Int j simul model 22 (2023) 1, 133-144
Original scientific paper

MODEL AND ALGORITHM FOR THE SKILL CAPACITATED VRP WITH TIME WINDOWS IN AIRPORTS

Zhao, P. X.[#]; Dai, M. N.; Han, X.; Xu, C. & Du, C. C.
School of Management, Shandong University, Jinan 250100, China
E-Mail: pxzhao@sdu.edu.cn ([#] Corresponding author)

Abstract
The scheduling efficiency of airport ground support vehicles is an important factor affecting the on-time performance of flights. In this paper, the Skill Capacitated Vehicle Routing Problem with Time Windows is proposed and the corresponding algorithm is designed for main airport ground support vehicles. Firstly, we decompose the transit process of flights using simple temporal network and time decoupling to obtain the time windows of each ground support service. Secondly, a vehicle scheduling model is constructed according to the characteristics of each vehicle, which aims at minimizing the number of vehicles used and balancing the workload of each vehicle. Finally, the elitist genetic algorithm with large neighbourhood search is designed to solve the problem and is compared with the standard genetic algorithm. The effectiveness of the model and the algorithm is illustrated by a real data example from Beijing Capital International Airport.
(Received in November 2022, accepted in January 2023. This paper was with the authors 1 month for 1 revision.)

Key Words: Airport Ground Support Vehicle, Collaborative Scheduling, Simple Temporal Network, Genetic Algorithm, Vehicle Routing Problem (VRP)

1. INTRODUCTION

With the rapid development of the civil aviation transportation industry in recent years, the contradiction between the increasing number of flight take-offs and landings and the limited ground service resources of most airports has become increasingly prominent, resulting in frequent flight delays. According to statistics, unfavourable ground support services account for up to 15 % of the causes of flight delays [1]. Especially in hub airports, the proportion of transit flights is relatively high. Since most transit flights require ground service vehicles such as ferry vehicles, baggage vehicles, potable water vehicles and tractors for a series of support services, how to improve the service efficiency is a key problem to improve the flight punctuality rate. At present, most of the ground support vehicles in civil airports are manually dispatched, which is inefficient and often causes flight delays. Therefore, cooperative dispatching of ground support vehicles plays an important role in improving airport service level.

With the rapid development of air transport, more and more scholars turned from the scheduling of single type ground support vehicle to the collaborative scheduling of multi-type vehicle [2, 3]. Cheung et al. [4] developed a collaborative scheduling optimization model to maximize the utilization of ground support vehicles and enhance the logistics of aircraft maintenance activities. Liu et al. [5] proposed a bi-objective mixed integer programming model to minimize the number of vehicles used and the total extra time cost of vehicles. Zeng and Jia [6] and Feng and Ren [7] both developed mixed integer programming models for the collaborative scheduling problem of fuelling vehicles and ferry vehicles. Zhao et al. [8] established a collaborative scheduling optimization model of ferry vehicles and tractors with the objectives of minimizing the number of vehicles used and balancing the workload of vehicles.

A few studies used the idea of decomposition to deal with the collaborative scheduling problem. Padron et al. [9] decomposed the collaborative scheduling problem of ground support vehicles into Temporal Constraints Level Procedure (TCLP) and Routing Level Procedure

(RLP). Feng et al. [10] established Simple Temporal Network (STN) models for flights and decoupled them to obtain the service time windows for various vehicles, then constructed a scheduling model for fuelling vehicles with the objective of minimizing the total service time. Chen et al. [11] proposed a STN-based flight transit service time planning model to optimize the collaborative planning of the flight transit support time.

Ground support vehicle scheduling problem is a kind of vehicle routing problem [12, 13]. A few scholars firstly decomposed the time of each ground support service and then established the basic Vehicle Routing Problem with Time Windows (VRPTW) model, which can effectively reduce the complexity of the model while satisfying the timing constraints among the services. However, since there are many types of airport ground support vehicles and great differences between them, the applicability of the model should be improved. Based on this, we propose the Skill Capacitated Vehicle Routing Problem with Time Windows (Skill CVRPTW) by considering the vehicle as an individual with specific service skills. Skill VRP, which was first proposed by Cappanera et al. [14], derives from the need to dispatch personnel with specific skills to complete after-sales services and has been widely applied in home health care, technician scheduling, and so on. In this paper, we firstly obtain the time windows of each service for transit flights using Simple Temporal Network (STN) and time decoupling algorithm, then the Skill CVRPTW model is established by combining the characteristics of vehicles such as ferry vehicles, tractors and potable water vehicles. Finally, the Elite Genetic Algorithm with Large Neighbourhood Search (EGA-LNS) is designed to solve the model.

