

Research Article

Hierarchical Swarm Model: A New Approach to Optimization

Hanning Chen,¹ Yunlong Zhu,¹ Kunyuan Hu,¹ and Xiaoxian He²

¹ Key Laboratory of Industrial Informatics, Shenyang Institute of Automation, Chinese Academy of Sciences, Faculty Office III, Nanta Street 114#, Dongling District, Shenyang 110016, China

² School of Information Science and Engineering, Central South University, Changsha 410083, China

Correspondence should be addressed to Hanning Chen, perfect.chn@hotmail.com

Received 1 September 2009; Revised 16 January 2010; Accepted 8 March 2010

Academic Editor: Aura Reggiani

Copyright © 2010 Hanning Chen et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

This paper presents a novel optimization model called hierarchical swarm optimization (HSO), which simulates the natural hierarchical complex system from where more complex intelligence can emerge for complex problems solving. This proposed model is intended to suggest ways that the performance of HSO-based algorithms on complex optimization problems can be significantly improved. This performance improvement is obtained by constructing the HSO hierarchies, which means that an agent in a higher level swarm can be composed of swarms of other agents from lower level and different swarms of different levels evolve on different spatiotemporal scale. A novel optimization algorithm (named PS²O), based on the HSO model, is instantiated and tested to illustrate the ideas of HSO model clearly. Experiments were conducted on a set of 17 benchmark optimization problems including both continuous and discrete cases. The results demonstrate remarkable performance of the PS²O algorithm on all chosen benchmark functions when compared to several successful swarm intelligence and evolutionary algorithms.

1. Introduction

Swarm intelligence (SI), which is inspired by the “swarm behaviors” of social animals [1], is an innovative artificial intelligence technique for solving hard optimization problems. In SI system, there are many simple individuals who can interact locally with one another and with their environments. Although such systems are decentralized, local interactions between individuals lead to the emergence of global behaviors or global properties. For instance, flock of birds and school of fish emerge spatial self-organized patterns through social foraging [2]. Similar phenomena can also be observed in colonies of single-cell bacteria, social insects like ants and bees, as well as multicellular vertebrates, which all display collective intelligence [3].

As a problem-solving technique, many algorithmic methods of SI were designed to deal with practical problems. In 1991, Dorigo proposed ant colony optimization (ACO) [4, 5] based on foraging behaviors of ant colonies. ACO has been successfully used to solve discrete

optimization problems, like traveling salesman problems (TSP) [6]. After that, another SI algorithm, namely, particle swarm optimization (PSO), was proposed by Kennedy and Eberhart [7], which gleaned ideas from the social behavior of bird flocking and fish schooling [8–10]. PSO is primarily concerned with continuous optimization problems. In 2001, Passino proposed a technique known as bacterial foraging optimization (BFO) that inspired by the pattern exhibited by bacterial foraging behaviors [11]. Other swarm optimization methods have been developed like artificial immune systems (AIS) [12], which are based on the metaphor of the immune system as a collective intelligence process [13]. Recently, Karaboga has described a bee swarm algorithm called artificial bee colony (ABC) algorithm [14], and Basturk and Karaboga compared the performance of ABC algorithm with the performance of genetic algorithm (GA) in [15]. These SI paradigms have already come to be widely used in many areas [8, 16–22].

In current artificial SI systems, however, researchers only take into account the collective behaviors of one level of individuals, and ignored the hierarchical nature [23] of the real world systems and animal society. In fact, for most social insects and animals, their organizational structures are not flat. They can form complex hierarchical (or multilevel) system structures by self-organization and division of labor features [24]. In other words, in a hierarchical system, a swarm of lower level individuals can be the infrastructure of a single individual at the higher level [25, 26]. Here the term “swarm” is used in a general sense to refer to any collection of interacting agents. In most of natural hierarchically complex systems, swarms of lower level agents interact with each other to constitute more complex high-level swarms’ constituent agents, repeatedly, until very complex structures with greatly enhanced macroscopical intelligence emerge. Such phenomenon is so common in the natural world, this guides us to design a multilevel algorithmic model to mimic hierarchical emergence of nature society.

First, this paper extends the traditional SI framework from flat (one level) to hierarchical (multiple level) by proposing a novel optimization model called hierarchical swarm optimization (HSO). In HSO model, collective behaviors of multiple levels are taken into account to solve complex problems. Then some initial insights into this method are provided by designing a two-level HSO algorithm (named PS^2O) based on canonical PSO model. Four versions of PS^2O are realized according to different structures of cooperation and interaction types in each level. In order to evaluate the performance of PS^2O , extensive studies based on a set of 17 benchmark functions (including both continuous and discrete cases) have been carried out. For comparison purposes, we also implemented the genetic algorithm (GA), covariance matrix adaptation evolution strategy (CMA-ES), artificial bee colony algorithm (ABC), and four state-of-the-art PSO variants on these functions. The experimental results are encouraging; the PS^2O algorithm achieved remarkable search performance in terms of accuracy, robustness, and convergence speed on all benchmark functions.

The rest of the paper is organized as follows. Section 2 describes the proposed hierarchical swarm optimization model. In Section 3, a novel HSO-based optimization algorithm, namely, PS^2O , is given. Section 4 tests the algorithm on the benchmarks and illustrates the results. Finally, Section 5 outlines the conclusions.

2. Hierarchical Swarm Optimization

2.1. From Flat Swarm to Hierarchical Swarm

In [3], Bonabeau et al. define swarm intelligence as “the emergent collective intelligence of groups of simple agents”. In such a perspective, the artificial SI systems, which are designed

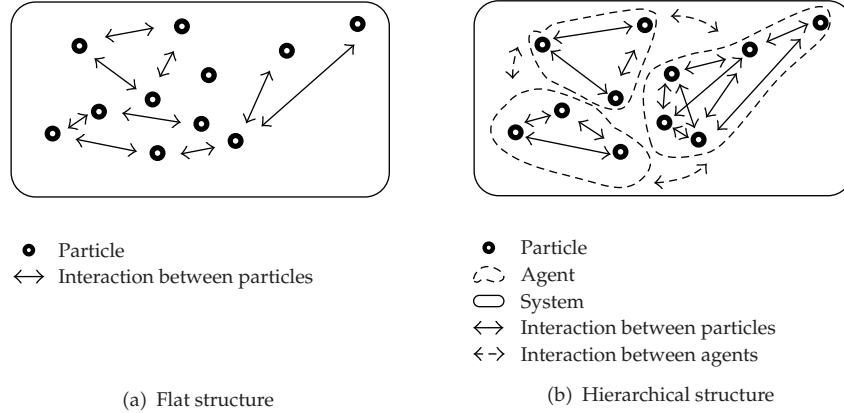


Figure 1: Two types of systems.

for complex problem solving, maintain a swarm made up of many isomorphic and relatively simple individuals that often share the same states and behaviors set. In such a swarm, all individuals have absolutely equal status in the whole life cycle. The interaction relations between these individuals are symmetrical and operate on the same spatiotemporal scale. One individual can be substituted by another one, while the function of the swarm remains steady. That is, the architecture and functionality of classical SI are flat (Figure 1(a)).

However, swarm intelligence only explains partial mechanisms of collective behavior of biology. The natural cases could be more complex: except for individual tasks, these units lie at a hierarchical level between an individual ant and the colony as a whole, and thus constitute what might be called “intermediate-level parts” [27]. Now consider two basic types of systems: hierarchical and nonhierarchical (flat). Flat systems can be regarded as a group of undifferentiated particles, such as traditional SI systems. Hierarchical swarm systems must have a structure requiring a minimum of two hierarchical levels (Figure 1(b)). In Figure 1(b), a particle is the minimum unit of the system, while an agent constitutes the intermediate hierarchical level, which is composed of a number of particles. In this perspective, it is evident that the complexity of the individual “agents” of SI systems is dramatically simplified to particles, which are the minimum unit of the systems. Hence, the hierarchical nature of swarm [23] is ignored in the traditional artificial SI systems, such as PSO and ACO.

Hierarchy is common in the real world. For examples, immune system antibodies continuously self-organize and evolve while being a part of the many “organism agents” of a bird, and that a bird is in turn an agent in the formation of a flock of birds, and the flock of birds is in turn an agent that is part of a particular ecosystem niche [28]. Genes, the elementary biochemical coding units are complicated macromolecular strings, as are the metabolic units, the proteins. Neurons, the basic elements of cognitive networks, themselves are cells. In any of these examples, it is evident that the interactions of the agents lead to a coherent structure at a higher level [29]. That is, the emergent characteristics of a particular lower level system frequently form an individual agent at a higher level of the hierarchical system. This aspect has been emphasized by many researchers on artificial intelligence and complex systems [23, 25, 29–32].

Hence, this paper strives to extend the traditional SI framework from flat to hierarchical, and propose the hierarchical swarm optimization model. By incorporating

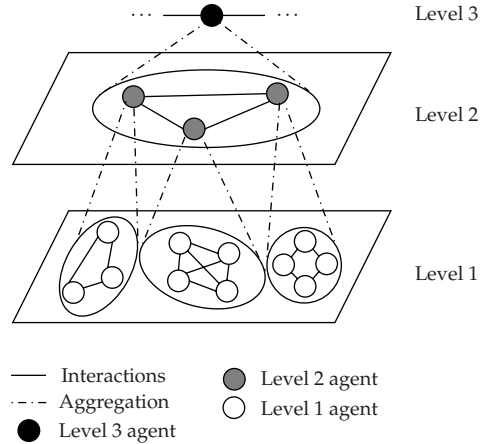


Figure 2: Hierarchical swarm optimization model using a multiagent system in nested hierarchies.

these new degrees of complexity, HSO-based optimization algorithm can accommodate a considerable potential for solving more complex problems.

2.2. HSO Model Description

HSO model accommodates a hierarchical multiagent system, in which an agent can itself be a swarm of other agents.

- (1) HSO is composed of a number of levels. Each level is a multiagent system composed of several swarms of agents.
- (2) Each swarm of level $n - 1$ agents is aggregated into a level- n agent.
- (3) Level- n behavior emerges from the organization of level 1 to n .

HSO naturally admits of a description in terms of higher level and lower level, where the lower level is nested within the higher level. Any agent at any level is both a component of a given swarm in its own level and a subsystem decomposable into a swarm of other agents at its adjacent lower level of HSO (shown as in Figure 2). Note that the agents in the lowest level are the particles that are the minimum unit, which are indecomposable of this hierarchical system. HSO is a heterogeneous system that each swarm in each level is evolved in its own population and adapts to the environment through the application of any SI method at hand. The interaction topology of HSO can also be heterogeneous hierarchical structures. Namely, the evolution rules and the interaction topology of distinct swarms can be different, and these different SI paradigms hierarchically construct the HSO model and lead to the hierarchical emergence of intelligence. In mathematical terms, the HSO model can be defined as in Table 1.

Figure 3 lists a general description of HSO containing four main functional blocks. In the first block of Figure 3, we show that under the external environment pressure (defined by the object function), each agent in the HSO model evolves and adapts as a consequence of internal and external hierarchical interactions. Both in higher level and lower level, the swarms can be manipulated by different SI algorithms (shown as in blocks 2 and 3 of Figure 3). In principle, any SI algorithms (such as PSO, ACO, BFO, and ABC) can be used

Table 1: The structure of HSO model.

