# Particle Swarm with Speciation and Adaptation in a Dynamic Environment

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

Jürgen Branke
Institute AIFB
University of Karlsruhe
76128 Karlsruhe, Germany
branke@aifb.uni-karlsruhe.de

Tim Blackwell
Department of Computing
Goldsmiths College
University of London
London, United Kingdom
t.blackwell@gold.ac.uk

## ABSTRACT

This paper describes an extension to a speciation-based particle swarm optimizer (SPSO) to improve performance in dynamic environments. The improved SPSO has adopted several proven useful techniques. In particular, SPSO is shown to be able to adapt to a series of dynamic test cases with varying number of peaks (assuming maximization). Inspired by the concept of quantum swarms, this paper also proposes a particle diversification method that promotes particle diversity within each converged species. Our results over the moving peaks benchmark test functions suggest that SPSO incorporating this particle diversification method can greatly improve its adaptability hence optima tracking performance.

## 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, Optimization in Dynamic Environments

## 1. INTRODUCTION

Many real-world optimization problems are dynamic and require optimization algorithms capable of adapting to the changing optima over time. For example, traffic conditions in a city change dynamically and continuously. What might be regarded as an optimal route at one time might not be optimal in the next minute. In contrast to optimization towards a static optimum, in a dynamic environment the goal is to track as closely as possible the dynamically changing optima.

Evolutionary Algorithms (EAs) are natural candidates for such problem types, as natural evolution occurs in a continuously changing environment. So it is not surprising that EA variants have been shown to offer a robust and effective optimization approach for solving dynamic optimization problems [7, 16]. EA's appeal over other traditional optimization methods comes from its use of a population of candidate solutions, which enables an EA to maintain useful information about characteristics of the environment. Particle Swarm Optimization in many ways similar to EAs, though it is inspired by the social behaviours of insects and animals. PSO has proven to be an effective optimization technique not only for locating optima in a static environment, but also for tracking time-varying optima in a dynamic environment [19, 20, 4].

This paper describes an extension to a speciation-based PSO (SPSO) proposed in [19], which is largely based on the idea of a speciation algorithm inspired by the clearing procedure first proposed by Petrowski in [21] and the Species Conserving GA (SCGA) by Li et al. in [18]. This speciation algorithm was later incorporated into a PSO algorithm for optimization in both static and dynamic environments [19, 20]. This revised SPSO adopts a number of useful techniques to further improve its tracking ability in a dynamic environment, including some techniques suggested in the multi-swarm approach by Blackwell and Branke [4].

The paper is organized as follows. Section 2 provides the background information on PSO, related research, and the motivation of this research. Section 3 describes the improved SPSO model, including the adopted techniques for tracking optima in a multimodal dynamic environment. Section 4 describes the experiments conducted, followed by section 5 on experimental setups and section 6 on results. Finally section 7 gives some general conclusions and directions for future research.

## 2. BACKGROUND

### 2.1 Particle Swarms

Particle Swarm Optimization (PSO) is a population-based optimization methods simulating the social behaviour of insects and animals [17]. PSO has been successfully applied to dynamic optimization problems [16]. PSO differs from EAs in the way it manipulates each individual in the population. Instead of using evolutionary operators such as crossover


*GECCO'06*, July 8–12, 2006, Seattle, Washington, USA.

and mutation, PSO modifies each individual's position in the search space, $\vec{x_i} = (x_{i1}, x_{i2}, \ldots, x_{iN})$, based on its velocity (ie., the rate of position change), $\vec{v_i} = (v_{i1}, v_{i2}, \ldots, v_{iN})$ and some best found solutions in the past. In particular, the velocity $\vec{v_i}$ is modified at each iteration step by the $i$-th particle's previous best position, ie., the position giving the best fitness value so far, $\vec{p_i} = (p_{i1}, p_{i2}, \ldots, p_{iN})$, and the population's best particle's position $\vec{p_g} = (p_{g1}, p_{g2}, \ldots, p_{gN})$. At each iteration, the velocity and position of each particle in the swarm are updated according to the following two equations [12]:

$$\vec{v_i} = \chi * (\vec{v_i} + c_1 * rand() * (\vec{p_i} - \vec{x_i}) + c_2 * rand() * (\vec{p_{gi}} - \vec{x_i})) \quad (1)$$

$$\vec{x_i} = \vec{x_i} + \vec{v_i} \quad (2)$$

where $c_1$ and $c_2$ are two positive constants. $rand()$ is a random function returning values in the range [0, 1]. $\chi$ is called the constriction coefficient and it is computed according to:

$$\chi = \frac{2}{\left|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}\right|} \quad (3)$$

where $\varphi = c_1 + c_2, \varphi > 4$. $\chi$ is used to prevent the particles from exploring too far away into the search space since $\chi$ always applies a damping effect to the oscillation size of each particle over time [17, 12].

Two common approaches of choosing $\vec{p_g}$ are known as *gbest* and *lbest* methods [17]. In a *gbest* PSO, the position of each particle in the search space is influenced by the best-fit particle in the entire population, whereas a *lbest* PSO only allows each particle to be influenced by the best-fit particle chosen from its neighborhood. The *lbest* PSO with a neighborhood size set to the population size is equivalent to a *gbest* PSO.

