Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2010, Article ID 572404, 20 pages
doi:10.1155/2010/572404

Research Article

Virtual Enterprise Risk Management Using Artificial Intelligence

Hanning Chen,[1] Yunlong Zhu,[1] Kunyuan Hu,[1] and Xuhui Li[2]

[1] *Key Laboratory of Industrial Informatics, Shenyang Institute of Automation, Chinese Academy of Sciences, Faculty Office III, Nanta Street no. 114, Dongling District, Shenyang 110016, China*
[2] *Jilin Petrochemical Information Network Technology Ltd. Corp., Jilin 132022, China*

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

Received 11 November 2009; Revised 28 February 2010; Accepted 8 March 2010

Academic Editor: Jyh Horng Chou

Virtual enterprise (VE) has to manage its risk effectively in order to guarantee the profit. However, restricting the risk in a VE to the acceptable level is considered difficult due to the agility and diversity of its distributed characteristics. First, in this paper, an optimization model for VE risk management based on distributed decision making model is introduced. This optimization model has two levels, namely, the top model and the base model, which describe the decision processes of the owner and the partners of the VE, respectively. In order to solve the proposed model effectively, this work then applies two powerful artificial intelligence optimization techniques known as evolutionary algorithms (EA) and swarm intelligence (SI). Experiments present comparative studies on the VE risk management problem for one EA and three state-of-the-art SI algorithms. All of the algorithms are evaluated against a test scenario, in which the VE is constructed by one owner and different partners. The simulation results show that the PS^2O algorithm, which is a recently developed SI paradigm simulating symbiotic coevolution behavior in nature, obtains the superior solution for VE risk management problem than the other algorithms in terms of optimization accuracy and computation robustness.

1. Introduction

A virtual enterprise (VE) [1] is a dynamic alliance of autonomous, diverse, and possibly geographically dispersed member companies composed of one owner and several partners that pool their resource 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. With the rapidly increasing competitiveness in global manufacturing area, VE is becoming essential approach to meet the market's requirements for quality, responsiveness, and customer satisfaction. As the VE environment continues to grow in size and complexity, the importance of managing such complexity increases. In the VE environment, there are various sources of risks that may threaten the success of the projects, such as market risk, credit risk, operational risk, and others [2]. Therefore, an effective approach that can actually deal with the risk measurement and management problem is a major concern in VE.

Up to date, risk management of VE has received considerable research attentions. Various models and algorithms are developed to provide a more scientific and effective way for managing the risk of a VE. Ma and Zhang [3] analyzed all kinds of risks during the organization of a VE. They then proposed the defensive measures on the established risk of a VE in order to offer a reference to risk management for the new form of the VE organization. Huang et al. [4] introduced a fuzzy synthetic evaluation model for evaluating the risk of the VE, which focuses on the project mode and the uncertain characteristics of the VE. Ip et al. [1] proposed a risk-based partner selection model, which considers minimizing the risk in selecting partners and ensuring the due date of a project in a VE. By exploring the characteristics of the problem considered and the knowledge of project scheduling, a Rule-based Genetic Algorithm with embedded project scheduling is developed to solve the problem. Sun et al. [5] employed a constructional distributed decision making (DDM) model for risk management of VE that focuses on the situation of team or enforced team relationship between partners. A taboo search algorithm was designed to solve the model. Lu et al. [6] introduced a DDM model for VE risk management that has two levels, namely, the top-model and the base-model, which describe the decision processes of the owner and the partners, respectively. A particle swarm optimization approach was then designed to solve the resulting optimization problem.

Nature serves as a fertile source of concepts, principles, and mechanisms for designing artificial computation systems to tackle complex computational problems. In the past few decades, many nature-inspired computational techniques were designed to deal with practical problems. Among them, the most successful are evolutionary algorithms (EA) and swarm intelligence (SI). Evolutionary algorithms are search methods that take their inspiration from natural selection and survival of the fittest in the biological world. Several different types of EA methods were developed independently. These include genetic programming (GP) [7], evolutionary programming (EP) [8], evolution strategies (ES) [9], and genetic algorithm (GA) [10]. Swarm intelligence (SI), which is inspired by the collective behavior of social systems (such as fish schools, bird flocks, and ant colonies), is an innovative computational way to solve hard optimization problems. Currently, SI includes several different algorithms, namely, ant colony optimization (ACO) [11], particle swarm optimization (PSO) [12, 13], bacterial foraging algorithm (BFA) [14–16], and artificial bee colony algorithm (ABC) [17]. In our previous works [18, 19], we also proposed a novel hierarchical swarm optimization algorithm called PS^2O, which extends the single population PSO to interacting multi-swarm model by constructing hierarchical interaction topologies and enhanced dynamical update equations. By incorporating the new degree of complexity, PS^2O can avoid premature convergence drawback of traditional SI algorithms and accommodate a considerable potential for solving more complex problems.

In this paper, we develop an optimization model for distributed decision making of risk management in VE based on the evolutionary and swarm-based methods. Here the VE risk management problem is described and formulated as a two-level DDM model, which is in order to minimize the aggregate risk level of the VE to a reasonable lower level. Then, in order to solve this complex problem effectively and efficiently, the optimization procedure based on EA and SI systems is developed. Experiments are performed on three VE risk management cases with different scales. In the experiments, a comprehensive comparative study on the performances of four well-known evolutionary and swarm-based algorithms, namely, GA, PSO, ABC, and the recently proposed $\mathrm{PS^2O}$, is presented. Results show that the performance of the $\mathrm{PS^2O}$ is better than or similar to those of other EA and SI algorithms with the advantage of maintaining suitable diversity of the whole population in optimization process.

The paper is organized as follows. Section 2 describes the two-level DDM model of VE risk management. In Section 3, the GA, PSO, ABC, and $\mathrm{PS^2O}$ algorithms are summarized. Section 4 describes a detailed design optimization procedure of risk management in VE by evolutionary and swarm-based algorithms. In Section 5, the simulation results obtained are presented and discussed. Finally, Section 6 outlines the conclusions.