HSO = (A, N, P, T, O, S)

A: agents that exist at various hierarchical levels.

N: the number of levels.

P: the populations of each swarm in each level.

T: the hierarchical interaction topology of HSO.

O: the objective optimization goals.

S: Swarm or Evolutionary optimization strategies used for each swarm to search the objective O.

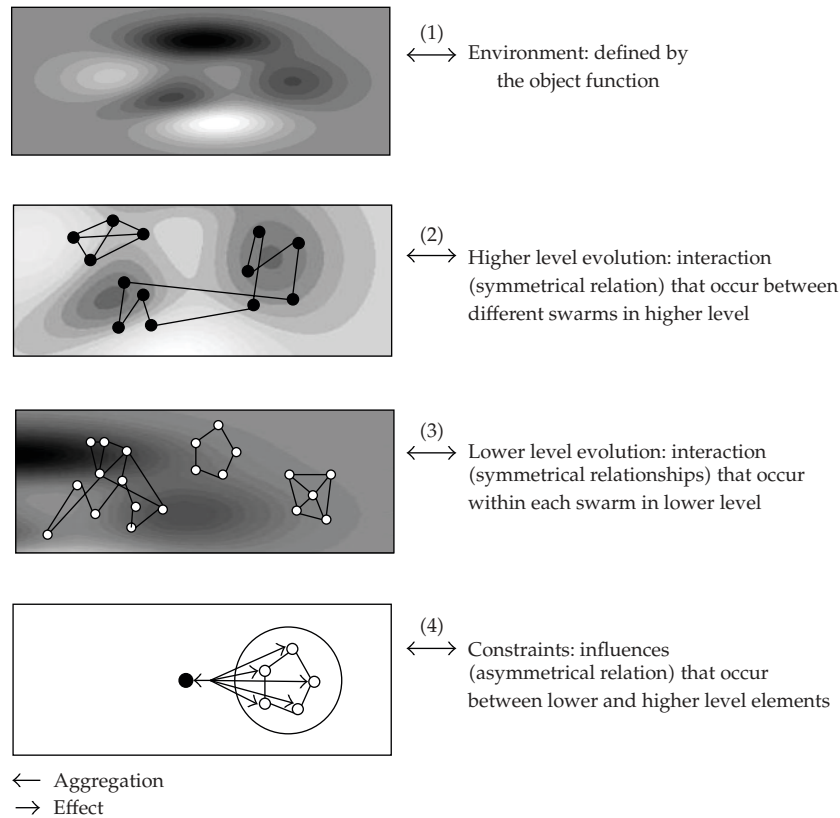


Figure 3: The main functional blocks of the HSO model.

by any swarm at any level, and we have first hand experience constructing HSO paradigms using PSO and BFO [33–35]. Interactions that occur within one level (each entity of the interaction is operating on the same spatiotemporal scale) are called symmetrical relations.

On the other hand, asymmetrical relationships occurring between different levels are called “constraints” [32]. The fourth block formed the constraint that higher level affects elements from lower level. When an agent of higher level transmits the information to its constituent swarms of other agents of lower level, the effect can be the according evolutionary actions of this agent’s swarm of constituent agents.

3. Case Study: The PS²O Algorithm

In this section, we implement a simple two-level HSO algorithm, which employs PSO method in each swarm of each level, and hence named it PS²O. Here the agents (particles) in the lower level (level-1) are analogous to individuals in a biological population (species) and the agents in the higher level (level-2) are analogous to species. As the hierarchical interactions that occur in the real ecosystems, from the macro view, dissimilar species establish symbiotic relationships to improve their survivability in level-1 of PS²O (i.e., interspecies cooperation); from the micro view, species' members (the particles) cooperatively interact with each other in level-2 of PS²O (i.e., intraspecies cooperation).

3.1. Levels Detail of PS²O

Here the basic goals are to find the minimum of $f(\vec{x})$, $\vec{x} \in \mathbb{R}^D$. We create an ecosystem in level-1 that contains a species set $\Omega = \{S_1, S_2, \dots, S_M\}$, and each species k possesses a member set, $S_k = \{X_{1k}, X_{2k}, \dots, X_{Nk}\}$, in level-2. The i th member of the k th species is characterized by the vector $X_{ik} = (X_{ik}^1, X_{ik}^2, \dots, X_{ik}^D)$. In each generation t , the evolution process of each level is detailed as follow.

3.1.1. Level 1

Level-1 agents are clustered into M swarms, each of which possesses N agents. Each swarm constitutes an agent of level-2. Each swarm of level-1 evolves within its own separate population via separate PSO algorithm. That is, there are M parallel PSO paradigms evolving separately in level-1. This process addresses the cooperation between individuals of the same species: within the species k , one or more members in the neighborhood of X_{ik} contribute their experiments to X_{ik} , and X_{ik} also share its knowledge with its neighbors. Then X_{ik} accelerate towards its personal best position and the best position found by its species members in neighborhood:

$$\alpha_{ik}^{t+1} = c_1 r_1 (pbest_{ik}^t - X_{ik}^t) + c_2 r_2 (sbest_k^t - X_{ik}^t), \quad (3.1)$$

where α_{ik} is the social acceleration vector of X_{ik} , $pbest_{ik}$ is the personal best position found so far by X_{ik} , $sbest_k$ is the best position found so far by its neighbors within species k , c_1 are individual learning rate, c_2 are social learning rate, and $r_1, r_2 \in \mathfrak{R}^d$ are two random vectors uniformly distributed in $[0, 1]$.

3.1.2. Level 2

All level-2 agents aggregate into a single swarm. This swarm of distinct symbiotic species coevolves via the social only version of the PSO [36] as the cognitive processes have already taken care of by the level-1 swarms. From the coevolution perspective, the species k accelerates towards the best position that the symbiotic partners of species k have found:

$$\beta_k^{t+1} = c_3 r_3 (cbest^t - S_k^t), \quad (3.2)$$

where β_k is the symbiotic acceleration vector of S_k , $cbest$ is the best position found so far by the symbiotic partners of the k th species, c_3 is the “symbiotic learning rate”, and $r_3 \in \mathfrak{R}^d$ is a uniform random sequence in the range $[0, 1]$.

3.1.3. Constraints

When species k in level-2 accelerates towards the best position, $cbest$, found by its more successful symbiotic partners, the according evolutionary action of this agent’s swarm of constituent agents from level-1 is that all the members of species k accelerate to $cbest$ too

$$\beta_{ik}^{t+1} = c_3 r_3 (cbest^t - X_{ik}^t), \quad (3.3)$$

where β_{ik} is the symbiotic acceleration vector of X_{ik} .

Then the velocity V_{ik} and position X_{ik} of each member of species k are updated according to

$$\begin{aligned} V_{ik}^{t+1} &= \chi (V_{ik}^t + \alpha_{ik}^{t+1} + \beta_{ik}^{t+1}), \\ X_{ik}^{t+1} &= X_{ik}^t + V_{ik}^{t+1}, \end{aligned} \quad (3.4)$$

where χ is known as the constriction coefficient [37].

3.2. Hierarchical Interaction Topologies

Systems of interacting agents—like many natural and social systems—are typically depicted by scientists as the graphs or networks, in which Individuals can be connected to one another according to a great number of schemes [38]. In PSO, since the original particle swarm model is a simulation of the social environment, a neighborhood that structured as the interaction topological graph is defined for an individual particle as the subset of particles it is able to communicate with. Four classical interaction topologies have been shown as in Figure 4.

Most particle swarm implementations use one of two simple interaction topologies. The first, namely, the fully-connected topology (see Figure 4(a)), conceptually connects all members of the population to one another. The effect of this topology is that each particle is influenced by the very best performance of any member of the entire population. This means faster convergence, which implies a higher risk to converge to a local minimum. Experiments show that the fully-connected topology is faster than the other neighborhoods, but it meets the optimal fewer times than any other one. The second, called ring topology (see Figure 4(b)), creates a neighborhood for each individual comprising itself and its two nearest neighbors in the population. The ring neighborhood is more robust if the maximum number of iterations was increased but much slower. However, experiments show that the ring neighborhood cannot meet the required precision for many complex problems. That is, it promotes the exploration, but unfortunately fails to provide the exploitation.

In our model, the interaction of agents occurred in a two-level hierarchical topology. By employing two simple topologies—the ring and the fully-connected topologies—for swarms in different levels, four hierarchically nested interaction topologies have been obtained.

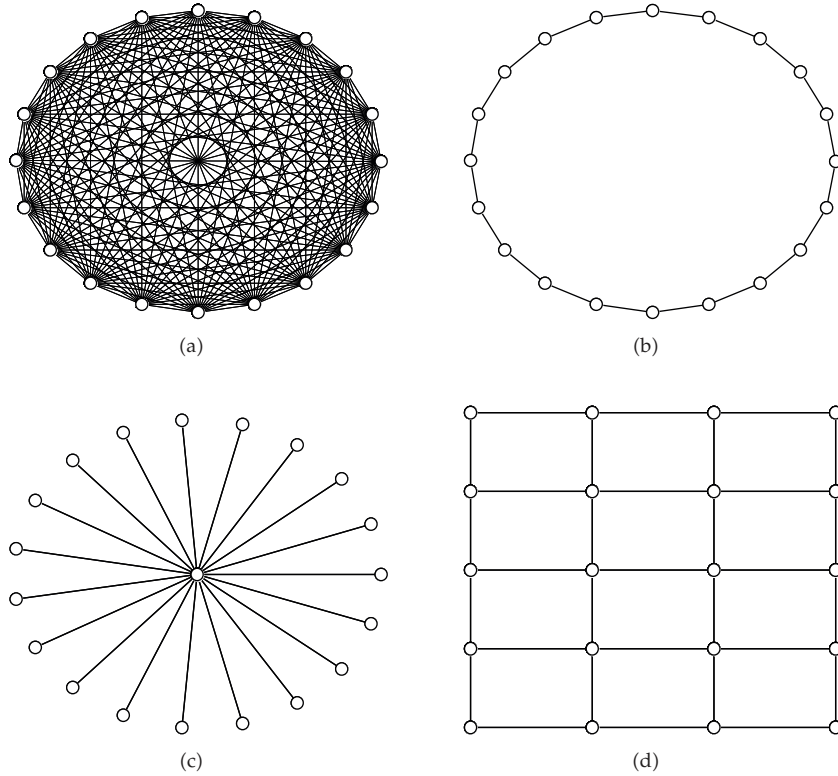


Figure 4: Four interaction topologies for PSO: (a) fully-connected, (b) ring, (c) star, (d) grid.

Shown as in Figure 5, each hierarchical topology is comprised of 4 swarms in level-2 and each swarm possesses 4 agents from level-1. The first two topologies have a homogeneous hierarchical structure (employ the ring or fully-connected topology in both levels) and the other two have the heterogeneous hierarchical structures (employ the ring and fully-connected topologies in different levels, resp.). Four variant versions of the PS^2O algorithms are studied, respectively, in this paper according to these four interaction topologies.

