Hindawi Publishing Corporation
Discrete Dynamics in Nature and Society
Volume 2012, Article ID 904815, 24 pages
doi:10.1155/2012/904815

Research Article

A Multiswarm Optimizer for Distributed Decision Making in Virtual Enterprise Risk Management

Yichuan Shao,[1,2] Xingjia Yao,[2] Liwei Tian,[3] and Hanning Chen[4]

[1] *College of Information Science and Engineering, Shenyang University, Shenyang 110044, China*
[2] *School of New Energy Engineering, Shenyang University of Technology, Shenyang 110036, China*
[3] *Science and Technology Agency, Shenyang University, Shenyang 110036, China*
[4] *Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang 110016, China*

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

Received 27 October 2011; Accepted 16 February 2012

Academic Editor: Leonid Shaikhet

We develop an optimization model for risk management in a virtual enterprise environment based on a novel multiswarm particle swarm optimizer called PS^{2}O. The main idea of PS^{2}O is to extend the single population PSO to the interacting multiswarms model by constructing hierarchical interaction topology and enhanced dynamical update equations. With the hierarchical interaction topology, a suitable diversity in the whole population can be maintained. At the same time, the enhanced dynamical update rule significantly speeds up the multiswarm to converge to the global optimum. With five mathematical benchmark functions, PS^{2}O is proved to have considerable potential for solving complex optimization problems. PS^{2}O is then applied to risk management in a virtual enterprise environment. Simulation results demonstrate that the PS^{2}O algorithm is more feasible and efficient than the PSO algorithm in solving this real-world problem.

1. Introduction

Swarm Intelligence (SI) is an innovative artificial intelligence technique for solving complex optimization problems. This discipline is inspired by the collective behaviors of social animals such as fish schools, bird flocks, and ant colonies. In SI systems, 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 behavior and properties.

In recent years, many algorithmic SI methods were designed to deal with practical problems [1–5]. Among them, the most successful is Particle Swarm Optimization (PSO) that drew inspiration from the biological swarming behaviors observed in flocks of birds, schools

of fish, and even human social behavior [6–8]. PSO is a population-based optimization tool, which could be implemented and applied easily to solve various function optimization problems. As a problem-solving technique, the main strength of PSO is its fast convergence, which compares favorably with Evolutionary Algorithms (EAs) and other global optimization algorithms [9–12]. However, when solving complex multimodal problems, PSO suffers from the following drawback [13]: as a population evolves, all individuals suffer premature convergence to the local optimum in the first generations. This leads to low population diversity and adaptation stagnation in successive generations. However, such loss of population diversity is not observed in natural systems. Because populations of species interact with one another in natural ecosystems, these species form biological communities which are large social systems typically consist of both heterogeneous and homogeneous aspects. The interaction between species and the complexity of their relationships in these communities exemplify what is meant by the term "symbiosis." According to different symbiotic interrelationships, symbiotic coevolution can be classified into several categories: mutualism, commensalism, parasitism, and competition. We found that all these types are suitable to be incorporated into the standard PSO model to improve PSO's performance on complex optimization problems. This should be a general extension of PSO with the purpose of accurately representing as many different forms of symbiotic coevolution as possible.

Thus, inspired by symbiotic cooperation (i.e., mutualism coevolution) phenomenon in nature, this paper proposed a novel multiswarm particle swarm optimizer called PS^{2}O, which extend the single population PSO to interacting multiswarms model by constructing hierarchical interaction topologies and enhanced dynamical update equations. In PS^{2}O, we implement a hierarchical interaction topology that consists of two levels (i.e., individual level and swarm level), in which information exchanges take place permanently. Each individual of the proposed model evolves based on the knowledge integration of itself (associate with individual's own cognition), its swarm members (associate social interaction within each swarm), and its symbiotic partners from other swarm (associate heterogeneous cooperation between different swarms). That is, we extend the control law (i.e., the dynamic update equation) of the canonical PSO model by adding a significant ingredient, which takes into account the symbiotic coevolution (or heterogeneous cooperation) between different swarms. By incorporating this new degree of complexity, PS^{2}O can accommodate a considerable potential for solving more complex problems. Here we provide some initial insights into this potential by evaluating PS^{2}O on both mathematical benchmark functions and a complex real-world problem-risk management in a virtual enterprise (VE). The 5 benchmark functions used in our experiments have been widely employed by other researchers to evaluate their algorithms [14–16]. In this paper, the risk management problem in VE is modeled as a distributed decision-making (DDM) system. This novel risk management model is a complex hierarchical optimization problem with two levels, namely, the top model and the base model, which take care of the continuous decision variables and the discrete ones, respectively. The simulation results, which are compared to other methods, are reported in this paper to show the merits of the proposed algorithm.

The paper is organized as follows. Section 2 gives a review of the canonical PSO algorithm and several multi-swarm PSO variants. Section 3 describes the proposed multiswarm coevolution algorithm. In Section 4, it will be shown that PS^{2}O outperforms the canonical PSO and its variants on 5 benchmark test functions. Section 5 describes the risk management optimization model in VE and a detailed design algorithm of risk management by PS^{2}O. The simulation result of risk management in a VE based on PS^{2}O compared with canonical PSO is also presented in this section. Finally, conclusions are drawn in Section 6.

2. Review of Canonical Particle Swarm Optimization

The canonical PSO is a population-based technique, similar in some respects to evolutionary algorithms, except that potential solutions (particles) move, rather than evolve, through the search space. The rules (or particle dynamics) that govern this movement are inspired by models of swarming and flocking [7]. Each particle has a position and a velocity, and experiences linear spring-like attractions towards two attractors:

(i) its previous best position,

(ii) best position of its neighbors.

In mathematical terms, the ith particle is represented as $x_i = (x_{i1}, x_{i2}, \ldots, x_{iD})$ in the D-dimensional space, where $x_{id} \in [l_d, u_d]$, $d \in [1, D]$, l_d, u_d are the lower and upper bounds for the dth dimension, respectively. The rate of velocity for particle i is represented as $v_i = (v_{i1}, v_{i2}, \ldots, v_{iD})$ is clamped to a maximum velocity $V_{\max}$ which is specified by the user. In each time step t, the particles are manipulated according to the following equations:

$$v_{id}(t) = \chi\left(v_{id}(t-1) + R_1 c_1 \left(p_{id} - x_{id}(t-1)\right) + R_2 c_2 \left(p_{gd} - x_{id}(t-1)\right)\right), \tag{2.1}$$

$$x_{id}(t) = x_{id}(t-1) + v_{id}(t), \tag{2.2}$$

where R_1 and R_2 are random values between 0 and 1, c_1 and c_2 are learning rates, which control how far a particle will move in a single iteration, p_{id} is the best position found so far of the ith particle, p_{gd} is the best position of any particles in its neighborhood, and χ is called constriction factor [17], given by

$$\chi = \frac{2}{\left|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}\right|}, \tag{2.3}$$

where $\varphi = c_1 + c_2,\ \varphi > 4$.

Kennedy and Eberhart [18] proposed a binary PSO in which a particle moves in a state space restricted to zero and one on each dimension, in terms of the changes in probabilities that a bit will be in one state or the other. The velocity formula (2.1) remains unchanged except that x_{id}, p_{id}, and p_{gd} are integers in $\{0,\ 1\}$, and v_{id} must be constrained to the interval $[0.0, 1.0]$. This can be accomplished by introducing a sigmoid function $S(v)$, and the new particle position is calculated using the following rule:

$$\text{if rand} < S(v_{id}),\ \text{then } x_{id} = 1;\ \text{else } x_{id} = 0, \tag{2.4}$$

where rand is a random number selected from a uniform distribution in [0.0, 1.0] and the function $S(v)$ is a sigmoid limiting transformation as follows:

$$S(v) = \frac{1}{1 + e^{-v}}. \tag{2.5}$$

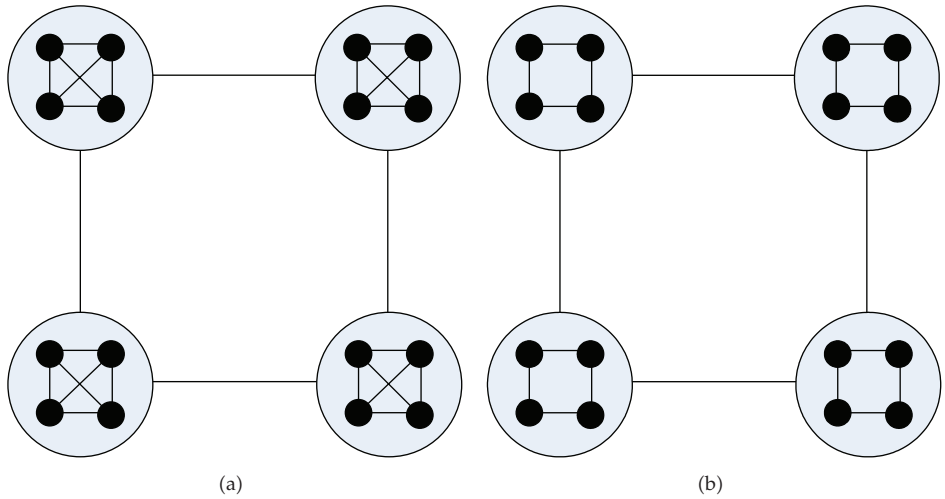

Figure 1: Hierarchical topologies of the multiswarm.

