Application of Improved Discrete Particle Swarm Algorithm in Partner Selection of Virtual Enterprise

Fang Gao[1], Gang Cui[1], Qiang Zhao[2], and Hongwei Liu[1]

[1]School of Computer Science and Technology, Harbin Institute of Technology, 150001 Harbin, China

[2]School of Traffic Transportation Engineering, Northeast Forestry University,150006 Harbin,China

Summary. Partner selection is a critical issue in the research of virtual enterprise. It is usually a combinatorial optimization problem. In this paper, an improved discrete particle swarm algorithm is proposed for the partner selection optimization. At first, an optimization model is presented, which includes the main crucial factors for partner selection, such as running cost, reaction time and failure risk. To decrease the probability of the particle swarm algorithm to trap in a local optimum, a new particle position updating formula is modified as follows. Particle velocity is divided into three regions and according the region a particle fall into, its position is correspondingly determined to keep unchanged or take value 0 and 1. As the iteration continues, these three regions can adaptively change their range to increase the global convergence. Numerical results have demonstrated the improved particle swarm algorithm show better performance in partner selection.An algorithm based on discrete binary particle swarm optimization (PSO) is then presented for the solution. A typical example is used to illustrate the effectiveness of the algorithm. The proposed optimization model and scheme provides a reference to partner selection in the actual VE operation.

Key words: Virtual Enterprise, Partner Selection, Discrete Particle Swarm Algorithm,

1 Introduction

Virtual enterprise is a temporary association of corporations linked by supply chain to respond to business opportunities and achieve maximum profit. With the rapid development of information technologies and continuous exacerbation of global competition, the virtual enterprise has been considered as the most advanced and efficient form of modern product chain. Partner selection is one of the crucial problems in the formation and operation of a virtual enterprise. It is actually a combinatorial optimization problem. In order to implement a project, a virtual enterprise needs to be formed with several business process types, and each process type may have several, even tens of candidates. Meanwhile one or more candidates can be selected to join the virtual enterprise. Thus the total number of combinations being considered may be too large to use simple enumeration solution.

However, a lot of research results have been reported for this problem. Zhan [1] discussed the partner relationship management principle and gave some qualitative analysis. Talluri and Baker [2] proposed a two-phase mathematical programming approach for partner selection, with phase 1 identifying efficient candidates by data envelopment analysis, and phase 2 using an integer (0-1) goal programming model to select an effective combination of efficient partners. Chu [3] proposed to employ analytic hierarchy process (AHP) method to select the best partners from the potential ones. It is suitable for the case that the total number of candidate is not too large. Feng et. al. [4] presented an optimization model for partner selection and the solution based on genetic algorithm. In the objective function, the factor of link time (such as transit time) between partners was not included. Qu and Sun discussed the resource optimization configuration in a real networked manufacturing system and employed genetic algorithm to select partners for five process types in a product chain [5]. However, the factor of running risk was not included in their objective function.

Improving the objective function for partner selection and finding an effective intelligent optimization approach become more desirable in the research of virtual enterprises. The Particle Swarm Optimization (PSO), proposed by Kennedy and Eberhart [6], has proved to be a very effective approach in the global optimization for complicated problems [7,8]. Although the continuous version of PSO algorithm has been successfully used to solve many real problems, research work remain relative fewer on the discrete version of PSO algorithm. Yin used a discrete PSO algorithm to solve the polygonal approximation of a digital curve. In his paper, a hybrid strategy embedding a local optimizer within the discrete PSO algorithm has been shown to outperform the original version [9]. In this paper, we will present the objective function for partner selection and use a modified discrete binary swarm optimization algorithm to search for the optimal solution.

2 Improved Discrete PSO Algorithm for Partner Selection

In this section, we first describe the partner selection problem and then use an improved discrete binary PSO algorithm to solve the presented problem.

2.1 Problem Definition

Suppose that a virtual enterprise consists of n business processes (core competencies): E_i , $i = 1,2,...,I$. For the i th business process, there are a_i potential candidates for the partners. Let m_j^i , $j = 1,2,...,a_i$, be the j th potential candidate for the i th business process. The partner selection problem can be described as follows: take one candidate from m_j^i for each business process E_i to form a virtual enterprise, and the resultant combination of selected partners must satisfy an optimization objective.

