A Novel Particle Swarm Optimization Applied to Multi-flight Refueling Service Scheduling

Jianli Ding[1] Xinru Wang[1] Tao Xu[1] Weiqiang Liu[2]

[1]College of Computer Science and Technology, Civil Aviation University of China, Tianjin 300300, P. R. China
[2]College of Computer Science, Sichuan University, Chengdu 610064, P. R. China

Abstract

Optimization of multi-flight refueling service scheduling is very important for the recovery efficiency of flight delays. After analyzing the constraints and objective of multi-flight refueling service scheduling problem, a model is put forward. A novel particle swarm optimization is designed to solve the problem. A two-dimensional particle representation and a method for generating flight scheduling solution are defined. Simulations are carried out by using flight data from a domestic airport. The results show that the novel algorithm has advantages of optimization over other scheduling algorithms, and the purposes of optimizing scheduling and improving recovery efficiency of flight delays are achieved.

Keywords: Particle swarm optimization, Multi-flight refueling service scheduling, Evolutionary strategy

1. Introduction

Airport refueling services are provided by refueling vehicles. When flight delays are recovered, many flights need to be refueled in a certain period of time. Refueling vehicles are mostly dispatched according to the sequence of flight queue at present. This method is simple but time-consuming, and then it is not conducive to the rapid recovery from flight delays and the rapid evacuation of stranded passengers. Efficient multi-flight refueling service scheduling can save the completion time of refueling services, and then the delayed flights can take off rapidly and safely and the stranded passengers are evacuated as soon as possible.

Particle swarm optimization (PSO) was originally introduced by Kennedy and Eberhart in 1995 [1]-[2]. PSO, a novel simulated evolutionary algorithm, inspired by social behaviors of a flock of birds, provides a new method for complicated combinatorial optimization problems. The algorithm is simple and it can be implemented easily. Currently, PSO has been successfully applied in function optimization, neural network training, fuzzy system control and other areas, and it has been continuously improved. However, there are seldom researches on PSO for discrete combinatorial optimization problems [3]-[5], especially in the field of production scheduling. Now, we try to apply PSO to optimize multi-flight refueling service scheduling.

2. Model of multi-flight refueling service scheduling problem

2.1. Assumptions

Before the model is established, the conditions are assumed as follows:

- There are some refueling vehicles in the airport. Loading capacities and refueling abilities of all the refueling vehicles are the same. They can provide refueling services at the same time. Each refueling vehicle can continuously provide refueling services for multi-flights.
- Each flight can be refueled by any refueling vehicle, and besides, the flight should be refueled without interruption until it has enough fuel.
- After a refueling vehicle provides a refueling service, the carrying fuel of the refueling vehicle is reduced based on the fuel demand of the flight. If the refueling vehicle has insufficient fuel to refuel a flight, it could be argued that the fuel has been depleted.
- Each refueling vehicle is initially loaded with a fixed quantity of fuel. Once its fuel is depleted, the refueling vehicle must immediately return to the fuel depot and it is loaded with the same fixed quantity of fuel. After the refueling vehicle is supplemented with the fuel, it must immediately return to provide a new refueling service.

2.2. Multi-flight refueling service scheduling problem

Multi-flight refueling service scheduling problem (MRSSP) can be described as follows:

- In a certain period of time, n flights, F_1, F_2,⋯, F_n, need to be refueled. The refueling time of the i-th flight is denoted as T_i (i=1,2,⋯,n). For each refueling vehicle, the road time, during which the refueling vehicle moves towards the i-th flight, is the same, and it is denoted as R_i.
- There are m refueling vehicles, V_1, V_2,⋯, V_m, in a certain airport. The time, during which initial fuel of the refueling vehicle is completely depleted only once, is denoted as C. The time, during which current residual fuel of the j-th refueling vehicle is completely depleted only once, is denoted as L_j (j=1,2,⋯,m). For each refueling vehicle, the time, during which the refueling vehicle goes to the fuel depot in order to be supplemented with the fuel and then returns, is the same, and it is denoted as S. After the refueling vehicle is supplemented with the fuel, set $L_j = C$.

The scheduling task is to give a flight scheduling solution, with which the n flights are sufficiently refueled by the m refueling vehicles as fast as possible.

