Population-Based Incremental Learning with Immigrants Schemes in Changing Environments

Michalis Mavrovouniotis
Centre for Computational Intelligence (CCI)
School of Computer Science and Informatics
De Montfort University
The Gateway, Leicester LE1 9BH, U.K.
Email: mmavrovouniotis@dmu.ac.uk

Shengxiang Yang
Centre for Computational Intelligence (CCI)
School of Computer Science and Informatics
De Montfort University
The Gateway, Leicester LE1 9BH, U.K.
Email: syang@dmu.ac.uk

***Abstract*—The population-based incremental learning (PBIL) algorithm is a combination of evolutionary optimization and competitive learning. PBIL has been successfully applied to dynamic optimization problems (DOPs). It is well known that maintaining the population diversity is important for PBIL to adapt well to dynamic changes. However, PBIL faces a serious challenge when applied to DOPs because at early stages of the optimization process the population diversity is decreased significantly. It has been shown that random immigrants can increase the diversity level maintained by PBIL algorithms and enhance their performance on some DOPs. In this paper, we integrate elitism-based and hybrid immigrants into PBIL to address slightly and severely changing DOPs. Based on a series of dynamic test problems, experiments are conducted to investigate the effect of immigrants schemes on the performance of PBIL. The experimental results show that the integration of elitism-based and hybrid immigrants with PBIL always improves the performance when compared with a standard PBIL on different DOPs. Finally, the proposed PBIL algorithms are compared with other peer algorithms and show competitive performance.**

I. Introduction

Evolutionary algorithms (EAs) are a class of metaheuristics used to solve different optimization problems. Traditionally, researchers have focused on addressing stationary optimization problems with EAs. However, many real-world problems have a dynamic environment in which the objective function, decision variables, problem instance, constraints, and so on, may vary over time [8]. Dynamic optimization problems (DOPs) are often more challenging to address because the moving optimum needs to be tracked. Since EAs are designed specifically to locate the static optimum quickly, on DOPs, they may not be able to adapt well once converged.

Over the years, several specific strategies have been proposed for EAs on DOPs, which can be classified into the following categories: a) diversity maintaining schemes via immigrants [6], [17]; b) diversity reinforcing schemes [5]; c) memory schemes [3], [19]; d) multi-population schemes [4], [14]; and e) memetic schemes [15]. From the above categories immigrants schemes have received a lot of attention due to their effectiveness and simplicity. They have been integrated with genetic algorithms (GAs) [10], [24], ant colony optimization [9], [11], and constrained hill climbing (CHC) algorithms [13]. Generally, the concept of random immigrants is to introduce randomly generated individuals and replace a small portion of individuals into the evolving population.

In this paper, we investigate the effect of immigrants integrated with the population based incremental learning (PBIL) algorithm. PBIL is an abstraction of EAs, and combines evolutionary optimization with competitive learning [1]. More precisely, PBIL explicitly maintains the statistics contained in an EA's population. Similarly to EAs, PBIL algorithms have been successfully applied to different benchmark problems and real-world applications [12], [20], [22], [23]. PBIL faces the same serious challenge as EAs when addressing DOPs, i.e., premature convergence. In fact, it has been confirmed that PBIL maintains significantly lower diversity than an EA does [23]. Different strategies taken from EAs were integrated into PBIL algorithms to address DOPs, including explicit memory schemes [23], hyper-learning scheme [20], multi-population schemes [18] and random immigrants [22].

Random immigrants were integrated into PBIL to maintain diversity and address the premature convergence. In this paper, other more advanced immigrants schemes are integrated into PBIL algorithms: elitism-based [17] and hybrid immigrants [21] are integrated into PBILs to address slightly and severely changing DOPs, respectively. Elitism-based immigrants use the best of the previous generation as the base to generate immigrants. Hybrid immigrants, apart from random and elitism-based immigrants, use dualism-based immigrants which uses the complementary of the best solution of the previous generation as the base to generate immigrants. Using the exclusive-or (XOR) DOP generator proposed in [16], a series of DOPs are systematically constructed as the dynamic test environments and experiments are carried out to investigate the performance of PBILs with different immigrants schemes and different immigrants replacement ratios. Based on the experimental results, the effect of the different immigrants schemes and different immigrants replacement ratios on the performance of PBILs in dynamic environments is analyzed. In addition, the proposed PBILs are compared with other peer EAs.

The rest of the paper is organized as follows. Section II describes the construction of DOPs used for this study. Section III introduces the standard PBIL algorithm. Section IV

978-1-4799-7560-0/15 $31.00 © 2015 IEEE
DOI 10.1109/SSCI.2015.205

describes the existing PBIL with random immigrants and the proposed PBIL algorithms with elitism and hybrid immigrants. The experimental study is presented in Section V. Finally, Section VI concludes this paper with several observations and relevant future work.

II. Constructing Dynamic Test Environments

The XOR DOP generator [16], [22], [23] can construct dynamic environments from any binary-encoded stationary function $f(\vec{x})$ ($\vec{x} \in \{0,1\}^l$, where l is the length of the binary representation) by a bitwise XOR operator. Suppose the environment changes in every τ algorithmic generations, the dynamics can be formulated as follows:

$$f(\vec{x},t) = f(\vec{x} \oplus \vec{M}(k)), \tag{1}$$

where "$\oplus$" is the XOR operator (i.e., $1 \oplus 1 = 0$, $1 \oplus 0 = 1$, $0 \oplus 0 = 0$), $k = \lceil t/\tau \rceil$ is the index of the period, and $\vec{M}(k)$ is the XORing mask that is constructed incrementally as follows:

$$\vec{M}(k) = \vec{M}(k-1) \oplus \vec{T}(k), \tag{2}$$

where $\vec{T}(k)$ is an intermediate binary template randomly created with $\rho \times l$ ones. Parameters $\rho \in (0.0, 1.0)$ and τ control the magnitude and frequency of change of a DOP, respectively. A higher value of ρ means severer dynamic changes, whereas a lower value of τ means faster dynamic changes.

