Proceedings of the 2007 IEEE Swarm Intelligence Symposium (SIS 2007)

Particle Swarm Optimization in High-Dimensional Bounded Search Spaces

Sabine Helwig and Rolf Wanka
Computer Science Department
University of Erlangen-Nuremberg, Germany
{sabine.helwig, rwanka}@informatik.uni-erlangen.de

***Abstract*— When applying Particle Swarm Optimization (PSO) to real world optimization problems, often boundary constraints have to be taken into account. In this paper, we will show that the bound handling mechanism essentially influences the swarm behavior, especially in high-dimensional search spaces. In our theoretical analysis, we will prove that all particles are initialized very close to the boundary with overwhelming probability, and that the global guide is expected to leave the search space in every forth dimension. Afterwards, we investigate the initialization process when optimizing the Sphere function, a widely used benchmark, in more detail in order to provide a first step towards explaining previously observed phenomena. Moreover, we will present a broad experimental study of commonly applied bound handling mechanisms on a variety of benchmark functions which is useful for choosing an appropriate strategy in real world applications. Finally, we will derive some guidelines for the practical application of the PSO algorithm in high-dimensional bounded search spaces.**

I. Introduction

Since its invention in 1995, *Particle Swarm Optimization (PSO)* [1] has successfully been applied to many continuous optimization problems. The algorithm is based on the social behavior of individuals living together in groups. Each individual, also called *particle*, tries to improve itself by evaluating own experiences as well as by imitating better group members. Translating this behavior to optimization problems, particles are flying through the search space S, each one having a *position* $\vec{x}_t$ (where t is the iteration counter), a fitness value $f(\vec{x}_t)$ (where f is the objective function of the optimization problem), and is moving through the search space with a *velocity* $\vec{v}_t$. To each particle P, a subset of the whole swarm is assigned as its neighborhood (the resulting neighborhood graph is called the particles' *topology*), and P changes its position and velocity according to the following equations:

$$\begin{aligned}\vec{v}_t &= \omega\cdot\vec{v}_{t-1}+\vec{U}[0,c_1]\otimes(\vec{p}_{t-1}-\vec{x}_{t-1}) \\ &\quad +\vec{U}[0,c_2]\otimes(\vec{l}_{t-1}-\vec{x}_{t-1}) \\ \vec{x}_t &= \vec{x}_{t-1}+\vec{v}_t\end{aligned}$$

where

- $\vec{U}[m,n]$ is a vector of random real numbers between m and n;
- $\vec{p}_{t-1}$ is P's best position so far, called its *private guide*;
- $\vec{l}_{t-1}$ is the best position ever visited by one of P's neighbors, called its *local guide*;
- $\otimes$ denotes element-by-element vector multiplication;
- c_1 determines the importance of P's own experiences;
- c_2 determines how strong P is attracted by its local guide;
- ω is called *inertia weight* and determines the influence of P's old velocity.

In some PSO algorithms, the position of a particle is perturbed at the end of each iteration with a very low probability in order to allow the particles to escape local optima. This procedure is called *turbulence* [2].

There is a lot of theoretical and experimental work on particle swarm optimization, providing parameter selection guidelines [3]–[6], or analyzing the influence of the swarm's neighborhood structure [7]. There, mostly optimization problems with unbounded search spaces are studied, i.e., the objective function f is defined in the whole $\mathbb{R}^n$, where n is the number of dimensions. Real world optimization problems often have restricted search spaces, i.e., a number of inequality and equality constraints is given by:

$$\begin{aligned}g_i(\vec{x}) &\le 0 \qquad i=1,\ldots,m_1 \\ h_i(\vec{x}) &= 0 \qquad i=1,\ldots,m_2\end{aligned}$$

In this paper, we focus on optimization problems with *boundary constraints*, which means that the i-th dimension of the objective function is restricted by a lower bound lb_i and an upper bound ub_i:

$$lb_i \le x_i \le ub_i \quad \forall i=1\ldots n$$

There have already been some proposals for bound handling mechanisms, which will be presented below. However, until now, these methods have only been compared experimentally [8], [9], if at all, and the effects of the bound handling strategies have hardly been explained. This paper will give insight into the peculiarities of high-dimensional search spaces, and it will provide first steps towards explaining some earlier mentioned experimental observations [7], [9], [10]. Moreover, we will demonstrate the importance of the bound handling strategy as part of the particle swarm optimization algorithm. We will show theoretically that bounds are violated very often in high-dimensional particle swarm optimization, and thus, the output quality strongly depends on the chosen bound handling method.

After introducing some bound handling mechanisms in Section II, we will provide first theoretical results on using particle swarm optimization in bounded high-dimensional search spaces in Section III. Afterwards, experimental results are discussed in Section IV.

II. Handling Boundary Constraints

In most analytical papers on particle swarm optimization, typical benchmark functions without any constraints are studied. But when applying particle swarm optimization on real world problems, we need strategies for dealing with boundary constraints due to the following reasons:

1) Boundary constraints may simplify the problem. Assume that one of the parameters is an angle. If we reduce our search space in the angle's dimension to $[0..2\pi]$, there are significantly fewer local optima without changing the optimization problem at all.
2) Using search space bounds can avoid needless objective function evaluations. If we already know that the global optimum and many local ones lie within a certain search space region before running an optimization algorithm, we can restrict the search to that region by introducing boundary constraints. This approach especially pays when the evaluation of the objective function is very expensive, as often is the case in real world problems.
3) Boundary constraints might be part of the optimization problem.