Since the running cost, reaction time, and running risk are the key points in the operation of virtual enterprise, we take them as the optimization objectives for partner selection:

1) Running cost: namely C , consists of the internal cost of each candidate and the link cost between any two candidates:

$$\min C = \min\left\{\sum_{i=1}^{I}\sum_{j=1}^{a_i}(\beta_j^i C_j^i) + \sum_{i=1}^{I}\sum_{j=1}^{a_i}\sum_{i'=1,i'\neq i}^{I}\sum_{j'=1}^{a_i}(\beta_j^i\beta_{j'}^{i'}C_{j,j'}^{i,i'})\right\} \quad (1)$$

where $$\beta_j^i = \begin{cases} 1 & M_j^i \text{ is selected} \\ 0 & M_j^i \text{ is not selected} \end{cases}$$

C_j^i is the internal cost of the candidate m_j^i . $C_{i,i'}^{j,j'}$ is the link cost between two candidates of m_j^i and $m_{j'}^{i'}$.

2) Reaction time: namely T , consists of the internal reaction time of each candidate and the link time between any two candidates:

$$\min T = \min\left\{\sum_{i=1}^{I}\sum_{j=1}^{a_i}(\beta_j^i T_j^i) + \sum_{i=1}^{I}\sum_{j=1}^{a_i}\sum_{i'=1,i'\neq i}^{I}\sum_{j'=1}^{a_i}(\beta_j^i\beta_{j'}^{i'}T_{j,j'}^{i,i'})\right\} \quad (2)$$

where T_j^i is the internal reaction time of the candidate m_j^i . $T_{i,i'}^{j,j'}$ is the link time between two candidates of m_j^i and $m_{j'}^{i'}$.

3) Running Risk: namely R , is defines as follows:

$$\min R = \min\left\{\sum_{i=1}^{I}\max_j(\beta_j^i R_j^i)\right\} \quad (3)$$

where R_j^i is the running risk generated by selecting m_j^i .

Here we take the weighted sum of Eq. (1), (2) , and (3) as the objective function:

$$F(X) = \frac{\omega_c}{E_c}\left(\sum_{i=1}^{I}\sum_{j=1}^{a_i}(\beta_j^i C_j^i) + \sum_{i=1}^{I}\sum_{j=1}^{a_i}\sum_{i'=1,i'\neq i}^{I}\sum_{j'=1}^{a_i}(\beta_j^i\beta_{j'}^{i'}C_{j,j'}^{i,i'})\right)$$

$$+\frac{\omega_t}{E_t}\left(\sum_{i=1}^{I}\sum_{j=1}^{a_i}(\beta_j^i T_j^i) + \sum_{i=1}^{I}\sum_{j=1}^{a_i}\sum_{i'=1,i'\neq i}^{I}\sum_{j'=1}^{a_i}(\beta_j^i\beta_{j'}^{i'}T_{j,j'}^{i,i'})\right)$$

$$+\frac{\omega_r}{E_r}\left(\sum_{i=1}^{I}\max_j(\beta_j^i R_j^i)\right) \quad (4)$$

where E_c , E_t , and E_r are the least desired values of C , T and R respectively, and ω_c , ω_t , and ω_r are the weight coefficients of C , T , and R respectively.

2.2 Overview of PSO Algorithm

Like the genetic algorithms, the PSO algorithm first randomly initializes a swarm of particles. Each particle is represented as $X_i = (x_{i1}, x_{i2},..., x_{in})$, $i = 1,2,..., N$, where N is the swarm size. Thus, each particle is randomly placed in the n -dimensional space as a candidate solution. Each particle adjusts its trajectory toward its own previous best position and the previous best position is attained by any particles of the swarm, namely $pbest_i$ and $gbest$. In each iteration of the PSO algorithm, the swarm is updated by the following equations:

$$v_{ij}^{k+1} = v_{ij}^k + c_1 r_1(pbest_{ij}^k - x_{ij}^k) + c_2 r_2(gbest_j^k - x_{ij}^k) \quad (5)$$

$$x_{i,j}^{k+1} = x_{i,j}^k + v_{i,j}^k \quad (6)$$

where k is the current iteration number， v_{ij} is the updated velocity on the j th dimension of the i th particle, c_1 and c_2 are acceleration constants, r_1 and r_2 are the real numbers drawn from two uniform random sequences of $U(0, 1)$.

