Struct Multidisc Optim (2009) 37:415–427
DOI 10.1007/s00158-008-0240-9

RESEARCH PAPER

Improving solution characteristics of particle swarm optimization using digital pheromones

Vijay Kalivarapu · Jung-Leng Foo · Eliot Winer

Received: 12 July 2007 / Revised: 9 November 2007 / Accepted: 20 January 2008 / Published online: 14 March 2008

Abstract In this paper, a new approach to particle swarm optimization (PSO) using digital pheromones to coordinate swarms within an n-dimensional design space is presented. In a basic PSO, an initial randomly generated population swarm propagates toward the global optimum over a series of iterations. The direction of the swarm movement in the design space is based on an individual particle's best position in its history trail (pBest), and the best particle in the entire swarm (gBest). This information is used to generate a velocity vector indicating a search direction toward a promising location in the design space. The premise of the research presented in this paper is based on the fact that the search direction for each swarm member is dictated by only two candidates—pBest and gBest, which are not efficient to locate the global optimum, particularly in multi-modal optimization problems. In addition, poor move sets specified by pBest in the initial stages of optimization can trap the swarm in a local minimum or cause slow convergence. This paper presents the use of digital pheromones for aiding communication within the swarm to improve the search efficiency and reliability, resulting in improved solution quality, accuracy, and efficiency. With empirical proximity analysis, the pheromone strength in a region of the design space is determined. The swarm then reacts accordingly based on the probability that this region may contain an optimum. The additional information from pheromones causes the particles within the swarm to explore the design space thoroughly and locate the solution more efficiently and accurately than a basic PSO. This paper presents the development of this method and results from several multi-modal test cases.

Keywords Particle swarm optimization · Digital pheromones · Efficient swarm coordination · Multimodal design spaces · Accuracy · Efficiency and reliability

1 Introduction

Heuristic evolutionary search methods such as genetic algorithms (GAs) and simulated annealing (SA) are capable of exhaustively exploring n-dimensional design spaces to find optimal solutions. Their probabilistic nature provides distinct advantages over deterministic methods in finding global optimums, especially in multi-modal optimization problems. These methods are iterative in nature and typically do not require derivative calculations, thus allowing discontinuous design spaces to be explored. Although heuristic methods cannot mathematically guarantee decreasing an objective function or locating an optimum, they are often much more reliable than deterministic methods such as a conjugate direction search. However, a downside to heuristic methods is their computational expense and, sometimes, implementation complexity.

V. Kalivarapu (✉) · J.-L. Foo · E. Winer
Department of Mechanical Engineering,
Iowa State University, 1620 Howe Hall,
Virtual Reality Applications Center,
Ames, IA 50011, USA
e-mail: vkk2@iastate.edu

J.-L. Foo
e-mail: foo@iastate.edu

E. Winer
e-mail: ewiner@iastate.edu

Particle swarm optimization (PSO; Kennedy and Eberhart 1995; Eberhart and Kennedy 1995) is a zero-order, population-based heuristic method retaining many characteristics of evolutionary search algorithms. Compared to genetic algorithms (GAs) and simulated annealing (SA), it is simple to implement, and there are a smaller number of parameters to adjust (Fourie and Groenwold 2002; Carlisle and Dozier 2001). PSO has been recently added to the list of global search methods (Eberhart and Shi 2001; Kennedy and Eberhart 2001) due to its reliability in finding the global optimum for a wide range of problems. In PSO, a randomly generated particle swarm (a collection of particles) propagates toward a global optimum in a series of iterations, based on the information provided by two members—the best position of a swarm member in its history trail (pBest) and the best position attained by all swarm members (gBest). Based on this information, a basic PSO generates a velocity vector indicating the direction of the swarm movement and updates the location of the particles.

The first drawback of this method is that the particle updates are influenced by a limited number of factors. At any instance, each swarm member is directed only by two candidates—pBest and gBest. Having just these two candidates potentially impedes the desirable exploratory characteristics. In an n-dimensional design space, information from these two candidates alone will not always suffice to propagate a swarm toward the global optimum efficiently. A second drawback is that the method is initial-condition-dependent. Poor locations specified by pBest and gBest in the initial stages of optimization can potentially offset the swarm from attaining the neighborhood of the solution in the design space (Schutte and Groenwold 2005). This results in the swarm either being trapped in a local minimum or could take a long time to recover from a bad location and reach the global optimum.

The reasons above warrant for a robust exploration and exploitation of the design space that can accommodate enhanced solution quality, accuracy, and efficiency. This has been achieved using digital pheromones to create an improved PSO method. Coupled with empirical proximity analysis on the pheromones, an efficient move set is generated to update the search direction of each particle.

2 Particle swarm optimization

PSO is a population-based zero-order optimization method that exhibits several evolutionary characteristics similar to GAs and SA. These are (a) initialization with a population of random solutions, (b) design space search for an optimum through updating generations of design points, and (c) update based on previous generations (Hu et al. 2003).

PSO is based on a simplified model of the social behavior exhibited by the swarming behavior of insects, birds, and fish. In this analogy, a swarm member (particle) uses information from its past behavior (best previous location—pBest) and the behavior of the rest of the swarm (the overall best particle—gBest) to determine suitable food or nesting locations (local and global optimums). The algorithm iteratively updates the search direction of the swarm propagating toward the optimum. Equations (1) and (2) define the mathematical simulation of this behavior.

$$\begin{aligned} V_{\mathrm{iter+1,i}}[] = {} & w_{\mathrm{iter}} \times V_{\mathrm{iter,i}}[] \\ & + c_1 \times \mathrm{rand_p}() \times \left(\mathrm{pBest_i}[] - X_{\mathrm{i}}[]\right) \\ & + c_2 \times \mathrm{rand_g}() \times \left(\mathrm{gBest}[] - X_{\mathrm{i}}[]\right) \end{aligned} \tag{1}$$

$$X_{\mathrm{iter+1}}[] = X_{\mathrm{iter}}[] + V_{\mathrm{iter+1}}[] \tag{2}$$

$$w_{\mathrm{iter+1}} = w_{\mathrm{iter}} \times \lambda_w \tag{3}$$

Equation (1) represents the velocity vector update of a basic PSO method in iteration 'iter,' for each design variable represented by square braces and for each swarm member, i. $\mathrm{rand_p}()$ and $\mathrm{rand_g}()$ are random numbers generated each for *pBest* and *gBest* between 0 and 1. c_1 and c_2 are user definable confidence parameters. Typically, these are set to values of 2.0. 'pBest' represents the best position of the ith particle in its history trail, and 'gBest' represents the best particle location in the entire swarm. w_{iter} is termed "inertia" weight and is used to control the impact of a particle's previous velocity on the calculation of the current velocity vector. A large value for w_{iter} facilitates global exploration, which is particularly useful in the initial stages of an optimization. A small value allows for more localized searching, which is useful as the swarm moves toward the neighborhood of the optimum (Shi and Eberhart 1998a, b). These characteristics are attributed to the swarm by implementing a decay factor, λ_w for the inertia weight, as shown in (3). Equation (2) denotes the updated swarm location in the design space.