We have chosen five simple and commonly used bound handling methods for analyzing the influence of boundary constraints on the optimization process:

- **Inf:** Mendes [7] proposes to modify the objective function such that invalid values are mapped to $+\infty$ in minimization problems and to $-\infty$ in maximization problems, respectively (see Fig. 1 (a)).
- **Nearest:** Each particle violating one or more boundary contraints is reset on the nearest search space border, as shown in Fig. 1 (b).
- **Nearest+Turb:** Like *Nearest*, but, additionally, the *turbulence* operator, as explained in Section I, is used.
- **Random:** If a particle exceeds the lower or upper limit of the i-th dimension, a random value, uniformly distributed between $[lb_i..ub_i]$, is assigned to the i-th component of the particle's position vector. This procedure is illustrated in Fig. 1 (c). *Random* and *Nearest* have been used by Zhang et al. [9] for comparison purposes.
- **Shr:** When using the *Shr* method [8], the magnitude of the velocity vector is shrunken such that the particle exactly reaches the search space bound, as illustrated in Fig. 1 (d).

Whenever a particle is reset into the search space or on the boundary, its velocity is adjusted according to the following equation:

$$\vec{v}_t = \vec{x}_t - \vec{x}_{t-1} \tag{1}$$

Other, more elaborate bound handling strategies also exist [8], but they are not discussed here, as they do not provide further insight into the swarm's behavior, and often, they even do not yield superior results in comparative experiments. Some experimental results on bound handling methods have already been published: Alvarez-Benitez et al. [8] have investigated four different bound handling methods. *Shr* has outperformed the other ones in their experiments. Mendes [7] noticed that sometimes, resetting particles on the boundary might lead to bad results. From the above methods, *Nearest*, *Shr*, and *Nearest+Turb* are bound resetting methods. Zhang et al. [9] observed experimentally that *Nearest* might lead to premature convergence on the boundary. Our theoretical analysis provides first steps for explaining these results.

Michalewicz and Schoenauer [11] present an overview of constraint handling mechanisms for evolutionary algorithms. These general constraint handling mechanisms like penalizing unfeasible solutions can of course also be applied in particle swarm optimization [12]. However, an analysis of such general mechanisms is beyond the scope of this paper.

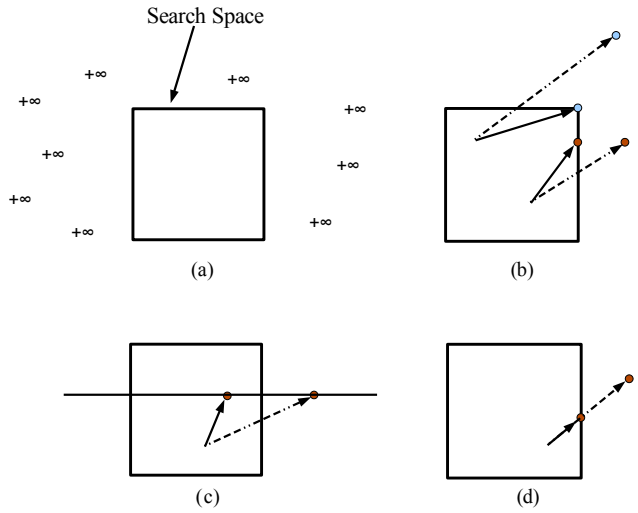

Fig. 1. In this figure, the bound handling methods analyzed in this paper are shown. (a): Inf, (b): Nearest, (c): Random, and (d): Shr

III. Theoretical Results

In this section, we will provide first theoretical results on particle swarms in high-dimensional search spaces with boundary constraints. The "curse of high dimensionality", which means that high-dimensional spaces are not intuitive, is well known in mathematics, physics, statistics, and some computer science research areas like data mining [13]. Our results show that it is also an important topic for particle swarm optimization, especially when solving optimization problems with boundary constraints.

We assume the so-called *gbest*-topology, which means that all particles are adjacent. The particle which has found the best known position of the whole swarm, and carries this information in its private guide, will from now on be called *global guide*. Its private guide serves as local guide for all particles (including itself).

In the following, the symbols Θ and Ω belong to the *big-O notation* for expressing asymptotic behavior [14, p. 108ff].

Since particle swarm optimization is a stochastic optimization algorithm, mostly we will only be able to evaluate the

probability of certain events. Thus, we will apply the following widely-used notion:

Definition 3.1: A probability $p(n)$ is *exponentially small* in n if there exists a constant $\alpha > 0$ such that $p(n) = e^{-\Omega(n^\alpha)}$. An event $A(n)$ happens *with overwhelming probability (w.o.p.)* with respect to n if $P(\text{not } A(n))$ is exponentially small in n.

In this section, we will at first analyze the initialization step of the PSO algorithm and show that in an n-dimensional search space, particles are initialized very close to the border w.o.p. This result already indicates the importance of the bound handling mechanism since particles which are located near the boundary often might leave the search space. We show afterwards that, under the simplifying assumption $\omega = 1$, the global guide violates a high number of search space bounds, namely $\frac{1}{4}n$. This means that w.o.p., a bound handling strategy has to be applied to the global guide. Hence, the bound handling mechanism essentially influences the behavior of the whole particle swarm.

Zhang et al. [9] have already observed experimentally that particles might converge on the boundary when using the *Nearest* method. Our experiments, which will be presented in Section IV, confirm this observation, and demonstrate that the particles also converge in simple optimization problems like the Sphere benchmark. Nevertheless, *Nearest* works well in other scenarios. We will provide first analytical steps for explaining these results. A necessary condition for premature convergence is the absence of good solutions at the beginning of the optimization process, as otherwise, the swarm would be attracted by them. In the third part of our theoretical analysis, we will evaluate the expected initial function values of the particles when solving the Sphere function and show that they can easily be outperformed by boundary solutions. Moreover, we will prove that the probability of a particle to be initialized on a position which is better than all boundary ones is exponentially small.

