A Multi-Agent Based Evolutionary Algorithm in Non-stationary Environments

Yang Yan, Hongfeng Wang, Dingwei Wang, Shengxiang Yang and Dazhi Wang

***Abstract*— In this paper, a multi-agent based evolutionary algorithm (MAEA) is introduced to solve dynamic optimization problems. The agents simulate living organism features and co-evolve to find optimum. All agents live in a lattice like environment, where each agent is fixed on a lattice point. In order to increase the energy, agents can compete with their neighbors and can also acquire knowledge based on statistic information. In order to maintain the diversity of the population, the random immigrants and adaptive primal dual mapping schemes are used. Simulation experiments on a set of dynamic benchmark problems show that MAEA can obtain a better performance in non-stationary environments in comparison with several peer genetic algorithms.**

I. Introduction

EVOLUTIONARY algorithms (EAs) have been applied widely to solve stationary optimization problems where the fitness curve or the objective function maintain unchanged during the searching process of EA. However most real-world optimization problems are dynamic since the objective function, environmental parameter and/or constraint conditions maybe change over time. For example, in job scheduling production systems, re-scheduling of the jobs is often required due to change of the specification of jobs or malfunction of the machines. For these dynamic optimization problems (DOPs), the goal of an algorithm is no longer to find an optimal solution but to track the moving optima in the search space. In order to enhance the performance of EAs for DOPs, many researchers have developed a number of approaches, including diversity-increasing methods [5], diversity-keeping methods [1], [7], memory-based approaches [3], and multi-population approaches [2], [4].

In recent years, there has been an interesting concern from the artificial intelligence community on agent-based computation methods, which are used to solve various optimization problems [6], [8], [10], [9], [14]. Liu et al. [11] introduced an application of distributed techniques for solving constraint satisfaction problems. They solved the 7000-queen problem by an energy-based multi-agent model. Zhong et al. [25] integrated multi-agent systems with GAs to form a new algorithm for solving the global numerical optimization problem. However, these problems for which agent-based computation methods have been applied are mainly stationary problems. They have rarely been used for addressing DOPs.

In this paper, a hybrid multi-agent based EA (MAEA) is proposed for DOPs. The effect of introducing several diversity schemes, such as immigrant scheme and dualism method, into MAEA to address DOPs is investigated. In the proposed MAEA, a population of agents that represent potential solutions are assigned in a lattice like environment and accomplish their update via competing in a local neighborhood or learning based on the statistical feedback information of the whole system. Just as other EAs, MAEA may gradually converge to one point in the search space during the run, especially when the environment has been stationary for some time. Hence, MAEA may lose its population diversity that is necessary for adapting efficiently to the changing environment. In order to address this problem, two diversity maintaining methods, random immigrants and adaptive dual mapping, are also introduced into our MAEA to improve its performance in dynamic environments.

The rest of this paper is organized as follows. Section II describes the proposed algorithm in detail. Section III presents the dynamic testing suit. Section IV reports the experimental results. Finally, Section V concludes this paper.

II. Proposed Algorithm

A. The Framework of MAEA

According to [9], an agent is a physical or virtual entity that essentially has the following properties:

- First, it is able to live and act in the environment.
- Second, it is able to sense its local environment.
- Third, it is driven by certain purposes.
- Fourth, it has some reactive behaviors.

Multi-agent systems are computational systems in which several agents interact or work together in order to achieve goals. As can be seen, the meaning of an agent is very comprehensive, and what an agent represents is different for different problems. In general, four elements should be defined when multi-agent systems are used to solve problems. The first two are the meaning and the purpose of each agent. The following element is the environment where all agents live and the last is how to define the local environment of one agent since it usually has only local perceptivity.

Yang Yan, Hongfeng Wang, Dingwei Wang and Dazhi Wang are with the School of Information Science and Engineering, Northeastern University, Shenyang 110004, P. R. China (email: yanyangmail@163.com, {hfwang, dwwang}@mail.neu.edu.cn,wongdz@gmail.com).

Shengxiang Yang is with the Department of Computer Science, University of Leicester, University Road, Leicester LE1 7RH, United Kingdom (email: s.yang@mcs.le.ac.uk).

The work by Yang Yan, Hongfeng Wang, Dingwei Wang and Dazhi Wang was supported by the Key Program of National Natural Science Foundation of China under Grant No. 70431003, the Science Fund for Creative Research Group of the National Natural Science Foundation of China under Grant No. 60521003 and the National Science and Technology Support Plan of China under Grant No. 2006BAH02A09. The work by Shengxaing Yang was supported by the Engineering and Physical Sciences Research Council of the United Kingdom under Grant No. EP/E060722/1.

In this paper, some dynamic 0-1 optimization problems will be discussed. Obviously, each agent is a 0-1 array, which represents a candidate solution. The energy value of an agent may be equal to the value of the objective function. More detailed definitions can be described as follows:

Definition 1: An agent L represents a candidate solution to the optimization problem in hand, and can be expressed as a 0-1 array:

$$L = (L^1, L^2, \cdots, L^n), L^i = 0 \ or \ 1, 1 \le i \le n, \qquad (1)$$

where n is the scale of the problem. The value of its energy $E(L)$ is equal to the value of the objective function $f(L)$, that is, $E(L) = f(L)$.

In order to realize the local perceptivity of agents conveniently, the environment is organized as a ring-shaped latticelike structure. All agents live in a latticelike environment, which is called an agent lattice LL. The size of environment is $L_{size} \times L_{size}$.

