A Multimodal Particle Swarm Optimizer Based on Fitness Euclidean-distance Ratio

Xiaodong Li
School of Computer Science and Information Technology
RMIT University
Melbourne, Australia
xiaodong@cs.rmit.edu.au

ABSTRACT

One of the most critical issues that remains to be fully addressed in existing multimodal evolutionary algorithms is the difficulty in pre-specifying parameters used for estimating how far apart optima are. These parameters are typically represented as some sorts of niching parameters in existing EAs. Without prior knowledge of a problem, it is almost impossible to determine appropriate values for such niching parameters. This paper proposes a PSO for multimodal optimization that removes the need of these niching parameters. Our results show that the proposed algorithm, Fitness Euclidean-distance Ratio based PSO (FER-PSO) is able to reliably locate multiple global optima on the search landscape over some widely used multimodal optimization test functions, given that the population size is sufficiently large.

Categories and Subject Descriptors

G.1 [**Numerical Analysis**]: Optimization; F.2.1 [**Analysis of Algorithms and Problem Complexity**]: Numerical Algorithms and Problems

General Terms

Algorithms, Performance, Experimentation

Keywords

Evolutionary Computation, Swarm Intelligence, Particle Swarm Optimization

1. INTRODUCTION

Evolutionary Algorithms, including more recently Swarm Intelligence Algorithms such as Particle Swarm Optimization (PSO), have shown to be effective and robust optimization methods for solving difficult optimization problems. Though a majority of the EAs and PSOs are specifically designed for locating a single global optimum, many real-world problems demands an EA to search for more than one global optima. The reasons are two folds - firstly many real-world problems are multimodal by nature, which means the search landscape is often characterized by multiple valleys and peaks, where many peaks are all potentially good solutions; secondly, by aiming to search for multiple peaks, the chance of getting stuck on a local peaks is reduced (or in other words, the chance of finding global peaks is increased). Sometimes it is in fact of practical importance to obtain multiple good (or good enough) solutions so that the decision maker can choose the appropriate one according to circumstances.

Many algorithms have been proposed in the EA literature for tackling multimodal optimization problems where multiple optima exist. Some representative ones are crowding methods [12, 17], fitness sharing[10], derating [1], restricted tournament selection[11], parallelization [2] and speciation [21, 15]. PSO variants were also developed to enhance their ability to handle multimodal optimization problems such as NichePSO [5] and SPSO [16, 20]. One of the key drawbacks of these existing algorithms is that some kind of *niching* parameters have to be pre-specified. For example the sharing parameter σ_{share} in fitness sharing [10], the species distance σ_s in SCGA [15], and the species radius r_s in SPSO [16, 20]. The performance of these EAs depends on how these parameters are specified. The main challenge is that different problems would require different appropriate parameter values in order to obtain satisfactory performance. Without prior knowledge of a problem, this is almost an impossible task. Some recent works in [3, 4] aimed to reduce the sensitivity of a speciation-based PSO to the choice of niching parameter values. However, this parameter remains, and some value still needs to be specified.

This paper proposes a multimodal PSO that removes the need to specify any niching parameter. The proposed PSO based on Fitness Euclidean-distance Ratio (FER-PSO) is inspired by FDR-PSO (Fitness-Distance-Ratio based PSO), a previously developed PSO for locating a single global optimum. Though FER-PSO follows the basic idea of FDR-PSO in encouraging the survival of fitter and closer particles, some significant changes have been made. More importantly, FER-PSO is designed for multimodal optimization (though it can also be used to locate just a single peak if there is only one). In FER-PSO, personal bests of particles are used to form a *memory-swarm* to provide a stable network retaining the best points found so far by the population, while the current positions of particles act as parts of an *explorer-swarm* to explore broadly around the search space. Instead of using a single global best, each particle is attracted towards a *fittest-and-closest* neighbourhood point which is identified via computing its FER (Fitness and Euclidean-distance Ratio) value (see section 4). FER-PSO is able to reliably locate all existing global optima over iterations, given that the population size is sufficiently large.

The paper is organized as follows. Section 2 provides a general introduction PSO, including the concepts of *memory-swarm* and *explorer-swarm*, which have direct relevance to the proposed FER-PSO. Section 3 gives some backgrounds on related works. Section 4 describes the proposed FER-PSO, followed by numerical results in section 5. Finally section 6 gives the concluding remarks.

2. PARTICLE SWARM

Particle Swarm Optimization (PSO) is a Swarm Intelligence technique originally developed from studies of social behaviours of animals or insects, eg., bird flocking or fish schooling [14]. Since its inception in 1995 [14], PSO has gained increasing popularity among researchers and practitioners as a robust and efficient technique for solving complex and difficult optimization problems.

Like an Evolutionary Algorithm (EA), PSO is population-based. However, PSO differs from EAs in the way it manipulates each particle (ie., a candidate solution) in the population. Instead of using evolutionary operators such as crossover and mutation, PSO modifies each particle's position in the search space, based on its velocity, some previous best positions it has found so far, and previous best positions found by its neighbours.

