*Advanced Engineering Forum* *Online: 2012-09-26*
*ISSN: 2234-991X, Vols. 6-7, pp 736-741*
*doi:10.4028/www.scientific.net/AEF.6-7.736*


# Research on Timetabling Problems Based on Particle Swarm Optimization Algorithm

Xinmin Ma[1, a], Linli Wu[2,b]

[1]Basic Courses Department, Bengbu Automobile Non-Commissioned Officer Academy, Bengbu, 233011 China

[2]NO 4 Department, Bengbu Navy Petty Officer Academy, Bengbu 233012, China

[a]superwt@qq.com, [b]4085011@qq.com

**Keywords:** particle swarm optimization algorithm; timetabling; fitness function

**Abstract.** A new algorithm for timetabling based on particle swarm optimization algorithm was proposed, and the key problems such as particle coding, fitness function fabricating, particle swarm initialization and crossover operation were settled. The fitness value declines when the evolution generation increases. The results showed that it was a good solution for course timetabling problem in the educational system.

## Introduction

Timetabling is a very important and practically complicated job in the education management. The main function for timetabling is to collect the courses application of such education departments as schools and departments and make a timetable for different classes in the whole university according to the teaching plan and teaching processes, the aim of which is to carry out the teaching activities by plan and schedule. The timetabling system shall guarantee that teachers, students and classrooms mustn't clash with each other in timetabling and the restrictive conditions such as teachers' demands and classrooms' establishment shall be satisfied.

Timetabling may be viewed as a restrictive optimization problem. The restrictive conditions are that there is no clash among teachers, classrooms and time in the set timetabling and the optimization aim is to utilize the least teachers, classrooms and time in the premise of education aim's satisfaction. Several algorithms such as genetic algorithm, divide-and-conquer algorithm and greedy algorithm are adopted in timetabling. However, those algorithms have a slow solving speed and easily get a local minimum. Particle swarm optimization algorithm is an evolution algorithm based on swarm intelligence, which was put forward by James Kennedy and Russ Eberhart with the enlightenment of bird swarm's preying in 1995. Compared with genetic algorithm, particle swarm optimization algorithm has a simple concept and has much less parameter to regulate, is easy to realize. It has been widely adopted in the fields of function optimization, patter classification and neural network training, etc., while it is rarely used in timetabling system. The paper puts forward a new timetabling algorithm based on particle swarm optimization algorithm to solve the timetabling problems in educational system.

## Particle Swarm Optimization Algorithm

Particle swarm optimization algorithm simulates the bird swarm's preying activity in which the birds cooperate to get their goal. In PSO system, every candidate solution is called one particle. Several particles cooperate to achieve optimization. Every particle, with its own experience and the contiguous particle swarm's most advantageous experience, directs itself to a better position in the problem space and searches the best solution. Particle swarm optimization algorithm may be described as follows [3]:

Supposing the whole particle swarm involves N particles and every particle's dimensionality is D. Num. i particle's position is described as Xi=(xi1,xi2,…,xid), and the velocity of Num. i particle's position movement is described as Vi=(vi1,vi2,…,vid). Presently, the best position searched by Num.

i particle is defined as pBest=(pi1,pi2,…,pid), the best position searched by the whole particle swarm is defined as gBest=(gi1,gi2,…,gid). Every particle's position changes according to the following equations:

$$V_{id}(t+1)=w\times v_{id}+c_1\times rand(\ )\times(p_{id}(t)-x_{id}(t))+c_2\times rand(\ )\times(g_d(t)-x_{id}(t)) \quad (1)$$

$$x_{id}(t+1)=x_{id}(t)+v_{id}(t+1) \quad (2)$$

$c_1$ and $c_2$ are two positive constants, rand( ) is random variables, w is inertia weight, $p_{id}$ represents the currently searched best position of Num. i particle, and $g_d$ represents the currently searched best position of the whole particle swarm. Every particle's initial position and velocity come out randomly and vary according to Eq. 1～2 until the best solution comes out.

The timetabling methods based on particle swarm optimization algorithm may be described as follows:

a) Randomly initialize every particle and its velocity in the particle swarm.

b) Calculate every particle's fitness function and the currently searched best position $p_{Best}$ of the particle.

c) Select the particle, which has the best fitness function in the particle swarm, as $g_{Best}$.

d) Calculate every particle's velocity according to Eq. 1.

e) Every particle evolves to get a middle generation particle swarm $M_{id}$ according to Eq. 2. The next generation particle swarm makes a choice between the middle generation particle swarm and the original generation particle swarm according to the following three conditions. If one meets the restrictive conditions, the other not, the one that meets the restrictive conditions directly evolves to the next generation. If both of them meet the restrictive conditions, the one that has a better fitness function evolves to the next generation. If both of them do not meet the restrictive conditions, the one that less violates the restrictive conditions evolves to the next generation.

f) Turn to the above b）, until the biggest iterative times and algorithm convergences are realized.

