are used for partial population rebuilding upon the critical event as will be noted. There are several effective combinatorial improvement methods in the AMP literature such as the insertion method, n-opt, ejection chain, and path relinking, to name a few. These methods can be separately used as a single improvement method or they can be combined as a compound one with a more complex neighborhood structure.

3.2. Cyber-EDA Learning Component. The probability model learning component in Cyber-EDA makes use of memory manipulations as has been contemplated in the AMP domain. The Cyber-EDA has two types of memory structures. The short-term memory tallies the probability distribution describing the new population. The long-term memory keeps records of both the probability model that is learned through successive populations and the AMP parameters which are used to guide the evolution conducted between successive restarts. The two types of memory manipulations are described as follows.

3.2.1. Thresholding with Short-Term Memory. To reduce the sampling bias incurred by the limited-size of the population and the sampling process, we propose the thresholding technique to manipulate the short-term memory which records the frequencies of the solutions selected from the current population. The thresholding technique restrains the highest frequency by an adaptive threshold and uniformly distributes the extra counts to the remaining frequency beams. As a result, the solution with the highest frequency will not dominate the probability model and we can avoid the possible oversampling of this solution. The adaptive threshold is tuned according to the number of executed function evaluations, which indicates the completion percentage of the planned evolution. Let T and t be the total number and the current number of the function evaluations. The current adaptive threshold value τ_t is computed as follows:

$$\tau_t = \tau_{t-1} + \left(\eta \frac{t}{T}\right). \tag{2}$$

The current value τ_t of the adaptive threshold is obtained by adding an increment to the previous threshold value τ_{t-1}. This increment is proportional to the completion percentage of the planned overall evolution where η is a constant for the relative scaling between the number of function evaluations and the threshold value. Hence, the adaptive threshold value is incrementally increasing within the evolution process and the increasing rate is also incrementally raised. This design allows the evolution to focus on exploration at the early stage by restraining the highest frequency of the sample solution. As the evolution proceeds, the adaptive threshold value is incrementally higher, which allows the evolution transition from exploration to exploitation of promising solutions.

3.2.2. Incremental Learning with Long-Term Memory. In classic EDA, the previous experience obtained through evolution is retained in two memory structures, the sample population, and the probability model. These two structures are interrelated and are updated along the evolution. In the machine learning domain, incremental learning suggests to target the objective function by combining immediate experience and previous experience. The previous experience is discounted to favor the importance of latest experience from the most immediate observations. The incremental learning technique has proved effective in many applications [37]. We thus implement the incremental learning scheme in the probability model building component. In our Cyber-EDA algorithm, a learning rate $r \in (0, 1)$ is used to control the relative importance of the probabilistic model built at different times. In particular, let $p(v)$ be the probability of observing solution v in the previous probability model and let $q(v)$ be the probability of v in the probability model built by the current population, and the probability model for generating the next pool of samples is incrementally updated as follows:

$$p(v) \longleftarrow (1 - r) \times p(v) + r \times q(v). \tag{3}$$

Hence, the probability model will be more significantly affected by the distributions of the current population than those of previous populations if r is close to 1.

Similar to the tuning of the adaptive threshold, we use a decreasing value for r as the number of function evaluations increases in order to make the incremental learning focus on exploration at the early stage and gradually transit to exploitation. In particular, the learning rate r is adaptively tuned by the following equation:

$$r_t = \max\left(r_{\min}, r_{t-1} \times \left(1 - \frac{t}{T}\right)\right), \tag{4}$$

where r_t is the value for the learning rate performed at t function evaluations and $r_{\min}$ is the minimum value for the learning rate. In other words, the value of r_t is monotone-decreasing until it reaches $r_{\min}$.

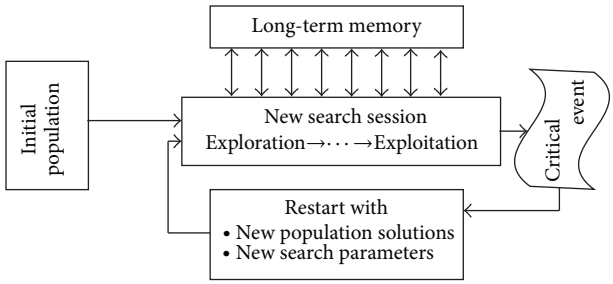

Figure 2: General concept of the AMP restart search strategy.