- (i) PS^2O-S : in level-1, agents interact with each other in each swarm. In level-2, each agent is influenced by the performance of all the other agents. That is, swarms of both levels are configured into the fully-connected topology (Figure 5(a)).
- (ii) PS^2O-R : in level-1, agents interact with 2 immediate agents in its neighborhood. In level-2, each agent is influenced by the performance of its two symbiotic partners only. That is, both levels are configured into the ring topology (Figure 5(b)).
- (iii) PS^2O-S_R : In level-1, agents interact with each other in each swarm. In level-2, each agent is influenced by the performance of its two symbiotic partners only. That is, the level-2 is configured into the fully-connected topology while the each swarm of level-1 is configured into the ring topology (Figure 5(c)).
- (iv) PS^2O-R_S : In level-1, each agent interacts with 2 immediate agents in its neighborhood. In level-2, each agent is influenced by the performance of all the other agents. That is, each swarm of the level-1 is configured into the ring topology while the level-2 is configured into the fully-connected topology (Figure 5(d)).

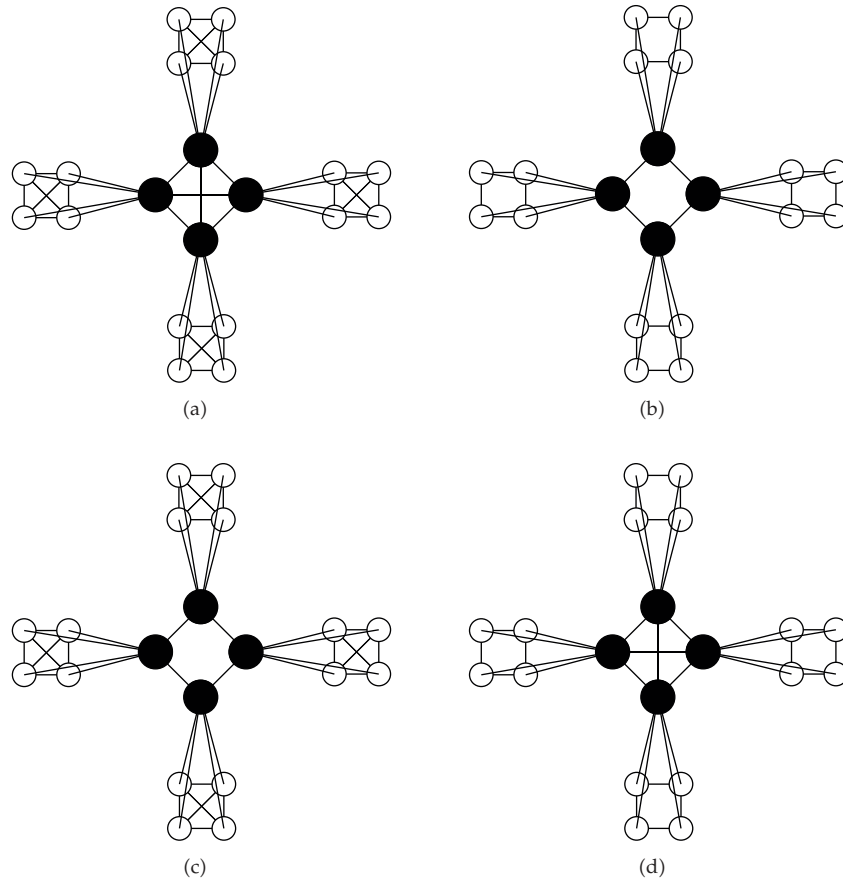


Figure 5: Four hierarchical interaction topologies for PS²O.

3.3. Matrix Representation

A multidimensional array representation of the PS²O algorithm is proposed in this section. The PS²O randomly initializes a number of M species with each possesses a number of N members to represent the biological community in the natural ecosystems. Then the positions X , velocities V , and personal best locations P of the biological community are all specified as the three-dimensional (3D) matrixes (showed as in Figures 6(a)–6(c)), where the first matrix dimension—Species number—is the number of species in level-2, the second matrix dimension—Swarm size—is the number of agents of each swarm in level-1, and the third matrix dimension—Dimension—is the number of dimensions of the object problem.

In PS²O model, in order to update the velocity and position matrixes, every agent in level-1 must accelerate to three factors: the previous best position of the agent itself (this factor is called “personal best”), the previous best position of other members in its neighborhood (we named this factor “species best”), and the previous best position found by other species (agents from level-2) that have the cooperative symbiotic relation to the species that this agent belongs to (we named this factor “community best”). The species best is represented by a 2D matrix S , which showed as in Figure 6(d) left, and the community best is referred to as a 1D matrix C , which showed as in Figure 6(d) right.

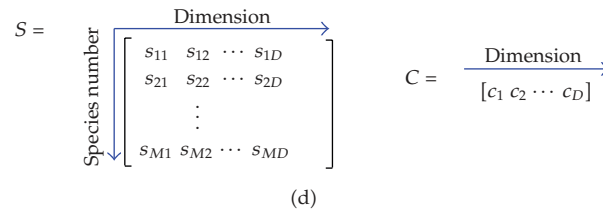
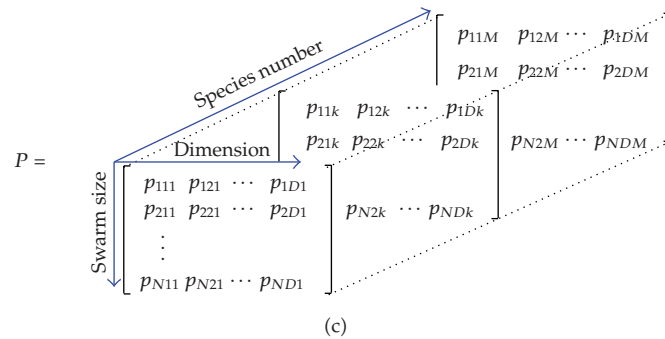
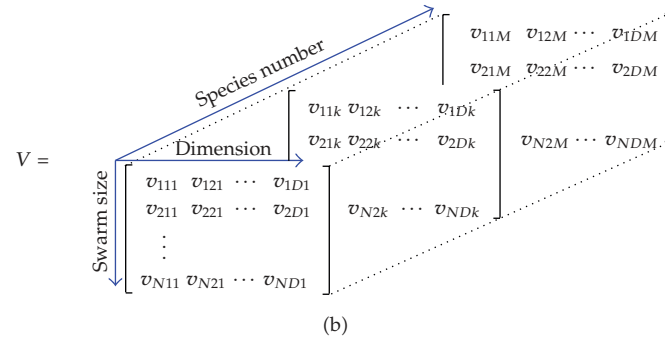
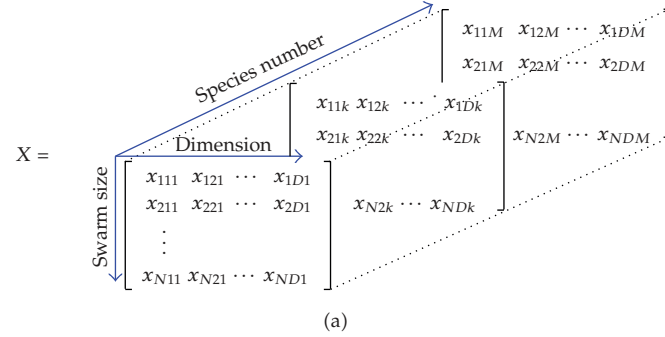


Figure 6: Matrix representation of the PS²O algorithm.

$X, V, P, S,$ and C matrixes together record all of the update information required by the PS²O algorithm. These 3D matrixes are elegantly updated in successive iteration to numerically model the hierarchical emergence. The velocity and position matrixes must be

updated element by element in each generation as:

$$V_{ijk}^{t+1} = \chi \left(V_{ijk}^t + \underbrace{\varphi_1 r_1 (P_{ijk}^t - X_{ijk}^t) + \varphi_2 r_2 (S_{jk}^t - X_{ijk}^t)}_{\text{lower level evolution}} + \underbrace{\varphi_3 r_3 (C_j^t - X_{ijk}^t)}_{\text{higher level evolution}} \right), \quad (3.5)$$

$$X_{ijk}^{t+1} = X_{ijk}^t + V_{ijk}^{t+1}$$

to obtain the intended behaviors. Note that these equations are exactly described in the previous section: the term $\varphi_1 r_1 (P_{ijk}^t - X_{ijk}^t)$ associates with each individual's own cognition, the term $\varphi_2 r_2 (S_{jk}^t - X_{ijk}^t)$ associates with cooperative coevolution within each swarm in level-1, and the term $\varphi_3 r_3 (C_j^t - X_{ijk}^t)$ associates with the symbiotic coevolution between dissimilar species in level-2.

The main difference between PS²O and PSO is the matrix implementation and the modified velocity updating equation, that is, the complexity of this new HSO algorithm is similar to the original PSO. The flowchart of the PS²O algorithm is presented in Figure 7, and according variables used in PS²O are summarized in Table 2.

4. Experimental Result and Discussion

In experimental studies, according to the no free lunch (NFL) theorem [39], a set of 17 benchmark functions (with continuous and discrete characters), which are listed in the appendix, was employed to fully evaluate the performance of the PS²O algorithm without a biased conclusion towards some chosen problems.

4.1. Experimental Setting

Experiments were conducted with four variations of PS²O (PS²Os) according to the four hierarchical interaction topologies. To fully evaluate the performance of the proposed PS²O, seven successful EA and SI algorithms were used for comparisons:

- (i) canonical PSO with constriction factor (PSO) [37],
- (ii) fully informed particle swarm (FIPS) [40],
- (iii) unified particle swarm (UPSO) [41],
- (iv) fitness-Distance-Ratio-based PSO (FDR-PSO) [42],
- (v) standard genetic algorithm (GA) [43],
- (vi) covariance matrix adaptation evolution strategy (CMA-ES) [44],
- (vii) artificial bee colony algorithm (ABC) [15].

Among these optimization tools, GA is the classical search technique that enables the fittest candidate among discrete strings to survive and reproduce based on random information search and exchange imitating the natural biological selection; the underlying idea of

Table 2: Parameters of the PS²O.

M	The number of agents (species) in level-2
N	Population size of each species in level-1
k	Species' ID counter from 1 to M
i	Individual's ID counter from 1 to N
j	Dimension counter from 1 to D
t	Generation counter from 1 to $max-gen$
X_{ijk}	The i th individual's (of the k th species) j th dimension's position value
V_{ijk}	The i th individual's (of the k th species) j th dimension's velocity value
P_{ijk}	The j th dimension value of the i th individual's (of the k th species) personal best position
S_{jk}	The j th dimension position value of the best position found by the k th level-2 species
C_j	The j th dimension value of the community best position
χ	The constriction coefficient
c_1	The learning rates for individual cognition
c_2	The learning rates for intraspecies cooperation
c_3	The learning rates for interspecies coevolution

CMA-ES is to gather information about successful search steps, and to use that information to modify the covariance matrix of the mutation distribution in a goal-directed, derandomized fashion; ABC is a recently developed SI paradigm simulating foraging behavior of bees; UPSO combined the global version and local version PSO together to construct a unified particle swarm optimizer; FIPS used all the neighbors' knowledge of the particle to update the velocity; when updating each velocity dimension, the FDR-PSO selects one other particle $nbest$, which has a higher fitness value and is nearer to the particle being updated.

In all experiments in this section, the values of the common parameters used in each algorithm such as population size and total generation number were chosen to be the same. Population size was 150 and the maximum evaluation number was 10000 for continuous functions and 1000 for discrete functions.

