Review Article

Jun Zhang, Zhi-hui Zhan, Ying Lin, Ni Chen, Yue-jiao Gong, and Jing-hui Zhong, *Sun Yat-Sen University, CHINA*
Henry S.H. Chung, *City University of Hong Kong, CHINA*
Yun Li, *University of Glasgow, UK*
Yu-hui Shi, *Xi'an Jiaotong-Liverpool University, CHINA*

Evolutionary Computation Meets Machine Learning: A Survey

I. Introduction

Evolutionary computation (EC) [1] is a kind of optimization methodology inspired by the mechanisms of biological evolution and behaviors of living organisms. In the literature, the terminology evolutionary algorithms (EA) [2] is frequently treated the same as EC. Generally speaking, EC algorithms include genetic algorithm (GA), evolutionary programming (EP), evolutionary strategies (ES), genetic programming (GP), learning classifier systems (LCS), differential evolution (DE), and estimation of distribution algorithm (EDA). Recently, swarm intelligence (SI) [3] algorithms like ant colony optimization (ACO) and particle swarm optimization (PSO) have also been proposed as optimization methodologies and have gained increasing popularity in the EC research community. SI algorithms share many common characteristics with EAs and are also regarded to be in the EC algorithm family [4]. In this article, we regard EC algorithms to include both EAs and SI algorithms.

Different EC algorithms have similar framework in implementation and algorithmic characteristics. Fig. 1 describes a general framework with three fundamental operations and two optional operations for most ECs. In an EC algorithm, the first step is the '*initialization*' step. Then the algorithm enters evolutionary iterations with two operational steps, namely, '*fitness evaluation and selection*' and '*population reproduction and variation*'. The new population is then evaluated again and the iteration continues until a termination criterion is satisfied. Besides the above three necessary steps, EC algorithms sometimes additionally perform an '*algorithm adaptation*' procedure or a '*local search*' (LS) procedure. The EC algorithms with LS are known as memetic algorithms (MA) [5].

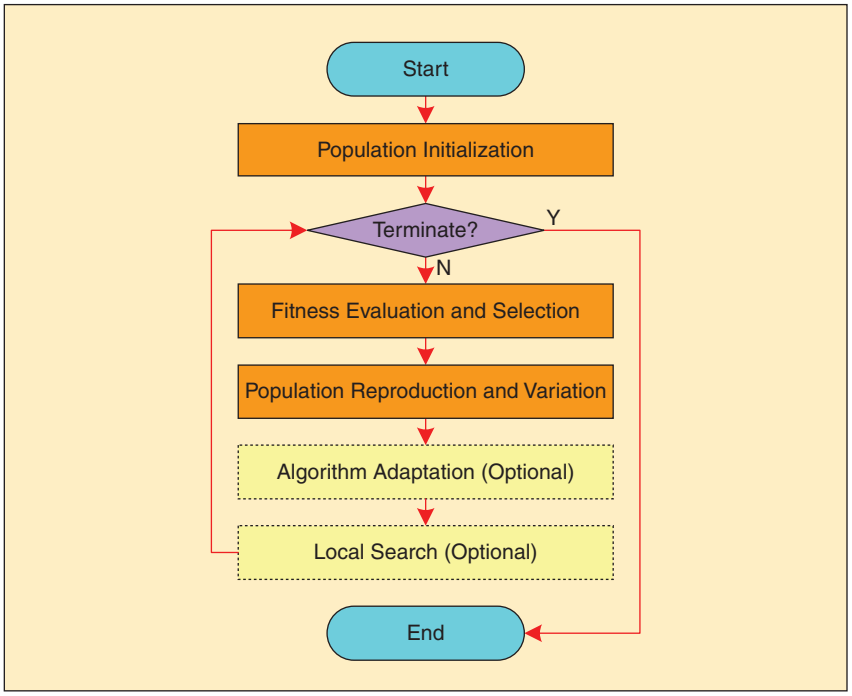

FIGURE 1 The general EC framework.

The developments and applications of EC algorithms have been one of the fastest growing fields in computing science. Moreover, research into enhancing the EC algorithms via machine learning (ML) techniques is witnessed to have played an important role in the literature [6][7]. ML is one of the most promising and salient research areas in artificial intelligence, which has experienced a rapid development and has become a powerful tool in a wide range of applications [8]. In studies relevant to both EC algorithms and ML techniques, many attempts have been made to apply variants EC algorithms as types of effective and efficient ML techniques [9].

In contrast with the view of using EC algorithms as ML techniques, this

Digital Object Identifier 10.1109/MCI.2011.942584
Date of publication: 20 October 2011

article focuses on making a survey of researches based on using ML techniques to enhance EC algorithms. In the framework of an ML-technique-enhanced-EC algorithm (MLEC), the main idea is that the EC algorithm has stored ample data about the search space, problem features, and population information during the iterative search process, thus the ML technique is helpful in analyzing these data for enhancing the search performance. In this way, useful information can be extracted to understand the search behavior and to assist with future searches for the global optimum. In many applications, EC algorithms incorporating ML techniques have been proven to be advantageous in both convergence speed and solution quality.

In the literature, variants ML techniques have been used in EC algorithms to enhance the algorithm performance. These ML techniques can include: statistical methods (e.g., mean and variance), interpolation and regression, clustering analysis (CA), principle component analysis (PCA), orthogonal experimental design (OED), opposition-based learning (OBL), artificial neural networks (ANN), support vector machines (SVM), case-based reasoning, reinforcement learning, competitive learning, and Bayesian network.