3.2.3. Probability Model Rebuilding with Long-Term Memory. One of the prominent search strategies in the AMP domain is the restart strategy with long-term memory. A common challenging issue to all of the evolutionary algorithms is the premature convergence problem, indicating that the best, however not acceptable, individual solution found so far has not improved for a large number of function evaluations. By referring to long-term memory, the restart strategy aims to guide the search towards an uncharted space. The restart strategy reinitiates a new search session by rebuilding the reference solutions, and the new search session may use a different set of the search scheme parameters in contrast to the previous session. The rebuilt solutions must have contrasting features that are not contained in previously found solutions and the employment of new values for search scheme parameters can create different neighborhood concept for the new search session. The general concept of restart search strategy is shown in Figure 2. In the same line, our Cyber-EDA algorithm adopts the restart strategy to rebuild the probability model in order to escape the trap from local optima.

In the context of EDA, restart might mean the probability model rebuilding. Its necessity emerges from the sort of critical event in which all of the new sampled solutions cannot replace any of the solutions in the population for a sufficiently large number of generations. The critical event can be detected by referring to the long-term memory which keeps record of population replacement status along generations. The restart strategy for the probability model rebuilding is facilitated in two aspects. First, the population is rebuilt by using new solutions which possess mutual diversity. This process is referred to as population rebuilding and will be articulated in description for the Cyber-EDA replacement component. Second, the system parameters can be adaptively tuned in accordance with the status of the long-term memory. These parameters include either or both the probability model parameters (i.e., probabilities) and the search strategy parameters. The probabilities can be either retained or reset to null value and learn from scratch in the new restart session. For the latter case, we do not need to worry about the performance deterioration due to the abandon of the probabilities because the elite solutions found in the search history have been tallied in the reference set. On contrary, the reconstruction of the probability model may be beneficial to future findings of potential solutions which are overlooked from the previous search trajectories. As indicated in Figure 2, the restart strategy splits the entire evolutionary process into separate search sessions. It is desired to have each session start with more emphasis on exploration search seeking for uncharted space and then gradually change to exploitation search for local optima. The adaptive tuning of the Cyber-EDA search strategy parameters can produce the desired search sessions. As noted, the parameter τ_t of the thresholding technique and the parameter r used by the incremental learning are adaptively tuned for determining the relative contribution from various sources of experience. Once a new search session is created by the restart strategy, we reset τ_t and r to their initial value and thus adapt the search strategy to focus on the exploration of new experience.

3.3. Cyber-EDA Sampling Component. Most EDA algorithms perform the probabilistic sampling according to the learned probability model to generate new solution samples. The contribution of every sample for the update of the probability model is the same. We consider a more aggressive form of sampling, which we call weighting sampling scheme, in the Cyber-EDA. Inspired by the success of the Cyber Swarm Algorithm [9] which aggressively scales the contribution from various local guides according to their fitness quality, and the Cyber-EDA gives multiplication of contribution to high-fitness samples. It is in a sense analogous to the situation in which the solution is sampled multiple times from the probability model, but actually the solution is only sampled once. This scheme is especially useful for solving the under-sampling problem. The proposed weighting sampling scheme gives each sampled solution s_i a weight w_i by the following equation:

$$w_i = \delta \times \frac{f_i}{\sum_{j=1}^{K} f_j}, \tag{5}$$

where f_i is the fitness of solution s_i, K is the number of sampled solutions, and δ is a constant for determining the contribution of a solution with average fitness.