In this paper, four 100-bit binary-encoded problems are selected as the stationary problems to generate DOPs. Each problem consists of 25 copies of 4-bit building blocks and has an optimum of 100. The first one is the *OneMax* function, which aims to maximize the number of ones in a chromosome. The second one is the *Plateau* function, where each building block contributes four (or two) to the total fitness if its unitation (i.e., the number of ones inside the building block) is four (or three); otherwise, it contributes zero. The third one is the *RoyalRoad* function where each building block contributes four to the total fitness if its unitation is four; otherwise, it contributes zero. The fourth one is the *Deceptive* function, where the building block is a fully deceptive sub-function. Generally, the difficulty of the four functions for optimization algorithms is increasing in the order from OneMax to Plateau to RoyalRoad to Deceptive.

III. Population-based Incremental Learning (PBIL)

The standard PBIL (SPBIL) algorithm, first proposed by Baluja [1], is a combination of evolutionary optimization and competitive learning. The aim of SPBIL is to generate a real-valued probability vector $\vec{P}(t) = \{P_1, \ldots, P_l\}$, at each generation t, which creates high quality solutions with high probability when sampled. Each element $P_i (i = 1, \ldots, l)$ in the probability vector is the probability of creating an allele "1" in locus i. More precisely, a solution is sampled from the probability vector $\vec{P}(t)$ as follows: for each locus i, if a randomly generated number $r \in \{0,1\} < P_i$, it is set to 1; otherwise, it is set to 0.

SPBIL starts from an initial (central) probability vector $\vec{P}(0)$ with values of each entry set to 0.5. This means when sampling by this initial probability vector random solutions are created because the probability of generating a "1" or "0" on each locus is equal. However, as the search progresses, the values in the probability vector are gradually moved towards values representing high evaluation solutions. The evolution process is described as follows.

For every generation t, a set $S(t)$ of samples (solutions) are created according to the current probability vector $\vec{P}(t)$. The set of samples are evaluated according to the problem-specific fitness function. Then, the probability vector is moved towards the solution with the highest fitness $\vec{x}^{bs}(t)$ of the set $S(t)$ as follows:

$$P_i(t+1) \leftarrow (1-\alpha) \times P_i(t) + \alpha \times \vec{x}^{bs}(t), \;\; i = \{1, \ldots, l\}, \tag{3}$$

where α is the learning rate, which determines the distance the probability vector is moved for each generation.

After the probability vector is updated toward the best sample, in order to maintain the diversity of sampling, it may undergo a bit-wise mutation process [2]. Mutation is applied to the SPBIL studied in this paper since diversity maintenance is important when addressing DOPs [23]. The mutation operation always changes the probability vector toward the central probability vector, where values are set to 0.5, to increase exploration. The mutation operation is carried out as follows. The probability of each locus P_i is mutated, if a random number $r \in \{0,1\} < p_m$ (p_m is the mutation probability), as follows:

$$P_i'(t) = \begin{cases} P_i(t) \times (1-\delta_m), & \text{if } P_i(t) > 0.5, \\ P_i(t) \times (1-\delta_m) + \delta_m, & \text{if } P_i(t) < 0.5, \\ P_i(t), & \text{otherwise,} \end{cases} \tag{4}$$

where δ_m is the mutation shift that controls the amount a mutation operation alters the value in each bit position. After the mutation operation, a new set of samples is generated by the new probability vector and this cycle is repeated.

As the search progresses, the entries in the probability vector move away from their initial settings of 0.5 towards either 0.0 or 1.0. The search progress stops when some termination condition is satisfied, e.g., the maximum allowable number of generations is reached or the probability vector is converged to either 0.0 or 1.0 for each bit position. The overall framework of SPBIL is illustrated in Algorithm 1.

PBIL has been applied to many optimization problems with promising results [12]. Most of these applications assume stationary environments, whereas only a few applications consider dynamic environments. To address DOPs with PBILs, the algorithm needs to be enhanced to maintain diversity. Existing strategies, mainly inspired by EAs, have been integrated with PBILs including: associative memory scheme [23], hyper-learning schemes [20], multi-population schemes [18] and random immigrants [22]. In this paper, we integrate to PBILs other immigrants schemes previously used to EAs, i.e., elitism-

Algorithm 1 SPBIL

1: $t \leftarrow 0$
2: $\vec{P}(0) \leftarrow 0.5$
3: $S(0) \leftarrow$ GenerateSamples($\vec{P}(0)$)
4: **while** (termination condition not satisfied) **do**
5: EvaluateSamples($S(t)$)
6: $\vec{x}^{bs}(t) \leftarrow$ FindBestSample($S(t)$)
7: $\vec{P}(t) \leftarrow$ LearnBestFromSample($\vec{x}^{bs}(t)$) using Eq. (3)
8: Mutate($\vec{P}(t)$) using Eq. (4)
9: $S(t) \leftarrow$ GenerateSamples($\vec{P}(t)$)
10: $t \leftarrow t + 1$
11: **end while**

based [17] and hybrid immigrants [21], to address slightly and severely changing DOPs, respectively.

IV. PBIL Enhanced with Immigrants Schemes

A. Random immigrants PBIL (RIPBIL)

The random immigrants scheme is a simple and effective method to address premature convergence in optimization algorithms. Initially, it was proposed in GAs [6] and later on to PBILs [22], denoted as RIPBIL in this paper, to address DOPs. Generally, the diversity of the population in GAs is maintained by introducing random immigrants to replace a small portion of individuals from the evolving population. Usually, the worst individuals are replaced in order to avoid the disruption of the optimization process.

Within PBILs, the immigration process occurs after the probability vector is sampled (e.g., after line 5 in Algorithm 1). More precisely, in every generation, $r_i \times n$ random immigrants are generated and replace the worst samples in the current set of samples $S(t)$, where r_i is the replacement ratio and n is the population size. In this way, the samples of the next set $S(t+1)$ will also consider the randomly generated immigrants, and thus, will have an effect to the diversity maintained.

B. Elitism-based immigrants PBIL (EIPBIL)

The traditional random immigrants schemes described above may increase the diversity level of the population and improve the performance of PBIL for DOPs. However, for slowly changing DOPs, random immigrants may disturb the optimization process, whereas for slightly changing DOPs they may have no effect because the samples in the previous environment may still be fitter than random immigrants in the new environment.

