An alternative approach for particle swarm optimisation using serendipity

Fábio Augusto Procópio Paiva*

Campus Parnamirim,
Federal Institute of Rio Grande do Norte, Brazil
Email: fabio.procopio@ifrn.edu.br
*Corresponding author

José Alfredo Ferreira Costa

Department of Electrical Engineering,
Federal University of Rio Grande do Norte, Brazil
Email: jafcosta@gmail.com

Cláudio Rodrigues Muniz Silva

Department of Communications Engineering,
Federal University of Rio Grande do Norte, Brazil
Email: claudio.rmsilva@gmail.com

Abstract: In the study of metaheuristic techniques, it is very common to deal with a problem known as premature convergence. This problem is widely studied in swarm intelligence algorithms such as particle swarm optimisation (PSO). Most approaches to the problem consider the generation and/or positioning of individuals in the search space randomly. This paper approaches the issue using the concept of serendipity and its adaptation in this new context. Several strategies that implement serendipity were evaluated in order to develop a PSO variant based on this concept. The results were compared with the traditional PSO considering the quality of the solutions and the ability to find global optimum. The new algorithm was also compared with a PSO variant of the literature. The experiments showed promising results related to the criteria mentioned above, but there is the need for additional adjustments to decrease the runtime.

Keywords: particle swarm optimisation; PSO; SBPSO; serendipity; swarm intelligence; global optimisation; bio-inspired computation; metaheuristic.

Reference to this paper should be made as follows: Paiva, F.A.P., Costa, J.A.F. and Silva, C.R.M. (2018) 'An alternative approach for particle swarm optimisation using serendipity', *Int. J. Bio-Inspired Computation*, Vol. 11, No. 2, pp.81–90.

Biographical notes: Fábio Augusto Procópio Paiva received his DSc in Electrical and Computer Engineering from Federal University of Rio Grande do Norte (UFRN) in 2016. Currently, he is Professor at the Federal Institute of Rio Grande do Norte. His current research interests include recommender systems and computational intelligence.

José Alfredo Ferreira Costa received his DSc in Electrical and Computer Engineering from Universidade Estadual de Campinas (Unicamp) in 1999. He is an Associate Professor at UFRN, Department of Electrical Engineering. His research interests include neural networks and self-organising systems and data mining.

Cláudio Rodrigues Muniz Silva received his DSc in Electrical and Computer Engineering from Universidade Estadual de Campinas (Unicamp) in 2003. He is currently an Associate Professor at Communication Engineering Department of UFRN. His research interests include computational intelligence and internet of things (IoT).

This paper is a revised and expanded version of a paper entitled 'Uma meta-heurística alternativa de inteligência de enxames baseada em serendipidade guiada' presented at Brazilian Congress on Computational Intelligence, Brazil, 13–16 October 2015.

1 Introduction

Several bio-inspired techniques based on swarm intelligence have been applied to real world optimisation problems in different domains areas such as telecommunications (Ram et al., 2014; Xu, 2015; Boudjemaa, 2015), power systems (Esmin et al., 2005; Singh and Swarnkar, 2011), prediction systems (Chen et al., 2009; Jin and Huang, 2012) and automation (Chaudhary et al., 2015; Wang et al., 2015).

Swarm intelligence is a widespread concept in bio-inspired computing and involves a set of metaheuristics whose emergent behaviours can result in an ability to solve complex problems (Silva and Bastos Filho, 2015). It implements the collective intelligence for groups containing simple agents that are based on the behaviour of natural insects.

Particle swarm optimisation (PSO) (Kennedy and Eberhart, 1995), for example, is one of the swarm intelligence algorithms that has been widely used in the context of global optimisation problems. PSO is inspired by the social behaviour of birds and fishes, where individuals of a population are represented by a point (or particle) of the search space. According to Del Valle et al. (2008), generally the algorithm is not largely affected by the size and nonlinearity of the problem and, moreover, it converges to optimal solutions, whereas most analytical methods fail during the convergence process.

PSO has some advantages such as easy implementation and fast convergence, but often it faces a problem in which its particles are ‘trapped’ in local optimum. This problem is often known as premature convergence. In another area of study, known as Recommender Systems, there is a concept that can be used as a strategy to deal with premature convergence. This concept is called *serendipity*.

In general, serendipity is a term that refers to fortunate findings that were apparently performed by chance. In an exhaustive study, Andel (1994) showed that serendipity has a great contribution to the progress of science, technology and art. According to Kuhn (1970), in science and technology, the serendipitous discoveries frequently occur. In the history of science, for example, there are several cases of serendipity such as:

1 vaseline in 1859

2 saccharin in 1878

3 X-ray in 1895

4 radioactivity in 1896

5 penicillin in 1928

6 microwave oven in 1945 and many others.

Recommender systems emerged as an approach to solve the problem of information overload. They are special applications, present in our daily lives, that provide personalised recommendations (for example, products or services) for users that may be interested (Paiva et al., 2014). In recent years, the researches in recommender systems have focused the accuracy of the recommendations (Iaquinta et al., 2009; Ge et al., 2010; Adamopoulos and Tuzhilin, 2011). However, high accuracy can cause a specific problem known as over-specialisation. This problem occurs when the system refers only to the items related with the user’s profile, there may be items which are more appropriate. Serendipity is a strategy that may be used to solve the problem of over-specialisation.

According to Paiva et al. (2015), when a recommendation system suggests only items related to the user’s interest, it converges to recommendations that do not meet the user’s expectations. Thus, it fails to suggest items that can be most adequate. Similarly, when a metaheuristic algorithm converges to a local optimum and does not consider other solutions that are more appropriate than the solution found, the similarity between over-specialisation and premature convergence is observed.