3. PS^{2}O Algorithm

Straight PSO uses the analogy of a single-species population and the suitable definition of the particle dynamics and the particle information network (interaction topology) to reflect the social evolution in the population. However, the situation in nature is much more complex than what this simple metaphor seems to suggest. Indeed, in biological populations there is a continuous interplay between individuals of the same species, and also encounters and interactions of various kinds with other species [19]. The points at issue can be clearly seen when one observes such ecological systems as symbiosis, host-parasite systems, and prey-predator systems, in which two organisms mutually support each other, one exploits the other, or they fight against each other. For instance, mutualistic relations between plants and fungi are very common. The fungus invades and lives among the cortex cells of the secondary roots and, in turn, helps the host plant absorb minerals from the soil. Another well-known example is the "association" between the Nile crocodile and the Egyptian plover, a bird that feeds on any leeches attached to the crocodile's gums, thus keeping them clean. This kind of "cleaning symbiosis" is also common in fish.

Inspired by mutualism phenomenon, we extend the single population PSO to the interacting multi-swarms model by constructing hierarchical information networks and enhanced particle dynamics. In our multi-swarms approach, the interaction occurs not only between the particles within each swarm but also between different swarms. That is, the information exchanges on a hierarchical topology of two levels (i.e., the individual level and the swarm level). Many patterns of connection can be used in different levels of our model. The most common ones are rings, two-dimensional and three-dimensional lattices, stars, and hypercubes. Two example hierarchical topologies are illustrated in Figure 1. In Figure 1(a), four swarms at the upper level are connected by a ring, while each swarm (possesses four individual particles at the lower level) is structured as a star. While in Figure 1(b), both levels are structured as rings. Then, we suggest in the proposed model that each individual moving through the solution space should be influenced by three attractors:

(i) its own previous best position,

(ii) best position of its neighbors from its own swarm,

(iii) best position of its neighbor swarms.

In mathematical terms, our multi-swarm model is defined as a triplet $\langle P, T, C\rangle$, where $P = \{S_1, S_2, \ldots, S_M\}$ is a collection of M swarms, and each swarm possesses a members set $S_k = \{X_1^k, X_2^k, \ldots, X_N^k\}$ of N individuals. T is the hierarchical topology of the multi-swarm. C is the enhanced control low of the particle dynamics, which can be formulated as

$$\begin{aligned} v_{id}^k(t) = \chi\Big(v_{id}^k(t-1) + R_1c_1\Big(p_{id}^k - x_{id}^k(t-1)\Big) + R_2c_2\Big(p_{gd}^k - x_{id}^k(t-1)\Big) \\ + R_3c_3\Big(p_{gd}^\theta - x_{id}^k(t-1)\Big)\Big), \end{aligned} \tag{3.1}$$

$$x_{id}^k(t) = x_{id}^k(t-1) + v_{id}^k(t), \tag{3.2}$$

where x_{id}^k represents the position of the ith particle of the kth swarm, p_{id}^k is the personal best position found so far by x_{id}^k, p_{gd}^k is the best position found so far by this particle's neighbors within swarm k, p_{gd}^θ is the best position found so far by the other swarms in the neighborhood of swarm k (here θ is the index of the swarm which the best position belongs to), c_1 are the individual learning rates, c_2 are the social learning rate between particles within each swarm, c_3 are the social learning rate between different swarms, and $R_1, R_2, R_3 \in \Re^d$ are random vectors uniformly distributed in [0, 1]. Here, the term $R_1c_1(p_{id}^k - x_{id}^k)$ is associated with cognition since it takes into account the individual's own experiences; the term $R_2c_2(p_{gd}^k - x_{id}^k)$ represents the social interaction within swarm k; the term $R_3c_3(p_{gd}^\theta - x_{id}^k)$ takes into account the symbiotic coevolution between dissimilar swarms.

When constriction factor is implemented as in the canonical PSO above, 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 φ of these two coefficients that determines what χ to use [17]. 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 PS^{2}O can be calculated by

$$\chi = \frac{2}{\left|2 - \phi - \sqrt{\phi^2 - 4\phi}\right|}, \tag{3.3}$$

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.

We should note that, for solving discrete problems, we still use (2.4) and (2.5) to discrete the position vectors in PS^{2}O algorithm. The pseudocode for the PS^{2}O algorithm is listed in Table 1. The flowchart of the PS^{2}O algorithm is presented in Figure 2, and according variables used in PS^{2}O are summarized in Table 2.

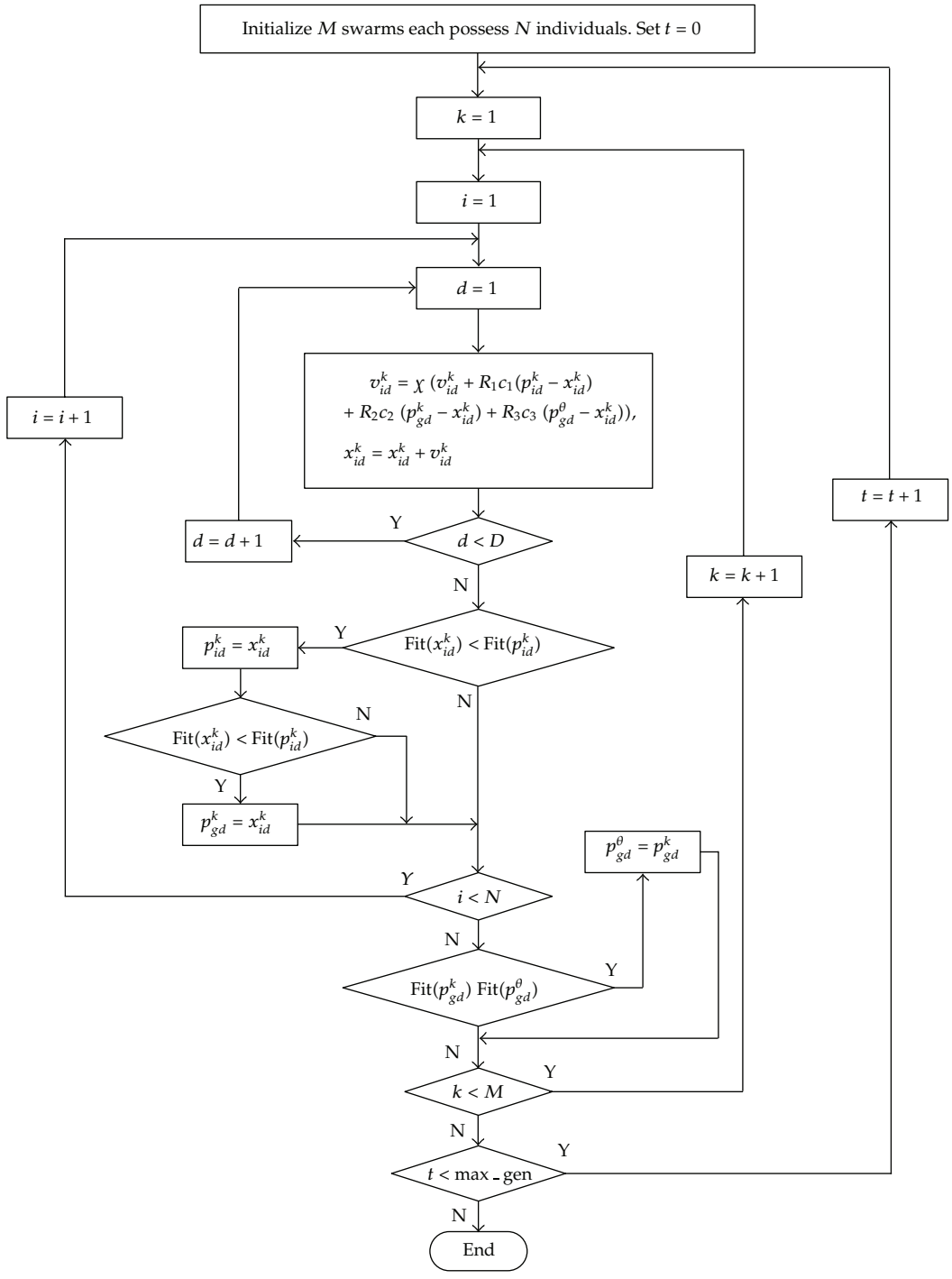

Figure 2: The flowchart of the PS^{2}O algorithm.