Each agent is fixed on a lattice-point and only interacts with its neighbors. Suppose that the agent located at (i, j) is represented as $L_{i,j}$. And the agents which can interact with $L_{i,j}$ could be decided by the apperceive range R_s. Thus, the neighbor agents of $L_{i,j}$ could be calculated as follows:

$$L_{k,l}, i - R_s \le k \le i + R_s, j - R_s \le l \le j + R_s. \qquad (2)$$

Due to the ring-shaped lattice structure, if $k < 1$, then $k = k + L_{size}$, if $k > L_{size}$, then $k = k - L_{size}$; if $l < 1$, then $l = l + L_{size}$, if $l > L_{size}$, then $l = l - L_{size}$. The agents that can interact with $L_{i,j}$ are called its neighbor $N_{i,j}$.

Therefore, the agent lattice can be represented as the one in Fig. 1. Each circle represents an agent, the data in a circle represents its position in the lattice, and two agents can interact with each other if and only if there is a line connecting them. In traditional GAs, those individuals that will generate offspring are usually selected from all individuals according to their fitness. Therefore, the global fitness distribution of a population must be determined. But in nature, a global selection does not exist, and the global fitness distribution cannot be determined either.

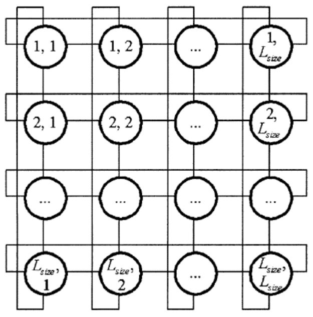


Fig. 1. Model of the agent lattice

```
Procedure general MAEA
initialize all the parameters
$t := 0$
initialize a $L_{size} \times L_{size}$ agent lattice $LL(0)$
evaluate every agent in $LL(0)$
repeat
  for each agent $L_{ij}$ in $LL(t)$ do
    execute one behavior to $L_{ij}$
  $t := t + 1$
until a termination condition is met
```

Fig. 2. Pseudocode for general MAEA

In fact, the real natural selection only occurs in a local environment, and each individual can only interact with those around it. That is, in some phase, the natural evolution is just a kind of local phenomenon. The information can be shared globally only after a process of diffusion. In the agent lattice, to achieve their purposes, agents will compete or cooperate with others so that they can gain more resources. Since each agent can only sense its local environment, its behaviors of competition and cooperation can only take place between the agent and its neighbors. At the same time, the feedback of the best agent can usually influence the behavior of the agents. An agent interacts with its neighbors so that information is transferred to them, and the information feedback from the Lattice can help agents to learn. In such a manner, the information is diffused to the whole agent lattice. As can be seen, the model of the agent lattice is more close to the real evolutionary mechanism in nature than the model of the population in traditional GAs. In summary, the general MAEA can be illustrated in Fig. 2.

B. Behaviors of Agents

In MAEA, all the agents update and evolve by executing a certain behavior. Here, competitive and learning behaviors are defined. For each agent, if its energy is not the best in its neighborhood, then it will execute the competitive behavior; if there's no better agent in the neighborhood, the statistics -based learning behavior is executed to use the information feedback from the whole multi-agent system. The detailed discussion is given as follows.

1) Competitive Behavior: The apperceive range of each agent is 1, and its neighborhood is called competitive neighborhood. Thus, there are 8 agents in the competitive neighborhood of each agent. For agent $L_{i,j}$, it will compare its energy with the agents in its neighborhood. If its energy is not less than any agents in the neighborhood, then it can live; else, it will extinct and its position will be replaced by the offspring of the best agent in its neighborhood. The details are further discussed as follows.

Assume $L_{i,j} = L^1_{i,j}, L^2_{i,j}, \cdots, L^n_{i,j}$, $L_{max} = L^1_{max}, L^2_{max}, \cdots, L^n_{max}$, and $\forall L \in N_{i,j}, E(L) < E(L_{max})$. If $E(L_{i,j}) < E(L_{max})$, then L_{max} is used to generate an offspring $L'_{i,j} = L'^{1}_{i,j}, L'^{2}_{i,j}, \cdots, L'^{n}_{i,j}$ to replace $L_{i,j}$. There are two kinds of generating methods.

In the first competitive method, the information of $L_{i,j}$ and L_{max} are utilized together to generate an offspring $L'_{i,j}$. It randomly chooses some position that $L_{i,j}$ differs from L_{max} to alter the corresponding position in $L_{i,j}$.

$$L'^{k}_{i,j} = \begin{cases} L^{k}_{max}, & k \notin D \text{ or } (k \in D \text{ and random(2)=0}) \\ 1 - L^{k}_{max}, & \text{otherwise,} \end{cases} \tag{3}$$

where $k = 1, 2, \cdots, n$, set D record the sequence number of the position that $L_{i,j}$ differs from $L_{i,j}$. Namely, $D = \{k | L^{k}_{i,j} \neq L^{k}_{max}, k = 1, \cdots, n,\}$. $random(2)$ means to generate 0 or 1 randomly.

The second competitive method is the position based mutation, which is often used in evolutionary computation.

$$L'^{k}_{i,j} = \begin{cases} L^{k}_{max}, & rand() > \frac{1}{n} \\ 1 - L^{k}_{max}, & \text{otherwise,} \end{cases} \tag{4}$$

where $k = 1, 2, \cdots, n$, and $rand()$ means to generate a real number between 0.0 and 1.0.