Ever since the inception of PSO in 1995, a significant number of modifications have been made to the basic algorithm for realizing performance improvements. Natsuki and Iba (2002), and Hu et al. (2003c) have explored the possibilities of performance improvement through introducing mutation factors in PSO, similar to the ones used in GAs. Various methods for constraint handling using PSO have been addressed. Venter and

Sobieski (2003) implemented a quadratic exterior penalty function method to solve nonlinear constrained optimization problems. Hu and Eberhart (2002) modified the basic PSO method so that the swarm is repeatedly initialized until all constraints are satisfied while also forcing pBest and gBest to be feasible in every iteration. Sedlaczek and Eberhard (2006) implemented the Augmented Lagrangian Method for solving constrained nonlinear optimization problems. Ray and Saini (2001) developed a method to improve swarm movement within the design space through information sharing between individual particle members. They have successfully implemented this strategy in solving both unconstrained and constrained problems as well. A preliminary implementation of digital pheromones for use by PSO has been addressed by the authors to improve design space exploration in single and parallel computing environments (Kalivarapu et al. 2006a, b). Other parallel implementations of PSO include works by Schutte et al. (2003) and Koh et al. (2006). PSO was used and modified for solving multi-objective problems as well (Hu et al. 2004; Coello et al. 2004). Gao et al. (2006) have obtained improvements in PSO through the use of a virus operator that propagates partial genetic information in the swarm by infection operators for enhanced design space search. Some of the recent advancements include solving traveling salesman problems using discrete PSO methods (Clerc 2004; Yang et al. 2006b; Li et al. 2006; Shen et al. 2006). Penalty function approaches have been used to solve mixed discrete nonlinear problems using PSO (Kitayama et al. 2005). Discrete PSO methods have been known to solve constrained optimization problems as well, and Yang et al. (2006a) have demonstrated it through converting constraint satisfaction problems into discrete optimization problems. Other areas include developments in the areas of solving integer programming (Liu et al. 2006) and continuous variable problems (Tayal and Wang 1999). Parsopoulos and Vrahatis (2002) have demonstrated the use of PSO for solving a wide range of problems including multi-objective, minimax, and integer programming problems. The same authors have developed methods to compute all global minimizers of an objective function using PSO (Parsopoulos and Vrahatis 2004). Similarly, He et al. presented methods to solve various mechanical design problems that tackled mixed variable types—integer, discrete, and continuous variables (He et al. 2004). A ‘fly-back’ constraint-handling mechanism was also introduced in this research to maintain a feasible population. A substantial amount of success has been achieved in utilizing PSO for applications such as aircraft design (Venter and Sobieski 2004; Pidaparti and Jayanti 2003), topology and shape optimization (Fourie and Groenwold 2001, 2002), structural optimization (Bochenk and Fory’s 2006; Schutte and Groenwold 2003), wireless network routing problems (Yang et al. 2006b), optimization in manufacturing and production operations (Onwubolu and Clerc 2004; Rameshkumar et al. 2005), collision detection problems (Tianzhu et al. 2006), and detection of optimal paths for unmanned aerial vehicles (UAVs; Batkiewicz et al. 2006; Foo et al. 2006) to name a few.

2.1 Digital pheromones

Pheromones are chemical scents produced by insects essentially as a means of communication in finding suitable food and nesting locations. The more insects that travel a path, the stronger the pheromone trail. A digital pheromone works on the same principle and is analogous to a natural pheromone in that it is a marker to determine whether or not a region in the design space is promising for further investigation. Digital pheromones have been used in applications such as the automatic adaptive swarm management of UAVs (Walter et al. 2005; Gaudiano et al. 2003). In this research, the implementation of digital pheromones allowed simulated swarms of UAVs to automatically adapt and navigate in potentially hazardous environments, dramatically reducing the requirement of human operators at the ground control stations. Other applications of digital pheromones include ant colony optimization for solving minimum cost paths in graphs (Colorni et al. 1991; Dorigo et al. 1996; Montgomery 2002) and solving network communication problems (White and Pagurek 1998).

2.2 PSO and digital pheromones

The concept of digital pheromones is relatively new (Parunak et al. 2002) and has not been explored to its full potential for investigating *n*-dimensional design spaces. The benefits of digital pheromones from swarm intelligence and the adaptive applications described above can be merged into PSO to improve design space exploration, particularly for a multi-modal optimization problem where swarm communication is essential to locating the global optimum.

In a basic PSO algorithm, the swarm movement is governed by the velocity vector computed in (1). Each swarm member uses information from its previous best and the best member in the entire swarm at any iteration. However, multiple pheromones released by the swarm members could provide more information on promising locations within the design space when

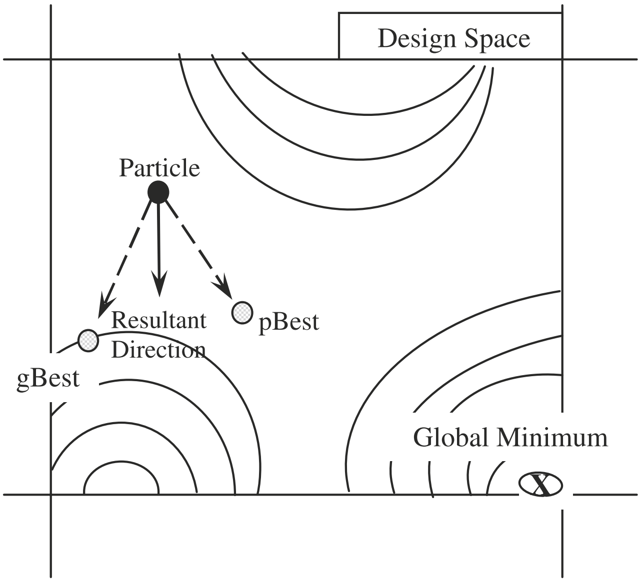

Fig. 1 Particle movement within a basic PSO

the information obtained from pBest and gBest are insufficient or inefficient. Figure 1 displays a scenario of a swarm member's movement whose direction is guided by pBest and gBest alone. If $c_1 >> c_2$, the particle is attracted primarily toward its personal best position. On the other hand, if $c_2 >> c_1$, the particle is strongly attracted to the gBest position. In the scenario dominated by c_2, as presented in Fig. 1, neither pBest nor gBest leads the swarm member to the global optimum, at the very least, not in this iteration adding additional computation to find the optimum.

Figure 2 shows the effect of implementing digital pheromones into the velocity vector. An additional pheromone component potentially causes the swarm member to result in a direction different from the combined influence of pBest and gBest, thereby increasing the probability of finding the global optimum.