2.3. The model

The maximum completion time of refueling vehicles is denoted as F. The scheduling objective is to minimize F. Therefore, the mathematical model of MRSSP is defined as follows:

$$\begin{aligned} &\min F = \min \max_{j=1}^{m} \sum_{i=1}^{n} (\alpha_{ij} T_i + \beta_{ij} R_i + \gamma_{ij} S) \\ &s.t. \sum_{j=1}^{m} \alpha_{ij} = 1 (i = 1,2,\cdots,n) \\ &\alpha_{ij} = 0,1 \\ &\beta_{ij} = 0,1 \\ &\gamma_{ij} = 0,1 \end{aligned} \tag{1}$$

where if the i-th flight is refueled by the j-th refueling vehicle, set $\alpha_{ij} = 1$, then if $L_j < T_i$, set $\beta_{ij} = 0$, $\gamma_{ij} = 1$ and $L_j = C - T_i$, which means the j-th refueling vehicle needs to be supplemented with the fuel and return, otherwise, set $\beta_{ij} = 1$, $\gamma_{ij} = 0$ and $L_j = L_j - T_i$, which means current residual fuel of the j-th refueling vehicle is enough to refuel the flight; if the i-th flight is not refueled by the j-th refueling vehicle, set $\alpha_{ij} = 0$, $\beta_{ij} = 0$ and $\gamma_{ij} = 0$. $\sum_{i=1}^{n} (\alpha_{ij} T_i + \beta_{ij} R_i + \gamma_{ij} S)$ means the completion time of the j-th refueling vehicle. $\sum_{j=1}^{m} \alpha_{ij} = 1$ means the i-th flight can be refueled by only one refueling vehicle.

3. Basic PSO

Particle swarm optimization simulates the way in which a flock of birds find a food source. In PSO, each single solution is a "bird" (position) in the search space. We call it "particle". All of particles have fitness values which are evaluated by the fitness function to be optimized, and velocities which direct the flying of the particles. The particles fly through the search space by following the current best particles. The best particles contain individual best particle and global best particle. The former is the best solution individual has achieved so far, and it is also called individual best position. The latter is the best solution the whole swarm has achieved so far, and it is also called global best position. Suppose that the search space is D-dimensional and the particle swarm consists of PN particles. The velocity and position of the particle can be represented according to the following updating equations:

$$\begin{aligned} V_i^{t+1} = V_i^t + c_1 \cdot r_1 \cdot (P_i^t - X_i^t) \\ + c_2 \cdot r_2 \cdot (P_g^t - X_i^t) \end{aligned} \tag{2}$$

$$X_i^{t+1} = X_i^t + V_i^{t+1} \tag{3}$$

where V_i means the velocity of the i-th particle, which is a D-dimensional vector. The value of V_i is a real number in the range of [v_{min}, v_{max}]. It is set to the corresponding upper or lower limit value when its value oversteps the range. X_i means the position of the i-th particle, which is another D-dimensional vector. The value of X_i is a real number in the range of [x_{min}, x_{max}]. c_1 and c_2 are two positive constants called acceleration factors, and the default values of c_1 and c_2 are set to 2.0. P_i means the individual best position of the i-th particle. P_g means the global best position of the whole swarm. r_1 and r_2 are two uniformly distributed random numbers in the range of [0, 1].

Generally, the position is denoted as a solution of the problem, and the velocity is denoted as the changes of the position. But PSO with this particle representation can't outperform other existed scheduling algorithms for solving MRSSP. Therefore, a novel particle swarm optimization (NPSO) is designed to solve the problem in this paper.

4. NPSO for solving MRSSP

4.1. Particle representation

Suppose that the total of flights, which need to be refueled, is denoted as n. The flights are labeled by the number 1,2,3,⋯,n. We use two-dimensional particle representation. The first n-dimensional vector is denoted as the position of the particle. It is denoted as $X_i = (x_{i1}, x_{i2}, \cdots, x_{ij}, \cdots, x_{in})$, where i=1,2,⋯, PN. The size of the particle swarm is denoted as PN. The value of x_{ij}, where j=1,2,⋯, n, is a real number in the range of [0,m]. The total of refueling vehicles is denoted as m. The second n-dimensional vector is denoted as a sequence of the flight numbers. The scheduling solution is generated according to the sequence. Sort the values of the position by size, and the corresponding indexes are sorted correspondingly. The sequence of the sorted indexes is the second n-dimensional vector. For example, suppose that n = 3 and m = 9. The values of the position are shown in Table 1. Sort the values of the position, and then the sequence of the sorted indexes is generated. The sequence is a sequence of the flight numbers and it is shown in Table 2.

Index	x_{ij}
1	0.56
2	0.02
3	0.78
4	1.82
5	1.36
6	1.91
7	0.4
8	0.69
9	0.87

Table 1: The values of the position.

Index	Flight number
1	2
2	7
3	1
4	8
5	3
6	9
7	5
8	4
9	6

Table 2: The sequence of the flight numbers.

The velocity is denoted as the changes of the values of the position. The value of the velocity is a real number in the range of [-m, m]. At every iteration, if the value oversteps the range, it is set to the corresponding upper or lower limit value.