The number of elements in set D actually equals to the Hamming distance between $L_{i,j}$ and L_{max}. When $|D|$ is small, it means the Hamming distance between $L_{i,j}$ and L_{max} is small, and the two agents are similar. Then the probability of using the first competitive pattern to generate better agent is small, thus we introduce a parameter $D^k \in (0, 1)$ to determine which pattern should be used to generate $L'_{i,j}$: If $|D|/n > D^k$, then the first method is adopted; otherwise, the second one is used.

2) Statistics Based Learning Behavior: The purpose of the learning behavior is to enhance the energy of an agent by statistics based adaptive non-uniform mutation (SANUM) [20] or statistics based adaptive non-uniform crossover (SANUX) [19] with the $elite$ (the best agent in the current generation). Because the resource is limited in the environment, only when the energy of an agent is not less than any of the agents in its neighborhood, it can get a chance to learn. The first learning scheme is statistics based mutation and the second is statistics based crossover with the $elite$ agent. The schemes are described as follows.

First of all, we decide how to calculate the uniform crossover or mutation probability of a locus using a statistics-based approach. Here we use a global information-based statistics, which has been proved to be effective to enhance the exploration capability of GAs by Yang [19], [20]. Let $p(i)$ denote the crossover or mutation probability of the locus i, f_{ki} denote the frequency of $k's$ in the gene locus i over all the past generations, where $i = 1, 2, \cdots, n$ and k is the allele value for the gene locus. Then, in the binary encoding space, we have $f_{1i} + f_{0i} = 1$, $0 \leq f_{0i}, f_{1i} \leq 1$, and f_{1i} can be regarded as the tendency to '1' for the locus i over all the past populations. SANUM and SANUX make use of this convergence information as feedback information to control mutation and crossover by adjusting the probability of each locus. Thus the one-dimension statistic vector $F1 = \{f_{11}, f_{12}, \cdots, f_{1n}\}$ can express the convergence degree of the population from the gene level. Then, $p(i)$ can be calculated from f_{1i} as follows:

$$p(i) = p_{max} - 2 \times |f_{1i} - 0.5| \times (p_{max} - p_{min}), \tag{5}$$

where $|y|$ denotes the absolute value of y, p_{min} and p_{max} denote the minimum and maximum allowable mutation or crossover probability for a locus respectively. For example, for the mutation probability, we have $p_{min} = 10^{-4}$ and $p_{max} = \frac{1}{n}$, and for the crossover probability, we have $p_{min} = 0.0$ and $p_{max} = 0.5$. We first calculate the distribution of 1's $f_1(i)$ for each locus i over the lattice, and from this obtain the crossover and mutation probability $p(i)$ for that gene locus. For the global statistics based method, the learning behavior can be adjusted by the convergence degree of the population.

The first learning scheme is the statistics based adaptive non-uniform mutation (SANUM). If L gets a chance to learn, L is first mutated to generate L'. If $E(L') > E(L)$, then agent L is replaced by L'.

Both SANUM and SANUX belong to the class of adaptive mechanisms that occur at the bottom level of mutation and crossover. Traditional bit mutation and crossover keep a constant mutation and crossover probability over all the loci. As the population converges, in fact fewer and fewer offspring generated by mutating and crossover will survive in the next generation. That is, many operations are wasted on those converged loci. Most of these wasted operations and hence wasted fitness evaluations are saved by SANUM and SANUX through adaptively decreasing $p(i)$ to p_{min} for those converged loci.

We introduce a population index ξ [13] in order to measure its diversity and it can be calculated as follows:

$$\xi = \frac{E_{best} - E_{ave}}{E_{best}}, \tag{6}$$

where E_{best} and E_{ave} are respectively the best and average energy among the energy values of the agents. Obviously, the index ξ can measure the state of agents via the energy calculation. When ξ decreases from 1 to 0, it means that the agents lose the diversity gradually.

The criterion of selecting a learning scheme is based on the value of ξ as follows.

- If $\xi < 0.1$, the first statistics based mutation learning scheme is applied since the mutation operation can help a converging population to jump from a local optimum.
- If $\xi > 0.9$, the second statistics based crossover learning scheme is applied since the crossover operation can accelerate the exploitation to a diversifying population.
- If $0.1 < \xi < 0.9$, one of the two learning schemes is chosen randomly. $Random$ means to generate a real number between 0 and 1. If $Random < 0.5$, the first learning scheme is selected; else, the second one is chosen.

C. Two Methods to Maintain the Diversity of Agents

In dynamic environments, the fitness landscape could change over time, that is, the current optimum point may become a local optimum and the past local optimum maybe become a new global optimum point. Considering a spread-out population can adapt to these changes more easily, two diversity-keeping methods, namely random immigrants (RI) and adaptive dual mapping (ADM), are introduced into our algorithm framework of MAEA for DOPs.

1) Random Immigrants Method (RI): Among the approaches developed for GAs for DOPs, the random immigrants scheme have proved to be beneficial for many DOPs. The random immigrants scheme aims to maintain the diversity of the population by replacing worst or randomly selected individuals from the population with randomly created individuals [26]. Here, we always choose the worst energy agents to be replaced by randomly generated individuals every generation. In order to avoid that random immigrants disrupt the ongoing search progress too much, especially during the period when the environment does not change, the ratio of the number of random immigrants to the population size is usually set to a small value, e.g., 0.1.