The above continuous particle swarm algorithm has been used to solve optimization problems. In the discrete binary version, a particle moves in a state space restricted to zero and one on each dimension, where each v_{ij} represents the probability of bit x_{ij} taking the value 1. Thus, the step for updating v_{ij} remains unchanged as shown in Eq. (5), except that $pbest_{ij}$ and $gbest_j$ are integers in {0, 1} in binary case. The resulted changes in position are defined as follows:

$$s(v_{i,j}^{k+1}) = 1/(1+\exp(-v_{i,j}^{k+1})) \quad (7)$$

$$\text{if } (r < s(v_{i,j}^{k+1}) \text{ then } x_{i,j}^{k+1} = 1 \text{ else } x_{i,j}^{k+1} = 0 \quad (8)$$

where r is random number drawn from uniform sequence of $U(0,1)$.

2.3 Improved Discrete PSO

To enhance the search ability and circumvent avoid the particle to converge to local optima, an improved PSO algorithm for partner selection can be described as follows. The velocity updating formula remains unchanged as shown in Eq. (5). The position updating of each particle is defined as follows:

$$\text{if } 0.5-\delta < s(v_{i,j}^{k+1}) < 0.5+\delta \text{ then } x_{i,j}^{k+1} = x_{i,j}^{k} \tag{9}$$

$$\text{if } s(v_{i,j}^{k+1}) < 0.5-\delta \text{ then } x_{i,j}^{k+1} = 0 \tag{10}$$

$$\text{if } s(v_{i,j}^{k+1}) > 0.5+\delta \text{ then } x_{i,j}^{k+1} = 1 \tag{11}$$

where the initial value of δ is 0.25, and as the iteration continues, its value decreased gradually. Eq. (9) makes each particle to keep its own inertia and to avoid that all the particles tend to move towards the same position to trap in a local optima. Therefore δ should be hold a big value in the initial phase, and then gradually decreased to enhance the local fine-tuning search ability. A linear variation rule of δ can be defined as follows:

$$\delta_k = \delta_{initial} - \frac{\delta_{initial} - \delta_{final}}{k_{\max}} k \tag{12}$$

where $\delta_{initial}$ is the initial value of δ. k is the current iteration number and $k_{\max}$ is the maximum number of iterations. δ_{final} is the final value of δ.

2.4 Particle Representation and Fitness Function Selection

Suppose n is the total number of all the business process types. Since each particle of the swarm is one of the candidate solutions for the problem, we represent each particle by a binary vector as follows:

$$X = \{b^i\} = \{b^1, b^2, ..., b^n\}, \text{ subject to } \sum_{j=1}^{a_i} b_j^i \geq 1 \tag{13}$$

where $b^i = \{b_1^i, b_2^i, ..., b_{a_i}^i\}$, as a bit in X, b_j^i represents the j th candidate for the i th process, which is restricted to take integers in {0, 1} according to

$$b_j^i = \begin{cases} 1 & m_j^i \text{ is selected} \\ 0 & m_j^i \text{ is not selected} \end{cases} \tag{14}$$

Based on the optimization objective function in Eq. (4), the fitness function $f(x)$ of each particle is set as $f(x) = 1/F(x)$.

2.5 Optimization Procedure

The procedure of the proposed binary PSO algorithm for solving the optimization problem of partner selection is described as follows:

Step 1. *Initialize N particles with random positions $x_1, x_2, ..., x_N$ according to Eq. (13). Generate velocities v_{ij}, $i = 1,2,...,N$ and $j = 1,2,...,n$, where $v_{ij} \sim U(0,1)$.*

Step 2. *Evaluate each particle according to $f(x) = 1/F(x)$ based on Eq. (4).*

Step 3. *Update individual and global best positions: If $f(pbest_i) < f(x_i)$, then $pbest_i = x_i$, and search for the maximum value $f_{\max}$ among $f(pbest_i)$, If $f(gbest) < f_{\max}$, $gbest = x_{\max}$, $x_{\max}$ is the particle associated with $f_{\max}$.*

Step 4. *Update velocity: update the i th particle velocity using the Eq. (5) restricted by maximum and minimum threshold $v_{\max}$ and $v_{\min}$.*

Step 5. *Update Position: update the i th particle position using Eq. (9), (10) and (11).*

Step 6. *Update δ according to Eq.(12).*

Step 7. *Repeat step 2 to 6 until a given maximum number of iterations is achieved.*

3 Numerical Simulations

In this section we use an example to illustrate the effectiveness of the algorithm for partner selection using the improved PSO algorithm. A supply chain for some product consists of 4 business processes: design (D), purchase (P), manufacture (M), and sale (S). Candidates for each business process are presented in Table 1. Our goal is to select one candidate from each business process to minimize the objective functions in Eq. (4). The internal running cost, reaction time, and running risk of each candidate are presented in Table 2. The link time and cost between candidates are as shown respectively in Table 3. In this table, elements on the left lower region of the diagonal line represent the link time, and elements on the right upper region represent the link cost between two candidates.