Table 1: Pseudocode for the PS^{2}O algorithm.

```
Set t := 0;
INITIALIZE. Randomize n swarms each possesses m particles;
WHILE (the termination conditions are not met)
    FOR (each swarm k)
        Find in the kth swarm neighborhood, the point with the best fitness;
        Set this point as p_gd^θ;
      FOR (each particle i of swarm k)
          Find in the particle neighborhood, the point with the best fitness;
          Set this point as p_gd^k;
          Update particle velocity using (4.1);
          Update particle position using (4.2);
      END FOR
    END FOR
    Set t := t + 1;
END WHILE
```

Table 2: List of variables used in PS^{2}O.

M	The number of swarms
N	Population size of each swarm
k	Swarm's ID counter from 1 to n
i	Individual's ID counter from 1 to m
d	Dimension of the problem
t	Generation counter from 1 to max generation
θ	The index of the best neighbor swarm of the kth swarm
x_{id}^k	The ith individual's (of the kth swarm) dth dimension's value
p_{id}^k	The ith individual's personal best (of the kth swarm)
p_{gd}^k	The best neighbor position of x_{id}^k in the kth swarm
p_{gd}^θ	The best neighbor position of the kth swarm

4. Benchmark Test

4.1. Test Function

A set of 5 benchmark functions, which are commonly used in evolutionary computation literature [16, 20] to show solution quality and convergence rate, was employed to evaluate the PS^{2}O algorithm in comparison to others. The first problem is the unimodal Sphere function that is easy to solve. The second problem is the Rosenbrock function, which has a narrow valley from the perceived local optima to the global optimum and can be treat as a multimodal problem. The remaining three functions are multimodal problem. Griewank's function has a $\prod_{i=1}^{D} \cos(x_i/\sqrt{i})$ component causing linkages among variables, thereby making it difficult to reach the global optimum. The Weierstrass function is continuous but differentiable only on a set of points. The composition functions are a set of novel challenging problems, which are constructed using some basic benchmark function with a random located global optimum and several randomly located deep local optima. The Gaussian function is used to combine

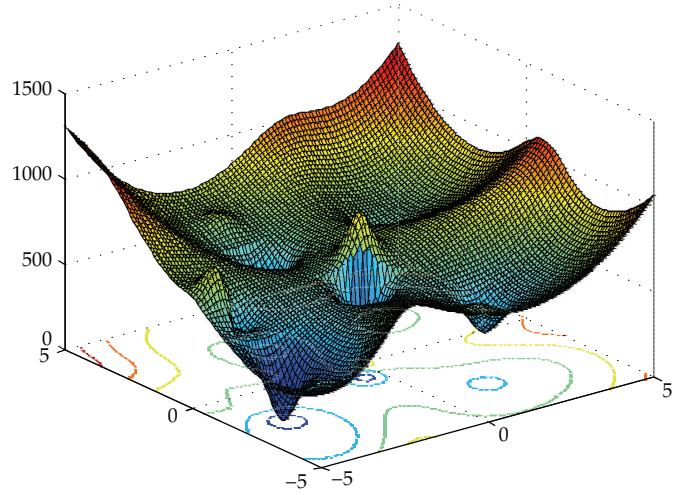

Figure 3: The landscape maps of CF1 function.

the basic functions and blur the function structures. CF1 is constructed using 10 sphere functions which is an asymmetrical multimodal function with 1 global optimum and 9 local optima (the landscape of CF1 is illustrated in Figure 3). The variables of the CF1 formulation can be referred to [20]. The formulas of these functions are presented below.

(1) Sphere function:

$$f_1(x) = \sum_{i=1}^{D} x_i^2. \tag{4.1}$$

(2) Rosenbrock function:

$$f_2(x) = \sum_{i=1}^{D} 100 \times \left(x_{i+1} - x_i^2\right)^2 + (1 - x_i)^2. \tag{4.2}$$

(3) Griewank function:

$$f_3(x) = \frac{1}{4000} \sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1. \tag{4.3}$$

(4) Weierstrass function:

$$f_4(x) = \sum_{i=1}^{D} \left(\sum_{k=0}^{k\max} \left[a^k \cos\left(2\pi b^k (x_i + 0.5)\right) \right] \right) - n \left(\sum_{k=0}^{k\max} \left[a^k \cos\left(2\pi b^k \bullet 0.5\right) \right] \right), \tag{4.4}$$

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

(5) Composition function 1:

$$f_5(x) = \sum_{i=1}^{n} \left\{ w_i * \left[f_i \left(\frac{(x - o_i + o_{\text{iold}})}{\lambda_i} * M_i \right) + \text{bias}_i \right] \right\} + f\_\text{bias}, \tag{4.5}$$

where n is the number of basic function, w_i is weight value for each $f_i(x)$, $f_i(x)$ is ith basic function used to construct the composition function (here f_1–f_{10}: Sphere Function), o_i is new shifted optimum position for each $f_i(x)$, o_{iold} is old optimum position for each $f_i(x)$, λ_i is used to stretch or compress the function, M_i is orthogonal rotation matrix for each $f_i(x)$, and bias_i is defines which optimum is global optimum.

4.2. Experimental Setting

Experiments were conducted with PS^{2}O compared with four successful variants of PSO:

(i) local version of PSO with constriction factor (CPSO) [21];

(ii) fully informed particle swarm (FIPS) [22];

(iii) unified particle swarm (UPSO) [23];

(iv) fitness-distance-ratio-based PSO (FDR-PSO) [24].

Among these variations, 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; the FDR-PSO selects one other particle, which has a higher fitness value and is nearer to the particle being updated, to update each velocity dimension.

The number of swarms M needs be tuned. Three 10D functions, namely Sphere, Rosenbrock, and Griewank, are used to investigate the impact of this parameter. Experiments were executed by changing the number of swarms and fixing each swarm size at 10. The average test results obtained form 30 runs are plotted in Figure 4. From Figure 4, we can observe that the performance of PS^{2}O is influenced by M. When M increases, we obtained faster convergence velocity and better results.

For fair comparison, the population size of all algorithms used in our experiments was set at 100 (all the swarms of PS^{2}O include the same particle numbers of 10). The maximum velocity of all PSO variants was set to be 5% of the search space for unimodal functions and 50% for multimodal functions. 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 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. For PS^{2}O, the interaction topology illustrated in Figure 1(b) is used; the constriction factor in PS^{2}O is also used with $\chi = 0.729$ according to Clerc's method; correspondingly, the ϕ coefficient must sum to 4.1 and then the learning rates $c_1 = c_2 = c_3 = \phi/3 \approx 1.3667$.

4.3. Simulation Results

The experiment runs 50 times, respectively, for each algorithm on each benchmark function of 30 dimensions. The numbers of generations were set to be 10000. The representative results

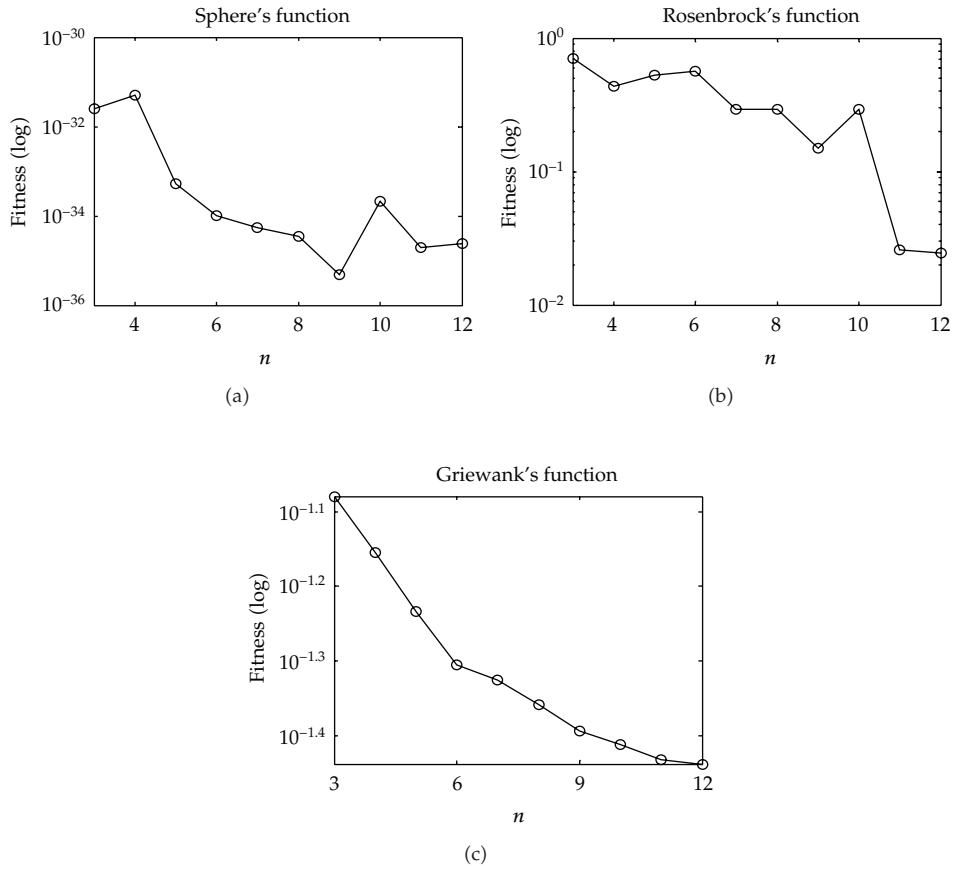

Figure 4: PS^{2}O's results on 3 test functions with different M.

obtained are presented in Table 4, including the best, worst, mean, and standard deviation of the function values found in 50 runs. Figures 5, 6, 7, 8, and 9 presents the evolution process for all algorithms according to the reported results in Table 3.