This paper proposes the implementation of the concept of serendipity to develop a variant of the traditional PSO algorithm called serendipity-based PSO. A prototype is developed as a conceptual proof of the viability of using serendipity in the context of metaheuristic algorithms. Some benchmark functions (unimodal and multimodal) were used to evaluate the prototype. The results showed that it outperformed traditional PSO considering the quality of the solutions and the ability to find global optimum, although there is a need to reduce its running time.

The paper is organised as follows. In Section 2, it is provided a brief background in order to introduce the basic concepts necessary for the reader’s understanding. Section 3 presents the proposed algorithm that implements the concept of serendipity in the metaheuristic context. Section 4 presents the experiments performed and discusses the results obtained. Finally, in Section 5, the final considerations are presented to conclude the work.

2 Theoretical foundation

This section presents some basic information necessary to the suitable understanding of the algorithm presented in Section 3. The concept of serendipity is presented together with some related work available in the literature.

2.1 Serendipity

What exists in common among the X-ray, nylon and vaccine? One possible answer is that they were discovered by chance (Campos and Figueiredo, 2002), or serendipitously.

In 1754, the novelist Horace Walpole wrote a letter addressed to a friend narrating a Persian story called *The Three Princes of Serendip*. The princes were observers and sagacious. As a result, they discovered solutions to some dilemmas by chance. Therefore, Walpole coined the term serendipity with purpose of expressing discoveries happen by chance.

The Cambridge Dictionary defines serendipity as *the fact of finding interesting or valuable things by chance*.

However, unlike many traditional definitions that just use the term as a synonymous of 'random', serendipity is closer to a mixture of sagacity and chance (Catellin, 2014).

Campos and Figueiredo (2002) were among the first researchers to express the concept of serendipity in a formal way through the identification of different categories. For this, they used logic equations called *serendipity equations* to present four events that can generate serendipity. These events are associated to different types of serendipity:

a pseudo-serendipity

b real serendipity

c serendipity without a metaphor inspiration

d serendipity with incorrect knowledge.

In recommender systems, serendipity can be implemented through four strategies (Toms, 2000):

1 blind luck

2 Pasteur's principle ('chance favours the prepared mind')

3 anomalies and exceptions

4 reasoning by analogy.

The *blind luck* approach aims to provide recommendations from information randomly generated. 'Pasteur's principle' is implemented by means of the user's profile to recommend items that have something to do with user's profile. The *anomalies and exceptions* approach is partially implemented by low similarities between the user's profile and the recommended items. Finally, the implementation of the 'reasoning by analogy' approach is unknown at the moment.

In the metaheuristic context, it is said that the set F is formed by elements that represent the solutions found during an iteration in a given search space. Thus, we say that f_{ij} is an element belonging to the set F that represents a solution i found during the iteration j. The solution i whose fitness value is the best value during the iteration j, is represented by the element $f^*_{ij} \in F$.

S is a set whose elements represent the possible solutions in a search space and F is a proper subset of S. So during the iteration j, when an element of S does not belong to the set F, this element is considered an *occasional* solution. Then, the set containing the elements that represent occasional solutions during the iteration j is given by the relative complement of F in S, according to equation (1):

$$CHANCE = S - F \tag{1}$$

When an element $chance_{ij}$ presents a fitness value better than the fitness presented by the element f^*_{ij}, $chance_{ij}$ is called *serendipitous*. The set SRD is formed by serendipitous elements that represent occasional solutions with fitness value better than the fitness value of the element f^*_{ij}, according to equation (2):

$$SRD = \{srd_{ij} | fitness(chance_{ij}) < fitness(f^*_{ij})\} \tag{2}$$

The concept of serendipity presented in this paper is adapted to the metaheuristic context. It consists of two essential aspects: *acceptability* and *occasionality*. So, an element of the set SRD can be used to represent a serendipitous solution because it is

1 *acceptable*, since it represents a solution whose fitness value is better than fitness value of the element f^*_{ij}

2 *occasional*, since it is an element that does not belong to the set F during the iteration j.

2.2 Related work

Serendipity is defined as finding something good or useful while not specifically searching for that in a certain occasion. When the concept of serendipity is adapted to the context of metaheuristic algorithms, 'something good or useful' may be seen as a candidate solution that it is better than the current one after a certain number of iterations.

A class of metaheuristic algorithms that has received much attention consists of swarm intelligence algorithms. They deal with natural and artificial systems composed of populations that coordinate using decentralised control and self-organisation (Zhang et al., 2014). The study of swarm intelligence involves the behaviour of ants, wasps, termites, fishes, birds and bat groups. In this context, PSO is one of the well-known algorithms.

PSO has gained much attention from research communities in real world optimisation problems due to its ease of implementation and good performance (Shim et al., 2013). PSO is a stochastic approach to model the social behaviour of birds. In this kind of algorithm, the solution is represented by a particle that 'flies' over search space looking for an optimum solution during a certain number of iterations (Parpinelli and Lopes, 2011; Souza et al., 2012).