These ML techniques can be incorporated into different EC algorithms in various ways and they affect EC also on various aspects. Though there are works like Jourdan *et al.*'s survey [10] to review the approaches of using data mining techniques to enhance metaheuristics, the work of MLEC algorithms is still scattered in the literature and need a systematic consolidation. Therefore, it is useful to conduct a comprehensive and systematic survey on this topic. In this article, we attempt to conduct such a survey, not only to provide a taxonomy to classify existing research spectrum, but also discuss potential future research directions in this research arena. The taxonomy can be divided into two levels. The higher level is based on the five evolutionary steps of an EC algorithm, namely, population initialization, fitness evaluation and selection, population reproduction and variation, algorithm adaptation, and local search. The lower level is according to the functionality of using the ML techniques in each evolutionary step of an EC algorithm. Such a taxonomy provides a picture that shows how an ML technique can be used to enhance an EC algorithm. In order to present the literature review, we collected and selected the representative references which are related to this research field, by considering their quality, popularity, number of citations, and the cover of different aspects of the topic surveyed in this article.

…this article focuses on making a survey of researches based on using ML techniques to enhance EC algorithms.

The rest of this article is organized as follows. Section II surveys existing researches with ML techniques incorporated into ECs for the algorithm performance enhancements. In Section III, future research directions are discussed and proposed, followed by conclusions in Section IV.

II. ML Techniques Used in EC Algorithms

This section reviews the EC algorithms that are enhanced by ML techniques. The survey is organized from the EC perspective, including population initialization, fitness evaluation and selection, population reproduction and variation, algorithm adaptation, and local search. An illustration of the taxonomy in this survey is shown in Fig. 2.

A. ML for Population Initialization

1. Organizing Initial Solutions Position

The OED technique is a kind of statistical method that can obtain knowledge from data. Therefore it is regarded as an ML technique here. The OED is developed for designing experiments with multiple *factors* and multiple choices (namely *levels*) per factor. Instead of testing all the combinations of factor levels, OED only tests orthogonal combinations that are constructed by following a statistic-based orthogonal array. In this way, OED is able to explore the design space comprehensively with a much smaller computational effort. Leung and Wang [11] used OED technique in GA to generate the initial population. By doing so, the algorithm can start exploring the search space evenly through the initialization step.

2. Improving Initial Solutions Quality

As opposed to OED technique which improves initialization by organizing the positions of initial solutions, ML techniques such as OBL, interpolation, and ANN act in a more straightforward way by generating initial solutions of higher quality. The EC algorithms are thus more likely to start from points closer to the global optimum and can converge in fewer generations.

OBL is a new machine intelligence algorithm inspired by the phenomena of sudden and radial changes in social revolution. Given a problem, OBL attempts to approximate the optimal solution from two opposite sides. As used in [12], the initialization step of a DE algorithm employs OBL to generate an opposite population to the current population. Then the algorithm merges the two populations and selects only half of the best solutions to compose the initial population.

Interpolation is also implemented on the basis of a set of random solutions when applied to initialize the population. It seeks better solutions by interpolating the selected random solutions. The best subset of the random and interpolated solutions then constitutes the initial population [13]. The application of ANN to population initialization can also be found in the literature [14].

In the framework of an ML-technique-enhanced-EC algorithm (MLEC), the main idea is that the EC algorithm has stored ample data about the search space, problem features, and population information during the iterative search process, thus the ML technique is helpful in analyzing these data for enhancing the search performance.

3. Incorporating Historical Search Experience

ML techniques can also help with population initialization by acquiring useful information from historical search experiences. An example is to use a case-based initialization approach [15]. This approach is derived from the idea of case-based reasoning in ML and maintains a case base of solutions of solved problems. When dealing with a new problem, an EC algorithm with case-based initialization retrieves solutions of similar problems from the case base and injects them into the initial population. In the context of dynamic problems, statistical methods are often used to study historical experiences to help reinitialize the population. Woldesenbet and Yen [16] presented a variable relocation approach to reinitialize the population after a change occurred. Kim *et al.* [17] also designed an evolutionary multi-objective optimization algorithm by using the information from previously solved problem instances to make the search start from a more efficient point.

B. ML for Fitness Evaluation and Selection

1. Modeling Objective Function

In many practical problems, the objective function cannot be expressed in an analytical form of decision variables or the analytical formula can be too costly to evaluate. In this case, an approximate model of the real objective function is needed for evaluating the population. Among various modeling methods, the ANN technique (for single objective function [18] and for multi-objective function [19]), statistical methods [20], polynomial regression [21], radial basis function interpolation [22], and Markov fitness model [23], to name just a few, have been incorporated into the evaluation step to model the objective function [24].