Wolpert and Macready have shown that if an optimization algorithm works exceptionally well for one class of optimization problems, its output quality must be poor for another class [15]. Although they just regard optimization algorithms which do not converge (but the PSO algorithm mostly does if the parameters are selected carefully), this statement has also shown to be true for bound handling mechanisms in our experiments. Thus, we will not provide something like "the best" bound handling strategy in our paper since it depends on the optimization problem which one performs best. Instead, we will provide well-founded insights into particle swarm optimization in high-dimensional bounded search spaces in this section, and demonstrate the effects of bound handling methods on a variety of benchmarks in Section IV.

A. Particle Initialization in High-Dimensional Search Spaces

In the standard PSO algorithm, particles are uniformly distributed in the search space at the beginning of the optimization process. This means that in high-dimensional search spaces, where most of the volume is concentrated in a small shell near its surface, particles are initialized very close to at least one boundary, w.o.p.:

Theorem 3.1: Consider the standard PSO algorithm presented in Section I in an n-dimensional search space bounded by $[-r \dots r]^n$. Then, for arbitrary $\varepsilon > 0$, the probability $p_B(r,n,\varepsilon)$ of a particle to be inititalized such that the distance to its nearest border is less than ε is $1 - e^{-\Theta(n)}$.

Proof: The volume of an n-dimensional hypercube with side length $2r$ is $(2r)^n$. Thus, $p_B(r,n,\varepsilon)$ evaluates to:

$$p_B(r,n,\varepsilon) = \frac{(2r)^n - (2r-2\varepsilon)^n}{(2r)^n} = 1 - e^{-\Theta(n)}$$

■

This result also explains why it is advantageous to not initialize particles with uniform distribution in some scenarios, as has been proposed by Richards and Ventura [10]: A global optimum which is located near the center of the search space can more easily be found if not all particles are placed near the boundary. However, if nothing is known about the optimization problem beforehand, it certainly makes sense to initialize the particles with uniform distribution as most of the search space volume is located near the boundary.

B. The Global Guide's Border Violations in the 1st Iteration

As we have seen in Theorem 3.1, particles are initialized very close to at least one border w.o.p., and therefore, it seems to be very probable that particles leave the search space in the first iteration. In fact, the number of borders violated by the global guide in the first iteration is very high as shown in the following theorem:

Theorem 3.2: Assume $\omega = 1$, and assume that both positions and velocities are initialized uniformly in $[-r \dots r]^n$. Then, the global guide is expected to leave the search space in $\frac{1}{4}n$ dimensions after the first iteration, where n is the number of search space dimensions.

Proof: In the standard PSO algorithm, the position of the global guide after the first iteration can be evaluated to

$$\vec{x}_1 = \vec{x}_0 + \vec{v}_1$$

with

$$\begin{aligned} \vec{v}_1 \quad = \quad & \omega \cdot \vec{v}_0 + \vec{U}[0,c_1] \otimes (\vec{p}_0 - \vec{x}_0) \\ & + \vec{U}[0,c_2] \otimes (\vec{l}_0 - \vec{x}_0) \ . \end{aligned}$$

In the first iteration, for every particle we obtain $\vec{p}_0 - \vec{x}_0 = \vec{0}$ since no position besides $\vec{x}_0$ has been visited so far. For the global guide we addionally get $\vec{l}_0 - \vec{x}_0 = \vec{0}$. As ω has been set to 1, the d-th component of the global guide's position vector computes to:

$$x_{1d} = x_{0d} + v_{0d}$$

with x_{0d} and v_{0d} uniformly distributed between $-r$ and r, respectively, for all $d = 1 \dots n$, as these are the initialization values. Thus, the density function for x_{0d} and v_{0d} evaluates to:

$$f_{x_{0d}}(z) = f_{v_{0d}}(z) = \begin{cases} \frac{1}{2r} & \text{for } -r \le z \le r \\ 0 & \text{otherwise} \end{cases}$$

As x_{0d} and v_{0d} are stochastically independent, the density function of x_{1d} computes to:

$$\begin{aligned} f_{x_{1d}}(z) &= \begin{cases} \int_{-r}^{z+r} f_{x_{0d}}(x) \cdot f_{v_{0d}}(z-x)\mathrm{d}x \\ \int_{z-r}^{r} f_{x_{0d}}(x) \cdot f_{v_{0d}}(z-x)\mathrm{d}x \\ 0 \end{cases} \\ &= \begin{cases} \frac{1}{4r^2}z + \frac{1}{2r} & \text{for } -2r \le z \le 0 \\ -\frac{1}{4r^2}z + \frac{1}{2r} & \text{for } 0 \le z \le 2r \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

Thus, for each dimension d, the probability of the global guide to exceed one of the search space boundaries evalutes to:

$$P(|x_{1d}| > r) = \int_{-2r}^{-r} f_{x_{1d}}(z)\mathrm{d}z + \int_{r}^{2r} f_{x_{1d}}(z)\mathrm{d}z = \frac{1}{4}$$

■

Corollary 3.3: With the assumptions of Theorem 3.2, the global guide leaves the search space in the first iteration, w.o.p.