Based on the above consideration, an elitism-based immigrants scheme, which has been initially proposed for GAs [17] to address DOPs, can be integrated with PBILs, denoted as EIPBIL in this paper, to address DOPs with the aforementioned characteristics. Elitism-based immigrants are generated as follows. For each generation t, before the mutation operation, the elite from previous sample $S(t-1)$, i.e., $\vec{x}^{bs}(t-1)$, is used as the base to generate immigrants. In every generation, $r_i \times n$ elitism-based immigrants are generated by mutation bitwise with a probability p_m^i, where r_i and n are defined as in RIPBIL. The generated immigrants replace the worst individuals in the current set of samples $S(t)$. In this way, the samples of the next set $S(t+1)$ will have more directions toward the elite (best solution) found in the previous environment.

C. Hybrid immigrants PBIL (HIPBIL)

The above immigrants schemes can be combined to form hybrid immigrants, which inherits the merits of several immigrants. The PBIL integrated with hybrid immigrants is denoted as HIPBIL in this paper. Within hybrid immigrants, for each generation, apart from the traditional random immigrants and elitism-based immigrants generated in the same way as in EIPBIL, dualism-based immigrants are also generated. Dualism-based immigrants are inspired from the dualism and complementary principle from nature and initially introduced in GAs [21] to address severely or extremely high degree of dynamic changes. Given a binary-encoded sample $\vec{x} = (x_i, \ldots, x_l) \in I = \{0,1\}^l$, its dual $\vec{x}^d$ is defined as follows:

$$\vec{x}^d = dual(\vec{x}) = (x_1^d, \ldots, x_l^d) \in I, \tag{5}$$

where $x_i^d = 1 - x_i (i = 1, \ldots, l)$. The dual of an individual is the one that is symmetric to it with respect to the central point of the search space. Dualism-based immigrants are generated from mutating the dual of the elite from the previous sample $\vec{x}^{bs}(t-1)$ bitwise with a probability p_m^i. The ratios of the three types immigrants that form the HIPBIL are adaptively adjusted according to their relative performance [21]. In this way, the immigrants scheme that performs better (i.e., the best immigrant generated has a higher fitness) is rewarded.

V. Experimental Study

A. Experimental setup

Experiments were carried out to compare different PBILs using the XOR DOP generator described above (see Section II). Dynamic test environments are constructed from the four aforementioned stationary functions with τ set to 10 and 50, indicating quickly and slowly changing DOPs, respectively, and ρ set to 0.1, 0.2, 0.5, 0.8 and 1.0, indicating slightly (i.e., 0.1, 0.2), severely (i.e., 0.5, 0.8), extremely (i.e., 1.0) changing DOPs.

The performance of the conventional SPBIL, is compared with the proposed immigrants-based algorithms, i.e., RIPBIL, EIPBIL and HIPBIL, for the test DOPs. In order to have fair comparisons among PBILs, the population size and immigrant replacement are set to such that each PBIL has 120 fitness evaluations per generation by satisfying the following:

$$(1 + r_i)n = 120 \tag{6}$$

Therefore, for SPBIL the population size $n = 120$, whereas for the immigrants-based algorithms the population size $n = 100$ with the immigrant replacement ratio $r_i = 0.2$ and immigrant mutation probability $p_m^i = 0.01$. For all PBILs, the parameters were set as follows: the learning rate $\alpha = 0.25$, mutation probability $p_m = 0.05$ with the mutation shift $\delta_m = 0.05$, and the elitism of size 1.

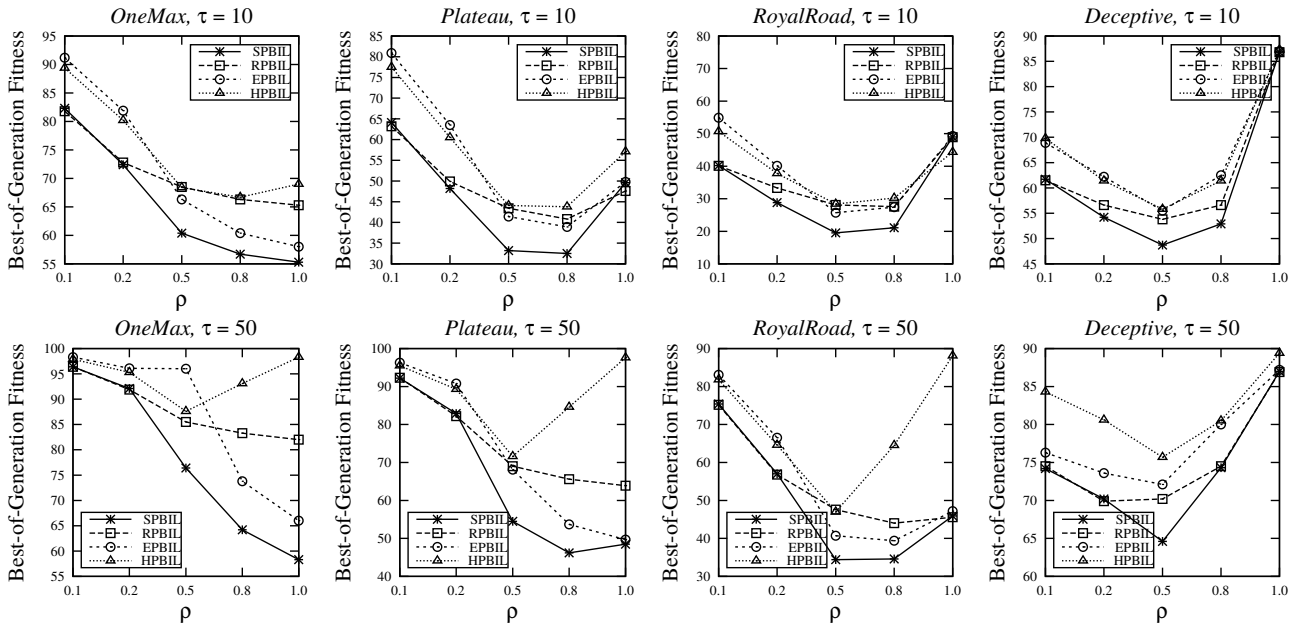

Fig. 1. Experimental results of PBILs on dynamic test problems.