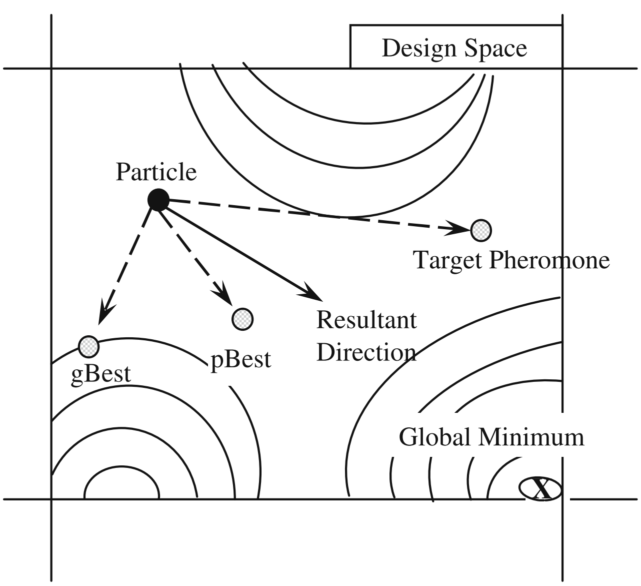

Fig. 2 Particle movement with digital pheromones

The research presented in this paper develops and implements this idea by exploring the benefits of an additional component into the velocity vector for achieving improved solution characteristics in PSO, particularly in multi-modal problems. The remaining sections of this paper focus on the method development, implementation into software, and then evaluation through several test cases.

3 Digital pheromone implementation

Figure 3 is an overview of PSO with steps involving digital pheromones highlighted. The method initialization is similar to a basic PSO except that 50% of the swarm within the design space is randomly selected to release pheromones in the first iteration. This parameter is user-defined, but experimentation has shown 50% to be a good default value. For subsequent iterations, each

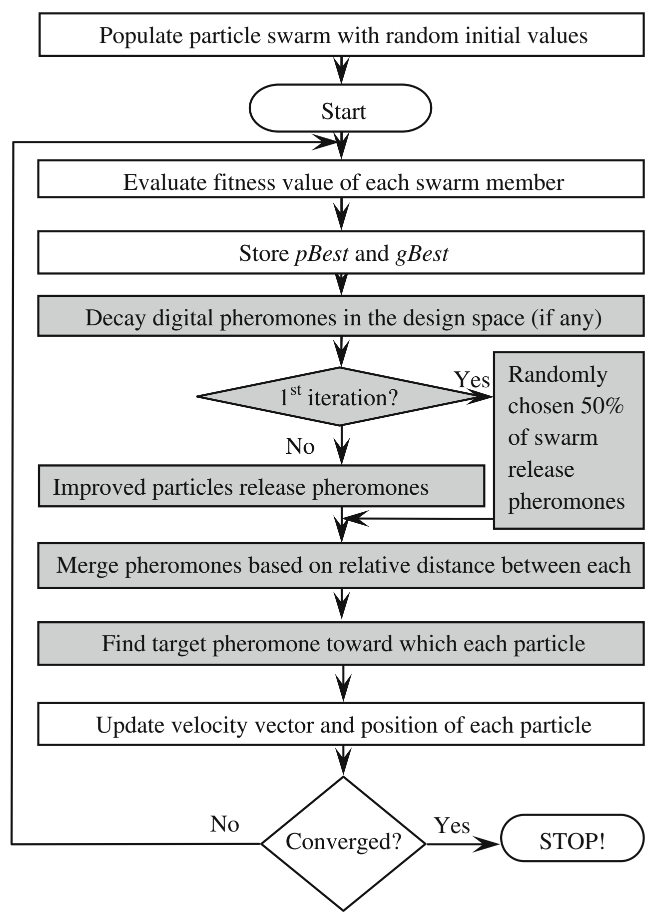

Fig. 3 Flowchart of PSO with digital pheromones

swarm member that realizes any improvement in the actual objective function value is allowed to release a pheromone. Pheromones from the current as well as the past iterations that are close to each other in terms of the design variable value are merged into a new pheromone location. Therefore, a pheromone pattern across the design space is created while keeping the number of pheromones manageable. In addition, the digital pheromones are decayed in every iteration just as natural pheromones. Based on the current pheromone level and its position relative to a particle, a ranking process is used to select a target pheromone for each particle in the swarm. This target position toward which a particle will be attracted is added as an additional velocity vector component to pBest and gBest. This procedure is continued until a prescribed convergence criterion is satisfied.

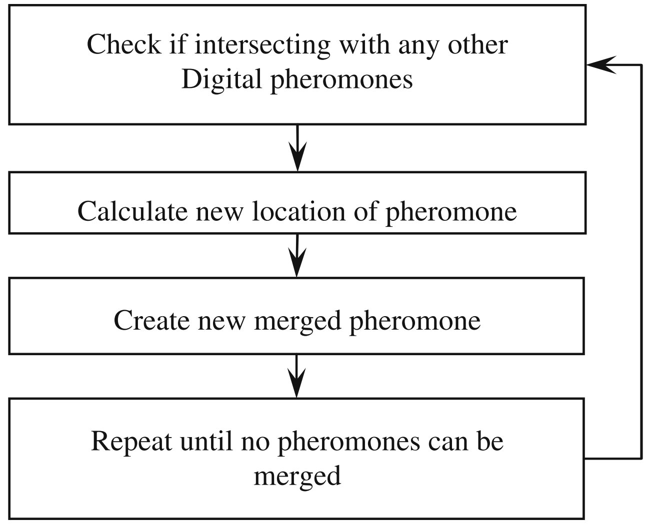

Fig. 4 Flowchart of pheromone merging process

3.1 Merging of digital pheromones

To populate the design space with an initial set of digital pheromones, 50% of the population is randomly selected to release pheromones, regardless of the objective function value. This is done so as to ensure a good spread of the swarm for exploring the design space in the initial stages of the optimization process. For subsequent iterations, the objective function value for each particle in the population is evaluated and only particles finding an improvement in the actual objective function value will release a pheromone. Any newly released pheromone is assigned a level P, with a value of 1.0. The pheromone levels are normalized between 0.0 and 1.0. Just as natural pheromones produced by insects decay in time, a user defined decay rate, λ_P (defaulting to 0.95), is assigned to the pheromones released by the particle swarm. Digital pheromones are decayed as the iterations progress forward to allow a swarm member to propagate toward a better design point by increasing the chances of attraction to a newer pheromone location with a better objective function value.