2) Adaptive Dual Mapping Method(ADM): Dualism and complementarity are quite common in nature, such as the double-stranded structure in DNA molecules. Inspired by the complementarity mechanism in nature, a primal-dual genetic algorithm has been proposed and applied for DOPs [21]. In this paper, we investigate the application of dualism into MAEA. For the convenience of description, we first introduce the definition of dual agent here. Given an agent L, its dual agent is defined as L', where $L'^k = 1 - L^k, k = 1, \cdots, n$. With this definition, an agent $elite$ is to evaluate its dual $(elite')$ firstly before executing statistics based learning. If its dual is evaluated to have more energy, that is, $E(elite') > E(elite)$, then $elite$ is replaced by $elite'$.

Given the above discussion, the proposed MAEA incorporate two diversity maintaining techniques (RI and ADM) for DOPs can be illustrated in Fig. 3.

```
Procedure proposed MAEA
initialize all the parameters
$t := 0$
initialize a $L_{size} \times L_{size}$ agent lattice $LL(0)$
evaluate every agent in $LL(0)$
calculate crossover and mapping prob. $p(i), i = 1, \cdots, n$
repeat
  for each agent $L_{i,j}$ in $LL(t)$ do
    if there exists an agent $L_{max} \in N_{i,j}$ with
        $E(L_{max}) > E(L_{i,j})$ then
      calculate $|D|/n$, where $|D|$ is the Hamming
        distance between $L_{max}$ and $L_{i,j}$
      if $|D|/n > D^h$ then
        select the first competitive scheme
      else Select the second competitive scheme
    else
      $L'_{i,j}$ := Dual($L_{i,j}$)
      if $(E(L'_{i,j}) > E(L_{i,j}))$ then
        $L_{i,j} := L'_{i,j}$
      if $(\xi < 0.1)$ then
        select the first learning scheme to mutate $L_{i,j}$
      else if $(\xi > 0.9)$ then
        select the second learning scheme to crossover
          $L_{i,j}$ with $elite$, which is the best agent so far
      else
        randomly select the first or second learning
          scheme and execute it for $L_{i,j}$
  calculate $\xi$
  $t := t + 1$
until a termination condition is met
```

Fig. 3. Pseudocode for the proposed MAEA for DOPs.

III. The Testing Suit

A set of well studied stationary problems, forming a range of difficulty levels for EAs, is selected as the experimental functions to compare the performance between MAEA, traditional standard GA (SGA), the Primal-Dual GA (PDGA) [21], and the GA with RI (RIGA), where the worst 10% individuals are replaced with random indiivduals every generation. In this paper, dynamic test problems are constructed from these stationary problems by a dynamic problem generator, which is called the XOR generator.

A. Stationary Test Problems

1) One-Max Function: This problem simply aims to maximize ones in a binary string. The fitness of a string is the number of ones it contains. A string length of 100 bits will be used for this study. And the unique optimum solution has a fitness of 100.

2) Royal Road Function: This function was proposed by Michel, Forrest and Holland [12]. It is defined on a 64 bit string consisting of eight contiguous building-blocks (BBs) and each BB has 8 adjacent bits. The fitness of this function is defined as follows:

$$f(x) = \sum_{i=1}^{8} c_i \delta_i(x), \tag{7}$$

where $c_i = 8$ and $\delta_i = \{1, if x \in S; 0, otherwise\}$. The optimal for this function is given as $f(111\ldots1) = 64$.

3) Deceptive Function: Deceptive functions are a family of functions where there exists low-order BBs that do not combine to form the higher-order BBs. In this study, a deceptive function is constructed, which consists of 10 copies of the order-4 fully deceptive function DF2 [18] and has an optimum fitness value of 300. DF2 is defined as follows:

```
f(0000)=28 f(0001)=26 f(0010)=24 f(0011)=18
f(0100)=22 f(0101)=6  f(0110)=14 f(0111)=0
f(1000)=20 f(1001)=12 f(1010)=10 f(1011)=2
f(1100)=8  f(1101)=4  f(1110)=6  f(1111)=30
```

4) Strongly Interrelated Deceptive Function: A six rank bipolar Strongly Interrelated Deceptive function [15] discussed here consists of 10 copies of 6 rank bipolar deceptive function DF1.

$$f(a) = \sum_{i=1}^{n/6} f_{bipolar6}(a_{6i-5}, a_{6i-4}, a_{6i-3}, a_{6i-2}, a_{6i-1}, a_{6i}) \tag{8}$$

DF1 is designed as follows:

$$f_{bipolar6}(a_1, a_2, a_3, a_4, a_5, a_6) = \begin{cases} 0.9, & \text{if } u = 3 \\ 0.8, & \text{if } u = 2 \text{ or } 4 \\ 0, & \text{if } u = 1 \text{ or } 5 \\ 1, & \text{if } u = 0 \text{ or } 6, \end{cases}$$

where u is the number of 1s in the variable. The strongly interrelated deceptive function has an optimum fitness of 10.

B. Generating Dynamic Test Problems

In this paper, the dynamic test environments are constructed from the above stationary functions using an "XOR" operator [22], [24]. Suppose that the environment is periodically changed every τ generations, the environemntal dynamics can be formulated as follows:

$$f(x, t) = f(x \oplus M(k)), \tag{9}$$

where $k = \lceil t/\tau \rceil$ is the period index, t is the generation counter, and $M(k)$ is the XOR mask for period k. Given a value for parameter ρ, $M(k)$ can be incrementally generated as follows:

$$M(k) = M(k-1) \oplus T(k), \tag{10}$$

where $T(k)$ is an intermediate binary template randomly created for period k containing $\rho \times n$ ones. For the period $k = 1$, $M(1)$ is initialized to be a zero vector.