In a canonical PSO, the velocity of each particle is modified iteratively by its *personal best* position (ie., the position giving the best fitness value so far), and the position of best particle from the entire swarm. As a result, each particle searches around a region defined by its personal best position and the position of the population best. Let's use $\vec{\mathbf{v}}_i$ to denote the velocity of the i-th particle in the swarm, $\vec{\mathbf{x}}_i$ its position, $\vec{\mathbf{p}}_i$ the best position it has found so far, and $\vec{\mathbf{p}}_g$ the best position found from the entire swarm. $\vec{\mathbf{v}}_i$ and $\vec{\mathbf{x}}_i$ of the i-th particle in the swarm are updated according to the following two equations [7]:

$$\begin{aligned}\vec{\mathbf{v}}_i &\leftarrow \chi(\vec{\mathbf{v}}_i + \vec{\mathbf{R}}_1[0, \frac{\varphi_{max}}{2}] \otimes (\vec{\mathbf{p}}_i - \vec{\mathbf{x}}_i) + \\ &\quad \vec{\mathbf{R}}_2[0, \frac{\varphi_{max}}{2}] \otimes (\vec{\mathbf{p}}_g - \vec{\mathbf{x}}_i)), \quad (1)\\ \vec{\mathbf{x}}_i &\leftarrow \vec{\mathbf{x}}_i + \vec{\mathbf{v}}_i, \quad (2)\end{aligned}$$

where $\vec{\mathbf{R}}_1[0, \frac{\varphi_{max}}{2}]$ and $\vec{\mathbf{R}}_2[0, \frac{\varphi_{max}}{2}]$ are two separate functions each returning a vector comprising random values uniformly generated in the range $[0, \frac{\varphi_{max}}{2}]$. φ_{max} is a positive constant. The symbol $\otimes$ denotes point-wise vector multiplication. A constriction coefficient χ is used to prevent each particle from exploring too far away in the search space, since χ applies a dampening effect to the oscillation size of the particle over time [14]. This Type 1" constricted PSO suggested by Clerc and Kennedy is often used with χ set to 0.7298, calculated according to $\chi = \frac{2}{\left|2-\varphi_{max}-\sqrt{\varphi_{max}^2-4\varphi_{max}}\right|}$, where $\varphi_{max} = 4.1$ [7][9].

In [7], Clerc and Kennedy showed that the Type 1" constriction PSO is equivalent to the early PSO proposed by Kennedy and Eberhart using an inertia weight [14]. Clerc and Kennedy also showed that equation (1) can be simplified to the following:

$$\vec{\mathbf{v}}_i \leftarrow \chi(\vec{\mathbf{v}}_i + \vec{\varphi}_m \otimes (\vec{\mathbf{p}}_m - \vec{\mathbf{x}}_i)), \quad (3)$$

where on each dimension d of $\vec{\mathbf{p}}_m$, $p_{md} = \frac{\varphi_{1d} \cdot p_{id} + \varphi_{2d} \cdot p_{gd}}{\varphi_{1d} + \varphi_{2d}}$. φ_{1d} and φ_{2d} are two random numbers uniformly and independently chosen from the range $[0, \frac{\varphi_{max}}{2}]$. For $\vec{\varphi}_m$, on each dimension d, $\varphi_{md} = \varphi_{1d} + \varphi_{2d}$. Equation (3) indicates that a particle tends to converge towards a point determined by $\vec{\mathbf{p}}_m$, which is a weighted average of its previous best $\vec{\mathbf{p}}_i$ and the neighbourhood's best $\vec{\mathbf{p}}_g$. $\vec{\mathbf{p}}_m$ can be further generalized to any number of terms:

$$\vec{\mathbf{p}}_m = \frac{\sum_{k \in \mathcal{N}} \vec{\mathbf{R}}[0, \frac{\varphi_{max}}{|\mathcal{N}|}] \otimes \vec{\mathbf{p}}_k}{\sum_{k \in \mathcal{N}} \vec{\varphi}_k}, \quad (4)$$

where $\vec{\mathbf{p}}_k$ denotes the best previous position found by the k-th particle in $\mathcal{N}$, which is a set of neighbours including the current particle itself. Note again that the division is a point-wise vector division operator here. Equation (3) and (4) are essentially the same as the Fully Informed PSO (FIPS) as proposed by Mendes [18]. If we set $k = 2$ and $\vec{\mathbf{p}}_1 = \vec{\mathbf{p}}_i$, and $\vec{\mathbf{p}}_2 = \vec{\mathbf{p}}_g$, with both $\vec{\mathbf{p}}_i, \vec{\mathbf{p}}_g \in \mathcal{N}$, then the Type 1" constriction PSO is just a special case of the more general PSO defined in equation (3).

A significant implication of equation (4) is that it allows us to think more freely about employing terms of influence other than just $\vec{\mathbf{p}}_i$ and $\vec{\mathbf{p}}_g$ (see also remarks by Mendes on FIPS [18] and Kennedy in [13]).

2.1 Memory-swarm vs. Explorer-swarm

We can never underestimate the significance of using "*memory*" in PSO. As remarked by Clerc in [6], a swarm can be viewed as comprising of two sub-swarms according to their differences in functionality. The first group, *explorer-swarm*, is composed of particles moving around in large step sizes and more frequently, each strongly influenced by its velocity and its previous position (see equation (1) and (2)). The *explorer-swarm* is more effective in exploring the search space. The second group, *memory-swarm*, made up of by personal bests of all particles. This *memory-swarm* is more stable than the *explorer-swarm* because personal bests represent positions of only best positions found so far by individual particles. The *memory-swarm* is more effective in retaining better positions found so far by the swarm as a whole.

As shown in Fig. 4 (in section 5), during a typical FER-PSO run, *memory-swarm* comprising of 'pBest' points (ie., personal bests) is successfully used to 'memorize' better positions, while *explorer-swarm* comprising of 'current' points (ie., positions of current particles) is more effective in exploring the search space. Note a 'pBest' is always better than (or equal to) its associated 'current' point.

3. RELATED WORKS

As discussed in the Introduction section, numerous EAs have been developed in the past to handle multimodal optimization problems [12, 17, 10, 1, 11, 2, 21, 15, 5, 16, 20]. A comprehensive review of these niching algorithms is beyond the scope of this paper. One observation is that most of these algorithms require a user to pre-determine some sorts of niching parameters in order to achieve satisfactory performance. Since setting these parameters requires prior knowledge about a specific problem domain, it poses a serious challenge to practitioners. It would be desirable if we can remove the need of pre-specifying niching parameters.

