Application Analysis on PSO Algorithm in the Discrete Optimization Problems

Qinglong Chen 1,a, Yong Peng1,b*, Miao Zhang1,c, Quanjun Yin1,d  
1College of Systems Engineering, National University of Defense Technology, Changsha, China  
aemail: chenqinglong14@nudt.edu.cn, cemail: zhangmiao@nudt.edu.cn, demail: yinquanjun@nudt.edu.cn,  
*Corresponding author: bemail: yongpeng@nudt.edu.cn

Abstract: Particle Swarm Optimization (PSO) is kind of algorithm that can be used to solve optimization problems. In practice, many optimization problems are discrete but PSO algorithm was initially designed to meet the requirements of continuous problems. A lot of researches had made efforts to handle this case and varieties of discrete PSO algorithms were proposed. However, these algorithms just focus on the specific problem, and the performance of it significantly degrades when extending the algorithm to other problems. For now, there is no reasonable unified principle or method for analyzing the application of PSO algorithm in discrete optimization problem, which limits the development of discrete PSO algorithm. To address the challenge, we first give an investigation of PSO algorithm from the perspective of spatial search, then, try to give a novel analysis of the key feature changes when PSO algorithm is applied to discrete optimization, and propose a classification method to summary existing discrete PSO algorithms.

1. Introduction
Particle Swarm Optimization (PSO) algorithm was inspired by flocking behavior of birds and first proposed by Kennedy and Eberhart[1] in 1995, which is an optimization algorithm to search for the optimal solution in the space containing all feasible solutions. Its core idea is that the particles in the population use the best solutions found by the individual and the population as the guidance information to further search and find a better solution. When the guidance information can no longer guide the particles to find a better solution, the algorithm search comes to an end. The algorithm is mainly characterized by simple calculation, fast converging rate and ideal performance as compared to other algorithms. Therefore, it has been widely used in various fields for more than 20 years, and become one of the research hotspots in the field of intelligent computing.

On the one hand, PSO algorithm has been extensively analyzed by scholars to improve its performance since its birth. These works mainly focus on parameter value range analysis [4] [9] [46] [47], convergence or exploration analysis[3] [7] -[11] and trajectory analysis[6], which ultimately all about the analysis of the parameters. On the other hand, PSO algorithm was initially designed to solve the continuous optimization problems, but many problems are discrete in practical. Thus, how to apply PSO algorithm to discrete optimization has attracted extensive attention of researchers, and it is vital to have a profound analysis for the differences between the use of PSO algorithm in continuous optimization problems and in discrete optimization problems. Unfortunately, although varieties of
discrete PSO algorithms have been proposed, they were the solutions to particular problems and did not analyze characteristics of the application of PSO in discrete optimization. In addition, it is difficult to find the difference between discrete and continuous application of PSO algorithm based on the analysis of parameters. Therefore, no unified analysis and summary principles or methods have been proposed for these discrete PSO algorithms at present, which will significantly affect the application of PSO algorithm in discrete field.

To encounter these challenges mentioned above, in this paper, we first give an overview of the general PSO algorithm (in this paper, the general PSO refers to the PSO algorithm proposed by Kennedy and Eberhart for optimal search in continuous fields), and then analyze them from the perspective of spatial search. It is concluded that the construction of the next-step exploration space of particles in the algorithm is the key part of the design of the general PSO algorithm. The other parts, such as the analysis and adjustment of parameters, indirectly affect the performance of the algorithm by the adjustment of the particles’ next-step exploration space. At the same time, this paper analyzes some default premises when the general PSO algorithm is executed and the changes of these default premises when the algorithm is applied to the discrete optimization problem, and finds out that the key factors affecting the discrete optimization performance of PSO algorithm is the distance measurement among solutions in space and the change of fitness value in space under the corresponding distance measurement. Based on the previous analysis, we use the distance measurement relationship between solutions in the solution space as the analysis principle to analyze and classify the discrete PSO algorithms.

The main contributions of this paper are as follows:

- It is pointed out that the next-step exploration space is the most key part of the design of the particle swarm optimization algorithm.
- The core changes of the PSO algorithm, the change of distance measurement method and the corresponding change of the solution space structure, applied to the discrete optimization problems, is presented.
- Existing discrete PSO algorithms is classified and summarized from the perspective of distance measurement.

In this paper, the next-step exploration space and the next-step search space imply the same concept. The rest of this paper is as follows: the second chapter summarizes the general PSO algorithm; chapter 3 analyzes the two premises and backgrounds of general PSO algorithm that will change in discrete optimization problems are analyzed; chapter 4 classifies and summarizes the discrete PSO algorithm based on the distance measurement relationship; finally, the conclusion and expectation are given in chapter 5.