In this way, we can change the fitness landscape but still keep certain properties of the original fitness landscape, e.g., the total number of optima and fitness values of optima though their locations shifted. For example, if we apply a template $T = 11111$ to a 5-bit One-Max function, the original optimal point $x^* = 11111$ becomes the least fit point while the original worst point $x = 00000$ becomes the new optimal point in the changed landscape, but the optimal fitness value (i.e., 5) and the uniqueness of optimum remain invariant.

In this paper, we construct dynamic versions of above stationary problems. In the periodically randomly shifting version, every 100 generations the fitness landscape is randomly shifted with a randomly created template T that contains 10 percent ones and 90 percent zeros, half ones and half zeros, 90 percent ones and 10 percent zeros respectively. In other words, the frequency of change is set to 100 generations and the severity of change is set to 0.1, 0.5 and 0.9 respectively, that is, the change varies from a small one to a moderate one, then to an intense one in the environment in terms of the Hamming distance.

IV. Experimental Study

A. Experimental Setting

For MAEA, $L_{size} = 10$. When the mutation probability is computated, p_{min} is set to be 10^{-4}, and p_{max} is set to $1/n$. And $p_{min} = 0.0$ and $p_{max} = 0.5$ are set for crossover probability computation. For SGA, RIGA and PDGA, parameters are set as follows: the population size pop_size is set to 100, one-point crossover with a fixed probability $pc = 0.6$, bit mutation with the mutation probability $pm = 0.001$, and fitness proportionate selection with roulette wheel. The best N chromosomes among all the parents and children are always transferred to the next generation population. The best-of-generation fitness was recorded every generation. And for each run of an algorithm on a dynamic problem, 10 periods of environmental changes are allowed. Each experimental result is averaged over 100 runs with different random seeds.

For dynamic optimization problems, there does not exist only one single optimal solution. Hence, it is not enough to compare only optimal solution obtained at last. It is more important to compare the performance of algorithms at every generation. The overall offline performance of an algorithm on a DOP is defined as:

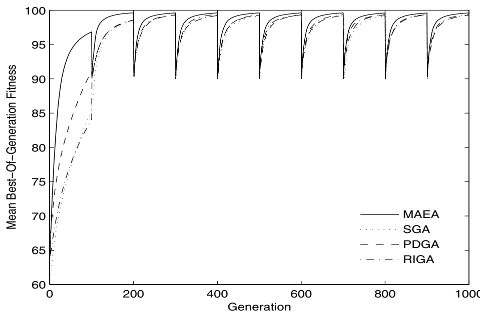
$$\overline{F}_{BOG} = \frac{1}{G} \sum_{i=1}^{G} \left(\frac{1}{100} \sum_{j=1}^{100} \overline{F}_{BOG_{ij}} \right), \tag{11}$$

where $G = 1000$ is the total number of generations for a run and $\overline{F}_{BOG_{ij}}$ is the best-of generation fitness of generation i of run j. The off-line performance $\overline{F}_{BOG}$ is the best-of-generation fitness averaged over 100 runs and then averaged over the data gathering period, i.e., 1000 generations.

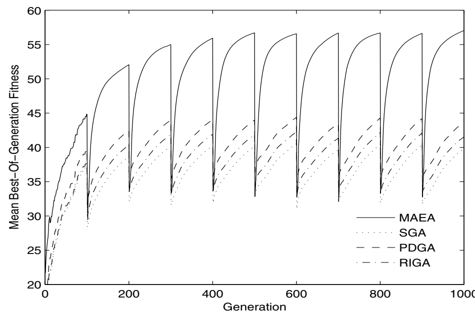
B. Experimental Results

The experimental results with respect to the average-of-generation fitness against evaluations of four EAs on dynamic test problems with $\rho = 0.1$, $\rho = 0.5$, and $\rho = 0.9$ are shown in Figs. 4, 5, and 6 respectively. The corresponding statistical results of comparing algorithms by one-tailed t-test with 38 degrees of freedom at a 0.05 level of significance are given in Table 1 and Table 2 respectively. In Table 1 and 2, the t-test result regarding Alg.1$-$Alg.2 is shown as "$+$", "$-$"and "$\sim$" when Alg.1 is significantly better than, significantly worse than, and statistically equivalent to Alg.2 respectively. From these tables and figures several results can be observed.

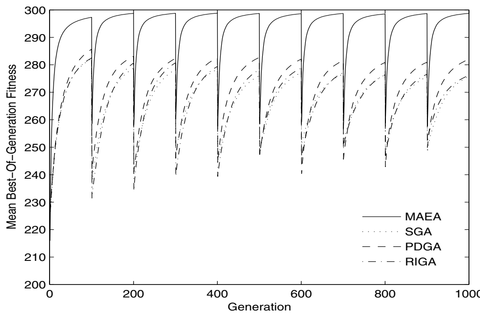
First, when the environment shifted, MAEA can always outperform other algorithms on almost all the problems. MAEA can converge faster and trace the optimal better in both the stationary and dynamic environments. On the One-Max problem , when the environment shifts a little (see Fig. 4(a)), MAEA performs as well as SGA, PDGA and RIGA; when $\rho = 0.5$ (see Fig. 5(a)), MAEA keeps performing perfectly well with the best-of-generation fitness staying in the optimum over all dynamic periods; when $\rho = 0.9$ (see Fig. 6(a)), both MAEA and PDGA performs