For each PBIL on a DOP, 30 independent runs were executed on the same set of random seeds. For each run of a PBIL, 1000 generations were allowed and the best-of-generation fitness after a dynamic change was recorded every generation. The overall performance of a PBIL on a DOP is defined as follows:

$$\bar{F}_{BOG} = \frac{1}{G}\sum_{i=1}^{G}\left(\frac{1}{N}\sum_{j=1}^{N}F_{BOG_{ij}}\right), \tag{7}$$

where G is the number of generations, N is the number of runs, $F_{BOG_{ij}}$ is the best-of-generation fitness of generation i of run j. Moreover, the diversity of the population was recorded every generation. The overall diversity of a PBIL on a DOP is defined as:

$$\bar{T}_{DIV} = \frac{1}{G}\sum_{i=1}^{G}\left(\frac{1}{N}\sum_{j=1}^{N}Div_{ij}\right), \tag{8}$$

where G and N are defined as in Eq. (7) and Div_{ij} is the diversity at generation i of run j, which is defined as:

$$Div_{ij} = \frac{1}{ln(n-1)}\sum_{p=1}^{n}\sum_{q\neq p}^{n}HD(p,q), \tag{9}$$

where l is the encoding length, n is the population size and $HD(p,q)$ is the hamming distance between the p-th sample and q-th sample.

B. Experimental results of PBILs

The experimental results of the investigated PBILs are shown in Fig. 1. The corresponding statistical results are presented in Table I, where Kruskal–Wallis tests were applied followed by posthoc paired comparisons using Mann–Whitney tests with the Bonferroni correction. The statistical results are shown as "$+$", "$-$" or "$\sim$" when the first algorithm is significantly better than the second algorithm, when the second algorithm is significantly better than the first algorithm, or when the two algorithms are insignificantly different, respectively. The dynamic performance of PBILs with respect to the best-of-generation fitness against generation on the DOPs with $\tau = 50$ and $\rho = 0.2$ is plotted in Fig. 2. Moreover, the population diversity is plotted in Fig. 3 on the corresponding DOPs to better understand the effect of immigrants schemes on PBILs. From Table I and Figs. 1, 2 and 3, the following observations can be drawn.

First, SPBIL is significantly outperformed by RIPBIL, EIPBIL and HIPBIL on most DOPs (except when ρ is set to 0.1 and 0.2 where SPBIL is comparable with RIPBIL); see the comparisons of RIPBIL $\Leftrightarrow$ SPBIL, EIPBIL $\Leftrightarrow$ SPBIL and HIPBIL $\Leftrightarrow$ SPBIL in Table I. This shows that immigrants schemes enhance the performance of PBIL, which can be clearly observed from Fig. 2, where RIPBIL, EIPBIL and HIPBIL maintain higher fitness than SPBIL during the changes. SPBIL may get stuck to a poor local optimum solution because its diversity decreases dramatically early as shown in Fig. 3. In contrast, RIPBIL, EIPBIL and HIPBIL maintain higher, either greater or lower, diversity right after a dynamic change occurs.

Second, EIPBIL outperforms RIPBIL on the OneMax, Plateau and RoyalRoad problems when ρ is set to 0.1 and 0.2 and all dynamic cases of Deceptive problem; see the comparisons of RIPBIL $\Leftrightarrow$ EIPBIL in Table I. Moreover, RIPBIL outperforms EIPBIL in most OneMax, Plateau and

TABLE I
STATISTICAL RESULTS REGARDING THE PERFORMANCE OF DIFFERENT PBILS

Algs & DOPs	*OneMax*					*Plateau*					*RoyalRoad*					*Deceptive*				
$\tau = 10$, $\rho \Rightarrow$	0.1	0.2	0.5	0.8	1.0	0.1	0.2	0.5	0.8	1.0	0.1	0.2	0.5	0.8	1.0	0.1	0.2	0.5	0.8	1.0
RIPBIL $\Leftrightarrow$ SPBIL	$-$	$+$	$+$	$+$	$+$	$-$	$+$	$+$	$+$	$-$	$\sim$	$+$	$+$	$+$	$\sim$	$\sim$	$+$	$+$	$+$	$\sim$
RIPBIL $\Leftrightarrow$ EIPBIL	$-$	$-$	$+$	$+$	$+$	$-$	$-$	$+$	$+$	$-$	$-$	$-$	$+$	$\sim$	$-$	$-$	$-$	$-$	$-$	$-$
RIPBIL $\Leftrightarrow$ HIPBIL	$-$	$-$	$\sim$	$-$	$-$	$-$	$-$	$-$	$-$	$-$	$-$	$-$	$\sim$	$-$	$+$	$-$	$-$	$-$	$-$	$-$
EIPBIL $\Leftrightarrow$ SPBIL	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$
EIPBIL $\Leftrightarrow$ HIPBIL	$+$	$+$	$-$	$-$	$-$	$+$	$+$	$-$	$-$	$-$	$+$	$+$	$-$	$-$	$+$	$-$	$+$	$-$	$+$	$-$
HIPBIL $\Leftrightarrow$ SPBIL	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$
$\tau = 50$, $\rho \Rightarrow$	0.1	0.2	0.5	0.8	1.0	0.1	0.2	0.5	0.8	1.0	0.1	0.2	0.5	0.8	1.0	0.1	0.2	0.5	0.8	1.0
RIPBIL $\Leftrightarrow$ SPBIL	$\sim$	$\sim$	$+$	$+$	$+$	$\sim$	$-$	$+$	$+$	$+$	$\sim$	$\sim$	$+$	$+$	$\sim$	$\sim$	$\sim$	$+$	$\sim$	$-$
RIPBIL $\Leftrightarrow$ EIPBIL	$-$	$-$	$-$	$+$	$+$	$-$	$-$	$+$	$+$	$+$	$-$	$-$	$+$	$+$	$-$	$-$	$-$	$-$	$-$	$-$
RIPBIL $\Leftrightarrow$ HIPBIL	$-$	$-$	$-$	$-$	$-$	$-$	$-$	$-$	$-$	$-$	$-$	$-$	$\sim$	$+$	$-$	$-$	$-$	$-$	$-$	$-$
EIPBIL $\Leftrightarrow$ SPBIL	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$
EIPBIL $\Leftrightarrow$ HIPBIL	$+$	$+$	$+$	$-$	$-$	$\sim$	$+$	$-$	$-$	$-$	$+$	$+$	$-$	$-$	$-$	$-$	$-$	$-$	$-$	$-$
HIPBIL $\Leftrightarrow$ SPBIL	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$