2. PSO Overview

The PSO algorithm usually maintains a population containing a certain number of individuals. The individuals in the population mainly depend on two variables to search in the search space: position \( \left( x_i = (x_{i1}, x_{i2}, \ldots, x_{in}) \right) \) and velocity \( \left( v_i = (v_{i1}, v_{i2}, \ldots, v) \right) \), where the position represents the feasible solution, while the velocity denotes the next-step search direction of the corresponding particle, in which \( n \) refers to the number of particles in the population, and \( i \) indicates that the particle is the \( i^{th} \) particle in the population. In the search process, the algorithm updates the position through velocity in order to find a better position in the search space, that is, a better solution. At the same time, each particle will maintain two aspects of guidance information to update the velocity: the optimal solutions currently found by each particle and the whole population, namely \( pbest \) and \( gbest \). The core idea of the PSO algorithm is to update the position (feasible solution) and velocity (search direction) by using the guidance information of \( gbest \) and \( pbest \). In 1998, Shi[2] added inertia weight to the algorithm to form the standard version of PSO algorithm. The updating of position and velocity is shown in Equations (1) and (2):

\[
x_i^{t+1} = x_i^t + v_i^t
\]

\[
v_i^{t+1} = \omega v_i^t + c_1 r_{i1}(pbest_i^t - x_i^t) + c_2 r_{i2}(gbest_i^t - x_i^t)
\]
Where, \( x^t_i \) represents the position of the \( i \)th particle after the \( t \) rounds of iteration. \( v^t_i \) means the velocity of the \( i \)th particle after the \( t \) rounds of iteration. \( \omega, c_1 \) and \( c_2 \) are constants, representing inertia weight, individual learning factor and group learning factor. \( r_{1i} \) and \( r_{2i} \) denote random numbers within \([0,1]\). In the iterative process, the position is updated by velocity, and the velocity update includes three parts: the first is the current velocity of the particle; the second is the guidance of the optimal position found by the particle individual; and the third is the guidance of the optimal position found by the population. Generally speaking, the three parts represent the inertia part (i.e., the memory of past search), the exploration part and the convergence part respectively. The three parts work together to realize the search of space, and the three, \( \omega, c_1 \) and \( c_2 \), are the adjustment of the action weight of the three parts.

The PSO algorithm is a process of the particles searching in the space composed of all feasible solutions. The next position a particle possibly reach is actually a tiny fraction of space, which shown in Figure 1. We called it the next-step exploration space of particles in this paper. The detailed analysis is given as follows.

At each iteration, the particles follow the vectors represented by two red solid lines (\( c_1(g_{best}^t_i - x^t_i) \) and \( c_2(p_{best}^t_i - x^t_i) \)) according to the guidance of \( p_{best} \) and \( g_{best} \) and the guidance of the current velocity. First, the random numbers \( r_{1i} \) and \( r_{2i} \) in \([0,1]\) form a possible search area \( c_1r_{1i}(g_{best}^t_i - x^t_i) + c_2r_{2i}(p_{best}^t_i - x^t_i) \) surrounded by a red rectangle in the lower half of Figure 1 (when \( c_1 \) and \( c_2 \) are less than 1, it is the case shown in the figure). Further, under the action of the inertia part \( \omega v^t_i \), the red area deviates from the blue rectangular surrounding area in the upper part of Figure 1 to form the next-step exploration space of the particle. The particle’s next position may fall anywhere in the next-step exploration space. The current position of the particle is \( x^t_i \). \( p_{best} \) and \( g_{best} \) determine the basic shape and size of the exploration space. The parameters \( c_1 \) and \( c_2 \) control the scaling and shape change of the exploration space. The parameter \( \omega \) affects the offset of the exploration area to the current velocity direction.

![Figure 1. Process of Particle Movement in PSO Algorithm and Schematic Diagram of Next-step Exploration Space](image)

At the same time, three different particles will be formed according to different particle positions:

- The \( g_{best} \) position of the particle is found in the current population. At this time, the two red real vectors \( c_1(g_{best}^t_i - x^t_i) \) and \( c_2(p_{best}^t_i - x^t_i) \) in Figure 1 are equal to \( \vec{0} \), and the particle’s next-step exploration space is reduced to a point, as shown in Figure 2.a. The particle only reaches the next position according to the inertia. When the algorithm fails to find a better
position than the current position within a period of time, the particle velocity will gradually change to 0, and the particle will be at a standstill near the position, and the algorithm will converge rapidly.

Figure 2. Next-step Exploration Space of Particles at the \( p_{best} \) and \( g_{best} \) Positions
- The particle is at the \( p_{best} \) position. In this case, the particle next-step exploration space \( c_2(p_{best} - x_i) = 0 \) will become a line segment as shown in Figure 2.b, and the particle will quickly approach \( g_{best} \).
- The particle is neither at the \( p_{best} \) position nor in the \( g_{best} \) position. The next-step exploration space of the particle is shown in Figure 1. The particle searches the area composed of individual optimization and global optimization.

Three parameters are used to adjust the shape, size and position of the particle’s next-step exploration space, where \( c_1 \) and \( c_2 \) control the size of the particle’s next-step exploration space. When \( c_1 = 0 \) and \( c_2 = 0 \), the algorithm does not use the guidance information of individual optimization and global optimization. The algorithm will conduct random search according to the speed of random initialization, and the performance is poor. With the increase of \( c_1 \) and \( c_2 \), the search ability of the algorithm increases gradually. But when the two are too large and the next-step search space of particles is too large, it will also make the algorithm gradually lose the use of individual-optimized and global-optimized guidance information, and the effect of the algorithm will decrease. The algorithm can achieve better results through using reasonable parameters to control the size of particle next-step exploration space. On the other hand, \( \omega \) controls the offset of the next-step exploration area of the particle to the current velocity direction, and balances the global exploration and local exploration of the algorithm. When \( \omega = 0 \), the particle’s next-step exploration space is not offset. In this case, the particles only explore in the vicinity of \( p_{best} \) and \( g_{best} \), which with good local exploration ability but lack of global exploration ability. With the gradual increase of \( \omega \), the next-step exploration space of the particle shifts beyond the region where the current optimal solution is located, and the exploration of other regions is increased. The global exploration ability of the algorithm is enhanced and the local exploration ability is reduced. When \( \omega \) is too large, the deviation will be too large and the use of the guidance information given by \( p_{best} \) and \( g_{best} \) will gradually be undermined which will degrade the performance of the algorithm.
Figure 3. Influence of Maximum Speed $V_{\text{max}}$ on PSO Algorithm