Proof: The probablity $p_C(n)$ of the global guide leaving the search space evaluates to:

$$p_C(n) = 1 - \left(\frac{3}{4}\right)^n = 1 - e^{-\Theta(n)}$$

■

In most PSO applications (and also in our experimental analysis in Section IV), ω is set to a value smaller than 1. However, Theorem 3.1 and Theorem 3.2 show that all particles are initialized near the search space border in high-dimensional search spaces (which makes sense as most of the search space volume is located near the borders) and that at least the global guide will leave the search space in the first iteration w.o.p. Moreover, it not only leaves the search space, but it is expected to violate a very high number of boundaries. Theorem 3.1 indicates a similar behavior for the other particles as they also are initialized very close to the search space bounds.

Hence, we conclude that how to deal with boundary constraints is not a rarely used mechanism in a PSO algorithm, but an important procedure applied very often, particularly in high-dimensional search spaces. It therefore has great influence on the swarm behavior, as will also be shown in our experiments in Section IV.

C. A concrete example: The Sphere function

As already mentioned, sometimes premature convergence has been observed when solving the bounded Sphere function which is a very simple optimization problem. Therefore, we will have a closer look at the Sphere function. We will show that boundary solutions can outperform the initial function value, and that w.o.p., particles are not initialized to positions which are better than the best boundary solution. These conditions are necessary for premature convergence on the search space bound, but they are not intuitive as they are not true for low-dimensional search spaces. The Sphere function description is as follows:

$$f(\vec{x}) = \sum_{i=1}^{n} x_i^2$$

where n is the number of search space dimensions and can be arbitrarly large. The search space is restricted to $[-r \dots r]^n$.

Some bound handling strategies such as *Nearest*, *Nearest+Turb*, and *Shr* reset particles on the search space bounds which might lead to premature convergence if the global guide arrives on a good position on the search space boundary. When a particle converges on one boundary by optimizing all other dimensions, the objective function value evaluates to $f(\vec{x}) = r^2$ which is much better than the expected initial function value:

Theorem 3.4: We assume that all particles are initialized with uniform distribution in an n-dimensional search space bounded by $[-r \dots r]^n$. Then, the expected initial function value of a particle solving the Sphere function is $\frac{nr^2}{3}$ and its standard deviation is $\frac{2\sqrt{n}r^2}{3\sqrt{5}}$.

Proof: Let $y = \sum_{i=1}^{n} x_i^2$ be the initial function value. Then, $x_1, x_2, \dots, x_n$ are stochastically independent and uniformly distributed in $[-r \dots r]$ each. We obtain:

$$E(y) = n \cdot E(x_i^2) = n \int_{-r}^{r} x^2 \frac{1}{2r}\mathrm{d}x = \frac{nr^2}{3}$$

and

$$\sigma_y = \sqrt{\mathrm{Var}(y)} = \sqrt{n(E(x_i^4) - E(x_i^2)^2)} = \frac{2\sqrt{n}r^2}{3\sqrt{5}}$$

■

For example, for $n = 100$ and $r = 100$ we get:

$$\begin{aligned} E(y) &\approx 333\,333 \\ \sigma_y &\approx 29\,814 \end{aligned}$$

As the standard deviation is very small compared to the expected initial function value, this means that nearly all particles have a poor initial function value. A particle which is reset on a boundary therefore might attract other particles at the beginning of the optimization process, and lead to premature convergence.

Whenever the swarm converges, this means that no better solution has been found during the whole optimization process, because if there had already been a better solution, it would attract the particles. The Sphere function has the following property:

$$\forall \vec{x}, \vec{y} \in S: \quad |\vec{x}| < |\vec{y}| \Rightarrow f(\vec{x}) < f(\vec{y})$$

where S is the search space. It is important to note that the results presented below are relevant not only for the Sphere function but for all functions with the above property. In Fig. 2, this property is illustrated. All solutions lying on the boundary of the circle have the same objective value. Solutions which lie inside the circle are better, solutions lying outside are worse. Let P be a particle which is located on one boundary in this two-dimensional example, by optimizing the other dimension. All solutions inside the circle are better, and thus, it seems to be very unlikely that P is the best solution visited so far. In fact, we evaluate:

$$\frac{\text{Circle area}}{\text{Square area}} = \frac{\pi r^2}{(2r)^2} = \frac{\pi}{4} \approx 0.785$$

$$V_S(r,n) = \int_{\beta_n=0}^{\pi}\int_{\beta_{n-1}=0}^{\pi}\cdots\int_{\beta_3=0}^{\pi}\int_{\alpha=0}^{2\pi}\int_{R=0}^{r}(R\sin\beta_n\ldots\sin\beta_3 d\alpha)(R\sin\beta_n\ldots\sin\beta_4 d\beta_3)\ldots(R\sin\beta_n d\beta_{n-1})(R d\beta_n)dR \quad (2)$$

which means, that already in the initialization step, 78.5% of the particles are expected to lie inside the circle.

However, high-dimensional search spaces are not intuitive:

Theorem 3.5: Assume the search space bounded by $[-r\ldots r]^n$. Then, the probability $p_A(n)$ of a particle to be initialized inside a hypersphere around the origin with radius r is exponentially small in n.