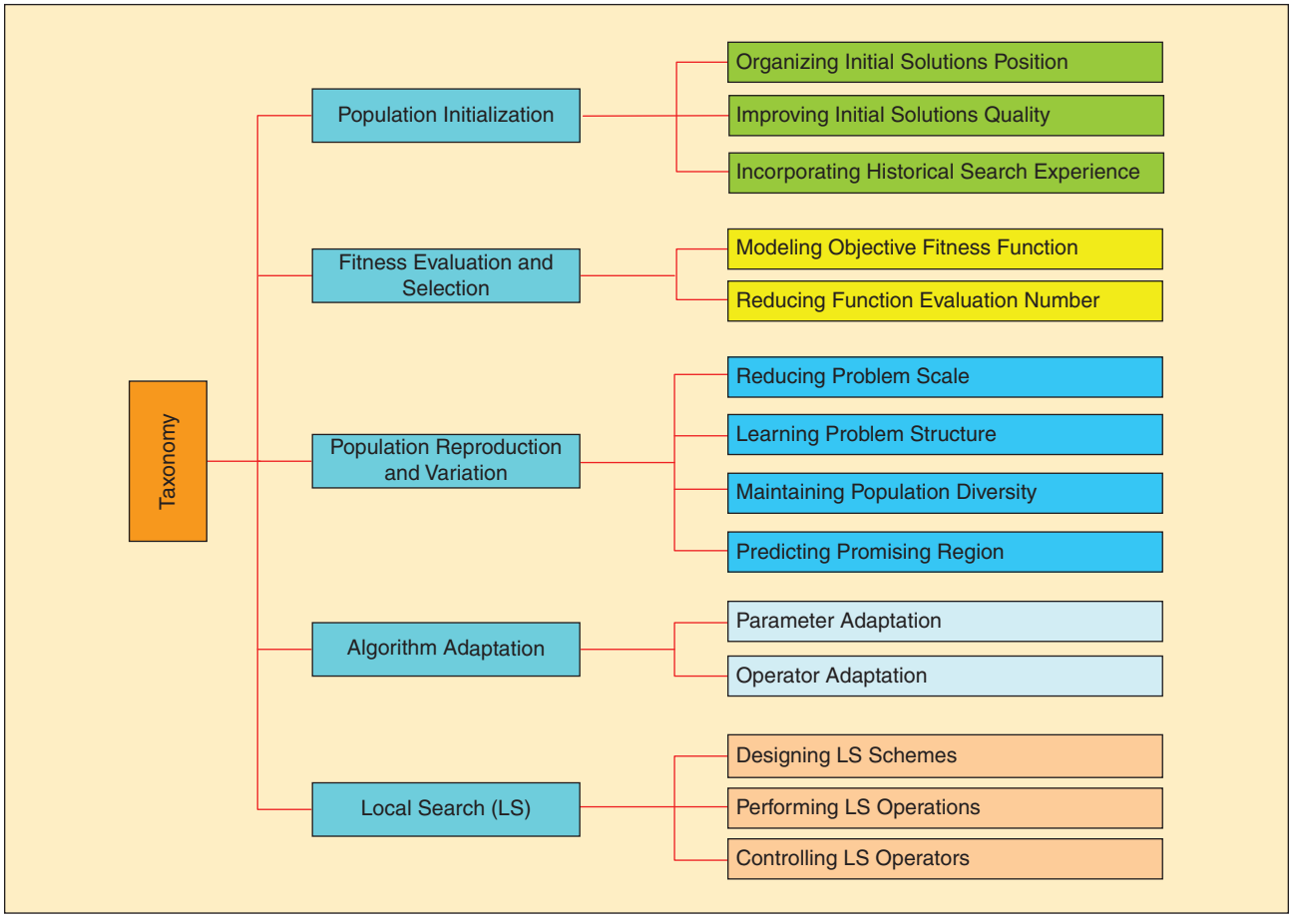

FIGURE 2 An illustration of the taxonomy.

2. Reducing Number of Function Evaluations

Some researches use ML techniques to reduce the number of function evaluations when the objective functions are very expensive for evaluation. For example, Zhou and Zhang [25] model the objective function of the Minimax problem via a Guassian process and evaluate the individuals based on the surrogate model, where only the best individual is evaluated by the actual objective function. Another typical example is to use the CA technique [26]. The CA technique first clusters the population and then selects a representative individual from each cluster. Only the representative individuals are evaluated by the objective function. The objective values of the other individuals are approximated in a simplified way. By doing so, the number of costly function evaluations can be reduced and the evaluation time is therefore shortened. Jin and Sendhoff reduced the number of function evaluations by using both the CA and ANN techniques [27].

C. ML for Population Reproduction and Variation

1. Reducing Problem Scale

When performing the population reproduction and variation operations, the problem scale has significant influences on the algorithm performance. ML techniques have drawn significant interests in reducing the problem scale to generate new individuals effectively and efficiently.

In [28], Senjyu *et al*. adopted a CA technique to cluster units with analogous characteristics to support a GA for the unit commitment optimization problem. Using CA techniques to reduce the problem scale is also appealing when solving large-scale traveling salesman problems [29]. The main idea is to cluster the cities into different groups according to their position distribution information. EC algorithms find the optimal path in each group first before they solve the whole problem by regarding each group as a city in a new problem. As the algorithm only handles a small number of cities each time, quality solutions are more likely to be generated [29].

In order to present the literature review, we collected and selected the representative references which are related to this research field, by considering their quality, popularity, number of citations, and the cover of different aspects of the topic surveyed in this article.

The ANN technique is also used to reduce the problem scale and the search space. Lee *et al*. [30] used an ANN to restrict the continuous search space so as to reduce the difficulty of applying GAs to solve large-scale instances of a reliability assignment or redundant allocation problem. Marim *et al*. [31] used an ANN to restrict the search space for the GA when optimizing the ground-state geometry of silicon clusters.

The PCA techniques have also been used in the EC algorithms to reduce the search space. In the ES algorithm with covariance matrix adaptation [32], the solution space is transformed into a new space via an adaptive covariance matrix, and the dimension of the new space is reduced by using the PCA technique. New population is generated in the reduced space, and the new population is reverse-transformed to the original solution space for evaluation.

2. Learning Problem Structure

The reproduction operators of an EC algorithm, e.g., crossover and mutation, may fail to generate high quality population because they are problem independent and can hence break up converging building blocks or fail to combine them effectively [33]. In order to address this issue, some researchers have proposed to learn the problem structure by using ML techniques and then to use the obtained information to guide the generation of new population. One representative example is the EDA, which uses a probability modeling technique to generate a new population. To construct probability models accurately and efficiently, various ML techniques are used. In the literature, statistical methods [33], competitive learning techniques [34], Bayesian network [35], and linkage learning techniques [36] are reportedly being used to improve probability models.