The SPSO described in this paper adopts a *lbest* approach. It uses a speciation algorithm to identify multiple species and their $\vec{p_g}$, treating them as separate PSOs. The SPSO is capable of optimizing simultaneously multiple static or dynamically changing optima over time [19, 20].

## 2.2 Related research

Two major issues must be resolved when dealing with dynamic problems: detecting that a change in the environment has actually occurred, and responding appropriately to the change so that the optima can still be tracked.

One of the early works on using PSO for dynamic optimization was by Eberhart and Shi in [13], where they used an inertia weight version of PSO to track the optimum of a 3-dimensional unimodal parabolic function which changes its maxima every 100 iterations. It was found under certain circumstances that the PSO's performance was comparable or better than that of previously published evolutionary algorithms [1, 2].

For detection, Carlisle and Dozier used a *sentry* particle which is randomly chosen at each iteration [10]. The sentry particle gets evaluated before each iteration and compares its fitness with its previous fitness value. If the two values are different, indicating the environment has changed, ie., the optimum has moved, then the whole population gets alerted and several possible responses can then be triggered. Hu and Eberhart proposed to re-evaluate $\vec{p_g}$ and a second best particle to detect if a change has occurred.

Various response strategies have also been proposed. Carlisle and Dozier proposed to re-evaluate all $\vec{p_i}$ of all particles when a change has been detected [11]. If the re-evaluated $\vec{p_i}$ is less fit than its current position $\vec{x_i}$, then this $\vec{p_i}$ is replaced by its associated $\vec{x_i}$, because only these two evaluations can be considered up-to-date. Hu and Eberhart studied the effects of re-randomizing various proportions of the swarm to maintain some degree of diversity in order to better track the optima after a change [14]. However, this approach suffers from possible information loss since the re-randomized portion of the population does not retain any information that might be useful from the past iterations.

In order to maintain better particle diversity throughout a run, Blackwell and Bentley introduced charged swarms where mutually repelling *charged* particles orbit a nucleus of neutral particles (conventional PSO particles) [3]. Whereas the charged particles allow the swarm to better adapt to changes in the environment, the neutral particles play the role of continuing to converge towards the optimum in finer details.

Inspired by multi-population EA approaches such as the multinational GA [22] and self-organizing scouts [7], Blackwell and Branke proposed an interacting multi-swarm PSO as a further improvement to the charged swarms [4]. The multi-swarm PSO aims at maintaining multiple swarm populations on different peaks. Multiple swarms are prevented from converging to the same optimum by randomizing the worse of two swarms that come too close. The multi-swarm PSO also replaces the charged particles by quantum particles whose position is solely based on a probability function centered around the swarm attractor. The resulting multi-quantum swarms outperform charged and standard PSO on the moving peaks function. This multi-swarm approach is particularly attractive because of its improved adaptability in a more complex multimodal dynamic environment where multiple peaks exist and need to be tracked.

With a similar aim to locate and track multiple peaks in a dynamic environment, Parrott and Li in [19, 20] proposed a species-based PSO (SPSO) incorporating a speciation algorithm first proposed by Petrowski [21]. The SPSO uses a local "species seed" which provides the local $\vec{p_g}$ to particles whose positions in the search space are within a user-specified radius of the seed. This encourages swarms to converge onto multiple local optima instead of a single global optimum, hence developing multiple sub-populations in parallel. In addition, the dynamic SPSO uses a parameter $p_{max}$ to limit the number of particles allowed in a species (or swarm), with the excess particles reinitialized at random positions in the total search space.

In another work, Janson and Middendorf proposed a PSO using a dynamic and hierarchical neighbourhood structure [15] to handle dynamic optimization problems. They demonstrated that such a structure is useful for maintaining some particle diversity in a dynamic environment.

## 2.3 Research motivation

If we can assume that changes are only slight in a dynamic environment, in other words, the new environment (after a change) is closely related to the old environment, it would be beneficial to use knowledge about the previous search space to help search for the new optimum. In [8] the benefit of keeping different aspects of the fitness landscape has been investigated, among others showing that it can be

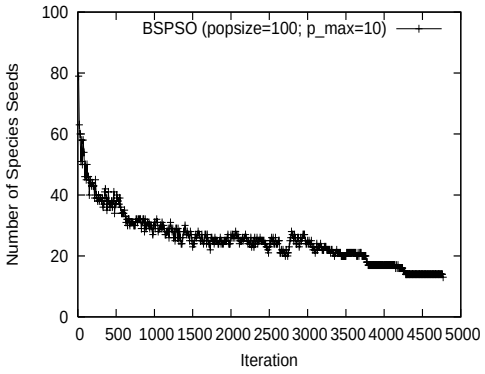


**Figure 1: Decreasing numbers of species seeds during a SPSO simulation run on a problem with 10 peaks.**

quite beneficial to maintain knowledge about previous local optima.

