

Combining Model-based and Genetics-based Offspring Generation for Multi-objective Optimization Using a Convergence Criterion

Aimin Zhou, Yaochu Jin, Qingfu Zhang, Bernhard Sendhoff, Edward Tsang

Abstract—In our previous work [1], it has been shown that the performance of multi-objective evolutionary algorithms can be greatly enhanced if the regularity in the distribution of Pareto-optimal solutions is used. This paper suggests a new hybrid multi-objective evolutionary algorithm by introducing a convergence based criterion to determine when the model-based method and when the genetics-based method should be used to generate offspring in each generation. The basic idea is that the genetics-based method, i.e., crossover and mutation, should be used when the population is far away from the Pareto front and no obvious regularity in population distribution can be observed. When the population moves towards the Pareto front, the distribution of the individuals will show increasing regularity and in this case, the model-based method should be used to generate offspring. The proposed hybrid method is verified on widely used test problems and our simulation results show that the method is effective in achieving Pareto-optimal solutions compared to two state-of-the-art evolutionary multi-objective algorithms: NSGA-II and SPEA2, and our previous method in [1].

I. INTRODUCTION

Multi-objective optimization or multi-criterion programming is one of the challenging problems encountered in various engineering and economic problems. In this paper, without the loss of generality, we consider the following multi-objective optimization problem (MOP) in the continuous search space:

$$\min_{X \in \Omega} F(X) = (f_1(X), \dots, f_m(X))^T, \quad (1)$$

where X is the decision vector, $F(X)$ is the corresponding objective vector, and $\Omega \subseteq R^n$ is the decision space.

Many evolutionary algorithms (EAs) have successfully been employed to tackle MOPs over the past decade [2]. Among the large amount of multi-objective evolutionary algorithms (MOEAs), NSGA-II [3] and SPEA2 [4] are two most popular ones. Several important techniques, such as the use of a second population (or an archive) [5], [6], [7], have proved to be able to greatly improve the performance of MOEAs.

In contrast to single objective optimization, the distribution of the Pareto-optimal solutions often shows a high degree of regularity. So far, this regularity has often been exploited implicitly by introducing a local search after evolutionary optimization [8]. A step further to take advantage of such

regularity is the use of a model that captures the regularity of the distribution of the Pareto-optimal solutions [1], [9], [10]. In that paper, a linear or quadratic model is used in odd generations and a crossover and mutation in even generations to produce offspring, which is quite heuristic. In this paper, we suggest a convergence criterion to determine whether the model or the crossover and mutation should be employed for offspring generation. The idea is that the algorithm will benefit from using a model-based offspring generation only when the population shows a certain degree of regularity, i.e., converged in a stochastic sense.

The model-based offspring generation method used in this paper is closely related to a large class of search algorithms known as estimation of distribution algorithms (EDAs) [11] in the evolutionary computation community. EDAs first build probabilistic models to approximate the distribution of selected solutions in the population. Then, new solutions are generated by sampling from the probabilistic models. EDAs have been successfully used in single-objective optimization problems [11], [12], [13].

EDAs have also been extended for multi-objective optimization. Thierens and Bosman have proposed the Mixture-based Iterated Density Estimation Evolutionary Algorithms (MIDEAs) [14], [15]. In their method, $\lfloor \tau N \rfloor$ best performing solutions from the current population (N is population size and $0.0 < \tau < 1.0$) were selected firstly. Then the randomization Euclidean leader algorithm was used to partition the selected points into several clusters. In each cluster, a Gaussian probability model was built to simulate the distribution of the solutions. Then $N - \lfloor \tau N \rfloor$ solutions were sampled one by one from the models. This algorithm has been employed to solve both discrete and continuous problems.

Goldberg and his colleagues have developed several multi-objective EDAs. The multi-objective Bayesian optimization algorithm (mBOA) [16], [17] incorporated the selection operator of NSGA-II into the Bayesian optimization algorithm (BOA) [18], [19]. Bayesian networks were used in this method to estimate the distribution of promising solutions. The multi-objective extended compact genetic algorithm (meCGA) [17] combined the selection operator with extended compact genetic algorithm (ECGA) [20] that used marginal product models to simulate the probability distribution of promising solutions. The multi-objective hBOA (mo-hBOA) [21] combined the hierarchical Bayesian optimization algorithm (hBOA) with the non-dominated sorting genetic algorithm (NSGA-II). All these algorithms used binary coding and they are applied to some deceptive discrete optimization

Aimin Zhou, Qingfu Zhang and Edward Tsang are with the Department of Computer Science, University of Essex, Colchester, Wivenhoe Park, CO4 3SQ, UK.

Yaochu Jin and Bernhard Sendhoff are with the Honda Research Institute Europe, Carl-Legien-Str. 30, 63073 Offenbach, Germany. (Email: yaochu.jin@honda-ri.de)

problems, and the results shown that these methods were better than NSGA-II.

Schwarz and Ocenasek also presented multi-objective Bayesian optimization algorithm (MBOA) [22], [23], in which Bayesian network was integrated into SPEA2 [4]. They applied this algorithm to discrete optimization problems. Laumanns and Ocenasek combined mixed BOA[24] with the selection strategy of SPEA2 and used the method to solve the knapsack problem.

Li and Zhang proposed a hybrid EDA (MOHEDA) for multi-objective 0/1 knapsack problems [25]. A stochastic clustering method was employed to divide a population into a couple of clusters and then a mixture univariate marginal distribution model was built to sample new solutions. A local search method was used on half of the new solutions to improve their performance. The experimental results indicated that MOHEDA outperformed some state-of-the-art algorithms.