Table 1. business processes E_i and their candidate enterprises m_j^i

Business Processes	Design		Purchase					Manufacture				sales		
Candidates	D_1	D_2	P_1	P_2	P_3	P_4	P_5	M_1	M_2	M_3	M_4	S_1	S_2	S_3

Table 2. internal cost C_j^i, reaction time T_j^i, and running risk R_j^i of each candidate enterprise

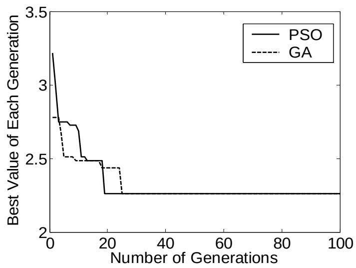
	D_1	D_2	P_1	P_2	P_3	P_4	P_5	M_1	M_2	M_3	M_4	S_1	S_2	S_3
$T_j^{(i)}$ /month	8.8	5.3	7.6	4.5	8.9	10.5	11.8	4.4	9.7	2.4	7.8	9.1	8.3	5.9
$C_j^{(i)}$ /10^4 RMB	85.3	74.6	55.5	82.6	91.2	63.2	51.6	99.4	96.5	86.4	92.9	89.7	90.2	84.0
R_j^i	0.3	0.5	0.5	0.2	0.2	0.4	0.4	0.3	0.4	0.2	0.5	0.3	0.3	0.2

Table 3. Link cost $C_{i,i'}^{j,j'}$ (10^4 RMB) and link time $T_{i,i'}^{j,j'}$ (month) between any two candidates

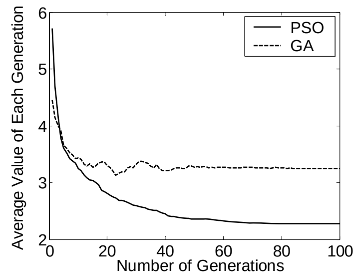
	D_1	D_2	P_1	P_2	P_3	P_4	P_5	M_1	M_2	M_3	M_4	S_1	S_2	S_3
D_1	0	5.3	1.7	2.5	5.8	5.0	5.7	3.2	0.7	3.9	2.3	3.2	3.8	3.7
D_2	5.9	0	2.1	1.6	5.9	4.2	4.1	2.3	4.8	3.6	5.9	2.1	1.2	4.1
P_1	1.9	5.1	0	4.1	2.3	1.4	4.1	5.0	1.6	5.9	3.5	5.9	3.0	3.3
P_2	5.7	1.6	3.7	0	1.1	0.6	1.3	2.0	1.0	4.7	1.1	3.7	1.6	5.5
P_3	5.7	3.3	3.1	2.6	0	3.6	0.8	0.9	4.5	4.3	1.4	4.8	0.6	2.1
P_4	4.1	3.6	0.0	3.8	3.5	0	0.9	5.5	4.5	0.7	4.9	5.1	5.7	4.5
P_5	3.0	2.1	2.4	2.0	4.4	4.9	0	2.6	5.0	1.5	1.1	3.2	3.6	3.5
M_1	0.4	5.2	4.8	5.8	5.1	2.6	4.7	0	5.5	4.5	3.1	0.6	4.8	5.8
M_2	5.9	1.4	1.6	1.5	0.2	0.2	5.9	0.5	0	3.2	3.3	5.7	0.9	0.7
M_3	2.0	1.6	5.2	0.5	1.0	1.7	4.7	3.2	0.8	0	0.9	5.2	0.6	3.5
M_4	3.8	3.2	5.0	4.1	0.0	4.6	2.2	0.8	4.2	2.2	0	1.3	5.6	4.3
S_1	5.6	3.0	2.2	2.8	2.6	1.6	5.3	2.6	4.2	3.9	3.8	0	4.9	5.0
S_2	4.3	5.6	5.8	2.1	2.6	4.8	5.5	1.8	4.2	6.0	3.5	1.8	0	3.6
S_3	3.1	0.5	5.7	5.0	3.3	2.0	0.1	4.7	3.3	3.3	1.5	0.7	5.7	0