With the above in mind, this paper aims to improve SPSO's adaptability in a multimodal dynamic environment so that the SPSO's tracking ability can be further enhanced. Since SPSO uses a speciation algorithm which finds the number of species and likely the number of existing optima towards the end of a model run, SPSO is able to identify the peaks and converge onto these peaks in parallel and adaptively. Fig. 1 shows a typical SPSO simulation run [1], where SPSO starts with a large number of species seeds, but over iterations this number decreases to a value close to the number of existing peaks (if they are found). It is shown below that SPSO is able to adapt to a series of dynamic test cases with varying number of peaks. This contrasts to the existing PSO models (such as the multi-swarm model) that often assume a user-specified variable for the number of peaks that need to be tracked. In the case when there is no prior knowledge of the number of peaks (which is often the case for real-world problems), using SPSO may be advantageous.

Some proven useful techniques, in particular the concept of quantum swarms introduced by the multi-swarm approach [4], can be adopted in SPSO to promote particle diversity and enable it to better handle the trade-off between convergence and diversity, which is important for effective optima tracking in a dynamic environment.

## 3. IMPROVED SPSO

The speciation-based PSO (SPSO) model was developed based on the notion of species [19]. The definition of species depends on a parameter $r_s$, which denotes the radius measured in Euclidean distance from the center of a species to its boundary. The center of a species, the so-called species seed, is always the best-fit individual in the species. All particles that fall within the $r_s$ distance from the species seed are classified as the same species.

### 3.1 Species seeds as neighborhood bests

The algorithm for determining species seeds, introduced by Petrowski in [21] and also Li et al. in [18], is adopted here. By applying this algorithm at each iteration step, different species seeds can be identified for multiple species and these seeds' $\vec{p_i}$ can be used as the $\vec{p_g}$ for different species accordingly. Algorithm 1 summarizes the steps for determining species seeds.

[1]on a moving peak scenario2 instance with 10 peaks; see section 5.1

**input** : $L_{sorted}$ - a list of all particles sorted in their decreasing $f(\vec{x_i})$ values
**output**: $S$ - a list of all dominating particles identified as species seeds

```
begin
    S = ∅;
    while not reaching the end of L_sorted do
        Get best unprocessed p ∈ L_sorted;
        found ← FALSE;
        for all s ∈ S do
            if d(s, p) ≤ r_s then
                found ← TRUE;
                break;
        if not found then
            let S ← S ∪ {p};
end
```

**Algorithm 1:** The algorithm for determining species seeds according to all $f(\vec{x_i})$ values.

Algorithm 1 is performed at each iteration step. The algorithm takes as an input $L_{sorted}$, a list containing all particles sorted in decreasing order of their $\vec{x_i}$ fitness. The species seed set $S$ is initially set to $\emptyset$. All particles' $\vec{x_i}$ are checked in turn (from best to least-fit) against the species seeds found so far. If a particle does not fall within the radius $r_s$ of all the seeds of $S$, then this particle will become a new seed and be added to $S$. Fig. 2 provides an example to illustrate the working of this algorithm. In this case, applying the algorithm will identify $s_1$, $s_2$ and $s_3$ as the species seeds. Note also that if seeds have their radii overlapped (e.g., $s_2$ and $s_3$ here), the first identified seed (such as $s_2$) will dominate over those seeds identified later from the list $L_{sorted}$. For example, $s_2$ dominates $s_3$ therefore $p$ should belong to the species led by $s_2$.

Since a species seed is the best-fit particle's $\vec{x_i}$ within a species, other particles within the same species can be made to follow the species seed's $\vec{p_i}$ as the newly identified neighborhood best (ie., $\vec{p_g}$ of a *lbest* PSO). This allows particles within the same species to be attracted to positions that make them even fitter. Because species are formed around different optima in parallel, making species seeds the new neighborhood bests provides the right guidance for particles in different species to locate multiple optima.

Since species seeds in $S$ are sorted in the order of decreasing fitness, the more highly fit seeds also have a potentially larger influence than the less fit seeds. This also helps the algorithm to locate the global optima before local ones.

Once the species seeds have been identified from the population, we can then allocate each seed's $\vec{p_i}$ to be the $\vec{p_g}$ to all the particles in the same species at each iteration step. The speciation-based PSO (SPSO) accommodating the algorithm for determining species seeds described above can be summarized in Algorithm 2.

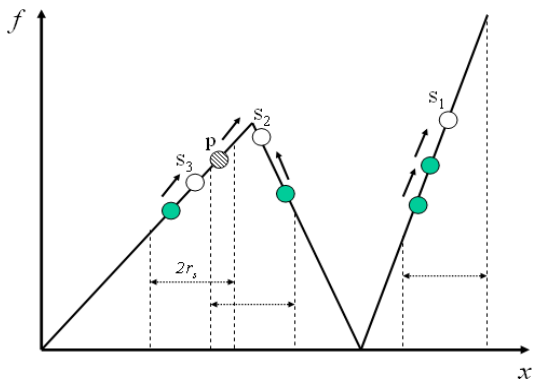


**Figure 2: An example of how to determine the species seeds from a population of particles. $s_1$, $s_2$ and $s_3$ are chosen as the species seeds. Note that $p$ follows $s_2$.**

## 3.2 Adopted techniques

This section describes several useful techniques incorporated into the SPSO to enhance its adaptability in a multimodal dynamic environment. Parrott and Li proposed to use a variable, $p_{max}$, to set a maximum number of particles allowed for a species [20]. Blackwell and Branke introduced the concepts of quantum particles and anti-convergence in their multi-swarm approach [5]. In addition to adopting these known techniques in SPSO, we introduce a new particle diversification method which can be used to promote particle diversity within a species according to some local convergence measurement criteria within the species. This method can help improve SPSO's tracking performance in a dynamic environment.