3.4. Cyber-EDA Replacement Component

3.4.1. Generational Population Update with Short-Term Memory. The Cyber-EDA applies the generational population update in the same way as that performed by most EDA algorithms. The sampled solutions obtained from the sampling component compete for survival with all the solutions in the current population. The survival competition is conducted based on a greedy selection on fitness, such that the best fitness solutions will form the next population. The generational population update is essential to the continuing improvement on the lower bound of the fitness for all the solutions in the current population. Hence the estimation of the distribution converges to that for the global optima. As the sampled solutions are temporarily stored in the short-term memory, they are removed after the replacement component process unless the sampled solutions satisfy the quality and diversity test for updating the reference set.

3.4.2. Population Rebuilding with Long-Term Memory. In contrast to the generational population update which is conducted at every generation by using the sampled solutions stored in the short-term memory, the population rebuilding is activated upon signal of critical events which are detected by reference to long-term memory. The Cyber-EDA facilitates two types of population rebuilding, namely, the restart population rebuilding and the partial population rebuilding, as presented as follows. First, the restart population rebuilding is part of the noted restart strategy which reinitiates a new search session. It is activated at the critical event when the previous search session stagnates in improving the best solution for α_1 successive iterations. In order to generate a population which has contrasting features to previous populations, a larger population with N random solutions is generated. Then a smaller subset with M diversified solutions is produced from the population by the max-min distance principle as previously noted in the diversification generation method.

Second, the partial population rebuilding is activated at a less critical event in which the best solution has not improved for α_2 successive iterations and $\alpha_2 < \alpha_1$. The purpose of partial population rebuilding is not for restart of another search session. Instead, it replaces part of the current population with diverse and quality solutions and tries to make the current search session more effective in improving the best solution. The diverse and quality solutions can be produced by using the reference set. The SS/PR template [6] has proposed a solution generation method which proceeds as follows. First, all the 2-element subsets of the reference set are generated. For each 2-element subset, one solution is selected to be the initiating solution and another solution is designated as a guiding solution. Then the initiating solution is transformed into the guiding solution by generating a succession of moves that introduce attributes from the guiding solution into the initiating solution. When relinking the initiating solution to the guiding solution, several trial solutions along the path will be produced. The best fitness solution of them is used for partial population update.

4. Experimental Results and Analysis

To evaluate the performance of the proposed Cyber-EDA algorithm, we have chosen the well-known traveling salesman problem (TSP) as the optimization test benchmark. Twenty problem instances covering different scales of problem size from 48 to 400 cities are selected from the TSPLIB. The platform for conducting the experiments is a personal computer equipped with a 2.26 GHz CPU and 4.0 GB RAM. All programs are codified in C# language. We have determined the parameter values for all the compared algorithms through the experimental design method. For each parameter, several typical values are tested with a subset of the dataset. The parameter value leading to the best result is adopted in subsequent experiments. In particular, for the parameter setting of all the compared EDA-based algorithms, the marginal histogram model with bivariate interactions is employed as the probability model; that is, the joint probability of any two cities observed in succession within the visiting route is estimated. The population size N is equivalent to the problem size (the number of city nodes). In the selection component, $M = \lfloor N/2 \rfloor$ best solutions are selected from the current population to be fed into the learning component for updating the probability model. In the sampling component, $K = \lfloor N/4 \rfloor$ solutions samples are produced from the learned probability model. The stopping criterion for execution of all tested programs is set to having performed 200,000 solution fitness evaluations. The additional parameters used by the Cyber-EDA algorithm are set as follows. The reference set adopted as the long-term memory used in the Cyber-EDA accommodates $\lfloor N/5 \rfloor$ quality and diverse solutions ever observed in the search history. The 2-opt local heuristic is employed as the improvement method in the selection component. The initial value of τ and r is set to $\lfloor N/4 \rfloor$ and 0.5, respectively. The minimum learning rate $r_{\min}$ is 0.1. The two criteria for the critical event are set as $\alpha_1 = 40{,}000$ and $\alpha_2 = 20{,}000$.

