DISCRETE PARTICLE SWARM OPTIMIZATION ALGORITHM BASED ON INFORMATION SHARING MECHANISM

Qinghe Xu[1], Noppadol Amdee[2*] and Adisak Sangsongfa[3]
[1]Student, Faculty of Industrial Technology, Muban Chom Bueng Rajabhat University,
46 Moo 3, Chombueng District, Ratchaburi 70150, Thailand,
64a951005@mcru.ac.th
[2*,3]Lecturers, Faculty of Industrial Technology, Muban Chom Bueng Rajabhat University,
46 Moo 3, Chombueng District, Ratchaburi 70150, Thailand,
[2*]noppadolamd@mcru.ac.th, [3]adisaksan@mcru.ac.th
** Corresponding author*

ABSTRACT

To better solve NP-complete problems, this paper proposes a Discrete Particle Swarm Optimization algorithm based on an Information Sharing Mechanism (ISM-DPSO). The algorithm uses a mechanism of sharing the pheromone of the ant colony algorithm to update particles; the particle swarm threshold function application is initialized. At the same time, to enhance the algorithm's global search ability, a particle swarm is initialized at a certain probability, and on this basis, using a nonlinear dynamic adjusting pheromone persistent rate strategy to guide the information memory. The proposed algorithm is applied to the knapsack problem with single constraints and the vehicle cargo loading problem with multiple constraints, and the optimization results are compared with the basic discrete particle swarm optimization algorithm (DPSO). The experimental results show that the ISM-DPSO algorithm outperforms the DPSO algorithm in terms of both the optimal value and the algorithm's fitted value, and the search results are superior to those of the DPSO algorithm.

KEYWORDS: Information sharing mechanisms, Pheromone persistence, Discrete particle swarm optimization, Knapsack problems, Cargo dispensing

1. Introduction

In 1997, Kennedy and Eberhart proposed a discrete binary particle swarm algorithm [1] based on the basic particle swarm algorithm [2] to solve combinatorial optimization problems.

This opened a research boom in improving and applying discrete particle swarm algorithms. Zhang et al [3] proposed a multi-particle swarm cooperative optimization to solve the complex coalition generation problem. Lin et al [4] applied a discrete particle swarm algorithm to data center network traffic scheduling research to achieve good results. Xu et al [5] used a discrete binary particle swarm algorithm based on the Pareto criterion to solve the multi-objective optimization problem of distribution network reconfiguration. Liu et al [6] used a hybrid discrete particle algorithm to optimize the Steiner minimum tree construction problem.

From numerous use cases, the study of discrete particle swarm algorithms is often aimed at optimizing NP-complete problems [7]. The 0-1 knapsack problem is a very classical class of combinatorial optimization NP-complete problems, and the study of this problem is of great importance both in theory and in practice, such as vehicle loading, traffic planning, investment decision, material cutting, and optimal resource allocation [8-10] can be modeled as knapsack problems. For the solution of knapsack problems, there are two main categories: the first category is the traditional methods, such as exact solution method, metaheuristic algorithm, hidden enumeration method, branch delimitation method, etc. [11-13], which can quickly obtain the actual optimal value for some small-scale knapsack problems, but with the increasing scale of the problem in practice, the increase of constraints and the exponential growth of the solution space, making These conventional algorithms are complex to obtain the optimal solution. Therefore, scholars have turned their attention to another class of evolutionary algorithms based on swarm intelligence. These swarm intelligence-based algorithms include genetic algorithms [14], ant colony algorithms [15], fish swarm algorithms [16], and particle swarm algorithms [17-20]. These algorithms are increasingly used due to simple implementation and robustness advantages.

For the binary discrete problem, this paper proposes the Discrete particle swarm algorithm based on an information sharing mechanism (ISM-DPSO), which uses the pheromone sharing mechanism of the ant colony algorithm [21] to update particle velocity, borrow the threshold function to initialize the particle swarm, and dynamically adjust the persistence rate of the pheromone. Experiments show that the algorithm can successfully solve the complex 0-1 backpack problem and vehicle cargo allocation problem, and its performance is superior compared with the basic discrete particle swarm algorithm.

2. Model Description

The 0-1 knapsack problem is a combinatorial optimization problem that requires selecting items from a set to maximize the total value without exceeding the capacity of a knapsack. This model is widely applied in various fields, such as investment portfolio selection, resource allocation, cargo loading, steel cutting, employee scheduling, advertising placement, the traveling salesman problem, and course scheduling. It involves making optimal decisions under limited resource constraints. By modeling these problems as 0-1 knapsack problems, one can use relevant algorithms to find the optimal or near-optimal solutions.

2.1 Mathematical model of the backpacking problem

The mathematical function of the backpacking problem can be described as

$$\max f(X) = \sum_{j=1}^{n} p_j x_j \tag{1}$$

$$s.t. \quad \sum_{j=1}^{n} g_j x_j \le C \tag{2}$$

Where n is the number of items, g_j is the item's weight, is the item's value, C is the backpack's capacity $X = \{x_1, x_2, ..., x_n\}$, x_j is a 01 variable, $j \in [1, \mathrm{n}]$, when x_j = 1 means the j item is selected, when x_j = 0 means the j th item is not selected.