According to Clerc's method [37], when constriction factor is implemented as in the canonical PSO algorithm, χ is calculated from the values of the acceleration coefficients (i.e., the learning rate) c_1 and c_2 ; importantly, it is the sum of these two coefficients that determines what χ to use. This fact implies that the particle's velocity can be adjusted by any number of terms, as long as the acceleration coefficients sum to an appropriate value. Thus, the constriction factor χ in velocity formula of PS²O can be calculated by

$$\chi = \frac{2}{\left|2 - \phi - \sqrt{\phi^2 - 4\phi}\right|}, \quad (4.1)$$

where $\phi = c_1 + c_2 + c_3$, $\phi > 4$. Then the algorithm will behave properly, at least as far as its convergence and explosion characteristics, whether all of ϕ is allocated to one term, or it is divided into thirds, fourths, and so forth. Hence, for each PS²O, except when different interaction topologies are used, the parameters were set to the values $c_1 = c_2 = c_3 = 1.3667$ (i.e., $\phi = c_1 + c_2 + c_3 \approx 4.1 > 4$) and then $\chi = 0.729$, which is calculated by (4.1).

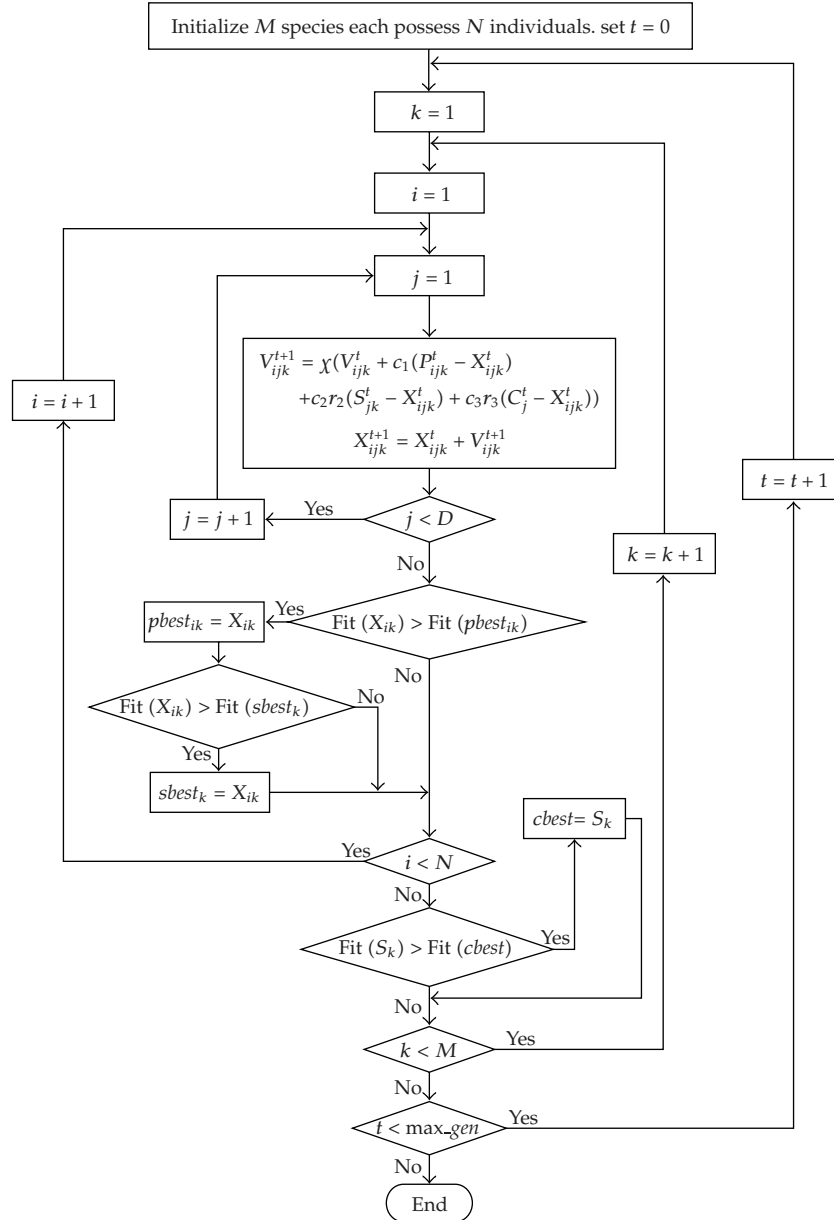


Figure 7: Flowchart of the PS²O algorithm.

All the control parameters for the other algorithms were set to be default of their original literatures. In continuous optimization experiment, for CMA-ES, initialization conditions are the same as in [44], and the number of offspring candidate solutions generated per time step is $\lambda = 4\mu$; for ABC, the limit parameter is set to be $SN \times D$, where D is the dimension of the problem and SN is the number of employed bees; for canonical PSO and UPSO, the learning rates c_1 and c_2 were both 2.05 and the constriction factor $\chi = 0.729$; for FIPS, the constriction factor χ equals to 0.729 and the U-ring topology that achieved highest success rate is used; for FDR-PSO, the inertia weight ω started at 0.9 and ended at 0.5 and a setting of $c_1 = c_2 = 2.0$ was adopted.

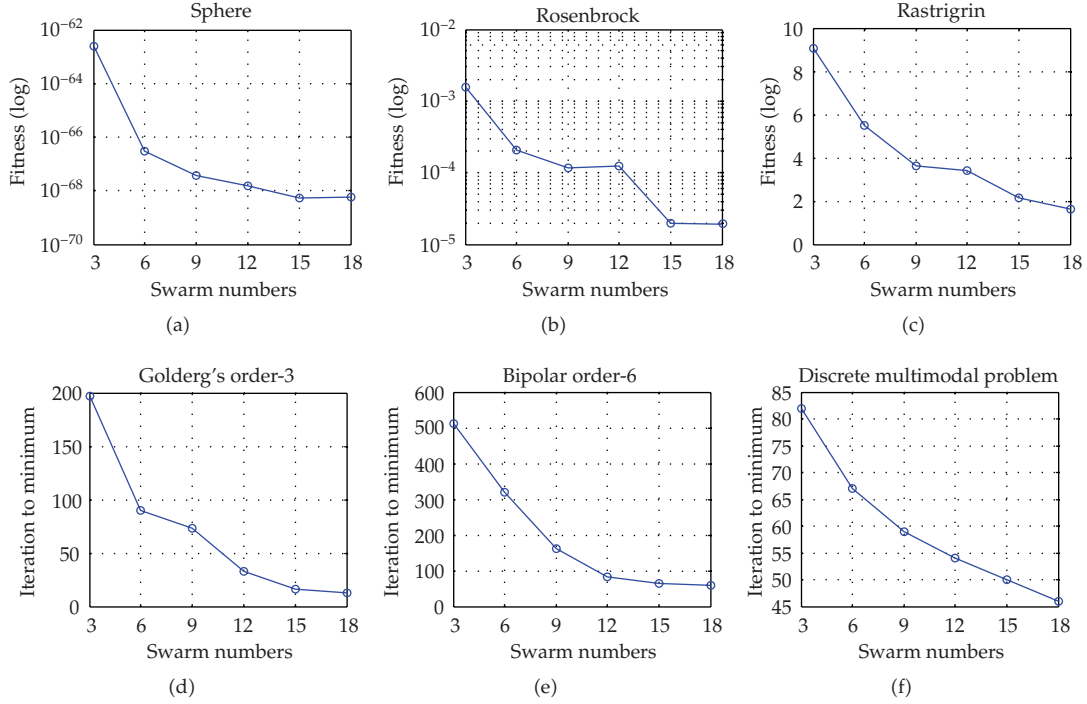


Figure 8: PS²O's results on six test functions with different swarm number M .

Since there are no literatures using CMA-ES, ABC, UPSO, FIPS, and FDR-PSO for discrete optimization so far, discrete optimization experiment just compares PS²O's with the binary version of canonical PSO and standard GA in discrete cases. For GA, single-point crossover operation with the rate of 0.8 was employed and mutation rate was set to be 0.01. For discrete PSO, the parameters were set to the values $c_1 = c_2 = 2$ and $\chi = 1$. For PS²O variants, the parameters were set to the values $c_1 = c_2 = c_3 = 2$ and $\chi = 1$. The sigmoid function S was used as the transfer function to discrete the position X of PSO and PS²O variants [45]. Then the velocity update equation remains unchanged, while the position update equation is defined by the following equation (4.2) for discrete problems:

$$\text{if}(\text{rand}() < S(V_{ijk}^{t+1})), \quad X_{ijk}^{t+1} = 1, \quad \text{else } X_{ijk}^{t+1} = 0. \quad (4.2)$$

The number of agents (species) in level-2 (i.e., swarm number M of the level-2 swarm) needs be tuned. Six benchmark functions—Sphere 10D, Rosenbrock 10D, Rastrigrin 10D, Goldberg 120D, Bipolar 60D, and Discrete multimodal problem 100D—are used to investigate the impact of this parameter. Experiments were executed with PS²O-R on Sphere, PS²O-S_R on Rosenbrock, PS²O-S on Rastrigrin, PS²O-R on Goldberg, PS²O-R_S on Bipolar, and PS²O-S on Discrete multimodal problem by changing the number of swarms and fixing each swarm size N at 10. The average test results obtained from 30 runs are plot in Figure 8. For continuous problems, the performance measurement is the average best-so-far fitness value; while for discrete cases, the performance measurement is the mean iteration to the function minimum 0. From Figure 8, we can observe that the performance of PS²O's is sensitive to the number of agents in level-2. When M increased, we obtained faster convergence velocity

and better results on all test functions. However, it can be observed that the performance improvement is not evident when $M > 15$ for most test functions. Thus, in our experiments, the parameter M of PS²O is set at 15 for all test functions (i.e., each swarm of level-2 possesses $N = 100/10 = 10$ agents of level 1).

The experiment runs 30 times, respectively, for each algorithm on each benchmark function. The numbers of generations for the 10 continuous benchmark functions were set to be 10000 and for the 7 discrete functions were 1000, respectively.

4.2. Continuous Unimodal Functions

Unimodal problems have been adopted to assess the convergence rates of optimization algorithms. We test the four PS²O variants on a set of unimodal functions (f_1 – f_5) in comparison with CMA-ES, ABC, PSO, FIPS, UPSO, and FDR-PSO algorithms. Table 3 lists the experimental results (i.e., the mean and standard deviations of the function values found in 30 runs) for each algorithm on f_1 – f_5 . Figure 9 shows the search progress of the average values found by the eight algorithms over 30 runs for f_1 – f_5 .

From Table 3 and Figure 9, the four PS²O variants converged much faster to significantly better results than all other algorithms. The PS²O-R_S, which has the heterogeneous hierarchical structures, is the fastest one for finding good results within relatively few generations. All PS²O variants were able to consistently find the minimum to functions f_1 , f_4 , and f_5 within 10000 generations.

From the comparisons between PS²O and other algorithms, we can see that, statistically, PS²O has significantly better performance on continuous unimodal functions f_1 – f_5 . From the rank values presented in Table 2, the search performance of the algorithms tested here is ordered as PS²O-S > PS²O-R > CMA-ES > PS²O-S_R > PS²O-R_S > UPSO > FDR-PSO > ABC > FIPS > PSO.