### 3.2.1 Limiting the number of particles in a species

The Algorithm 1 favours the species seeds with higher fitness values, and as a result, more particles are allocated to species which have a fitter species seed than lower ones. On a multimodal fitness landscape, this causes too many particles to be assigned to just a few very best peaks, while leaving other lower peaks unoccupied. In order to distribute more evenly the number of particles across different species, a parameter $p_{max}$ is used to set a maximum number of particles that a species is allowed [20]. When the Algorithm 1 is called, only the best $p_{max}$ particles will be allocated as members of a species. Lower fitness members that causes the species population to exceed $p_{max}$ are reinitialized as neutral (see below) particles at random positions in the search space, as a side effect increasing the chance of finding other peaks.

### 3.2.2 Quantum swarm

In a dynamic environment, convergence to peaks implies diversity loss. To maintain diversity at a reasonable level is a critical issue. By maintaining a better diversity, the optimization algorithm is better equipped to detect if there are changes occurred and track more effectively the moved peaks.

In [4], Blackwell and Branke proposed a quantum swarm model inspired by quantum physics. In this quantum swarm model, a swarm is made up of "neutral" and "quantum" particles (see Fig. 3). Neutral particles follow the conventional PSO movement (see equation (1) and (2)) providing the convergence that is necessary in finding the peaks, whereas the quantum particles are positioned as a "cloud" centered around the $\vec{p_g}$, providing a constant level of particle diversity within a species. In particular, they are positioned within a hypersphere of radius $r_{cloud}$ (which is user-specified) centered on $\vec{p_g}$ according to a uniform shell distribution, $p(r, dr) = \rho(r)dr = const$, where $\rho$ is a probability density, $r$ a shell radius, $dr$ a volume element and $p$ a probability. When a peak moves within the distance from the $\vec{p_g}$, the new peak position is likely to be covered by the "cloud", therefore the algorithm will be more likely to find the changed peak position than otherwise. Both neutral and quantum particles share the same information network through $\vec{p_g}$ over the combined population of all neutral and quantum particles.[2]

```
//initialization;
for i=1 to popSize do
    randomly initialize i-th particle: v_i, x_i;
    p_i ← x_i
repeat
    for i=1 to popSize do
        evaluate f(x_i);
        if f(x_i) > f(p_i) then
            p_i ← x_i
    if change is detected then
        p_i ← x_i
    1) sort all particles according to their fitness values
    (from the best-fit to the least-fit);
    2) call the speciation procedure in Algorithm 1 to
    identify species seeds;
    3) assign each identified species seed's p_i as the p_g
    to all individuals identified in the same species;
    4) adjust neutral particle positions according to
    equation (1) and (2); or quantum particles
    following the quantum particle update rule;
    5) check each species to see if the numParticles >
    p_max; If so, replace the excess particles with
    random particles into the search space;
    //model variant;
    model variant called.
until numEvals > maxNumEvals;
```

**Algorithm 2:** The species based PSO algorithm.

### 3.2.3 Replacing the worst species

Blackwell and Branke [5] proposed to use an *anti-convergence* method to re-randomize the worst swarm (or species) into the search space when all species have converged. These randomized particles help explore different parts of the search space, thereby increasing the likelihood of locating more peaks.

A similar method is also adopted in SPSO. Let us define the particle diversity for a species, $d_p$, as the distance of the species seed to the furthermost particle in the same species (see Fig. 4). The average particle diversity is the average of all $d_p$ values across all species, which becomes 0 only if all

[2]In SPSO, $\vec{p_g}$ is a species seed identified by the speciation procedure in Algorithm 1.

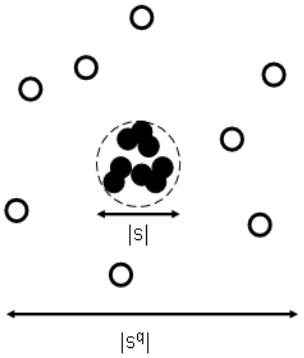


**Figure 3: An example of a swarm containing both the neutral and quantum particles. A solid circle denotes a neutral particle, whereas a hollow circle denotes a quantum particle. $|s|$ denotes the swarm size**

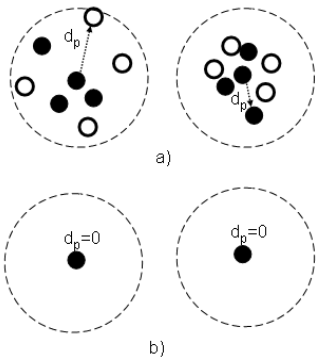


**Figure 4: Measuring particle diversity. a) shows two species with various *seed-to-furthest particle* distance values; b) shows an extreme case where the species diversity for the two species are 0.0.**

species have converged completely (eg., Fig. 4b).