2. Problem Formulation of Risk Management in a VE

In this paper, the two-level risk management model suggested by Lu et al. [6] is employed to evaluate the performance of the proposed methods. This model can be described as a two-level distributed decision making (DDM) system that is depicted in Figure 1.

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 allocate the optimal budget to each member in order to minimize 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:

$$\min_{I} \quad F_T(I) = \sum_{i=0}^{n} w_i R_i(I_i), \tag{2.1}$$

$$\text{s.t.} \quad \sum_{i=0}^{n} I_i \leq I_{\max}, \tag{2.2}$$

$$R_i(I_i) \leq R_{\max}, \tag{2.3}$$

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 according to the top-level's instruction (i.e., the budget to partners). The base-level risk management is that the decision maker selects 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. 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:

$$\begin{aligned} \min_{A} \quad & F_B(A) = \sum_{i=1}^{n} w_i R_i(A_i \mid I_i), \\ \text{s.t.} \quad & \sum_{j=1}^{m} C_j^i\left(a_j^i\right) \leq I_i, \\ & a_j^i \in \{0, 1, 2, \ldots, W\}, \end{aligned} \tag{2.4}$$

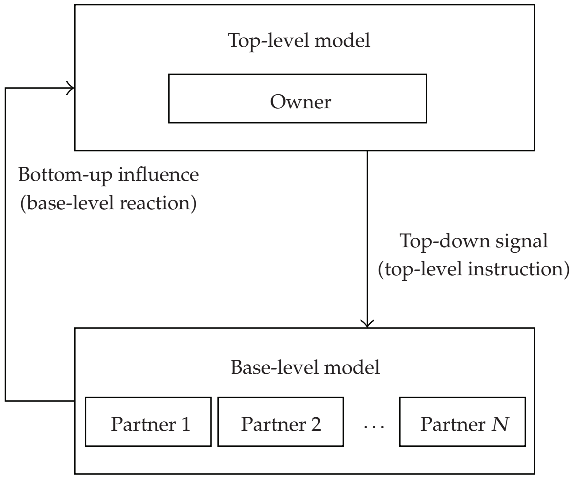


Figure 1: DDM model for risk management in a VE.

(1) **Begin**
(2) Initialize population
(3) **Repeat**
(4) Evaluation
(5) Reproduction
(6) Crossover
(7) Mutation
(8) **Until** requirements are met
(9) **End**

Algorithm 1

where $R_i(A_i \mid I_i)$ 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.

3. Description of the Involved Evolutionary and Swarm Intelligence Algorithms

3.1. The Genetic Algorithm

Genetic algorithm is a particular class of evolutionary algorithms that use techniques inspired by evolutionary biology such as inheritance, mutation, selection, and crossover. A basic GA consists of five components. These are a random number generator, a fitness evaluation unit, genetic operators for reproduction, crossover, and mutation operations. The basic algorithm is summarized in Algorithm 1 .

At the start of the algorithm, the population initialization step randomly generates a set of number strings. Each string is a representation of a solution to the optimization problem

(1) **Begin**
(2) Initialize population
(3) **Repeat**
(4) Place the employed bees on their food sources
(5) Place the onlooker bees on the food sources depending on their nectar amounts
(6) Send the scouts to the search area for discovering new food sources
(7) Memorize the best food source found so far
(8) **Until** requirements are met
(9) **End**

Algorithm 2

being addressed. Continuous and discrete strings are both commonly employed. Associated with each string is a fitness value computed by the evaluation unit. The reproduction operator performs a natural selection function known as seeded selection. Individual strings are copied from one set (representing a generation of solutions) to the next according to their fitness values; the better the fitness value, the greater the probability of a string being selected for the next generation. The crossover operator chooses pairs of strings at random and produces new pairs. The simplest crossover operation is to cut the original parent strings at a randomly selected point and to exchange their tails. The number of crossover operations is governed by a crossover rate. The mutation operator, which is determined by a mutation rate, randomly mutates or reverses the values of bits in a string. A phase of the algorithm consists of applying the evaluation, reproduction, crossover, and mutation operations. A new generation of solutions is produced with each phase of the algorithm [20].

3.2. *The Artificial Bee Colony Algorithm*

Artificial bee colony (ABC) algorithm is one of the most recently introduced SI algorithms. ABC simulates the intelligent foraging behavior of a honeybee swarm. In ABC model, the foraging bees are classified into three categories: employed bees, onlookers, and scout bees. The main steps of the algorithm are as shown in Algorithm 2 .

ABC starts by associating all employed bees with randomly generated food sources (solution). In mathematical terms, S is total number of food sources; the ith food source position can be represented as $X_i = (x_{i1}, x_{i2}, \ldots, x_{iD})$ in the D-dimensional space. $F(X_i)$ refers to the nectar amount of the food source located at X_i. In each iteration t, every employed bee determines a food source in the neighborhood of its current food source and evaluates its nectar amount (fitness). This comparison of two food source position by each employed bee is manipulated according to the following equations:

$$x_{ij}(t) = x_{ij}(t-1) + \operatorname{rand}[-1,1]\left(x_{ij}(t-1) - x_{kj}(t-1)\right), \tag{3.1}$$

where $k \in [1, 2, \ldots, S]$ and $j \in [1, 2, \ldots, D]$ are randomly chosen indexes, and $k \neq j$. Equation (3.1) controls the production of neighbor food sources around X_i. If its new fitness value is better than the best fitness value achieved so far, then the bee moves to this new food source abandoning the old one; otherwise it remains in its old food source. When all employed bees have finished this process, they share the fitness information with the onlookers; each of which selects a food source according to probability p_i defined as

$$p_i = \frac{F(X_i)}{\sum_{n=1}^{S} F(X_n)}. \tag{3.2}$$

For each onlooker that selects the food source X_i, it will find a new neighborhood food source in the vicinity of X_i by using (3.1). Also, the greedy selection mechanism is employed by this onlooker as the selection operation between the old and the new food sources. With this scheme, good food sources will get more onlookers than the bad ones. In ABC, if a food source position cannot be improved further through a predetermined number of cycles, then that food source is assumed to be abandoned and the scout bee will randomly choose a new food source position in the search space.