Every particle that finds a solution improvement releases a pheromone that potentially makes the number of pheromones unmanageably large as iterations progress. Therefore, an additional step to reduce them to a manageable number, yet retaining the functionality, was implemented. Pheromones that are closely packed within a small region of the design space are merged together. To check for merging, each pheromone is associated with an additional term 'radius of influence' (ROI). For each design variable of a pheromone, an ROI is computed and stored. The value of this ROI is a function of the pheromone level and the bounds of the design variables. Any two pheromones for a design variable less than the sum of the ROIs are merged into one. This is analogous to two spheres merging into one if the distance between them is less than the sum of their radii. A resultant pheromone level, whose location is the mid-point of the two merged pheromones, is then computed for the merged pheromones. Through this approach, regions of the design space with stronger resultant pheromone levels will attract more particles, and therefore, pheromones that are closely packed would indicate a high chance of optimality.

Also similar to the pheromone level decay, the ROI also has its own decay factor, λ_{ROI}, whose value is set equal to λ_P as a default. This is to ensure that both the pheromone levels and the radius of influence decay at the same rate. Figure 4 illustrates the pheromone merging process.

3.2 Proximity analysis to determine target pheromone

With numerous digital pheromones generated within the design space, a target pheromone needs to be identified for each swarm member. A criterion that is a function of both the pheromone level and its proximity from each particle needs to be considered in selecting the target pheromone. This is based on (a) the distance between the particle and the pheromone and (b) the pheromone level. For each particle, a target pheromone attraction factor P' is computed to this effect, which is a product of the pheromone level and the normalized distance between the particle and the pheromone. Equation (4) shows how the attraction factor P' is computed, and (5) computes the distance between the pheromone

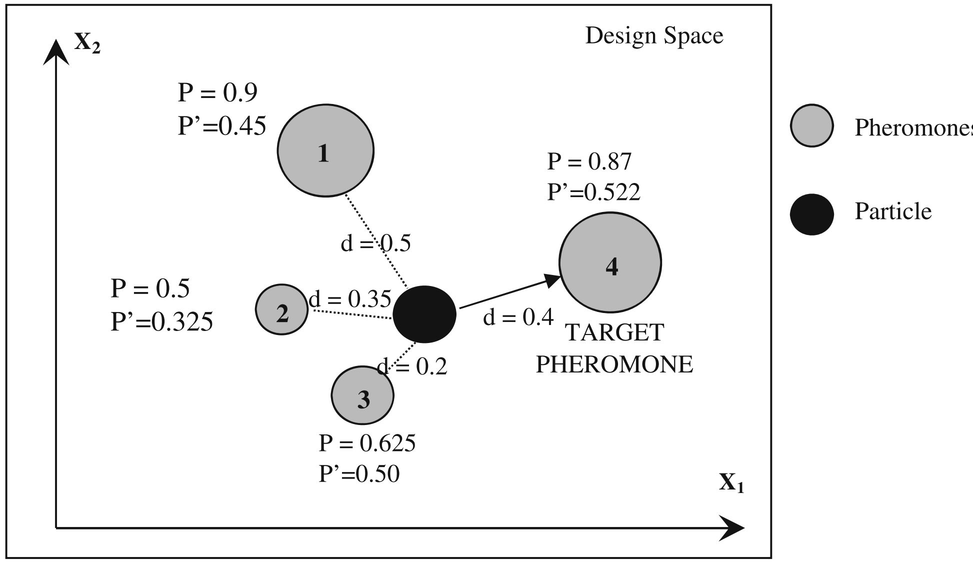

Fig. 5 Illustration of target pheromone selection

and each particle in the swarm. The variable range_k is the difference in the upper and lower limits of the kth design variable. Figure 5 shows an example scenario of a particle being attracted to a target pheromone. The particle in the figure is attracted to pheromone 4 that has the highest P' value based on its proximity to other pheromones and their pheromone levels.

$$P' = (1-d)P$$
$$d = \sqrt{\sum_{l}^{k}\left(\frac{Xp_k - X_k}{\text{range}_k}\right)^2} \quad (4)$$

$$\begin{aligned} k &= 1{:}n, \ \text{No. of design variables} \\ \mathrm{X_p} &= \text{Location of pheromone} \\ X &= \text{Location of particle} \end{aligned} \quad (5)$$

3.3 Velocity vector update

The velocity vector update, shown in (6), implements digital pheromones described in the methodology, Sections 3.1 and 3.2, as a new component called the target pheromone component.

$$\begin{aligned} V_{\text{iter}+1,\text{i}}[] = {} & w_{\text{iter}} \times V_{\text{iter,i}}[] \\ & + c_1 \times \text{rand}_{\text{p}}() \times (\text{pBest}_{\text{i}}[] - X_i[]) \\ & + c_2 \times \text{rand}_{\text{g}}() \times (\text{gBest}[] - X_i[]) \\ & + c_3 \times \text{rand}_{\text{T}} \times (\text{TargetPheromone}_{\text{i}}[] - X_i[]) \end{aligned} \quad (6)$$

c_3 is a user-defined confidence parameter for the pheromone component of the velocity vector similar to c_1 and c_2 in a basic PSO. c_3 combines the knowledge from the cognitive and social components of the velocity of a particle and complements their deficiencies. In a basic PSO, the particle swarm does not have a memory of the entire path traversed in the design space apart from the best position of an individual particle (pBest) and the best member's position in the entire swarm (gBest). The target pheromone component addresses this issue. It is a container that functionally stores the trail path of the swarm and utilizes the best features of pBest and gBest in steering toward a promising location in the design space. The confidence parameter c_3 determines the extent of influence a target pheromone can have on the swarm when the information from pBest and gBest alone are not sufficient or efficient to determine a particle's next move. The use of the target pheromone relies heavily on pBest and gBest. If $c_3 = 0$, there is no influence of pheromones and the swarm behaves as if in a regular PSO. If either of c_1 or c_2 is 0 and $c_3 > 0$, then the target pheromone location is essentially determined only by the non-zero component of pBest or gBest and propagated into the velocity vector. This creates a bias, thereby doubling the influence of non-zero pBest or gBest components on the swarm. This means that the swarm either explores or exploits the design space with double the intensity, either of which will prevent the swarm from converging. It is, therefore, essential that the influence of pBest and gBest be balanced (i.e., equal) for the pheromone component to provide accurate assistance in reaching the optimum.

Although analytical determination of a value for c_3 is out of the scope of this research, an empirical value has been determined through experimentation. A value between 2.0 and 5.0 has shown good performance characteristics and solved a variety of problems. A higher value of c_3 causes the velocity vector's magnitude to increase and places the swarm in a more general exploratory mode. However, it is desirable to make the swarm perform a tighter, local search as the swarm approaches the optimum. In this case, a lower value of c_3 is desirable. Therefore, decreasing c_3 can help the swarm to move from an exploratory mode to an exploitation mode. To achieve this effect, a decay of c_3 has been investigated in this research in addition to a constant c_3 to adapt to the swarm movement as required. Automatic adaption of the confidence parameters is not new. Literature shows the use of such approaches in basic PSO as well (Ratnaweera et al. 2002; Suganthan 1999). The result section provides an explanation of how useful c_3 decay was in various test problems. An inertia weight, w_i, of value 1.0 is initially chosen to preserve the influence of the velocity vector from previous iterations and gradually decreased using an inertia weight decay factor similar to the one used in a basic PSO.