**Timetabling Problems Description**

As to timetabling problems, the key task is to arrange classes, classrooms, courses and teachers in a week without time clash. According to this point, we make the following description: There are R classrooms, T teachers, S courses, C classes and P periods in the school. Classrooms aggregate is $R(R_1,R_2,\ldots,R_n)$, and every classroom can accommodate $(X_1,X_2,\ldots,X_n)$ students. Classes aggregate is $C(C_1,C_2,\ldots,C_n)$ and every class can accommodate $(K_1,K_2,\ldots,Kc)$ students. Courses aggregate is $S(S_1,S_2,\ldots,S_n)$ and every course has $C_i(1\leq C_i<C_n)$ classes and one teacher. Teachers aggregate is $T(T_1,T_2,\ldots,T_n)$ and every teacher teaches $S_m(1\leq S_m<S_n)$ courses. Periods aggregate is $P(P_1,P_2,\ldots,P_n)$, in a week every day has classes, every day has five teaching periods, and every period has two hours, that is, every morning has two teaching periods, every afternoon has two teaching periods and every evening has one teaching period and thus periods aggregate includes thirty five periods. Supposing 11 represents Monday's first teaching period (the first and second hour in Monday), 12 represents Monday's second teaching period (the third and fourth hour in Monday), by analogy, these teaching periods form a periods aggregate P(11,12,…,75).

The proper and logical timetable must meet the following forcibly restrictive conditions: a) In the same teaching period, there cannot be two or more courses for one teacher. b) In the same teaching period, there cannot be two or more courses for one class. c)In the same teaching period, there cannot be two or more courses in one classroom. d)The number accommodated by the classroom must be bigger than that attending the class.

In addition to the above forcibly restrictive conditions, we should consider some flexible restrictive conditions. These flexible restrictive conditions may make the timetable more reasonable. a)The required courses had better be arranged in the morning, the selective courses had better be arranged in the afternoon, and in the evening there had better be no courses. b) If some teachers have specific

demands for time and classrooms, we'd better meet them. c)In order to avoid excessive tiredness, the same teacher's different classes should not be arranged in the same day. d)Time for the students' courses shouldn't be converged together in certain period, and we'd better try to avoid one whole day's classes for students and the other day non. e)One course had better be arranged in different periods. We cannot arrange one course in a day or in two continuous days so that teachers have enough time for making preparations and rectifying schoolwork and students have enough time for review.

Now, the timetabling problems can convert to the problems of seeking a five-dimension (R,C,T,S,P) vector space. The five-dimension vector in the space should meet the above restrictive conditions and there are no clashes between two of the five-dimension vectors. Respectively, the five dimensions are as follows: classroom (R): all the useable classrooms in the school, involving different classrooms' property, for example, big or small classrooms, multimedia or language learning classrooms; class (C): all the teaching classes in the current semester, including classes' property, for example, students number in the class, combined classes or not; teacher (T): all acting teachers of all courses in the school; course (S): all courses in the current semester; period (P): periods for teaching.

## The Key Problems in Realizing Algorithm

**Particle Coding.** Particle coding is the crux of particle swarm algorithm, the quality of which decides the algorithm's function. According to the above analysis, particle coding adopts decimal coding in this algorithm, and its frame is as fig one. For example, the particle whose code is 4314050707070300753851 represents the following timetabling information: Num 07030 teacher will give a lesson of C language on Friday morning's first two hours to Num 050707 class in classroom 314 of building 4.

| classroom | class | teacher | course | time |
|---|---|---|---|---|

Fig. 1 Particle coding sketch map

Every particle represents one possible timetabling result. According to the fitness function the particle will be evaluated.

**Fitness Function.** Fitness function is the basis of particle swarm algorithm choosing the next generation particle swarm in evolution. The particle whose fitness function is smaller is more likely to be involved in the next generation particle swarm. Therefore, the good or bad quality of fitness function decides particle swarm algorithm's convergence velocity and the best solution. In this paper's algorithm, the instance that every particle breaks the restrictive conditions is adopted as the important basis of designing fitness function. The fitness function in this paper is defined as Eq. 3, that is, every particle's fitness function is the weighted sums of the particle breaking restrictive conditions.

$$f = \sum_{i=1}^{k} w_i \times p_i \tag{3}$$

In the equation, $p_i$ represents if the particle breaks the restrictive conditions i. If the particle breaks the restrictive conditions i, $p_i$ is set as 1. If not, $p_i$ is set as 0. Weight $w_i$ represents the importance of Num i restrictive condition. If the particle's fitness function is smaller, it represents the particle breaks less restrictive conditions, the effectiveness of timetabling is better and the particle has a greater survival probability in the evolution of particle swarm.