From the results, we can observe that the PS^{2}O algorithm obtains an obviously remarkable performance. We can see it clearly that PS^{2}O converged with greatly faster speed to significantly better results than the other PSO variants for both unimodal and multimodal cases. It should be mentioned that the PS^{2}O were the only ones able to consistently find the minimum of the Sphere function, Griewank's function, Weierstrass function, and Composition function 1, while the other algorithms generated poorer results on them. The result on Rosenbrock obtained by PS^{2}O is also very good. Since a result within 40.0 on 30 D Rosenbrock reported in other EA and SI works is considered well, the PS^{2}O algorithm's performance on Rosenbrock function is remarkable good.

With the hierarchical interaction topology, a suitable diversity in the whole population can be maintained. At the same time, the enhanced dynamical update rule significantly speeds up the multi-swarm to converge to the global optimum. Because of this, the PS^{2}O performs considerably better than many PSO variants.

Table 3: Performance of all algorithms. In bold are the best results.

Func. (dim.30)		PS^{2}O	CPSO	FIPS	UPSO	FDR-PSO
f_1	Best	**0**	$2.4787e-116$	$1.1874e-030$	$3.4459e-185$	$6.9665e-190$
	Worst	**0**	$1.3486e-113$	$9.7762e-029$	$1.9929e-182$	$7.4365e-168$
	Mean	**0**	$2.4205e-114$	$1.7391e-029$	$3.7072e-183$	$2.4789e-169$
	Std	**0**	$3.3966e-114$	$2.2995e-029$	**0**	**0**
f_2	Best	$\mathbf{1.5203e-015}$	5.8889	17.4217	0.7070	0.0012
	Worst	$\mathbf{5.1336e-014}$	7.4375	23.4450	4.0368	4.0879
	Mean	$\mathbf{1.0412e-014}$	6.6172	22.5407	2.0983	0.2797
	Std	$\mathbf{9.7087e-015}$	0.4028	1.2748	0.7696	1.0224
f_3	Best	**0**	**0**	**0**	**0**	**0**
	Worst	**0**	0.1152	0.0123	0.0388	0.0737
	Mean	**0**	0.0183	0.0016	0.0347	0.0179
	Std	**0**	0.0266	0.0038	0.0478	0.0182
f_4	Best	**0**	$2.8242e-005$	**0**	**0**	**0**
	Worst	**0**	3.7591	0.2856	8.1054	1.5086
	Mean	**0**	1.3510	0.0201	4.4244	0.1581
	Std	**0**	1.1606	0.0558	2.6022	0.4569
f_5	Best	**0**	0.0051	**0**	**0**	**0**
	Worst	**0**	100.0071	45.5672	0.0467	300.00
	Mean	**0**	50.0061	33.7051	0.0136	100.00
	Std	**0**	70.7121	25.8645	0.0143	141.42

5. Virtual Enterprise Risk Management Based on PS^{2}O

A virtual enterprise (VE) [25] is a temporary consortium of autonomous, diverse, and possibly geographically dispersed organizations that pool their resource to meet short-term objectives and exploit fast changing market trends. A VE is a dynamic alliance of member companies (owner and partners), which join to take advantage of a market opportunity. Each member company will provide its own core competencies in areas such as marketing, engineering, and manufacturing to the VE. When the market opportunity has passed, the VE is dissolved. In a VE environment, there are various sources of risks that may threaten the security of VE, such as market risk, credit risk, operational risk, and others [26]. Recently, risk management of a VE has attracted much research attention.

5.1. The Two-Level Optimization Model for Risk Management in a Virtual Enterprise

In this paper, the two-level risk manage model suggested by Huang and Lu [27] is employed to evaluate the performance of the proposed PS^{2}O algorithm. This model can be described as a two-level Distributed Decision Making (DDM) system that is depicted in Figure 10(a).

In the top level, the decision maker is the owner who allocates the budget (i.e., the risk cost investment) to each member of VE. The decision variables are therefore given by $I = (I_0, I_1, \ldots, I_n)$. Here I_0 denotes the budget to owner and I_i $(i = 1, 2, \ldots, n)$ represents the budget to Partner i. That is, there are $n + 1$ members in a VE. Then the top-level objective of risk management in a VE is to best allocate the budget of each member so as to minimize

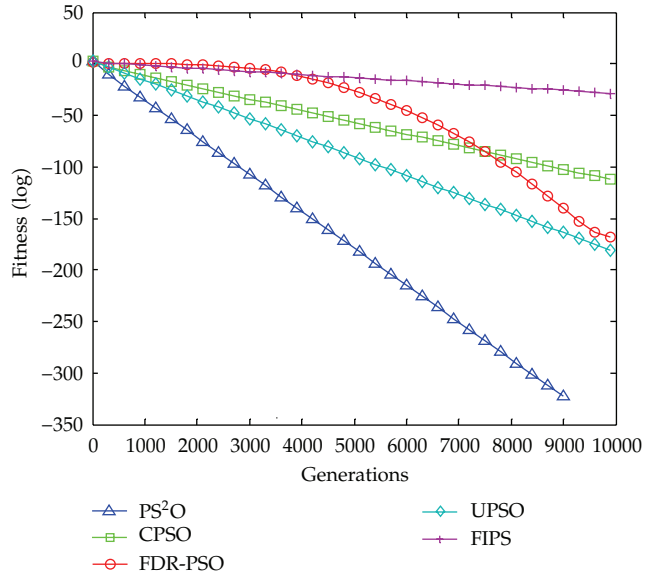

Figure 5: The median convergence results of Sphere function.

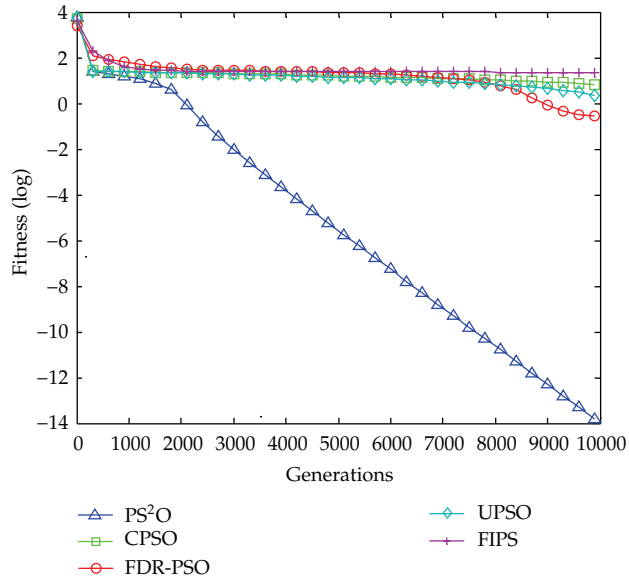

Figure 6: The median convergence results of Rosenbrock function.

the total risk level of the VE. The top-level model can be formulated as a continuous optimization problem that is given in what follows:

$$\begin{aligned} \min_{I} \quad & F_T(I) = \sum_{i=0}^{n} w_i R_i(I_i), \\ \text{s.t.} \quad & \sum_{i=0}^{n} I_i \le I_{\max}, \\ & R_i(I_i) \le R_{\max}, \end{aligned} \tag{5.1}$$

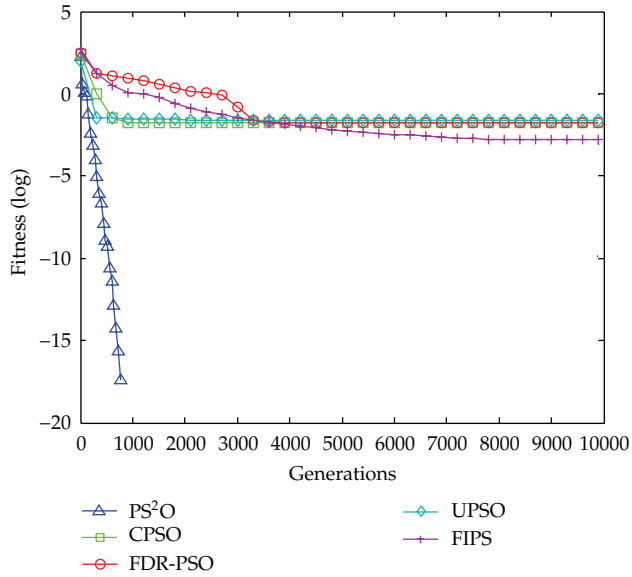

Figure 7: The median convergence results of Griewank function.