If this average particle diversity is below a threshold $\Delta$, the anti-convergence method is invoked and the worst species is replaced by randomized neutral particles.

#### 3.2.4 Particle diversification within a species

Maintaining some local particle diversity around a found peak is critical for optima tracking. To see if a species has converged onto a peak, we can check if the particle diversity, $d_p$, of a species is smaller than a threshold $\delta$. If so, then the species can be considered as having converged (Fig. 4b). To regain diversity locally around the converged species seed, all particles except the species seed in the converged species are replaced by the same quantity of particles, centered around the species seed. The species seed will remember the found peak, whereas the newly generated quantum particles form a "cloud" to cover the area around the peak (following the uniform shell distribution as discussed in Section 3.2.2). After creation of the cloud, in order to continue to maintain the species' ability to converge, half of the particles in the cloud will be turned into neutral particles while the remaining half of the particles continue to act as quantum particles. Have the peak moved within the radius of the cloud, it would be more likely to be re-located by some of the particles in this cloud.

## 4. EXPERIMENTS

The moving peaks benchmark proposed in [6] claims to cover many of the interesting characteristics of dynamic environments. It consists of a number of peaks that may vary their shapes, positions and heights over time. A suboptimal peak could turn into a global optimal peak and vice versa, making it necessary to "jump" from one peak to another to track the optimum. Intuition and experience tell us that one strategy to ensure good tracking of the global optimum in such an environment is to maintain species at several peaks that have been found, whether they are globally or locally optimal. This strategy has been adopted in all the experiments described in this section. Two sets of experiments have been carried out, one on studying the adaptability of SPSO to a series of dynamic test cases with varying number of peaks, and the 2nd set on examining the effect of applying the particle diversification method (as described in section 3.2.4) to promote particle diversity within each converged species.

### 4.1 Adaptability to varying number of peaks

In the first set of experiments, we are interested in finding out how SPSO performs on the moving peaks scenario2 (see section 5.1) with varying number of peaks. SPSO's ability to adapt to this variation in dynamic environments is studied. We set $p_{max}$ to 10 for each species, since using 10 particles for a species seems to provide a sufficient level of convergence. A population size of 100 is used for experiments over a series of scenario2 test instances, which have 3, 5, 8 10, 12, 15, 18, 50, 100, 200, and 500 peaks respectively. A baseline SPSO model and a SPSO variant incorporating the method of replacing the worst species (section 3.2.3) are used:

1. *BaselineSPSO*: this is the baseline SPSO model with a set maximum number of particles $p_{max}$ allowed for each species. The excess particles (over such a limit) will be replaced by random neutral particles to the total search space. Since SPSO tends to favour fitter species seeds over less-fit seeds, the purpose of using such a $p_{max}$ is to ensure the particles are more evenly distributed across different species (even those less-fit ones).

2. *ReplaceWSpecies*: similar to the multi-swarm's *anti-convergence* procedure. At each iteration, the SPSO checks to see if the average particle diversity (see section 3.2.3) is below a threshold $\Delta$ (set to 0.5 in this paper). If so, then replaces the worst species with randomly generated particles in the search space. It is expected that this method only becomes effective when the model approaches to convergence, hence diversity becomes lower. By using this method, it is hoped the redistributed random particles help increase the population diversity again, leading to better exploration of the search space.

### 4.2 Particle diversification

Maintaining a sufficient level of particle diversity, and handling the trade-off between convergence and diversity during a run is critical for optima tracking in a dynamic environment. Quantum particles (see section 3.2.2) are especially

**Table 1: Averaged offline errors (and std. error) on scenario2 with 10 peaks, when varying $p_{max}$.**

| $p_{max}$ | BSPSO | RWS |
|---|---|---|
| 4 | 6.75(0.16) | 6.80(0.15) |
| 6 | 3.85(0.10) | 3.88(0.12) |
| 8 | 3.09(0.10) | 3.04(0.11) |
| 10 | 2.96(0.08) | **2.92(0.09)** |
| 12 | 3.64(0.14) | 3.08(0.12) |
| 14 | 3.54(0.12) | 3.06(0.11) |

useful in promoting particle diversity within each species. After a species converges onto a peak, the goal will change to tracking the peak, has its position changed. By having more quantum particles around a peak, the species maintains its ability to respond to such changes.

In the 2nd set of experiments, we are interested in studying the effect of using a particle diversification method (see section 3.2.4) to increase particle diversity within each species once the species meets a convergence measurement, ie., below the particle diversity threshold $\delta$. In particular, we are interested in finding out how sensitive the SPSO's performance is with respect to this $\delta$ parameter. The details of the implementation of the particle diversification method in SPSO (SPSO-PD) is given below.