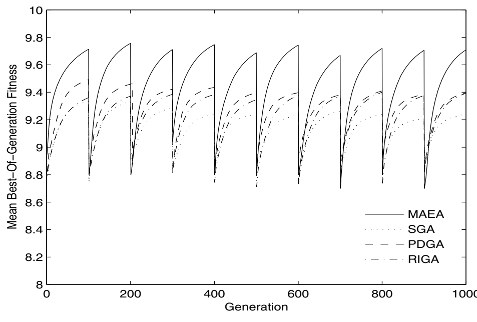
(a)



(b)

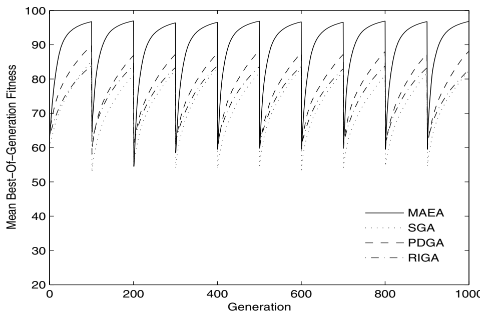


(c)

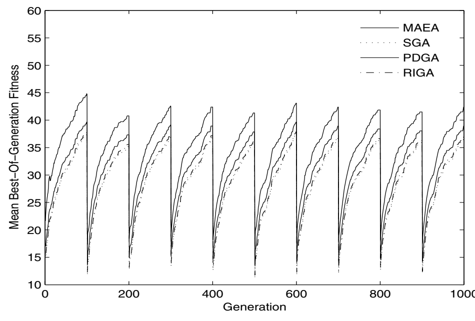


(d)

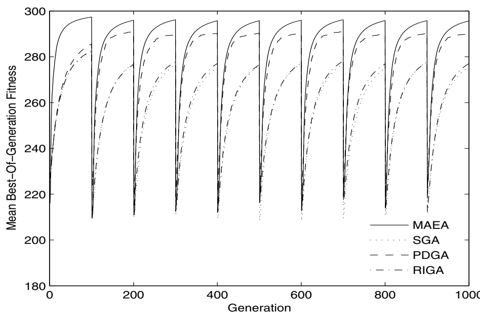
Fig. 4. Experimental results with respect to the mean best-of-generation fitness against evaluations of four EAs on dynamic test problems with $\rho = 0.1$: (a) One-Max, (b) Royal Road, (c) Deceptive, and (d) Strongly Interrelated Deceptive.



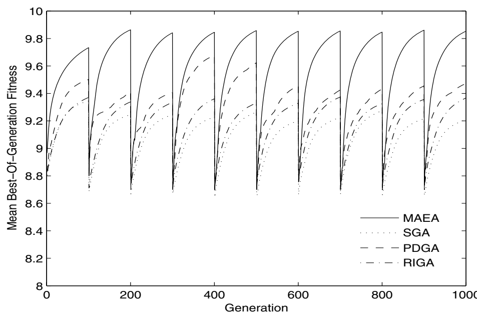
(a)



(b)



(c)



(d)

Fig. 5. Experimental results with respect to mean best-of-generation fitness against evaluations of four EAs on dynamic test problems with $\rho = 0.5$: (a) One-Max, (b) Royal Road, (c) Deceptive, and (d) Strongly Interrelated Deceptive.

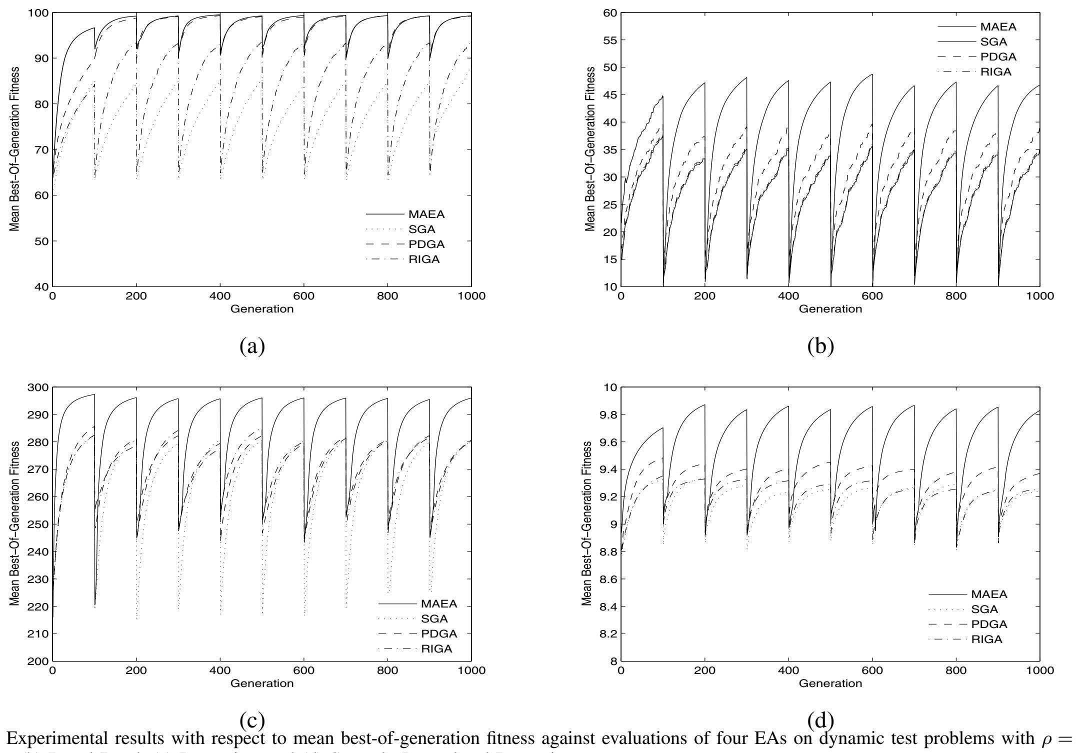


Fig. 6. Experimental results with respect to mean best-of-generation fitness against evaluations of four EAs on dynamic test problems with $\rho = 0.9$: (a) One-Max, (b) Royal Road, (c) Deceptive, and (d) Strongly Interrelated Deceptive.