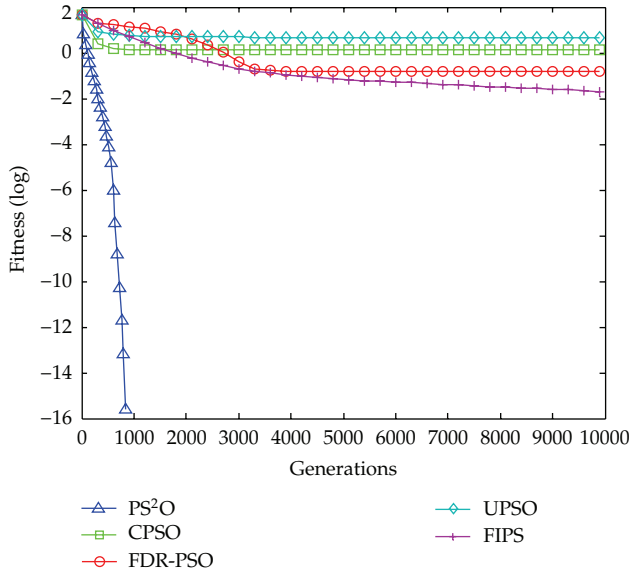

Figure 8: The median convergence results of Weierstrass function.

where $R_i(I_i)$ is the risk level of ith member under risk cost investment I_i, w_i represents the weight of member i, $I_{\max}$ is the maximum total investment budget, and $R_{\max}$ stands for the maximum risk level for each member in the VE.

In the base level, the partners of VE are making their decisions in view of the top-level's instruction (i.e., the budget to partners). The base-level risk management is that the decision makers select the optimal series of risk control actions $A_i = (a_1^i, a_2^i, \ldots, a_m^i)$ for each partner i $(i = 1, 2, \ldots, n)$ to minimize the risk level with respect to the allocated budget I_i.

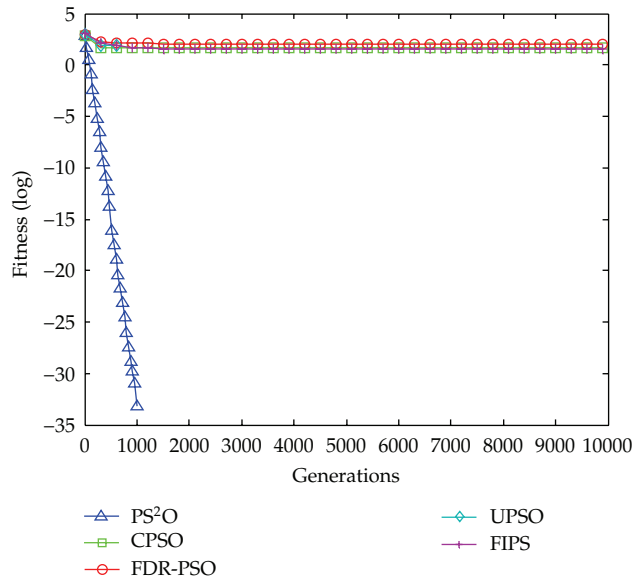

Figure 9: The median convergence results Composition function 1.

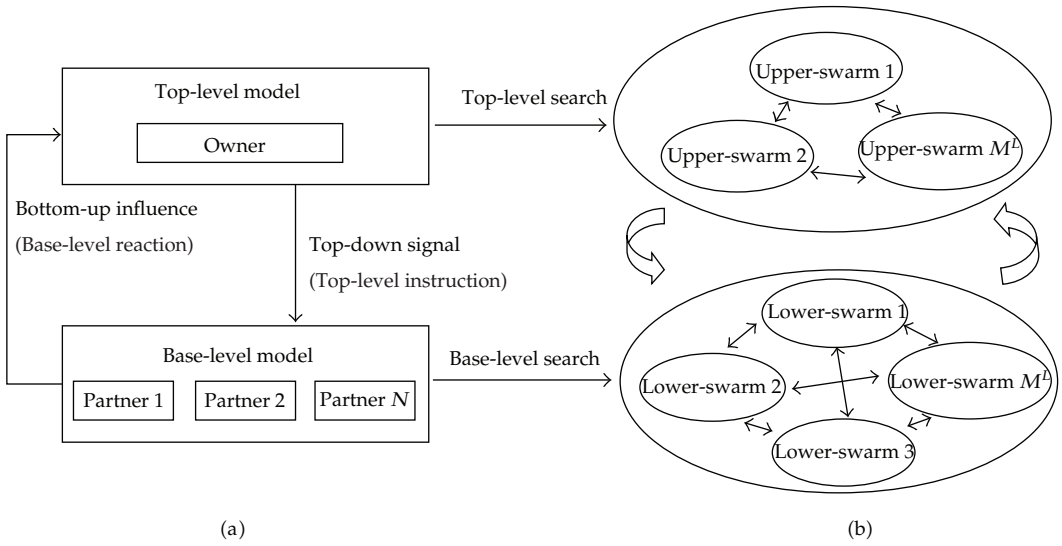

Figure 10: The exchange of information between (a) the owner and partners, (b) the upper-swarms, and lower-swarms.

Here m is the number of risk factors that affect each partner's security. Then the base-level model can be formulated as a discrete optimization problem that is given in what follows:

$$\begin{aligned} \min_{A} \quad & F_B(A) = \sum_{i=1}^{n} w_i R_i(A_i I_i), \\ \text{s.t.} \quad & \sum_{j=1}^{m} C_j^i\left(a_j^i\right) \le I_i, \\ & a_j^i \in \{0, 1, 2, \ldots, W\}, \end{aligned} \tag{5.2}$$

where $R_i(A_i|I_i)$ is the risk level of ith partner under risk control action A_i with respect to the top-level investment budget I_i, $C_j^i(a_j^i)$ represents the cost of partner i under the risk control action a_j^i for the risk factor j, and W stands for the number of available actions for each risk factor of each partner.

5.2. Risk Management in VE Based on PS^{2}O

The detailed design of risk management algorithm based on PS^{2}O is introduced in this section. Since the optimizing problem has a two-level hierarchical structure, this risk management algorithm is composed of two types of swarms that search in different levels, respectively, namely, the upper swarm and the lower swarm. The algorithm design reflects a two-phase searching process as Figure 10(b) illustrates. In the top-level searching process, the upper swarms that are designed based on the continuous PS^{2}O, search a continuous space for the investment budget allocation for all VE members. While the lower swarms, which are designed based on the discrete PS^{2}O, receive information from upper swarms, and must search the discrete space for a best action combination for risk management of all partners. The overall searching process can be described as follows.

(a) Definition of Continuous Particle

In each upper-swarm, each particle has a dimension equal to $n + 1$ (i.e., the number of VE members). Each particle has a real number representation and is a possible allocation of investment budget for all members. The ith particle of the kth upper swarm is defined as follows:

$$X_{ik}^T = \left(x_{i1k}^T, x_{i2k}^T, \ldots, x_{i(n+1)k}^T\right), \quad x_{ijk}^T \in \Re. \tag{5.3}$$

For example, a real-number particle (286.55, 678.33, 456.78, 701.21, 567.62) is a possible allocation of investment budget of 5 VE members. The first bit means that the owner received investment of 286.55 units. The 2 to 5 bits mean that the amount of investment allocated to partner 1 to 4 is 678.33, 456.78, 701.21, and 567.62 respectively.