For multimodal optimization problems, ML techniques have been used to construct mixture probability models to avoid misleading the search to local optima. In [37], Lu and Yao reported the use of 'rival penalized competitive learning' clustering technique to classify a selected population without prior knowledge. After clustering the selected population, Gaussian distribution associated with each group is then estimated. By selecting a different Gaussian distribution according to the average fitness of the corresponding group, a new population with higher diversity can be generated. Similar work that uses clustering techniques to build mixture probability models can be found in [38].

Recently, some researchers find that Optinformatics is helpful for extracting knowledge by using the optimization data during the search process. Specially, Le *et al*. [39] and Le and Ong [40] presented a frequent schemas analysis technique for binary GA to mine for interesting frequent schemas that exist implicitly within the GA data. This Optinformatics technique is efficient for the algorithm to learn the problem structure and is helpful for the algorithm to generate a better population.

3. Maintaining Population Diversity

Maintaining the population diversity is significant for improving the performance of EC algorithms, especially in their applications to multimodal and multi-objective problems. ML techniques such as CA and OBL have been

utilized in various ways to achieve this goal.

Using CA to divide the population into sub-populations has been a popular method for maintaining the population diversity. However, different researchers may have different strategies in using the obtained sub-populations. Some researchers regard each sub-population as a niche, and emphasize the independence of each sub-population. In [41], the selection operator of a GA is applied to different sub-populations separately. The population diversity is maintained because only individuals with similar features (i.e., in the same cluster) compete for survival. Conversely, some researchers consider different sub-populations to contain different information, and emphasize the interaction among sub-populations for generating a better new population [42]. On another hand, some researchers argue that different sub-populations should run independently but communicate using designated strategies [43]. Using CA techniques to maintain the population diversity is also appealing when designing multi-objective evolutionary computation algorithms, e.g., multi-objective PSO [44] and multi-objective DE [45].

Besides the CA technique, the OBL technique is also adopted in EC operations to help increase population diversity. In [12], Rahnamayan *et al.* first used OBL in DE to design a generation jumping operator for creating a new population every generation. This OBL based population generation method, i.e., generating a new individual opposite to the current individual, is helpful to increase the population diversity because both the original population and the generated OBL-based population are considered in the next generation.

4. Predicting Promising Region

Variants of ML techniques such as statistics and analysis methods have been used to predict promising regions in the search space and to guide the generation of new populations. Liang and Suganthan [46] proposed using a history learning strategy to create statistics on the historical data and to predict the promising moving direction of a particle when designing PSO algorithm. Zhang *et al.* [47] used a statistical method to design intelligent crossover operator for GP so as to predict a promising search region. Li and Tam [48] used the CA technique to cluster the individuals into different groups during the search process and preserved the best individual of each group to predict a promising search direction and to help generate a better population. CA techniques are also used in PSO to predict a promising leader for guiding the flying behavior of particles [49] and used in DE to generate promising new population according to the cluster centers [50].

Some researchers extended the OED technique to predicting promising regions, by using factor analysis (FA) procedure in the OED. After generating orthogonal combinations, the OED utilizes FA to analyze the contribution of different levels in each factor. The combination of the favored levels of different factors is likely to be a promising one. The OED with FA was first introduced into EAs by Ho *et al.* [51] for generating a new population. Zhan *et al.* [52] also used the OED in PSO to enhance the algorithm performance and developed orthogonal learning PSO (OLPSO). OLPSO performs OED with FA on a particle's personal historical best position and its neighborhood's best position to construct an efficient exemplar, which indicates a promising search direction. This constructed exemplar is used in the velocity update equation to guide the particle to fly steadily towards the global optimum.

The SVM technique is also used to predict promising regions to generate new populations. In [53], Yan *et al.* improved the global search ability of the DE algorithm based on SVM regression function approximation. First, several best individuals are selected, and then the SVM is performed to approximate the search region. All the best points of the approximation function are generated and are included in the population. These new generated individuals can indicate the promising search region and can help the algorithm generate a better new population.

D. ML for Algorithm Adaptation

1. Parameter Adaptation

Among various adaptive approaches used in EC algorithms, the most popularly adopted approach is the use of statistical methods to analyze search process data for a better understanding of the search process and hence adaptively control the algorithm's parameters. For example, Zhan *et al.* [54] used statistical method to analyze population distribution information and fitness information of PSO. An evolutionary factor is defined to classify the evolutionary state to be an exploration, an exploitation, a convergence, or a jumping-out state in a fuzzy manner. Then the PSO parameters are adaptively controlled according to the identified evolutionary state, leading to the development of 'adaptive PSO (APSO)'. The idea of using statistical methods for parameter adaptation are also reported in the self-adaptive DE proposed by Qin *et al.* [55] and in the adaptive DE proposed by Zhang and Sanderson [56]. Moreover, Rosca and Ballard [57] proposed to plug statistical methods into GP to adaptively control the algorithm representation.

Besides these simple ML techniques, some advanced techniques like CA are also used. Zhang *et al.* [58] proposed the use of CA to divide the chromosomes of GA into several groups based on their position information. The evolutionary states are determined by considering the relationship between the size (number of chromosomes) and the fitness information of different groups. The crossover and mutation probabilities are adaptively controlled according to the evolutionary states. Such a CA based parameter adaptation strategy is also used in Zhan *et al.*'s work [59] for PSO and Gong *et al.*'s work [60] for ACO, respectively.

2. Operator Adaptation

The adaptive control of operators refers to the selection of one or multiple

operators from a candidate set of operators adaptively, according to the information of the optimization process. It works in a similar way to parameter adaptation except that the adapted object is the algorithm's operators instead of parameters.