Proof: Extending an approach of Jägersküpper who evaluated the hypersurface of an n-dimensional sphere [16], we compute the hypervolume $V_S(r,n)$ of an n-dimensional sphere with radius r for $n \geq 3$ to Equation (2), which then can be simplified to:

$$V_S(r,n) = \frac{1}{n}r^n 2\pi\prod_{i=1}^{n-2}\int_0^{\pi}(\sin\beta)^i d\beta$$

The volume of an n-dimensional hypercube with side length $2r$ is $V_C(r,n) = (2r)^n$. Obviously,

$$g(i) = \int_0^{\pi}(\sin\beta)^i d\beta$$

is a monotonic decreasing function. Furthermore, we evaluate $g(1) = 2$ and $g(6) = \frac{5}{16}\pi < 1$. Thus, the probability of a particle to be initialized inside the n-dimensional hypersphere computes to:

$$\begin{aligned} p_A(n) &= \frac{V_S(r,n)}{V_C(r,n)} = \frac{2\pi\prod_{i=1}^{n-2}\int_0^{\pi}(\sin\beta)^i d\beta}{n2^n} \\ &= \frac{2\pi}{n2^n}\cdot\prod_{i=1}^{5}\int_0^{\pi}(\sin\beta)^i d\beta\cdot\prod_{i=6}^{n-2}\int_0^{\pi}(\sin\beta)^i d\beta \\ &\leq \frac{2\pi}{n2^n}\cdot 2^5\cdot 1 = e^{-\Theta(n)} \end{aligned}$$

■

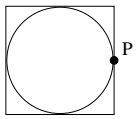

Fig. 2. In the Sphere function the following property holds: the farer away a particle is from the origin, the poorer is its quality.

For example, for 30-dimensional search spaces, $p_A(n)$ computes to $2.04\cdot 10^{-14}$. This means that in high-dimensional search spaces bounded by $[-r\ldots r]^n$, particles are usually initialized outside a sphere around the origin with radius r, which is a necessary condition for the particles to converge on a boundary when solving an arbitrary problem with the above property.

Our experimental results presented in the next section show that when solving the bounded Sphere function in high-dimensional search spaces, premature convergence is a serious problem when using bound handling strategies which reset the particles on the search space bound.

Our theoretical analysis provided a first step towards explaining the swarm's behavior in high-dimensional bounded search spaces. However, further research has to be done to completely understand the experimental results discussed in the next section.

IV. Experimental Results

In the following experiments, the effects of the bound handling strategy on the algorithm's output quality are studied on six benchmark functions. The chosen test functions are commonly used to analyze the PSO algorithm [1], [3]–[7]. It has to be noticed that many benchmarks are designed such that the global optimum is located at the center of the search space. In order to get a broader overview on the effects of bound handling mechanisms, we have studied those functions in both balanced (commonly used) und unbalanced search spaces.

A. Test Functions

All test functions used in the experiments are bounded minimization problems. The function descriptions and the lower and upper limits of the search spaces can be found below. All chosen benchmarks can have arbitrary many dimensions. If not stated otherwise, 30 dimensions have been used.

- **Sphere:** $x_i \in [-100\ldots 100], \forall i = 1\ldots n$

  $$f(\vec{x}) = \sum_{i=1}^{n} x_i^2$$

  Sphere is a very simple function with only one local optimum (which also is the global one): $f(0,\ldots,0) = 0$.
- **Rastrigin:** $x_i \in [-5.12\ldots 5.12], \forall i = 1\ldots n$

  $$f(\vec{x}) = 10\cdot n + \sum_{i=1}^{n}\left(x_i^2 - 10\cdot\cos(2\cdot\pi\cdot x_i)\right)$$

  The Rastrigin function has many local optima, which are regularly distributed. The global minimum is $f(0,\ldots,0) = 0$.
- **Rosenbrock:** $x_i \in [-2.048\ldots 2.048], \forall i = 1\ldots n$

  $$f(\vec{x}) = \sum_{i=1}^{n-1}\left(100\cdot\left(x_{i+1} - x_i^2\right)^2 + (1-x_i)^2\right)$$

  Rosenbrock is an unimodal function whose optimum $f(1,\ldots,1) = 0$ is inside a long and narrow valley.
- **Schwefel:** $x_i \in [-500\ldots 500], \forall i = 1\ldots n$

  $$f(\vec{x}) = \sum_{i=1}^{n}\left(-x_i\cdot\sin\left(\sqrt{|x_i|}\right)\right)$$

  The global optimum $f(420.9687,\ldots,420.9687) = -n\cdot 418.9829$ is located far away from the second best local minimum, in one corner of the search space. There are many local optima.

- **Griewank:** $x_i \in [-600 \dots 600], \forall i = 1 \dots n$

$$f(\vec{x}) = \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$$

  With its many local minima regularly distributed over the search space, the Griewank function resembles the Rastrigin test function. The global minimum is $f(0,\dots,0) = 0$.
- **Michalewicz:** $x_i \in [0 \dots 3.14], \forall i = 1 \dots n$

$$f(\vec{x}) = -\sum_{i=1}^{n} \left(\sin(x_i) \cdot \left(\sin\left(\frac{i \cdot x_i^2}{\pi}\right)\right)^{2 \cdot m}\right)$$

  Michalewicz is a parameterized, multi-modal function with many local optima located between plateaus. The higher the parameter m, the more difficult it is to find the global optimum. In our experiments, m is set to 10.

B. Settings

In the following experiments, the impact of the bound handling mechanism on the swarm's behavior is studied. All other parameters of the PSO algorithm are kept constant for all experiments, and have been set to commonly used values [7], [17]: The control parameters c_1 and c_2 are set to 1.49445 and the inertia weight ω is set to 0.729. A population of 20 particles is used, whose positions and velocities are initialized randomly, with uniform distribution, in the respective search space. The *gbest* topology is used, which means that all particles are connected. When using turbulence, the position $\vec{x}_t$ of each particle is reinitialized with a probability of 0.01 at the end of each iteration. If turbulence took place, the velocity is updated according to Equation 1. Each optimization run terminates after 1000 iterations. In order to provide reliable statistic results, each experiment, i.e., each combination of test function and bound handling method, was repeated 1000 times. In all diagrams and tables below, mean values and standard deviations are presented.