Parameters of the problem are set as: $E_c = 150$, $E_t = 60$, $E_r = 0.2$, $\omega_c = 0.25$, $\omega_c = 0.45$, and $\omega_c = 0.3$. Parameters of the algorithm are set as: $N = 20$, $c_1 = 2$, $c_2 = 2$, $v_{\max} = 4$, and $v_{\min} = -4$. Fig. 1-(a) shows a typical running of the PSO algorithm and it shows that the objective function $F(x)$ quickly converges to the optimal solution when the iteration generation increases. The optimum represented by *gbest* equals to {10010000010001} with the objective functions $F(gbest) = 2.2625$, which means that the best partner combination is {D_1, P_2, M_3, S_3}. Fig. 1-(b) shows that the average value of the whole swarm descends and approaches quickly to the best value in Fig. 1-(a).

Simulation with standard genetic algorithm (GA) is also performed as a comparison with our improved PSO algorithm. Parameters of GA are set as follows: population size N_g =20, crossover probability P_c =0.8, mutation probability P_m =0.3. We take the roulette wheel as the selection method and set the population size to be the same with the swarm size of the PSO. The results using GA is also presented in Fig. 1. It shows that the PSO algorithm has faster converging speed and better average value than GA.



(a)



(b)

Figure 1. The iteration process of the algorithms

4 Conclusions

In this paper, an approach based on an improved discrete binary particle swarm optimization is used to solve the problem of partner selection in virtual enterprise. Simulation results show that the proposed approach as quick convergence and better performance.

Acknowledgement

This work is supported by the National Science Foundation of China under grant No. 60503015.

References

1. Zhang, S., Poulin, D.: Partnership Management Within the Virtual Enterprise in a Network. International Conference on Engineering Management and Control. (1996)645-650
2. Talluri, S., Baker, R. C.: A quantitative Framework for Designing Efficient Business Process Alliances. International Conference on Engineering Management and Control. (1996)656-661
3. Chu, X. N., Tso, S. K., Zhang, W. J., Li, Q.: Partners Selection for Virtual Enterprises. Proceedings of the 3th World Congress on Intelligent Control and Automation.(2000)164-168
4. Feng, W. D., Chen, J., Zhao, C. J.: Partners Selection Process and Optimization Model for Virtual corporations Based on Genetic Algorithms. Journal of Tsinghua University(Science and Technology). 40(2000):120-124 (in Chinese)
5. Qu, X. L., Sun, L. F.: Implementation of Genetic Algorithm to the Optimal Configuration of Manufacture Resources. Journal of Huaqiao University.26(2005)93-96 (in Chinese)
6. Kennedy, J. E., Eberhart, R. C.: A Binary Binary Version of the Particle Swarm Algorithm. IEEE International Conference on Systems, Man, and Cybernetics. 5(1997)4104-4105
7. Qiang , Z., Shaoze,Y.: Collision-Free Path Planning for Mobile Robots Using Chaotic Particle Swarm Optimization. Lecture Notes in Computer Science.(3612)632-635
8. Yangmin, Y., Xin, C.: Mobile Robot Navigation Using Particle Swarm Optimization and Adaptive. Lecture Notes in Computer Science.(3612)628-631
9. Yin, P. Y.: A Discrete Particle Swarm Algorithm for Optimal Polygonal Approximation of Digital Curves. J. Vis. Commun. Image. 15(2004) 241-240

Fang Gao received the B. degree in computer software engineering in 1996, the M. Degree in computer theory and application, in 2002, both from Harbin institute of Technology, China. Now she is a doctor candidate in Harbin Institute of Technology. Her research interests include supply chain management.

Gang Cui Professor, Doctor adviser, in School of Computer Science, Harbin Institute of Technology. His research interests include computer architecture , high dependency computation and supply chain management.

Qiang Zhao Ph.D, associate professor. He received the Ph.D degree from Harbin institute of Technology. He research interests include supply chain management and logistics optimization.

Hongwei Liu Ph. Doctor, Associate professor, in School of Computer Science, Harbin Institute of Technology. His research interests include high dependency computation and software test.