In fact, the parameter value of a single extreme will lead to extreme conditions in the next-step exploration space of particles, thus resulting in a serious decline in the performance of the algorithm. Three parameters ($c_1$, $c_2$ and $\omega$) are required to make the algorithm achieve better results. The maximum particle velocity $V_{\text{max}}$ works together to produce the appropriate particle next-step exploration space (as shown in Figure 3, it is a way for the maximum velocity to further adjust the particle next-step exploration space, limiting the original space to the shadow part in Figure 3.) Many researchers have studied the parameter value range and convergence of the general PSO algorithm, among which the works of Poli[46][47] is representative, in which the reasonable value region of the three control parameters, $\omega$, $c_1$ and $c_2$, was derived under the stagnation assumption and represent as Equation (3):

$$c_1 + c_2 < \frac{24(1-\omega^2)}{7-5\omega};$$

(3)

All of these works focus on the parameter analysis, actually, the directly affect factors of the performance of PSO algorithm is the particle next-step search space. Parameters $\omega$, $c_1$, $c_2$ and $V_{\text{max}}$ indirectly affect the algorithm by adjusting the next-step exploration space. The methods that can change the size, shape and position of the next-step exploration space of particles in the algorithm don’t have to be or are not limited to the three parameters, and other methods can be used to modify the next-step exploration space, as shown in Figure 4, which is an adjustment method.
Therefore, when designing and using the PSO algorithm to solve the optimization problems, we can design a PSO algorithm that better meets the needs of the problems from the perspective of how to better design and control the next-step exploration space of particles.

The basic flow of the PSO algorithm is shown in Figure 5:

![Figure 5. Basic Flow of PSO Algorithm](image)

At the beginning of the algorithm, the population is initialized, including the particle position, velocity, \( p_{best} \), \( g_{best} \) and the calculation of the corresponding fitness value of the particles. Then the iterative solution process of the algorithm initiated. In this process, \( p_{best} \) and \( g_{best} \) are firstly updated according to the particle fitness, and then the next position and speed of the particles are also updated based on the \( p_{best} \) and \( g_{best} \). In this process, mutation operation can be added to increase the randomness of the algorithm, reduce the probability of premature convergence of the algorithm. Finally, the algorithm stops when the termination condition is reached, where the termination condition can be a certain number of iterations or the optimal value condition.

To sum up, the PSO algorithm is a population-based search algorithm. Its core idea is to update the particle position and velocity by using the Guidance information get from \( p_{best} \) and \( g_{best} \), and the
The essence of the algorithm is how to effectively explore the search space by controlling the size, shape, and position of the particle’s next-step search space. Too large space will make $p_{best}$ and $g_{best}$ lose their guiding function, and too small space will also cause insufficient use of guiding information. Similarly, the shape and position of space will affect the search effect of the algorithm. When using the PSO algorithm to solve the optimization-related problem, we can get better ideas of algorithm design from the perspective of how to design and control particles’ next-step search space.

3. Analysis on Solving the Discrete Optimization Problems by PSO

The general PSO algorithm is designed to solve the continuous optimization problems. It is quite necessary to analyze the differences between the use of PSO in continuous problems and in discrete problems when apply it to discrete case, but there is no related work. To address this challenge, we mainly analyzes the background knowledge, which we called them default premise, of the PSO algorithm which are easy to ignore by researchers and the changes and impacts when PSO algorithm is applied to the field of discrete optimization are also take into consideration.

3.1. Premised Background Knowledge of PSO Continuous Optimization

The PSO algorithm was originally designed to solve the continuous optimization problems, so some default premises are related to these problems. When the algorithm is applied to solving the discrete optimization problems, these default premises will change instead of in the default form. The default premises that have a key impact on the PSO algorithm mainly include the following two premises:

**Default premise 1** General PSO algorithm measures the distance between solutions in solution space (the space of the particles) based on Euclidean distance.

The problem considered by the general PSO algorithm is the search problem of continuous space. In this case, the solution space is continuous space, and the distance between solutions is measured by Euclidean distance by default. In discrete optimization, the solutions in the solution space are represented by integer variables. If there is no continuity constraint between each solution (that is, every point in the solution space), the distance between solutions can be expressed in many ways, and different distance measurement methods will correspond to different structures of the solution space.

| Solutions in the solution space | Euclidean distance | Hamming distance |
|---------------------------------|--------------------|------------------|
| (1,2,3,4) - (1,2,3,10)          | 6                  | 1                |
| (1,2,3,4) - (2,1,4,3)           | 2                  | 4                |

As shown in Table 1, for the distance relationship between the same two pairs of solutions, when different distance metrics are used, the distance relationship is completely different, and different distance metric relationships will produce different solution space structures. Different spatial structures have different effects when using PSO algorithm.