This paper is organized as follows. In Section 2, the STN model is established based on the process of ground support services for a transit flight and decoupled using decoupling algorithm to obtain the time windows of each service. A bi-objective Skill CVRPTW model for support vehicle scheduling is established in Section 3. EGA-LNS is designed in Section 4. In Section 5, the real data of Beijing Capital International Airport are used to verify the effectiveness of the model and algorithm. The conclusions are presented in Section 6.

2. TIME DECOMPOSITION OF GROUND SUPPORT SERVICE PROCESS

2.1 Construction of STN

STN [15] is a network formed by a series of point-in-time variables as nodes and the constraints between them as arcs. It is a class of methods for describing and solving the Simple Temporal Problems (STP), which can transform multiple tasks with temporal relations into a weighted directed graph, and has a wide range of applications in resource scheduling and task planning. This paper establishes the STN of ground support services for transit flights. The process of ground support services for transit flights is shown in Fig. 1.

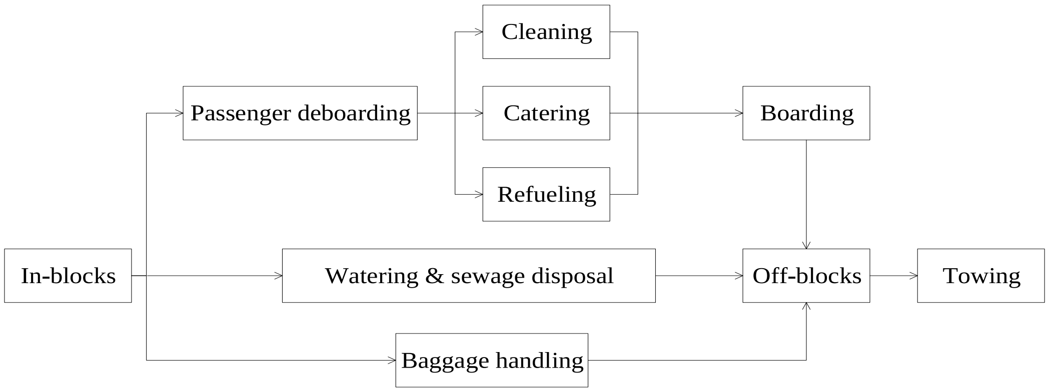

Figure 1: The process of ground support services.

All the ground support services, except for the towing service, must be completed between the in-blocks and off-blocks, and the catering, cleaning and refuelling all need to start after the deboarding. Each service is represented by two time point variables, representing the start time and end time respectively. Taking flight CZ6125/CZ6126 as an example, the related point variables are shown in Table I.

Table I: Time point variables and meanings for flight CZ6125/CZ6126.

Time variable	**Meaning**	**Time variable**	**Meaning**
t_0	The time reference point that takes the value of 0. Here the selected reference time is 0:00.	t_{12}, t_{13}	Start and end times of watering.
t_1	STA.	t_{14}, t_{15}	Start and end times of sewage disposal.
t_2, t_3	Start and end times of deboarding.	t_{16}, t_{17}	Start and end times of boarding.
t_4, t_5	Start and end times of baggage handling.	t_{18}	STD.
t_6, t_7	Start and end times of cleaning.	t_{19}	Arrive time of tractor.
t_8, t_9	Start and end times of refuelling.	t_{20}	End time of towing.
t_{10}, t_{11}	Start and end times of catering.		

The duration constraint for each service and the interval constraints between services form the constraint set C, as shown in Table II. C_1 is the set of constraints on the sequence and interval time between the services, and C_2 is the set of constraints on the time required for each service. For example, the constraint $0 \leq t_2 - t_1 \leq 2$ indicates that the deboarding should begin after, but no later than two minutes after the in-blocks. Constraint $9.5 \leq t_3 - t_2 \leq 11.5$ indicates that the standard operating time for the deboarding of flight CZ6125 is 9.5 minutes, and a certain margin is added to cope with the unexpected situations encountered in the actual ground support process, but the maximum operating time should not exceed 11.5 minutes.

Table II: Time constraint set *C* for flight CZ6125/CZ6126.