3.4 Move limits

The additional pheromone term in the velocity vector update can considerably increase the computed velocity. Therefore, a move limit was applied to impose an upper bound on the maximum value of the velocity vector. To ensure a fair amount of freedom in exploring the design space, the swarm is allowed to digress up to 10% of the range of the design variables initially. A decay factor of 0.95 is applied to this move limit in subsequent iterations. This means that the freedom to explore the design space decreases as the iterations progress forward. The initial 10% value in the move limit showed good performance characteristics in the test cases used.

4 Test cases and results

Several unconstrained minimization problems of varying dimensionality were used as test problems to evaluate the developed method. These problems were also used to determine default values for the new parameters developed. The full mathematical descriptions for these problems can be found in (Engelbrecht 2005; Web Reference for Test Problems 2007a, b). The dimensionalities are given in Table 1.

From Table 2, there are 128 unique combinations of parameters. Each of these combinations was used to solve a problem 20 times to test repeatability of the method. This was performed for problems 4.1–4.4, yielding a total of 10,240 independent solution runs. Test problems 4.5–4.10 were tested, with the pheromone parameters that consistently provided the best answers from problems 4.1–4.4.

Table 1 Test problem matrix

Number	Test problem	Dimensions
4.1	Camelback function	2
4.2	Himmelblau function	2
4.3	Rosenbrock function	5
4.4	Ackley's path function	10
4.5	Dixon and Price function	15
4.6	Ackley's path function	20
4.7	Levy function	25
4.8	Sum of Squares function	30
4.9	Spherical function	40
4.10	Griewank function	50

The value for c_3 ranged from 1.0 through 10.0 to investigate if the pheromone influence needed to be large or small compared to pBest and gBest. Also, it was useful for studying the effect of the pheromone decay factor. The influence of pheromone levels and move limits were tested with decay rates ranging from 0.995 (0.5%) to 0.85 (15%) of their values in the previous iteration. The lower limit of the pheromone level decay was capped at 15% because a significant drop in the pheromone level could cause the influence of pheromones to be counter-productive for the swarm in reaching the global optimum. A similar reason was attributed for setting a lower bound on the move limit decay at 0.85.

The solutions obtained for test problems 4.1–4.4 were ranked in order of smallest average objective function value. Conclusions were made based on the results (in terms of solution accuracy compared to published solutions), and suitable values for pheromone parameters were determined. These parameters were then used into the developed method to solve problems 4.5–4.10.

The solutions (accuracy and solution times) from solving these test cases with experimentally determined digital pheromone parameters were compared against those obtained from a basic PSO. The swarm size was defined as ten times the number of design variables and was capped at a maximum of 500. All test cases were performed on a PC with an Intel Xeon processor (2.8 GHz) with 2 GB of system memory running the RedHat Enterprise Linux Operating System.

Table 3 provides a summary of the results obtained from solving problems 4.1–4.4. Problems solved using a basic PSO are designated with a "B" in parentheses next to the problem number, and those using pheromones have a "P." It tabulates the best-ranked solutions from 128 combinations of pheromone parameter values. For example, the pheromone parameter values determined for the camelback 2D problem

Table 2 Digital pheromone parameters

Pheromone parameters	Combination of values tested	Number of combinations
c_3 decay (0.5%)	No, yes (decayed once every 10 iterations)	2
c_3	10.0, 5.0, 2.0, 1.0	4
Pheromone level decay, λ_P	0.995, 0.95, 0.9, 0.85	4
Move limit decay, λ_{ML}	0.995, 0.95, 0.9, 0.85	4

(problem 4.1) were c_3 = 1, c_3Decay = NO, pheromone decay factor = 0.995, and maximum velocity decay = 0.85, which resulted in an optimum solution of 1.031618.

It can be seen from Table 3 that all pheromone parameters consistently produced an average objective function value less than when a basic PSO was used. As averages are not a true measure for performance, two other columns—the smallest objective function value achieved and the standard deviation—are noted in the table. The smallest objective function is the lowest value obtained in 20 trial runs for each test problem. The results demonstrate that the use of digital pheromones to search the design space provided substantially more information for the swarm to investigate the design space and attain the global optimum at a higher accuracy than a basic PSO, even when different parameter values were used. Although the solution value changed, it was still substantially better than using a basic PSO.

It should also be noted that PSO with digital pheromones required longer times to solve the Camelback 2D, Himmelblau 2D, and Rosenbrock 5D problems. This behavior was found true in all 128 test runs. The reason is attributed to the additional number of pheromone operations needed. However, as the complexity of the objective function in terms of number of design variables increases, the pheromones provided more information about the design space to the swarm, thereby converging in significantly less iterations. This is evident from the average solution times of the Ackley 10D problem where the solution duration with digital pheromones was smaller compared to a basic PSO. This suggests that decreased solution times become more prominent as the complexity of the optimization problem increases.

It can be seen from the table that the camelback 2D and Himmelblau 2D problems performed best with a c_3 value of 1.0, but Rosenbrock 5D and Ackley 10D performed best with c_3 = 2.0. Although it is inconclusive evidence, it provides some evidence for requiring a higher value for c_3 as the dimensionality of the problem increases. Also, Rosenbrock 5D required a decay in the c_3 value with the progress in iterations, while other test problems did not. In addition, the average number of iterations for Rosenbrock 5D was 30.8, which means that the value of c_3 was decayed only three times by a factor of 5% (since decay is performed once every ten iterations). Although solution sets from all pheromone parameter combination possibilities are not shown in this paper, it was observed that the performance of the method was influenced by the value of c_3 but relatively insensitive to decay in c_3. For example, in Ackley 10D (problem 4.4), the pheromone parameters that ranked second in terms of average objective function value

Table 3 Solution averages obtained from solving preliminary test problems

Problem no.	Average objective function	Smallest objective function	Standard deviation	Average duration per run (s)	Average no. of iterations	c_3	c_3 decay	λ_P	λ_{ML}
4.1 (B)	−1.018154	−1.030545	0.01049	0.40	19.65	–	–	–	–
4.1 (P)	−1.031618	−1.031628	0.00001	0.43	21.25	1	No	0.995	0.85
4.2 (B)	0.473549	0.000192	0.687921	0.43	21.25	–	–	–	–
4.2 (P)	0.000168	0.000001	0.000234	0.61	30.55	1	No	0.995	0.85
4.3 (B)	0.117105	0.000186	0.16704	1.06	21.1	–	–	–	–
4.3 (P)	0.000371	0.000000	0.00042	1.57	30.8	2	Yes	0.95	0.95
4.4 (B)	3.542873	0.002991	3.22811	10.26	102.5	–	–	–	–
4.4 (P)	0.001433	0.000661	0.00086	8.21	79.5	2	No	0.85	0.9

4.1 Camelback 2D, 4.2 Himmelblau 2D, 4.3 Rosenbrock 5D, 4.4 Ackley's path 10D, *B* results from basic PSO, *P* results from PSO with digital pheromones implemented required a c_3 value of 10.0 to produce a solution of 0.001496, as opposed to the parameter values that produced a solution of 0.001433. The solution accuracy between the two parameter sets was significant only in the fifth decimal place. However, it was noticed that in the first ten ranked solution sets for Ackley 10D, four out of ten cases required a c_3 value of 10, and three out of ten cases required a c_3 value of 5.0. This suggests that higher dimensional problems might require a higher value of c_3 so as to increase the influence of pheromones over the swarm. From these observations, a value of c_3 ranging from 2.0 to 5.0 is suggested.