**Default premise 2**: During the continuous optimization, the fitness values of the points in the solution space changes regularly.

The regular changes mean that the fitness value of points in space changes with a small and reasonable range in the adjacent region. As shown in Figure 6, both figures are composed of 1600 data points in two-dimensional space, and the height coordinate value is the fitness value of the midpoint in the space. The left figure is generated by the equation $f = \cos(x) * \cos(y) + \frac{1}{\sqrt{x^2+y^2+0.1}}$ according to the spacing of 0.5 between -10 and 10, and the values of each point in the right figure are randomly generated data between 0 and 10. When we talk about the fitness value of the space point change regularly, it means that the fitness values of the points in a small area are change mildly. As shown in the left figure, although there are many extreme points in the figure, the value of the whole space point changes regularly. In this case, even if the fitness value of the solution space point changes greatly and there are many extreme points, the best solution is around the suboptimal solution and there will be no best solution around the worse solution. In this case, the PSO algorithm can find the optimal solution (at least the local optimal solution) through iteration. Because the core idea of the PSO algorithm is to
randomly explore around the best solution ($p_{best}$ and $g_{best}$) through the guidance of finding the best solution at present, in order to find a better solution, and the premise is that the fitness value at the midpoint of the space changes regularly, and that only can the algorithm be found when the best solution is around the suboptimal solution. When the fitness values of points in the space change irregularly, the extreme irregularity is shown in the right figure where the fitness values of all points are randomly generated values, the guidance information of the PSO algorithm when searching in this space is completely useless and the search of the algorithm is pure random search.

![Figure 6. Schematic Diagram of Regular Changes and Irregular Changes](image)

In the general PSO algorithm, since the solution space changes continuously, the fitness value in the space also changes continuously. In this case, the fitness value of the solution in the adjacent region changes continuously, so it can be considered as being regular. In discrete optimization, the space is discrete, and the fitness values of the points in the space are determined by the specific problems involved. At the same time, the distance measurement relationship between the points in the space is uncertain, so the change of the fitness value in the space is also uncertain.

To sum up, for the above two default premises, default premise 2 is the guarantee that the algorithm can search effectively, while default premise 1 will affect default premise 2. These two points are no longer the default in discrete optimization problems, which will be analyzed in the next section.

### 3.2. Changes in Discrete Optimization by PSO

As mentioned above, some default premises for the application of the general PSO algorithm are analyzed. When these default premises are applied to discrete cases, they are no longer the default cases, which will change and affect the performance of the algorithm. It mainly includes two aspects. First, the discrete domain distance measurement is no longer limited to the Euclidean distance, which can be Hamming distance, probability based or other methods. Second, solution spaces with different structures can be built by using different distance measurement methods, while different structures of solution space have different rules of fitness value change, and the rules of fitness value change will determine the performance of algorithm search. The distance measurement more in line with the problem itself can make the change of fitness value in space more regular. In fact, no optimization algorithm can achieve good results in all problems, and only the algorithm that meets the needs of specific problems can achieve good results in this matter. David H. Wolpert and William G. MacReady[5] [33] put forward the “no-free-lunch theory”, which proved that the average performance of all algorithms in all optimization problems is the same in the search or optimization problem with finite space. When the algorithm have a superior performance on this problem, it will definitely get the inferior results in face of some other optimization or search problems. The problem of discrete optimization problem is a search optimization-related problem in finite space. When the PSO algorithm is applied to discrete optimization problems, it is necessary to reconsider the distance measurement relationship between solutions in search space, and design a discrete PSO algorithm by adopting the distance measurement method that makes the change of fitness value in solution space
more regular, contributing to more effective search.

Discrete PSO algorithm has been widely studied and a variety of discrete forms have been proposed. Some of them are still based on the original Euclidean distance, while others use other measurement methods, but they are designed from the perspective of the problems to be solved without considering the factor of distance measurement. There are few summary studies on the existing discrete PSO algorithms. At the same time, the existing studies are only the lists of many algorithms [45] and do not use a certain standard for classification. In the next chapter, we will classify and summarize the discrete PSO algorithms from the perspective of the distance measurement between solutions in the search space used by the algorithm, and also make a brief analysis.

4. Literature Review Related to Discrete PSO

The features of PSO algorithm and its changes in the discrete optimization field were introduced in the previous section. It was then deemed through the analysis that the key to the effect of PSO algorithm was the next-step search space of particles, which was inseparable from the measurement of distance relation between spatial points. In the previous analysis, we got that the distance relation between solutions in the solution space of this algorithm was a key influencing factor. When a continuous optimization problem is solved using the PSO algorithm, Euclidean distance is generally used by default. However, there are diversified distance measurement methods with varied effects when the application of PSO algorithm is extended to the discrete optimization field. In this chapter, the discrete PSO algorithms in the existing literatures were simply summarized and classified according to the distance measurement methods used.