(b) Definition of Discrete Particle

For the lower swarms, in order to appropriately represent the action combination by a particle, we design an "action-to-risk-to-partner" representation for the discrete particle. Each discrete particle in each lower swarm has a dimension equal to the number of $n \times m \times W$, here W is the number of available actions for each risk factor, m is the number of risk factors of each partner, and n is the number of VE partners. The ith particle of the kth lower swarm is defined as follows:

$$X_{ik}^L = \left(x_{i(111)k}^L, x_{i(112)k}^L, \ldots, x_{i(n\times m\times W)k}^L\right), \quad x_{i(\alpha\beta\gamma)k}^L \in \{0,1\}, \tag{5.4}$$

where $x_{i(\alpha\beta\gamma)k}^L$ equals 1 if the risk factor β of VE partner α is solved by the γth action and 0 otherwise. For example, set $n = 2$, $m = 4$, $W = 4$, suppose the action combination of two

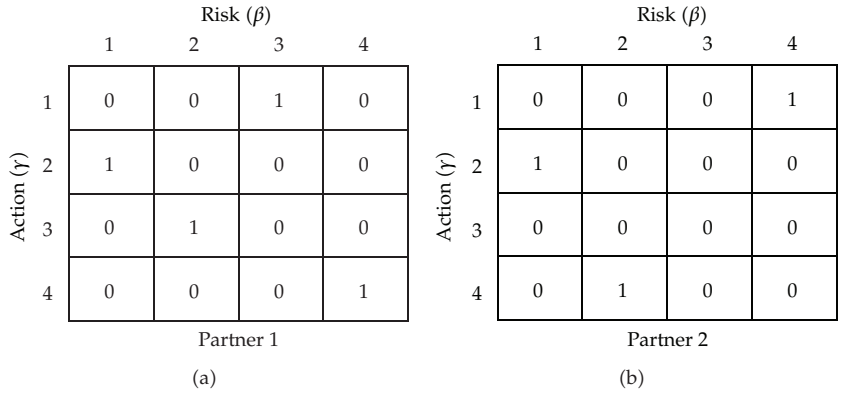

Figure 11: Definition of a discrete particle (2314, 2401) for the action combination of two partners.

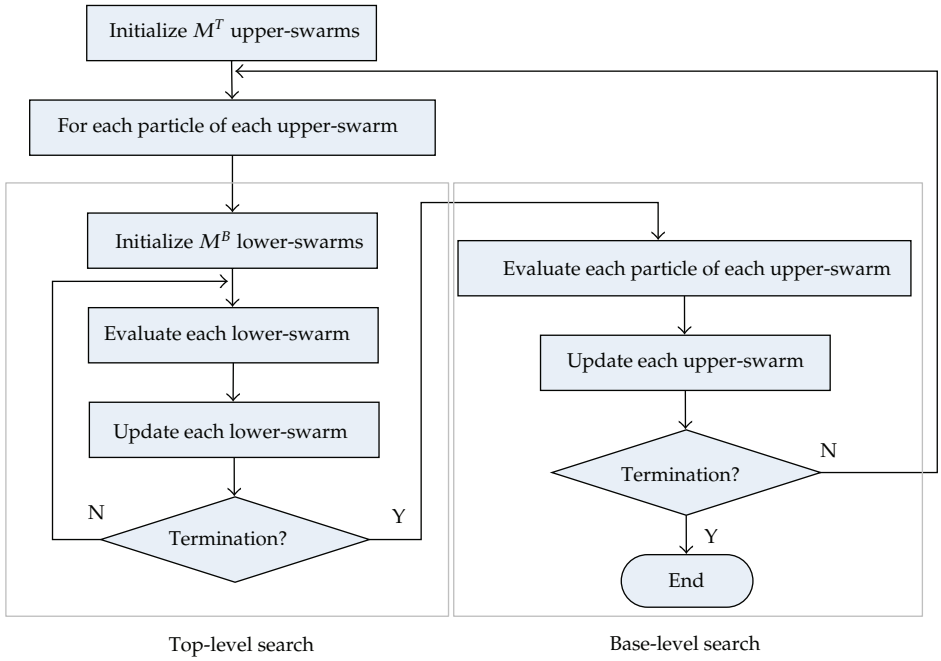

Figure 12: The risk management algorithm based on PS^{2}O.

partners is (2314, 2401), here 0 stands for no action is selected for the third risk factor of the second partner in VE. By our definition, we have $x^L_{i(112)k} = x^L_{i(123)k} = x^L_{i(131)k} = x^L_{i(144)k} = x^L_{i(212)k} = x^L_{i(224)k} = x^L_{i(241)k} = 1$ and all other $x^L_{i(\alpha\beta\gamma)k} = 0$ (see Figure 11).

(2) Risk Management Procedure

The processing performed by this algorithm is best illustrated in the diagram given in Figure 12.

Step 1. The first step in top level is to randomly initialize M^T upper swarms each possesses N^T particles (totally $M^T \times N^T$ individuals). Each particle X_{ik}^T in the top level is an *instruction* and is communicated to the base level to drive a base level search process (Steps 2–4).

Step 2. For each top-level *instruction* X_{ik}^T, the base-level randomly initialize M^B lower swarms each possesses N^B particles (totally $M^B \times N^B$ individuals). At each iteration in base level, for each particle X_{ik}^L (i.e., the ith particle of the kth lower-swarm), evaluate its fitness using the base-level optimization function as follows:

$$\begin{aligned} F_B\left(X_{ik}^L\right) = \sum_{\alpha=1}^{n} w_\alpha R_\alpha\left(X_{i\alpha k}^L \mid X_{i\alpha k}^T\right) = \sum_{\alpha=1}^{n}\sum_{\beta=1}^{m}\sum_{\lambda=1}^{l} w_\alpha u_\beta f_{\beta\lambda}\left(\left|x_{i\alpha\beta k}^L\right|\right) d_\lambda, \\ +\varphi\sum_{\alpha=1}^{n}\left(\sum_{\beta=1}^{m} C_\beta^\alpha\left(\left|x_{i\alpha\beta k}^L\right|\right) - x_{i\alpha k}^T\right)^+, \end{aligned} \tag{5.5}$$

where u_β is the weight of the risk factor β, d_λ is the value corresponding to the risk rating λ, l is the number of risk ratings, and φ is the punishment coefficient. $x_{i\alpha\beta k}^L = (x_{i(\alpha\beta 1)k}^L, x_{i(\alpha\beta 2)k}^L, \ldots, x_{i(\alpha\beta W)k}^L)$ and $|x_{i\alpha\beta k}^L|$ is defined as the position index of 1 in $x_{i\alpha\beta k}^L$. For example, if $x_{i\alpha\beta k}^L = (0010)$, the value of $|x_{i\alpha\beta k}^L|$ is 3. $f_{\beta\lambda}(|x_{i\alpha\beta k}^L|)$ is a convex decreasing function, which is approximated by

$$f_{\beta\lambda}\left(\left|x_{i\alpha\beta k}^L\right|\right) = \exp\left(-\theta_{\beta\lambda}\left|x_{i\alpha\beta k}^L\right|\right) \tag{5.6}$$

to assesses the probability of risk occurrence at risk rating λ under action $|x_{i\alpha\beta k}^L|$. Here the parameter $\theta_{\beta\lambda}$ is used to describe the effects of different risk factors under different risk ratings. The cost of the action $C_\beta^\alpha(|x_{i\alpha\beta k}^L|)$ is assumed to be a concave increasing function of the corresponding action, which is approximated by

$$C_\beta^\alpha\left(\left|x_{i\alpha\beta k}^L\right|\right) = 100\left(1 - \exp\left(-\tau_\beta^\alpha\left|x_{i\alpha\beta k}^L\right|\right)\right) \tag{5.7}$$

and the parameter τ_β^α describes the effects of different risk factors of different partners. The notation $(\cdot)^+$ is defined as follows:

$$(x)^+ = \begin{cases} x & \text{if } x > 0, \\ 0 & \text{else.} \end{cases} \tag{5.8}$$

Step 3. Compare the evaluated fitness values and select *pbest*, *sbest*, and *cbest* for each lower swarm. Then update the velocity $v_{i(\alpha\beta\gamma)k}^L$ of each base level particle according to (3.1). For our problem, each partner can only select one action for each risk factor or do nothing with this

Begin
Let X^{temp} be a zero vector that has a dimension equal to $n \times m \times W$.
For ($\alpha = 1$ *to* n)
  For ($\beta = 1$ *to* m)
    For ($\gamma = 1$ *to* W)
      If (rand $\leq p_{i(\alpha\beta\gamma)k}$)
        //Action γ is selected for risk β of partner α
        $X_{\alpha\beta\gamma}^{\text{temp } t} = 1$;
        Break;
      End if
    End for
  End for
End for
$X_{ik}^{L} = X^{\text{temp}}$
END

Algorithm 1

factor. In order to take care of this problem, for each particle, action γ is selected for risk factor β of partner α according to following probability:

$$p_{i(\alpha\beta\gamma)k} = \frac{s\left(v_{i(\alpha\beta\gamma)k}^{L}\right)}{\sum_{\gamma=1}^{W} s\left(v_{i(\alpha\beta\gamma)k}^{L}\right)}. \tag{5.9}$$

Then the position of each base-level particle is updated by Algorithm 1.