An existing algorithm that inspires us to tackle this issue is FDR-PSO (Fitness-Distance-Ratio based PSO) [22]. FDR-PSO was originally designed as a global optimizer aiming at locating a single global optimum. However, the idea of attracting each particle towards a fitter-and-closer point in its neighbourhood can be used effectively to locate multiple equally good global optima.

We will first describe how FDR-PSO works, and identify problems associated with it with respect to multimodal optimization. We will then demonstrate how these problems can be addressed by proposing a multimodal PSO modifying FDR-PSO in a number of significant ways.

3.1 FDR-PSO

In FDR-PSO [22], a new term called $\vec{\mathbf{p}}_n$ (which is a neighbourhood best based on FDR values) was added to the canonical PSO velocity update equation in order to increase the influence from other fitter and nearer particles:

$$\begin{aligned}\vec{\mathbf{v}}_i \quad \leftarrow \quad & \chi(\vec{\mathbf{v}}_i + \vec{\mathbf{R}}_1[0,\varphi_1] \otimes (\vec{\mathbf{p}}_i - \vec{\mathbf{x}}_i) + \\ & \vec{\mathbf{R}}_2[0,\varphi_2] \otimes (\vec{\mathbf{p}}_g - \vec{\mathbf{x}}_i)) + \vec{\mathbf{R}}_3[0,\varphi_3] \otimes (\vec{\mathbf{p}}_n - \vec{\mathbf{x}}_i)). \quad (5)\end{aligned}$$

Basically the d-th dimension of the i-th particle's velocity is updated using a vector $\vec{\mathbf{p}}_n$, which is determined by maximizing a FDR (Fitness-Distance-Ratio) value for the d-th dimension (assuming maximization):

$$FDR_{(j,i,d)} = \frac{f(\vec{\mathbf{p}}_j) - f(\vec{\mathbf{x}}_i)}{|p_{jd} - x_{id}|}. \quad (6)$$

The d-th dimension of the $\vec{\mathbf{p}}_n$ vector is set to the value of the d-th dimension from the j-th particle's personal best $\vec{\mathbf{p}}_j$, whose $FDR_{(j,i,d)}$ is the largest among all particles. For the i-th particle in the population, Algorithm 1 shows the pseudocode of calculating its $\vec{\mathbf{p}}_n$ vector which collects useful information about good points that are both fit and closer, from all personal bests.

In FDR-PSO, $\{\varphi_1, \varphi_2, \varphi_3\}$ were set to various combinations of values, e.g., {1,1,1}, {1,1,2}, {1,0,2}, {0,1,2}, and {0,0,2}. FDR-PSO has shown competitive performance for locating a single global optimum [22].

4. FER-PSO

As described in the section 2.1, the *memory-swarm* contains all personal bests, therefore it can be seen as a way of retaining better and more stable points found in the search space so far. In contrast, the *explorer-swarm* can be seen as a means of exploring new points. If a particle finds a better point, it will update its corresponding personal best. At the swarm level, this results in updating all personal bests of the *memory-swarm*. With the aim to locate multiple optima, these personal bests retained in the *memory-swarm* can be used as "anchor" points for the swarm to keep track of the multiple better points found so far. In addition, each of these points can be further improved, by moving towards its "fittest-and-closest" neighbours, which can be identified via computing a particle's FDR value (see equation (6)). Over iterations, particles would form niches naturally around multiple optima, given that there are sufficient numbers of particles.

```
input  : A list of all particles in the population
output : Neighbourhood best $\vec{\mathbf{p}}_n$ based on the $i$-th particle's FDR values
for $d = 1$ to Dimensions do
    $FDR \leftarrow 0$, $tmp \leftarrow 0$ ;
    for $j = 1$ to Population Size do
        if ($p_{jd}$ not equal to $x_{id}$) then
            $FDR \leftarrow \frac{f(\vec{\mathbf{p}}_j) - f(\vec{\mathbf{x}}_i)}{|p_{jd} - x_{id}|}$;
            if ($j$ equal to 1) then $tmp \leftarrow FDR$;
            if ($FDR > tmp$) then
                $tmp \leftarrow FDR$ ;
                $p_{nd} \leftarrow p_{jd}$ ;
return $\vec{\mathbf{p}}_n$
```

Algorithm 1: The pseudocode of calculating $\vec{\mathbf{p}}_n$ for the i-th particle ($\vec{\mathbf{x}}_i$) under consideration, according to maximizing its FDR values. To obtain $\vec{\mathbf{p}}_n$ for all particles, this algorithm needs to be iterated over the population.

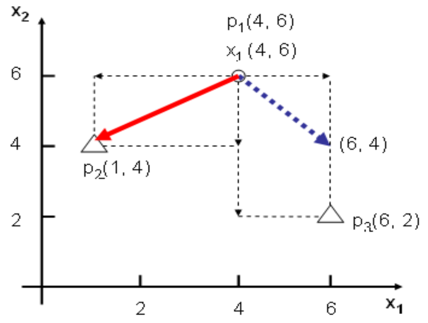


Figure 1: An example of computing $\vec{\mathbf{p}}_n$ vector (for $\vec{\mathbf{x}}_1$) in a 2-dimensional search space. Only 3 particles are in the population. Personal best $\vec{\mathbf{p}}_2$ is equally fit as $\vec{\mathbf{p}}_3$. $f(\vec{\mathbf{p}}_2) - f(\vec{\mathbf{x}}_1) = f(\vec{\mathbf{p}}_3) - f(\vec{\mathbf{x}}_1) = 1$. Maximization is assumed.