#### 4.2.1 SPSO-PD

The initial population comprises of neutral particles only. The quantum particles will be only generated through substitution of a converged species, after this species is considered to have converged (ie., its $d_p$ value is below the required $\delta$ value). All particles except the species seed in the converged species are replaced by the same quantity of quantum particles, which are positioned as a "cloud" centered around the species seed. These particles are created within a hypersphere of radius $r_{cloud}$ (which is set to 1.0, the same value as for the amount of shift $s$ specified for scenario2). To maintain a good balance between convergence and diversity, 50% of the particles in the quantum cloud are converted into neutral particles and the remaining 50% continue to function as quantum particles. The $\vec{v_i}$ of these neutral particles are set to the seed's $\vec{v_i}$. Note that a quantum particle' velocity value does not have any effect on its future movement. The above setup is also compared with the situation where no such conversion from quantum particles is allowed (SPSO-PD*). This comparison will allow us see if neutral particles are really effective local hill-climbers. It is hoped that by providing the species with the needed diversity while still maintaining its ability to converge, better tracking behaviours can be produced. In this implementation, we disregard the species containing only a single particle from the above process. At each iteration, the population made up of both quantum and neutral particles will be subject to the clearing procedure (ie., Algorithm 1), therefore the best among both neutral and quantum particles will have an opportunity to be selected as the new species seeds.

If a species exceeds $p_{max}$ number of particles, the excess particles will be replaced by randomly generated neutral particles into the search space (see section 3.2.1). *ReplaceWSpecies* is not invoked in this implementation.

# 5. EXPERIMENTAL SETUP

## 5.1 Moving Peaks

In this research, the moving peak benchmark (MPB) scenario2 is used to test the improved SPSO. Scenario2 has the following settings: the search space has 5 dimensions, with each dimension $X = [0, 100]$. There are 10 peaks[3], with peak heights varying randomly in the range [30, 70], peak width varying randomly within [1, 12]. The peaks change position by a distance of $s = 1$ in a random direction, in every 5000 evaluations. The peak movements are uncorrelated (the MPB coefficient $\lambda = 0$).

## 5.2 Parameter settings

The standard constriction version of PSO is used. $c_1$ and $c_2$ were set to 2.05, and $\chi$ set to 0.72984. A swarm population size of 100 was used. The offline error (as defined in [7]) after 5000 iterations is recoded and used as the performance measure for each model variant. These results are averaged over 50 runs with one standard error. In each of the 50 runs, the moving peaks scenario2 will be generated from a new random seed (using time stamp of the machine). This means the 50 runs are tested over 50 different scenario2 fitness landscapes produced. For scenario2, a species radius of 30 is used, which is based on Blackwell and Branke's results in [4] and also from our preliminary experimental runs.

## 5.3 Detecting changes

To detect whether a change has occurred, we simply re-evaluate the recorded $\vec{p_i}$ positions of the top 5 best species seeds at each iteration. If any of these 5 re-evaluations is smaller than its corresponding $\vec{p_i}$'s recorded fitness, then a change is considered to have occurred. If a change is confirmed to have occurred, all particles' $\vec{p_i}$ are reset to their associated $\vec{x_i}$ since these recorded $\vec{p_i}$ are outdated (see Carlisle and Dozier's approach discussed in section 2.2).

# 6. RESULTS

In this section we summarize the experimental results on the effect of using varying $p_{max}$ values, and the adaptability of the baseline SPSO on scenario2 instances with varying numbers of peaks. We also provide the results of using the proposed particle diversification method (SPSO-PD), which is shown to enhance the SPSO's adaptability in dynamic environments.

## 6.1 Effect of varying $p_{max}$

Table 1 shows the results of varying $p_{max}$ values over BSPSO and RWS on scenario2 case with 10 peaks. $p_{max} = 10$ gives the best result. This can be explained by the fact that there are 10 peaks, and allocating on average 10 particles per species is the best setting for covering 10 peaks. A lower $p_{max}$ value will cause difficulty in convergence within each species, and not effectively use the excess particles; whereas a higher $p_{max}$ value will cause not having enough particles to cover all 10 peaks. RWS becomes more effective than BSPSO in cases where there are more than 10 peaks.

## 6.2 Effect of varying the number of peaks

Table 2 shows the results of varying the number of peaks on the scenario2 test case. It can be seen that BSPSO's

[3]However, peak numbers are varied in this paper.

**Table 2: Averaged offline errors on scenario2 with varying number of peaks, using BSPSO and RWS ($p_{max} = 10$).**

| no. peaks | BSPSO | RWS |
|---|---|---|
| 3 | 3.92(0.32) | 3.98(0.22) |
| 5 | 3.20(0.17) | 3.50(0.21) |
| 8 | 2.91(0.12) | 2.89(0.09) |
| 10 | 3.04(0.13) | 3.04(0.09) |
| 12 | 3.39(0.10) | 2.99(0.08) |
| 15 | 3.54(0.09) | 3.09(0.07) |
| 18 | 4.28(0.12) | 3.41(0.09) |
| 50 | 5.06(0.12) | 3.97(0.08) |
| 100 | 5.38(0.12) | 4.07(0.09) |
| 200 | 5.24(0.13) | 3.95(0.06) |
| 500 | 4.82(0.11) | 3.72(0.06) |