As for the time complexity of all the compared EDA variants, most researchers working in the EDA domain use the number of the solution fitness evaluations as the measure of the time complexity because the solution fitness evaluation is often the most time-consuming component in the designed algorithms. We adopted the same approach in our experiments and fixed the number of the solution fitness evaluations to 200,000 for all the compared EDA variants. By fixing the number of the solution fitness evaluations, the effectiveness for obtaining the global optimal solutions by each competing algorithm can be observed. Moreover, as for the space complexity issue, the Cyber-EDA uses an extra long-term memory structure called the reference set which intends to reserve the information diversity of the population and the probability model. The size of the reference set is negligible since the population size varies from 48 to 400 depending on the problem instance.

4.1. Comparative Performances. In this section we present the comparative performances of the classic EDA and the Cyber-EDA. Because the Cyber-EDA is coupled with various AMP strategies, we have conducted preliminary experiments to identify the principal strategies which significantly affect the performance of the Cyber-EDA. We found that such strategies as the diversification generation, improvement method, incremental learning, and the weighting sampling scheme are the principal strategies and, if they are employed simultaneously, can improve the performance of the classic EDA by at least 30% over all of the test problem instances. We deem these principal strategies as the default strategies as our baseline Cyber-EDA, which we denote as E^*. Next, we proceed with the performance evaluation for the baseline Cyber-EDA with the adaptive thresholding technique (E^*_T), the partial population rebuilding with reference set (E^*_{ref}), the restart population rebuilding (E^*_{retart}), and the full-set Cyber-EDA with all the proposed AMP strategies.

All the compared EDA variants are stochastic method. Each single run of the same algorithm may produce different results depending on the random values generated. In order

to conduct a fair performance comparison, each compared algorithm is executed 30 times on every problem instance with various initial random numbers and the mean and the standard deviation of the TSP tour cost calculated over these multiple runs are used as the performance indices. The mean cost reflects the quality level of the solution obtained by an algorithm and the standard deviation indicates the stabilization of the corresponding algorithm to obtain such a solution. The experimental results on the 20 TSP problem instances are listed in Table 2. The mean and the standard deviation (Std) for the cost of the obtained tour are calculated over 30 independent runs. It is observed that all of the Cyber-EDA variants significantly outperform the classic EDA in terms of the mean cost on the whole test problem set. As for the three Cyber-EDA test strategies, it seems that the adaptive thresholding technique (E_T^*) has minor contributions compared to those by the two population rebuilding strategies (E_{ref}^* and E_{retart}^*). The E_T^* strategy may be effective for very small problem instances such as Att48, Berlin52, Eil51, and Eil76. This is because small size instances are more likely to suffer oversampling of a particular solution, and this sampling bias has been avoided by the E_T^* strategy. However, for the larger problem instances, the effect of undersampling in some regions of the solution space is more profound. In this case, the two population rebuilding strategies provide fruitful sources of diversified and quality solutions to guide the search towards uncharted space. The distinct properties of E_{ref}^* and E_{retart}^* will be noted by the epoch analysis in the next subsection. Finally, the Cyber-EDA algorithm which employs the full set of all the proposed AMP strategies produces the best mean cost in 15 out of the 20 problem instances. The high percentage of winning by the Cyber-EDA algorithm indicates the proposed AMP strategies cooperate well under the EDA framework.

4.2. Epoch Analysis. In this section we present the epoch analysis for the critical events observed during the simulation runs. Recall that the Cyber-EDA uses two distinct levels of critical event to trigger the restart population rebuilding and the partial population rebuilding. The two distinct levels indicate two different lengths of stagnation in improving the best solution. The restart population rebuilding is designed for escaping the longer stagnation by creating a new random population with diversification control. The epoch analysis is shown in Figure 3. Since the start of the evolution, the cost of the best solution in the current population decreases as the number of function evaluations increases, but before long it stops improving the best solution at around 30,000 function evaluations. This stagnation phenomenon lasts for a sufficiently long period (α_1 successive evaluations) and satisfies the restart critical event. The restart strategy is thus activated and it rebuilds the entire population. It is worth noting that the best solution in the rebuilt population may have a very worst cost, but the high diversity among the solutions in the rebuilt population and the new settings of the system parameters by the model rebuilding guide the search towards new space and increase the likelihood to escape from local optima. It is seen in the figure that after the first epoch the cost of the best solution dramatically improves to a better value than its previous minimal value, and this gain might not be obtained if the restart strategy is not activated. Similarly, the same evolutionary behavior is observed again at the second epoch where the cost of the best solution improves until it reaches a satisfactory value.