FDR calculation in equation (6) however has some serious drawbacks when used for handling multimodal problems. Firstly using a particle's current position often leads to unstable convergence behaviours since it changes more frequently than its corresponding personal best. Secondly, calculating FDR according to the difference between other particles' personal bests and the current position of the particle under consideration does not always lead to a desirable convergent behaviour. Fig. 1 provides a simple example of calculating $\vec{\mathbf{p}}_n$ in FDR-PSO, in a 2-dimensional search space. $\vec{\mathbf{p}}_n$ is obtained from dimensions of 2 known personal bests $\vec{\mathbf{p}}_2$ and $\vec{\mathbf{p}}_3$. In this case, we want to calculate $\vec{\mathbf{p}}_n$ for $\vec{\mathbf{x}}_1$. On dimension 1, we have $FDR_{(2,1,1)} = \frac{1}{|1-4|} = 1/3$, and $FDR_{(3,1,1)} = \frac{1}{|6-4|} = 1/2$. Similarly on dimension 2, we have $FDR_{(2,1,2)} = \frac{1}{|4-6|} = 1/2$, and $FDR_{(3,1,2)} = \frac{1}{|2-6|} = 1/4$. Since we need to maximize FDR values in each dimension, we obtain $\vec{\mathbf{p}}_n(6,4)$ as a result. Noticeable from Fig. 1 is that $\vec{\mathbf{p}}_n$ (6, 4) is not a desirable attraction point. In this example, since both $\vec{\mathbf{p}}_2$ and $\vec{\mathbf{p}}_3$ are equally fit, the computed $\vec{\mathbf{p}}_n$ for $\vec{\mathbf{x}}_1$ is always a different point than $\vec{\mathbf{p}}_2$ and $\vec{\mathbf{p}}_3$. $\vec{\mathbf{x}}_1$ will eventually converge onto $\vec{\mathbf{p}}_n$ (6,4), rather than any of the two promising equally fit points $\vec{\mathbf{p}}_2$ or $\vec{\mathbf{p}}_3$.

To rectify this problem and propose a PSO that is better suited to multimodal optimization, FDR calculation in equation (6) has to be modified in a number of significant ways. Since Euclidean distance is used in the calculation of fitness-and-distance ratio, we name it FER (Fitness-Euclidean distance Ratio):

$$FER_{(j,i)} = \alpha \cdot \frac{f(\vec{\mathbf{p}}_j) - f(\vec{\mathbf{p}}_i)}{||\vec{\mathbf{p}}_j - \vec{\mathbf{p}}_i||}, \tag{7}$$

where $\alpha = \frac{||s||}{f(\vec{\mathbf{p}}_g) - f(\vec{\mathbf{p}}_w)}$ is a scaling factor, to ensure that neither fitness nor Euclidean distance becomes too dominated over one another. $||s||$ is the size of the search space, which can be estimated by its diagonal distance $\sqrt{\sum_{k=1}^{Dim}(x_k^u - x_k^l)^2}$ (where x_k^u and x_k^l are the upper and lower bounds of the k-th dimension of the search space). $\vec{\mathbf{p}}_w$ is the worst-fit particle in the current population. Compared with equation (6), several changes have been made:

- Instead of $\vec{\mathbf{x}}_i$, its corresponding $\vec{\mathbf{p}}_i$ is used. This is because $\vec{\mathbf{p}}_i$ represents an equal or better point, and is much more stable than $\vec{\mathbf{x}}_i$, thereby producing a more reliable FER value;
- Instead of measuring the absolute value of difference from the j-th particle's personal best $\vec{\mathbf{p}}_j$ to $\vec{\mathbf{x}}_i$ in each dimension, Euclidean distance $||\vec{\mathbf{p}}_j - \vec{\mathbf{p}}_i||$ is used. This will eliminate the issue of having undesirable attraction points. As again shown in Fig. 1, $\vec{\mathbf{p}}_n$ based on FER values will get assigned to be $\vec{\mathbf{p}}_2$, therefore $\vec{\mathbf{x}}_1$ will be attracted towards this better point;
- In the proposed FER-PSO, instead of having a single $\vec{\mathbf{p}}_g$ for the entire population, each particle is allocated with its own distinct $\vec{\mathbf{p}}_n$, as determined by its FER value (see Algorithm 2). This will allow each particle to move towards its *fittest-and-closest* neighbouring point. Since each particle must be compared with every other particle in order to calculate its FER, the complexity of calculating FER values for all particles is $O(N^2)$, where N is the population size.

Equation (1) is now rewritten as follows:

$$\vec{\mathbf{v}}_i \leftarrow \chi(\vec{\mathbf{v}}_i + \vec{\mathbf{R}}_1[0, \frac{\varphi_{max}}{2}] \otimes (\vec{\mathbf{p}}_i - \vec{\mathbf{x}}_i) + \vec{\mathbf{R}}_2[0, \frac{\varphi_{max}}{2}] \otimes (\vec{\mathbf{p}}_n - \vec{\mathbf{x}}_i)). \tag{8}$$

Note that we do not need to pre-specify how far apart optima are in equation (7) and (8). The calculation of α would require some knowledge of x_k^u and x_k^l, but this information is readily available in most situations. Equation (8) can be seen as a special case of the more generalized FIPS (see equation (4)) where each particle's neighbourhood best is chosen based on its computed FER value.