The movement of the particles is based on two pieces of information: *gBest* and *pBest*. Information *gBest* influences the motion of the particle towards the best position found by the swarm whereas *pBest* moves the particle to the best position found by itself. After finding *gBest* and *pBest*, each particle updates its velocity and its position by equations (3) and (4), respectively:

$$\begin{aligned} v_{id}^{(t+1)} &= w \cdot v_{id}^t + c_1 \cdot r_1(p_{id}^t - x_{id}^t) \\ &\quad + c_2 \cdot r_2(p_{gd}^t - x_{id}^t) \end{aligned} \tag{3}$$

$$x_{id}^{(t+1)} = x_{id}^t + v_{id}^{(t+1)} \tag{4}$$

In equation (3), the term w is called inercial factor and represents the particle inertial velocity. v_{id}^t and x_{id}^t are the velocity and position of the particle i in the instant t, respectively. p_{id}^t and p_{gd}^t are the best fitness values found by particle i and the best fitness value found among all swarm particles until the instant t, respectively. c_1 is a coeficient that contributes to the auto exploration of the particle, whereas c_2 contributes with the particle movement towards a swarm global dislocation in the search space. r_1

and r_2 are random values uniformly distributed in the range [0, 1].

Different variant approaches have emerged to improve the performance and convergence behaviour for PSO algorithm. Wang et al. (2007) proposed the opposition-based PSO (OPSO) method to accelerate the convergence of PSO and avoid premature convergence. OPSO employs opposition-based learning and applies dynamic Cauchy Mutation on the best particle of the swarm. The experiments showed that OPSO could successfully deal with multimodal benchmark functions while maintaining fast search speed on simple unimodal functions.

In Ling et al. (2008), it is proposed a PSO variant that combines mutation operator with wavelet theory. This theory increases the operating space to find best solutions. Benchmark functions and three industrial applications were employed to evaluate the performance and applicability of the proposed method.

In Li et al. (2011), it is presented a hybridisation that combines PSO with a modified method of Broyden-Fletcher-Goldfarb-Shanno (BFGS). The modified BFGS method is integrated in the PSO context to improve the ability of local searches. Two techniques are combined to maintain the diversity. They are: reposition and territory.

In Esmin and Matwin (2013), it is presented a method that combines the mutation process, commonly used in the implementation of genetic algorithms, with the traditional PSO. The presented algorithm is called hybrid PSO with genetic mutation (HPSOM) and its particles have a probability of mutating at each iteration. The experiments showed that HPSOM was successful when evaluated on unimodal e multimodal functions.

In the literature, PSO variants that use the concept of serendipity were not found, especially with the purpose to deal with convergence premature.

3 Proposal: serendipity-based PSO

Serendipity was presented as a ability to perform fortunate discoveries by chance and sagacity. Here, the term 'chance' should be understood as a possibility of anything happens. 'Sagacity' should be understood as the quality of having good perception about an event (or something related). The concept of serendipity is useful since it is the main hint to define where serendipity could be applied to a traditional PSO algorithm and how the sagacity should be used to improve PSO behaviour with serendipity-inspired decisions.

This work adopts a proposed variant by Paiva et al. (2015) to generate serendipity based on a perceptive model (Lawley, 2013) that combines two strategies generally used in the domain of recommender systems (Toms, 2000). They are: blind luck and Pasteur's principle.

The *blind luck* strategy is used to implement the chance dimension. For this, a swarm particle is randomly chosen and a neighbour is generated. The *Pasteur's principle* is used to 'recognise a potential' and 'seize the moment'. *Recognise a potential* is the identification of a solution better than current *gBest* particle. *Seize the moment* is related to the replacement of the current *gBest* particle by a best solution. Pasteur's principle is used to implement the concept of sagacity (or 'prepared mind').

The proposed algorithm is called serendipity-based particle swarm optimisation (SBPSO) and its pseudocode is presented in Algorithm 1. The use of serendipity aims identify solutions better than current $gBest$ particle, but has not yet been identified. The new solutions can be represented as elements of the set SRD, defined in Subsection 2.1.

Algorithm 1 SBPSO pseudocode

```
1:  for each particle i in S do
2:    randomly initialise velocity and position
3:  end for
4:  while stopping criterion not satisfied do
5:    for each particle i in S do
6:      calculate fitness value
7:      if fitness(x_id) < fitness(pBest_i) then
8:        pBest_i ← x_id
9:      end if
10:     if fitness(pBest_i) < fitness(gBest) then
11:       gBest ← pBest_i
12:     end if
13:   end for
14:
15:   /*Implementation of serendipity*/
16:   repeat
17:     SRD ← null
18:     /*Chance dimension*/
19:     create one AP to raffled particle [equation (5)]
20:     if fitness(AP) < fitness(gBest) then
21:       gBest ← AP
22:       SRD ← append(AP)
23:     end if
24:     /*Sagacity dimension*/
25:     create n AP to gBest [equation (5)]
26:     for each AP do
27:       if fitness(AP) < fitness(gBest) then
28:         gBest ← AP
29:         SRD ← append(AP)
30:       end if
31:     end for
32:   until SRD = null
33:
34:   for each particle i in S do
35:     calculate particle velocity according to
        equation (3)
36:     update particle position according to equation (4)
37:   end for
38: end while
```

In Algorithm 1, lines 16 to 32 present the strategy to implement the concept of serendipity by means of chance and sagacity dimension. Chance dimension is implemented raffling a particle (other than the *gBest*) to create a neighbour solution. Sagacity dimension, in turn, is implemented creating n neighbours around to *gBest* particle. Both dimensions use equation (5) to create neighbours that are called adjacent points (AP):

$$AP = p + \alpha \cdot R \tag{5}$$

where p is a swarm particle, α is a positive real constant, R is a matrix $1 \times dim$ that contains random values uniformly distributed in the range [–1, 1] and dim is the dimension of the particle p.

The initial steps of the SBPSO algorithm are similar to the traditional PSO. The velocity and position are randomly initialised for each particle (lines 1–3). Then, for each particle, the fitness value is calculated to find the *gBest* particle (lines 5–13).