4.1. Euclidean Distance

The simplest and most direct method of measuring the relation between solutions in a solution space is Euclidean distance, in Euclidean distance-based PSO algorithms, there is only one additional process, the particle rounding, in the iteration process in comparison with the ordinary PSO algorithm. During the whole calculation process, discrete variables or integer variables are regarded as continuous variables for the operation, and the generated non-integral variables are finally rounded by some methods, including round-off [14] [15] [25], ranking and modular reduction [16] [17] [29] [32] decimal removal and modular reduction [18], penalty function [20] [22] and rounding through space transformation technology [26], among which round-off means transforming a non-integral variable value into the nearest integer variable after the particle migration each time. As for the ranking and modular reduction, the particle variables are ranked according to the dimensionality, and the ranking value is used to replace the original non-integral value, which is then controlled within a required range by means of modular reduction. When it comes to the decimal removal and modular reduction, the original non-integral value is multiplied by \(10^k\), the decimals are abandoned, and the value range is controlled in way of modular reduction, thus completing the rounding process, as shown in Equations (4)-(6):

\[
y_{i}^{t+1} = \text{rounding}((x_{i}^{t} + v_{i}^{t+1}) \times 10^k), \quad (4)
\]

\[
p_{i}^{t+1} = y_{i}^{t+1} \mod M, \quad (5)
\]

\[
x_{i}^{t+1} = \begin{cases} p_{i}^{t+1}, & p_{i}^{t+1} > 0 \\ M, & \text{otherwise} \end{cases} \quad (6)
\]

As for the penalty function method, the non-integral variable generated after the particle migration is transformed into an integer variable value. For instance, S. Kitayama et al. [21] realized rounding by using the penalty function as seen in Equation (7):

\[
\Phi(x) = \sum_{i=1}^{n} \frac{1}{2} \sin^{2}(x_{m+1}^{-0.25(d_{i+1} + 3d_{i}))} + 1 \quad (7)
\]

The space transformation technology is used to transform the variables in a continuous space into the variables in a discrete space by some means, which meet the needs of discrete problem, as mentioned in the literature [26]. In that study, the variables \(X = (x_1, x_2, ..., x_n)\) in a continuous space were transformed into the variables \(\pi = (\pi_1, \pi_2, ..., \pi_n)\) in a discrete space by means of Great Value
priority (GVP), and the transformation rules are expressed as seen in Equations (8) and (9).

\[ k = 1 + \sum_{j=1}^{n} \mathbbm{1}(f(x_j > x_i)) \text{, } \text{else} \quad 0 \]

(8)

\[ \pi_k = i \]

(9)

Besides the rounding methods mentioned in the above literatures, there are also many other methods, and the specific rounding methods have been given by different researchers to the problems involved. All the rounding methods are diversified with different application effects, but the particle migration is always based on the search of Euclidean distance. This particle migration mode can be easily implemented, not needing to change the original ordinary PSO algorithm too much, but instead, only the rounding operation is needed after the particle migration each time. However, this method also has very apparent defects. For the most discrete problems, due to the complexity of problems, the fitness value of solution space established based on Euclidean distance usually changes irregularly, and the effect achieved by the PSO algorithm is also poor under this circumstance.

4.2. Hamming Distance

Hamming distance refers to the number of different characters at the corresponding positions of two character strings or vectors with the same length. As Hamming distance is a discrete distance measurement method, the discrete PSO algorithm based on Hamming distance is naturally of discreteness, without any need of rounding. A typical PSO algorithm that realizes the search in a space based on Hamming distance is the binary PSO algorithm (BPSO). In this algorithm, the value of each particle dimension is expressed by 0 or 1, and the distance between solutions in the solution space is measured using Hamming distance. BPSO algorithm is a discrete version of PSO algorithm proposed by the founders of PSO algorithm, Kennedy and Eberhart, in 1997[19], and its basic calculation process is as shown in Equations (10) and (11):

\[ V_{t+1}^{i,d} = \omega V_{t}^{i,d} + c_1 * r_1 * (P_{best}^{t,i,d} - V_{t}^{i,d}) + c_2 * r_2 * (G_{best}^{t,i,d} - V_{t}^{i,d}), \]

(10)

\[ X_{t+1}^{i,d} = \begin{cases} 1, & \text{if } r_3 \leq \text{sigmoid}(V_{t+1}^{i,d}) \\ 0, & \text{otherwise} \end{cases} \]

(11)

Where \( V_{t}^{i,d} \) and \( X_{t}^{i,d} \) represent the values of \( d \)-dimensional variables in the position and velocity of the \( i \) (th) particle in the \( t \) (th) iteration, respectively. It can be seen from the equations that the velocity in the BPSO algorithm is the threshold for each dimension of particle velocity variable to be transformed into 0 or 1 according to \( p_{best} \) and \( g_{best} \), the value of each particle dimension is either the same as the values of \( p_{best} \) and \( g_{best} \) or not, so the distance between solutions is measured through Hamming distance. As a discrete PSO algorithm, the BPSO algorithm has been widely applied and explored a lot by researchers, including: study on the change threshold between 0 and 1 [23]; modification of next search space for particles by adding new guidance information[36] [37]; control of particle change rate by combining the probability[38]. The BPSO algorithm is capable of achieving a good effect when handling the binary coding problem, which is its main advantage. However, as the binary coding is required by this algorithm, the scope of its application is greatly restricted. In addition, the ternary PSO algorithm [39] proposed by Moradi and Fotuhi-Firuzabad is also a PSO algorithm based on Hamming distance. This algorithm can also be regarded as a variant of BPSO algorithm, in which each particle dimension has three states in the coding process, and the particle migration is implemented through the transformation of each dimension among the three states, so as to realize the search.