The results also suggest that a value of 0.9 to 0.95 is an appropriate choice of value for the pheromone decay factor. The test cases revealed that a value greater than 0.95 or less than 0.9 allowed the pheromone component to become too large or small compared to the pBest and gBest components in the velocity vector. To achieve the maximum benefit from digital pheromones, some sort of balance needs to be in effect for all of these components of the velocity vector.

A pattern was observed in the value of move limit decay for the top ten ranked solutions for all the test problems. Camelback 2D and Himmelblau 2D required a move limit decay value of 0.85, Rosenbrock 5D required 0.95, and Ackley 10D required 0.9 for attaining global optimum solutions. Although the scales of objective function values for each test problems are different, a range of values between 0.85 and 0.95 seemed to be appropriate. It is to be noted that, although pheromone parameters are suggested in this section, they are user-defined parameters that can be altered to suit to a specific optimization problem.

Based on the knowledge gained about the pheromone parameters, the following values are suggested and used for solving problems 4.5–4.10:

- $c_3 = 5.0$ with no decay
- Pheromone decay = 0.95
- Move limit decay = 0.95

Table 4 provides the summary of results from these test runs. The table shows that digital pheromones, when used in PSO, consistently displayed superior performance when compared with solutions from a basic PSO.

Since the published solutions for most of the problems in Table 4 are 0.000, there was no measure for percentage accuracy. Therefore, a tolerance was given, and accuracy was measured based on the number of times the obtained solution was within the tolerance limits. For example, a tolerance limit of ±0.5 was assigned for a 20-design variable problem. If the solution was within this tolerance limit 85 times in 100 runs of the problem, the solution accuracy was 85%.

The solution accuracies noted in the table were within a tolerance limit of ±0.5. In all test cases, the solution accuracy of PSO with digital pheromones was either equal or superior when compared to basic PSO. In problem 4.6, for example (Ackley's 20D), the basic PSO was not able to solve the problem whereas the pheromone PSO attained the solution within the specified tolerance limits 85 out of 100 runs.

For the Dixon and Price function (problem 4.5), a solution accuracy of 65% was achieved by basic PSO. However, the average objective function value was

Table 4 Summary of results from solving problems 4.5–4.9

Problem no.	Solution accuracy (%)	Objective function			Average no. of iterations	Duration (s/run)
		Average	Smallest	SD		
4.5 (B)	65	48.366	0.0007	143.556	166.3	25.29
4.5 (P)	85	0.148	0.0003	0.466	92.0	14.72
4.6 (B)	0	4.658	2.659	2.740	166.7	16.899
4.6 (P)	85	0.171	0.003	0.418	143.3	15.095
4.7 (B)	100	0.143	0.131	0.041	96.5	24.819
4.7 (P)	100	0.132	0.130	0.001	40.8	11.289
4.8 (B)	0	16.301	0.521	48.692	212.45	66.10
4.8 (P)	85	0.084	0.006	0.128	138.25	46.46
4.9 (B)	100	0.033	0.0174	0.0078	162.65	68.35
4.9 (P)	100	0.002	0.0007	0.0007	85.15	39.38
4.10 (B)	0	1.189	1.056	0.133	186.1	99.014
4.10 (P)	100	0.008	0.003	0.005	158.1	93.801

4.5 Dixon and Price function 15D, 4.6 Ackley's path function 20D, 4.7 Levy function 25D, 4.8 sum of squares function 30D, 4.9 spherical function 40D, 4.10 Griewank function 50D, *B* results from basic PSO, *P* results from PSO with digital pheromones implemented

48.366, meaning that, when a swarm did not locate the optimum within the tolerance limits, it was very far away. The pheromone PSO method resulted in an 85% accuracy with an average objective function value of 0.148 and a standard deviation of 0.466. Therefore, even when the optimum was not located within the tolerance, the solution was still in the neighborhood of the optimum. A benefit, if restarting the method, is an option. Also, the average duration per run was significantly lower for pheromone PSO when compared to basic PSO.

The solution accuracy measure for the Levy function (problem 4.7) was essentially equal between basic and pheromone PSO. Both the methods solved the problem with 100% accuracy. Although the average, smallest, and standard deviation between the methods was very close, there is almost a 50% decrease in the solution time for pheromone PSO.

The basic PSO failed to solve the 30 dimensional sum of squares function (problem 4.8) within the specified tolerance limit of ±0.5. The smallest objective function value returned was 0.521. However, the pheromone PSO solved the problem within the tolerance limits on all 20 trial runs, along with improved average objective function value, standard deviation, and duration. Given the uni-modal nature of the test problem, the failure to solve the problem by basic PSO may be attributed to the high swarm size (300) causing substantial swarm activity negatively impacting convergence in the design space. The pheromone PSO, on the other hand, is relatively unaffected by the swarm size and converged faster when compared to basic PSO.

Both basic PSO and pheromone PSO were able to solve the 40D spherical function (problem 4.9) with 100% accuracy. However, as seen from the table, the average objective function evaluated by the pheromone PSO (0.002) is about 16 times better than the average objective function returned by basic PSO (0.033). Moreover, the solution time of pheromone PSO (39.38 s) is 42% faster when compared to basic PSO (68.35 s). Although the variation of the results in basic PSO is small, the pheromone PSO shows superior consistency as evident from the standard deviation of the objective function values.

The developed method also was able to solve a 50D highly multi-modal Griewank problem (problem 4.10) with 100% solution accuracy, whereas the basic PSO could not reach the global minimum in any of the 20 runs. There was a 5% improvement in the solution duration as well, and the standard deviation is significantly better than a basic PSO.

To emphasize that that the suggested pheromone parameters determined from problems 4.1–4.4 were good default values, they were used to solve problems 4.5–4.10. However, fine tuning of the pheromone parameters could potentially produce superior solutions than those in Table 4. For example, in problem 4.5, a nondecaying c_3 value of 5.0, a pheromone decay factor of 0.995, and a move limit decay of 0.95 produced a solution of 0.00534 when compared to 0.148 with the suggested pheromone parameters. This means that, although the suggested values perform well, additional performance improvement can be realized through refinements in the pheromone parameters.