The strategy to implement serendipity is initiated when the *gBest* particle is found. A particle is randomly chosen and a neighbour solution (or adjacent point, AP) is created. The fitness value of the AP is calculated and compared to fitness value of the *gBest* particle. If the value is better then the AP becomes the new *gBest* particle and it is appended to set SRD. In the next step, n AP are created around to *gBest* particle. For each AP, the fitness value is calculated and compared to the *gBest* particle. When the fitness value of the AP is better than *gBest* particle fitness, the AP becomes the new *gBest* particle and it is appended to set SRD. In the current iteraction, the procedure is performed while there are AP better than *gBest* particle. Then, the normal flow of the traditional PSO is continued with the calculations of the positions and velocities of the particles.

The choice of a value to the parameter n (that defines the number of AP around to *gBest* particle) is essential for a good performance of the algorithm since its runtime can be compromised if a high value is assigned to this parameter. In the next section, the value for this parameter will be discussed.

4 Experiments and results

In order to evaluate the proposed algorithm, six benchmark functions were chosen. All of them are associated to high-dimensional problems on which a minimisation process is applied. These functions have been used in several studies of PSO such as Ling et al. (2008), Murthy et al. (2009), Li et al. (2011), Esmin and Matwin (2013), Cuevas et al. (2015) and other. The functions are described below:

- Spherical function – it has no interaction between its variables. It is very simple, convex and unimodal:

$$f_1(x) = \sum_{i=1}^{d} x_i^2$$

- Rosenbrock function – it has interaction between some variables, whose global minimum is in a parabolic valley. Although it is easy to find the parabolic valley, the convergence to the minimum is difficult (Picheny et al., 2013):

$$f_2(x) = \sum_{i=1}^{d-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$$

- Griewank function – it has interactions between variables. The function has many widespread local minimum that are regularly distributed:

$$f_3(x) = \frac{1}{400} \sum_{i=1}^{d} x_i^2 - \prod cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$$

- Rastrigin function – it has several local minima and the locations of the minimum are regularly distributed:

$$f_4(x) = \sum_{i=1}^{d} [x_i^2 - 10cos(2\pi x_i) + 10]$$

- Ackley function – it has a large funnel shape and its surface has many local minima:

$$f_5(x) = -20\ exp\left(-0.2\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2}\right) - exp\left(\frac{1}{d}\sum_{i=1}^{d}\cos(2\pi x_i)\right) + 20 + exp(1)$$

- Noisy Quartic function – it is a simple unimodal function padded with gaussian noise:

$$f_6(x) = \sum_{i=1}^{d} ix_i^4 + rand(0, 1)$$

For each benchmark function, Table 1 presents the boundaries of the search space and the initialisation range.

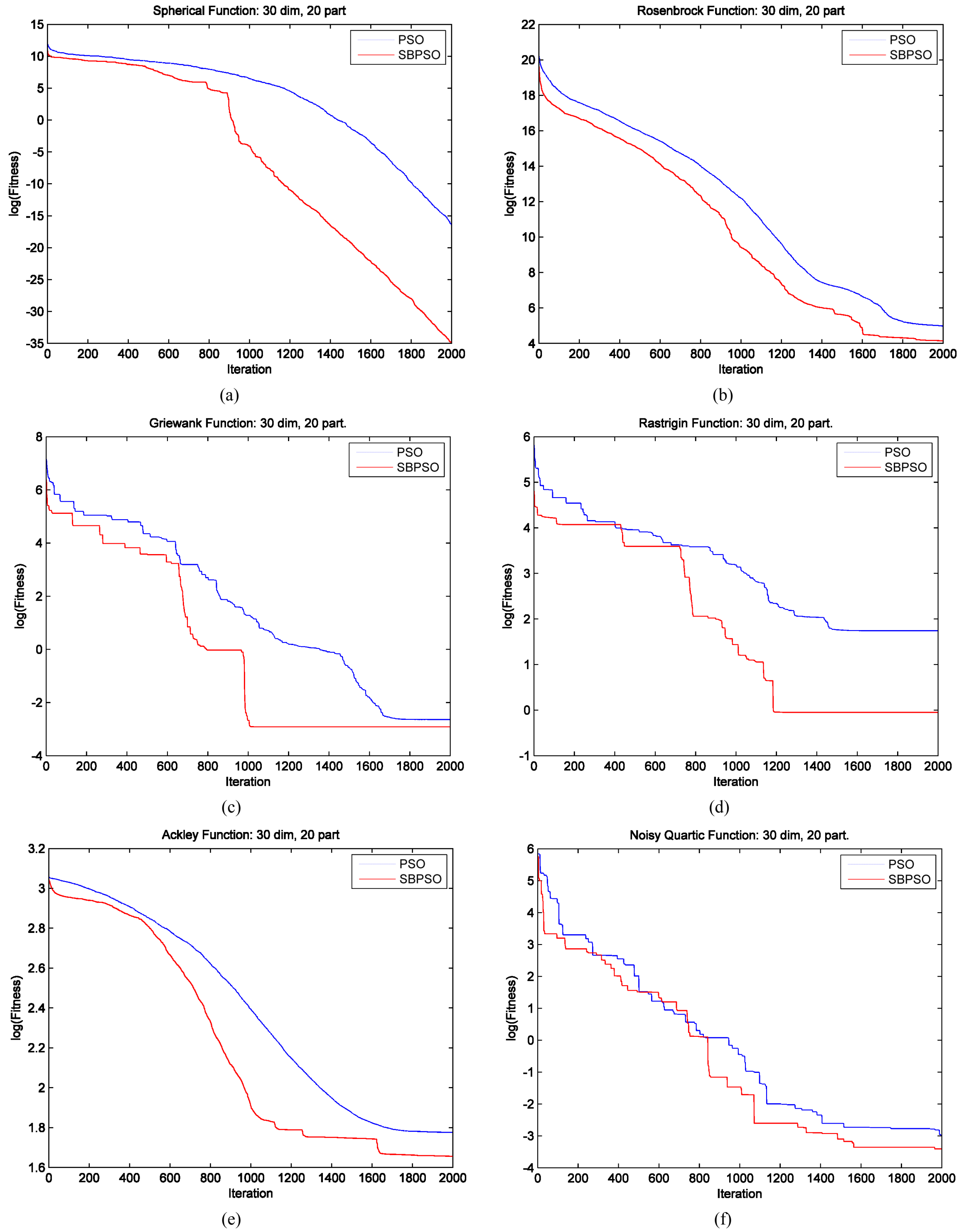
The experiments compared PSO and SBPSO using various configurations to vary the population size, the dimensionality of the functions and the number of iterations. For each benchmark function, the population was set at 20, 40 and 80 and the number of iterations was set at 1,000, 1,500 and 2,000, that corresponding to dimensions 10, 20 and 30, respectively. All experiments were executed 100 times.