The discrete PSO algorithm based on Hamming distance has been scarcely studied, and there are no other related methods except BPSO at present. But actually, Hamming distance accords with the features of many discrete problems, such as deployment problem, task allocation problem and knapsack problem. Hence, the related discrete PSO algorithm can be designed according to Hamming distance in the future.

4.3. Ways combined with the probability or fuzzy theory

The combination with the probability or fuzzy theory is also an important strategy of applying the PSO
algorithm to the discrete optimization field. The task allocation problem, travelling salesman problem, etc. were solved by combining the PSO algorithm and the fuzzy theory [40] -[42] or mixing PSO algorithm with probability [13] [30] [31]. Relative to the mobile search mode in a space, this mode is no longer the search based on the movement in a space based on Euclidean distance or Hamming distance. The core idea of discrete PSO algorithm combining the probability or fuzzy theory is still to use individual optimum and population optimum as the guidance information, but its essence no longer lies in constructing an appropriate the next-step search space of particle. As for the discrete PSO algorithm based on the probability or fuzzy theory, the next search step of particles will be guided by a variable related to the probability or fuzziness, which is also adjusted by \( p_{best} \) and \( g_{best} \). In this way, the probability or membership grade substitutes the next search space of articles, turning the original search process based on movement in the space into a selective search process. Movement, which is a continuous process, will generate a non-integral variable, but selection, which is a discrete process, will not, so as to realize the application of PSO algorithm to the discrete optimization field. This method is featured by stable effect, favorable universality, and high applicability to the most discrete optimization problems, with satisfying performance. Nevertheless, it mainly has two defects: It fails to achieve a particularly good effect, which is stable though, and this coincides with the “No Free Lunch theory” proposed by David H. Wolpert and William G. Macready. The algorithm can harvest identical average effect among all problems. Second, relative to the ordinary PSO algorithm, the probability or membership variable will substantially aggravate the complexity of calculation, especially the space complexity, thus imposing additional burden to the search process.

4.4. Other Distance Measurements Designed for Specific Problems

Besides the aforementioned distance measurement methods, there are also many other distance measurements designed by researchers for specific problems. As these measurement methods meet the needs of specific problems, they are capable of obtaining favorable performances, but the scope of their application is still very restricted due to their problem-specific nature. For instance, Clerc [24] and Wang et al.[28] put forward that basic exchange sequence was used as the distance between solutions to solve the travelling salesman problem (TSP). D. Y. Sha and C. Hsu [27] also used the exchange sequence as the foundation for particle migration to solve the job shop scheduling (JSP) problem. Both the abovementioned problems are actually a ranking process of multiple elements, while the basic exchange sequence is just the value exchange between two positions in a ranking. Different from Euclidean distance, Hamming distance and probability-based method, this distance measurement method will not generate any unreasonable solution (the same element is ranked repeatedly), which relatively conforms to the distance relation between the solutions in the ranking problem, so it has achieved a good effect when handling the two problems. Actually, this method is also able to perform very well in the PSO algorithm-based optimization of ranking problem, but meanwhile, its scope of application is also restricted to such problem.

4.5. Ways combined with the other algorithms

The ways combined with the other algorithms means combining the core idea (using the currently searched good solution as the guidance information to guide the next search process of particles) with the other algorithms, like local search [34] [35] and genetic algorithm (GA) [43] [44], which mainly follow the core idea of PSO algorithm. When the other algorithms are combined to solve the particle migration and search, this can save the discretization step, which is required if the PSO algorithm is directly used. Although the algorithm design, which should be implemented by combining the other algorithms, is usually complicated under this combination mode, it is adaptable to solve many discrete optimization problems while giving full play to the core idea of PSO algorithm. Moreover, this method has gradually become a hotspot research method regarding the discrete optimization problem based on the PSO algorithm.
4.6. Summary of the Chapter
In this chapter, the related algorithms were analyzed, summarized and classified from the perspective of distance measurement methods adopted by the discrete PSO algorithm in a solution space, including the distance measurement method based on Euclidean distance, that based on Hamming distance, that combining the probability or fuzzy theory, distance measurement method designed for specific problems and that combining the other algorithms. Each method has their respective characteristics and scope of application. Hence, the features of different problems should be considered when studying the discrete optimization problem and designing a new discrete PSO algorithm to solve the discrete optimization problem. Besides, the algorithm design should be carried out using a proper pertinent distance measurement method. There is no perfect discrete PSO algorithm that can solve all discrete optimization problems, but instead, it should be designed according to specific problems.

5. Conclusion and Expectation
In this study, the ordinary PSO algorithms were analyzed from the perspective of next particle search space. It is deemed that the factor directly influencing the algorithm effect is the next-step search space of particles, while the algorithm parameters just indirectly affect the algorithm performance by adjusting the search space. Therefore, the PSO algorithm should be analyzed by taking the next-step search space as the cut-in point but not directly from the perspective of algorithm parameters. Furthermore, two default premises of ordinary PSO algorithm, the default measurement in the solution space based on Euclidean distance and fitness value of each solution in the solution space is change regularly are presented. Both the two aspects may change in the discrete optimization problems, so they are no longer the default preconditions, and will affect the performance of PSO algorithms when solving discrete optimization problems. Moreover, it is obtained that the key to the PSO algorithm lies in the selection of distance measurement when applied to the discrete optimization, because under a deterministic problem to be solved by the algorithm, the distance measurement method chosen by the algorithm is the main factor influencing the change laws of fitness value of each solution in the solution space, and whether the algorithm search is effective depends on whether the change is regular. Finally, the existing discrete PSO algorithms are classified and summarized according to the distance measurement method.