Okabe et al proposed a Voronoi-based EDA (VEDA) for multi-objective optimization [10]. The K -means method was used to group the population. In each cluster, the principal component analysis (PCA) was used to map the high-dimensional search space to a low-dimensional space. A Voronoi mesh was built in the low-dimensional space, where each mesh was assigned a probability based on the rank of the solution in the mesh. During offspring generation, a mesh was selected in proportion to the assigned probability and a new solution was randomly created in the mesh. Finally, the offspring were mapped back to the original search space. The selection strategy of the VEDA was the same as that of NSGA-II. The methods was applied to several continuous MOPs and the authors demonstrated that their method performs better than NSGA-II when a limited number of fitness evaluations is allowed.

Different to the conventional EDAs, the model in the multi-objective algorithm we suggested in [1] consists of two parts, namely, a deterministic part and a stochastic part. The deterministic model aims to capture the regularity in the distribution of the population, while the stochastic model attempts to describe the local dynamics of the individuals. The model-based offspring generation method is then hybridized with the crossover and mutation in a heuristic way, i.e., in all odd generations the model-based method, and in all even generations the genetics-based method, is employed to generate offspring.

This paper extends the work in [1] in two main aspects. First, we combine the model-based and genetics-based methods according to a convergence criterion. That is to say, when the population is not converged, the genetics-based approach is used, and when the population is converged, the model-based method is used to generate offspring. Second, a more sophisticated method to construct the stochastic part of the model is introduced.

The remainder of this paper is organized as follows. Section II describes the proposed algorithm, where the modeling technique based on a simplified form of the principal

curve [26], [27], and the convergence criterion are given. Section II defines the experimental setup, including the test problems and the performance metrics used in this work. The simulation results comparing the performance of the proposed algorithm with NSGA-II, SPEA2, and the algorithm in [1] are presented in Section IV. Section V draws some conclusions and proposes further work.

II. THE HYBRID ALGORITHM

A. Principal Curve based Modeling of Regularity

A principal curve is a smooth one-dimensional (1-D) curve that passes through the middle of a set of data points [26]. A principal surface is a two-dimensional (2-D) version of the principal curve. Fig. 1 shows such a principal curve in a 2-D space.

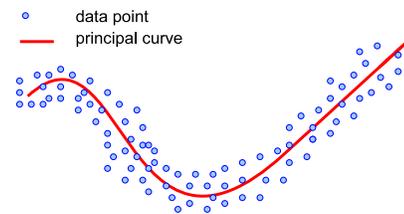


Fig. 1. An example of a principal curve.

Suppose the population $P(t)$ is partitioned into K clusters (sub-populations) $C^k = \{X_i^k | i = 1, \dots, N^k\}, k = 1, \dots, K$ using a local principal component analysis (local PCA) clustering algorithm [28]. The local PCA clustering algorithm is advantageous over the widely used k -means clustering method when the distribution of the data can be better described by a linear curve (a reference vector) rather than a reference point (cluster center in k -means clustering). In the k -th cluster C^k , the i -th biggest eigenvalue is $\lambda_i^k, i = 1, \dots, n$, its corresponding normalized eigenvector is V_i^k , and the mean of cluster C^k is $\bar{X}^k, k = 1, \dots, K$. So we can calculate the projections on the first and second eigenvectors for each point:

$$\begin{aligned} s_{1,i}^k &= (X_i^k - \bar{X}^k)^T V_1^k, \\ s_{2,i}^k &= (X_i^k - \bar{X}^k)^T V_2^k, \end{aligned} \quad (2)$$

where $k = 1, \dots, K$ and $i = 1, \dots, N^k$.

With the partition of the data, we can build a group of linear models to approximate a principal curve or a principal surface. One model is built in each data cluster.

If the MOP is a 2-objective problem, in cluster C^k , a 1-D linear model will be built, which is a line passing through the point with the value of \bar{X}^k . The first eigenvector determines the direction. And the model can be described by:

$$\begin{aligned} H^k(s) &= sV_1^k + \bar{X}^k, \\ s_{min}^k &= \min_{i=1, \dots, N^k} \{s_{1,i}^k\}, \\ s_{max}^k &= \max_{i=1, \dots, N^k} \{s_{1,i}^k\}, \end{aligned} \quad (3)$$

where the latent variable s is a scalar.

For MOPs with three or more objectives, the local principal curve becomes a linear manifold¹. In this case, the point with the value of \bar{X}^k and the first two eigenvectors, V_1^k , and V_2^k , can determine a 2-D plane surface to approximate the principal surface:

$$\begin{aligned} H^k(s) &= s_1 V_1^k + s_2 V_2^k + \bar{X}^k, \\ s_{1,min}^k &= \min_{i=1,\dots,N^k} \{s_{1,i}^k\}, \\ s_{1,max}^k &= \max_{i=1,\dots,N^k} \{s_{1,i}^k\}, \\ s_{2,min}^k &= \min_{i=1,\dots,N^k} \{s_{2,i}^k\}, \\ s_{2,max}^k &= \max_{i=1,\dots,N^k} \{s_{2,i}^k\}, \end{aligned} \quad (4)$$

where the latent variable is a vector $s = (s_1, s_2)^T$.

B. The Probabilistic Model

In our algorithm, the probabilistic model consists of a deterministic model that captures the regularity and a Gaussian model that simulates the local dynamics of the population:

$$H^k = H^k(s) + \xi^k, \quad (5)$$

where $H^k(s)$ is the deterministic model describing the distribution of the solutions, and ξ^k is a random vector with a normal distribution $N(0, (\delta^k)^2 I)$, where I is an $n \times n$ identity matrix and $k = 1, \dots, K$.