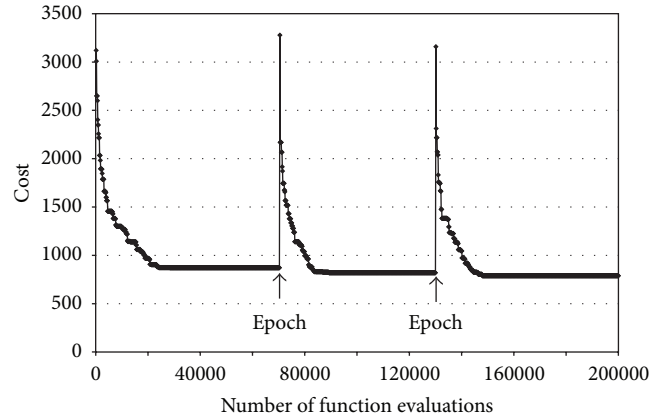

Figure 3: Epoch analysis of the restart population rebuilding.

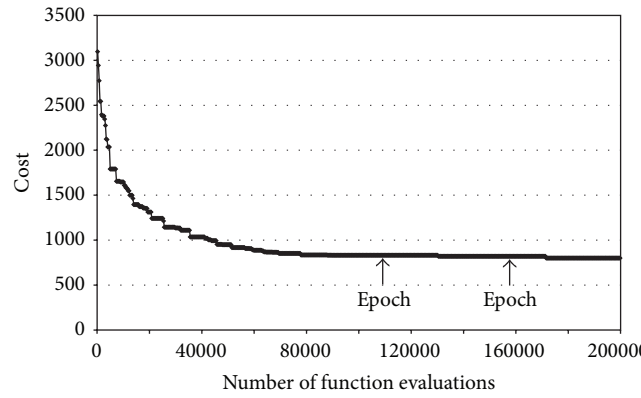

Figure 4: Epoch analysis of the partial population rebuilding.

On the other hand, the partial population rebuilding is designed to make the current search session more effective by replacing part of the current population with diverse and quality solutions produced from the reference set, a novel notion in the SS/PR template. The partial population rebuilding is activated at a less critical event than that used by the restart population rebuilding. Figure 4 shows the epoch analysis for the partial population rebuilding strategy. It is seen that the cost of the best solution in the current population drastically decreases as the number of function evaluations increases, and the decreasing rate on the cost becomes gently lower after execution of 50,000 function evaluations. After the execution of 89,900 function evaluations, the cost value cannot further improve and the learned probability model seems to suffer the premature convergence. Not until 109,900 function evaluations when the partial population rebuilding strategy detects the critical event that the best solution in the current population has not improved for a period of 20,000 function evaluations, a set of diverse and quality solutions produced by the SS/PR template

Table 2: The mean and the standard deviation (Std) for the cost of the obtained tour.

Problem	EDA		E_T^*		E_{ref}^*		E_{retart}^*		Cyber-EDA	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
Att48	50532	4271	35038	672	35637	623	34846	675	35046	622
Berlin52	9732	816	8098	202	8167	211	8115	221	8132	225
Eil51	658	20	449	8	452	8	449	9	448	7
Eil76	1173	17	605	11	609	18	607	25	601	9
Pr76	250519	4768	122428	8785	121878	3295	120713	4832	119689	2712
St70	1559	38	767	21	750	11	766	19	754	19
KroA100	75500	1327	39739	1691	37008	1786	37251	1453	35487	1244
kroB100	73802	1975	39947	1550	37401	1151	37734	1852	36923	982
kroC100	74348	1416	38927	1453	37210	1331	37266	1526	37236	1152
lin105	53509	1499	25150	1676	23509	1149	24103	2360	22939	1613
pr124	306891	6291	159876	4234	161007	2992	177010	7723	158744	3098
pr144	383242	5754	197845	5023	192221	4601	207098	10282	192230	5584
pr152	483534	11007	238234	5873	239714	5537	257858	14711	238367	5983
rat195	11538	172	6945	154	6898	90	6941	203	6877	143
kroA200	170498	1531	95000	1552	94595	1560	93986	1408	94286	1345
kroB200	166499	4315	94211	1691	93910	1388	93975	1492	93944	1371
pr226	837048	11014	436945	10364	436573	10587	437816	13274	436872	10477
tsp225	22220	320	13543	278	13091	230	13429	171	13043	245
lin318	331904	3233	228198	2588	227380	3391	227597	3222	227394	3024
rd400	122917	1105	88641	1242	88727	3975	88361	970	87380	1012