2.2 Mathematical model for vehicle cargo dispensing

The vehicle cargo assembly problem mainly studies how the cargo is assembled to reasonably use the vehicle volume and load. Suppose the truck's capacity is C and the load is G. There are i (i = 1,2,3,...,n) cargoes of different weights and volumes waiting to be assembled. Assume that.

1) The boxed cargo has no priority, and the cargo and OD volumes need to be more compressible.

2) The goods are delivered to the same destination, and the volume and weight of each type differ.

Then, the objective function is as follows.

$$\max f(x) = \lambda \sum_{i=1}^{n} g_i x_i + \eta \sum_{i=1}^{n} c_i x_i \tag{3}$$

$$s.t. \quad \sum_{i=1}^{n} g_i x_i \le G \tag{4}$$

$$\sum_{i=1}^{n} c_i x_i \le C \tag{5}$$

$$0 \le \mathrm{g}_i \le G \tag{6}$$

$$0 \le c_{\mathrm{i}} \le C \tag{7}$$

$$\lambda + \eta = 1 \tag{8}$$

Where: g_i is the weight of cargo i, c_i is the volume of freight i, $\lambda \in (0,1)$, when $\lambda = 1$, it means that the maximum load capacity is sought; $\eta \in (0,1)$, when $\eta = 1$, it means that the maximum volume utilization is desired; x_i is a 01 variable, then x_i takes the value of 1, it means that this batch of cargo needs to be loaded with cargo i, and vice versa. If the optimal solution is (1,1,0,1,0,...), the cargo to be loaded is 1, 2, 4, ...

3. Discrete particle swarm algorithm based on information sharing mechanism

3.1 Basic particle swarm algorithm

The individuals in the particle swarm algorithm, which we call particles, search for the optimal value in the solution space by mimicking the predatory behavior of birds and tracking two "extremes": one is to search for the current optimal value, called the individual extremum and the other is to search for the current optimal value of the whole population, called the global extremum. The speed and position update equations are shown below

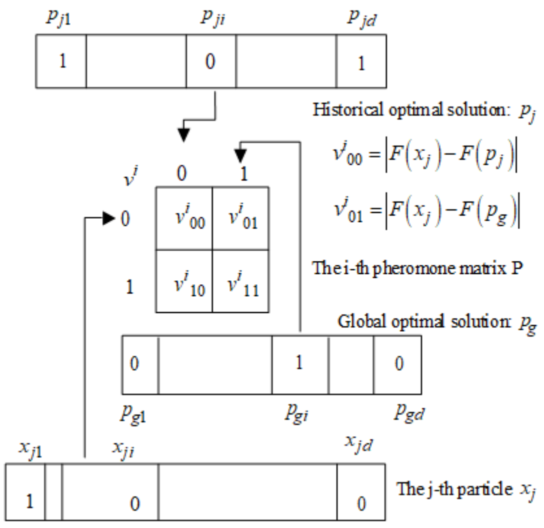
$$v_j^{k+1} = \omega v_j^k + c_1 \xi (p_j^k - x_j^k) + c_2 \eta (p_g^k - x_j^k) \tag{9}$$

$$x_j^{k+1} = x_j^k + v_j^{k+1} \tag{10}$$

where $v_j = [v_{j1}, v_{j2}, ..., v_{jm}]$ is the velocity of a particle j, which represents the distance between the present position of a particle j and its next target position; ω is called the inertia weight; c_1 is the weight coefficient of the particle tracking its own historical optimum, which represents the particle's knowledge and is called the cognitive constant; c_2 is the weight coefficient of the particle monitoring the population optimum, which means the particle's knowledge of the whole population and is reached the social knowledge constant; ξ and η is a random number uniformly distributed in the interval [0,1]; p_j is the individual optimum searched so far; p_g is the optimum searched so far for the whole population; $x_j = [x_{j1}, x_{j2}, ..., x_{jm}]$ is the current position of the particle j. The particle population is updated iteratively according to Eq. (9) and (10) Until a satisfactory solution is found or a certain number of iterations is reached, the algorithm stops computing and outputs the search result.