5. NUMERICAL RESULTS

Table 1 provides a set of test functions to assess FER-PSO's performance in locating multiple known global optima. All test functions have more than one global optima,

```
input  : A list of all particles in the population
output : Neighbourhood best $\vec{\mathbf{p}}_n$ based on the $i$-th particle's FER value
$FER \leftarrow 0$, $tmp \leftarrow 0$, $euDist \leftarrow 0$ ;
for $j = 1$ to Population Size do
    Calculate the Euclidean distance $euDist$ from $\vec{\mathbf{p}}_i$
    to the $j$-th particle's personal best $\vec{\mathbf{p}}_j$;
    if (euDist not equal to 0) then
        Calculate $FER$ according to equation (7) ;
        if ($j$ equal to 1) then $tmp \leftarrow FER$;
        if ($FER > tmp$) then
            $tmp \leftarrow FER$ ;
            $\vec{\mathbf{p}}_n \leftarrow \vec{\mathbf{p}}_j$ ;
return $\vec{\mathbf{p}}_n$
```

Algorithm 2: The pseudocode of calculating $\vec{\mathbf{p}}_n$ for the i-th particle under consideration, according to its FER value. To obtain $\vec{\mathbf{p}}_n$ for all particles, this algorithm needs to be iterated over the population.

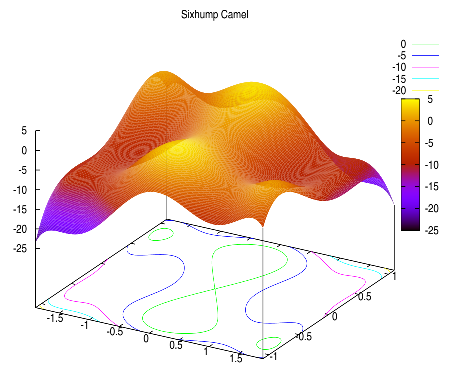


Figure 2: f_3 Six-Hump Camel Back function.

hence the objective is to locate as many as possible these global optima. f_1, f_2, f_3 and f_4 are relatively simple multimodal test functions. However, f_5 Shubert 2D is a much more challenging function because it has a large number of global and local optima (760 optima including 18 global optima). Especially, the 18 global peaks form 9 pairs, with 2 very close global peaks in each pair. This function would pose a serious challenge to any niching algorithm whose performance relies on its setting of the niching parameters. Fig. 2 and 3 show the search landscape of f_3 and f_5 respectively.

Performance measurement is carried out by using an algorithm for identifying species seeds in SPSO [20]. For each test function, the species radius is set to a value not greater than the distance between 2 closest global peaks (so that particles on two found peaks can be treated as from different species). The species seeds identification algorithm will be able to produce a list of best as well as different personal bests based on the preset species radius and a given list of all personal bests from the entire swarm. Since the exact number of global optima is known *a priori*, and also roughly how far apart between 2 closest global optima, a multimodal optimization algorithm's performance can be measured in

Table 1: Multimodal test functions.

Function	Range	Comments
Deb's 1st function [8]: $f_1(x) = sin^6(5\pi x)$	$0 \leq x \leq 1$	5 equally spaced global optima
Himmelblau [1]: $f_2(x, y) = 200 - (x^2 + y - 11)^2 - (x + y^2 - 7)^2$	$-6 \leq x, y \leq 6$	4 global optima
Six-Hump Camel Back [19]: $f_3(x, y) = -4[(4 - 2.1x^2 + \frac{x^4}{3})x^2 + xy + (-4 + 4y^2)y^2]$	$-1.9 \leq x \leq 1.9$; $-1.1 \leq y \leq 1.1$	2 global optima and 4 local optima
Branin RCOS [19]: $f_4(x, y) = (y - \frac{5.1}{4\pi^2}x^2 + \frac{5}{\pi}x - 6)^2 + 10(1 - \frac{1}{8\pi})cos(x) + 10$	$-5 \leq x \leq -10$; $0 \leq y \leq 15$	3 global optima
Shubert 2D [15]: $f_5(x, y) = \sum_{i=1}^{5} icos[(i + 1)x + i] \sum_{i=1}^{5} icos[(i + 1)y + i]$	$-10 \leq x, y \leq 10$	760 optima including 18 global optima

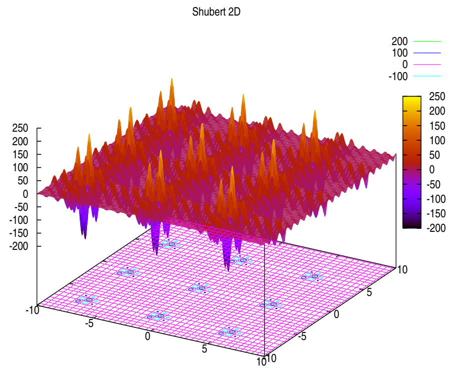


Figure 3: f_5 Shubert 2D function.

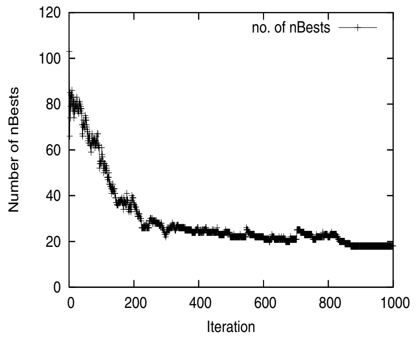


Figure 5: The number of $\vec{\mathbf{p}}_n$ decreases over a simulation run, on the f_5 Shubert 2D function.

terms of the *number of evaluations required* to achieve a pre-specified accuracy for a run. In this case, we can check species seeds, which are the dominant particles sufficiently different from each other. We can determine if a global optimum is found by checking each species seed to see if it is close enough to a known different global optimum. An expected accuracy acceptance threshold (typically $0 < \epsilon \leq 1$) is defined to determine if a solution is close enough to a global optimum. Note that this species seeds identification algorithm is only used for performance measurement in determining if a sufficient number of global peaks has been found, but not in any part of the FER-PSO optimization procedure.

The performance can be also measured in terms of *success rate*, which is the percentage of runs in which all global optima are successfully located.

To study the effect of population sizes on FER-PSO's performance, population sizes ranging from 50 up to 200 are used for f_1 to f_4. For f_5, due to a larger number of global optima, population sizes ranging from 200 to 500 are used.