4.3. Continuous Multimodal Functions

The first four multimodal functions f_6 – f_9 are regarded as the most difficult functions to optimize since the number of local minima increases exponentially as the function dimension increases. According to the results reported in [22], the methods CL-PSO, PSO, CMA-ES, G3-PCX, DE, and the algorithms used for comparison all failed to find the minimum of the six composition function designed by Liang. Since these mentioned methods have demonstrated their excellent performance on standard benchmark functions, the six composition functions are very complex. In this paper, we only test PS²O on the first composition function f_{10} and the test on the other five composition functions will be studied in the future works. The mean and standard deviations of the function values found in 30 runs for each algorithm on each function are listed in Table 4. Figure 10 shows the search progress of the average values found by the ten algorithms over 30 runs for functions f_6 – f_{10} .

From Table 4 and Figure 10, it is clear to see that for most of the tested continuous benchmark functions, all PS²O algorithms except PS²O-R_S markedly outperformed the other algorithms. For example, PS²O-R and PS²O-S_R found the global minimum every time of run on function f_8 – f_{10} , and PS²O-R can also consistently found the minimum of f_{10} within relatively fewer generations, while the other algorithms generated poorer results on them. On functions f_6 and f_7 , the four PS²O algorithms yielded similar results to the other four algorithms. From the rank values presented in Table 4, the search performance of the algorithms tested here is ordered as PS²O-R > PS²O-S_R > ABC > PS²O-S > FIPS CMA-ES > FDR-PSO > PSO > UPSO > PS²O-R_S.

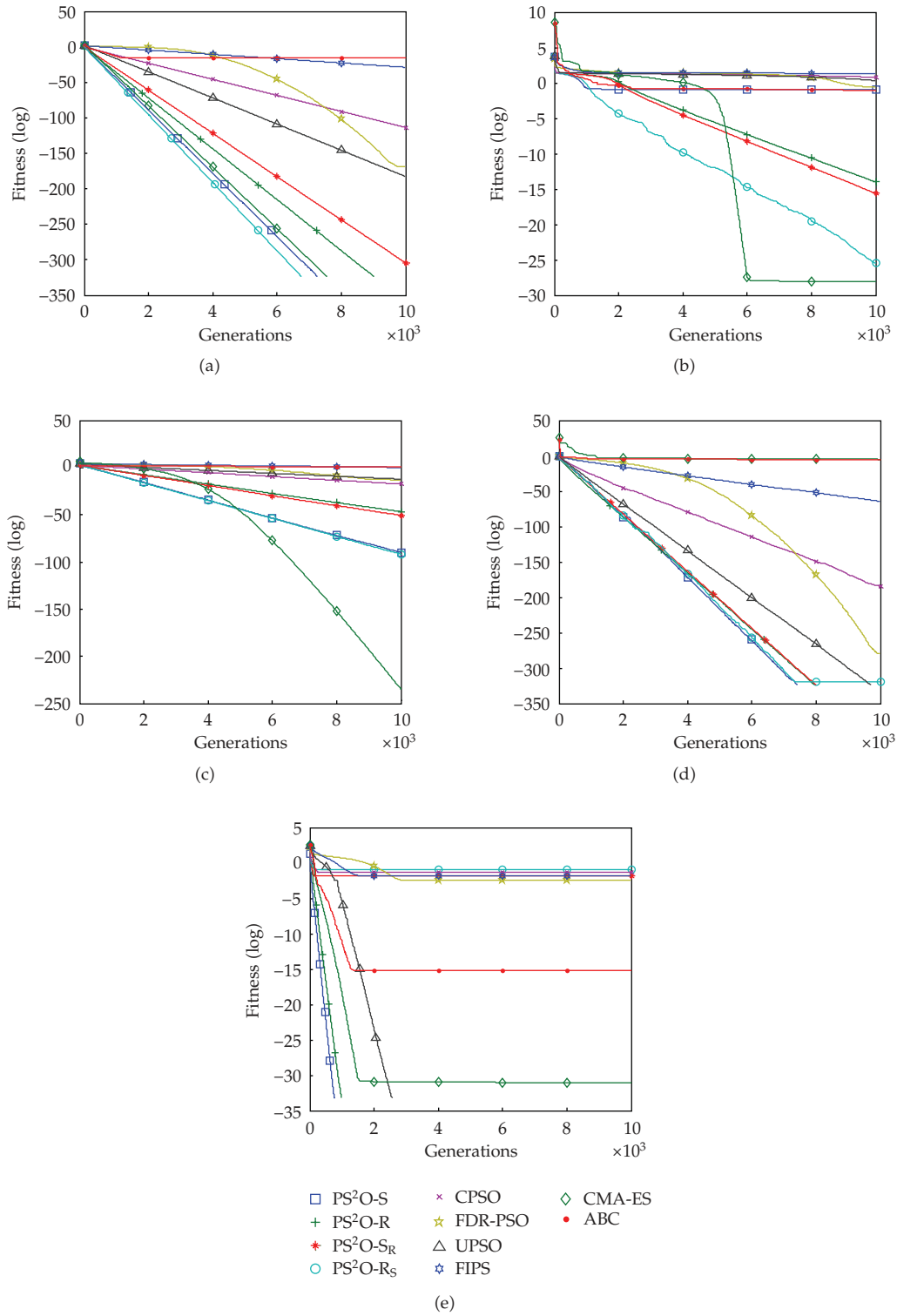


Figure 9: The median convergence results of 30D unimodal continuous functions. (a) Sphere function. (b) Rosenbrock's function. (c) Quadric function. (d) Sum of different powers. (e) Sin function.

Table 3: Performance of all algorithms on benchmark functions f_1-f_5 . In bold are the best results.

Func.	PS ² O-S	PS ² O-R	PS ² O-SR	PS ² O-R _S	PSO	FIPS	UPSO	FDR-PSO	CMA-ES	ABC
f_1	Mean	0	6.24e-305	0	2.42e-114	1.73e-029	3.71e-183	2.47e-169	0	5.52e-016
	Std	0	0	0	3.39e-114	2.29e-029	0	0	0	8.47e-017
	Rank	1	4	1	7	8	6	5	1	10
f_2	Mean	1.32e-001	1.04e-014	2.55e-016	3.71e-026	2.25e+001	2.09e+000	2.79e-001	7.91e-029	8.80e-002
	Std	7.27e-001	9.71e-015	1.70e-016	1.09e-025	4.02e-001	1.27e+000	7.69e-001	1.02e+000	1.77e-029
	Rank	6	4	3	2	9	10	8	7	1
f_3	Mean	1.04e-090	2.19e-047	1.42e-051	2.50e-092	3.51e-018	6.12e-013	4.25e-014	3.95e-235	1.16e+001
	Std	5.01e-090	6.82e-047	4.28e-051	1.02e-091	7.88e-018	1.00e+000	6.06e-013	2.32e-013	4.57e+000
	Rank	3	5	4	2	6	10	8	7	1
f_4	Mean	0	0	0	8.23e-320	4.54e-185	5.03e-065	0	8.64e-280	9.48e-006
	Std	0	0	0	0	0	2.38e-064	0	0	1.27e-005
	Rank	1	1	1	5	7	8	1	6	10
f_5	Mean	0	1.38e-002	1.14e-001	4.84e-002	1.38e-002	0	3.50e-003	9.28e-032	5.68e-016
	Std	0	3.58e-002	2.80e-001	8.04e-002	4.50e-002	0	1.89e-002	6.57e-033	8.19e-017
	Rank	1	1	7	10	9	8	1	6	4
Aver. Rank	24	2.4	3.8	4	9.5	8.8	4.8	6.2	3.4	7.6
Final Rank	1	1	4	5	10	9	6	7	3	8

Table 4: Performance of all algorithms on benchmark functions f_6-f_{10} . In bold are the best results.

Func.	PS ² O-S	PS ² O-R	PS ² O-Sr	PS ² O-R _s	PSO	FIPS	UPSO	FDR-PSO	CMA-ES	ABC
Mean	7.11e-015	4.74e-015	4.97e-015	8.32e-002	6.16e-015	2.04e-014	3.55e-015	2.74e-014	1.96e+001	4.79e-014
Std	0	1.70e-015	1.77e-015	3.17e-002	1.80e-015	1.08e-014	3.16e-015	1.03e-014	4.30e-001	6.09e-015
Rank	1	3	4	9	5	6	2	7	10	8
Mean	4.47e+001	3.25e+001	3.61e+001	4.61e+001	2.78e+001	4.13e+001	4.96e+001	2.60e+001	7.56e+000	1.64e-013
Std	1.20e+001	8.41e+000	8.96e+000	1.11e+001	5.46e+000	2.34e+001	9.80e+000	8.32e+000	2.15e+000	6.25e-014
Rank	9	5	6	8	4	7	10	3	2	1
Mean	8.37e-003	0	0	9.26e-003	1.83e-002	1.56e-003	3.47e-003	1.79e-002	0	1.03e-015
Std	1.09e-002	0	0	1.07e-002	2.66e-002	3.78e-003	4.78e-003	1.82e-002	0	1.17e-015
Rank	7	1	1	8	10	5	6	9	1	4
Mean	1.17e+000	0	0	1.22e+000	1.35e+000	2.01e-002	4.42e+000	1.58e-001	2.15e+000	0
Std	7.68e-001	0	0	8.64e-001	1.16e+000	5.58e-002	2.60e+000	4.56e-001	2.98e+000	0
Rank	6	1	1	7	8	4	10	5	9	1
Mean	0	0	0	4.00e+001	5.00e+001	3.37e+001	8.00e+001	1.00e+002	1.30e+002	3.83e-002
Std	0	0	0	8.94e+000	1.82e+002	2.58e+001	4.42e+001	4.42e+001	1.63e+002	1.21e-001
Rank	1	1	1	6	7	5	8	9	10	4
Aver. Rank	4.8	2.2	2.6	7.6	6.8	5.4	7.2	6.6	6.4	3.6
Final Rank	4	1	2	10	8	5	9	7	6	3