Step 4. (1) Particle Representation

The base-level search process is repeated until the maximum number of base-level iteration is met. Then send the last best base-level decision variable X_{ik}^{L*} to the top-level for the fitness computation of the top-level particle X_{ik}^{T}.

Step 5. With the base-level reaction X_{ik}^{L*}, each top-level particle X_{ik}^{T} is evaluated by the following top-level fitness function:

$$\begin{aligned} F_T\left(X_{ik}^{T}\right) = \sum_{\alpha=0}^{n} w_\alpha R_\alpha\left(X_{i(\alpha+1)k}^{T}\right) = w_0 R_0\left(X_{i1k}^{T}\right) + F_B\left(X_{ik}^{L*}\right) + \phi\left(\sum_{\alpha=1}^{n+1} x_{i\alpha k}^{T} - I_{\max}\right)^{+} \\ + \eta \sum_{\alpha=1}^{n}\left(\sum_{\beta=1}^{m}\sum_{\lambda=1}^{l} w_\alpha u_\beta f_{\beta\lambda}\left(\left|x_{i\alpha\beta k}^{L}\right|\right) d_\lambda - R_{\max}\right)^{+}, \end{aligned} \tag{5.10}$$

Table 4: Criterion of risk rating.

Value of risk probability	Risk level
[0.00, 0.38]	Low risk
(0.38, 0.67]	Medium risk
(0.67, 1.00]	High risk

Table 5: The weights of the risk factors.

Risk factor	1	2	3	4	5	6	7	8	9	10
u_β	0.1	0.15	0.10	0.05	0.10	0.10	0.15	0.10	0.05	0.10

where ϕ and η is the punishment coefficient and the risk level of the owner $R_0(X_{i1k}^T)$ is approximated by a convex decreasing function as follows:

$$R_0\left(X_{i1k}^T\right) = \exp\left(-0.001X_{i1k}^T\right). \tag{5.11}$$

Step 6. Compare the evaluated fitness values and select *pbest*, *sbest*, and *cbest* for each upper swarm. Then update the velocity and position of each top-level particle according to (3.1) and (3.2). The top-level computation is repeated until the maximum number of top-level iteration is met.

5.3. An Illustrative Example

In this section, a numerical example of a VE is conducted to validate the capability of VE risk management based on the proposed PS^{2}O. In order to show the superiority of PS^{2}O, the risk management algorithm designed by canonical PSO is also applied to the same case.

In this case, the VE is constructed by one owner and four partners (i.e., $n = 4$) and the total investment is $B_{\max} = 3500$; 10 risk factors are considered for each partner and 4 actions can be selected for each risk factor (i.e., $m = 10$ and $W = 4$); the number of risk ratings $l = 3$ and the value of each rating is $d_1 = 0.165$, $d_2 = 0.335$, and $d_3 = 0.500$, respectively (according to the values of ratings, the criterion of risk rating is shown in Table 4); the maximum risk level $R_{\max} = 0.67$, which means that the risk level of each member must be below the medium level; the weight of risk level of each VE member is $w_0 = w_1 = w_2 = w_3 = w_4$ and the weights u_β of each risk factors for each partner are listed in Table 5; the values of the parameter $\theta_{\beta\lambda}$ and τ_β^α are presented in Tables 6 and 7, respectively; the punishment coefficient ϕ, η, and φ are given as 1.5, 28 and 0.2.

In applying PS^{2}O and PSO to this case, the continuous and binary algorithms are used in top level and base level of the optimization model respectively. For the top-level algorithms, the maximum generation in each execution for each algorithm is 100; the initialized population size of 10 particles is the same for PS^{2}O and PSO, while the whole population is divided into 2 swarms (each possesses 5 individuals) for PS^{2}O in the initialization step; the interaction topology illustrated in Figure 1(a) is used for continuous PS^{2}O; the other parameters of continuous PS^{2}O and PSO were set to the same values as in Section 4. For the base-level algorithms, the maximum generation for each algorithm is 100; the initialized population size of 20 particles is the same for PS^{2}O and PSO, while the whole population is divided into 4 swarms (each possesses 5 individuals) for PS^{2}O in the initialization step;

Table 6: The summary of parameter $\theta_{\beta\lambda}$.

β	λ 1	2	3
1	0.10	0.07	0.13
2	0.23	0.20	0.17
3	0.33	0.27	0.30
4	0.37	0.40	0.43
5	0.50	0.47	0.53
6	0.63	0.57	0.60
7	0.73	0.70	0.67
8	0.83	0.77	0.80
9	0.87	0.90	0.93
10	1.00	0.97	1.03

Table 7: The summary of parameter τ_β^α.

Risk factor	1	2	3	4	5	6	7	8	9	10
τ_β^α	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0

the interaction topology illustrated in Figure 1(b) is used for binary PS^{2}O; the other parameters of binary PS^{2}O and PSO were set to the same values as in Section 4. The experiment runs 30 times, respectively, for each algorithm.

The top-level and base-level search progresses of the averaged best-so-far fitness values over 20 runs are shown in Figures 13 and 14, respectively. It should be noted that the total iteration of base-level searching is 100 (base-level maximum generation) ×10 (top-level population size) ×10 (top-level maximum generation) = 10^4. That is, the base-level algorithms will be restarted after every 100 iterations. From the figures, we can see that PS^{2}O converges with a higher speed compared to PSO and obtains better results in both levels searching progresses.

The average solutions over 30 runs obtained by PS^{2}O and PSO are summarized in Table 8. Before proceeding with the risk management procedure, the risk levels are one for both the VE and the partner, which is a high risk level. Table 8 shows that the resulting risk levels of the VE and the owner are in the low risk level, while all the partners are in the medium risk level. Therefore the budget and the actions selected by the owner and the partner are very effective to reduce the risk of the VE.

To fully demonstrate the risk management performance using the PS^{2}O algorithm, risk investment budget, and risk level controlling processes of each VE member based on PS^{2}O is shown in Figure 15. Generally, an effective actions sequence corresponds to a higher cost and a lower risk level. From the figures, it can be concluded that the additional cost of selecting effect actions can not result in a remarkable decrease in the risk level.

6. Conclusions

In this paper, we develop an optimization model for minimizing the risks of the virtual enterprise based on a novel multi swarm optimizer PS^{2}O. In PS^{2}O, the hierarchical interaction topology consists of two levels (i.e., the individual level and the swarm level), in which

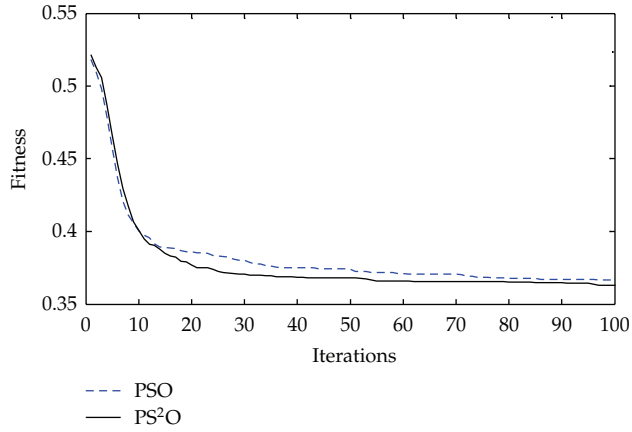

Figure 13: The top-level search process based on PS^{2}O and PSO.

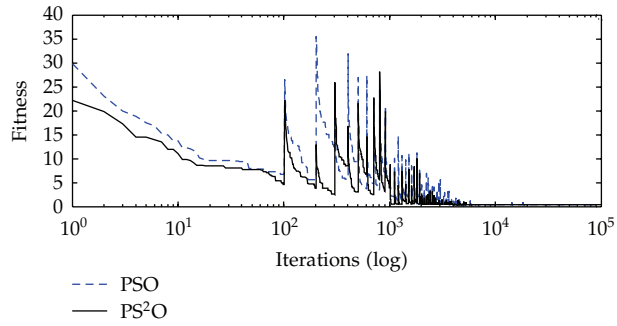

Figure 14: The base-level search process based on PS^{2}O and PSO.

information exchanges take place not only between the particles within each swarm but also between different swarms. The dynamical update equations of our multi-swarm approach are enhanced by a significant ingredient, which takes into account the symbiotic coevolution (or heterogeneous cooperation) between different swarms. Because of this, each individual of the proposed model evolves based on the knowledge integration of itself (associate with individual's own cognition), its swarm members (associate social interaction within each swarm), and its symbiotic partners from other swarm (associate heterogeneous cooperation between different swarms). With five mathematical benchmark functions, PS^{2}O is proved to have significantly better performance than four successful variants of PSO.