The operator adaptation mechanisms frequently embed the idea of reinforcement learning. They utilize historical information about the performance of candidate operators to select operators. A good example is given by Zhang and Lu [61] where the mutation operator of EP is adapted by accumulated fitness rewards obtained from reinforcement learning. The idea of reinforcement learning can also be found in [62] for the adaptation of mutation and crossover operators in GA. Similarly, in Qin *et al.*' study [55], four mutation schemes for DE are used adaptively during the evolutionary process. The probabilities of selecting different mutation schemes are adaptively adjusted by the statistics on the historical successful experiences.

E. ML for Local Search

1. Designing Local Search Schemes

The LS aims to refine a solution within the solution's neighborhood. Several researchers have used ML to design suitable LS operators. In [63], Zhang *et al.* proposed a new local search scheme by using the OED technique to extend the ACO to continuous search space. In [64], Lin *et al.* proposed a contour based LS method. By analyzing the solutions in the population, the contour of the fitness landscape is interpolated and the centroid of the contour is calculated to approach the optimal position.

2. Performing Local Search Operations

Determining where and when to perform the LS operation is important because they affect not only the computational burden, but also the search efficiency. It may be worthwhile to apply LS on every individual if the computational complexity of the LS is relatively low. However, if the LS is costly, selecting appropriate subset of individuals to apply the LS would become more efficient. The simplest way would be to select good individuals to perform the LS operation. Martinez-Estudillo *et al.* [65] proposed a clustering based method to select a subset of individuals to perform the LS. In this method, a current population is clustered into several groups and only the best individual of each group is refined by the LS operator. Lim *et al.* [66] proposed introducing a classifier into the constrained MA optimization. The classifier-assisted constrained MA only performs LS on the misclassified individuals that are more likely to locate near the global optimum. Handoko *et al.* [67] adopted the Optinformatics concept to design a feasibility structure modeling technique in order to assess whether to perform the LS operation on an individual.

3. Controlling Local Search Operators

In order to control the LS operators, e.g., adaptively control the search intensity and search depth of a single LS operator, or adaptively control the utilization of multiple LS operators, various ML techniques have been used to obtain search information during the evolutionary process.

In [68], Nguyen *et al.* used statistical methods to derive a theoretical bound for individual learning intensity in their proposed probabilistic memetic framework. In addition, ML techniques are also used to help control the utilization of multiple LS operators. In [69], statistical methods are applied to measure the fitness diversity of the population. The statistical values are then used to control two designed LS operators, including the dynamic parameter settings and the adaptive launch of two local searchers according to the need of the evolution. Noman *et al.* [70] used Lamarckian learning strategies to adaptively determine the length of the search by taking feedback from the search process.

…we found that the researches in this field are still not systematic and many issues still remain unexplored. … we will discuss several potential directions for this research area.

III. Potential Research Directions

The idea of using ML techniques to enhance the performance of EC algorithms has been successfully implemented by many researchers. While a lot of work has been done and the studies have been surveyed in the last section, we found that the researches in this field are still not systematic and many issues still remain unexplored. In this section, we will discuss several potential directions for this research area.

A. Broader Use of ML Techniques

In this survey, it is noticed that the most frequently used ML techniques in EC are statistical methods, CA, OED, and ANN. These are likely because these ML techniques are the most well known and are easily applied to EC algorithms. However, different ML techniques may be suitable to different application environments and offer different advantages. Therefore, it may be interesting to try to use a broader variant of ML techniques in the EC algorithms, and investigate how to use different ML techniques to enhance different EC algorithms. This may bring new ideas to the design of novel MLEC algorithms.

B. Using ML Techniques As EC Operators

In the literature reviewed, most of the studies use ML techniques as data analysis tools to extract useful information from the EC search data during the run in order to help the EC algorithms. However, it could also be interesting to investigate the utilization of ML techniques as EC operators to enhance the algorithm performance. Even

The survey consists of five categories: ML for population initialization, ML for fitness evaluation and selection, ML for population reproduction and variation, ML for algorithm adaptation, and ML for local search.

though some previous studies have been conducted to investigate some kinds of ML techniques (such as the OED method as an LS operator) to be an EC operator and hybrid it into the EC algorithms, there are still many issues for research. For example, using different ML techniques, hybriding with different EC algorithms, and designing as different operators, are critical issues that can be considered.

C. Better Cooperation Between ML and EC

While ML techniques can help the EC algorithms search more effectively and efficiently, they also add to computational burden. There is a tradeoff between the benefits and the computational costs. Better understanding and improving cooperation between ML and EC will play a significant role in enhancing EC algorithms with ML techniques efficiently. This thus could be a very promising research direction which includes, e.g., use of various ML techniques (simple or complex), balancing the performance improvement and computational burden caused by the ML techniques (e.g., how often to execute the ML methods), and so on.

D. Systematical Control of MLEC Algorithms

Another problem caused by the ML techniques is that more parameters would have been introduced into the algorithms. This makes the MLEC algorithms become much more complex. Therefore, determining how to systematically control the MLEC algorithms and how to automatically determine their parameters during the search have become significant research topics to be undertaken in the MLEC field.

E. Deeper Theoretical Analysis

As the MLEC algorithms become more complex by incorporating with the ML techniques, it will be more difficult to make clear why and how the MLEC algorithms converge. Besides the numerical experiments based on benchmark functions, it may be appealing and significant to make deeper theoretical analysis on the new MLEC algorithms in order to clarify their search behaviors and convergence mechanisms. Therefore, it could be a considerable research direction to take in allowing for deeper theoretical analysis on the MLEC algorithms' convergence and stability.

F. Further Test on Complex Problems

The good performance of MLEC algorithms are mainly studied on numerical benchmark functions and are mainly in the field of single objective optimization problems. However, the applications of ML enhanced EC algorithms in multi-objective, dynamic and uncertain, large scale, and constrained problems are still insufficient. Therefore, exploring how to design effective and efficient MLEC algorithms for these complex problems is a potential research direction in the near future.