In the local PCA clustering process, we need to calculate the distance (denoted by d_i^k) between a point and the reference vector of the k -th cluster. We can determine the standard deviation of the Gaussian model by the following equation:

$$\delta^k = \frac{\sum_{i=1}^{N^k} d_i^k}{N^k \sqrt{n}}, \quad (6)$$

where $k = 1, \dots, K$, N^k is the number of points in the k -th cluster and n is the dimension of decision variables.

The Gaussian model is quite different to the one used in our previous work [1], where we used the second to n -th eigenvalues and eigenvectors to determine the standard deviation of the Gaussian model.

C. Offspring Generation by Sampling from the Model

In the model building stage, we obtain K models:

$$\begin{aligned} H^k &= H^k(s) + \xi^k, \\ \text{s.t. } &s \in [s_{min}^k, s_{max}^k], \end{aligned} \quad (7)$$

where $k = 1, \dots, K$.

With these models, we can create offspring by sampling from the model. The sampling is quite straightforward. For each model $H^k, k = 1, \dots, K$, we can uniformly choose N^k (the number of individual in k -th cluster) latent variables in each range $[s_{min}^k, s_{max}^k]$ and create new solutions from the model in equation (7).

To improve the exploration capability of the algorithm, we also generate offspring by extrapolating the models at the two

extremes, refer to Fig. 2. This can be realized by generating new points in the range of

$$[s_{min}^k - \varepsilon(s_{max}^k - s_{min}^k), s_{max}^k + \varepsilon(s_{max}^k - s_{min}^k)] \quad (8)$$

instead of in the range of $[s_{min}^k, s_{max}^k]$. In Equation (8), ε is the extension ratio, which is a parameter of the algorithm to be defined by the user.

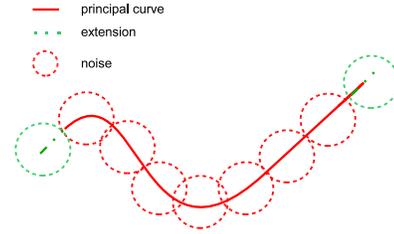


Fig. 2. Model Sampling with Extension.

The extension strategy is another important difference to conventional EDAs. This strategy enables our algorithm to explore the search space at the extremes of the Pareto-optimal front. In our work, the extension ratio is fixed to $\varepsilon = 0.2$. We can use an adaptive strategy to determine this value: if the solutions generated from the extended range can survive in the next generation, we can increase ε in the next generation. Otherwise, we decrease the ratio in the next generation.

D. Convergence Criterion

As we mentioned, the proposed algorithm uses either the genetics-based or the model-based method for generating offspring. Whether the genetics-based or the model-based mechanism should be used depends on the distribution of the population. In other words, when the population has not converged, offspring is generated using the traditional genetics-based operators. When the population shows a certain degree of convergence, the model is used to generate offspring.

There might be different criteria to check if a population has converged. Notice that the convergence in multi-objective optimization is very different to that in single objective optimization. That is, in a converged status, the distribution of an MOO population may be a curve or a surface, but not a point. In this work, we define a convergence criterion for the k -th cluster as follows:

$$\Psi(k) = \begin{cases} \sqrt{\frac{\lambda_2^k}{\lambda_1^k}}, & \text{for 2-objective problems;} \\ \sqrt{\frac{\lambda_3^k}{\lambda_2^k}}, & \text{for 3-objective problems.} \end{cases} \quad (9)$$

In case when $\Psi(k) = 0$, all points in the k -th cluster will lie in a line for 2-objective problems or a plain for 3-objective problems.

The above convergence criterion is used to measure the degree of convergence for the individuals in a cluster to determine which offspring generation strategy should be used. This definition is reasonable because the eigenvalues represent the variance of a population and the squared

¹We assume here that the Pareto front is a $(m-1)$ -dimensional manifold in the decision space if the number of objectives is m . However, for some ill-conditioned problems, the dimension of the Pareto front of an m -objective problem may be lower or higher than $m-1$ in the decision space.

root of the eigenvalues represents the deviation in different directions.

E. Algorithm Framework

The main framework of the proposed algorithm is described in Fig. 3, which works for bi-objective or 3-objective problems.

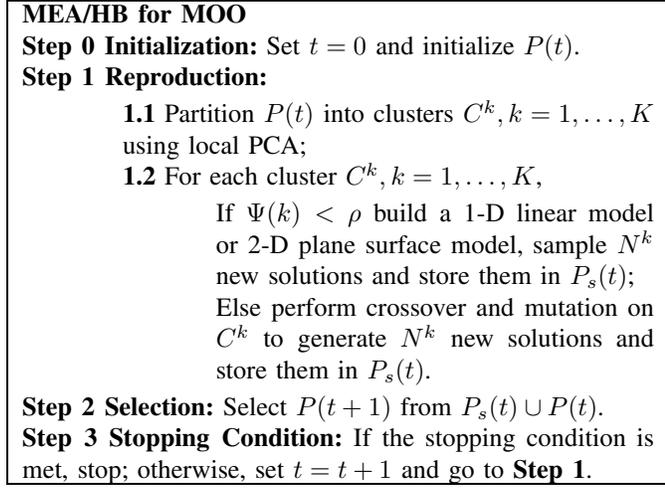


Fig. 3. The framework of the hybrid algorithm.

To distinguish this algorithm with the one published in [1], we term the method in [1] a model-based evolutionary algorithm with a hybrid strategy A (MEA/HA) and the one proposed in this paper a model-based evolutionary algorithm with a hybrid strategy B (MEA/HB).