3.3. The Particle Swarm Optimization

The canonical PSO is a successful SI-based technique. In PSO model, the rules that govern particles' movements are inspired by models of fish schooling and bird flocking [21]. 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 $x_{id} \in [l_d, u_d]$, $d \in [1, D]$, and 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 $v_i = (v_{i1}, v_{i2}, \ldots, v_{iD})$ 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:

$$\begin{gathered} v_{id}(t) = \chi\left(v_{id}(t-1) + R_1 c_1\left(p_{id} - x_{id}(t-1)\right) + R_2 c_2\left(p_{gd} - x_{id}(t-1)\right)\right), \\ x_{id}(t) = x_{id}(t-1) + v_{id}(t), \end{gathered} \tag{3.3}$$

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, given by:

$$\chi = \frac{2}{\left|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}\right|}, \tag{3.4}$$

where $\varphi = c_1 + c_2$, $\varphi > 4$. Main steps of the PSO procedure are as shown in Algorithm 3.

Kennedy and Eberhart [13] 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

(1) **Begin**
(2) Initialize Population
(3) **Repeat**
(4) Calculate fitness values of particles
(5) Modify the best particles in the swarm
(6) Choose the best particle
(7) Calculate the velocities of particles
(8) Update the particle positions
(9) **Until** requirements are met
(10) **End**

Algorithm 3

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:

$$\text{if rand} < S(v_{id}), \ \text{then}\ x_{id} = 1, \ \text{else}\ x_{id} = 0, \tag{3.5}$$

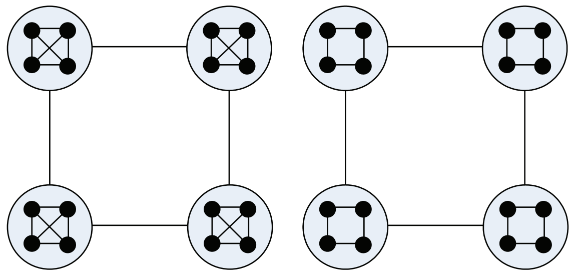
where rand is a random number selected from a uniform distribution in $[0.0, 1.0]$ and the function $S(v)$ is a sigmoid-limiting transformation as follows:

$$S(v) = \frac{1}{1 + e^{-v}}. \tag{3.6}$$

3.4. The Multi-Swarm Optimizer: PS^{2}O

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 [22]. 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, in the previous works [18, 19] we extend the single population PSO to the interacting multi-swarm 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



(a) (b)

Figure 2: Hierarchical topology of the multi-swarm.

(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 2. In Figure 2(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 2(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

$$\begin{aligned} v_{id}^k(t) = \chi\Big(&v_{id}^k(t-1) + R_1 c_1\left(p_{id}^k - x_{id}^k(t-1)\right) + R_2 c_2\left(p_{gd}^k - x_{id}^k(t-1)\right) \\ &+R_3 c_3\left(p_{gd}^\theta - x_{id}^k(t-1)\right)\Big), \end{aligned} \tag{3.7}$$

$$x_{id}^k(t) = x_{id}^k(t-1) + v_{id}^k(t), \tag{3.8}$$

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 is the individual learning rates, c_2 is the social learning rate between particles within each swarm, c_3 is the social learning rate between different swarms, and $R_1, R_2, R_3 \in \mathfrak{R}^d$ are random vectors uniformly distributed in $[0, 1]$. Constriction factor χ is calculated by

$$\chi = \frac{2}{\left|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}\right|}, \tag{3.9}$$

```
Begin
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^θ_gd;
                For (each particle i of swarm k)
                    Find in the particle neighborhood, the point with the best fitness;
                    Set this point as p^k_gd;
                    Update particle velocity using equations (3.7);
                    Update particle position using equations (3.8);
                End For
        End For
    End While
End
```

Algorithm 4

where $\varphi = c_1 + c_2 + c_3$, $\varphi > 4$. Here, the term $R_1c_1(p^k_{id} - x^k_{id})$ is associated with cognition since it takes into account the individual's own experiences; the term $R_2c_2(p^k_{gd} - x^k_{id})$ represents the social interaction within swarm k; the term $R_3c_3(p^{\theta}_{gd} - x^k_{id})$ takes into account the symbiotic coevolution between dissimilar swarms. The pseudocode for the PS^{2}O algorithm is listed in Algorithm 4 .

We should note that, for solving discrete problems, we still use (3.5) and (3.6) to discretize the position vectors in PS^{2}O algorithm.

4. Risk Management in VE Base on Evolutionary and SI Algorithms

The detailed design of Risk management algorithm based on EA and SI algorithms is introduced in this section. Since the risk management model described in Section 2 has a two-level hierarchical structure, the proposed EA and SI-based risk management algorithm is composed of two types of evolving population that search in different levels, respectively, namely, the upper-population and the lower-population. This designed algorithm reflects a two-phase search process, that is, top-level searching phase and base-level searching phase. In the top-level searching phase, the upper-population searches a continuous space for the optimal investment budget allocation by the sponsor for all VE members. While in the base-level searching phase, the lower-population receives information from upper-population and searches the discrete space for a best action combination for risk management of all VE partners.