4.1 Testing longer objective function evaluation times

The test cases presented thus far are academic in nature with easily computed analytical objective functions. They are not a true representative of the type of problems solved in industrial settings, where function evaluations can take a considerable amount of computational time. To test if longer function evaluation time has any impact on the performance of the developed method, a sleep time was added when evaluating the objective function. This was done to simulate an objective function with a longer evaluation time. Table 5 shows the solution times for solving Ackley's path function of 20 design variables when sleep times of 0, 5, 10, and 20 ms were added. The other parameter values used were $c_3 = 5.0$, $\lambda_P = 0.95$, and $\lambda_{ML} = 0.95$.

Table 5 Summary of results for Ackley 20D with variable objective function evaluation time

Sleep time (ms)	Basic PSO		Pheromone PSO		Solution time, % improved
	Avg. obj. func.	Duration (s)	Avg. obj. func.	Duration (s)	
0.0	4.659	16.898	0.171	15.095	10.67
5.0	4.622	115.671	0.146	103.148	10.83
10.0	5.016	185.863	0.065	177.323	4.59
20.0	4.258	374.963	0.051	333.056	11.18

The table summarizes average objective function values, solution times, and the improvement in solution times between the basic and pheromone PSO from 20 solution runs. The results indicate that, whereas the basic PSO attained a local minimum in all four sleep time scenarios, PSO with digital pheromones solved the problem with superior accuracy levels, very close to the global solution. The time improvement for a 10-ms sleep time was only 4.59%, but basic PSO converged prematurely. Also, about two times out of 20 runs, PSO with digital pheromones converged to a local minimum at ~1.5, which increased the average objective function value in the case of 0 and 5 ms sleep time. Overall, when compared to the performance of basic PSO in all sleep time scenarios, PSO with digital pheromones displayed substantial improvement.

4.2 Summary, conclusions and future work

This paper presents a new PSO method for improved design space exploration using digital pheromones. Four different benchmarking problems were tested to investigate the feasibility of this method. One hundred twenty-eight different combinations of digital pheromone parameters were tested in three different test cases, and suitable values were suggested. These values were further given as input to a higher dimensional problem to test the validity of the suggested pheromone parameters. The objective function values from solving the minimization test problems using digital pheromones were significantly lower than the results from basic PSO and better approached published solutions. Additionally, significant reduction in solution times for PSO with digital pheromones was noticed, indicating the method's improved efficiency. An additional observation was that solutions obtained using PSO with digital pheromones had smaller standard deviations, indicating a more consistent performance when compared to basic PSO. In addition, the solution duration improved when sleep times were introduced in the higher dimensional test case, indicating the viability of the proposed method to solve more realistic multimodal optimization problems. In spite of the increase in computations due to pheromone operations, the results showed a trend for decreased solution time as the global optimum was located faster. This is attributed to the information provided by the digital pheromones, thereby facilitating the swarm in propagating toward the global optimum faster.

Refining the performance of digital pheromones to solve a wide range of optimization problems is an ongoing venture; some of the near-future goals include implementing methods to solve constrained optimization problems and parallelization using synchronous and asynchronous schemes. An additional future direction is to perform statistical significance tests that will allow for an unbiased evaluation of the performance of the method.

References

Batkiewicz T, Dohse K, Kalivarapu V, Dohse T, Walter B, Knutzon J, Parkhurst D, Winer E, Oliver J (2006) Multimodal UAV ground control system. 11th AIAA/ISSMO multidisciplinary analysis and optimization conference, AIAA 2006-6963, Portsmouth, VA, 6–8 September

Bochenk P, Fory's P (2006) Structural optimization for post-buckling behavior using particle Swarm. Struct Multidisc Optim 32(6):521–530

Carlisle A, Dozier G (2001) An off-the-shelf PSO. In Proceedings of the workshop on particle swarm optimization, Indianapolis

Clerc M (2004) Discrete particle swarm optimization. New optimization techniques in engineering. Springer, Berlin

Coello CC, Pulido GT, Lechuga MS (2004) Handling multiple objectives with particle swarm optimization. IEEE Trans Evol Comput 8(3):256–279

Colorni A, Dorigo M, Maniezzo V (1991) Distributed optimization by ant colonies. In: Varela F, Bourgine P (eds) Proceedings of the European conference on artificial life. Elsevier, Amsterdam

Dorigo M, Maniezzo V, Colorni A (1996) Ant system optimization by a colony of cooperating agents. IEEE Trans Evol Comput B 26(1):29–41

Eberhart RC, Kennedy J (1995) A new optimizer using particle swarm theory. Proceedings of the sixth international symposium on micro machine and human science. Institute of Electrical and Electronics Engineers, Piscataway, NJ, pp 39–43

Eberhart RC, Shi Y (2001) Particle swarm optimization: developments, applications, and resources. In: Proceedings of the 2001 congress on evolutionary computation, pp 81–86

Engelbrecht A (2005) Fundamentals of computational swarm intelligence. Wiley, New York

Foo J, Knutzon J, Oliver J, Winer E (2006) Three-dimensional path planning of unmanned aerial vehicles using particle swarm optimization. 11th AIAA/ISSMO multidisciplinary analysis and optimization conference, AIAA 2006-6995, Portsmouth, VA, 6–8 September

Fourie PC, Groenwold AA (2001) The particle swarm algorithm in topology optimization. In Proceedings of the fourth world congress of structural and multidisciplinary optimization, Dalian, China

Fourie PC, Groenwold AA (2002) The particle swarm optimization algorithm in size and shape optimization. Struct Multidisc Optim 234:259–267

Gao F, Liu H, Zhao Q, Cui G (2006) Virus-evolutionary particle swarm optimization algorithm, vol 4222/2006. Springer, Berlin, pp 156–165

Gaudiano P, Shargel B, Bonabeau E, Clough B (2003) Swarm intelligence: a new c2 paradigm with an application to control of swarms of UAVs. In: Proceedings of the 8th international command and control research and technology symposium

He S, Prempain E, Wu QH (2004) An improved particle swarm optimizer for mechanical design Optimization problems. Eng Optim 36(5):585–605

Hu X, Eberhart R (2002) Solving constrained nonlinear optimization problems with particle swarm optimization. 6th world multiconference on systemics, cybernetics and informatics (SCI 2002), Orlando, USA

Hu X, Eberhart R, Shi Y (2003a) Engineering optimization with particle swarm. IEEE swarm intelligence symposium, pp 53–57

Hu X, Eberhart R, Shi Y (2003b) Particle swarm with extended memory for multiobjective optimization. Proceedings of 2003 IEEE swarm intelligence symposium, Indianapolis, IN, USA, IEEE Service Center, pp 193–197, Apr 2003