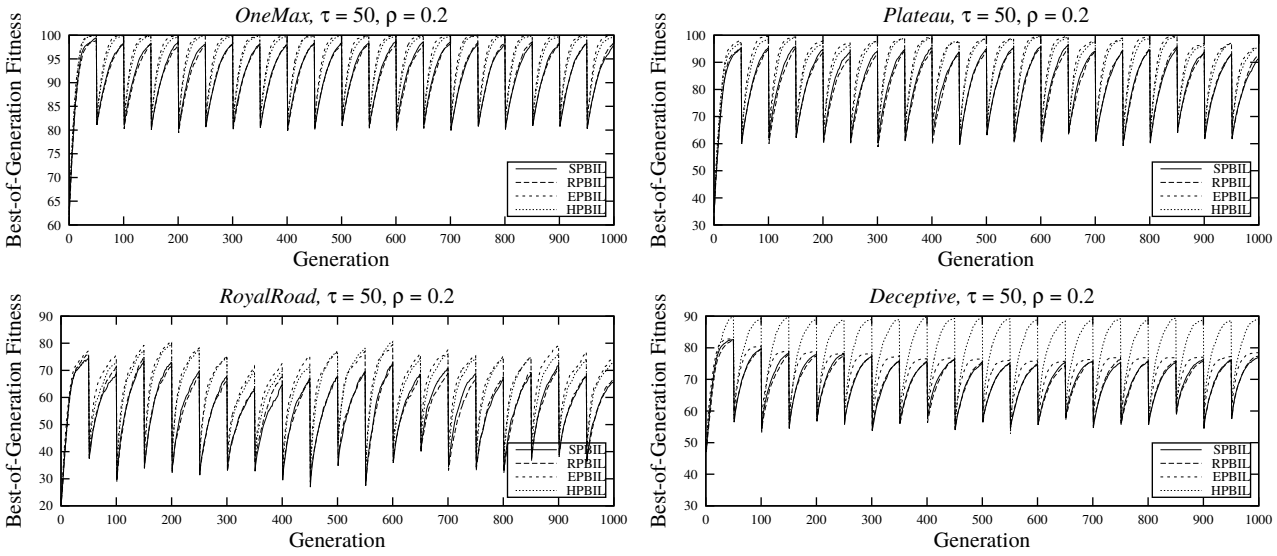

Fig. 2. Dynamic performance of PBILs on dynamic test problems.

RoyalRoad problems when ρ is set to 0.5, 0.8 and 1.0. This supports our expectation above that the performance of PBIL will be improved on slightly and severely changing DOPs when elitism-based and random immigrants are integrated into PBIL, respectively. This is natural because EIPBIL maintains diversity via transferring knowledge for previous environments. Hence, when the changing environments are similar, the knowledge transferred may help to adapt faster to dynamic changes. On the other hand, RIPBIL generates diversity without transferring knowledge. Hence, RIPBIL generates higher levels of diversity, together with the mutation operator, and may disturb the ongoing optimization process. This can be observed from Fig. 3, where RIPBIL maintains higher population diversity than EIPBIL does.

Third, HIPBIL outperforms RIPBIL on most DOPs; see the comparisons of RIPBIL $\Leftrightarrow$ HIPBIL in Table I. In contrast, HIPBIL outperforms EIPBIL on most DOPs when ρ is set to 0.5, 0.8 and 1.0 while it is outperformed on most DOPs (except on the Deceptive) when ρ is set to 0.1 and 0.2; see the comparisons of EIPBIL $\Leftrightarrow$ HIPBIL in Table I. The combination of different immigrants further improves the performance of HIPBIL over RIPBIL in significantly changing DOPs, which supports our expectation above. This may be due to the controlled diversity of HIPBIL caused by elitism-based immigrants. This addresses possible disturbances of the ongoing optimization process that may be caused by random immigrants. This can be supported from Fig. 3, where HIPBIL maintains lower diversity than RIPBIL and higher diversity than EIPBIL.

C. Experimental results of the r_i parameter

The immigrant replacement ratio determines the number of immigrants generated at every generation. In the basic experiments, the replacement ratio was set to a typical value

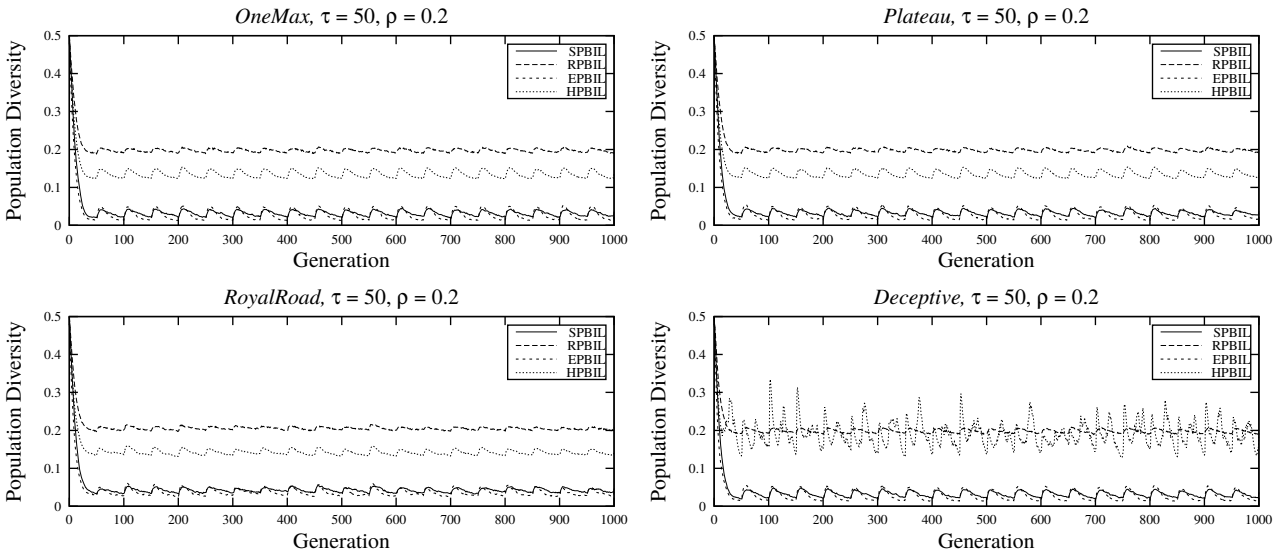

Fig. 3. Dynamic diversity of PBILs on dynamic test problems.