3.2 Discrete particle swarm algorithm based on information sharing mechanism

The core of the primary particle swarm algorithm adjusted to a binary discrete particle swarm algorithm is to change the successive states of each particle to only two states of 0 or 1. There are only four possibilities for transforming the current state of a particle to the next state: transforming from 1 state to 1 state, from 1 state to 0 state, from 0 state to 0 state, and from 0 state to 1 state. The state change of each bit of a particle will be determined by the pheromone stored in the pheromone matrix P shared by all particles in that bit. For example, if the state of the i th bit of the j th particle is 0, the state of the historical optimum at the i th bit is 0. The state of the global optimum particle is 1. The pheromone generated by the particle changing from the current state 0 to the historical optimum 0 is v_{00}^i, and the pheromone generated by the particle changing from the current state 0 to the global optimum one is v_{01}^i, The basic principle is shown in Figure 1.



p_{j1}
p_{ji}
p_{jd}
1
0
1
Historical optimal solution: p_j
$v^j_{00} = |F(x_j) - F(p_j)|$
$v^j_{01} = |F(x_j) - F(p_g)|$
v^j
0
1
v^j_{00}
v^j_{01}
v^j_{10}
v^j_{11}
The i-th pheromone matrix P
Global optimal solution: p_g
0
1
0
p_{g1}
p_{gi}
p_{gd}
x_{j1}
x_{ji}
x_{jd}
1
0
0
The j-th particle x_j

Figure 1 Basic schematic diagram

A binary sequence is constructed using the threshold function for initializing the particle population, and the mathematical expression is shown below.

$$\eta_s(x) = \begin{cases} 0, & x < s \\ 1, & x > s \end{cases} \tag{11}$$

A binary sequence of corresponding d-bit length is obtained by iterating the Logistic mapping d times, constituting the particle's initial state. Where: s is a constant, taken as 0.5 in this paper.

Also, to enhance the global search capability of the ISM-DPSO algorithm, the particle population is reinitialized with probability k, which can be expressed as

$$P_j = R_j(P_j), \quad rand() < k \tag{12}$$

Where: P_j denotes the j th particle, $R_j(P_j)$ denotes re-initialization of P_j, $rand()$ is a random number between [0,1], k is a constant, and k =0.01 is taken in this paper.

In this paper, the particle swarm velocity update formula is reselected by drawing on the pheromone-sharing mechanism in the ant colony algorithm. The updated formula is as follows.

$$v(x_j(i), p_j(i), i)^{t+1} = \rho \times v(x_j(i), p_j(i), i)^t + (1-\rho)(F(p_j) - F(x_j)) \tag{13}$$

$$v(x_j(i), p_g(i), i)^{t+1} = \rho \times v(x_j(i), p_j(i), i)^t + (1-\rho)(F(p_g) - F(x_j)) \tag{14}$$

Where: $i = 1,...,d$ is the length (dimension) of the particle; j denotes the j th particle $j = 1,...,n$; x_j denotes the current position of the particle, p_j denotes the historical optimal position of the particle, p_g denotes the optimal position of the particle population; $x_j(i)$ denotes the value of the i th position of the j th particle, $p_j(i)$ denotes the value of the historical optimal i th position of the j th particle, $p_g(i)$ denotes the value of the global optimal i th position; F is the particle fitness function, t represents the number of iterations; v_i^{t+1} is the velocity pheromone matrix of the i th position at t+1th iteration, using a velocity pheromone update one particle after another. ρ denotes the pheromone persistence rate, and 1- ρ represents the evaporation rate. The smaller the value ρ, the faster the pheromone evaporates, and the information retained in the past will be forgotten more quickly. This paper uses a strategy of nonlinear dynamic adjustment of pheromone persistence rate to guide the memorability of information. The specific expression is as follows.

$$\rho = -(\rho_{\max} - \rho_{\min}) \times (iter / Gen)^m + \rho_{\max} \tag{15}$$

Where ρ takes values in the range [0.9,0.1], m is a constant, $\rho_{\max}$ and denotes the maximum and minimum pheromone persistence rates.

This approach converts the continuous particle swarm algorithm into a discrete binary particle swarm algorithm on the one hand and, on the other hand, enables the particles to judge the flight direction based on the persistence and timeliness of their information and also to refer to the information left behind by other particles, thus allowing all particle information in the population to be shared adequately and promptly.

The ISM-DPSO algorithm, through the information-sharing mechanism, can effectively handle the correlated factors in the problem. It allows particles to consider the information of other particles during the search process, thereby capturing the factor correlations within the problem. The strategy of dynamically adjusting the pheromone persistence rate enables the algorithm to adapt to the search requirements at different stages. For problems with high correlations, the algorithm can enhance its adaptability to correlated factors by adjusting the pheromone persistence rate.

3.3 ISM-DPSO algorithm steps

This paper proposes a discrete particle swarm algorithm based on an information-sharing mechanism. The algorithm's detailed steps are as follows.

Step 1: Initialization and Parameter Settings.

- Set the total number of particles, maximum number of iterations, and particle length.
- Define the value and weight of the items as well as the capacity of the knapsack.

Step 2: Initialization of the Velocity Information Matrix.

- Initialize a three-dimensional matrix u for storing velocity information.

Step 3: Construction of Particles.

- Randomly initialize the position of particles, where a binary array of length n represents each particle, indicating whether the corresponding item is selected.

Step 4: Particle Correction to Create Feasible Solutions.

- Check if each particle exceeds the knapsack capacity, and if so, remove items until the capacity is not exceeded.

Step 5: Start the Particle Swarm Optimization Algorithm.