4.1. Chromosome Representation Scheme and Model Transformation

4.1.1. Definition of Continuous Individual

In upper-population, each individual has a dimension equal to $n + 1$ (i.e., the number of VE members). Each individual is a possible allocation of investment budget for all members that

Partner 1 (a)

Action (γ) \ Risk β	1	2	3	4
1	0	0	1	0
2	1	0	0	0
3	0	1	0	0
4	0	0	0	1

Partner 2 (b)

Action (γ) \ Risk β	1	2	3	4
1	0	0	0	1
2	1	0	0	0
3	0	0	0	0
4	0	1	0	0

Figure 3: Definition of a discrete particle (2314, 2401) for the action combination of two individuals.

have a real number representation. The ith individual of the upper-population T is defined as follows:

$$X_i^T = \left(x_{i1}^T, x_{i2}^T, \ldots, x_{i(n+1)}^T\right), \quad X_i^T \in \Re^{n+1}. \tag{4.1}$$

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

4.1.2. Definition of Discrete Individual

For the lower-population, in order to appropriately represent the action combination by a particle, we design an "action-to-risk-to-partner" representation for the discrete individual. Each discrete individual in each lower-population has a dimension equal to the number of $n \times m \times W$, where 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 individual of the lower-population L is defined as follows:

$$X_i^L = \left(x_{i(111)}^L, x_{i(112)}^L, \ldots, x_{i(n\times m\times W)}^L\right), \quad x_{i(\alpha\beta\gamma)}^L \in \{0,1\}, \tag{4.2}$$

where $x_{i(\alpha\beta\gamma)}^L$ equals 1 if the risk factor β of VE partner α is solved by the γth action and 0 otherwise. That is, each partner can only select one action for each risk factor or do nothing with this factor. For example, set $n = 2$, $m = 4$, and $W = 4$; suppose that the action combination of two partners is (2314, 2401), where 0 stands for no action and is selected for the third risk factor of the second partner in VE. By our definition, we have $x_{i(112)}^L = x_{i(123)}^L = x_{i(131)}^L = x_{i(144)}^L = x_{i(212)}^L = x_{i(224)}^L = x_{i(241)}^L = 1$ and all other $x_{i(\alpha\beta\gamma)}^L = 0$ (see Figure 3).

4.1.3. Model Transformation

Then, the base-level objectives, that is, (2.4), formulated in Section 2 are equivalent to the following optimization problem:

$$\begin{aligned}\min_X F_B\left(X_i^L\right) &= \sum_{\alpha=1}^{n} w_\alpha R_\alpha\left(X_{i\alpha}^L \mid X_{i\alpha}^T\right) = \sum_{\alpha=1}^{n}\sum_{\beta=1}^{m}\sum_{\lambda=1}^{l} w_\alpha u_\beta f_{\beta\lambda}\left(\left|x_{i\alpha\beta}^L\right|\right) d_\lambda \\ &\quad + \varphi \sum_{\alpha=1}^{n}\left(\sum_{\beta=1}^{m} C_\beta^\alpha\left(\left|x_{i\alpha\beta}^L\right|\right) - x_{i\alpha}^T\right)^+,\end{aligned} \tag{4.3}$$

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. $x_{i\alpha\beta}^L = (x_{i\alpha\beta 1}^L, x_{i\alpha\beta 2}^L, \ldots, x_{i\alpha\beta W}^L)$ and $|x_{i\alpha\beta}^L|$ is defined as the position index of 1 in $x_{i\alpha\beta}^L$. For example, if $x_{i\alpha\beta}^L = (0010)$, the value of $|x_{i\alpha\beta k}^L|$ is 3. Here $f_{\beta\lambda}(|x_{i\alpha\beta}^L|)$ is approximated by the convex decreasing function

$$f_{\beta\lambda}\left(\left|x_{i\alpha\beta}^L\right|\right) = \exp\left(-\theta_{\beta\lambda}\left|x_{i\alpha\beta}^L\right|\right) \tag{4.4}$$

to assess the probability of risk occurrence at risk rating λ under action $|x_{i\alpha\beta}^L|$. 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 $C_\beta^\alpha(|x_{i\alpha\beta}^L|)$ is assumed to be a concave increasing function of the corresponding action, which is approximated by

$$C_\beta^\alpha\left(\left|x_{i\alpha\beta}^L\right|\right) = 100\left(1 - \exp\left(-\tau_\beta^\alpha\left|x_{i\alpha\beta}^L\right|\right)\right), \tag{4.5}$$

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

$$(x)^+ = \begin{cases} x & \text{if } x > 0, \\ 0 & \text{else.}\end{cases} \tag{4.6}$$

And the top-level objectives, that is, (2.1)–(2.3), formulated in Section 2 are equivalent to the following optimization problem:

$$\begin{aligned}\min_X F_T\left(X_i^T\right) &= \sum_{\alpha=0}^{n} w_\alpha R_\alpha\left(X_{i(\alpha+1)}^T\right) = w_0 R_0\left(X_{i1}^T\right) + F_B\left(X_i^{L*}\right) \\ &\quad + \phi\left(\sum_{\alpha=1}^{n+1} x_{i\alpha}^T - I_{\max}\right)^+ + \eta \sum_{\alpha=1}^{n}\left(\sum_{\beta=1}^{m}\sum_{\lambda=1}^{l} w_\alpha u_\beta f_{\beta\lambda}\left(\left|x_{i\alpha\beta}^L\right|\right) d_\lambda - R_{\max}\right)^+,\end{aligned} \tag{4.7}$$

where ϕ and η are the punishment coefficients and $X_i^{L^*}$ is the last best base-level decision. Here the risk level of the owner $R_0(X_{i1}^T)$ is approximated by a convex decreasing function as follows:

$$R_0\left(X_{i1}^T\right) = \exp\left(-0.001X_{i1}^T\right). \tag{4.8}$$

In order to easily use EA and SI algorithms to treat the risk management problem in VE, it is clear that the rewritten model, that is, (4.3)–(4.8), is much complex than the original model, that is, (2.1)–(2.4), for there are more variables described in it.

4.2. Risk Management Procedure

The overall risk management process based on EA and SI algorithms can be described as follows.