The following two directions are suggested for the future research:
* The influence the next-step search space of particle on the ordinary PSO algorithm can be explored and analyzed, but not from the perspective of three classical parameters, because the parametric influence on the algorithm is indirect and replaceable and the next-step search space of particle is the vital and direct influencing factor.
* The discrete PSO algorithm may be designed to solve the discrete optimization problem from the perspective of distance relation between solutions in the search space.

References
[1] Kennedy, J., & Eberhart, R. (1995). Particle Swarm Optimization. Icnn95-international Conference on Neural Networks. IEEE.
[2] Shi, Y. (1998). A Modified Particle Swarm Optimizer. Proc of IEEE Icec Conference.
[3] Cleghorn, C. W., & Engelbrecht, A. P. (2014). Particle swarm convergence: an empirical investigation. IEEE.
[4] Shi, Y., & Eberhart, R. C. (1998). Parameter selection in particle swarm optimization. International Conference on Evolutionary Programming. Springer, Berlin, Heidelberg.
[5] Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. IEEE Transactions on Evolutionary Computation, 1(1), 67-82.
[6] F. van den Bergh, A.P. Engelbrecht,(2006). A study of particle swarm optimization particle trajectories. Information Sciences, 176(8), 937-971.
[7] Wakasa, Y., Tanaka, K., & Nishimura, Y. (2010). Control-theoretic analysis of exploitation
and exploration of the PSO algorithm. IEEE International Symposium on Computer-aided Control System Design. IEEE.

[8] Zheng, Y. L., Ma, L. H., Zhang, L. Y., & Qian, J. X. (2003). On the convergence analysis and parameter selection in particle swarm optimization. Machine Learning and Cybernetics, 2003 International Conference on. IEEE.

[9] Erskine, A., Joyce, T., & Herrmann, J. M. (2016). Parameter selection in particle swarm optimisation from stochastic stability analysis. Springer International Publishing.

[10] Pan, F., Zhang, Q., Liu, J., Li, W., & Gao, Q. (2014). Consensus analysis for a class of stochastic pso algorithm. Applied Soft Computing, 23(Complete), 567-578.

[11] Wakasa, Y., Tanaka, K., & Nishimura, Y. (2010). Control-theoretic analysis of exploitation and exploration of the PSO algorithm. IEEE International Symposium on Computer-aided Control System Design. IEEE.

[12] Bisoy, S. K., & Pradhan, A. (2020). A novel load balancing technique for cloud computing platform based on pso. Journal of King Saud University - Computer and Information Sciences.

[13] Chen, W. N., et al. (2010). A novel set-based particle swarm optimization method for discrete optimization problems. IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, 14(2), 278-300.

[14] C. Jianfang, C. Junjie, Z. Qingshan, (2014). An optimized scheduling algorithm on a cloud workflow using a discrete particle swarm. Cybernetics & Information Technologies, 14(1).

[15] W.-C. Wu, M.-S. Tsai, (2011). Application of enhanced integer coded particle swarm optimization for distribution system feeder reconfiguration, IEEE Trans.Power Syst. 26 (3) 1591–1599.

[16] Alkayal, E. S., Jennings, N. R., & Abulkhair, M. F. (2017). Efficient Task Scheduling Multi-Objective Particle Swarm Optimization in Cloud Computing. Local Computer Networks Workshops. IEEE.

[17] Ramezani, F., Lu, J. & Hussain, F. K. (2014), Task-Based System Load Balancing in Cloud Computing Using Particle Swarm Optimization, Int J Parallel Prog 42, 739–754.

[18] Beegom, A.S.A., Rajasree, M. S. (2019), Integer-PSO: "a discrete PSO algorithm for task scheduling in cloud computing systems," Evol. Intel. 12, 227–239.

[19] Kennedy, J., & Eberhart, R. C. (1997). A discrete binary version of the particle swarm algorithm. 1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation. IEEE.

[20] Li D, Wang B, KitaY ama S, Y amazaki K, Arakawa M (2005) Application of particle swarm optimization tohe mixed discrete non-linear problems. In: Artificial intelligence applications and innovations, vol 187.IFIP—The International Federation for Information Processing. Springer, USA, pp 315–324. doi:10.1007/0-387-29295-0_34

[21] Kitayama S, Arakawa M, Y amazaki K (2006) Penalty function approach for the mixed discrete nonlinear problems by particle swarm optimization. Struct Multidiscip Optim 32(3):191–202

[22] Kitayama S, Y asuda K (2006) A method for mixed integer programming problems by particle swarm optimizat-ion. Electr Eng Jpn 157(2):40–49.

[23] Tan, B., Hui, M., Yi, M., & Zhang, M. (2021). Evolutionary multi-objective optimization for web service location allocation problem. IEEE Transactions on Services Computing, PP(99), 1-1.

[24] M. Clerc and J. Kennedy, (2002). The particle swarm - explosion, stability, and convergence in a multidimensional complex space. IEEE Transactions on Evolutionary Computation, 6(1), 58-73.