Table 1 Boundaries of the benchmark functions

Fun	*Search space*	*Initialisation range*
f_1	$-100 \leq x_i \leq 100$	$50 \leq x_i \leq 100$
f_2	$-30 \leq x_i \leq 30$	$15 \leq x_i \leq 30$
f_3	$-600 \leq x_i \leq 600$	$300 \leq x_i \leq 600$
f_4	$-5.12 \leq x_i \leq 5.12$	$2.56 \leq x_i \leq 5.12$
f_5	$-32.768 \leq x_i \leq 32.768$	$16.384 \leq x_i \leq 32.768$
f_6	$-1.28 \leq x_i \leq 1.28$	$0.64 \leq x_i \leq 1.28$

In order to have a fair comparison, all common parameters of PSO and SBPSO were set to the same value: $c_1 = c_2 = 2.0$ and an inertia weight w that decreases linearly starting at 0.9 and ending at 0.4. The maximum velocity (V_{max}) of each particle was set to be half the length of the search space in one dimension (Zhang and Xie, 2003).

Figure 1 Performance comparison between PSO and SBPSO for each benchmark function (see online version for colours)



Note: It is observed that the SBPSO convergence speed is greater than traditional PSO one.

In equation (5), the parameter α was set to be 1% the length of the search space in one dimension. This equation may be used to create adjacent points to *gBest* particle. For each benchmark function, several experiments were conducted in order to find an appropriate value for n. Based on a set of predefined settings (population size, dimensionality of the functions and number of iterations), the best mean and standard deviation were defined as criteria to evaluate the best value n. At the end of the experiments, $n = 5$ was the value that showed better results in the criteria evaluated. When values greater than 5 were attributed to n, significant improvements in value of the best mean were not found. In addition, the average runtime of the algorithm was compromised.

For each benchmark function, Figure 1 shows the average behaviour of PSO and SBPSO on 100 executions. Tables 2 and 3 compare PSO and SBPSO when they were evaluated for functions f_1, f_2, f_3, f_4, f_5 and f_6. The tables show the function, the population size, the dimensionality of the function, the number of iterations, the best mean and the standard deviation. The last column shows the *p-value* obtained from the Wilcoxon Test.

Wilcoxon Test (Demšar, 2006) was applied with significance level 0.05 to evaluate the solutions found by PSO and SBPSO. Wilcoxon is a non-parametric statistical test used to compare two independent samples. The null hypothesis H_0 indicates that two samples come from the same population, whereas the alternative hypothesis H_1 indicates that one has higher values than the other. When *p-value* is less than the significance level then it decides to reject H_0, i.e., there is significant difference between the samples.

In 100% of the experiments, it is observed that the SBPSO outperformed PSO, according to Tables 2 and 3. Nevertheless, the experiments for function Griewank (f_3) with 40 particles, 20 dimensions, 1,500 iterations and 80 particles, 20 dimensions, 1,500 iterations *p-values* (they are highlighted with *) were not less than 0.05 (significance level). As a consequence, the null hypothesis H_0 was not rejected. Although the SBPSO performance has been better than the PSO one, this means that the results do not come from different populations and therefore they are not significant statistically.

However, to ensure statistical significance in all the experiments, the configuration for these two experiments were changed. So, the number of particles was increased by 10% of the initial value. Then, as a result, it was observed that the *p-values* were respectively 4.7533E–02 and 2.5618E–02. Therefore, with this additional adjustment, in 100% of the experiments, SBPSO statistically showed significant results: *p-value* < 0.05.

Table 4 compares traditional PSO, SBPSO and a variant called HWPSO (Ling et al., 2008). The functions used to compare the algorithms were Spherical (f_1), Rosenbrock (f_2), Griewank (f_3) and Rastrigin (f_4) whose dimensions (10, 20 and 30) and the number of iterations (1,000, 1,500 and 2,000) were varied. Only the population size was fixed to 20 particles. The mean best fitness and standard deviation obtained by the algorithms were recorded. It was observed that HWPSO outperformed SBPSO when the Spherical function was evaluated. However, the performance of SBPSO outperformed HWPSO when functions Rosenbrock, Griewank and Rastrigin were evaluated.