- Begin the iterative process to calculate the fitness of each particle (total value).
- Update the global and local historical best particles.
- Update the velocity information matrix u of the particles.
- Update the position of the particles based on the velocity information matrix.
- Correct the particles to ensure they do not exceed the knapsack capacity.
- Record the global best fitness value and plot the changes in the best fitness during the iteration process.
- Output the final global best fitness value.

4. Experimental Analysis

4.1 KP problem example analysis

This paper uses four widely cited examples of well-known KP problems to analyze the performance of the discrete particle swarm algorithm based on the information-sharing mechanism. The hardware used in this experiment is an Intel(R) Core(TM) i5-6200U CPU @ 2.30 GHz with 8.00 GB RAM; the software is Windows 10 and MATLAB.

Example 1: The set of weights of items W=[71,34,82,23,1,88,12,57,10,68,5,33,37,69, 98,24,26,83,16,26,18,43,52,71,22,65,68,8,40,40,24,72,16,34,10,19,28,13,34,98,29,31,79,33, 60,74,44,56,54,17], the set of values of items P = [26,59,30,19,66,85,94,8,3,44,5,1,41,82, 76,1,12,81,73,32,74,54,62,41,19,10,65,53,56,53,70,66,58,22,72,33,96,88,68,45,44,61,78,78, 6,66,11,59,83,48], the maximum capacity of the backpack is C=300 and the size is d= 50.

Example 2: The set of weights of items W=[19,53,61,74,98,70,15,59,64,29,98,79,74, 85,52,70,84,91,84,75,78,72,5,46,26,95,38,79,28,92,12,37,37,58,94,44,25,3,12,67,2,98,12,2, 34,68,68,81,92,16], the set of values of items P = [12,61,63,78,49,46,44,36,37,66,43,16, 14,73,72,72,83,15,83,65,21,49,36,20,22,80,94,3,73,1,91,62,8,58,79,67,53,8,85,82,70,43,22, 53,22,14,41,41,77,75], the maximum capacity of the backpack is C=500 and the size is d=50.

Example 3: The set of weights of items W=[3,68,24,80,76,9,24,2,46,75,56,41,95,46, 23,34,64,76,6,48,25,73,87,67,58,7,93,66,55,75,38,27,53,6,100,36,26,17,53,88,21,9,34,90,32, 47,4,6,57,50,30,25,41,24,12,74,92,17,32,96,35,76,52,93,64,55,1,70,26,35,2,97,82,22,41,37, 63,28,90,13], the set of values of the items P=[38,16, 29,47,22,25,17,49,15,15,75,11, 56,99,51,92,59,37,13,98,61,50,32,17,44,79,41,53,45,29,62,64,2,23,31,45,57,68,57,26,51,26, 86,83,94,20,98,24,91,89,1,63,21,46,74,56,64,72,58,8,74,24,27,35,94,49,65,21,16,25,1,45,63, 4,37,25,39,68,49,11], the maximum capacity of the backpack C=800 and the scale d=80.

Example 4: The set of weights of items W=[42,30,27,93,8,34,47,64,82,76,70,79,23,5, 67,9,97,29,7,61,73,3,44,85,7,51,49,90,59,38,55,39,62,85,54,81,38,42,90,90,26,50,22,71,52, 41,77,32,49,2,96,84,20,48,17,62,87,94,84,26,73,52,12,70,42,47,94,13,47,89,90,7,51,39,24,6, 74,69,5,47], the set of values of the items P = [15,64,82,87,81,54,65,98,42,99,6,50,90,99,96, 57,76,12,47,18,46,73,99,60,40,60,15,5,65,69,19,72,51,33,11,72,69,64,97,95,32,59,34,3,27,99, 82,44,66,83,72,28,64,90,15,38,65,91,68,28,17,80,1,29,54,8,11,11,70,40,93,65,51,49,75,35,41, 60,72,57], the maximum capacity of the backpack C=1000 and the scale d=80.

The population size of all experiments is 20, and each function is run independently for 30 times with 1000 iterations. This paper takes the maximum and minimum retention factors $\rho_{\min} = 0.1$ for pheromones. The inertia weights for both ISM-DPSO and DPSO algorithms are linearly decreasing strategies, which reduce linearly with iterations between [0.95,0.4], and the obtained data are shown in Table 1 and Figure 2. Where OPT denotes the optimal value that the instance can be found, Best, Worst, and Mean are the best, worst, and average values of the results obtained by each algorithm in 30 independent calculations; Gap is the relative difference between the optimal value OPT and the average value Mean, as shown in equation (16). The closer the Gap fitting value is to 0, the better the algorithm's average performance. Time is the average duration over 30 runs.