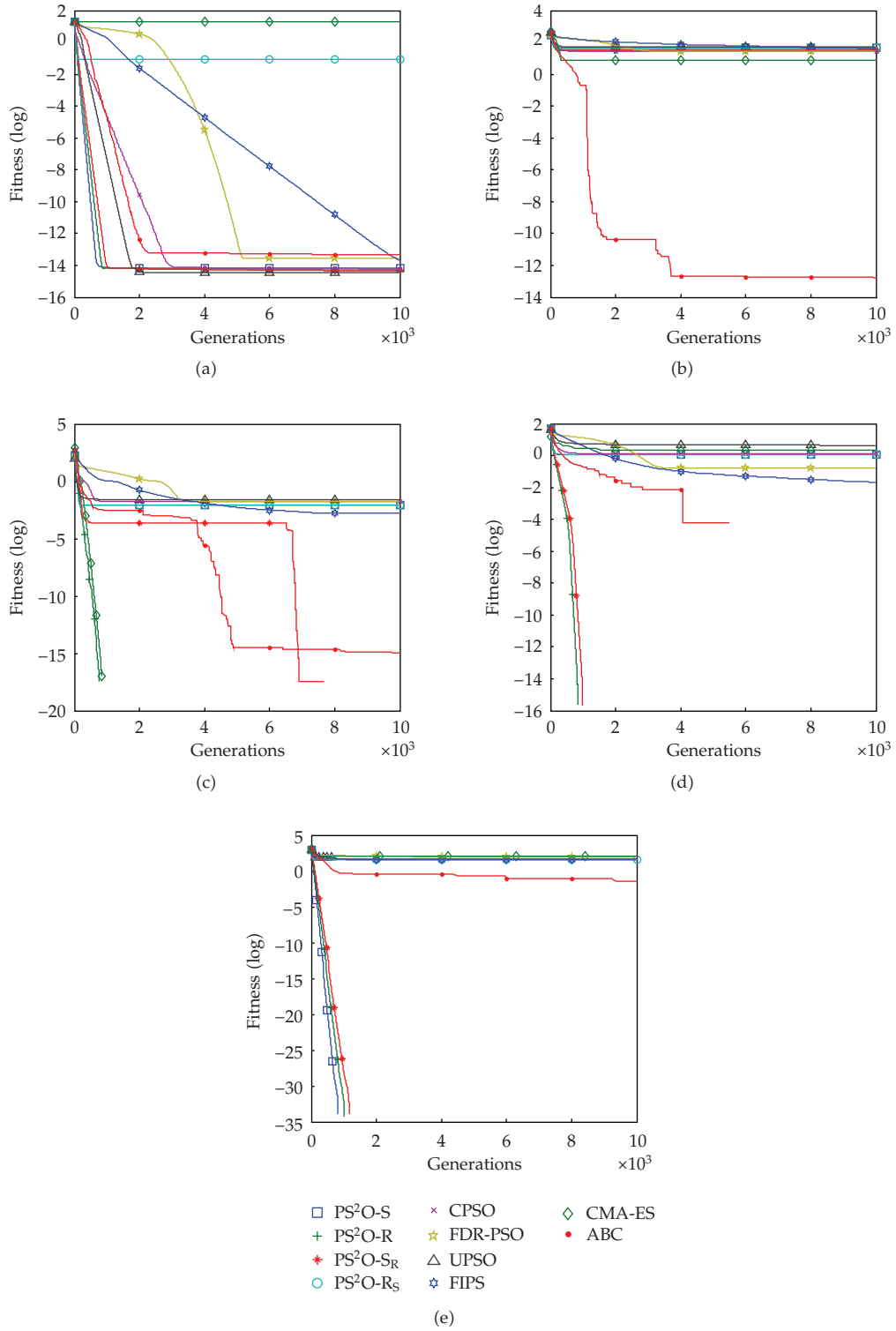


Figure 10: The median convergence results of 30D multimodal continuous functions. (a) Ackley's function. (b) Rastrigin's function. (c) Griewank's function. (d) Weierstrass function. (e) Composition function 1.

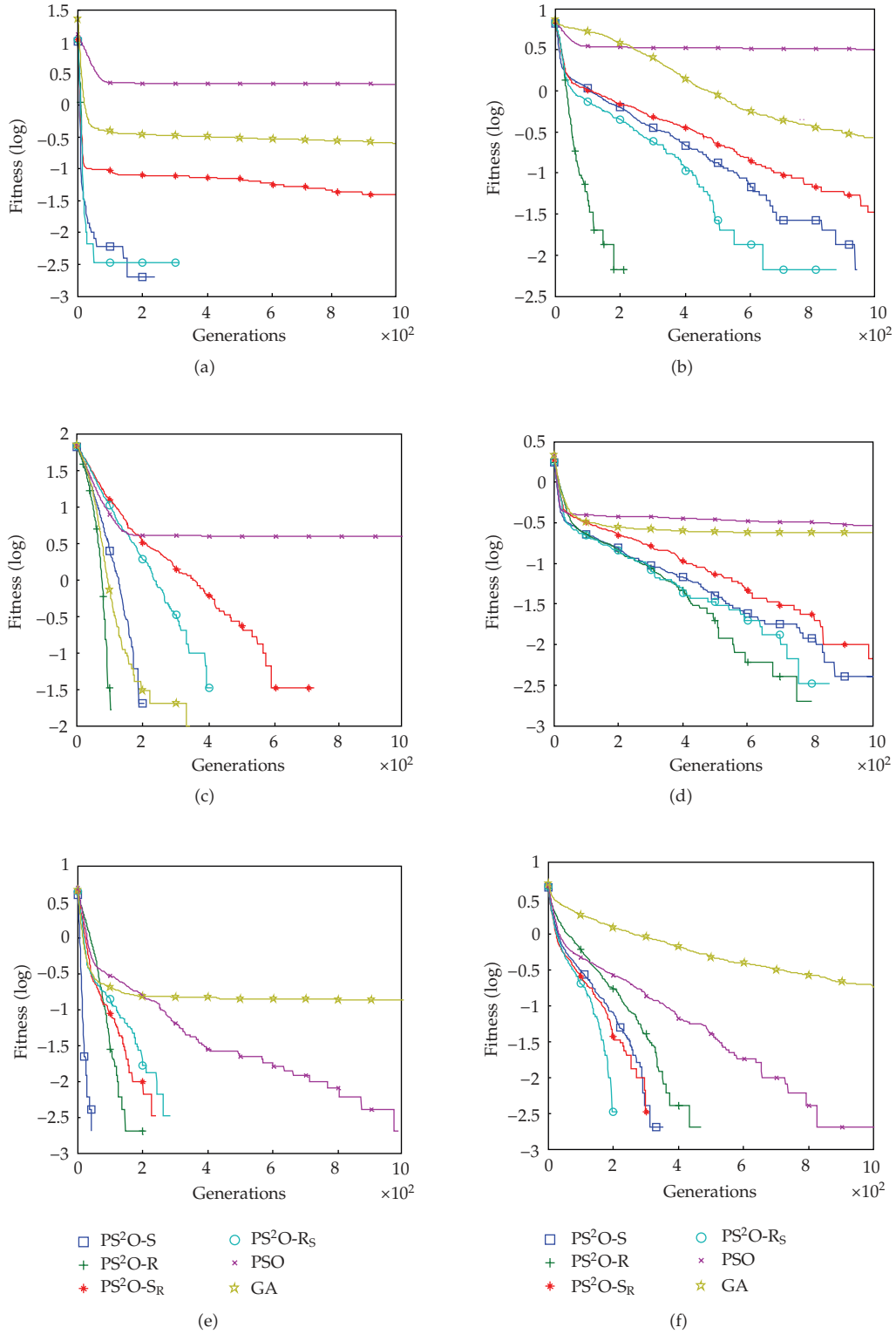


Figure 11: Continued.

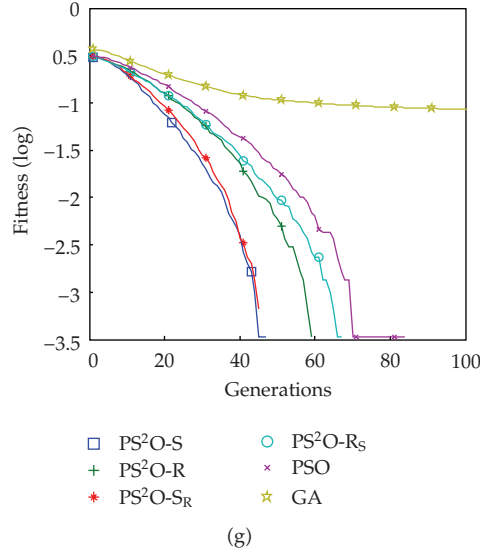


Figure 11: The median results of discrete functions. (a) Goldberg’s order-3. (b) Bipolar order-6. (c) Mulenbein’s order-5. (d) Clerc’s Zebra-3. The median convergence results of discrete functions. (e) Clerc’s order-3 problem 1. (f) Clerc’s order-3 problem 2. (g) Discrete multimodal problem.

It should be mentioned that PS²O were the only ones able to consistently find the minimum to the composition function f_1 that reported in the literatures so far.

4.4. Discrete Functions

In binary optimization, it is very easy to design some algorithms that are extremely good on some benchmarks (and extremely bad on some others) [46]. In order to fully evaluate the performance of PS²O on discrete problems, we have employed a carefully chosen set of discrete benchmark functions f_{11} – f_{17} . The results obtained by the GA, PSO, and four PS²O algorithms on each discrete benchmark function are listed in Table 5, including the mean number of iterations required to reach the minimum, mean, and standard deviations of the function values found in 30 runs. Figure 11 shows the search progress of the average values found by the five algorithms over 30 runs for functions f_{11} – f_{17} .

From the results, we can observe that PS²O obtain an obviously remarkable performance. It can be seen from Figure 11 that all PS²O variants converged greatly faster and to significantly better results than the other two algorithms for all discrete cases. From the rank values presented in Table 5, the search performance of the algorithms tested here is ordered as PS²O-R > PS²O-S > PS²O-R_S > PS²O-S_R > GA > PSO. It is worth mentioning that the PS²O-R and PS²O-R_S were able to consistently find the minimum to all discrete benchmark functions.

5. Conclusion

This paper first describes the hierarchical swarm intelligence phenomenon: the emergence of high-level intelligence aggregates properties from low-level. This mechanism is so common in nature, and provides initial evidence of the potential problem solving capabilities.

Table 5: Performance of all algorithms on benchmark functions f_{11} – f_{17} . In bold are the best results.

	Func.	PS ² O-S	PS ² O-R	PS ² O-S _R	PS ² O-R _S	PSO	GA
f_{11}	Mean	0	0	$1.67e - 002$	0	$2.14e + 000$	$1.26e - 001$
	Std	0	0	$5.31e - 002$	0	$2.60e - 001$	$1.41e - 001$
	Iter.	14	24	335	28	∞	∞
	Rank	1	2	4	3	6	5
f_{12}	Mean	0	0	$3.33e - 002$	0	$3.19e + 000$	$2.64e - 001$
	Std	0	0	$1.18e - 001$	0	$3.81e - 001$	$1.06e - 001$
	Iter.	310	69	493	328	∞	∞
	Rank	2	1	4	3	6	5
f_{13}	Mean	0	0	0	0	$3.96e + 000$	0
	Std	0	0	0	0	$1.33e + 000$	0
	Iter.	134	82	426	275	∞	335
	Rank	2	1	5	3	6	4
f_{14}	Mean	$4.00e - 003$	0	$6.70e - 003$	0	$2.94e - 001$	$2.38e - 001$
	Std	$1.98e - 002$	0	$2.54e - 002$	0	$1.33e - 002$	$8.05e - 002$
	Iter.	871	367	538	398	∞	∞
	Rank	4	1	3	2	6	5
f_{15}	Mean	0	0	0	0	0	$1.36e - 001$
	Std	0	0	0	0	0	$1.30e - 001$
	Iter.	17	89	147	107	356	∞
	Rank	1	2	4	3	5	6
f_{16}	Mean	0	0	0	0	$2.00e - 003$	$1.80e - 001$
	Std	0	0	0	0	$1.41e - 002$	$2.44e - 001$
	Iter.	203	272	187	151	∞	∞
	Rank	3	4	2	1	5	6
f_{17}	Mean	0	0	0	0	0	$7.44e - 002$
	Std	0	0	0	0	0	$1.83e - 002$
	Iter.	39	51	39	55	61	∞
	Rank	1	3	1	4	5	6
	Aver. Rank	2	2	3.29	2.71	5.57	5.14
	Final Rank	1	1	4	3	6	5

Furthermore, this paper presents the hierarchical swarm optimization (HSO) model that simulates the hierarchical swarm intelligence for function optimization. HSO is an artificial hierarchical complex system, in which agents are composed of swarms of other agents in nested hierarchies. That is, HSO is configured into several levels and each level is composed of a number of independent swarms. Note that any traditional SI methods or evolutionary algorithms can be used to manipulate any swarm of any level in HSO. HSO can be considered as not only an extension of the traditional SI model to design novel optimization algorithms, but also an open framework to hybrid traditional SI or EA algorithms to tackle hard optimization problems.