Step 1. The first step in top-level is to randomly initialize the EA and SI-based upper-population. Each individual X_i^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_i^T, the base-level randomly initializes a corresponding lower-population. At each iteration in base-level, for each particle X_i^L, evaluate its fitness using the base-level optimization function, that is, (4.3).

Step 3. Compare the evaluated fitness values for all individuals in lower-population. Then update each base-level individual by its updating rules according to the selected EA and SI algorithms. For our problem, each partner can only select one action for each risk factor or do nothing with this 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:

$$p_{i(\alpha\beta\gamma)} = \frac{s\left(v_{i(\alpha\beta\gamma)}^L\right)}{\sum_{\gamma=1}^{W} s\left(v_{i(\alpha\beta\gamma)}^L\right)}. \tag{4.9}$$

Then the position of each base-level particle is updated by Algorithm 5 .

Step 4. 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 $X_i^{L^*}$ to the top-level for the fitness computation of the top-level individual X_i^T.

Step 5. With the base-level reaction $X_i^{L^*}$, each top-level individual X_i^T is evaluated by the following top-level fitness function, that is, (4.7).

Step 6. Compare the evaluated fitness values for all individuals in upper-population. Then update each top-level individual by its updating rules according to the selected EA and SI algorithms. The top-level computation is repeated until the maximum number of top-level iteration is met.

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 $\leq p_{i(\alpha\beta\gamma)}$)
        // Action γ is selected for risk β of partner α
      $X^{\text{temp}t}_{\alpha\beta\gamma} = 1$;
      Break;
      End if
    End for
  End for
End for
$X_i^L = X^{\text{temp}}$
END

Algorithm 5

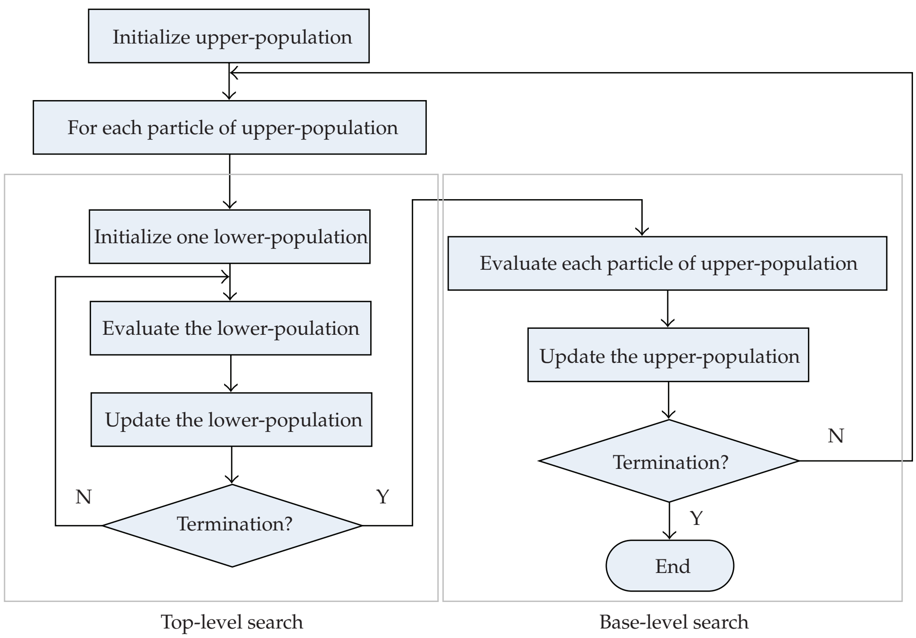


Figure 4: The risk management process based on EA and SI.

The flowchart of this risk management process is illustrated in the diagram given in Figure 4.

5. Experiments Analysis

In this section, a numerical example of a VE is conducted to validate the capability of the proposed VE risk management method. Experiments were conducted with four EA and SI-based algorithms, namely, GA, PSO, ABC, and PS^{2}O, to fully evaluate the performance of the proposed optimization model.

Table 1: 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 2: 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

Table 3: 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

5.1. Illustrative Examples

In this section, 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 is $l = 3$ 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 1); 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 factor for each partner are listed in Table 2; the values of the parameters $\theta_{\beta\lambda}$ and τ_β^α are presented in Tables 3 and 4, respectively; the punishment coefficients ϕ, η, and φ are given as 1.5, 28, and 0.2.

This simulated VE environment can be constructed by one owner and different number of partners. For scalability study purpose, all involved algorithms are tested on three illustrative VE examples with 2, 4, and 9 partners (i.e., $n = 2, 4, 9$), respectively.

5.2. Settings for Involved Algorithms

In applying EA and SI algorithms to this case, the continuous and binary versions of these algorithms are used in top-level and base-level of the DDM optimization model, respectively. For the top-level algorithms, the maximum generation in each execution for each algorithm is 50; the initialized population size of 10 individuals is the same for all involved algorithms,

Table 4: 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

Table 5: Results of all algorithms. In bold are the best.

Scale of VE		PSO	PS^{2}O	GA	ABC
3 members	Best	0.2167	0.2065	**0.1901**	0.2240
	Worst	0.3034	**0.2614**	0.5420	0.2650
	Mean	0.2514	**0.2354**	0.2356	0.2436
	Std	0.0202	**0.0109**	0.0626	0.0112
5 members	Best	0.3396	0.3218	**0.2727**	0.3166
	Worst	0.4804	**0.3566**	3.2243	0.3891
	Mean	0.3739	**0.3363**	0.5139	0.3593
	Std	0.0331	**0.0091**	0.6110	0.0148
10 members	Best	0.3320	0.2641	**0.1970**	0.3940
	Worst	4.7246	**1.9785**	2.7449	4.2746
	Mean	0.7786	**0.6045**	0.8987	0.7578
	Std	0.9049	0.6972	**0.3886**	0.9817

while the whole population is divided into 2 swarms (each possesses 5 individuals) for PS^{2}O in the initialization step. For the base-level algorithms, the maximum generation for each algorithm is 100; the initialized population size of 20 particles is the same for all involved algorithms, while the whole population is divided into 4 swarms (each possesses 5 individuals) for PS^{2}O in the initialization step. The experiment runs 30 times, respectively, for each algorithm. The other specific parameters of algorithms are given below.