C. Results

The results of our experiments for balanced search spaces, i.e., using the upper and lower search space limits presented above with the function descriptions, are shown in Table I. In all test functions besides the Schwefel function, using *Nearest* or *Nearest+Turb* as boundary handling mechanism resulted in very poor solution quality on average, and also the *Shr* strategy was slightly worse than *Inf* or *Random*.

In Fig. 3, the average objective values and standard deviations (vertical bars) of solving the Sphere function with different bound handling mechanisms are shown. The high standard deviations when using *Nearest* or *Nearest+Turb* indicate that mostly, the optimum is found, but sometimes very bad solutions are produced. A closer look at the outputs of the PSO algorithm applying *Nearest* reveals that either the global optimum has been found (in 922 from 1000 runs), or the swarm converged on at least one boundary (in the remaining 78 runs). This results are conform with Zhang et al. [9] who mentioned that particles might converge on the boundary when using *Nearest*. In our experiments, even turbulence often could not prevent premature convergence. In Theorem 3.4 we have already learned that boundary solutions can easily outperform the initial ones. Theorem 3.1 and Theorem 3.2 indicate that particles often leave the search space at the beginning of the optimization process, especially in high-dimensional search spaces. Thus, we expect the particles to converge more often on the boundary in high-dimensional spaces than in low-dimensional ones. In further experiments we varied the search space dimensions for the Sphere function. Non-surprisingly when resuming our theoretical analysis, we yielded the following results: The higher the search space dimension, the more often the particles converged on at least one boundary. In two-dimensional search spaces, the global optimum has been found in all runs no matter which bound handling strategy is applied. In an 100-dimensional search space, however, only 488 of 1000 optimization runs using the *Nearest* mechanism succeeded to find the global optimum of the sphere function, whereas in 512 runs the particles converged on at least one boundary. More details can be found in Table II.

Kennedy and Eberhart propose to initialize positions as well as velocities to random values [1, p. 314], but velocities might also be set to zero at the beginning [18]. Then, when solving the 30-dimensional Sphere function using *Nearest*, the global optimum has been found in 980 from 1000 runs, and the swarm only converged on the boundary in the remaining 20 runs. When solving the Sphere function in 100 dimensions, the swarm converged on at least one boundary in 275 runs. Hence, initializing velocities to zero yields to better results on average when solving the Sphere function, but cannot prevent the particles from converging on the boundary.

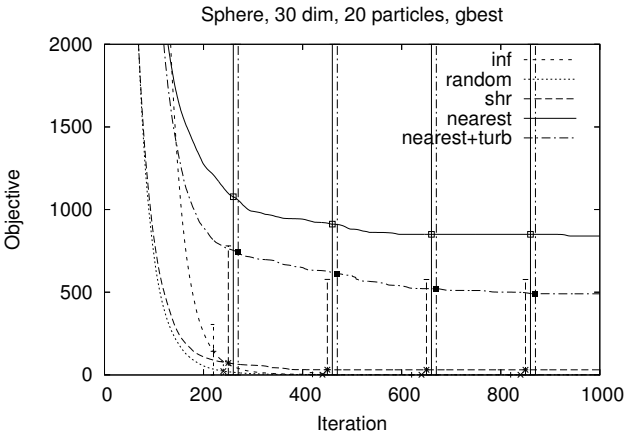

Fig. 3. For most of the benchmark functions, the bound handling methods *Nearest* and *Nearest+Turb* lead to very poor output quality.

In the Sphere, Rastrigin and Griewank function, the global optimum is located exactly in the center of the search space which is unrealistic for real PSO applications. Thus, we run further experiments in unbalanced search spaces. The new lower and upper bounds as well as mean values and standard

TABLE I

EFFECTS OF THE BOUND HANDLING METHOD ON THE PARTICLES' CONVERGENCE BEHAVIOR IN BALANCED SEARCH SPACES.

	Sphere	**Schwefel**	**Rosenbrock**	**Rastrigin**	**Michalewicz**	**Griewank**
Inf	0 ± 0	-8205.30 ± 641.45	30.71 ± 18.58	84.59 ± 19.79	-21.58 ± 1.94	0.09 ± 0.19
Random	0 ± 0	-7866.42 ± 571.86	28.50 ± 15.50	62.18 ± 14.89	-24.71 ± 1.16	0.09 ± 0.26
Shr	30.00 ± 546.90	-7818.01 ± 795.88	30.89 ± 27.58	96.21 ± 25.38	-22.95 ± 1.51	0.38 ± 4.97
Nearest	840.00 ± 2845.07	-8862.64 ± 717.41	91.81 ± 212.40	131.33 ± 33.35	-19.72 ± 1.75	8.37 ± 26.56
Nearest+Turb	490.04 ± 2158.67	-9666.85 ± 673.25	80.90 ± 183.64	123.95 ± 33.42	-19.15 ± 1.94	5.28 ± 21.49

This table shows the average best found objective value and its standard deviation for each combination of test function and bound handling method.

TABLE II

WHEN SOLVING THE BALANCED SPHERE FUNCTION, THE PARTICLES OFTEN CONVERGED ON ONE OR MORE BOUNDARIES WHEN USING THE *Nearest* METHOD IN HIGH-DIMENSIONAL SEARCH SPACES.