$r_i = 0.2$ for RIPBIL, EIPBIL and HIPBIL. In order to investigate the effect of the replacement ratio further experiments were carried out. The value of the replacement ratio r_i was set from 0.1 to 0.9 and the population size n was set according to Eq. (6). The remaining experimental settings were the same as in the basic experiments above. The experimental results of RIPBIL, EIPBIL and HIPBIL with different immigrants replacement ratios are shown in Fig. 4. From Fig. 4, the following observations can be drawn.

For RIPBIL, smaller replacement ratios (i.e., $r_i = 0.1$ and 0.2) often achieves better performance whereas larger replacement ratios (i.e., $r_i \geq 0.5$) achieves the worst or similar performance on different DOPs. Especially, on most DOPs with $\tau = 50$ and $\rho = 0.1$ and 0.2, the performance is degraded as the replacement ratio increases. This is because the changing environments are similar and the randomly generated immigrants may disturb the optimization process.

For EIPBIL, larger replacement ratios (i.e., $r_i \geq 0.5$) often achieve better performance on most DOPs with $\rho = 0.1$ and 0.2. This is because the generated elitism-based immigrants are still fit in the new environment. On the other hand, different replacement ratios have no effect on the performance for different DOPs because the knowledge transferred may misguide the ongoing optimization process.

For HIPBIL, larger replacement ratios often achieve better performance on different DOPs. This is natural because the hybrid immigrants scheme combines the merits of all immigrants schemes. The ratio of different generated immigrants is adjusted according to the their contribution with respect to the performance.

In summary, the replacement ratio parameter r_i affects the performance of PBILs on the DOPs. EIPBIL and HIPBIL always significantly outperform PBIL with no immigrants (i.e., SPBIL), whereas RIPBIL is not significantly different from SPBIL on DOPs with $\tau = 50$ and $\rho = 0.1$ and 0.2 (see the statistical results in Table I). This shows that the replacement ratio depends on the properties of the DOP and on the immigrant scheme type itself.

D. Experimental results on pairwise comparisons of PBILs with GAs

Since the three immigrants schemes integrated with PBILs were initially introduced and integrated to GAs [6], [17], [21], in this section further pairwise comparisons are performed. Specifically, SPBIL, RIPBIL, EIPBIL and HIPBIL are compared with standard genetic algorithm (SGA) [7], random immigrants GA (RIGA) [6], elitism-based immigrants GA (EIGA) [17] and hybrid immigrants GA (HIGA) [21], respectively. The GAs are executed on the same DOPs and settings with PBILs above (i.e., $G = 1000$ and $N = 30$). For GAs, the parameters were set as follows: generational, uniform crossover with $p_c = 0.6$, flip mutation with $p_m = 0.01$, and fitness proportionate selection with elitism of size 1. For SGA the population size was set to $n = 120$ and for RIGA, EIGA and HIGA it was set to $n = 100$ with immigrant replacement rate of $r_i = 0.2$ according to Eq. (6). The pairwise comparisons regarding the performance are given in Table II. A bold value indicates that the algorithm is significantly better than the other using Mann–Whitney tests. In case both values are bold, it indicates that the algorithms are not significantly different.

From Table II it can be observed that RIPBIL and EIPBIL outperform RIGA and EIGA, respectively, in most DOPs (except RoyalRoad), whereas HIPBIL is outperformed by HIGA in most DOPs. GAs show better performance than PBILs on the RoyalRoad dynamic cases because of the intrinsic characteristics of the function, i.e., only when a building block contains ones it will contribute 4 to the whole fitness,

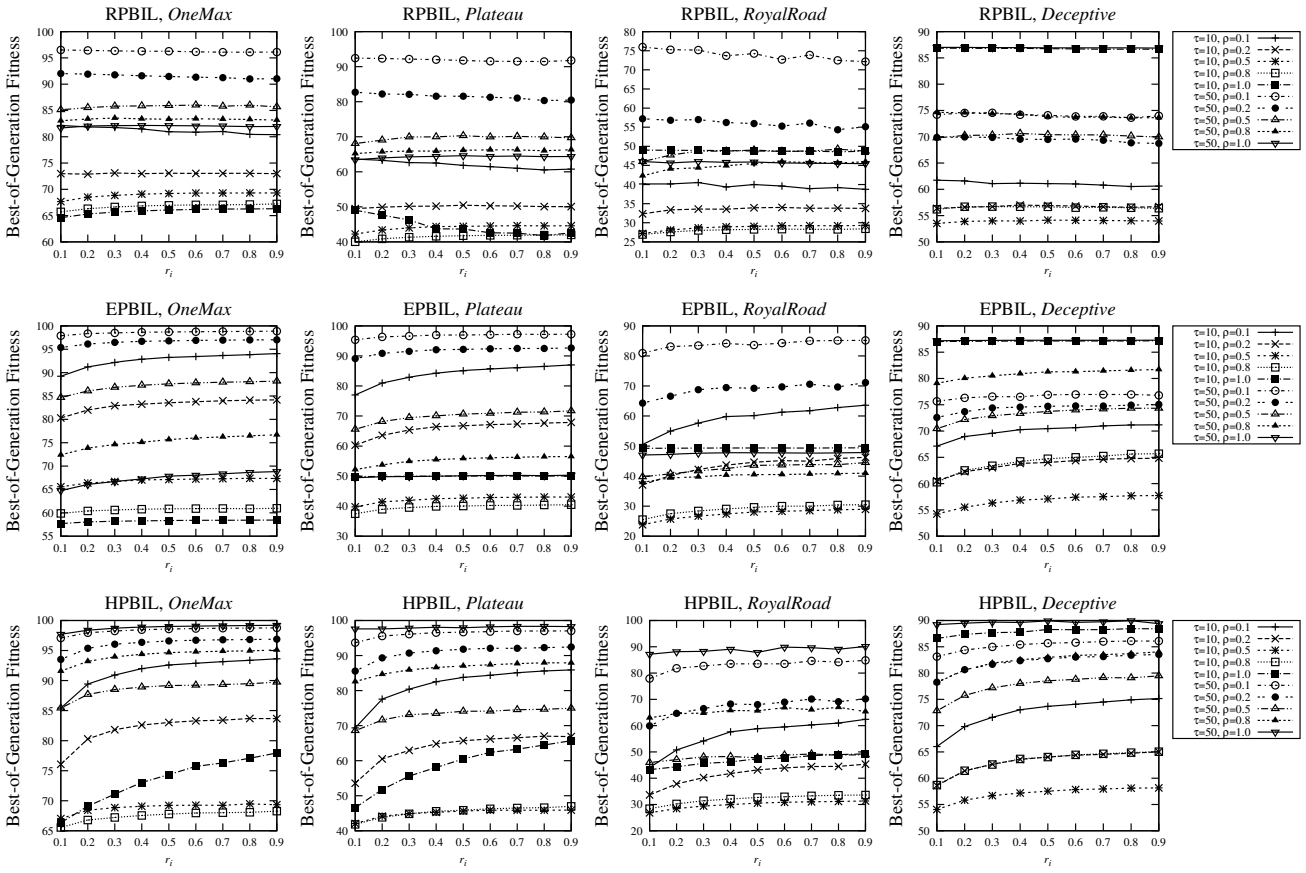

Fig. 4. Experimental results of RIPBIL, EIPBIL and HIPBIL with different immigrant replacement ratios on dynamic test problems.