5.1 Neighbourhood Best $\vec{\mathbf{p}}_n$

Fig. 4 shows snapshots of 4 iteration steps of a typical FER-PSO run on f_3. Note that a 'pBest' (ie., personal best) is always better than (or equal to) its corresponding 'current' position (they are connected via a line in Fig. 4). A few 'nBest' (ie., $\vec{\mathbf{p}}_n$) positions are identified from among all current and past 'pBest' positions. Within 50 iterations, 2 global optima are located by FER-PSO. Note that even with the presence of 2 local peaks on the top-left and bottom-right corners, the 2 global peaks would eventually dominate over the 2 local peaks as better points for $\vec{\mathbf{p}}_n$ for the population.

The total number of $\vec{\mathbf{p}}_n$ generally decreases to a number close to the actually number of global peaks on the search landscape, as these points are most likely to be the best candidates for $\vec{\mathbf{p}}_n$. Fig. 5 shows that on f_5 where 18 global peaks are present, the number of different $\vec{\mathbf{p}}_n$ identified from the population decreases over iterations, until it reaches to a number close to 18.

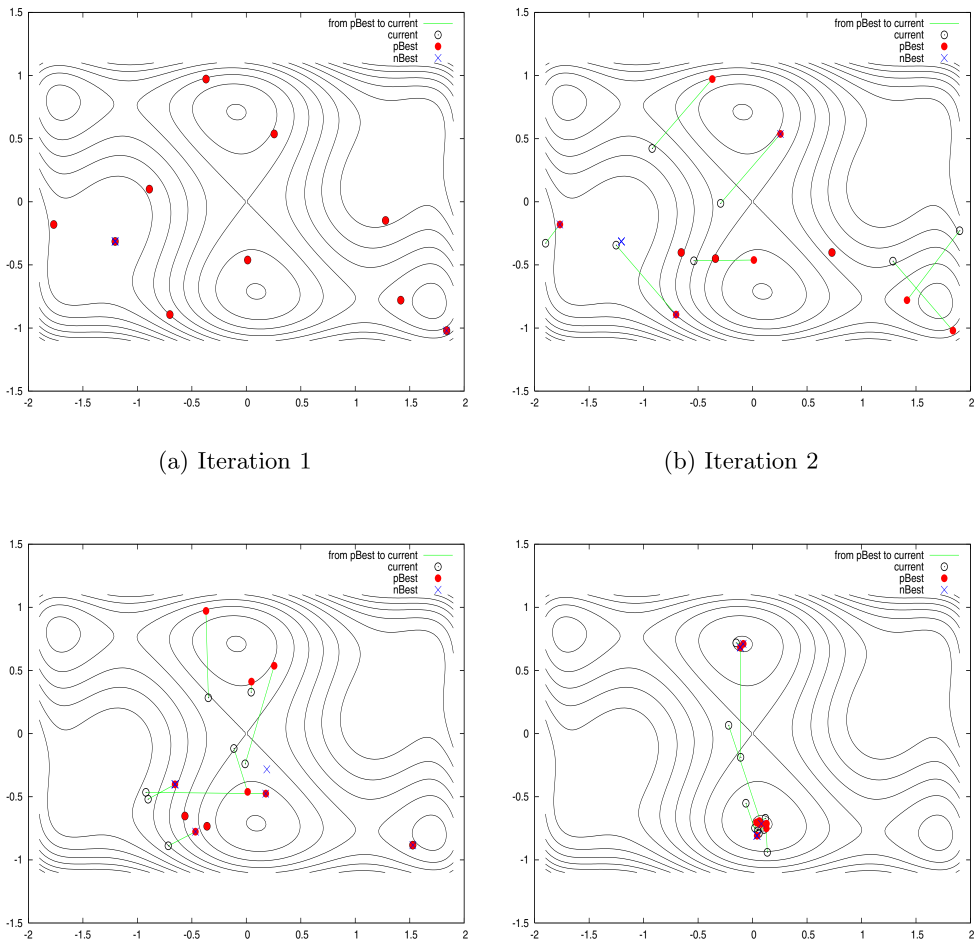
5.2 Acceptance Threshold

Fig. 6 shows that for most test functions the success rates can vary with respect to different expected accuracy acceptance thresholds. As expected, the smaller the threshold value is set to, the lower the success rates will be.

5.3 Effect of Population Size

Fig. 7 shows the success rates increase when larger population sizes are used for f_1 to f_4. The only exception is Himmelblau function, where the success rate is decreased to just below 0.9 when a population size of 200 is used. A closer look at the results reveals that this is largely because the smaller acceptance threshold used (0.0001). FER-PSO in fact almost always find all 4 global peaks. If this threshold value were set higher, then a better success rate can be obtained.

For the more challenging f_5, since there are in total 9 clusters, with each cluster containing 2 very close global peaks (Fig. 3), instead of using *success rate*, we measure the average number of global peaks found over 50 runs, as shown



(a) Iteration 1 (b) Iteration 2

(c) Iteration 5 (d) Iteration 50

Figure 4: Snapshots of the entire population at 4 different iteration steps, on the fitness landscape of f_3 the Six Hump-Camel Back.

in Fig. 8. This allows us to see more accurately how well FER-PSO performs in terms of locating the percentage of all 18 global peaks.

Fig. 8 shows that FER-PSO becomes increasingly capable of finding more global optima as a larger population size is used. In Fig. 9, with a population size of 20, 50, 100, and 200, the number of global peaks found is 5, 13, 17 and 18 respectively. With just a population size of 200 or more, and without the need of using any niching parameter, FER-PSO can reliably find all 18 global peaks. Although most of the time, global peaks are captured by $\vec{\mathbf{p}}_n$, this is not always the case. Sometimes a global optimum was captured by a personal best, instead of $\vec{\mathbf{p}}_n$ (see Fig. 9 c) and d)). However, since all $\vec{\mathbf{p}}_n$ must be either current or past personal bests in the population, we can simply check if existing personal bests have located all 18 global peaks.