[25] Salman, A., Ahmad, I., & Al-Madani, S. (2002). Particle swarm optimization for task assignment problem. Microprocessors and Microsystems, 26(8), 363-371.

[26] Wei, P., Wang, K. P., Zhou, C. G., Dong, L. J., & Wang, J. Y. (2004). Modified particle swarm optimization based on space transformation for solving traveling salesman problem. Machine Learning and Cybernetics, 2004. Proceedings of 2004 International
Conference on. IEEE.

[27] Sha, D. Y. , & Hsu, C. Y. . (2006). A hybrid particle swarm optimization for job shop scheduling problem. Computers & Industrial Engineering, 51(4), 791-808.

[28] K.-P . Wang, L. Huang, C.-G. Zhou, and W. Pang. (2003). Particle swarm optimization for traveling salesman problem, in Proc. 2nd Int. Conf.Mach. Learning Cybern., pp. 1583–1585.

[29] Rui, Z. , S. Song , and W. Cheng. (2012) A two-stage hybrid particle swarm optimization algorithm for the stochastic job shop scheduling problem. Knowledge-Based Systems 27:393-406.

[30] Xz, A , A. Zf , and B. Sx . (2020) A discrete particle swarm optimization method for assignment of supermarket resources to urban residential communities under the situation of epidemic control – Science Direct. Applied Soft Computing 98.

[31] Zhang, M. , et al. (2021). A discrete PSO-based static load balancing algorithm for distributed simulations in a cloud environment, Futur Generation Computer Systems 115:497-516.

[32] A, M. F. T. , B, Y. C. L. , C, M. S. , & D, G. G. . (2007). A particle swarm optimization algorithm for makespan and total flowtime minimization in the permutation flowshop sequencing problem - sciencedirect. European Journal of Operational Research, 177(3), 1930-1947.

[33] David, H. , & William, G. . (1995). No free lunch theorems for search. Working Papers, 122(1431), 431-434.

[34] Consoli S., Pé rez J.A.M., Darby-Dowman K., Mladenović N. (2008) "Discrete Particle Swarm Optimization for the Minimum Labelling Steiner Tree Problem," In: Krasnogor N., Nicosia G., Pavone M., Pelta D. (eds) Nature Inspired Cooperative Strategies for Optimization (NICO S 2007). Studies in Computational Intelligence, vol 129. Springer, Berlin, Heidelberg.

[35] Chitra S., Madhusudhanan B., Sakthidharan G.R., Saravanan P. (2014). "Local Minima Jump PSO for Workflow Scheduling in Cloud Computing Environments," In: Jeong H., S. Obaidat M., Park J. (eds) Advances in Computer Science and its Applications. Lecture Notes in Electrical Engineering, vol 279. Springer, Berlin, Heidelberg.

[36] Baa, B . , Bgk, A . , Na, C . , D, A. C. , & Bt, E . . (2020). Application of binary pso for public cloud resources allocation system of video on demand (vod) services. Applied Soft Computing.

[37] Deligkaris KV, Zaharis ZD, Kampitaki DG, Goudos SK, Rekanos IT, Spasos MN (2009) Thinned planar array design using Boolean PSO with velocity mutation. IEEE Trans Magn 45(3):1490–1493

[38] Wang J (2007) A novel discrete particle swarm optimization based on estimation of distribution. Adv Intell Comput Theories Appl Aspects Artif Intell, 791–802.

[39] Moradi A, Fotuhi-Firuzabad M (2008) Optimal switch placement in distribution systems using trinary particle swarm optimization algorithm. IEEE Trans Power Deliv 23(1):271–279.

[40] Ling, S. H. , Jiang, F. , Nguyen, H. T. , & Chan, K. Y. . (2011). Permutation flow shop scheduling: Fuzzy particle swarm optimization approach. IEEE International Conference on Fuzzy Systems. IEEE.

[41] Pang, W. , Wang, K. P. , Zhou, C. G. , Huang, L. , & Xiao-Hui, J. I. . (2005). Fuzzy discrete particle swarm optimization for solving travel salesman problem. Mini-micro Systems.

[42] NS Niasar, J Shanbezade, MM Perdam, & M Mohajeri. (2009). Discrete Fuzzy Particle Swarm Optimization for Solving Traveling Salesman Problem. International Conference on Information & Financial Engineering. IEEE.

[43] Jyoti Sharma, Ravi Shankar Singhal, (2014). Genetic algorithm and hybrid genetic algorithm for space allocation problems - a review. International Journal of Computer Applications, 95(4), 33-37.

[44] Sharma, J. , & Singhal, R. S. . (2015). Comparative research on genetic algorithm, particle
swarm optimization and hybrid GA-PSO. 2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom). IEEE.

[45] Ahmad, et al. (2015), Particle swarm optimisation for discrete optimisation problems: a review. Artificial Intelligence Review.

[46] R. Poli and D. Broomhead, (2007). Exact analysis of the sampling distribution for the canonical particle swarm optimiser and its convergence during stagnation. Genetic and Evolutionary Computation Conference. New York, NY, USA: ACM Press, pp. 134–141.

[47] Poli, R. (2009). Mean and variance of the sampling distribution of particle swarm optimizers during stagnation. IEEE Transactions on Evolutionary Computation, 13(4), 712-721.