As in MEA/HA, the simulated binary crossover (SBX- η_c) and the simulated binary mutation (SBM- η_m) proposed in [2] are adopted as the genetic-based method for generating offspring, where η_c and η_m are the distribution parameter of the two operators, respectively.

The differences between MEA/HA and MEA/HB can be summarized as follows:

- 1) In MEA/HA, the model-based method and genetics-based method are used alternatively while in MEA/HB, they are used adaptively according to the convergence criterion $\Psi(k)$. This means that, in MEA/HB, at the beginning stage, the genetics-based offspring generation method will play a major role, while at the later stage, the model-based method will more often be used to generate offspring.
- 2) A new strategy to estimate the standard deviation of the Gaussian model is suggested in MEA/HB and thus the offspring generated by the model have a better quality. Instead of generating only $\frac{N^k}{3}$ individuals in each cluster in MEA/HA, all N^k new solutions are generated from the model in MEA/HB, where N^k is the number of individuals in the k -th cluster.
- 3) In MEA/HB, the SPEA2 archive updating strategy is adopted as the selection operator instead of the NSGA-II selection strategy.

III. EXPERIMENTAL SETUP

A. Test Problems

As shown in Table I, we choose five test problems to check the performance of our method. The Pareto set of OKA5 [29] in the decision space is a highly nonlinear curve. ZDT2.2 [1] is a modified version² of the original ZDT2 [2], whose Pareto set is a continuous curve. ZDT3.2 is a modified version of the original ZDT3 [2] of which the Pareto Set contains 5 discontinuous nonlinear curves. DTLZ2.1 and DTLZ7.1 are rotated versions of DTLZ2 and DTLZ7 [30] by multiplying an orthogonal matrix Q on decision variable X . The Pareto Set of DTLZ2.1 is a plane while DTLZ7.1 has 4 discontinuous parts that lie in a plane. The orthogonal matrix Q used in this work is provided in the Appendix.

The modification of ZDT problems and rotation of DTLZ problems are for the following reasons: (1) the Pareto Sets of these problems are parallel to coordinates; (2) there are no strong interactions among variables of these problems.

B. Performance Metrics

We use four different metrics to measure the performance of the algorithms: Υ metric [3] measures the convergence (closeness of the non-dominated solutions to the Pareto front) of a population; Δ metric [3] measures the diversity of a population; coverage metric C [31] is used to compare two populations; and Λ metric [1] records the run-time performance of an algorithm.

Suppose S is a set of solutions and S^* is a set of known Pareto-optimal solutions, then these four metrics can be defined as follows:

- 1) Convergence metric $\Upsilon(S, Q)$ [3]

This metric is similar to the so called General Distance(GD)³ [32] metric that measures the closeness to the Pareto front. $\Upsilon(S, Q)$ is defined as:

$$\Upsilon(S, S^*) = \frac{1}{|S|} \sum_{X \in S} d(X, S^*), \quad (10)$$

where

$$d(X, S^*) = \min_{Y \in S^*} \|F(X) - F(Y)\|^2.$$

The smaller $\Upsilon(S, S^*)$ is, the closer S to S^* and it is hopeful that $\Upsilon(S, S^*) = 0$ once $S \subseteq S^*$. Since S^* are just part of the true Pareto set and there is an average distance between points in S^* :

$$\bar{d}(S^*, S^*) = \frac{1}{|S^*|} \sum_{X \in S^*} \left\{ \min_{Y \in S^*, Y \neq X} \|F(X) - F(Y)\|^2 \right\}, \quad (11)$$

we can say that S is very close to the true Pareto front once $\Upsilon(S, S^*) \doteq \bar{d}(S^*, S^*)$. The requirement that $\Upsilon(S, S^*)$ equals zero is meaningless in this situation.

- 2) Diversity metric $\Delta(S, Q)$ [3]

²The modified versions of ZDT problems are provided by Hui Li.

³ $GD(S, Q) = \frac{1}{|S|} \sqrt{\sum_{X \in S} d(X, Q)^2}$.

TABLE I
TEST PROBLEMS USED IN THIS PAPER

MOP	n	Constraints	Functions
OKA5[29]	3	$-\pi \leq x_1 \leq \pi$ $-5 \leq x_2, x_3 \leq 5$	$f_1(X) = x_1$ $f_2(X) = \pi - x_1 + x_2 - 5 \cos(x_1) + x_3 - 5 \sin(x_1) $
ZDT2.2[1]	30	$0 \leq x_i \leq 1$ $i = 1, \dots, n$	$f_1(X) = x_1$ $f_2(X) = g(X) \times (1.0 - \frac{f_1}{g(X)})$ $g(X) = 1.0 + \frac{9}{n-1} \sum_{i=2}^n (x_i^2 - x_1)^2$
ZDT3.2	30	$0 \leq x_i \leq 1$ $i = 1, \dots, n$	$f_1(X) = x_1$ $f_2(X) = g(X)[1 - \sqrt{x_1/g(X)} - \frac{x_1}{g(X)} \sin(10\pi x_1)]$ $g(X) = 1 + 9(\sum_{i=2}^n (x_i^2 - x_1)^2)/(n-1)$
DTLZ2.1	12	$-2 \leq x_i \leq 2$ $i = 1, \dots, n$ $0 \leq Y \equiv QX \leq 1$	$f_1(X) = \cos(\frac{\pi}{2}y_1) \cos(\frac{\pi}{2}y_2)(1 + g(Y))$ $f_2(X) = \cos(\frac{\pi}{2}y_1) \sin(\frac{\pi}{2}y_2)(1 + g(Y))$ $f_3(X) = \sin(\frac{\pi}{2}y_1)(1 + g(Y))$ $g(Y) = \sum_{i=3}^n (y_i - 0.5)^2$
DTLZ7.1	12	$-2 \leq x_i \leq 2$ $i = 1, \dots, n$ $0 \leq Y \equiv QX \leq 1$	$f_1(X) = y_1$ $f_2(X) = y_2$ $f_3(X) = [1 + g(Y)]\{3 - \sum_{j=1}^2 \frac{f_j(X)}{g(Y)} [1 + \sin(3\pi f_j(X))]\}$ $g(Y) = 1 + \frac{9}{n-2} \sum_{i=3}^{12} y_i$