TABLE I
EXPERIMENTAL RESULTS OF ALGORITHMS ON DYNAMIC PROBLEMS (ALG.1 TO ALG.4 DENOTE $MAEA$, SGA, $PDGA$ AND $RIGA$ RESPECTIVELY).

Function	One-Max Problem				Royal Road Function				Deceptive Function				Strongly Interrelated Deceptive Function			
ρ	Alg.1	Alg.2	Alg.3	Alg.4	Alg.1	Alg.2	Alg.3	Alg.4	Alg.1	Alg.2	Alg.3	Alg.4	Alg.1	Alg.2	Alg.3	Alg.4
0.1	99.6	96.0	98.3	97.4	51.3	36.2	40.1	37.8	295.8	270.1	273.6	273.6	9.49	9.21	9.38	9.29
0.5	91.2	76.3	80.7	78.1	37.7	29.8	32.1	30.4	288.5	268.3	283.8	270.5	9.42	9.17	9.24	9.21
0.9	97.4	78.2	96.1	83.6	41.1	28.3	32.2	29.0	288.6	270.3	279.2	272.3	9.46	9.18	9.30	9.20

TABLE II
STATISTICAL RESULTS OF ALGORITHMS ON DYNAMIC PROBLEMS (ALG.1 TO ALG.4 DENOTE $MAEA$, SGA, $PDGA$ AND $RIGA$ RESPECTIVELY).

t-test Result	One-Max Problem			Royal Road Function			Deceptive Function			Strongly Interrelated Deceptive Function		
$\rho \rightarrow$	0.1	0.5	0.9	0.1	0.5	0.9	0.1	0.5	0.9	0.1	0.5	0.9
Alg.1 - Alg.2	+	+	+	+	+	+	+	+	+	+	+	+
Alg.1 - Alg.3	$\sim$	+	+	+	+	+	+	+	+	+	+	+
Alg.1 - Alg.4	+	+	+	+	+	+	+	+	+	+	+	+
Alg.2 - Alg.3	−	−	−	−	−	−	−	−	−	−	−	−
Alg.2 - Alg.4	−	−	−	−	$\sim$	−	$\sim$	−	$\sim$	−	−	$\sim$
Alg.3 - Alg.4	+	+	+	+	+	+	+	+	+	+	+	+

much better than SGA and RIGA, and also after a change took place, it can be seen from the figure that both MAEA and PDGA can converge fast and make good performance when intense shift happens. On the Royal Road Problem and Deceptive Problem, when $\rho = 0.1$ and $\rho = 0.9$ (see Fig. 4(b), Fig. 4(c), and Fig. 6(b), Fig. 6(c)), MAEA outperforms SGA, PDGA and RIGA; and when the environment half shift (see Fig. 5(b), Fig. 5(c)), both MAEA and PDGA perform well. The average-of-generation fitness first drops a little when changes occur, then rises up quickly to near optimum value. This is because the complementary and dominance mechanisms embedded in MAEA work perfectly under the condition of extreme environment change. On the the Strongly Interrelated Deceptive problem, MAEA outperforms SGA, PDGA and RIGA for all periods in all the shifting environments (see Fig. 4(d), Fig. 5(d) and Fig. 6(d)). For MAEA, whenever the fitness function changes, the statistics based mechanism and diversity maintaining mechanisms in MAEA rapidly draw back the best individuals in the last generation of previous period. And the lattice construction also helps to suit the new environment.

Second, MAEA has a more quickly and robust converge capability than SGA, PDGA and RIGA. And also, it can be seen from the figures, for all the tseting problems, that PDGA has the ability to track the optimal faster than SGA and RIGA due to the primal dual mapping.

Third, the competitive and learning behaviors in MAEA make MAEA have a better exploitation ability than GAs. All the agents accomplish their update via competing in a local neighborhood or learning based on the statistical feedback information of the whole system, so MAEA has a rapid convergence capability.

Fourth, the combination of the two diversity maintaining schemes RI and ADM are introduced into MAEA, hence improve the adaptability of MAEA in non-stationary environments. The information of the former generation is reserved and the diversity is increased, therefore, MAEA with reserved information and multiplex diversity can adapts to the dynamic environment due to effectiveness of the combined maintaining scheme of RI and ADM.

V. Conclusions

This paper investigates the application of a multi-agent based evolutionary algorithm (MAEA) for DOPs. Two special behaviors of agents, competing and learning, are proposed. Two diversity schemes, random immigrants and adaptive dual mapping scheme, are integrated into MAEA in order to improve its performance in dynamic environments. From the experimental results, we can draw the following conclusions on the dynamic test problems.

First, the competing and learning behaviors of agents can always help MAEA obtain a better performance than the studied GAs can do. Second, random immigrants scheme and dual mapping method can both help to increase the population diversity of EAs. Combining them together also improves the performance of MAEA in dynamic environments efficiently. Third, some dynamic characteristics of the environments may affect the performance of algorithms.

Generally speaking, the experimental results indicate that the MAEA with the random immigrants and adaptive dual mapping schemes exhibits very good performance in dynamic environments. MAEA combining some diversity schemes seems to be a good choice to address DOPs.