Hu X, Eberhart R, Shi Y (2003c) Swarm intelligence for permutation optimization: a case study of n-queens problem. IEEE swarm intelligence symposium, Indianapolis, IN

Kalivarapu V, Foo J, Winer E (2006a) Implementation of digital pheromones for use in particle swarm optimization. 47th AIAA/ASME/ASCE/AHS/ASC structures, structural dynamics and materials conference, Newport, RI, AIAA-2006-1917-941, May

Kalivarapu V, Foo J, Winer E (2006b) A parallel implementation of particle swarm optimization using digital pheromones. 11th AIAA/ISSMO multidisciplinary analysis and optimization conference. AIAA-2006-6908-694, Portsmouth, VA, Sept 2006

Kennedy J, Eberhart RC (1995) Particle swarm optimization. Proceedings of the 1995 IEEE international conference on neural networks, vol 4. Institute of Electrical and Electronics Engineers, Piscataway, NJ, pp 1942–1948

Kennedy J, Eberhart R (2001) Swarm intelligence. Morgan Kaufmann, San Mateo, CA

Kitayama S, Arakawa M, Yamazaki K (2005) Penalty function approach for the mixed discrete non-linear problems by particle swarm optimization. Struct. Multidisc Optim 32(3):191–202

Koh B, George AD, Haftka RT, Fregly B (2006) Parallel asynchronous particle swarm optimization. Int J Numer Methods Eng 67:578–595

Li X, Tian P, Hua J, Zhong N (2006) A hybrid discrete particle swarm optimization for the traveling salesman problem. Lecture notes in computer science, vol 4247/2006. Springer, Berlin, pp 181–188

Liu J, Sun J, Xu W (2006) Quantum-behaved particle swarm optimization for integer programming. Lecture notes in computer science, vol 4233/2006. Springer, Berlin, pp 1042–1050

Montgomery J (2002) Towards a systematic problem classification scheme for ant colony optimization. Technical report tr02-15, School of Information Technology, Bond University, Australia

Natsuki H, Iba H (2003) Particle swarm optimization with Gaussian mutation. Proceedings of IEEE swarm intelligence symposium, Indianapolis, pp 72–79

Onwubolu G, Clerc M (2004) Optimal path for automated drilling operations by a new heuristic approach using particle swarm optimization. Int J Prod Res 42(3):473–491

Parsopoulos KE, Vrahatis MN (2002) Recent approaches to global optimization problems through particle swarm optimization. Nat Comput Ser 1:235–306

Parsopoulos KE, Vrahatis MN (2004) On the computation of all global minimizers through particle swarm optimization. IEEE Trans Evol Comput 8(3):211–224

Parunak H, Purcell M, O'Conell R (2002) Digital pheromones for autonomous coordination of swarming UAVs. In: Proceedings of first AIAA unmanned aerospace vehicles, systems, technologies, and operations conference, AIAA, Norfolk, VA

Pidaparti R, Jayanti S (2003) Corrosion fatigue through particle swarm optimization. AIAA J 41(6):1167–1171

Rameshkumar K, Suresh R, Mohanasundaram K (2005) Discrete particle swarm optimization (DPSO) algorithm for permutation flowshop scheduling to minimize makespan. Lect Notes Comput Sci 3612(2005):572–581

Ratnaweera AC, Halgamuge SK, Watson HC (2002) Particle swarm optimiser with time varying acceleration coefficients. In: Proceedings of the international conference on soft computing and intelligent systems, pp 240–255

Ray T, Saini P (2001) Engineering design optimization using a swarm with an intelligent information sharing among individuals. Eng Optim 33:735–748

Schutte J, Groenwold A (2003) Sizing design of truss structures using the particle swarms. Struct Multidisc Optim 25: 261–269

Schutte F, Groenwold AA (2005) A study of global optimization using particle swarm. J Glob Optim 31:93–108

Schutte J, Reinbolt J, Fregly B, Haftka R, George A (2003) Parallel global optimization with the particle swarm algorithm. Int J Numer Meth Eng 61:2296–2315

Sedlaczek K, Eberhard P (2006) Using augmented Lagrangian particle swarm optimization for constrained problems in engineering. Struct Multidisc Optim 32:277–286

Shen B, Yao M, Yi W (2006) Heuristic information based improved fuzzy discrete PSO method for solving TSP. Lect Notes Comput Sci 4099(2006):859–863

Shi Y, Eberhart R (1998a) A modified particle swarm optimizer. Proceedings of the 1998 IEEE international conference on evolutionary computation. IEEE, Piscataway, NJ, pp 69–73

Shi Y, Eberhart R (1998b) Parameter selection in particle swarm optimization. Proceedings of the 1998 annual conference on evolutionary computation, Mar 1998

Suganthan PN (1999) Particle swarm optimiser with neighborhood operator. Proceedings of the IEEE congress on evolutionary computation. IEEE, Piscataway, NJ, pp 1958–1962

Tayal M, Wang B (1999) Particle swarm optimization for mixed discrete, integer and continuous variables. 10th AIAA/ISSMO multidisciplinary analysis and optimization conference, Albany, New York, 30–31 Aug

Tianzhu W, Wenhui L, Yi W, Zihou G, Dongfeng H (2006) An adaptive stochastic collision detection between deformable objects using particle swarm optimization. Lect Notes Comput Sci 3907/2006:450–459

Venter G, Sobieski J (2003) Particle swarm optimization. AIAA J 41(8):1583–1589

Venter G, Sobieski J (2004) Multidisciplinary optimization of a transport aircraft wing using particle swarm optimization. Struct Multidisc Optim 26(1–2):121–131

Walter B, Sannier A, Reiners D, Oliver J (2005) UAV swarm control: calculating digital pheromone fields with the GPU. The Interservice/Industry training, simulation and education conference (I/ITSEC), vol 2005 (Conference theme: One team. One fight. One training future).

Web Reference for Test Problems (2007a) Global optimization. http://www-optima.amp.i.kyoto-u.ac.jp/member/student/hedar/Hedar_files/TestGO_files/Page364.htm. Cited 8 Nov 2007

Web Reference for Test Problems (2007b) GEATbx: example functions (single and multi-objective functions) 2 parametric optimization. http://www.geatbx.com/docu/fcnindex-01.html. Cited 8 Nov 2007

White T, Pagurek B (1998) Towards multi-swarm problem solving in networks, ICMAS. Third international conference on multi agent systems (ICMAS'98), pp 333

Yang Q, Sun J, Zhang J, Wang C (2006a) A hybrid particle swarm optimization for binary CSPs. Lect Notes Comput Sci 4115/2006:39–49

Yang S, Huang R, Shi H (2006b) Mobile agent routing based on a two-stage optimization model and a hybrid evolutionary algorithm in wireless sensor networks. Lect Notes Comput Sci 4222/2006:938–947