G. Wider Real-World Applications

The good results of MLEC algorithms on numerical benchmark functions also encourage the research of applying the MLEC algorithms to numerous real-world problems. Owing to the improved search speed and accuracy, these algorithms are appealing for a wider range of complex real-world applications. Moreover, new ideas of using proper ML techniques and improving current ML techniques when applying to EC algorithms may be inspired in the application process. We believe that applying MLEC algorithms to wider areas of practical problems could be a useful trend to follow.

IV. Conclusion

In this article, we have provided a comprehensive survey on the application of ML techniques to enhance EC algorithms. Taxonomy of research in this area has been defined according to the evolutionary steps of the EC framework and the functionalities of the ML techniques. The survey consists of five categories: ML for population initialization, ML for fitness evaluation and selection, ML for population reproduction and variation, ML for algorithm adaptation, and ML for local search. In each category, the utilization of ML techniques is further classified according to their motivations and functionalities. Such a structural taxonomy provides a better understanding of usages and effects of ML techniques in an EC algorithm. It also reveals why and how ML techniques can improve the algorithm's performance in various aspects. Based on the survey of the existing approaches in the literature, some potential future research directions in this area have also been discussed and proposed.

V. Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (NSFC) No.61070004, by NSFC Joint Fund with Guangdong under Key Project U0835002, and by the National High-Technology Research and Development Program ("863" Program) of China No. 2009AA01Z208.

References

[1] D. B. Fogel, *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence*. Piscataway, NJ: IEEE Press, 1995.

[2] T. Back, M. Emmerich, and O. M. Shir, "Evolutionary algorithms for real world applications," *IEEE Comput. Intell. Mag.*, vol. 3, no. 1, pp. 64–67, Jan. 2008.

[3] J. Kennedy, R. Eberhart, and Y. Shi, *Swarm Intelligence*, 1st ed. San Mateo, CA: Morgan Kaufmann, 2001.

[4] Y. H. Shi and R. C. Eberhart, "Comparison between genetic algorithms and particle swarm optimization," in *Proc. 7th Int. Conf. Evolutionary Programming*, Mar. 1998, pp. 611–616.

[5] Y. S. Ong, M. H. Lim, and X. Chen, "Memetic computation—Past, present and future," *IEEE Comput. Intell. Mag.*, vol. 5, no. 2, pp. 24–31, May 2010.

[6] J. Zurada, M. Mazurowski, R. Ragade, A. Abdullin, J. Wojtudiak, and J. Gentle, "Building virtual community in computational intelligence and machine learning," *IEEE Comput. Intell. Mag.*, vol. 4, no. 1, pp. 43–54, Jan. 2009.