are used to replace part of the current population, and the cost of the best solution starts improving again at 118,400 function evaluations (it is noted that the cost change is too little to clearly see on the figure, but a visible change can be seen at 129,800 function evaluations). Similar behavior can be observed at the second epoch. This is because the partial population rebuilding guides the search towards a new region in the solution space, and more quality solutions which have not been seen in the previous search course are sampled and used to update the probability model.

4.3. Convergence Analysis. We further conduct a statistical test using the 95% confidence interval analysis. Thirty runs are executed for the Cyber-EDA algorithm. Figure 5 shows the evolution of the cost with the 95% confidence interval for the best solution in the current population. The range of the confidence interval is within sufficiently narrow bounds and is very stable throughout the entire evolution cycle. We can note that the quality of the final solution obtained by the Cyber-EDA algorithm is guaranteed with a high confidence level.

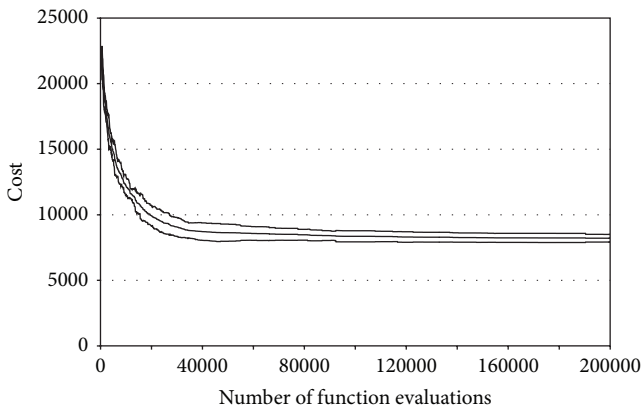

Figure 5: Convergence analysis with 95% confidence interval.

5. Concluding Remarks

The research for investigating the variable (in)dependence relationships through the evolutionary successions is critical for preserving important schema against genetic operations. The Estimation of Distribution Algorithms (EDAs) create a promising domain that learns the probability model of variable (in)dependence. Our Cyber-EDA algorithm draws on strategies from the adaptive memory programming (AMP) for improving the performance of each EDA component. Notable strategies, namely, the diversification generation, improvement method, thresholding, incremental learning, weighting sampling scheme, probability model rebuilding, and population rebuilding were exploited in this paper. The experimental result on benchmark TSP instances supports our anticipation that the AMP strategies can enhance the performance of classic EDA by deriving a better approximation for the true distribution of the target solutions. This study reveals a promising research direction to develop a more effective EDA by combining various AMP strategies that adaptively respond to different scenarios (such as the problem size and the landscape profile) of the optimization problems.

Acknowledgment

This research is partially supported by National Science Council of ROC, under Grant NSC 98-2410-H-260-018-MY3.

References

[1] J. H. Holland, *Adaptation in Natural and Artificial Systems*, The University of Michigan Press, 1975.

[2] S. Baluja, "Population based incremental learning: a method for integrating genetic search based function optimization and competitive learning," Tech. Rep. CMU-CS-94-163, Carnegie Mellon University, 1994.

[3] H. Mühlenbein and G. Paaß, "From recombination of genes to the estimation of distributions I. Binary parameters," in *Proceedings of the 4th Parallel Problem Solving from Nature (PPSN '96)*, vol. 4, pp. 178–187, 1996.