GA Settings

The experiment employed a binary coded standard GA having random selection, crossover, mutation, and elite units. Stochastic uniform sampling technique was the chosen selection method. Single-point crossover operation with the rate of 0.8 was employed. Mutation operation restores genetic diversity lost during the application of reproduction and crossover. Mutation rate in the experiment was set to be 0.01.

PSO Settings

For continuous PSO, the learning rates c_1 and c_2 were both 2.05 and the constriction factor $\chi = 0.729$; for binary PSO, the parameters were set to the values $c_1 = c_2 = 2$ and $\chi = 1$. The ring topology was used for both versions of PSO.

ABC Settings

The basic ABC is used in the study. Since there are no literatures using ABC for discrete optimization so far, this experiment just used crossover operation to update individuals in

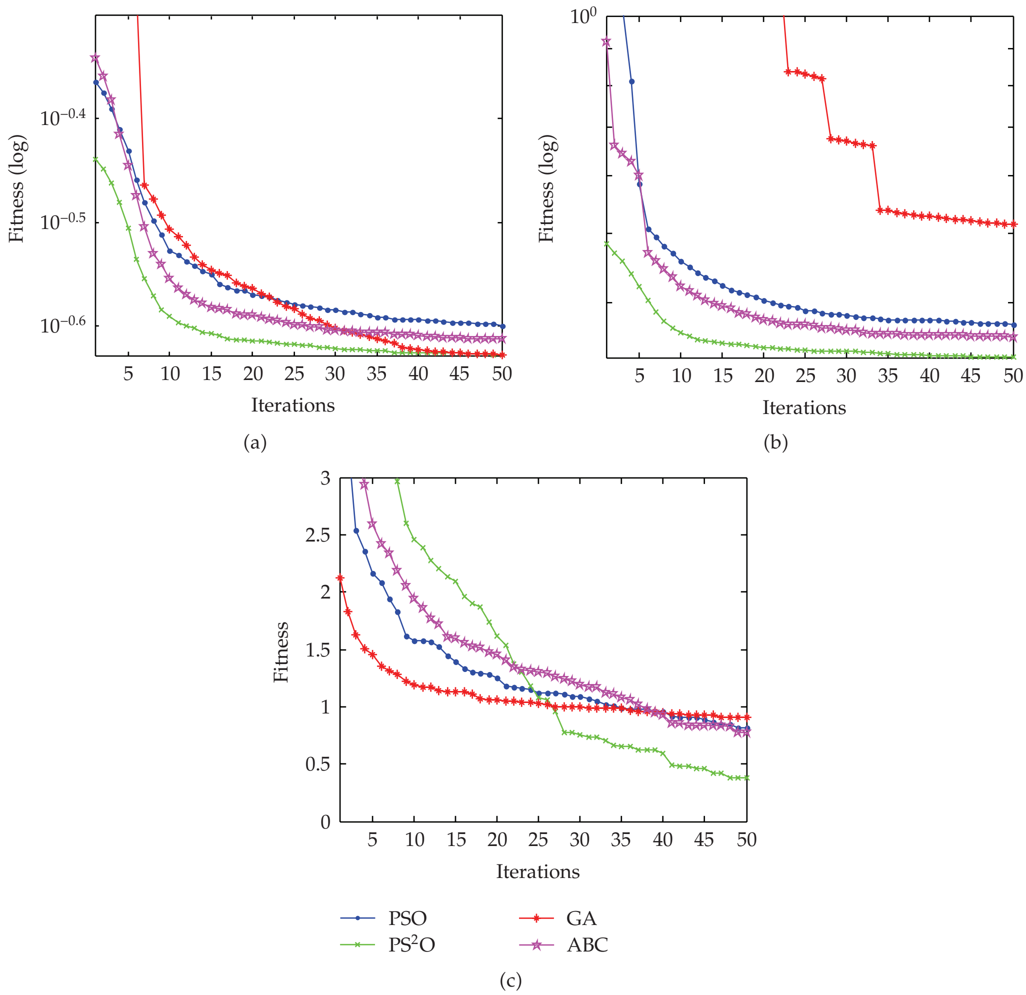


Figure 5: The iteration courses of all algorithms on different VE scales. (a) 3 members. (b) 4 members. (c) 10 members.

ABC population. That is, the ABC position update (3.1) can be changed to the following equation (5.1) for discrete problems:

$$x_{ij}(t) = x_{kj}(t-1). \tag{5.1}$$

Then the limit parameter is set to be $SN \times D$ for ABC in both continuous and discrete search, where D is the dimension of the problem and SN is the number of employed bees.

PS^{2}O Settings

For continuous PS^{2}O, the parameters were set to the values $c_1 = c_2 = c_3 = 1.3667$ (i.e., $\phi = c_1 + c_2 + c3 \approx 4.1 > 4$) and then $\chi = 0.729$, which is calculated by (3.9); the interaction topology illustrated in Figure 2(a) is used. For discrete PS^{2}O, the parameters were set to the values $c_1 = c_2 = c_3 = 2$ and $\chi = 1$; the interaction topology illustrated in Figure 2(b) is used.