HSO model has a considerable potential in optimization domain. This paper provides some initial insights into this potential by designing a two-level HSO algorithm, namely

PS²O, which employs PSO method in each swarm of each level. This algorithm is conceptually simple, has low complexity, and is easy to implement. A set of 17 benchmark functions (including continuous and discrete cases) have been used to test four PS²O variants in comparison with GA, CMA-ES, ABC, canonical PSO, FIPS, UPSO, and FDR-PSO. The simulation results show that, for all the test functions, the PS²O reach remarkably better performance in terms of accuracy, convergence rate, and robustness than the other classical powerful algorithms.

It should be mentioned that the PS²O were the only ones that are able to consistently find the minimum of Sphere, Sum of different powers, Griewank, Weierstrass, Composition function 1 and all discrete test problems. Future research will address designing more robust and efficient HSO algorithms by integrating other SI algorithms to HSO and applying them to solve complex optimization problems. We should note that the two-level HSO algorithm is conceptually simple and easy to implement. However, in our future work, HSO algorithms with more hierarchical levels could be studied and tested on some complex benchmark functions and real-world problems.

Appendix

List of Test Functions

These benchmark functions can be grouped as unimodal continuous functions f_1 – f_5 , multimodal continuous functions f_6 – f_{10} , unimodal discrete functions f_{11} – f_{16} , and multimodal discrete function f_{17} . Functions f_1 – f_9 were test widely in evolutionary computation domain to show solution quality and convergence rate. Function f_{10} is a novel composition benchmark function developed by Liang et al. [47]. The discrete functions f_{11} – f_{17} were used in Clerc's literature [46, 48] and can be found at <http://clerc.maurice.free.fr/ps/>.

(1) Sphere Function

$$f_1(x) = \sum_{i=1}^n x_i^2. \quad (\text{A.1})$$

(2) Rosenbrock's Function

$$f_2(x) = \sum_{i=1}^n 100 \times (x_{i+1} - x_i^2)^2 + (1 - x_i)^2. \quad (\text{A.2})$$

(3) Quadric Function

$$f_3(x) = \sum_{i=1}^n \left(\sum_{j=1}^i x_j \right)^2. \quad (\text{A.3})$$

(4) Sum of Different Powers

$$f_4(x) = \sum_{i=1}^n |x_i|^{i+1}. \quad (\text{A.4})$$

(5) *Sin Function*

$$f_5(x) = \frac{\pi}{n} \left\{ 10 \sin^2 \pi x_1 + \sum_{i=1}^{n-1} (x_i - 1)^2 (1 + 10 \sin^2 \pi x_{i+1}) + (x_n - 1)^2 \right\}. \quad (\text{A.5})$$

(6) *Ackley's Function*

$$f_6(x) = -20 \exp \left(-0.2 \sqrt{\frac{1}{n} \sum_{i=1}^n x_i^2} \right) - \exp \left(\frac{1}{n} \sum_{i=1}^n \cos 2\pi x_i \right) + 20 + e. \quad (\text{A.6})$$

(7) *Rastrigrin's Function*

$$f_7(x) = \sum_{i=1}^n (x_i^2 - 10 \cos(2\pi x_i)) + 10. \quad (\text{A.7})$$

(8) *Griewank's Function*

$$f_8(x) = \frac{1}{4000} \sum_{i=1}^n x_i^2 - \prod_{i=1}^n \cos \left(\frac{x_i}{\sqrt{i}} \right) + 1. \quad (\text{A.8})$$

(9) *Weierstrass Function*

$$f_9(x) = \sum_{i=1}^n \left(\sum_{k=0}^{k \max} \left[a^k \cos(2\pi b^k (x_i + 0.5)) \right] \right) - n \left(\sum_{k=0}^{k \max} \left[a^k \cos(2\pi b^k \cdot 0.5) \right] \right), \quad (\text{A.9})$$

where $a = 0.5$, $b = 3$, and $k \max = 20$.

(10) *Composition Function 1*

The composition function 1 is constructed using 10 unimodal sphere functions. This results in an asymmetrical multimodal function with 1 global optimum and 9 local optima. The variables of the formulation can be referred to [47]

$$f_{10}(x) = \sum_{i=1}^n \left\{ w_i * \left[f_i \left(\frac{(x - o_i + o_{iold})}{\lambda_i} * M_i \right) + \text{bias}_i \right] \right\} + f_bias. \quad (\text{A.10})$$

(11) *Goldberg's Order-3*

The fitness f of a bit-string is the sum of the result of separately applying the following function to consecutive groups of three components each:

$$f_{11}(x) = \begin{cases} 0.9 & \text{if } |y| = 0, \\ 0.6 & \text{if } |y| = 1, \\ 0.3 & \text{if } |y| = 2, \\ 1.0 & \text{if } |y| = 3. \end{cases} \quad (\text{A.11})$$

If the string size (the dimension of the problem) is D , the maximum value is $D/3$ for the string 111...111. In practice, we will then use as fitness the value $D/3 - f$ so that the problem is now to find the minimum 0.

(12) *Bipolar Order-6*

The fitness f is the sum of the result of applying the following function to consecutive groups of six components each:

$$f_{12}(x) = \begin{cases} 1.0 & \text{if } |y| = 0 \text{ or } 6, \\ 0.0 & \text{if } |y| = 1 \text{ or } 5, \\ 0.4 & \text{if } |y| = 2 \text{ or } 4, \\ 0.8 & \text{if } |y| = 3. \end{cases} \quad (\text{A.12})$$

The maximum value is $D/6$. In practice, we will use as fitness the value $D/6 - f$ so that the problem is now to find the minimum 0.

(13) *Mulenbein's Order-5*

The fitness f is the sum of the result of applying the following function to consecutive groups of five components each:

$$f_{13}(x) = \begin{cases} 4.0 & \text{if } y = 00000, \\ 3.0 & \text{if } y = 00001, \\ 2.0 & \text{if } y = 00011, \\ 1.0 & \text{if } y = 00111, \\ 3.5 & \text{if } y = 11111, \\ 0.0 & \text{otherwise.} \end{cases} \quad (\text{A.13})$$

The maximum value is $3.5D/5$. In practice, the value $3.5D/5 - f$ is used as the fitness so that the problem is now to find the minimum 0.

(14) *Clerc's Zebra3*

The fitness f is the sum of the result of applying the following function to consecutive groups of three components each, if the rank of the group is even (first rank = 0):

$$f_{14}(x) = \begin{cases} 0.9 & \text{if } |y| = 0, \\ 0.6 & \text{if } |y| = 1, \\ 0.3 & \text{if } |y| = 2, \\ 1.0 & \text{if } |y| = 3. \end{cases} \quad (\text{A.14})$$

If the rank of the group is odd,

$$f_{14}(x) = \begin{cases} 0.9 & \text{if } |y| = 3, \\ 0.6 & \text{if } |y| = 2, \\ 0.3 & \text{if } |y| = 1, \\ 1.0 & \text{if } |y| = 0. \end{cases} \quad (\text{A.15})$$

The maximum value is $D/3$. In practice, we will then use as fitness the value $D/3 - f$ so that the problem is now to find the minimum 0.

(15) *Clerc's Order-3 Problem 1*

The fitness f is the sum of the result of applying the following function to consecutive groups of three components each:

$$f_{15}(x) = \begin{cases} 0.9 & \text{if } y = 000, \\ 0.6 & \text{if } y = 001, \\ 0.3 & \text{if } y = 010, \\ 1.0 & \text{if } y = 011, \\ 0.2 & \text{if } y = 100, \\ 0.4 & \text{if } y = 101, \\ 0.6 & \text{if } y = 110, \\ 0.8 & \text{if } y = 111. \end{cases} \quad (\text{A.16})$$

The maximum value is $D/3$. In practice, we will then use as fitness the value $D/3 - f$ so that the problem is now to find the minimum 0.

Table 6: Parameters of the test functions.

f	Dimensions	Initial range	x^*	$f(x^*)$
f_1	30	$[-100, 100]^D$	$[0, 0, \dots, 0]$	0
f_2	30	$[-30, 30]^D$	$[1, 1, \dots, 1]$	0
f_3	30	$[-65.536, 65.536]^D$	$[0, 0, \dots, 0]$	0
f_4	30	$[-1, 1]^D$	$[0, 0, \dots, 0]$	0
f_5	30	$[-10, 10]^D$	$[0, 0, \dots, 0]$	0
f_6	30	$[-32.768, 32.768]^D$	$[0, 0, \dots, 0]$	0
f_7	30	$[-5.12, 5.12]^D$	$[0, 0, \dots, 0]$	0
f_8	30	$[-600, 600]^D$	$[0, 0, \dots, 0]$	0
f_9	30	$[-0.5, 0.5]^D$	$[0, 0, \dots, 0]$	0
f_{10}	10	$[-5, 5]^D$	Predefined rand number distributed in the search range	0
f_{11}	120	$[0, 1]$	$[1, 1, \dots, 1]$	0
f_{12}	120	$[0, 1]$	$[0, 0, \dots, 0]$ or $[1, 1, \dots, 1]$ or $[\dots, 6 * 0, \dots, 6 * 1, \dots]$	0
f_{13}	120	$[0, 1]$	$[0, 0, \dots, 0]$	0
f_{14}	30	$[0, 1]$	$[0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, \dots]$	0
f_{15}	60	$[0, 1]$	$[0, 1, 1, 0, 1, 1, \dots]$	0
f_{16}	60	$[0, 1]$	$[0, 0, 1, 0, 0, 1, \dots]$	0
f_{17}	200	$[0, 1]$	Predefined rand number distributed in the search range	0

(16) Clerc's Order-3 Problem 2

The fitness f is the sum of the result of applying the following function to consecutive groups of three components each:

$$f_{16}(x) = \begin{cases} 0.2 & \text{if } y = 000, \\ 1.0 & \text{if } y = 001, \\ 0.3 & \text{if } y = 010, \\ 0.8 & \text{if } y = 011, \\ 0.6 & \text{if } y = 100, \\ 0.4 & \text{if } y = 101, \\ 0.6 & \text{if } y = 110, \\ 0.9 & \text{if } y = 111. \end{cases} \tag{A.17}$$

The maximum value is $D/3$. In practice, we will then use as fitness the value $D/3 - f$ so that the problem is now to find the minimum 0.

```

// Definition of the fitness landscape.
For ( $i = 1$  to number of peaks  $p$ )
  For ( $j = 1$  to number of Dimension  $D$ )
    landscape ( $i, j$ ) = rand();
  End for
End for
// the fitness  $f$  is computed as follows:
 $f = 0$ ;
For ( $i = 1$  to number of peaks  $p$ )
   $q = 0$ ;
  For ( $j = 1$  to number of Dimension  $D$ )
    If ( $x(j) = \text{landscape}(i, j)$ )
       $q = q + 1$ ;
    End if
  End for
  If ( $f < q$ )
     $f = q$ ;
  End if
End for
 $f = 1 - f/D$ .