[7] G. G. Yen, "Learning and intelligence," *IEEE Comput. Intell. Mag.*, vol. 5, no. 2, p. 2, May 2009.
[8] E. Mjolsness and D. DeCoste, "Machine learning for science: State of the art and future prospects," *Science*, vol. 293, no. 5537, pp. 2051–2055, 2001.
[9] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading, MA: Addison-Wesley, 1989.
[10] L. Jourdan, C. Dhaenens, and E. G. Talbi, "Using datamining techniques to help metaheuristics: A short survey," *Hybrid Metaheuristics*, pp. 57–69, 2006.
[11] Y. W. Leung and Y. Wang, "An orthogonal genetic algorithm with quantization for global numerical optimization," *IEEE Trans. Evol. Comput.*, vol. 5, no. 1, pp. 41–53, 2001.
[12] S. Rahnamayan, H. R. Tizhoosh, and M. M. A. Salama, "Opposition-based differential evolution," *IEEE Trans. Evol. Comput.*, vol. 12, no. 1, pp. 64–79, 2008.
[13] M. Pant, M. Ali, and V. P. Singh, "Differential evolution using quadratic interpolation for initializing the population," in *Proc. IEEE Int. Conf. Advance Computing*, 2009, pp. 375–380.
[14] T. Yalcinoz and H. Altun, "Power economic dispatch using a hybrid genetic algorithm," *IEEE Power Eng. Rev.*, 2001, pp. 59–60.
[15] S. J. Louis and J. McDonnell, "Learning with case-injected genetic algorithms," *IEEE Trans. Evol. Comput.*, vol. 8, no. 4, pp. 316–328, Aug. 2004.
[16] Y. G. Woldesenbet and G. G. Yen, "Dynamic evolutionary algorithm with variable relocation," *IEEE Trans. Evol. Comput.*, vol. 13, no. 3, pp. 500–513, June 2009.
[17] J. H. Kim, Y. H. Kim, S. H. Choi, and I. W. Park, "Evolutionary multi-objective optimization in robot soccer system for education," *IEEE Comput. Intell. Mag.*, vol. 4, pp. 31–41, Feb. 2009.
[18] J. Hunger and G. Huttner, "Optimization and analysis of force field parameters by combination of genetic algorithm and neural networks," *J. Comput. Chem.*, vol. 20, no. 4, pp. 455–471, Mar. 1999.
[19] B. K. Pathak, S. Srivastava, and K. Srivastava, "Neural network embedded multiobjective genetic algorithm to solve non-linear time–cost tradeoff problems of project scheduling," *J. Sci. Ind. Res.*, vol. 67, no. 2, pp. 124–131, Feb. 2008.
[20] D. Jones, "A taxonomy of global optimization methods based on response surface," *J. Global Optim.*, vol. 21, no. 4, pp. 345–383, 2001.
[21] Z. Zhou, Y. S. Ong, M. H. Nguyen, and D. Lim, "A study on polynomial regression and Gaussian process global surrogate model in hierarchical surrogate-assisted evolutionary algorithm," in *Proc. IEEE Congr. Evolutionary Computation*, 2005, pp. 2832–2839.
[22] R. G. Regis and C. A. Shoemaker, "Local function approximation in evolutionary algorithms for the optimization of costly functions," *IEEE Trans. Evol. Comput.*, vol. 8, no. 5, pp. 490–505, 2004.
[23] A. Brownlee, O. Regnier-Coudert, J. McCall, and S. Massie, "Using a Markov network as a surrogate fitness function in a genetic algorithm," in *Proc. IEEE Congr. Evolutionary Computation*, 2010, pp. 1–8.
[24] Y. Jin, "A comprehensive survey of fitness approximation in evolutionary computation," *Soft Comput.*, vol. 9, no. 1, pp. 3–12, 2005.
[25] A. M. Zhou and Q. F. Zhang, "A surrogate-assisted evolutionary algorithm for minimax optimization," in *Proc. IEEE Congr. Evolutionary Computation*, 2010, pp. 1–7.
[26] H. Kim and S. Cho, "An efficient genetic algorithm with less fitness evaluation by clustering," in *Proc. IEEE Congr. Evolutionary Computation*, 2001, pp. 888–894.
[27] Y. Jin and B. Sendhoff, "Reducing fitness evaluations using clustering techniques and neural network ensembles," in *Proc. Genetic and Evolutionary Computation Conf.*, 2004, pp. 688–699.
[28] T. Senjyu, A. Y. Saber, T. Miyagi, K. Shimabukuro, N. Urasaki, and T. Funabashi, "Fast technique for unit commitment by genetic algorithm based on unit clustering," *Proc. Inst. Elect. Eng.—Gener. Transm. Distrib.*, vol. 152, no. 5, pp. 705–713, Sept. 2005.
[29] X. B. Hu and X. Y. Huang, "Solving TSP with characteristic of clustering by ant colony algorithm," *J. Syst. Simul.*, vol. 16, no. 12, pp. 2683–2686, 2004.
[30] C. Lee, M. Gen, and W. Kuo, "Reliability optimization design using a hybridized genetic algorithm with a neural-network technique," *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, vol. E84A, no. 2, pp. 627–637, Feb. 2001.
[31] L. R. Marim, M. R. Lemes, and A. Dal Pino, "Neural-network-assisted genetic algorithm applied to silicon clusters," *Phys. Rev. A*, vol. 67, no. 3, Mar. 2003.
[32] A. Auger and N. Hansen, "Performance evaluation of an advanced local search evolutionary algorithm," in *Proc. IEEE Congr. Evolutionary Computation*, 2005, pp. 1777–1784.
[33] M. Pelikan, D. E. Goldberg, and F. Lobo, "A survey of optimization by building and using probabilistic models," *Comput. Optim. Appl.*, vol. 21, no. 1, pp. 5–20, 2002.
[34] S. Baluja, "Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning," School Comput. Sci., Carnegie Mellon Univ., Pittsburgh, PA, Tech. Rep. CMU-CS-94-163, 1994.
[35] J. Pena, J. Lozano, and P. Larrãnaga, "Globally multimodal problem optimization via an estimation of distribution algorithm based on unsupervised learning of Bayesian networks," *IEEE Trans. Evol. Comput.*, vol. 13, no. 1, pp. 43–66, Jan. 2005.
[36] C. K. Ting, W. M. Zeng, and T. C. Lin, "Linkage discovery through data mining," *IEEE Comput. Intell. Mag.*, vol. 5, pp. 10–13, Feb. 2010.
[37] Q. Lu and X. Yao, "Clustering and learning Gaussian distribution for continuous optimization," *IEEE Trans. Syst., Man, Cybern. C*, vol. 35, no. 2, pp. 195–204, May 2005.
[38] L. Emmendorfer and A. Pozo, "Effective linkage learning using low-order statistics and clustering," *IEEE Trans. Evol. Comput.*, vol. 13, no. 6, pp. 1233–1246, Dec. 2009.
[39] M. N. Le, Y. S. Ong, and Q. H. Nguyen, "Opt-informatics for schema analysis of binary genetic algorithms," in *Proc. Genetic and Evolutionary Computation Conf.*, 2008, pp. 1121–1122.