4.2. Fitness function

The fitness function is defined as the maximum completion time of refueling vehicles, namely F in equation (1). The fitness value of the particle depends on the sequence of the flight numbers. The process of calculating the fitness value is explained as follows: according to the sequence of the flight numbers and the spare state of refueling vehicles, the flights are allocated to spare refueling vehicle one by one. Then a flight scheduling solution is generated. Calculate the completion time of each refueling vehicle, and the maximum one is the fitness value.

4.3. Updating equations of the particle

The velocity updating equation is the same as equation (2).

The value of the position must be a real number in the range of [0,m], so we reconstruct the position updating equation as follows:

$$x_{ij}^{t+1} = \begin{cases} x_{ij}^{t} + v_{ij}^{t+1} + m, & if \quad x_{ij}^{t} + v_{ij}^{t+1} < 0 \\ x_{ij}^{t} + v_{ij}^{t+1} - m, & if \quad x_{ij}^{t} + v_{ij}^{t+1} > m \\ x_{ij}^{t} + v_{ij}^{t+1}, & else \end{cases} \tag{4}$$

where i means the i-th particle of the swarm, and j means the j-th weight of the position vector of the i-th particle.

4.4. Algorithm description

The algorithm can be summarized according to the following steps:

Step 1. Set the size of the particle swarm and the value of the maximum iteration. For each particle, the value of the position is initialized with a random real number in the range of [0,m]. The value of the velocity is initialized with a random real number in the range of [-m, m]. Sort the values of the position, and then the sequence of the flight numbers is generated. Calculate the fitness value, and initialize the individual best position (P_i) and the global best position (P_g).

Step 2. If the maximum iteration is reached, go to step 5.

Step 3. For each particle, calculate its velocity and position.

Step 3-1. Calculate the new velocity according to equation (2). If the value of the velocity is smaller than -m, it is set to -m. If the value of the velocity is larger than m, it is set to m.

Step 3-2. Calculate the new position according to equation (4).

Step 3-3. The new sequence of the flight numbers is generated according to the new position. Calculate the fitness value. If the current fitness value is better than the fitness value of P_i, update P_i.

Step 4. If the fitness value of P_i is better than the fitness value of P_g, update P_g. Then go to step 2.

Step 5. The best flight scheduling solution is generated according to the corresponding sequence of

the flight numbers of the global best position. Output the solution and the best fitness value.

5. Simulation results and discussions

There are 10 refueling vehicles in a domestic airport, and 30 flights need to be refueled in a certain period of time. The flight numbers and the refueling time and road time of each flight are shown in Table 3. The time, during which initial fuel of the refueling vehicle is completely depleted only once, is set to 100 minutes. The time, during which the refueling vehicle goes to the fuel depot in order to be supplemented with the fuel and then returns, is set to 30 minutes.

Flight No.	Ref. time	Road time	Flight No.	Ref. time	Road time
1	12	5	16	31	15
2	13	5	17	55	11
3	24	8	18	24	7
4	16	7	19	27	12
5	20	8	20	57	15
6	27	9	21	45	9
7	39	10	22	27	10
8	52	12	23	31	14
9	16	6	24	24	13
10	48	14	25	27	5
11	40	12	26	52	12
12	54	13	27	60	8
13	57	11	28	54	13
14	45	7	29	60	7
15	31	6	30	52	11

Table 3: The refueling time and road time of 30 flights (min.).

Using the data in Table 3, the result of production scheduling, which is obtained according to the sequence of flight queue, is 192 minutes. The parameters of NPSO are set as follows: the size of the particle swarm is set to 50, the value of the maximum iteration is set to 800, and the values of c_1 and c_2 are set to 2.0. 10 consecutive optimization results are shown in Table 4. It is observed from Table 4 that 10 results of NPSO are better than the result of production scheduling. The best result of NPSO is 165 minutes. The best result of NPSO is 27 minutes less than the result of production scheduling. After MRSSP is solved by NPSO, the completion time of refueling services is saved and the recovery efficiency of flight delays is improved. The best flight scheduling solution of NPSO is shown in Table 5.

We use basic PSO and evolutionary strategy (ES) to compare with NPSO. In ES, (μ + λ)-strategy is selected. Set μ = 50 and λ = 30. And create the encoding method based on the flight numbers. Recombination operator is extended based on order crossover. It constructs an offspring individual from two parent individuals. It is explained as follows: two parent individuals are chosen randomly from μ individuals. Two different crossover points are chosen randomly on both parent individuals. In order to generate a new offspring individual, the substring between the two crossover points in the first parent individual replaces the corresponding substring in the second parent individual. Mutation operator is extended based on inversion. It is explained as follows: two different points are chosen randomly on the mutated individual. Exchange the left substring of the individual and its right substring in order to generate a new individual.