**Initialization.** The aim of initialization is to provide the initial particle swarm for the following evolution operation. In this paper's algorithm, the corresponding particle will be generated by randomly selecting combination among all the possible classroom aggregate, teacher aggregate, class aggregate, course aggregate and period aggregate of the school. When the particle swarm is initialized, the restrictive conditions should be considered among classrooms, teachers, classes, courses and time, which has been explained in describing timetabling problems.

**Crossover Operation.** Under many circumstances, particle swarm algorithm may get local minimum and bring in premature convergence, which mainly results from population's diversity debasing in the searching space. In evolution algorithm, the crossover operator can increase population's diversity to avoid premature convergence. The algorithm in this paper also adopts crossover operation to overcome the above flaw.

Tow particles are selected from the particle swarm and generate the random number between [0, 1]. If the random number is smaller than the given crossover probability, the two particles will get crossover operation. The elements that represent the class, teacher, and course in the two particles are selected to make a crossover operation; it will get two new particles. The crossover operation is as the following fig two. Though the crossover operation reduces the algorithm's convergence velocity, the population's diversity get increased, which increases the algorithm's global searching ability, and avoids local minimum.

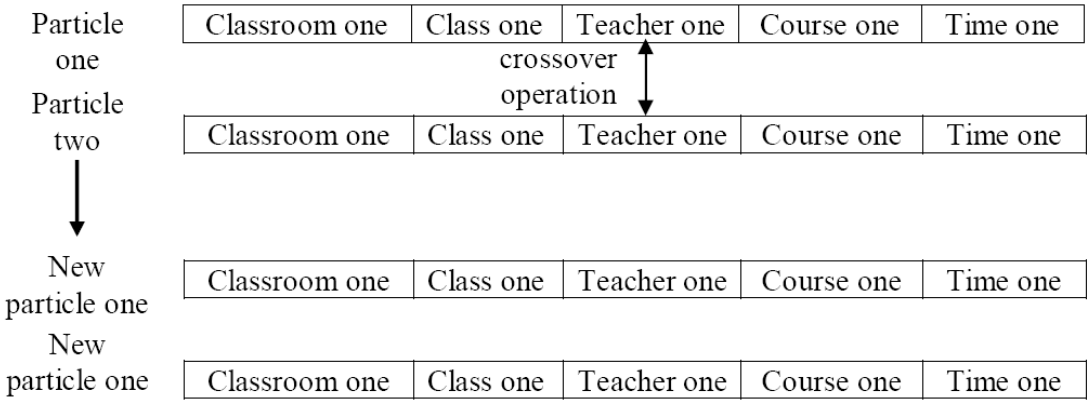


Fig. 2 Crossover operation sketch map

## The Key Problems in Realizing Algorithm

When the above particle swarm algorithm is applied to the timetabling subsystem of the educational administration's information systems, the results indicate as follows: compared with the traditional manual timetabling, timetabling by particle swarm algorithm is much faster, the results is more reasonable. The clashes become less and only a small quantity of manual adjustment work is needed to meet satisfaction.

**Choice of Inertia Weight.**In Eq. 1, the function of the inertia weight w is to keep particles' motion inertia, so that particles are able to continue enlarging searching space and to search new area. If w is bigger, the algorithm will have more ability to make global searching. If w is smaller, the algorithm tends to make local searching. Generally, it is to make w become smaller with iterative times(method one), that is, $w=w_{max}-iter\times(w_{max}-w_{min})/iter_{max}$. $w_{max}$ is the biggest inertia weight, $w_{max}= 0.9$; $w_{min}$ is the smallest weighting coefficient, $w_{min}= 0.4$; iter is the current iterative time, and $iter_{max}$ is the overall iterative time of the algorithm, $iter_{max}=500$.

Table 1 The results of two methods comparison

| Test number | method | Average fitness | Convergence generations |
|---|---|---|---|
| 1 | Method one | 0.4950 | 220 |
| | Method two | 0.4665 | 217 |
| 2 | Method one | 0.4605 | 230 |
| | Method two | 0.4409 | 228 |
| 3 | Method one | 0.5004 | 216 |
| | Method two | 0.4992 | 205 |
| 4 | Method one | 0.4983 | 227 |
| | Method two | 0.4638 | 219 |

There is the other method of calculating inertia weight (method two). In this method, the inertia weight w is given a bigger value to enhance the ability of global searching at the beginning, so that a larger area can be searched and the general position of the best solution can be more quickly located. Then, we gradually decrease the inertia weight w to enhance the ability of local searching, that is, the particle's velocity becomes slow to carefully make local searching. The algorithm in this paper adopts these two methods of changing the inertia weight respectively. Table one is the results of two methods comparison. From table one, we can see the effects are better by adopting the second method.