[4] F. Glover, "Tabu search and adaptive memory programming—advances, applications and challenges," in *Interfaces in Computer Science and Operations Research*, H. Barr and J. L. Kennington, Eds., pp. 1–75, Kluwer Academic Publishers, 1996.

[5] M. Laguna and R. Marti, *Scatter Search: Methodology and Implementation in C*, Kluwer Academic Publishers, London, UK, 2003.

[6] F. Glover, "A template for scatter search and path relinking," in *Artificial Evolution*, vol. 1363 of *Lecture Notes in Computer Science*, pp. 13–54, 1998.

[7] F. Glover, "Ejection chains, reference structures and alternating path methods for traveling salesman problems," *Discrete Applied Mathematics*, vol. 65, no. 1–3, pp. 223–253, 1996.

[8] T. A. Feo and M. G. C. Resende, "Greedy randomized adaptive search procedures," *Journal of Global Optimization*, vol. 6, no. 2, pp. 109–133, 1995.

[9] P.-Y. Yin, F. Glover, M. Laguna, and J.-X. Zhu, "Cyber swarm algorithms—improving particle swarm optimization using adaptive memory strategies," *European Journal of Operational Research*, vol. 201, no. 2, pp. 377–389, 2010.

[10] S. Nakano, A. Ishigame, and K. Yasuda, "Particle swarm optimization based on the concept of tabu search," in *Proceedings of IEEE Congress on Evolutionary Computation (CEC '07)*, pp. 3258–3263, September 2007.

[11] T. Taillard and L. Gambardella, "Adaptive memories for the quadratic assignment problems," Tech. Rep., Istituto Dalle Molle Di Studi Sull Intelligenza Artificiale (IDSIA), 1997.

[12] P. Y. Yin, "Towards more effective metaheuristic computing," in *Modeling, Analysis, and Applications in Metaheuristic Computing: Advancements and Trends*, P. Y. Yin, Ed., IGI-Global Publishing, February 2012.

[13] P.-Y. Yin and E.-P. Su, "Cyber Swarm optimization for general keyboard arrangement problem," *International Journal of Industrial Ergonomics*, vol. 41, no. 1, pp. 43–52, 2011.

[14] P.-Y. Yin, F. Glover, M. Laguna, and J.-X. Zhu, "Cyber swarm algorithms—improving particle swarm optimization using adaptive memory strategies," *European Journal of Operational Research*, vol. 201, no. 2, pp. 377–389, 2010.

[15] P. Y. Yin and Y. T. Chiang, "Cyber swarm algorithms for multiobjective nurse rostering problem," *International Journal of Innovative Computing, Information and Control*, vol. 9, no. 5, pp. 2043–2063, 2013.

[16] J. Grahl, S. Minner, and P. A. N. Bosman, "Learning structure illuminates black boxes: an introduction into estimation of distribution algorithms," in *Advances in Metaheuristics for Hard Optimization*, Z. Michalewicz and P. Siarry, Eds., pp. 365–396, Springer, Berlin, Germany, 2008.

[17] G. R. Harik, F. G. Lobo, and D. E. Goldberg, "Compact genetic algorithm," in *Proceedings of IEEE International Conference on Evolutionary Computation (ICEC '98)*, pp. 523–528, May 1998.

[18] S. J. De Bonet, C. L. Isbell, and P. Viola, "MIMIC: finding optima by estimating probability densities," in *Advances in Neural Information Processing Systems*, M. C. Mozer, M. I. Jordan, and T. Petsche, Eds., vol. 9, p. 424, The MIT Press, 1997.

[19] S. Baluja and S. Davies, "Using optimal dependency-trees for combinatorial optimization: learning the structure of the search space," in *Proceedings of the International Conference on Machine Learning*, D. H. Fisher, Ed., pp. 30–38, 1997.

[20] M. Pelikan and H. Mühlenbein, "The bivariate marginal distribution algorithm," in *Advances in Soft Computing—Engineering Design and Manufacturing*, R. Roy, T. Furuhashi, and P. K. Chawdhry, Eds., pp. 521–535, 1999.