For future work, it is interesting to hybridize MAEA with other advanced diversity schemes, e.g., hybrid memory schemes [17], [23]. Another interesting work is to further investigate the idea of MAEA to solve other optimization problems with sequential and real encodings in non-stationary environments.

References

[1] T. Blackwell, "Particle swarms and population diversity," *Soft Computing*, vol. 9, pp. 793-802, 2005.

[2] T. Blackwell and J. Branke, "Multi swarms, exclusion, and anti-convergence in dynamic environments," *IEEE Trans. on Evol. Comput.*, vol. 10, no. 4, pp. 459-472, August 2006.

[3] J. Branke, "Memory enhanced evolutionary algorithms for changing optimization problems," *Proc. of the 1999 IEEE Congress on Evol. Comput.*, pp. 1875-1882, 1999.

[4] J. Branke, T. Kaußler, C. Schmidth, and H. Schmeck. "A multi-population approach to dynamic optimization problems," *Proc. of the 5th Int. Conf. on Adaptive Computing in Design and Manufacturing*, pp. 299–308, 2000.

[5] H. G. Cobb and J. J. Grefenstette. Genetic algorithms for tracking changing environments. *Proc. of the 5th Int. Conf. on Genetic Algorithms*, pp. 523–530, 1993.

[6] J. Ferber, *Multi-Agent Systems: An Introduction to Distributed Artificial Intelligenc*, New York: Addison-Wesley, 1999.

[7] J. Grefenstette, "Genetic algorithms for changing environments," *Proc. of the 2nd Int. Conf. on Parallel Problem Solving from Nature*, pp. 137-144, 1992.

[8] N. R. Jennings, K. Sycara, and M. Wooldridge, "A roadmap of agent research and development," *Autonomous Agents and Multi-Agent Systems Journal*, vol. 6, no. 4, pp. 317-331, 1998.

[9] J. Liu, "Autonomous Agents and Multi-Agent Systems," *Explorations in Learning Self-Organization, and Adaptive Computation*, Singapore: World Scientific, 2001.

[10] J. Liu, Y. Y. Tang, and Y. C. Cao, "An evolutionary autonomous agents approach to image feature extraction," *IEEE Trans. on Evol. Comput.*, vol. 1, pp. 141-158, Feb. 1997.

[11] J. Liu, H. Jing, and Y. Y. Tang, "Multi-agent oriented constraint satisfaction," *Artif. Intell.*, vol. 136, no. 1, pp. 101-144, 2002.

[12] M. Mitchell, S. Forest, and J. H. Holland, "The royal road for genetic algorithms: fitness landscape and GA performance," *Proc. of the 1st European Conf. on Artificial Life*, pp. 245-254, 1992.

[13] F. Neri, J. Toivanen, and R. A. E. Makinen, "An adaptive evolutionary algorithm with intelligent mutation local searchers for designing multidrug therapies for HIV," *Applied Intelligence*, 2007.

[14] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, Englewood Cliffs, NJ: Prentice-Hall, 1995.

[15] D. E. Goldberg, K. Deb, and B. Korb, "Messey genetic algorithm revisited: studies in mixed size and scale," *Complex Systems,* vol. 4, no. 4, pp. 145-444, 1990.

[16] H. Wang and D. Wang, "An improved primal-dual genetic algorithm for optimization in dynamic environments," *Proc. of the 13th Int Conf. on Neural Information Processing*, LNCS 4234, vol. 3, pp. 836-844, 2006.

[17] H. Wang, D. Wang, and S. Yang, "Triggered memory-based swarm optimization in dynamic environments," *Applications of Evolutionary Computing*, LNCS 4448, pp. 637-646, 2007.

[18] L. D. Whitley, "Fundamental principles of deception in genetic search," *Foundations of Genetic Algorithms I*, pp. 221-241, 1991.

[19] S. Yang, "Adaptive non-uniform crossover based on statistics for genetic algorithms," *Proc. of the 2002 Genetic and Evolutionary Computation Conference*, pp. 650-657, 2002.

[20] S. Yang, "Adaptive mutation using statistics mechanism for genetic algorithms," In F. Coenen, A. Preece and A. Macintosh (editors), *Research and Development in Intelligent Systems XX*, London: Springer-Verlag, pp. 19-32, 2003.

[21] S. Yang, "PDGA: the primal-dual genetic algorithm," In A. Abraham, M. Koppen, and K. Franke (editors), *Design and Application of Hybrid Intelligent Systems*, Sydney: IOS Press, pp. 214-223, 2003.

[22] S. Yang, Non-stationary problem optimization using the primal-dual genetic algorithm. *Proc. of the 2003 Congress on Evol. Comput.*, vol. 3, pp. 2246-2253, 2003.

[23] S. Yang, "Memory-based immigrants for genetic algorithms in dynamic environments," *Proc. of the 2005 Genetic and Evol. Comput. Conference*, vol. 2, pp. 1115-1122, 2005.

[24] 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.

[25] W. C. Zhong, J. Liu, M. Z. Xue, and L. C. Jiao, "A multiagent genetic algorithm for global numerical optimization," *IEEE Trans. on System, Man, and Cybernetics-Part B.*, vol. 34, no. 2, pp. 1128-1141, 2004.

[26] F. Vavak and T. C. Fogarty, "A comparative study of steady state and generational genetic algorithms for use in nonstationary environments," In T. C. Fogarty(editor), *DAISB Workshop on Evolutionary Computing*, LNCS 1443, pp. 297-304, 1996.