[40] M. N. Le and Y. S. Ong, "A frequent pattern mining algorithm for understanding genetic algorithms," in *Proc. Int. Conf. Intelligent Computing*, 2008, pp. 131–139.
[41] F. Streichert, G. Stein, H. Ulmer, and A. Zell, "A clustering based niching method for evolutionary algorithms," in *Proc. Genetic and Evolutionary Computation Conf.*, 2003, pp. 644–645.
[42] O. Aichholzer, F. Aurehammer, B. Brandstatter, T. Ebner, H. Krasser, C. Magele, M. Muhlmann, and W. Renhart, "Evolution strategy and hierarchical clustering," *IEEE Trans. Magn.*, vol. 38, no. 2, pp. 1041–1044, Mar. 2002.
[43] Y. Yang, J. Vincent, and G. Littlefair, "A coarse-grained parallel genetic algorithm employing cluster analysis for multi-modal numerical optimization," in *Proc. 6th Int. Conf. Artificial Evolution*, 2004, pp. 229–240.
[44] G. T. Pulido and C. A. C. Coello, "Using clustering techniques to improve the performance of a multi-objective particle swarm optimizer," in *Proc. Genetic and Evolutionary Computation Conf.*, 2004, pp. 225–237.
[45] S. Y. Park and J. J. Lee, "Improvement of a multi-objective differential evolution using clustering algorithm," in *Proc. IEEE Int. Symp. Industrial Electronics*, 2009, pp. 1213–1217.
[46] J. J. Liang and P. N. Suganthan, "Adaptive comprehensive learning particle swarm optimizer with history learning," in *Proc. Int. Conf. Simulated Evolution and Learning*, 2006, pp. 213–220.
[47] M. J. Zhang, X. Y. Gao, and W. J. Lou, "A new crossover operator in genetic programming for object classification," *IEEE Trans. Syst., Man, and Cybern. B.*, vol. 37, no. 5, pp. 1332–1343, Oct. 2007.
[48] M. Li and H. Y. Tam, "Hybrid evolutionary search method based on clusters," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 23, no. 8, pp. 786–799, Aug. 2001.
[49] J. Kennedy, "Stereotyping: Improving particle swarm performance with cluster analysis," in *Proc. Congr. Evolutionary Computation*, 2000, pp. 1507–1512.
[50] Y. J. Wang, J. S. Zhang, and G. Y. Zhang, "A dynamic clustering based differential evolution algorithm for global optimization," *Eur. J. Oper. Res.*, vol. 183, no. 1, pp. 56–73, Nov. 2007.
[51] S. Y. Ho, L. S. Shu, and J. H. Chen, "Intelligent evolutionary algorithms for large parameter optimization problems," *IEEE Trans. Evol. Comput.*, vol. 8, no. 6, pp. 522–541, Dec. 2004.
[52] Z. H. Zhan, J. Zhang, Y. Li, and Y. H. Shi, "Orthogonal learning particle swarm optimization," *IEEE Trans. Evol. Comput.*, to be published.
[53] X. T. Yan, M. Q. Wu, and B. Sun, "An adaptive LS-SVM based differential evolution algorithm," in *Proc. Int. Conf. Signal Processing System*, 2009, pp. 406–409.
[54] Z. H. Zhan, J. Zhang, Y. Li, and H. S. H. Chung, "Adaptive particle swarm optimization," *IEEE Trans. Syst., Man, and Cybern. B.*, vol. 39, no. 6, pp. 1362–1381, Dec. 2009.
[55] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Trans. Evol. Comput.*, vol. 13, no. 2, pp. 398–417, Apr. 2009
[56] J. Q. Zhang and A. C. Sanderson, "JADE: Adaptive differential evolution with optional external archive," *IEEE Trans. Evol. Comput.*, vol. 13, no. 5, pp. 945–958, Oct. 2009.
[57] J. P. Rosca and D. H. Ballard, "Learning by adapting representations in genetic programming," in *Proc. 1st Conf. Evolutionary Computation*, 1994, pp. 407–412.
[58] J. Zhang, H. S. H. Chung, and W. L. Lo, "Clustering-based adaptive crossover and mutation probabilities for genetic algorithms," *IEEE Trans. Evol. Comput.*, vol. 11, no. 3, pp. 326–335, June 2007.
[59] Z. H. Zhan, J. Xiao, J. Zhang, and W. N. Chen, "Adaptive control of acceleration coefficients for particle swarm optimization based on clustering analysis," in *Proc. IEEE Congr. Evolutionary Computation*, Singapore, Sept. 2007, pp. 3276–3282.
[60] Y. J. Gong, R. T. Xu, J. Zhang, and O. Liu, "A clustering-based adaptive parameter control method for continuous ant colony optimization," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, 2009, pp. 1827–1832.
[61] H. Zhang and J. Lu, "Adaptive evolutionary programming based on reinforcement learning," *Inf. Sci.*, vol. 178, pp. 971–984, 2008.
[62] G. Magyar, M. Johnsson, and O. Nevalainen, "An adaptive hybrid genetic algorithm for the three-matching problem," *IEEE Trans. Evol. Comput.*, vol. 4, no. 2, pp. 135–146, 2000.
[63] J. Zhang, H. S. H. Chung, W. L. Lo, and T. Huang, "Extended ant colony optimization algorithm for power electronic circuit design," *IEEE Trans. Power Electron.*, vol. 24, no. 1, pp. 147–162, Jan. 2009.
[64] Y. Lin and J. Zhang, "A contour method in population-based stochastic algorithms," in *Proc. IEEE Congr. Evolutionary Computation*, 2008, pp. 2393–2400.
[65] A. C. Martinez-Estudillo, C. Hervas-Martinez, and F. J. Matinez-Estudilo, "Hybridization of evolutionary algorithms and local search by means of a clustering method," *IEEE Trans. Syst., Man, and Cybern. B*, vol. 36, no. 3, pp. 534–545, June 2006.
[66] D. Lim, Y. S. Ong, R. Setiawan, and M. Idris, "Classifier-assisted constrained evolutionary optimization for automated geometry selection of orthodontic retraction spring," in *Proc. IEEE Congr. Evolutionary Computation*, 2010, pp. 1–8.
[67] S. D. Handoko, C. K. Kwoh, and Y. S. Ong, "Feasibility structure modeling: An effective chaperon for constrained memetic algorithms," *IEEE Trans. Evol. Comput.*, vol. 14, no. 5, pp. 740–758, Oct. 2010.
[68] Q. H. Nguyen, Y. S. Ong, and M. H. Lim, "A probabilistic memetic framework," *IEEE Trans. Evol. Comput.*, vol. 13, no. 3, pp. 604–623, June 2009.
[69] A. Caponio, G. L. Cascella, F. Neri, N. Salvatore, and M. Sumner, "A fast adaptive memetic algorithm for online and offline control design of PMSM drives," *IEEE Trans. Syst., Man, and Cybern. B*, vol. 37, no. 1, pp. 28–41, Feb. 2007.
[70] N. Noman and H. Iba, "Accelerating differential evolution using an adaptive local search," *IEEE Trans. Evol. Comput.*, vol. 12, no. 1, pp. 107–125, Feb. 2008.