The original metric calculates the distance between two consecutive solutions, which works only for 2-objective problems. We make an extension by calculating the distance from a point to its nearest neighbor:

$$\Delta(S, S^*) = \frac{\sum_{i=1}^m d(e_i, S) + \sum_{X \in S^*} |d(X, S) - \bar{d}|}{\sum_{i=1}^m d(e_i, S) + |S^*| \bar{d}}, \quad (12)$$

where $\{e_1, \dots, e_m\}$ are m extreme solutions in S^* and

$$d(X, S) = \min_{Y \in S, Y \neq X} \|F(X) - F(Y)\|^2,$$

$$\bar{d} = \frac{1}{|S^*|} \sum_{X \in S^*} d(X, S).$$

If the achieved solutions are well distributed and include those extreme solutions, $\Delta(S, S^*) = 0$.

3) Run-time metric $\Lambda(t)$ [1]

This metric measures the run-time performance of the algorithms. Let $S(t)$ be the non-dominated set obtained in generation t and if $|S^*| \gg |S(t)|$, $\Upsilon(S^*, S(t))$ represents the diversity to a certain degree. And $\Lambda(t)$ is defined as:

$$\Lambda(t) = \frac{1}{2} [\Upsilon(S(t), S^*) + \Upsilon(S^*, S(t))]. \quad (13)$$

Note that if $\Upsilon(S(t), S^*) \gg \bar{d}(S^*, S^*)$, $\Lambda(t)$ will just represent the convergence and only when $\Upsilon(S(t), S^*) \doteq \bar{d}(S^*, S^*)$, $\Lambda(t)$ will represent convergence and diversity simultaneously.

4) Coverage metric $C(S_1, S_2)$ [31]

This metric is used to compare the achieved non-dominated solutions.

$$C(S_1, S_2) = \frac{|\{X | X \in S_2, \exists Y \in S_1 : F(Y) \prec F(X)\}|}{|S_2|}, \quad (14)$$

where $C(S_1, S_2)$ represents the percentage of the solutions in S_2 that are dominated by at least one solution from S_1 . $C(S_1, S_2)$ is not necessarily equal to $1 - C(S_2, S_1)$. If $C(S_1, S_2)$ is large and $C(S_2, S_1)$ is small, then S_1 are better than S_2 in a sense.

IV. EXPERIMENTAL RESULTS

We compare the two hybrid algorithms (MEA/HA and MEA/HB) with the two well-known algorithms: NSGA-II [3] and SPEA2 [4]. For all 2-objective problems, both the population size and the generations of all methods are 100 while for 3-objective problems, they are 200. For NSGA-II, the crossover probability is 1.0, the mutation probability is 0.1, and the $\eta_c = 20$, $\eta_m = 10$, as suggested in [2]. For SPEA2, the selection tournament size is 2, the individual crossover probability is 1.0, individual mutation probability is 1.0, and the individual swap probability is 0.5. For our proposed method, the extension ratio (ϵ) is 0.2, the training steps for the local PCA is 100, the maximum cluster number (K) is 5, and the convergence threshold is $\rho = 0.4$. The statistical results are based on 30 runs of each method on each problem.

The codes of NSGA-II are from

TABLE II
RESULTS ON DIVERSITY METRIC(Δ)

Method	OKA5		ZDT2.2		ZDT3.2		DTLZ2.1		DTLZ7.1	
	mean	std.								
MEA/HA	0.76398	0.00614	0.77973	0.01963	0.42006	0.01347	N/A	N/A	N/A	N/A
MEA/HB	0.49629	0.02953	0.43829	0.02405	0.34086	0.00902	0.14969	0.00090	0.20325	0.00182
NSGA-II	0.64602	0.02434	0.93524	0.00057	0.75214	0.00332	0.45323	0.00135	0.50941	0.00187
SPEA2	1.06972	0.04723	0.97872	0.00070	0.94170	0.00065	0.47603	0.00156	0.68852	0.00374

TABLE III
RESULTS OF CONVERGENCE METRIC(Υ)

Method	OKA5		ZDT2.2		ZDT3.2		DTLZ2.1		DTLZ7.1	
	mean	std.								
MEA/HA	0.06699	0.00009	0.01409	0.00007	0.00995	0.00000	N/A	N/A	N/A	N/A
MEA/HB	0.10886	0.00215	0.01637	0.00005	0.00392	0.00000	0.04400	0.00006	0.08922	0.00025
NSGA-II	0.07078	0.00041	0.01036	0.00001	0.00754	0.00001	0.08360	0.00007	0.09865	0.00039
SPEA2	0.36103	0.06899	0.00347	0.00000	0.00824	0.00009	0.10639	0.00007	0.15664	0.00365

<http://www.iitk.ac.in/kangal/codes.shtml> and the codes of SPEA2 come from <http://www.tik.ee.ethz.ch/pisa/>.