The mathematical expression is.

$$Gap = \frac{|OPT - mean|}{OPT} \times 100\% \tag{16}$$

Table 1 Comparison table of the optimization results of the two algorithms

No	Examples	OPT	Algorithm	Best	Worst	Mean	Gap	Time
1	Example 1	1063	DPSO	1048	1015	1030.8	0.030	4.72
		1063	ISM-DPSO	1063	1034	1052.2	0.010	4.65
2	Example 2	1153	DPSO	1153	1118	1136.8	0.014	6.94
		1153	ISM-DPSO	1153	1148	1152	0.001	6.88
3	Example 3	2085	DPSO	1959	1888	1920	0.079	7.15
		2085	ISM-DPSO	2085	1963	2012.3	0.035	7.04
4	Example 4	2337	DPSO	2267	2176	2234.9	0.044	10.12
		2337	ISM-DPSO	2337	2267	2302.8	0.015	9.36

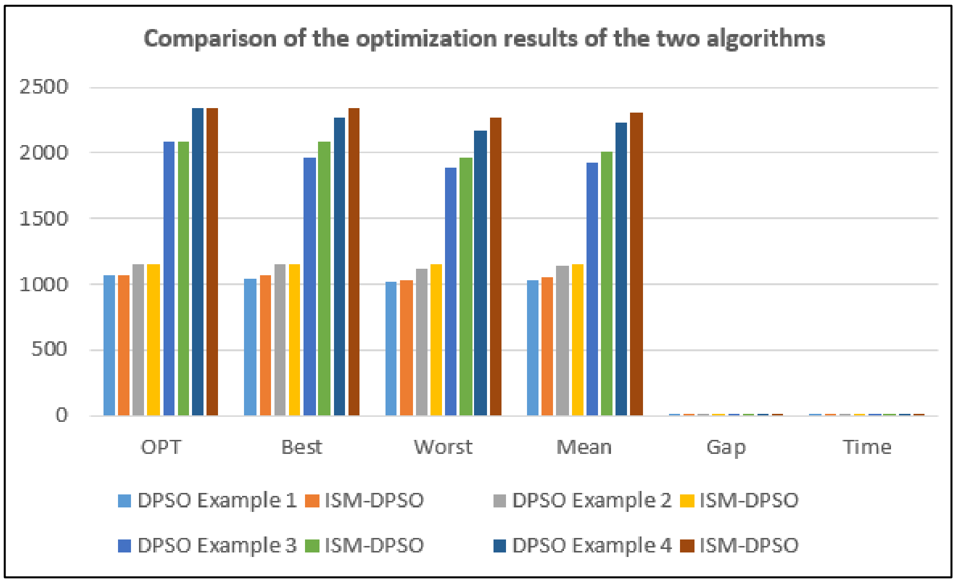


Figure 2 Comparison of the optimization results of the two algorithms

As seen from Table 1 and Figure 2, for the above four examples, ISM-DPSO shows its superiority over the DPSO algorithm in terms of both the optimal value and the fitted value of the algorithm, and the search result is better than that of the DPSO algorithm.

Figure 3-figure Supplement 6 shows the curve of the maximum value of each instance as the number of iterations increases. Figure 3-figure supplement 5 shows that the maximum value is generated at 735 iterations for instance 1, 396 for instance 2, 995 for instance 3, and 826 for instance 2. Thus, we find that more iterations are required to apply ISM-DPSO to obtain the maximum value.