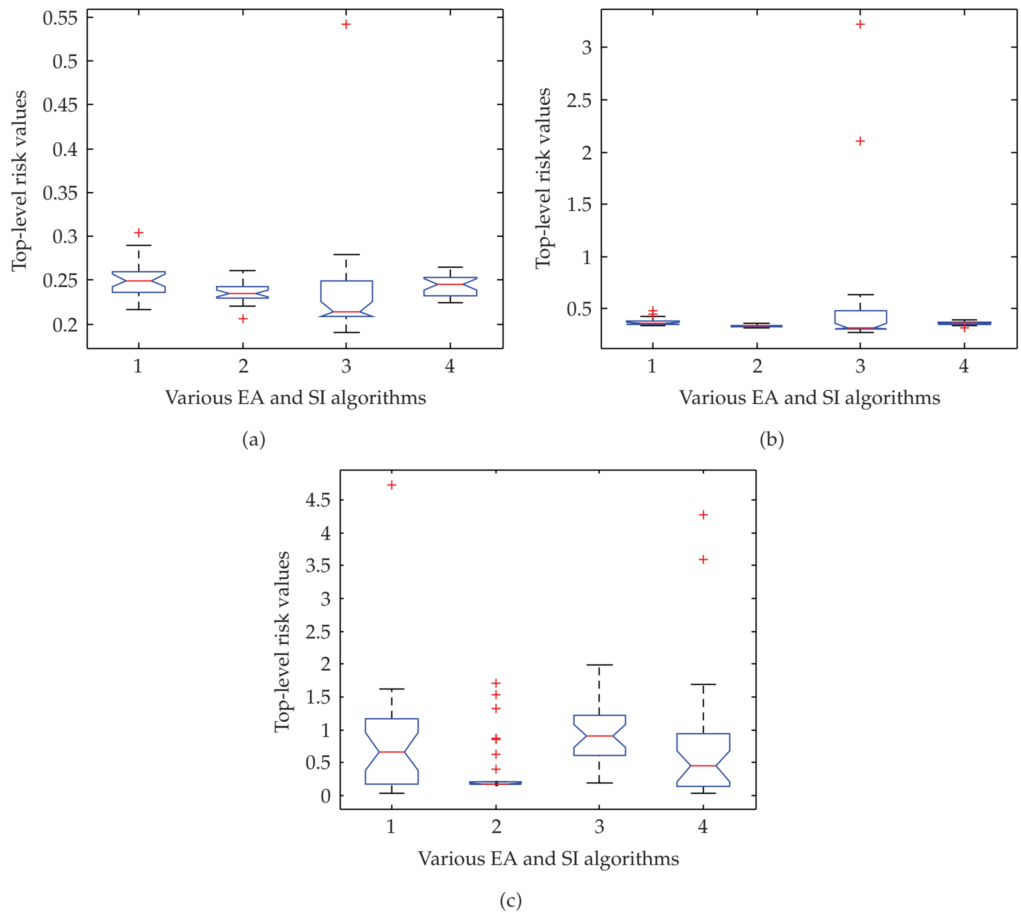


Figure 6: ANOVA test for all candidate algorithms on different VE scales of (a) 3 members, (b) 4 members, and (c) 10 members. (Here 1, 2, 3, and 4 are the algorithm index of PSO, PS^{2}O, GA, ABC, resp.).

All algorithms are tested on the risk management problems with 3, 5, and 10 VE members. The representative results obtained are presented in Table 5, including the best, worst, mean, and standard deviation of the risk values of VE found in 30 runs. Figures 5(a)–5(c) present the evolution process of all algorithms for the minimization of the VE risks in 3 different scales. For each trial, we can see before proceeding with the EA and SI-based risk management procedure, the risk levels are very high for the VE. Table 5 shows that the resulting risk levels of the VE are in the lower risk level. Therefore the budget and the actions selected by the four EA and SI algorithms for the owner and the partners are very effective to reduce the risks of the VE. From the results, it is clear that the PS^{2}O algorithm can consistently converge to better results than the other three algorithms for all test cases. Also, PS^{2}O is the most fast one for finding good results within relatively few generations.

In this experiment, the analysis of variance (ANOVA) test was also carried out to validate the efficacy of four tested EA and SI methods. The graphical statistics analyses are done through box plot. A box plot is a graphical tool, which provides an excellent visual summary of many important aspects of a distribution. The box stretches from the lower hinge (defined as the 25th percentile) to the upper hinge (the 75th percentile) and therefore contains