The diversity metric and convergence metric results are shown in Table II and Table III. Note that MEA/HA works only for two-objective problems, thus no results have been obtained on the two three-objective test functions *DTLZ2.1* and *DTLZ7.1*. From the tables, we can see that: (1) for all test problems, MEA/HB shows the best performance according to the diversity metric; (2) MEA/HA performs a little better than MEA/HB based on the convergence metric on test functions OKA5 and ZDT2.2, while on other three test problems, MEA/HB has the best performance; (3) although for ZDT2.2, SPEA2 has the best convergence performance, it has the worst diversity performance because it can not cover the whole Pareto Front.

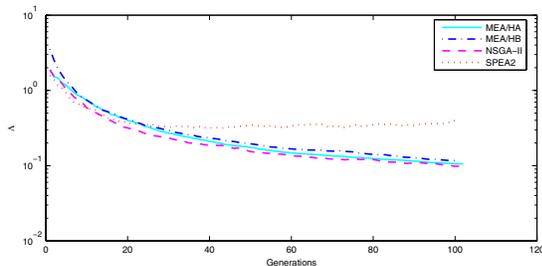


Fig. 4. The run-time performance of the four compared algorithms on OKA5.

The run-time performance (Δ) of the four algorithms are shown in Figs. 4-8. On ZDT2.2, ZDT3.2 and DTLZ2.1, MEA/HB converges faster than other methods, while on test function OKA5, MEA/HA, MEA/HB and NSGA-II have similar convergence speed. On DTLZ7.1, MEA/HB and NSGA-II show similar performance, though MEA/HB is a

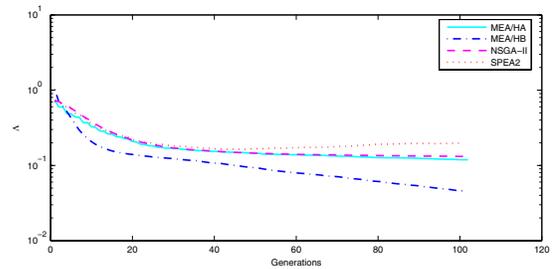


Fig. 5. The run-time performance of the four compared algorithms on ZDT2.2.

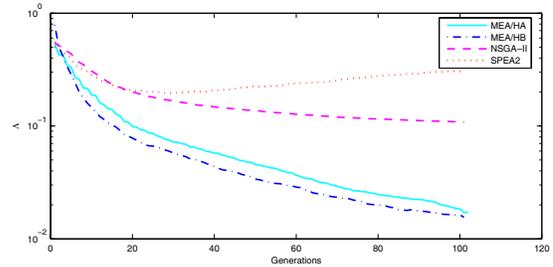


Fig. 6. The run-time performance of the four compared algorithms on ZDT3.2.

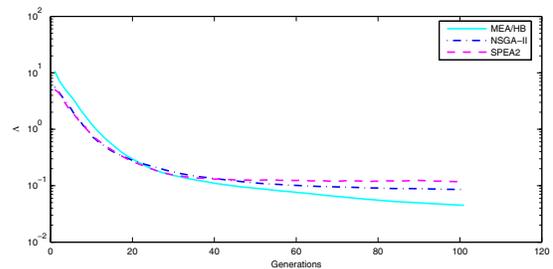


Fig. 7. The run-time performance of the four compared algorithms on DTLZ2.1.

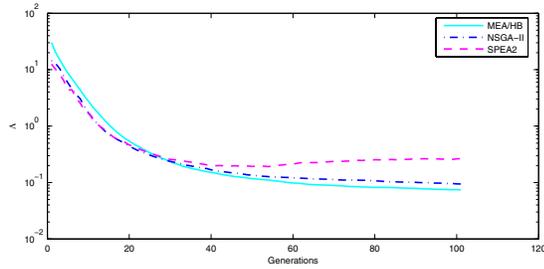


Fig. 8. The run-time performance of the four compared algorithms on *DTLZ7.1*.

little faster than NSGA-II.

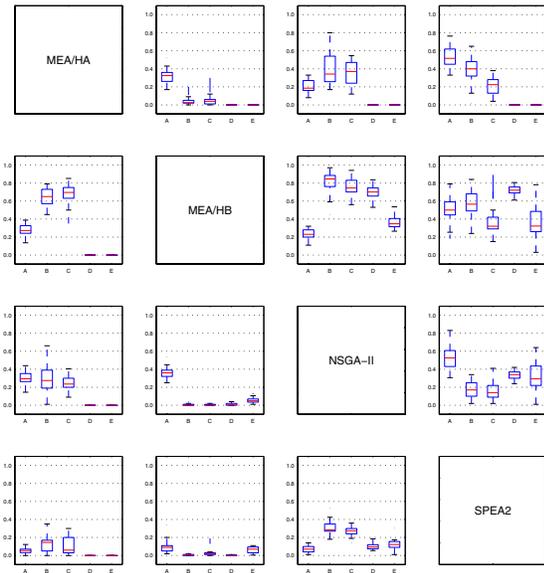


Fig. 9. The coverage test results(A,B,C,D,E are OKA5, ZDT2.2, ZDT3.2, DTLZ2.1 and DTLZ7.1).

The results of the coverage metric are shown in Fig. 9. From the second row of Fig. 9, we can see that on all five test problems, over 20% of solutions of the MEA/HA, NSGA-II and SPEA2 are covered by the solutions of MEA/HB. On the other hand, we can see that except for OKA5, only few solutions of MEA/HB are covered by those of other three algorithms, refer to the second column of Fig. 9. These results indicate that the MEA/HB outperforms the NSGA-II and SPEA2 on all the five test functions, and outperforms the MEA/HA on four of the five test functions. The performance of the MEA/HA and MEA/HB on OKA5 is comparable.