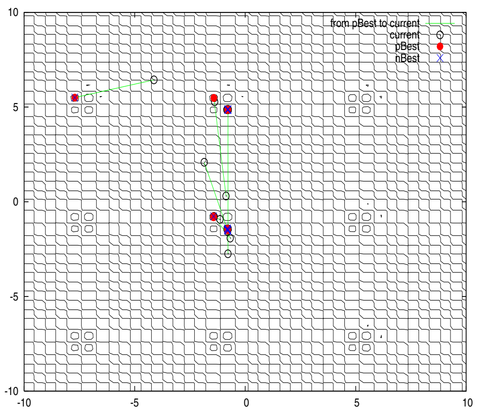
5.4 Comparison

f_5 Shubert 2D function was also used in SCGA [15], where a niching parameter, i.e., species distance σ_s specifying the distance between two closest global optima, was assumed known a priori. SCGA with its σ_s set to 1.6 and a population size of 1000 was shown to be successful in locating all 18 global peaks. In comparison, FER-PSO did not require pre-specifying any niching parameter. Furthermore, FER-PSO was able to locate all 18 global peaks with a smaller population size of 200, rather than 1000.

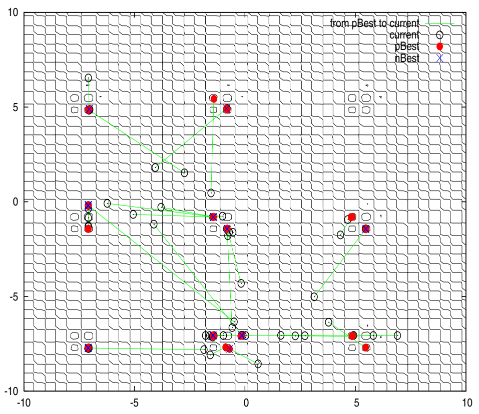
Two more recent speciation-based PSO variants, ANPSO and ESPSO, which were specifically designed to reduce the sensitivity of its species radius parameter (though not removing it) also suggested a minimal population size of 200 in order to locate all 18 global peasks for f_5 Shubert 2D function [3, 4]. FER-PSO again compares well with these results.

6. CONCLUSIONS

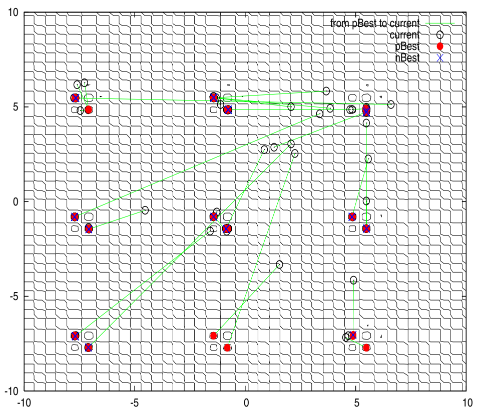
This paper has proposed a multimodal PSO, FER-PSO, which makes use of the concepts of *memory-swarm* and *explorer-swarm*. To encourage particles to locate multiple global optima, FER (Fitness and Euclidean distance Ratio) is calculated based on the fitness difference and the



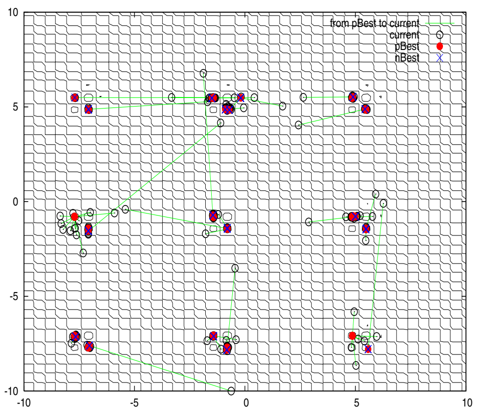
(a) popsize = 20



(b) popsize = 50



(c) popsize = 100



(d) popsize = 200

Figure 9: The number of global optima found by FER-PSO increases as a larger population size is used, ranging from 20 to 200. All snapshots are after 500 iterations.

Euclidean distance between a particle's personal best and other personal bests of the particles in the population. The key advantage is that FER-PSO removes the need of pre-specifying niching parameters that are commonly required in existing niching EAs for multimodal optimization. Our results show that FER-PSO provides comparable performance to existing EAs on some widely-used simple multimodal test functions. And more importantly, without using any niching parameter, FER-PSO can give superior performance on a more challenging multimodal function f_5, which proves to be difficult to many existing niching EAs.

Future works will be carried out on evaluating FER-PSO's performance on multimodal functions with higher dimensions. Another aim is to further improve the efficiency of FER-PSO. Since population size plays an important role in FER-PSO, a self-adaptive population sizing strategy would be desirable. More specifically strategies that can adjust FER-PSO's population size according to different problems will be studied.

7. ACKNOWLEDGMENTS

The author would like to thank Stefan Bird for interesting discussions which further contributed to the improvement of FER-PSO.

8. REFERENCES

[1] D. Beasley, D. R. Bull, and R. R. Martin. A sequential niche technique for multimodal function optimization. *Evolutionary Computation*, 1(2):101–125, 1993.

[2] M. Bessaou, A. Pétrowski, and P. Siarry. Island model cooperating with speciation for multimodal optimization. In *Parallel Problem Solving from Nature - PPSN VI.* Springer Verlag, 16-20.