[21] G. Harik, "Linkage learning via probabilistic modeling in the ECGA," Tech. Rep. 99010, IlliGAL, University of Illinois, Urbana, Ill, USA, 1999.

[22] M. Pelikan, D. E. Goldberg, and E. Cantú-Paz, *BOA: The Bayesian Optimization Algorithm*, 1999.

[23] M. Pelikan, K. Sastry, M. V. Butz, and D. E. Goldberg, "Hierarchical BOA on random decomposable problems," IlliGAL Report 2006002, Illinois Genetic Algorithms Laboratory, University of Illinois, Urbana-Champaign, Ill, USA, 2006.

[24] R. Exteberria and P. Larrañaga, "Global optimization using bayesian networks," in *Proceedings of the 2nd Symposium on Artificial Intelligence (CIMAF '1999)*, pp. 332–339, 1999.

[25] H. Mühlenbein and T. Mahnig, "FDA—a scalable evolutionary algorithm for the optimization of additively decomposed functions," *Evolutionary Computation*, vol. 7, no. 4, pp. 353–376, 1999.

[26] M. Sebag and A. Ducoulombier, "Extending population-based incremental learning to continuous search spaces," in *Proceedings of the 5th International Conference on Parallel Problem Solving from Nature (PPSN '98)*, vol. 5, pp. 418–427, 1998.

[27] P. Larranaga, R. Etxeberria, J. A. Lozano, and J. M. Pena, "Optimization in continuous domains by learning and simulation of Gaussian networks," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '00)*, pp. 201–204, 2000.

[28] P. Larrañaga, R. Etxeberria, J. A. Lozano, and J. M. Pena, "Optimization by learning and simulation of Bayesian and Gaussian networks," Tech. Rep. EHU-kZAA-IK-4/99, University of the Basque Country, 1999.

[29] P. A. N. Bosman and D. Thierens, "Expanding from discrete to continuous estimation of distribution algorithms: the IDEA," in *Proceedings of the 6th International Conference on Parallel Problem Solving from Nature (PPSN '00)*, vol. 6, pp. 767–776, 2000.

[30] H. Handa, "Estimation of distribution algorithms with niche separation mechanism," in *Proceedings of IEEE Congress on Evolutionary Computation (CEC '07)*, pp. 119–126, September 2007.

[31] C. W. Ahn and H.-T. Kim, "Estimation of particle swarm distribution algorithms: bringing together the strengths of PSO and EDAs," in *Proceedings of the 11th Annual Genetic and Evolutionary Computation Conference (GECCO '09)*, pp. 1817–1818, July 2009.

[32] Q. Zhang, J. Sun, E. Tsang, and J. Ford, "Hybrid estimation of distribution algorithm for global optimization," *Engineering Computations*, vol. 21, no. 1, pp. 91–107, 2004.

[33] F. Glover, "Future paths for integer programming and links to artificial intelligence," *Computers and Operations Research*, vol. 13, no. 5, pp. 533–549, 1986.

[34] S. Kessentini, D. Barchiesi, T. Grosges, L. Giraud-Moreau, and M. L. de la Chapelle, "Adaptive non-uniform particle swarm application to plasmonic design," *International Journal of Applied Metaheuristic Computing*, vol. 2, no. 1, pp. 18–28, 2011.

[35] G. Maquera, M. Laguna, D. A. Gandelman, and A. P. Sant'Anna, "Scatter search applied to the vehicle routing problem with simultaneous delivery and pickup," *International Journal of Applied Metaheuristic Computing*, vol. 2, no. 2, pp. 1–20, 2011.

[36] M. Hassannezhad and N. Javadian, "Utilizing the modified self-adaptive differential evolution algorithm in dynamic cellular manufacturing system," *International Journal of Applied Metaheuristic Computing*, vol. 3, no. 2, pp. 1–17, 2012.

[37] P. Joshi and P. Kulkarni, "Incremental learning: areas and methods—a survey," *International Journal of Data Mining & Knowledge Management Process*, vol. 2, no. 5, pp. 43–51, 2012.