Trial run	NPSO result	PSO result	ES result
1	168	186	178
2	165	191	177
3	168	187	175
4	166	185	180
5	166	185	175
6	168	183	180
7	165	187	183
8	167	188	177
9	166	189	173
10	167	187	181

Table 4: 10 consecutive optimization results of NPSO, PSO and ES (min.).

Refueling vehicle	Flight number
1	6, 15, 30
2	24, 25, 27
3	12, 20
4	11, 18, 26
5	3, 7, 14
6	16, 17, 22
7	10, 19, 23
8	1, 8, 28
9	4, 5, 9, 13
10	2, 21, 29

Table 5: The best flight scheduling solution of NPSO.

The parameters of PSO are the same as NPSO. The value of the maximum iteration is also set to 800 in ES. 10 consecutive optimization results of PSO and ES are also shown in Table 4. It is also observed from Table 4 that the best result of NPSO is 165 minutes, the worst result of NPSO is 168 minutes, the best result of PSO is 183 minutes, and the best result of ES is 173 minutes. 10 consecutive optimization results of NPSO are obviously better than the results of PSO and ES. Furthermore, the worst result of NPSO is better

than the best results of PSO and ES. Obviously, NPSO has stronger search optimization ability than PSO and ES.

Typical convergence histories for three algorithms are shown in Fig. 1. It is observed from Fig. 1 that the initial fitness value of NPSO is obviously much better than those of PSO and ES. The best fitness value of NPSO is obviously better than those of PSO and ES. The value of the corresponding iteration of NPSO, at which the best fitness value is obtained for the first time, is smaller than those of PSO and ES. Obviously, NPSO outperforms PSO and ES in optimizing efficiency and global search optimization ability because the whole swarm moves towards the best solution faster.

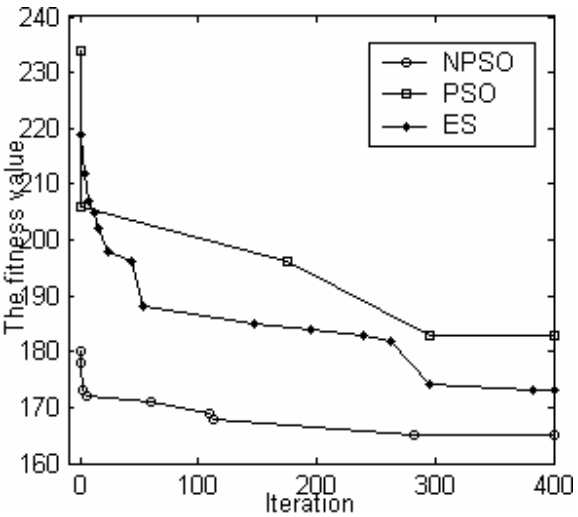

Fig. 1: Typical convergence histories for NPSO, PSO and ES.

6. Conclusions

This paper analyzes the constraints and objective of multi-flight refueling service scheduling problem and puts forward its model. NPSO is designed to solve the problem. A two-dimensional particle representation and a method for generating flight scheduling solution are defined. The effectiveness and feasibility of NPSO are verified by simulations and algorithm comparisons, and the purposes of optimizing scheduling and improving recovery efficiency of flight delays are achieved. After MRSSP is solved by NPSO, the completion time of refueling services is saved, and then the delayed flights can take off rapidly and safely and the stranded passengers are evacuated as soon as possible.

Acknowledgement

This work is partially supported by High Technology Research and Development Programme of China (Grant No. 2006AA12A106) and National Nature Science Foundation of China (Grant No. 60572167).

References

[1] J. Kennedy, R.C. Eberhart, Particle swarm optimization. *IEEE International Conference on Neural Networks*, pp. 1942-1948, 1995.

[2] R.C. Eberhart, J. Kennedy, A new optimizer using particle swarm theory. *Proc. Sixth International Symposium on Micro Machine and Human Science*, pp. 39-43, 1995.

[3] L. Huang, K.P. Wang, C.G. Zhou, Particle swarm optimization for traveling salesman problems. *Journal of Jilin University*, 41: 477-480, 2003.

[4] S. Gao and J.Y. Yang, Solving multiprocessor scheduling problem by particle swarm optimization algorithm. *Computer Engineering and Applications*, 27: 72-73, 104, 2005.

[5] Z.X. Liu and S.M. Wang, Research on parallel machines scheduling problem based on particle swarm optimization algorithm. *Computer Integrated Manufacturing Systems*, 12: 183-187, 2006.