	2-dim	30-dim	50-dim	100-dim
Global optimum found	1000	922	847	488
Convergence on 1 bound	0	76	143	348
Convergence on 2 bounds	0	2	9	133
Convergence on 3 bounds	0	0	1	28
Convergence on ≥ 4 bounds	0	0	0	3

For each number of dimensions (2, 30, 50, 100), 1000 optimization runs have been performed. This table shows the exact number of runs in which the global optimum has been found and in which the particles converged on one or more boundaries, respectively.

deviations of the best found solutions of 1000 runs are shown in Table III. The obtained results totally differ from running the PSO algorithm in balanced search spaces. When solving the Sphere function, using *Random* and *Inf* lead to slow convergence, whereas *Nearest*, *Nearest+Turb*, and *Shr* produced better results. However, convergence on the boundary has not been prevented. On contrary, when using *Nearest*, the particles converged on the boundary in all optimization runs. But this swarm behavior of course nevertheless produces good solutions when good, near optimal solutions are located on the boundary.

TABLE III

EFFECTS OF THE BOUND HANDLING METHOD ON THE PARTICLES' CONVERGENCE BEHAVIOR IN UNBALANCED SEARCH SPACES.

	Sphere $[-1..100]$	**Rastrigin** $[-0.512..5.12]$	**Griewank** $[-6..600]$
Inf	247.76 ± 711.51	110.75 ± 24.02	3.21 ± 6.55
Random	3578.4 ± 1127.5	91.37 ± 19.48	33.21 ± 10.15
Shr	9.95 ± 84.94	125.92 ± 30.09	1.23 ± 1.90
Nearest	61.73 ± 631.13	104.14 ± 35.42	1.56 ± 5.68
Nearest+Turb	22.16 ± 2.61	90.50 ± 32.94	1.20 ± 0.02

This table shows the average best found objective value and its standard deviation for each combination of test function and bound handling method.

Our experimental results show that the bound handling mechanism has great influence on the swarm's behavior. In many scenarios, *Nearest*, *Nearest+Turb*, and *Shr* have proven to promote premature convergence on the boundary, as has already been observed by Mendes [7] and Zhang et al. [9]. However, when solving the Schwefel function, *Nearest* and

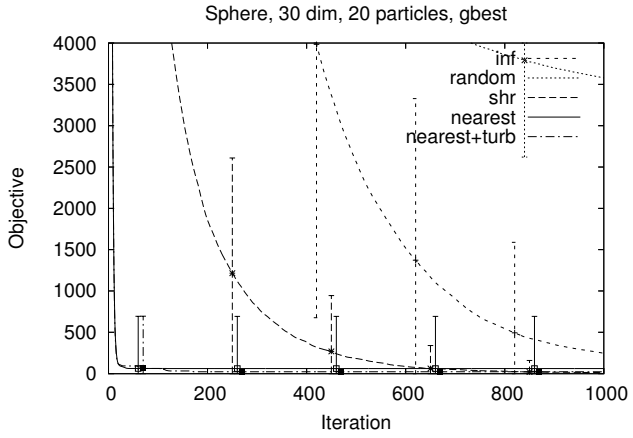

Fig. 4. In unbalanced search spaces, e.g. using $x_i \in [-1..100], \forall i = 1..n$ for the Sphere function, the effects of the bound handling method are totally different than in balanced search spaces (see Fig. 3). Here, applying *random* and *inf* resulted in very slow convergence.

Nearest+Turb lead to outstandingly good results (see Fig. 5). Moreover, Alvarez-Benitez et al. [8] demonstrated that *Shr*, which also is a bound resetting strategy, on average performed best in their experiments. Thus, we definitely cannot conclude that bound resetting generally is a bad idea. It depends on the concrete structure of the optimization problem which bound handling mechanism works best.

Our analysis of particle swarm optimization in high-dimensional search spaces has shown that the bound handling mechanism essentially influences the swarm behavior. We provided first steps towards explaining why particles are in danger of converging on the boundary. Based on our theoretical analysis, we now can understand some of the experimental observations mentioned earlier [7], [9], [10], and we provided deep insights into high-dimensional particle swarm optimization. Our broad experimental study of commonly applied bound handling mechanisms is useful for choosing an appropriate strategy in real world applications. Some hints for practical application of the PSO algorithm will be discussed in the next section.

V. CONCLUSION

In this paper, we presented both theoretical and experimental results demonstrating that the bound handling mechanism is

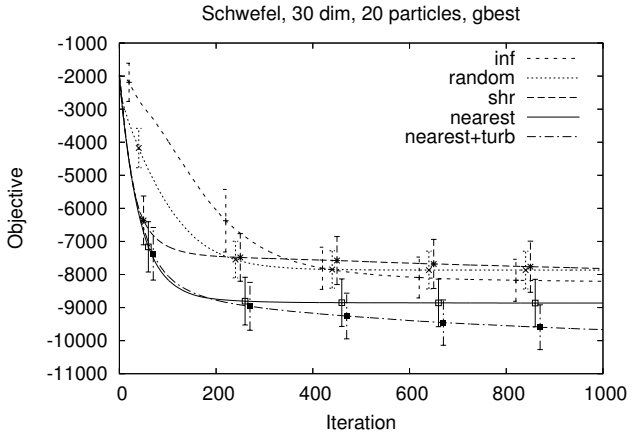

Fig. 5. Applying the bound handling method *Nearest+Turb* or *Nearest* lead to remarkably good solutions for the Schwefel function.