TABLE II
PAIRWISE COMPARISONS REGARDING THE PERFORMANCE OF PBILS AND GAS

Algs & DOPs	*OneMax*					*Plateau*					*RoyalRoad*					*Deceptive*				
$\tau = 10$, $\rho \Rightarrow$	0.1	0.2	0.5	0.8	1.0	0.1	0.2	0.5	0.8	1.0	0.1	0.2	0.5	0.8	1.0	0.1	0.2	0.5	0.8	1.0
SPBIL (vs)	**82.3**	**72.4**	60.4	56.7	55.3	**64.1**	48.2	33.2	32.5	**49.6**	40.2	28.8	19.5	21.1	**49.0**	**61.7**	**54.2**	48.7	**52.9**	**86.8**
SGA	73.4	69.6	**64.4**	**62.8**	**62.0**	57.6	**49.9**	**39.5**	**36.8**	41.3	**44.1**	**36.3**	**27.1**	**27.1**	39.8	55.1	52.9	**51.2**	**52.8**	67.1
RIPBIL (vs)	**81.8**	**72.8**	**68.5**	**66.3**	**65.3**	**63.2**	49.9	**43.4**	**40.8**	**47.6**	40.1	33.3	28.1	**27.6**	**48.9**	**61.5**	**56.6**	**53.8**	**56.6**	**86.8**
RIGA	74.2	71.0	66.5	64.5	63.8	58.8	**51.4**	42.1	38.9	37.7	**44.3**	**36.7**	**28.3**	27.1	38.7	53.6	52.2	51.3	52.0	56.6
EIPBIL (vs)	**91.2**	**81.9**	**66.3**	**60.4**	**58.0**	**80.9**	**63.5**	**41.4**	**38.9**	**49.8**	**54.9**	40.1	25.7	27.4	**49.3**	**68.9**	**62.2**	**55.5**	**62.5**	**87.0**
EIGA	88.7	80.0	65.5	59.2	56.5	76.6	60.5	40.2	37.0	46.6	**55.8**	**41.2**	**26.8**	**28.0**	44.7	67.6	60.1	54.0	60.1	86.8
HIPBIL (vs)	**89.4**	**80.2**	68.4	66.7	69.0	77.5	60.5	44.1	43.8	**57.1**	50.7	37.8	28.4	30.2	44.3	**69.8**	**61.4**	55.8	**61.4**	**87.3**
HIGA	88.7	**80.3**	**70.0**	**70.7**	**74.1**	**77.9**	**62.0**	**46.3**	**47.3**	56.4	**57.2**	**43.3**	**31.7**	**33.2**	**59.7**	68.7	**61.4**	**57.3**	**61.3**	87.1
$\tau = 50$, $\rho \Rightarrow$	0.1	0.2	0.5	0.8	1.0	0.1	0.2	0.5	0.8	1.0	0.1	0.2	0.5	0.8	1.0	0.1	0.2	0.5	0.8	1.0
SPBIL (vs)	**96.4**	**92.1**	**76.4**	64.2	58.3	**92.2**	**82.9**	54.5	46.2	**48.5**	**75.4**	**57.0**	34.4	34.6	**45.9**	**74.2**	**70.2**	**64.6**	**74.3**	**87.0**
SGA	82.7	79.2	72.3	**68.0**	**65.6**	75.6	69.0	**56.4**	**49.0**	45.8	66.6	**57.9**	**44.8**	**40.3**	41.6	64.2	61.5	58.7	62.2	72.5
RIPBIL (vs)	**96.4**	**91.9**	**85.5**	**83.3**	**82.0**	**92.3**	**82.2**	**69.0**	**65.6**	**63.9**	**75.2**	56.8	**47.6**	**44.0**	**45.6**	**74.5**	**69.9**	**70.2**	**74.5**	**86.9**
RIGA	81.8	78.8	75.3	74.1	73.9	76.6	69.9	60.2	58.4	58.1	67.5	**58.6**	**47.2**	42.8	41.0	61.6	58.7	57.6	59.0	67.9
EIPBIL (vs)	**98.3**	**96.1**	**96.0**	73.8	66.0	**96.3**	**90.8**	68.1	53.7	**49.7**	83.1	66.5	40.7	39.4	**47.2**	76.3	**73.6**	**72.1**	**80.0**	**87.2**
EIGA	97.6	95.1	85.5	**75.5**	**68.8**	95.0	89.5	**69.7**	**55.6**	**49.7**	**85.2**	**73.3**	**51.3**	**43.7**	43.2	**77.1**	**74.0**	71.5	78.7	87.0
HIPBIL (vs)	**97.9**	**95.3**	**87.6**	**93.1**	**98.3**	**95.5**	89.3	71.6	**84.6**	**97.6**	81.8	64.6	47.0	64.5	**88.1**	**84.3**	**80.6**	**75.7**	**80.5**	**89.4**
HIGA	97.7	95.1	**87.7**	88.2	90.1	95.2	**89.8**	**74.3**	80.1	87.4	**85.5**	**73.7**	**55.2**	**66.4**	85.8	83.7	79.7	75.4	79.8	88.6

otherwise it will contribute 0. Hence, the effect of the mutation operator in GAs is significant on such cases.

VI. CONCLUSIONS

Immigrants schemes have been successfully integrated with different metaheuristics to address different DOPs [9], [13],

[17]. In this paper, immigrants schemes are integrated with PBIL to investigate their performance on DOPs. Traditional random immigrants showed promising results on different DOPs when integrated with PBIL previously [22]. Elitism-based and hybrid immigrants are integrated with PBIL to address slightly and severely changing DOPs, respectively, in this paper.