**Choice of Acceleration Coefficient.** In Eq. 1, the acceleration coefficients $c_1$ and $c_2$ are used to adjust the particle's individual and social experience's function in its motion, which represents the acceleration weight when every particle is pushed to its individual best position and the swarm's best position. If the value is low, the particle is allowed to rove around the object area. If the value is high, the particle will be accelerated to rush at or get across the object area. If $c_1$=0, it represents the particle itself has no cognitive ability and it may arrive at a new searching space with the mutual operation among particles, but it will easily get in local minimum and bring premature convergence. If $c_2$=0, it represents there wasn't information communion among particles and the particle swarm algorithm will become a randomly searching algorithm with many starting points. Generally, the range of $c_1$ and $c_2$ is 0~4. In table two, different combined values of $c_1$ and $c_2$ are adopted to program the path in order to get the best fitness function's average fitness. As to every combined value, algorithm functions thirty times. From the data in the table, we can see $c_1$=1.5 and $c_2$=2.0 are better choices for this paper's problems.

Table 2 Two the influence on the best fitness function of different combination between $c_1$ and $c_2$

| $c_1$ | $c_2$ | | | | |
|---|---|---|---|---|---|
| | 1.0 | 1.5 | 2.0 | 2.5 | 3.0 |
| 1.0 | 0.4986 | 0.4565 | 0.4612 | 0.4335 | 0.4834 |
| 1.5 | 0.4816 | 0.4847 | 0.4293 | 0.4436 | 0.4676 |
| 2.0 | 0.4632 | 0.4713 | 0.4585 | 0.4397 | 0.4469 |
| 2.5 | 0.4865 | 0.4501 | 0.4356 | 0.4636 | 0.4662 |
| 3.0 | 0.4542 | 0.4616 | 0.4402 | 0.4616 | 0.4725 |

**Choice of Crossover Probability.** The choice of crossover probability in algorithm has a great influence on the results. If the crossover probability is too small, it is hard to generate new particle fabric and it will influence population's diversity. If the crossover probability is too big, the algorithm will become simply random searching and it is difficult for particles to keep individual and global experience. Fig. 3 is the comparison of average fitness functions after twenty times test with different crossover probability when other parameters are the same. From fig three, we can see when the crossover probability is gradually increasing, particles' average fitness function begin to decrease and then it will gradually increase, which indicates that cashes existing in particles gradually decrease at first and then gradually increase. Therefore, crossover probability 0.1 is a better choice in the algorithm.

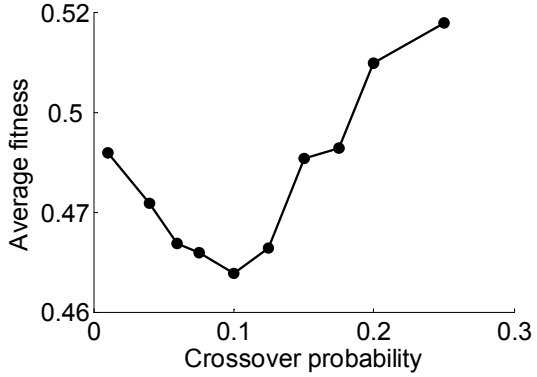


Fig. 3 The influence of crossover probability on result of the algorithm

## Conclusion

This paper introduces the program of solving timetabling problems in colleges and universities by particle swarm algorithm. In fact the program has been applied and gets a better effect. Compared with divide-and-conquer algorithm, greedy algorithm and genetic algorithm, particle swarm algorithm is faster to get the solution and greatly increases the effects of actual timetabling algorithm. Practices indicate that particle coding program and fitness function is feasible in this paper. The introduction of crossover operation increases the diversity of particle swarm, avoids premature convergence and guarantees the effectiveness and correctness of timetabling results.

## References

[1] Wang Tao, Fang Zhi-geng, Wu Hui and etc, Study on knowledge flows of disciplinary construction and innovative strategy based on the complex networks[C]//Proc of the IEEE conference on Grey Systems and Intelligent Services, (2007)1650-1654.

[2] Tao Wang, Linli Wu, Ji Zhang, The military academy teaching quality's grey correlation evaluation model and its application[C]//Proc of the IEEE conference on E-business and E-Government, (2012)2252-2255.

[3] Wu Lin-li Zhao Hai-na; Wang Tao and etc. New particle swarm optimization algorithm with global-local best minimum [J].Journal of Computer Applications, 2009, 29(12):3270-3272.

[4] Wang, Tao, Ma, Guang-Bao, Wu Lin-Li, Identification of important sections for emergency rescue road network [C]//Proc of the IEEE conference on Industrial and Information Systems, (2010)529-532.