[3] S. Bird and X. Li. Adaptively choosing niching parameters in a PSO. In M. Cattolico, editor, *Genetic and Evolutionary Computation Conference, GECCO 2006, Proceedings, Seattle, Washington, USA, July 8-12, 2006*, pages 3–10. ACM, 2006.

[4] S. Bird and X. Li. Enhancing the robustness of a speciation-based PSO. In e. a. Gary G. Yen, editor,

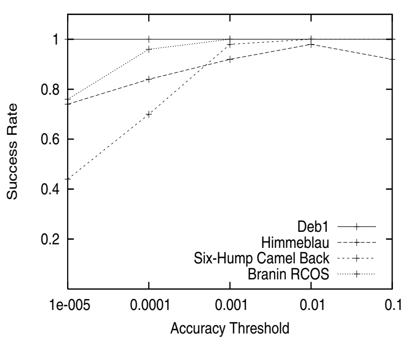


Figure 6: Success rates vary depending on the chosen accuracy acceptance threshold.

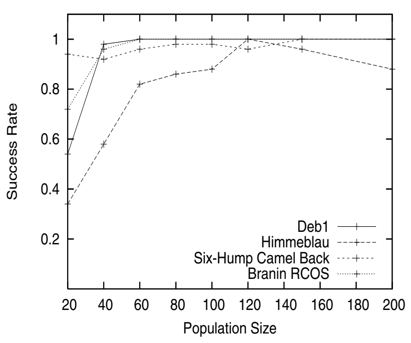


Figure 7: Success rates in terms of different population sizes for f_1 to f_4 (averaged over 50 runs). The acceptance threshold is 0.0001. FER-PSO is run for a maximum of 200000 iterations.

Proceedings of the 2006 IEEE Congress on Evolutionary Computation, pages 843–850, Vancouver, BC, Canada, 16-21 July 2006. IEEE Press.

[5] R. Brits, A. Engelbrecht, and F. van den Bergh. A niching particle swarm optimizer. In *Proceedings of the 4th Asia-Pacific Conference on Simulated Evolution and Learning 2002(SEAL 2002)*, pages 692–696, 2002.

[6] M. Clerc. *Particle Swarm Optimization.* ISTE Ltd, London, UK, 2006.

[7] M. Clerc and J. Kennedy. The particle swarm - explosion, stability, and convergence in a multidimensional complex space. *IEEE Trans. Evol. Comput.*, 6:58–73, Feb. 2002.

[8] K. Deb and D. Goldberg. An investigation of niche and species formation in genetic function optimization. In J. Schaffer, editor, *Proceedings of the Third International Conference on Genetic Algorithms*, pages 42–50, 1989.

[9] R. Eberhart and Y. Shi. Comparing inertia weights and constriction factors in particle swarm optimization. In *Proc. of IEEE Int. Conf. Evolutionary Computation*, pages 84–88, 2000.

[10] D. E. Goldberg and J. Richardson. Genetic algorithms with sharing for multimodal function optimization. In J. Grefenstette, editor, *Proceedings of the Second International Conference on Genetic Algorithms*, pages 41–49, 1987.

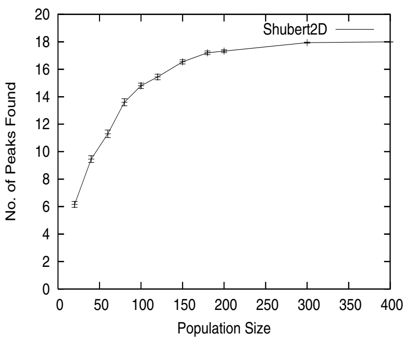


Figure 8: Number of peaks found (mean and one standard error over 50 independent runs) with increasing population sizes.

[11] G. R. Harik. Finding multimodal solutions using restricted tournament selection. In *Proceedings of the Sixth International Conference on Genetic Algorithms.* Morgan Kaufmann.

[12] K. A. D. Jong. *An analysis of the behavior of a class of genetic adaptive systems.* PhD thesis, University of Michigan, 1975.

[13] J. Kennedy. In search of the essential particle swarm. In *Proc. of 2006 IEEE Congress on Evolutionary Computation*, pages 6158–6165, 2006.

[14] J. Kennedy and R. Eberhart. *Swarm Intelligence.* Morgan Kaufmann, 2001.

[15] J.-P. Li, M. E. Balazs, G. T. Parks, and P. J. Clarkson. A species conserving genetic algorithm for multimodal function optimization. *Evol. Comput.*, 10(3):207–234, 2002.

[16] X. Li. Adaptively choosing neighbourhood bests using species in a particle swarm optimizer for multimodal function optimization. In K. Deb, editor, *Proc. of Genetic and Evolutionary Computation Conference 2004(LNCS 3102)*, pages 105–116, 2004.

[17] S. W. Mahfoud. Crowding and preselection revisited. In R. Männer and B. Manderick, editors, *Parallel problem solving from nature 2*, pages 27–36, Amsterdam, 1992. North-Holland.

[18] R. Mendes, J. Kennedy, and J. Neves. The fully informed particle swarm: simpler, maybe better. *IEEE Trans. Evol. Comput.*, 8:204–210, Jun. 2004.

[19] Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs.* Springer-Verlag, New York, New York, 1996.

[20] D. Parrott and X. Li. Locating and tracking multiple dynamic optima by a particle swarm model using speciation. *IEEE Transactions on Evolutionary Computation*, 10(4):440–458, August 2006.

[21] A. Pétrowski. A clearing procedure as a niching method for genetic algorithms. In *Proceedings of the 3rd IEEE International Conference on Evolutionary Computation*, pages 798–803, 1996.

[22] K. Veeramachaneni, T. Peram, C. Mohan, and L. Osadciw. Optimization using particle swarm with near neighbor interactions. In *Proc. of Genetic and Evolutionary Computation Conference*, pages 110 – 121, Chicago, Illinois, 2003.