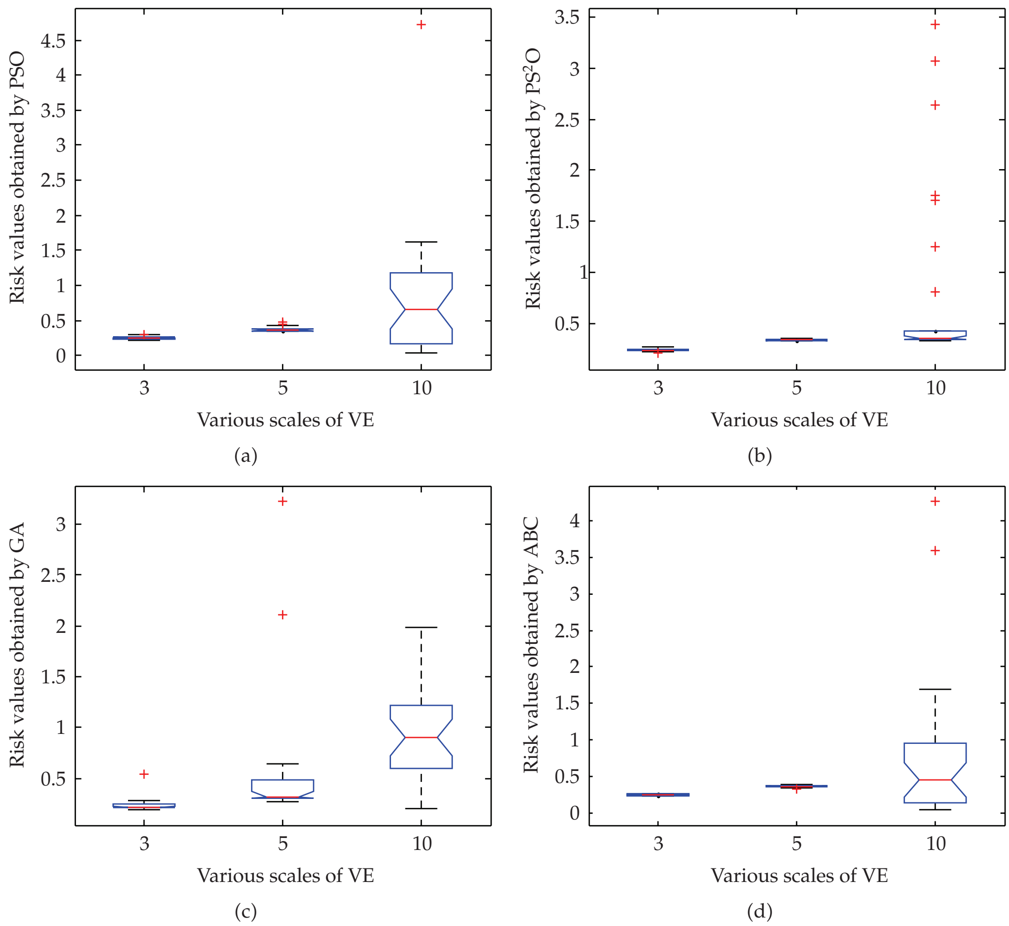


Figure 7: ANOVA test for the risk management results with different VE scales optimized by (a) PSO, (b) PS^{2}O, (c) GA, and (d) ABC.

the middle half of the scores in the distribution. The median is shown as a line across the box. Therefore, one-fourth of the distribution is between this line and the top of the box and one-fourth of the distribution is between this line and the bottom of the box.

First, the box plots for the results presented in Table 5 are shown in Figures 6(a)–6(c). Figure 6 implies the graphical performance representation of all algorithms in 30 runs. From this box plot representation, it is clearly visible and proved that the PS^{2}O provides better results for all the test cases than those of the other three algorithms.

Second, to compare the robustness of the involved algorithms on the risk manage problem, the experiment can be statistically considered as one-factor experiment, in which the optimization result was the response variable and the scales of the VE were the factor, which had 3 levels: 3, 5, and 10. The results of the ANOVA for the VE risk management in different scales using these algorithms were presented in Figure 7 by the box plot. From Figure 7, we can observe that the main effects of the problem scale are not significant for the optimization results obtained by the PS^{2}O algorithm. Therefore, against the scales variation of the testing VE cases, the robustness of the PS^{2}O is much better than those of the PSO, GA, and ABC.

6. Conclusions

In this paper, we develop an optimization model for minimizing the risks of the virtual enterprise based on evolutionary and swarm intelligence methods. First, a two-level risk management model 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 management problem in VE, is elaborately developed. We should note that the proposed optimization model is genetic and extendible: the model does not depend on the optimization algorithm used and other evolutionary and swarm intelligence techniques could be equally well adopted, which enable a comparison of various algorithms for the same application scenario.

Experiments show comparative studies on the VE risk management problem for the GA, PSO, ABC, and PS^2O. The simulation results show that the PS^2O algorithm obtains superior solutions on three testing cases than the other algorithms in terms of optimization accuracy and computation robustness. That is, in PS^2O, 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.

Acknowledgments

This work is supported by the National 863 plans projects of China under Grants nos. 2006AA04A117 and 08H2010201. The first author would like to thank Dr. Fuqing Lu for helpful discussions and constructive comments.

References

[1] 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 & Operations Research*, vol. 30, no. 2, pp. 213–231, 2003.

[2] R. L. Kliem and I. S. Ludin, *Reducing Project Risk*, Gower, Hampshire, UK, 1997.

[3] J. Ma and Q. Zhang, "The search on the established risk of enterprise dynamic alliance," in *Proceedings of International Conference on Management Science and Engineering*, pp. 727–731, 2002.

[4] M. Huang, H.-M. Yang, and X.-W. Wang, "Genetic algorithm and fuzzy synthetic evaluation based risk programming for virtual enterprise," *Acta Automatica Sinica*, vol. 30, no. 3, pp. 449–454, 2004.

[5] X. Sun, M. Huang, and X. Wang, "Tabu search based distributed risk management for virtual enterprise," in *Proceedings of the 2nd IEEE Conference on Industrial Electronics and Applications (ICIEA '07)*, pp. 2366–2370, Harbin, China, May 2007.

[6] F.-Q. Lu, M. Huang, W.-K. Ching, X.-W. Wang, and X.-L. Sun, "Multi-swarm particle swarm optimization based risk management model for virtual enterprise," in *Proceedings of the 1st ACM/SIGEVO Summit on Genetic and Evolutionary Computation (GEC '09)*, pp. 387–392, Shanghai, China, June 2009.

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

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

[9] T. Bäck and H. P. Schwefel, "Evolution strategies I: variants and their computational implementation," in *Genetic Algorithms in Engineering and Computer Science*, pp. 111–126, Wiley, Chichester, UK, 1995.

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

[11] 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.

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

[13] J. Kennedy and R. C. Eberhart, "A discrete binary version of the particle swarm algorithm," in *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, vol. 5, pp. 4104–4108, Orlando, Fla, USA, October 1997.

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

[15] 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.

[16] 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.

[17] D. Karaboga, "An idea based on honeybee swarm for numerical optimization," Tech. Rep. TR06, Computer Engineering Department, Engineering Faculty, Erciyes University, 2005.

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

[19] H. Chen, Y. Zhu, K. Hu, and X. He, "Hierarchical swarm model: a new approach to optimization," *Discrete Dynamic in Nature and Society*. In press.

[20] D. T. Pham and D. Karaboga, "Optimum design of fuzzy logic controllers using genetic algorithms," *Journal of Systems Engineering*, pp. 114–118, 1991.

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

[22] M. Tomassini, *Spatially Structured Evolutionary Algorithms: Artificial Evolution in Space and Time*, Natural Computing Series, Springer, Berlin, Germany, 2005.