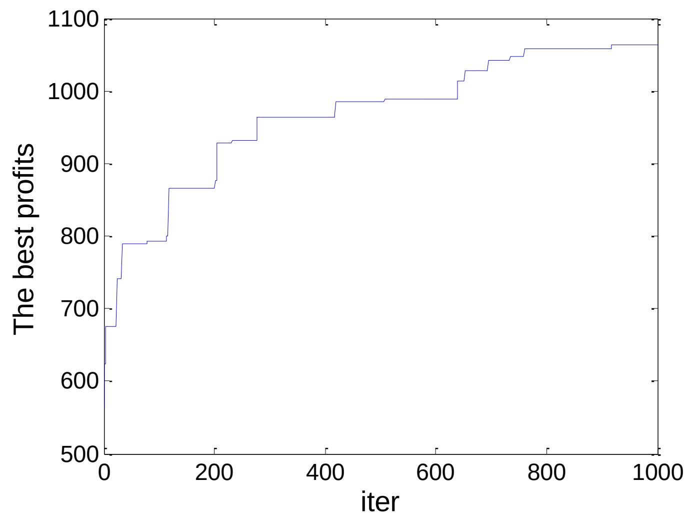


Figure 3 Example 1 Maximum value

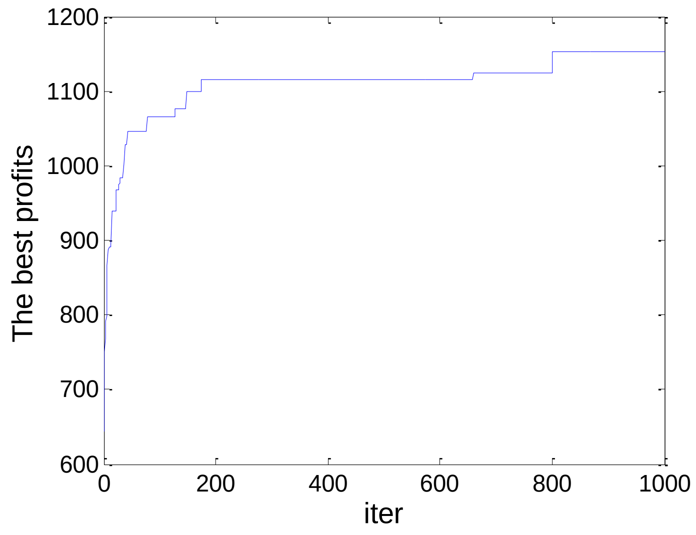


Figure 4 Example 2 Maximum value

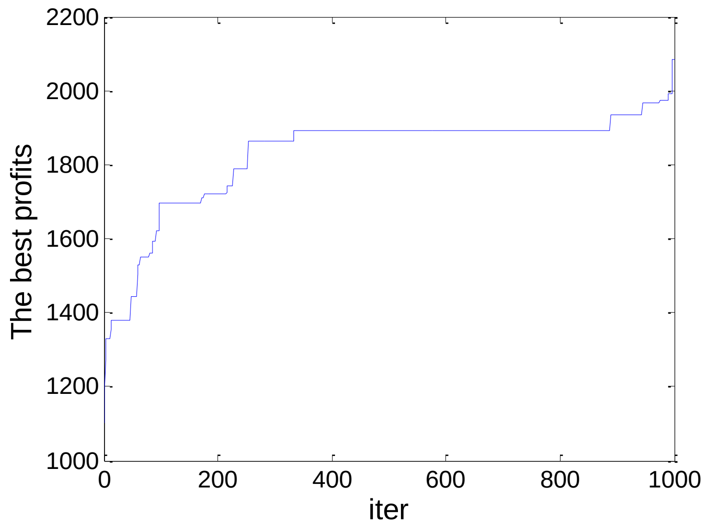


Figure 5 Example 3 Maximum value

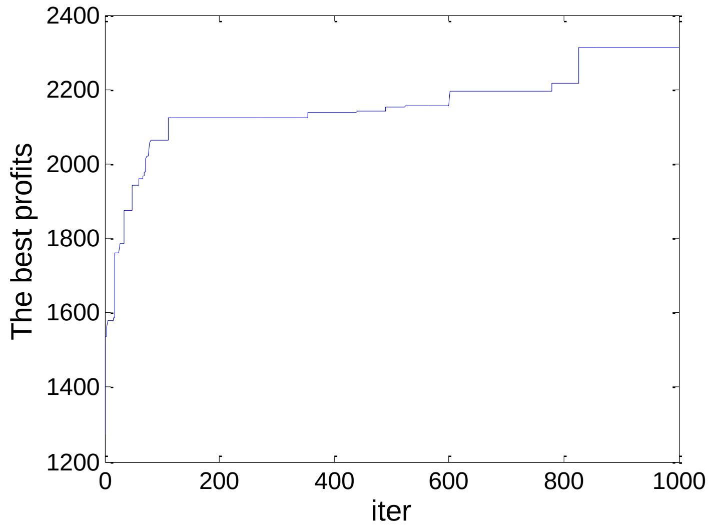


Figure 6 Example 4 Maximum value

To verify the variation of ISM-DPSO performance with the number of iterations, the above four examples were experimented with 1000, 2000, 3000, 5000, and 10000 iterations. Other parameters are kept constant, and the mean values obtained from 30 independent runs are shown in Table 2. Mean1 is the mean value when the number of iterations is 1000, Mean2 is the mean value when the number of iterations is 2000, Mean3 is the mean value when the number of iterations is 3000, Mean4 is the mean value when the number of iterations is 5000, and Mean5 is the mean value when the number of iterations is 10000.

Table 2 Variation of the mean value with the number of iterations

No	Examples	OPT	Algorithm	Mean1	Mean2	Mean3	Mean4	Mean5
1	Example 1	1063	DPSO	1030.8	1032.1	1033.6	1033.4	1034
		1063	ISM-DPSO	1052.2	1054	1055.4	1058	1058
2	Example 2	1153	DPSO	1136.8	1140	1144	1143	1144.2
		1153	ISM-DPSO	1152	1152.2	1152.4	1152.5	1152.8
3	Example 3	2085	DPSO	1920	1928	1930	1930.3	1929
		2085	ISM-DPSO	2012.3	2022.8	2032.6	2038.4	2038.5
4	Example 4	2337	DPSO	2234.9	2250	2248	2260	2252
		2337	ISM-DPSO	2302.8	2308	2314	2317.6	2319.8

As seen from Table 2, the mean value of ISM-DPSO is closer to the optimal value as the number of iterations increases in a specific pack size range. And the mean value of DPSO does not change significantly as the number of iterations increases. With the increase in the backpack size, especially for the super-large backpack problem, ISM-DPSO is better than DPSO in finding the optimal value. Still, it needs to be revised, which indicates that the information-sharing mechanism-based optimization approach needs to be stronger after the complexity of the problem exceeds a specific size. New methods need to be found to optimize it.