Table 2 Comparison between PSO and SBPSO for functions Spherical (f_1), Rosenbrock (f_2) and Griewank (f_3)

Function	*Part*	*Dim*	*Iter*	*PSO*		*SBPSO*		*p-value*
				Best mean	*Standard deviation*	*Best mean*	*Standard deviation*	
f_1	20	10	1,000	1.2368E–20	3.1403E–20	7.6441e–27	1.7221E–26	3.5416E–30
		20	1,500	2.9396E–11	1.8370E–10	5.4127E–20	9.5036E–20	4.5648E–30
		30	2,000	4.6804E–08	1.3386E–07	7.2297E–16	1.0016E–15	2.1174E–30
	40	10	1,000	2.2365E–24	1.3369E–23	6.8652E–32	8.2562E–32	1.9531E–30
		20	1,500	8.1334E–15	2.5881E–14	2.4116E–22	6.7894E–22	4.7831E–30
		30	2,000	8.0544E–11	1.3452E–10	1.3991E–18	2.1623E–18	2.8749E–30
	80	10	1,000	1.5394E–28	4.8125E–28	1.7458E–35	3.2779E–35	7.7831E–30
		20	1,500	5.1700E–18	1.4221E–17	4.0153E–26	5.1204E–26	6.8431E–30
		30	2,000	1.4817E–13	2.9377E–13	1.0514E–20	1.3133E–20	2.6721E–30
f_2	20	10	1,000	58.3417	133.7896	17.2359	0.3171	6.0041E–07
		20	1,500	104.9516	162.9876	33.6759	0.7352	2.8809E–05
		30	2,000	151.5238	239.0893	63.2329	0.8647	1.0251E–06
	40	10	1,000	30.80349	114.8610	3.0525	0.4582	4.4074E–14
		20	1,500	81.5949	141.4203	13.6073	0.6328	9.8240E–14
		30	2,000	132.6704	204.0919	43.6027	0.7854	4.3609E–10
	80	10	1,000	20.5744	43.8337	15.1468	0.6653	1.2180E–13
		20	1,500	65.7612	105.0775	50.3721	0.5982	3.5119E–10
		30	2,000	86.87974	123.5531	79.6384	0.4327	5.9386E–04
f_3	20	10	1,000	0.1012	0.0516	0.0829	0.0436	4.7702E–02
		20	1,500	0.0334	0.0336	0.0284	0.0295	3.1776E–05
		30	2,000	0.0146	0.0171	0.0098	0.0127	7.1333E–03
	40	10	1,000	0.0845	0.0438	0.7557	0.3572	4.6468E–02
		20	1,500	0.0276	0.0291	0.0247	0.0414	5.4982E–02*
		30	2,000	0.0592	0.1690	0.0235	0.0225	8.0848E–05
	80	10	1,000	0.0772	0.0396	0.0649	0.0274	4.9053E–02
		20	1,500	0.0357	0.0325	0.0232	0.0209	5.3169E–02*
		30	2,000	0.0127	0.0133	0.0103	0.0163	1.1977E–03

Table 3 Comparison between PSO and SBPSO for functions Rastrigin (f_4), Ackley (f_5) and Noisy Quartic (f_6)

Function	*Part*	*Dim*	*Iter*	*PSO*		*SBPSO*		*p-value*
				Best mean	*Standard deviation*	*Best mean*	*Standard deviation*	
f_4	20	10	1,000	4.5782	2.1132	4.3285	2.7361	2.7832E–07
		20	1,500	22.8061	10.0912	17.7843	7.7291	4.8954E–06
		30	2,000	49.7192	13.7956	43.5283	12.7813	7.0931E–06
	40	10	1,000	3.9224	1.5118	2.7192	4.7218	8.1023E–07
		20	1,500	16.1082	1.5722	14.4542	6.8310	5.8930E–06
		30	2,000	38.6344	2.2039	35.8212	13.7321	4.0027E–06
	80	10	1,000	2.2854	0.4780	1.7721	3.9831	5.3179E–07
		20	1,500	12.3630	2.2804	9.7831	1.7035	3.8943E–06
		30	2,000	31.5714	6.2924	27.7831	11.6492	4.9836E–06
f_5	20	10	1,000	0.40498	2.8493	3.8831E–14	6.2307E–14	1.5777E–30
		20	1,500	1.8615	5.8614	0.9933	2.3520	2.4173E–15
		30	2,000	5.9035	9.0652	1.2317	1.8429	2.0970E–12
	40	10	1,000	3.6913E–13	4.966E–13	5.5067E–15	1.6363E–15	3.1554E–30
		20	1,500	2.029	6.1178	0.9994	0.3784	5.8996E–17
		30	2,000	3.4461	7.6525	1.8738	1.0294	1.7269E–12
	80	10	1,000	7.7449E–15	4.0477E–15	4.5119E–15	4.9989E–16	2.9143E–16
		20	1,500	0.61094	3.4916	0.5959	3.4057	1.8501E–18
		30	2,000	3.8243	7.9365	1.7716	1.6619	2.5803E–12
f_6	20	10	1,000	8.2418E–03	4.8127E–02	7.3593E–03	7.7831E–02	9.6734E–06
		20	1,500	0.0253	0.0163	0.01082	0.0157	2.7812E–05
		30	2,000	0.5030	0.0372	0.0330	0.0284	5.8931E–05
	40	10	1,000	3.4688E–03	4.4512E–02	1.7882E–03	2.7123E–02	9.7827E–06
		20	1,500	0.0173	0.0187	0.0134	0.0165	8.0914E–05
		30	2,000	0.0493	0.0294	0.0343	0.0261	4.8120E–05
	80	10	1,000	3.1784E–03	6.7841E–02	1.3653E–03	3.7620E–02	5.7619E–06
		20	1,500	0.0127	0.0143	0.0093	0.0120	8.1002E–05
		30	2,000	0.0460	0.0451	0.0169	0.0173	6.9183E–05