```

Algorithm 1

(17) Discrete Multimodal Problem

This problem is defined by Algorithm 1.

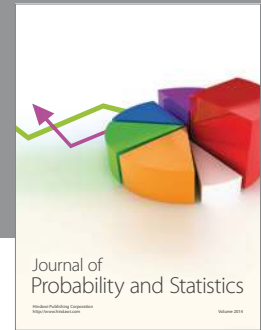
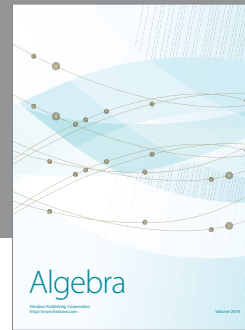
The dimensions, initialization ranges, global optima x^* , and the corresponding fitness value $f(x^*)$ of each function are listed in Table 6.

References

- [1] S. Liu and K. Passino, "Swarm intelligence: literature overview," Department of Electrical Engineering, the Ohio State University, March 2000.
- [2] D. Chowdhury, K. Nishinari, and A. Schadschneider, "Self-organized patterns and traffic flow in colonies of organisms: from bacteria and social insects to vertebrates," *Phase Transitions*, vol. 77, no. 5–7, pp. 601–624, 2004.
- [3] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm Intelligence: From Natural to Artificial Systems*, Oxford University Press, Oxford, UK, 1999.
- [4] M. Dorigo, V. Maniezzo, and A. Coloni, "Ant system: optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 26, no. 1, pp. 29–41, 1996.
- [5] M. Dorigo, G. Di Caro, and L. M. Gambardella, "Ant algorithms for discrete optimization," *Artificial Life*, vol. 5, no. 2, pp. 137–172, 1999.
- [6] M. Dorigo and L. M. Gambardella, "Ant colony system: a cooperative learning approach to the traveling salesman problem," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 53–66, 1997.
- [7] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in *Proceedings of IEEE International Conference on Neural Networks*, vol. 4, pp. 1942–1948, Piscataway, NJ, USA, 1995.
- [8] J. Kennedy and R. C. Eberhart, *Swarm Intelligence*, chapter 9, Morgan Kaufmann Publishers, San Francisco, Calif, USA, 2001.
- [9] K. E. Parsopoulos and M. N. Vrahatis, "On the computation of all global minimizers through particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 211–224, 2004.
- [10] D. N. Wilke, *Analysis of the particle swarm optimization algorithm*, M.S. thesis, Department of Mechanical and Aeronautical Engineering, University of Pretoria, Pretoria, South Africa, 2005.

- [11] K. M. Passino, "Biomimicry of bacterial foraging for distributed optimization and control," *IEEE Control Systems Magazine*, vol. 22, no. 3, pp. 52–67, 2002.
- [12] D. Dasgupta, *Artificial Immune Systems and Their Applications*, Springer, Heidelberg, Germany, 1999.
- [13] E. Benjamini, G. Sunshine, and S. Leskowitz, *Immunology: A Short Course*, Wiley, New York, NY, USA, 3rd edition, 1996.
- [14] D. Karaboga, "An idea based on honeybee swarm for numerical optimization," Tech. Rep. TR06, Computer Engineering Department, Engineering Faculty, Erciyes University, 2005.
- [15] B. Basturk and D. Karaboga, "An artificial bee colony (ABC) algorithm for numeric function optimization," in *Proceedings of IEEE Swarm Intelligence Symposium*, Indianapolis, Ind, USA, May 2006.
- [16] R. C. Eberhart and Y. Shi, "Particle swarm optimization: developments, applications and resources," in *Proceedings of the IEEE Conference on Evolutionary Computation (ICEC '01)*, vol. 1, pp. 81–86, Piscataway, NJ, USA, 2001.
- [17] S. Mishra, "A hybrid least square-fuzzy bacterial foraging strategy for harmonic estimation," *IEEE Transactions on Evolutionary Computation*, vol. 9, no. 1, pp. 61–73, 2005.
- [18] M. Tripathy, S. Mishra, and G. K. Venayagamoorthy, "Bacteria foraging: a new tool for simultaneous robust design of UPFC controllers," in *Proceedings of IEEE International Joint Conference on Neural Networks (IJCNN '06)*, pp. 2274–2280, Vancouver, Canada, July 2006.
- [19] M. Tripathy, S. Mishra, L. L. Lai, and Q. P. Zhang, "Transmission loss reduction based on FACTS and bacteria foraging algorithm," in *Proceedings of the 9th International Conference on Parallel Problem Solving from Nature (PPSN '06)*, vol. 4193 of *Lecture Notes in Computer Science*, pp. 222–231, September 2006.
- [20] C. A. Coello and N. C. Cores, "A parallel implementation of the artificial immune system to handle constraints in genetic algorithms: preliminary results," in *Proceedings of the Congress on Evolutionary Computation (CEC '02)*, pp. 819–824, Honolulu, Hawaii, USA, 2002.
- [21] A. Swiecicka, F. Seredynski, and A. Y. Zomaya, "Multiprocessor scheduling and rescheduling with use of cellular automata and artificial immune system support," *IEEE Transactions on Parallel and Distributed Systems*, vol. 17, no. 3, pp. 253–262, 2006.
- [22] V. Maniezzo and A. Colorni, "The ant system applied to the quadratic assignment problem," *IEEE Transactions on Knowledge and Data Engineering*, vol. 11, no. 5, pp. 769–778, 1999.
- [23] N. Minar, R. Burkhard, C. Langton, and M. Askenazi, "The swarm simulation system: a toolkit for building multi-agent simulations," Tech. Rep. 96-06-042, Santa Fe Institute, Santa Fe, NM, USA, 1996, <http://www.santafe.edu/media/workingpapers/96-06-042.pdf>.
- [24] C. Anderson, N. R. Franks, and D. W. McShea, "The complexity and hierarchical structure of tasks in insect societies," *Animal Behaviour*, vol. 62, no. 4, pp. 643–651, 2001.
- [25] H. A. Simon, *Sciences of the Artificial*, MIT Press, Cambridge, Mass, USA, 1969.
- [26] N. Boccara, *Modeling Complex Systems*, Graduate Texts in Contemporary Physics, Springer, New York, NY, USA, 2004.
- [27] C. Anderson and D. W. McShea, "Intermediate-level parts in insect societies: adaptive structures that ants build away from the nest," *Insectes Sociaux*, vol. 48, no. 4, pp. 291–301, 2001.
- [28] M. J. Jacobson, "Problem solving about complex systems: differences between experts and novices," in *Proceedings of the 4th International Conference of the Learning Sciences*, pp. 14–21, Erlbaum, Mahwah, NJ, USA, 2000.
- [29] F. Atay and J. Jost, "On the emergence of complex systems on the basis of the coordination of complex behavior of their elements," Tech. Rep. 04-02-005, Santa Fe Institute, Santa Fe, NM, USA, 2003, <http://www.santafe.edu/media/workingpapers/04-02-005.pdf>.
- [30] S. Halloy, "A theoretical framework for abundance distributions in complex systems," Tech. Rep. 99-04-031, Santa Fe Institute, Santa Fe, NM, USA, 1999, <http://www.santafe.edu/media/workingpapers/99-04-031.pdf>.
- [31] S. Salthe, *Evolving Hierarchical Systems: Their Structure and Representation*, Columbia University Press, New York, NY, USA, 1985.
- [32] H. H. Pattee, Ed., *Hierarchy Theory: The Challenge of Complex Systems*, George Braziller, New York, NY, USA, 1973.
- [33] H. N. Chen, Y. L. Zhu, and K. Y. Hu, "Cooperative bacterial foraging optimization," *Discrete Dynamics in Nature and Society*, vol. 2009, Article ID 815247, 17 pages, 2009.
- [34] H. Chen and Y. Zhu, "Optimization based on symbiotic multi-species coevolution," *Applied Mathematics and Computation*, vol. 205, no. 1, pp. 47–60, 2008.
- [35] H. Chen, Y. Zhu, and K. Hu, "Multi-colony bacteria foraging optimization with cell-to-cell communication for RFID network planning," *Applied Soft Computing Journal*, vol. 10, no. 2, pp. 539–547, 2010.

- [36] J. Kennedy, "Particle swarm: social adaptation of knowledge," in *Proceedings of the IEEE Conference on Evolutionary Computation (ICEC '97)*, pp. 303–308, Indianapolis, Ind, USA, April 1997.
- [37] M. Clerc and J. Kennedy, "The particle swarm-explosion, stability, and convergence in a multidimensional complex space," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 58–73, 2002.
- [38] H. L. Zhang, C. H. C. Leung, and G. K. Raikundalia, "Classification of intelligent agent network topologies and a new topological description language for agent networks," in *Proceedings of the 4th IFIP International Federation for Information Processing*, vol. 228, pp. 21–31, Springer, 2006.
- [39] D. H. Wolpert and W. G. Macready, "No free lunch theorems for search," Tech. Rep. SFI-TR-95-02-010, Santa Fe Institute, Santa Fe, NM, USA, 1995.
- [40] R. Mendes, J. Kennedy, and J. Neves, "The fully informed particle swarm: simpler, maybe better," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 204–210, 2004.
- [41] K. E. Parsopoulos and M. N. Vrahatis, "UPSO—a unified particle swarm optimization scheme," in *Proceedings of the International Conference of Computational Methods in Sciences and Engineering (ICCMSE '04)*, vol. 1 of *Lecture Series on Computational Sciences*, pp. 868–873, Zeist, The Netherlands, 2004.
- [42] T. Peram, K. Veeramachaneni, and C. K. Mohan, "Fitness-distance-ratio based particle swarm optimization," in *Proceedings of the IEEE Swarm Intelligence Symposium*, pp. 174–181, 2003.
- [43] S. Sumathi, T. Hamsapriya, and P. Surekha, *Evolutionary Intelligence: An Introduction to Theory and Applications with Matlab*, Springer, New York, NY, USA, 2008.
- [44] N. Hansen and A. Ostermeier, "Completely derandomized self-adaptation in evolution strategies," *Evolutionary Computation*, vol. 9, no. 2, pp. 159–195, 2001.
- [45] J. Kennedy and R. Eberhart, "A discrete binary version of the particle swarm optimization algorithm," in *Proceedings of the Conference on Systems, Man, and Cybernetics (SMC '97)*, pp. 4104–4109, Orlando, Fla, USA, 1997.
- [46] M. Clerc, "Binary particle swarm optimisers: toolbox, derivations, and mathematical insights," February 2005, <http://clerc.maurice.free.fr/pso/>.
- [47] J. J. Liang, P. N. Suganthan, and K. Deb, "Novel composition test functions for numerical global optimization," in *Proceedings of IEEE Swarm Intelligence Symposium (SIS '05)*, pp. 71–78, Pasadena, Calif, USA, June 2005.
- [48] M. Clerc, *Discrete Particle Swarm Optimization, New Optimization Techniques in Engineering*, Springer, New York, NY, USA, 2004.



Hindawi

Submit your manuscripts at
<http://www.hindawi.com>