In the proposed risk management model of VE, a two-level optimization scheme was introduced to describe the decision processes of the owner and the partners. This DDM model considers the situation that the owner allocates the budget to each member of the VE in order to minimize the risk level of the VE. Accordingly, a transfer optimization model, which can easily use EA and SI algorithms to treat the risk manage problem in VE, is elaborately developed. PS^{2}O is then employed to solve the real-world VE risk management problem. The simulation studies, which compared to Canonical PSO algorithm, show that the PS^{2}O obtains superior risk management solutions than PSO methods in terms of optimization accuracy and convergence speed.

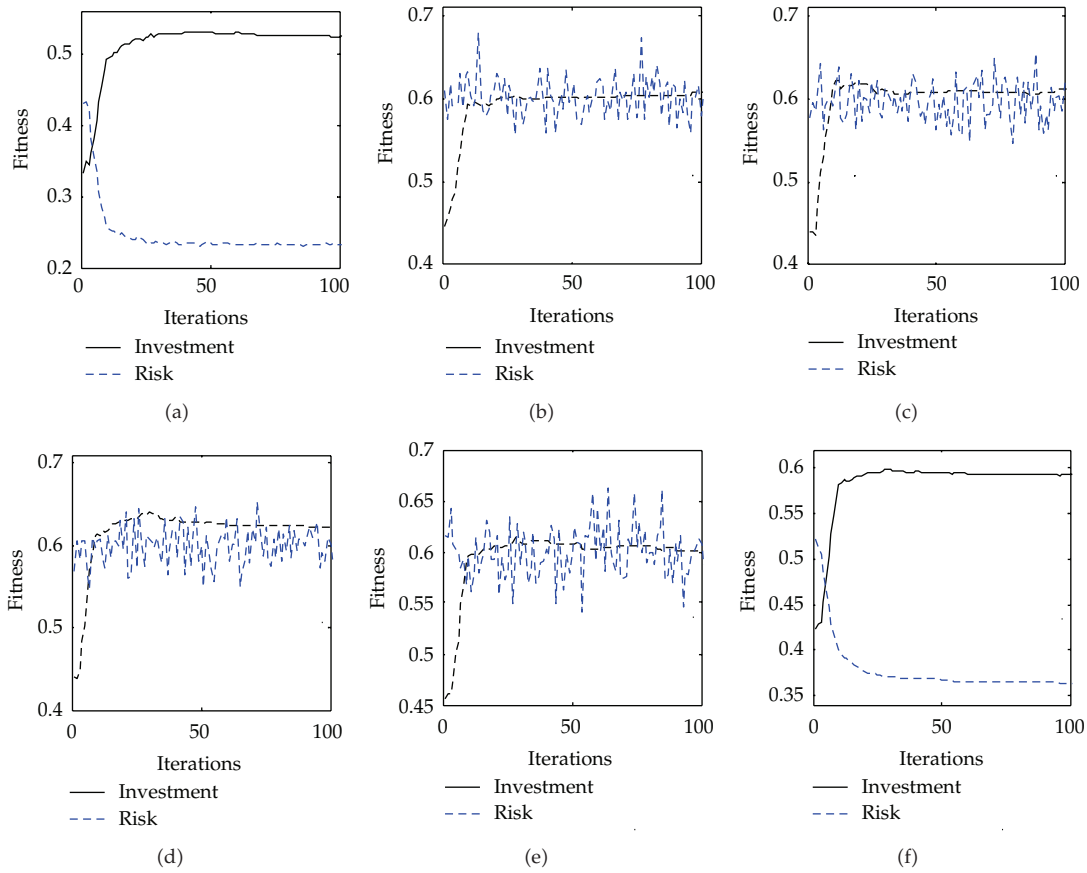

Figure 15: The base-level search process based on PS^{2}O and PSO.

Table 8: Simulation results of both algorithms.

Value of risk probability	PSO	PS^{2}O
Risk level of VE	0.3667	0.3628
Risk level of owner	0.2268	0.2340
Risk level of partner	0.6191, 0.6172, 0.6006, 0.6285	0.6148, 0.6180, 0.5694, 0.5751
Total budget	$3.4455e + 003$	$3.4223e + 003$
Each member's budget	754.12, 670.47, 693.35, 674.78, 652.81	734.17, 668.18, 673.69, 684.84, 661.39

Acknowledgments

This work is supported by the Natural Science Foundation of Liaoning Province of China under Grant 20082006, the Support Program for the Outstanding Technological Person in Liaoning Province of China under Grant lr2011035, and the National Natural Science Foundation of China under Grants 61105067 and 61174164.

References

[1] 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.

[2] D. Karaboga and B. Basturk, "On the performance of artificial bee colony (ABC) algorithm," *Applied Soft Computing Journal*, vol. 8, no. 1, pp. 687–697, 2008.

[3] K. M. Passino, "Biomimicry of bacterial foraging for distributed optimization and control," *IEEE Control Systems Magazine*, vol. 22, no. 3, pp. 52–67, 2002.

[4] H. Chen, Y. Zhu, and K. Hu, "Cooperative bacterial foraging optimization," *Discrete Dynamics in Nature and Society*, vol. 2009, Article ID 815247, 17 pages, 2009.

[5] 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.

[6] R. C. Eberchart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proceedings of the 6th International Symposium on Micromachine and Human Science*, pp. 39–43, Nagoya, Japan, 1995.

[7] J. Kennedy and R. C. Eberhart, *Swarm Intelligence*, Morgan Kaufmann, San Francisco, Calif, USA, 2001.

[8] H. Chen and Y. Zhu, "Optimization based on symbiotic multi-species coevolution," *Applied Mathematics and Computation*, vol. 205, no. 1, pp. 47–60, 2008.

[9] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press, Cambridge, Mass, USA, 1992.

[10] X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 2, pp. 82–102, 1999.

[11] T. Bäck and H.-P. Schwefel, *Evolution Strategies I: Variants and Their Computational Implementation*, Genetic Algorithms in Engineering and Computer Science, Wiley, Chichester, UK, 1995.

[12] J. H. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, Mich, USA, 1975.

[13] P. J. Angeline, "Evolutionary optimization versus particle swarm optimization: philosophy and performance difference," in *Proceedings of the 7th International Conference on Evolutionary Programming*, pp. 601–610, San Diego, Calif, USA, 1998.

[14] Y. Shi and R. C. Eberhart, "Empirical study of particle swarm optimization," in *Proceedings of the 1999 IEEE Congress on Evolutionary Computation*, pp. 1945–1950, Piscataway, NJ, USA, 1999.

[15] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 3, pp. 281–295, 2006.

[16] M. Clerc, Binary Particle Swarm Optimizers: Toolbox, Derivations, and Mathematical Insights, 2005, http://clerc.maurice.free.fr/pso/.

[17] 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.

[18] J. Kennedy and R. C. Eberhart, "A discrete binary version of the particle swarm algorithm," in *Proceedings of the World Multiconference on Systemics, Cybernetics and Informatics*, pp. 4104–4109, Piscataway, NJ, USA, 1997.

[19] M. Tomassini, *Spatially Structured Evolutionary Algorithms: Artificial Evolution in Space and Time*, Springer-Verlag, Berlin, Germany, 2005.

[20] J. J. Liang, P. N. Suganthan, and K. Deb, "Novel composition test functions for numerical global optimization," in *Proceedings of the 2005 Swarm Intelligence Symposium (SIS '05)*, pp. 68–75, June 2005.

[21] J. Kennedy and R. Mendes, "Population structure and particle swarm performance," in *Proceedings of the Congress on Evolutionary Computation*, pp. 1671–1676, Honolulu, Hawaii, USA, 2002.

[22] 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.

[23] K. E. Parsopoulos and M. N. Vrahatis, "UPSO: a unified particle swarm optimization scheme," in *Lecture Notes in Computer Science*, pp. 868–873, 2004.

[24] K. Veeramachaneni, T. Peram, C. Mohan, and L. A. Osadciw, "Optimization using particle swarms with near neighbor interactions," in *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 110–121, Chicago, Ill, USA, 2003.

[25] W. H. Ip, M. Huang, K. L. Yung, and D. Wang, "Genetic algorithm solution for a risk-based partner selection problem in a virtual enterprise," *Computers and Operations Research*, vol. 30, no. 2, pp. 213–231, 2003.

[26] R. L. Kliem and I. S. Ludin, *Reducing Project Risk*, Gower, London, UK, 1997.
[27] M. Huang, F.-Q. Lu, W.-K. Ching, and T. K. Siu, "A distributed decision making model for risk management of virtual enterprise," *Expert Systems with Applications*, vol. 38, no. 10, pp. 13208–13215, 2011.