**Table 3: Averaged offline errors on scenario2 with 10 peaks, when varying the particle diversity threshold $\delta$, using SPSO-PD.**

| $\delta$ | SPSO-PD |
|---|---|
| 0.0001 | 3.13(0.09) |
| 0.005 | 2.93(0.09) |
| 0.01 | 2.99(0.09) |
| 0.05 | 2.15(0.08) |
| 0.1 | **1.99(0.07)** |
| 0.2 | **1.93(0.06)** |
| 0.5 | 2.29(0.08) |
| 0.8 | 2.76(0.08) |
| 1.0 | 2.86(0.09) |
| 1.5 | 3.33(0.07) |
| 3.0 | 3.98(0.11) |

performance is similar to RWS for cases ranging from 3 to 10 peaks. However, when there are more than 10 peaks, BSPSO's performance is considerably worse than RWS. This shows again that RWS becomes increasingly effective when larger numbers of peaks are present. It is interesting to note that the worse results were in the range from 50 to 200 peaks, but when there are more than 200 peaks, both BSPSO and RWS in fact reverses its trend of poor performance. Although this is counter-intuitive, it can be explained by the fact that scenario2 uses a fixed range [0, 100] for variables. When the fixed range is overcrowded with larger numbers of peaks, the distances between these peaks will be reduced. Since SPSO uses a fixed $r_s$, this means there will be a large number of peaks located within the same species. As a result, SPSO regains its ability to move particles from lower to higher peaks.[4] Considering that SPSO just has a population of 100 particles, and does not have any prior knowledge about the number of peaks, these results indicate that SPSO is able to adaptively find and track the optimum most of the time with more or less consistent results.

## 6.3 Effect of particle diversification

Table 3 shows the results of using different $\delta$ values. The particle diversification method SPSO-PD performs better when $\delta$ is smaller (i.e., around 0.1), indicating that a good trade-off between convergence and diversification can be reached at these values. Too much diversification (if $\delta$ is high) will be detrimental to the convergence required in order to locate a peak; whereas too little diversification (if $\delta$ is very low) will prevent the algorithm from producing quantum clouds that cover the shift distance before the peaks move, hence losing the tracking ability. Fig. 5 shows that as the model approaches the end of a run, the number of quantum particles increases to just below 50% of the population. This is because for each converged species SPSO-PD converts 50% of the generated new particles into neutral particles.

[4]This is because SPSO favours fitter peaks over less-fit peaks (see section 3.1).

**Table 4: Averaged offline errors on scenario2 with varying number of peaks, using SPSO-PD ($\delta = 0.1$). The 3rd column SPSO-PD* shows the results when no conversion from quantum particles is allowed.**

| no. peaks | SPSO-PD | SPSO-PD* |
|---|---|---|
| 3 | 2.41(0.08) | 2.58(0.10) |
| 5 | 1.98(0.05) | 2.10(0.07) |
| 8 | 1.89(0.06) | 2.10(0.06) |
| 10 | **1.98(0.05)** | 2.10(0.06) |
| 12 | 2.39(0.08) | 2.37(0.06) |
| 15 | 2.78(0.08) | 2.75(0.08) |
| 18 | 2.94(0.08) | 3.04(0.08) |
| 50 | 3.47(0.06) | 3.81(0.10) |
| 100 | 3.60(0.07) | 3.68(0.07) |
| 200 | 3.47(0.07) | 3.43(0.05) |
| 500 | 3.12(0.04) | 3.20(0.05) |

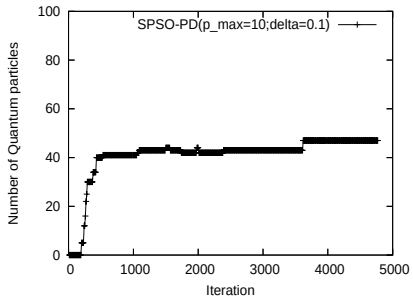


**Figure 5: The number of quantum particles generated during a run of SPSO-PD.**

We rerun the experiments in Table 2 on adaptability to varying number of peaks, but this time SPSO-PD was used. The results are summarized in Table 4. Comparing with Table 2, SPSO's offline errors in Table 4 are dramatically reduced on all test cases. This clearly demonstrates that using the particle diversification method can substantially improve SPSO's tracking performance in a dynamic environment. Table 4 also shows that SPSO-PD is better than SPSO-PD* (where no conversion from quantum particles is allowed) for most test cases. This shows that SPSO-PD with a good balance of neutral and quantum particles is more effective in deriving better tracking behaviours than using quantum particles alone.