V. CONCLUSIONS

In this paper, we have introduced a new hybrid algorithm for multi-objective optimization by combining model-based and genetics-based offspring generation strategies using a convergence criterion. With the help of this convergence criterion, the algorithm is able to choose the model-based method and the genetics-based method in a more reasonable way. At the early search stage, the genetics-based method is

used more frequently, whereas at the latter stage, offspring are generated more often using the model-based method, thus taking advantage of the regularity in the distribution of the Pareto-optimal solutions. Better performance of the proposed hybrid method is demonstrated on five test functions compared with NSGA-II, SPEA2, and a hybrid algorithm suggested in our previous work.

In the present work, the genetics-based and the model-based offspring generation methods are used separately. Our future work is to develop offspring generation strategies that are able to combine the genetics-based and model-based methods. Besides, modeling techniques other than the local PCA will also be investigated.

REFERENCES

- [1] Aimin Zhou, Qingfu Zhang, Yaochu Jin, Edward Tsang, and Tatsuya Okabe. A model-based evolutionary algorithm for bi-objective optimization. In *Congress on Evolutionary Computation*, Edinburgh, U.K., September 2005. IEEE.
- [2] Kalyanmoy Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, LTD, Baffins Lane, Chichester, 2001.
- [3] Kalyanmoy Deb. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. on Evolutionary Computation*, 6(2):182–197, 2002.
- [4] Eckart Zitzler, Marco Laumanns, and Lothar Thiele. SPEA2: Improving the strength Pareto evolutionary algorithm for multiobjective optimization. In *Evolutionary Methods for Design, Optimisation and Control*, pages 95–100, Barcelona, Spain, 2002. CIMNE.
- [5] Shigeru Obayashi, Shinichi Takahashi, and Yukihiko Takeguchi. Niching and elitist models for mogas. In *Parallel Problem Solving from Nature - PPSN V*, volume 1498 of *Lecture Notes in Computer Science*, pages 260–269, Amsterdam, The Netherlands, September 1998. Springer.
- [6] Geoffrey T. Parks and I. Miller. Selective breeding in a multiobjective genetic algorithm. In *Parallel Problem Solving from Nature - PPSN V, 5th International Conference*, volume 1498 of *Lecture Notes in Computer Science*, pages 250–259, Amsterdam, The Netherlands, September 1998. Springer.
- [7] David A. Van Veldhuizen. *Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations*. PhD thesis, Graduate School of Engineering, the Air Force Institute of Technology, Air University, Wright-Patterson AFB, OH, June 1999.
- [8] Yaochu Jin and Bernhard Sendhoff. Connectedness, regularity and the success of local search in evolutionary multi-objective optimization. In *Congress on Evolutionary Computation*, pages 1910–1917, Canberra, Australia, December 2003. IEEE.
- [9] Yaochu Jin. Rethinking multi-objective evolutionary algorithms. Invited talk, February 2004. Dagstuhl Seminar on Theory of Evolutionary Algorithms, Schloss Dagstuhl, Germany.
- [10] Tatsuya Okabe, Yaochu Jin, Bernhard Sendhoff, and Markus Olhofer. Voronoi-based estimation of distribution algorithm for multi-objective optimization. In *Congress on Evolutionary Computation*, pages 1594–1601, Portland, Oregon, USA, June 2004. IEEE.
- [11] Pedro Larrañaga and Jose A Lozano, editors. *Estimation of Distribution Algorithms*. Kluwer Academic Publishers, 2001.
- [12] Jianyong Sun, Qingfu Zhang, and Edward Tsang. DE/EDA: A new evolutionary algorithm for global optimization. *Information Sciences*, 169(3):249–262, 2005.
- [13] Qingfu Zhang, Jianyong Sun, and Edward Tsang. Evolutionary algorithm with the guided mutation for the maximum clique problem. *IEEE Transactions on Evolutionary Computation*, 9(2):1–9, April 2005.
- [14] Dirk Thierens and Peter A N Bosman. Multi-objective mixture-based iterated density estimation evolutionary algorithms. In *Genetic and Evolutionary Computation Conference*, pages 663–670, San Francisco, California, 2001. Morgan Kaufmann.
- [15] Peter A N Bosman and Dirk Thierens. The naive MIDEA: A baseline multi-objective EA. In Carlos A. Coello Coello, Arturo Hernández Aguirre, and Eckart Zitzler, editors, *Third International Conference on Evolutionary Multi-Criterion Optimization(EMO-2005)*, volume 3410

of *Lecture Notes in Computer Science*, pages 428–442, Guanajuato, Mexico, March 2005. Springer.