Table 4 Comparison between PSO, SBPSO and HWPSO

Function	*Dim*	*Iter*	*PSO*		*HWPSO*		*SBPSO*	
			Best mean	*Standard deviation*	*Best mean*	*Standard deviation*	*Best mean*	*Standard deviation*
f_1	10	1,000	1.2368E–20	3.1403E–20	6.2868E–56	1.506E–55	7.6441E–27	1.7221E–26
	20	1,500	2.9396E–11	1.8370E–10	6.2830E–45	2.033E–44	5.4127E–20	9.5036E–20
	30	2,000	4.6804E–08	1.3386E–07	3.7940E–36	1.406E–35	7.2297E–16	1.0016E–15
f_2	10	1,000	58.3417	133.7896	36.4736	0.1844	17.2359	0.3171
	20	1,500	104.9516	162.9876	65.6678	0.5870	33.6759	0.7352
	30	2,000	151.5238	239.0893	70.7275	0.4813	63.2329	0.8647
f_3	10	1,000	0.1012	0.0516	0.1333	0.3399	0.0829	0.0436
	20	1,500	0.0334	0.0336	2.9333	2.7439	0.0284	0.0295
	30	2,000	0.0146	0.0171	9.2333	6.1455	0.0098	0.0127
f_4	10	1,000	4.5782	2.1132	4.6100	2.5364	4.3285	2.7361
	20	1,500	22.8061	10.0912	19.6670	6.7661	17.7843	7.7291
	30	2,000	49.7192	13.7956	44.723	13.968	43.5283	12.7813

5 Conclusions

This work implemented a PSO variant called SBPSO that consists in a strategy to adapt serendipity in the context of metaheuristic algorithms. The new variant is based on a perceptual model of serendipity whose essence is related to Pasteur's principle that is generally used in recommender systems area to induce serendipity.

A set of benchmark functions was used to compare the proposed variant with traditional PSO and one variant of the literature called HWPSO. The experiments showed that SBPSO presented ability to escape from local optimum.

In 75% of experiments, SBPSO outperformed HWPSO variant. In all experiments, SBPSO outperformed traditional PSO in quality and solutions stability, but PSO runtime was better than SBPSO one.

The results are promising for the context of metaheuristic algorithms. It was observed that serendipity is a simple alternative that offers strategies for the metaheuristic algorithms benefit with the generation of optimal solutions.

The new variant was tested only in benchmark functions that are associated to multi-dimensional problems. Although these functions have been applied in several studies of PSO, it is interesting to evaluate the SBPSO performance in real world applications. For future works, the variant will be also tested in telecommunications area problems.

References

Adamopoulos, P. and Tuzhilin, E. (2011) 'On unexpectedness in recommender systems: or how to expect the unexpected', in *Proc. of RecSys '11 Intl. Workshop on Novelty and Diversity in Recommender Systems*.

Andel, P.V. (1994) 'Anatomy of the unsought finding. Serendipity: origin, history, domains, traditions, appearances, patterns and programmability', *The British Journal for the Philosophy of Science*, Vol. 45, No. 2, pp.631–648.

Boudjemaa, R. (2015) 'Maximisation of weather radar network coverage based on accelerated particle swarm optimisation', *International Journal of Metaheuristics*, Vol. 4, Nos. 3–4, pp.205–219.

Campos, J. and Figueiredo, A.D. (2002) 'Programming for serendipity', in *Proceedings of Fall Symposium on Chance Discovery – The Discovery and Management of Chance Events*, pp.48–60.

Catellin, S. (2014) *Sérendipité: Du conte au Concept*, Seuil, Paris.

Chaudhary, N., Raj, R., Kiran, K., Nema, S. and Padhy, P.K. (2015) 'Design of multivariable PID controller using DE-PSO', *International Journal of Automation and Control*, Vol. 9, No. 3, pp.173–185.

Chen, A-P., Huang, C-H. and Hsu, Y-C. (2009) 'A novel modified particle swarm optimization for forecasting financial time series', in *IEEE International Conference on Intelligent Computing and Intelligent Systems*, Vol. 1, pp.683–687.

Cuevas, E., González, A., Zaldívar, D. and Pérez-Cisneros, M. (2015) 'An optimisation algorithm based on the behaviour of locust swarms', *Int. J. Bio-Inspired Comput.*, Vol. 7, No. 6, pp.402–407.

Del Valle, Y., Venayagamoorthy, G., Mohagheghi, S., Hernandez, J-C. and Harley, R. (2008) 'Particle swarm optimization: Basic concepts, variants and applications in power systems', *Evolutionary Computation, IEEE Transactions on*, Vol. 12, No. 2, pp.171–195.

Demšar, J. (2006) 'Statistical comparisons of classifiers over multiple data sets', *The Journal of Machine Learning Research*, Vol. 7, pp.1–30.

Esmin, A. and Matwin, S. (2013) 'HPSOM: a hybrid particle swarm optimization algorithm with genetic mutation', *International Journal of Innovative Computing, Information and Control*, Vol. 9, No. 5, pp.1919–1934.

Esmin, A., Lambert-Torres, G. and de Souza, A. (2005) 'A hybrid particle swarm optimization applied to loss power minimization', *Power Systems, IEEE Transactions on*, Vol. 20, No. 2, pp.859–866.