4.2 Vehicle cargo dispensing problem-solving

A box truck with a load capacity of 30t and a compartment volume of 70m3 is to be fitted to 14 kinds of cargoes selected, the weight set of various cargoes W=[2.00, 3.80, 3.05, 5.00, 7.10, 5.00, 4.30, 1.05, 2.05, 1.80, 5.05, 2.85, 4.30, 4.50] in t; the volume set of cargoes V = [5.30, 12.00, 7.10, 10.00, 6.00, 13.70, 12.00, 1.50, 5.90, 3.05, 3.90, 3.00, 10.50, 8.50], in m3, which was obtained from the literature (Ma et al., 2015). The population size of all experiments at this time was 20, the dimensionality of the search space was 14 dimensions, the maximum number of iterations was 1000, and other parameter settings were kept constant. The functions were run 30 times independently using DPSO and ISM-DPSO, respectively, and the results obtained are shown in Table 3.

Table 3 Comparison of the results obtained by the two methods

Algorithm	DPSO	ISM-DPSO
Optimal solution code	10110101110011	11110011010011
Total volume of loaded cargo (m3)	65.55	69.95
Volume utilization (%)	93.60	99.93
Total weight of loaded cargo (t)	28.75	29.8
Load utilization (%)	95.83	99.93

As shown in Table 3, the volume utilization rate of ISM-DPSO is as high as 99.93%, while the volume utilization rate of DPSO is only 93.60%. In terms of load, the load utilization of ISM-DPSO is 99.93%, while the load utilization of DPSO is only 95.83%. It can be seen that ISM-DPSO can balance the load and volume utilization rates well, making full use of the vehicle loading space and effectively solving the problem of vehicle cargo distribution.

4.3 Scenario Application Guidance

This article experimentally validated the performance of the ISM-DPSO algorithm in solving 0-1 knapsack problems and vehicle cargo allocation problems. The results indicated that the ISM-DPSO algorithm outperformed the traditional DPSO algorithm in multiple independent experiments, particularly in finding global optimal solutions and demonstrating higher stability and quality.

The article calculated the average performance metric of the algorithm, known as the Gap value, which is the relative difference between the algorithm's average and the optimal value. The closer the Gap value is to 0, the closer the algorithm's average performance is to the optimal solution, indicating higher efficiency. In our experiments, the ISM-DPSO algorithm showed a smaller Gap value in all test cases, proving its high efficiency. To effectively apply the ISM-DPSO algorithm in various scenarios, it is first necessary to confirm whether the problem is suitable for modeling as a 0-1 knapsack problem or a combinatorial optimization problem with a similar structure, such as investment portfolio selection, resource allocation, cargo loading, steel cutting, employee scheduling, advertising placement, and the traveling salesman problem, all of which can be modeled as 0-1 knapsack problems. Subsequently, it transforms the actual problem into a mathematical model that ISM-DPSO can handle, defining the necessary decision variables, objective functions, and constraints. Then, the algorithm parameters are adjusted according to the characteristics of the problem, such as the size of the particle swarm and the pheromone persistence rate, and the algorithm is implemented and tested on a series of well-designed test problems to evaluate its performance. Assess the algorithm's efficiency through metrics such as the Gap value and average running time and compare it with other algorithms to showcase its advantages. Adjust parameters or strategies based on the evaluation results to optimize the algorithm's performance, apply it to real-world problems, and monitor and adjust to fit actual situations. Finally, the application process and algorithm results will be documented and shared.

5. Conclusions

The proposed discrete particle swarm optimization algorithm based on an information-sharing mechanism (ISM-DPSO) significantly improves performance in solving 0-1 knapsack problems and vehicle cargo allocation problems by enhancing the initialization of the particle swarm and incorporating a pheromone-sharing mechanism. Experimental results demonstrate that ISM-DPSO outperforms the traditional DPSO algorithm in finding global optimal solutions and exhibits higher stability and quality across multiple independent experiments.

Despite its superior performance in many cases, ISM-DPSO shows limitations when addressing extremely large-scale problems. Future research could explore combining other optimization techniques, such as hybrid intelligent algorithms or adaptive strategies, to enhance further the algorithm’s global search ability and solution quality. Additionally, the applicability of ISM-DPSO could be extended to a broader range of real-world optimization problems to validate its wide adaptability and practicality.

Overall, ISM-DPSO is an effective optimization algorithm with promising applications. It demonstrates significant advantages in solving complex discrete optimization problems and provides a solid foundation for further research and application.

Acknowledgment

The authors would like to thank the Department of Industrial Technology Management, Faculty of Industrial Technology, Muban Chombueng Rajabhat University, for supporting the research tools used in this research.

References

[1] Kennedy J, Eberhart R. A discrete binary version of the particle swarm algorithm. Proceedings of the 1997 Conference on System, Man, and Cybernetics; 1997 Oct 12-15; Orlando, USA. Piscataway, USA: IEEE; 1997. p.4104-8.

[2] Kennedy J, Eberhart R. Particle swarm optimization. Proceedings of the IEEE International Conference on Neural Networks; 1995 Nov 27-Dec 1; Perth, Australia. Piscataway, USA: IEEE; 1995. p. 1942-8.