- [16] Nazan Khan, David E Goldberg, and Martin Pelikan. Multi-objective bayesian optimization algorithm. In *Genetic and Evolutionary Computation Conference*, page 684, New York, USA, July 2002. Morgan Kaufmann.
- [17] Kumara Sastry, Martin Pelikan, and David E Goldberg. Decomposable problems, niching, and scalability of multiobjective estimation of distribution algorithms. IlliGAL Report 2005004, Illinois Genetic Algorithms Laboratory, 2005.
- [18] Martin Pelikan, David Goldberg, and Erick Cantú-Paz. Boa: The bayesian optimization algorithm. Technical Report 99003, Illinois Genetic Algorithms Laboratory (IlliGAL), 1999.
- [19] Martin Pelikan and Kumara Sastry. Fitness inheritance in the bayesian optimization algorithm. Technical Report 2004009, Illinois Genetic Algorithms Laboratory (IlliGAL), 2004.
- [20] Georges Harik. Linkage learning via probabilistic modeling in the ECGA. Technical Report 99010, Illinois Genetic Algorithms Laboratory (IlliGAL), 1999.
- [21] Martin Pelikan, Kumara Sastry, and David Goldberg. Multiobjective hboa, clustering, and scalability. Technical Report 2005005, Illinois Genetic Algorithms Laboratory (IlliGAL), 2005.
- [22] Josef Schwarz and Jiří Očenášek. Multiobjective bayesian optimization algorithm for combinatorial problems: Theory and practice. *Neural Network World*, 11(5):423–441, 2001.
- [23] Jiří Očenášek and Josef Schwarz. Estimation of distribution algorithm for mixed continuous-discrete optimization problems. In *2nd Euro-International Symposium on Computational Intelligence*, pages 227–232, Kosice, Slovakia, 2002. IOS Press.
- [24] Marco Laumanns and Jiří Očenášek. Bayesian optimization algorithms for multi-objective optimization. In *Parallel Problem Solving From Nature - PPSN VII*, volume 2439 of *Lecture Notes in Computer Science*, pages 298–307, Granada, Spain, September 2002. Springer.
- [25] Hui Li, Qingfu Zhang, Edward P. K. Tsang, and John A. Ford. Hybrid estimation of distribution algorithm for multiobjective knapsack problem. In *Evolutionary Computation in Combinatorial Optimization, 4th European Conference, EvoCOP 2004*, volume 3004 of *Lecture Notes in Computer Science*, pages 145–154, Coimbra, Portugal, April 2004. Springer.
- [26] Trevor Hastie and Werner Stuetzle. Principal curves. *Journal of the American Statistical Association*, 84(406):502–516, June 1989.
- [27] Kui-Yu Chang and Joydeep Ghosh. A unified model for probabilistic principal surfaces. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 23(1):22–41, January 2001.
- [28] Nandakishore Kambhatla and Todd K. Leen. Dimension reduction by local principal component analysis. *Neural Computation*, 9(7):1493–1516, October 1997.
- [29] Tatsuya Okabe. *Evolutionary Multi-Objective Optimization: On the Distribution of Offspring in Parameter and Fitness Space*. PhD thesis, Faculty of Engineering, University of Bielefeld, May 2004.
- [30] Kalyanmoy Deb, Lothar Thiele, Marco Laumanns, and Eckart Zitzler. Scalable test problems for evolutionary multi-objective optimization. In *Evolutionary Computation Based Multi-Criteria Optimization: Theoretical Advances and Applications*. Springer, 2004.
- [31] Eckart Zitzler and Lothar Thiele. Multiobjective optimization using evolutionary algorithms - a comparative case study. In *Parallel Problem Solving from Nature - PPSN V*, volume 1498 of *Lecture Notes in Computer Science*, pages 292–304, Amsterdam, The Netherlands, September 1998. Springer.
- [32] David A. Van Veldhuizen and Gary B. Lamont. Evolutionary computation and convergence to a Pareto front. In *Late Breaking Papers at the Genetic Programming 1998 Conference*, pages 221–228, Madison, Wisconsin, USA, July 1998. Stanford University Bookstore.

APPENDIX

$$Q = \begin{pmatrix} -0.3406 & -0.2320 & 0.0808 & -0.0187 & 0.0978 & -0.2202 & 0.3975 & 0.0514 & -0.1949 & -0.1571 & 0.7358 & -0.0214 \\ -0.2475 & -0.2606 & -0.3147 & 0.3296 & 0.2913 & 0.3231 & -0.1339 & -0.4796 & -0.0366 & -0.3396 & -0.0627 & 0.3247 \\ -0.3013 & -0.0142 & -0.3500 & 0.0798 & 0.0641 & 0.4648 & 0.3311 & 0.5394 & 0.2483 & -0.0308 & -0.1384 & -0.2788 \\ -0.1952 & -0.2310 & -0.2583 & -0.5346 & -0.1156 & -0.1116 & 0.3203 & -0.1678 & -0.4549 & 0.1032 & -0.4253 & 0.0317 \\ -0.2530 & -0.3474 & 0.2895 & -0.0330 & -0.1387 & -0.2800 & -0.2299 & 0.4440 & 0.1243 & -0.4398 & -0.2836 & 0.3057 \\ -0.1320 & 0.1116 & -0.2828 & -0.0711 & 0.1374 & 0.0528 & -0.6330 & 0.3468 & -0.5419 & 0.0825 & 0.1905 & -0.0757 \\ -0.3388 & -0.0854 & 0.5657 & -0.1934 & -0.1994 & 0.5058 & -0.1969 & -0.2334 & -0.0738 & -0.0202 & 0.0148 & -0.3572 \\ -0.4444 & 0.4982 & 0.0410 & -0.1610 & -0.1323 & 0.1129 & 0.0305 & 0.0093 & 0.1467 & 0.2784 & 0.0935 & 0.6227 \\ -0.3096 & 0.0336 & 0.3291 & 0.4501 & 0.4575 & -0.2352 & 0.1135 & 0.0344 & -0.2056 & 0.3959 & -0.3245 & -0.0996 \\ -0.3743 & 0.2032 & -0.2771 & 0.3471 & -0.5612 & -0.3602 & -0.0984 & -0.2098 & 0.0499 & -0.0908 & -0.0511 & -0.3311 \\ -0.1222 & -0.5723 & -0.1697 & -0.0846 & -0.0419 & -0.0888 & -0.2657 & -0.0548 & 0.4076 & 0.5937 & 0.1405 & -0.0016 \\ -0.2220 & 0.2572 & -0.0768 & -0.4462 & 0.5165 & -0.2809 & -0.1596 & -0.1674 & 0.3810 & -0.2296 & -0.0357 & -0.2852 \end{pmatrix}$$