Ge, M., Delgado-Battenfeld, C. and Jannach, D. (2010) 'Beyond accuracy: evaluating recommender systems by coverage and serendipity', in *Proceedings of the Fourth ACM Conference on Recommender Systems, RecSys '10*, ACM, New York, NY, USA, pp.257–260.

Iaquinta, L., de Gemmis, M., Lops, P., Semeraro, G. and Molino, P. (2009) *Can a Recommender System Induce Serendipitous Encounters?*, IN-TECH.

Jin, L. and Huang, Y. (2012) 'A particle swarm optimization-neural network prediction model for typhoon intensity based on isometric mapping algorithm', in *Fifth International Joint Conference on Computational Sciences and Optimization*, pp.857–861.

Kennedy, J. and Eberhart, R. (1995) 'Particle swarm optimization', in *Neural Networks Proceedings, IEEE International Conference on*, Vol. 4, pp.1942–1948.

Kuhn, T.S. (1970) *The Structure of Scientific Revolutions*, 2nd ed., International Encyclopedia of Unified Science, Foundations of the Unity of Science, University of Chicago Press, Chicago.

Lawley, J. (2013) 'Maximising serendipity: the art of recognising and fostering unexpected potential – a systemic approach to change', in *Neuro Linguistic Psychotherapy and Counselling Association*.

Li, S., Tan, M., Tsang, I. and Kwok, J-Y. (2011) 'A hybrid pso-bfgs strategy for global optimization of multimodal functions', *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, Vol. 41, No. 4, pp.1003–1014.

Ling, S-H., Iu, H. H.C., Chan, K.Y., Lam, H-K., Yeung, C.W. and Leung, F.H.F. (2008) 'Hybrid particle swarm optimization with wavelet mutation and its industrial applications', *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, Vol. 38, No. 3, pp.743–763.

Murthy, G.R., Arumugam, M.S. and Loo, C.K. (2009) 'Hybrid particle swarm optimization algorithm with fine tuning operators', *Int. J. Bio-Inspired Comput.*, Vol. 1, Nos. 1/2, pp.14–31.

Paiva, F.A.P., Costa, J.A.F. and Silva, C.R.M. (2014) 'An ontology-based recommender system architecture for semantic searches in vehicles sales portals', in *Hybrid Artificial Intelligence Systems, Lecture Notes in Computer Science*, Vol. 8480, pp.537–548, Springer Int. Publishing.

Paiva, F.A.P., Costa, J.A.F. and Silva, C.R.M. (2015) 'Uma meta-heurística alternativa de inteligência de enxames baseada em serendipidade guiada', in *2nd LA-CCI (Latin American) and 12th CBIC Brazilian Congress on Computational Intelligence*.

Parpinelli, R.S. and Lopes, H.S. (2011) 'New inspirations in swarm intelligence: a survey', *Int. J. Bio-Inspired Comput.*, Vol. 3, No. 1, pp.1–16.

Picheny, V., Wagner, T. and Ginsbourger, D. (2013) 'A benchmark of kriging-based infill criteria for noisy optimization', *Structural and Multidisciplinary Optimization*, Vol. 48, No. 3, pp.607–626.

Ram, G., Mandal, D., Kar, R. and Ghoshal, S.P. (2014) 'Optimal design of non-uniform circular antenna arrays using pso with wavelet mutation', *Int. J. Bio-Inspired Comput.*, Vol. 6, No. 6, pp.424–433.

Shim, V.A., Tan, K.C., Chia, J.Y. and Al Mamun, A. (2013) 'Multi-objective optimization with estimation of distribution algorithm in a noisy environment', *Evolutionary Computation*, Vol. 21, No. 1, pp.149–177.

Silva, E. and Bastos Filho, C. (2015) 'PSO efficient implementation on GPUS using low latency memory', *Latin America Transactions, IEEE (Revista IEEE America Latina)*, Vol. 13, No. 5, pp.1619–1624.

Singh, A. and Swarnkar, K.K. (2011) 'Power system restoration using particle swarm optimization', *International Journal of Computer Applications*, Vol. 30, No. 2, pp.25–32.

Souza, G.R., Goldbarg, E. F.G., Goldbarg, M.C. and Canuto, A.M.P. (2012) 'A multiagent approach for metaheuristics hybridization applied to the traveling salesman problem', in *Proceedings of the Brazilian Symposium on Neural Networks, SBRN '12*, IEEE Computer Society, Washington, DC, USA, pp.208–213.

Toms, E. (2000) 'Serendipitous information retrieval', in *First DELOS Network of Excellence Workshop on Information Seeking, Searching and Querying in Digital Libraries*, pp.11–12.

Wang, H., Li, H., Liu, Y., Li, C. and Zeng, S. (2007) 'Opposition-based particle swarm algorithm with cauchy mutation', in *IEEE Congress on Evolutionary Computation*, pp.4750–4756.

Wang, X.W., Shi, Y.P., Yu, R. and Gu, X.S. (2015) 'Intelligent welding robot path planning', in *Chinese Intelligent Automation Conference*, Vol. 1, pp.25–32.

Xu, C-h. (2015) 'An efficient clustering method for mobile users based on hybrid PSO and ABC', *International Journal of Innovative Computing and Applications*, Vol. 6, Nos. 3–4, pp.163–170.

Zhang, W-J. and Xie, X-F. (2003) 'DEPSO: hybrid particle swarm with differential evolution operator', in *IEEE International Conference on Systems, Man and Cybernetics*, Vol. 4, pp.3816–3821.

Zhang, M-X., Zhang, B. and Zheng, Y-J. (2014) 'Bio-inspired meta-heuristics for emergency transportation problems', *Algorithms*, Vol. 7, No. 1, pp.15.