[3] Zhang G, Jiang J, Xia N, Su Z. Solutions of complicated coalition generation based on discrete particle swarm optimization. Acta Electronica Sinica 2007;35(2):323-7.

[4] Lin Z, Gao W, Wu C, Li Y. Data center network flow scheduling based on DPSO algorithm. Acta Electronica Sinica 2016;44(9):2197-202.

[5] Xu Z, Yang W, Zhang W, Chen S. Multi-objective distribution network reconfiguration based on chain loops and improved binary particle swarm optimization. Power System Protection and Control 2021;49(6):114-23. (In Chinese)

[6] Liu G, Huang Y, Wang X, Guo W, Chen G. Hybrid discrete particle swarm algorithm for X-architecture steiner minimal tree construction with slew constraints. Chinese Journal of Computers 2021; 4(12):2542-59. (In Chinese)

[7] Garey MR, Johnson DS. Computers and intractability: a guide to the theory of NP-completeness. San Francisco, USA: W. H. Freeman and Company;1979.

[8] Bas E. A capital budgeting problem for preventing workplace mobbing by using analytic hierarchy process and fuzzy 0-1 bidimensional knapsack model. Expert Systems with Applications 2011;38(10):12415-22.

[9] Han D, Huang W, Yan Z. Virtual Bidding Strategy in Electricity Market Based on Deep Reinforcement Learning. Proceedings of the CSEE 2022;42(4):1443-54. (In Chinese)

[10] Shih W. A branch and bound method for the multiconstraint zero-one knapsack problem. Journal of the Operational Research Society 1979;30(4):369-78.

[11] Storn R, Price K. Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization 1997;11(4): 341-59.

[12] He Y, Wang J, Zhang X, Li H, Liu X. Encoding transformation-based differential evolution algorithm for solving knapsack problem with single continuous variable. Swarm and Evolutionary Computation 2019;50: 981-92.

[13] Jiang H, Guo T, Yang Z. Research progress of vehicle routing problem. Acta Electronica Sinica 2022;50(2):480-92.

[14] He Y, Wang X, Li W, Zhang X, Chen Y. Research on genetic algorithms for the discounted {0-1} knapsack problem. Chinese Journal of Computers 2016; 39(12):2614-30. (In Chinese)

[15] Ren Z, Zhao S, Huang S, Liang Y. Hybrid optimization algorithm of ant colony optimization and Lagrangian relaxation for solving multidimensional knapsack problem. Control and Decision 2016; 31(7):1178-84. (In Chinese)

[16] Li Y, Zhang J, Liu Q, Zhang W. Advanced artificial fish swarm algorithm for large scale multiple knapsack problem. Systems Engineering and Electronics 2018; 40(3):710-6.

[17] Liu Y, Ma L. Solving 0-1 knapsack problem by fuzzy particle swarm optimization. Application Research of Computers 2011;28(11):4026-31. (In Chinese)

[18] Geng Y, Wu F S. Research on knapsack problem based on the hybrid algorithm of particle swarm optimization and simulated annealing. Control Engineering of China 2019;26(5):991-6. (In Chinese)

[19] Xu Q, Wu T, Wang W. Nonlinear dissipative particle swarm algorithm and its applications. IEEE ACCESS 2021;(9):158862-71.

[20] Xu Q, Amdee N, Sangsongfa A. Design study of high-power PV grid-connected inverter system based on the particle swarm algorithm. Primera Scientific Engineering 2024;5(3):16-38.

[21] Xiao J, Yu X, Zhou G, Sun K, Zhou Z. An improved ant colony algorithm for indoor AGV path planning. Chinese Journal of Scientific Instrument 2022;43(3):277-85. (In Chinese)

[22] Ma H, Qiu X. A heuristic algorithm for the ordinary scattered cargoes' container-loading question. In: Liu R, Zhang J, Guan C, editors. Logistics: The Emerging Frontiers of Transportation and Development in China. Reston, USA: ASCE; 2009. p. 4790-95.

Author's Profile

Mr.Qinghe Xu, (Ph.D. Student) Department of Industrial Technology Management, Faculty of Industrial Technology, Muban Chom Bueng Rajabhat University, Ratchaburi, Thailand.

Email: 64a951005@mcru.ac.th

Interested Research: Artificial Intelligence Algorithm and New Energy

Asst. Prof. Dr.Noppadol Amdee, Department of Industrial Technology Management, Faculty of Industrial Technology, Muban Chom Bueng Rajabhat University, Ratchaburi, Thailand.

Email: noppadolamd@mcru.ac.th

Interested Research: Design of Experiment, Artificial Intelligence Algorithm, and Cost Analysis

Dr.Adisak Sangsongfa, Department of Industrial Technology Management, Faculty of Industrial Technology, Muban Chom Bueng Rajabhat University, Ratchaburi, Thailand.

Email: adisaksan@mcru.ac.th

Interested Research: Artificial Intelligence Algorithm

Article History:

Received: September 29, 2024

Revised: December 15, 2024

Accepted: December 16, 2024