an important feature of the PSO algorithm in high-dimensional search spaces, and it should therefore be chosen carefully. Our theoretical analysis about high-dimensional bounded search spaces gave valuable insight for practical particle swarm application. Our experiments have shown that it strongly depends on the optimization problem which bound handling strategy performs best. In real world applications, often a priori knowledge about the optimization problem is available, and can be exploited to choose an adequate bound handling mechanism. From our analysis, we can give the following guidelines for applying particle swarm optimization in high-dimensional bounded search spaces:

- If the optimum is expected to lie near or on the search space boundary, bound handling mechanisms like *Random* or *Inf* distract particles from the boundary, and therefore lead to slow convergence (often too slow for practical applications where the evaluation of the objective function is very expensive). This observation is illustrated in Fig. 4 and Fig. 5.
- If the optimum is expected to lie near the center of the search space, better results can be obtained if the particles' positions and velocities are not initialized with uniform distribution since particles are then located very close to the boundary (see Theorem 3.1), and at least the global guide leaves the search space with overwhelming probability (see Corollary 3.3). Initialization with Gaussian distribution or the calculation of Voronoi diagrams, as was proposed by Richards and Ventura [10], might lead to better results.
- If the optimum is expected to lie near the center of the search space, bound handling mechanisms which reset particles on the boundary, such as *Nearest* or *Shr*, should not be used as particles might converge on the boundary. This phenomenon has already been noticed experimentally by Zhang et al. [9]. Our theoretical analysis provides first results for understanding this observation: We have shown in Theorem 3.1 that particles are initialized very close to the boundary w.o.p., and proven in Theorem 3.2 that the global guide is expected to be set on $\frac{1}{4}n$ boundaries when using the *Nearest* method. Premature convergence on the boundary has been especially surprising when optimizing a simple function like the Sphere benchmark. However, Theorems 3.4 and 3.5 reveal that boundary solutions easily outperform the initialization values. Moreover, we have shown in our experiments that even turbulence or initializing the velocities to zero often cannot prevent premature convergence.

REFERENCES

[1] James Kennedy and Russell C. Eberhart. *Swarm Intelligence*. Morgan Kaufmann Academic Press, 2001.

[2] Jonathan E. Fieldsend and Sameer Singh. A Multi-Objective Algorithm based upon Particle Swarm Optimization, an Efficient Data Structure and Turbulence. In *Proceedings of the 2002 U.K. Workshop on Computational Intelligence*, pages 37–44, 2002.

[3] Maurice Clerc and James Kennedy. The Particle Swarm - Explosion, Stability, and Convergence in a Multidimensional Complex Space. *IEEE Transactions on Evolutionary Computation*, 6(1):58–73, 2002.

[4] Ioan Cristian Trelea. The particle swarm optimization algorithm: convergence analysis and parameter selection. *Information Processing Letters*, 85(6):317–325, 2003.

[5] Frans van den Bergh. *An analysis of particle swarm optimizers*. PhD thesis, University of Pretoria, 2002.

[6] F. van den Bergh and A.P. Engelbrecht. A study of particle swarm optimization particle trajectories. *Information Sciences*, 176(8):937–971, 2006.

[7] Rui Mendes. *Population Topologies and Their Influence in Particle Swarm Performance*. PhD thesis, Departamento de Informática, Escola de Engenharia, Universidade do Minho, 2004.

[8] Julio E. Alvarez-Benitez, Richard M. Everson, and Jonathan E. Fieldsend. A MOPSO Algorithm Based Exclusively on Pareto Dominance Concepts. In *Evolutionlary Multio-Criterion Optimization (EMO 2005)*, pages 459–473, 2005.

[9] Wen-Jun Zhang, Xiao-Feng Xie, and De-Chun Bi. Handling Boundary Constraints for Numerical Optimization by Particle Swarm Flying in Periodic Search Space. In *Proceedings of the 2004 Congress on Evolutionary Computation*, volume 2, pages 2307–2311, 2004.

[10] Mark Richards and Dan Ventura. Choosing a Starting Configuration for Particle Swarm Optimization. In *Proceedings of the International Joint Conference on Neural Networks*, volume 3, pages 2309–2312, 2004.

[11] Zbigniew Michalewicz and Marc Schoenauer. Evolutionary Algorithms for Constrained Parameter Optimization Problems. *Evolutionary Computation*, 4(1):1–32, 1996.

[12] Gregorio Toscano Pulido and Carlos A. Coello Coello. A Constraint-Handling Mechanism for Particle Swarm Optimization. In *Proceedings of the 2004 Congress on Evolutionary Computation*, volume 2, pages 1396–1403, 2004.

[13] Jiyuan An, Yi-Ping P. Chen, Qinying Xu, and Xiaofang Zhou. A New Indexing Method for High Dimensional Dataset. In *DASFAA*, volume 3453, pages 385–397. Springer, 2005.

[14] Donald E. Knuth. *The Art of Computer Programming, Volume 1*. Addison Wesley, 3rd edition, 1997.

[15] David H. Wolpert and William G. Macready. No Free Lunch Theorems for Optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82, April 1997.

[16] Jens Jägersküpper. Analysis of a Simple Evolutionary Algorithm for the Minimization in Euclidian Spaces. Technical Report CI 140/02, SFB 531, Universität Dortmund, 2002.

[17] Russell C. Eberhart and Yuhui Shi. Comparing Inertia Weights and Constriction Factors in Particle Swarm Optimization. In *Proceedings of the 2000 Congress on Evolutionary Computation*, pages 84–88, 2000.

[18] Edwin Peer, Gary Pampara, and Andries Engelbrecht. Cilib - Computational Intelligence Library. http://cilib.sourceforge.net, 2006.