The PBILs integrated with different immigrants schemes are compared against a conventional PBIL and other peer EAs. Moreover, the effect of the immigrant replacement ratio is investigated. From the experimental results, the following concluding remarks can be drawn. First, random immigrants improve the performance in most severely changing DOPs, whereas there is no any significant improvement in slightly changing DOPs. Second, elitism-based immigrants improve the performance in most slightly and slowly changing DOPs because of the knowledge transferred between similar environments. Third, hybrid immigrants further improve the performance on severely changing DOPs when compared with random immigrants whereas it is outperformed on slightly changing DOPs when compared with elitism-based immigrants. Fourth, the immigrant replacement ratio parameter is dependent on the properties of DOP and the immigrant scheme type. Fifth, PBILs with immigrants schemes outperform GAs with immigrants schemes in most DOPs. Finally, the diversity maintained in conventional PBIL is extremely low to address DOPs. Increasing the diversity level of PBIL improves the performance. However, it does not mean that a higher level of diversity will always achieve better performance for DOPs.

For future work, it would be interesting to integrate other immigrants with PBIL, e.g., memory-based immigrants to address cyclically changing DOPs [17] or adaptive immigrants to address both slightly and severely changing DOPs [10]. Since the immigrant replacement ratio is problem-dependent, it would be interesting to self-adapt it in order to avoid possible disturbances of the ongoing optimization process.

ACKNOWLEDGMENT

This work was supported by the Engineering and Physical Sciences Research Council (EPSRC) of UK under Grant EP/K001310/1.

REFERENCES

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

[2] ——, "An empirical comparison of seven iterative and evolutionary function optimization heuristics," Carnegie Mellon University, Pittsburgh, PA, USA, Tech. Rep. CMU-CS-95-193, 1995.

[3] J. Branke, "Memory enhanced evolutionary algorithms for changing optimization problems," in *Proceedings of the 1999 Congress on Evolutionary Computation*, vol. 3, 1999, pp. 1875–1882.

[4] J. Branke, T. Kaussler, C. Smidt, and H. Schmeck, "A multi-population approach to dynamic optimization problems," in *Evolutionary Design and Manufacture*, I. Parmee, Ed. Springer London, 2000, pp. 299–307.

[5] H. G. Cobb, "An investigation into the use of hypermutation as an adaptive operator in genetic algorithms having continuous, time-dependent nonstationary environments," Naval Research Laboratory, Tech. Rep. AIC-90-001, 1990.

[6] J. J. Grefenstette, "Genetic algorithms for changing environments," in *2nd International Conference on Parallel Problem Solving From Nature*, 1992, pp. 137–144.

[7] J. Holland, *Adaption in Natural and Artifical Systems*. Ann Arbor: University of Michigan Press, 1975.

[8] Y. Jin and J. Branke, "Evolutionary optimization in uncertain environments–a survey," *IEEE Transactions on Evolutionary Computation*, vol. 9, no. 3, pp. 303–317, 2005.

[9] M. Mavrovouniotis and S. Yang, "Ant colony optimization with immigrants schemes for dynamic environments," in *Parallel Problem Solving from Nature, PPSN XI*, ser. Lecture Notes in Computer Science, vol. 6239, 2010, pp. 371–380.

[10] ——, "Genetic algorithms with adaptive immigrants for dynamic environments," in *2013 IEEE Congress on Evolutionary Computation (CEC)*, June 2013, pp. 2130–2137.

[11] ——, "Interactive and non-interactive hybrid immigrants schemes for ant algorithms in dynamic environments," in *2014 IEEE Congress on Evolutionary Computation (CEC)*, July 2014, pp. 1542–1549.

[12] P. L. P and J. Lozano, Eds., *Estimation of distribution algorithms: a new tool for evolutionary computation*. Norwell, MA: Kluwer, 2002.

[13] A. Simões and E. Costa, "CHC-based algorithms for the dynamic traveling salesman problem," in *Applications of Evolutionary Computation*, ser. Lecture Notes in Computer Science, vol. 6624, 2011, pp. 354–363.

[14] R. K. Ursem, "Multinational GAs: Multimodal optimization techniques in dynamic environments," in *2000 Genetic and Evolutionary Computation Conference*, 2000, pp. 19–26.

[15] H. Wang, D. Wang, and S. Yang, "A memetic algorithm with adaptive hill climbing strategy for dynamic optimization problems," *Soft Computing*, vol. 13, no. 8-9, pp. 763–780, 2009.

[16] S. Yang, "Non-stationary problem optimization using the primal-dual genetic algorithm," in *The 2003 Congress on Evolutionary Computation CEC '03*, vol. 3, Dec 2003, pp. 2246–2253.

[17] ——, "Genetic algorithms with memory- and elitism-based immigrants in dynamic environments," *Evolutionary Computation*, vol. 16, no. 3, pp. 385–416, 2008.

[18] ——, "Population-based incremental learning with memory scheme for changing environments," in *Proceedings of the 7th Annual Conference on Genetic and Evolutionary Computation*, 2005, pp. 711–718.

[19] ——, "Associative memory scheme for genetic algorithms in dynamic environments," in *Applications of Evolutionary Computing*, ser. Lecture Notes in Computer Science, vol. 3907, 2006, pp. 788–799.

[20] S. Yang and H. Richter, "Hyper-learning for population-based incremental learning in dynamic environments," in *Evolutionary Computation, 2009. CEC '09. IEEE Congress on*, 2009, pp. 682–689.

[21] S. Yang and R. Tinós, "A hybrid immigrants scheme for genetic algorithms in dynamic environments," *International Journal of Automation and Computing*, vol. 4, no. 3, pp. 243–254, 2007.

[22] S. Yang and X. Yao, "Experimental study on population-based incremental learning algorithms for dynamic optimization problems," *Soft Computing*, vol. 9, no. 11, pp. 815–834, 2005.

[23] ——, "Population-based incremental learning with associative memory for dynamic environments," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 5, pp. 542–561, 2008.

[24] X. Yu, K. Tang, T. Chen, and X. Yao, "Empirical analysis of evolutionary algorithms with immigrants schemes for dynamic optimization," *Memetic Computing*, vol. 1, no. 1, pp. 3–24, 2009.