## 6.4 Comparison with other works

The self-organizing scouts (SOS) represents a state-of-art evolutionary algorithm for solving dynamic optimization problems. The SOS has been tested over a number of dif-

ferent moving peak scenarios, including scenario2. Since in this paper the SPSO and the multi-swarm model in [4] use the same settings for scenario2 as the SOS, it makes sense to compare SPSO's performance with those of SOS and the multi-swarm model. It was reported in [9] that SOS gives a best offline error of 4.6 on the scenario2 instance with 10 peaks. Even the baseline SPSO results in Table 1 and 2 are better. In [4], a multi-swarm with pre-specified 10 sub-swarms, each comprising of 5 neutral and 5 quantum particles, was used to track the optimum in the scenairo2 with 10 peaks. The best offline error reported is 2.16(0.06). In comparison, the best offline error obtained by SPSO-PD in Table 3 is 1.93(0.06) for 10 peaks.

# 7. CONCLUSIONS

This paper has focused on a study on the adaptability of a speciation-based PSO (SPSO) in a dynamic environment. Two key aspects have been investigated - whether SPSO is able to adapt to a series of dynamic environments with varying number of peaks; and how to promote particle diversity within each species when it is needed. Our results suggest that SPSO can adapt consistently well on most test cases (each has a different number of peaks), which have been considered as a difficulty to existing EAs and PSO models. In addition, this paper has proposed a particle diversification method that can further improve SPSO's adaptability in maintaining a good balance between convergence and diversity within each species, thereby leading to better optima tracking behaviours.

There are issues deserving future studies. For example, finding an appropriate level of balance between the number of neutral and quantum particles which would be suitable for different dynamic problem scenarios; identifying the desirable characteristics of an ideal hill-climbing technique for tracking a peak before and after it has moved. Another important topic is to further improve the computational efficiency of the SPSO model.

# 8. REFERENCES


[1] J. Angeline. Tracking extrema in dynamic environments. In *Proc. of the Evolutionary Programming VI*, pages 335–345, Indianapolis, 1997. Berlin: Springer-Verlag.

[2] T. Back. On the behaviour of evolutionary algorithms in dynamic environments. In *Proc. of International Conference on Evolutionary Computation*, pages 446–551. NJ: IEEE Press, 1998.

[3] T. Blackwell and P. Bentley. Dynamic search with charged swarms. In *Proc. the Workshop on Evolutionary Algorithms Dynamic Optimization Problems (EvoDOP-2003)*, pages 19–26, 2002.

[4] T. Blackwell and J. Branke. Multi-swarm optimization in dynamic environments. In *LNCS, No. 3005, Proc. of Applications of Evolutionary Computing: EvoWorkshops 2004: EvoBIO, EvoCOMNET, EvoHOT, EvoISAP, EvoMUSART, and EvoSTOC*, pages 489–500, 2004.

[5] T. Blackwell and J. Branke. Multi-swarms, exclusion, and anti-convergence in dynamic environments. *IEEE Transactions on Evolutionary Computation*, to appear.

[6] J. Branke. Memory enhanced evolutionary algorithms for changing optimization problems. In *Proc. Congress on Evolutionary Computation CEC'99*, pages 1875–1882, 1999.

[7] J. Branke. *Evolutionary Optimization in Dynamic Environments*. Kluwer Academic Publishers, Norwell, MA, 2002.

[8] J. Branke, E. Salihoglu, and S. Uyar. Towards an analysis of dynamic environments. In *Genetic and Evolutionary Computation Conference*, pages 1433–1439. ACM, 2005.

[9] J. Branke and H. Schmeck. Designing evolutionary algorithms for dynamic optimization problems. In S. Tsutsui and A. Ghosh, editors, *Theory and Application of Evolutionary Computation: Recent Trends*, pages 239–262. Springer, 2002.

[10] A. Carlisle and G. Dozier. Adapting particle swarm optimization to dynamic environments. In *Proc. the International Conference on Artificial Intelligence (ICAI2000)*, pages 429–434, 2000.

[11] A. Carlisle and G. Dozier. Tracking changing extrema with adaptive particle swarm optimizer. In *Proc. World Automation Cong.,*, pages 265–270, Orlando FL USA, 2002.

[12] 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.

[13] R. C. Eberhart and Y. Shi. Tracking and optimizing dynamic systems with particle swarms. In *Proc. the 2001 Congress on Evolutionary Computation CEC2001*, pages 94–100. IEEE Press, 27-30 May 2001.

[14] X. Hu and R. Eberhart. Adaptive particle swarm optimisation: detection and response to dynamic systems. In *Proc. Congress on Evolutionary Computation*, pages 1666–1670, 2002.

[15] S. Janson and M. Middendorf. A hierarchical particle swarm optimizer for dynamic optimization problems. In G. Raidl, editor, *Applications of Evolutionary Computing, LNCS (3005)*, pages 513–524. Springer, 2004.

[16] Y. Jin and J. Branke. Evolutionary optimization in uncertain environments - a survey. *IEEE Trans. Evol. Comput.*, 9:303–317, 2005.

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

[18] 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.

[19] 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.

[20] D. Parrott and X. Li. A particle swarm model for tacking multiple peaks in a dynamic environment using speciation. In *Proc. of the 2004 Congress on Evolutionary Computation*, pages 98–103, 2004.

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

[22] R. Ursem. Multinational GAs: Multimodal optimization techniques in dynamic environments. In D. Whitley, editor, *Proc. Congress on Evolutionary Computation*, pages 1666–1670, 2002.