Service interval constraint set C_1	**Service duration constraint set C_2**
$0 \leq t_2 - t_1 \leq 2$	
$t_4 - t_1 \geq 0$	
$t_{10} - t_3 \geq 0$	$9.5 \leq t_3 - t_2 \leq 11.5$
$t_6 - t_3 \geq 0$	$31.8 \leq t_5 - t_4 \leq 50$
$t_8 - t_3 \geq 0$	$20 \leq t_7 - t_6 \leq 30$
$0 \leq t_{12} - t_1 \leq 30$	$17 \leq t_9 - t_8 \leq 35$
$0 \leq t_{14} - t_1 \leq 30$	$22.2 \leq t_{11} - t_{10} \leq 35$
$t_{16} - t_{11} \geq 0$	$20 \leq t_{13} - t_{12} \leq 40$
$t_{16} - t_7 \geq 0$	$20 \leq t_{15} - t_{14} \leq 40$
$t_{16} - t_9 \geq 0$	$18 \leq t_{17} - t_{16} \leq 28$
$t_{18} - t_{13} \geq 20$	$7 \leq t_{20} - t_{18} \leq 7$
$t_{18} - t_{15} \geq 20$	$3 \leq t_{18} - t_{19} \leq 5$
$t_{18} - t_5 \geq 0$	$7 \leq t_{20} - t_{18} \leq 7$
$0 \leq t_{18} - t_{17} \leq 10$	

In summary, considering the nine services of deboarding and boarding, baggage handling, catering, cleaning, refuelling, watering, sewage disposal and towing, the decomposition of the ground support service process can be described as a simple temporal problem $S = (T, C)$, where T includes 21 time point variables in Table I and $C = C_1 \cup C_2$. This simple time problem can be converted to a STN, as shown in Fig. 2.

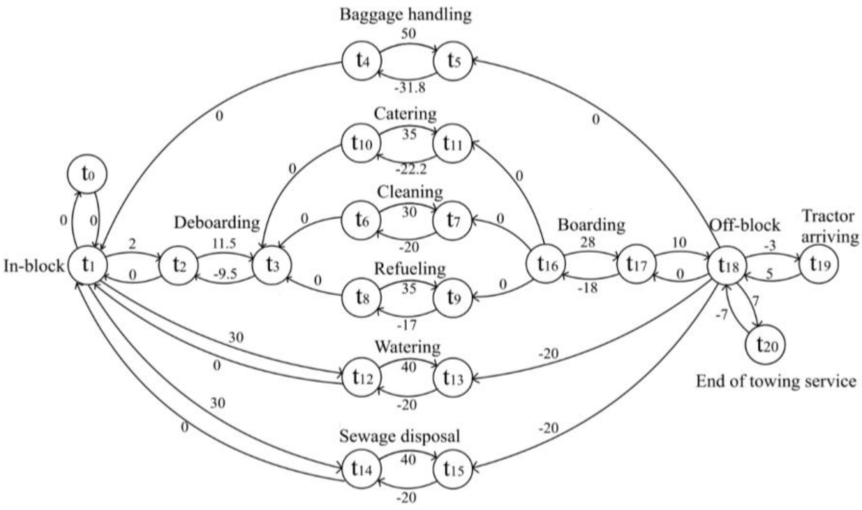

Figure 2: The STN of the ground support service for flight CZ6125/CZ6126.

2.2 Decoupling of STN

There are generally multiple agents with interdependencies between the tasks they have to perform in collaborative scheduling problems. The decoupling of the collaborative scheduling problems is called Time Decoupling Problem (TDP) if the dependencies between the tasks are time-constraints [16]. The basic idea of TDP is to divide the STN into multiple sub-STNs and remove the time constraints between the sub-STNs by adding appropriate constraints between the time reference point and each time point variable.

In this paper, each service is regarded as a sub-STN, and 11 sub-STNs corresponding to the in-blocks, deboarding, baggage handling, catering, cleaning, refuelling, watering, sewage disposal, boarding, off-blocks and towing services are obtained by decoupling. The start time windows of each service can be obtained by calculating the shortest path matrix of each sub-STN. By applying the above decoupling algorithm to the STN constructed in Section 2.1, the planning time windows of some services are obtained, as shown in Table III.

Table III: Time windows of services for flight CZ6125/CZ6126.

Flight number	Start time of deboarding	Start time of boarding	Start time of towing	End time of towing	Start time of watering
CZ6125/CZ6126	[555,557]	[591.85,597]	[610,612]	622	[555,575]

Note: 00:00 is the time reference point, and its corresponding value is 0.

The planning time windows of each service can be obtained for all transit flights using this method. Each support department can arrange its own support service within all the executable time based on the above planning results, which can not only meet the actual needs of each department, but also ensure that multiple support service departments can coordinate with each other to complete various services of flights.

3. SCHEDULING OPTIMIZATION MODEL CONSTRUCTION

3.1 Problem description

The time windows of various services obtained in Section 2 are used to optimize the scheduling of ferry vehicles, tractors, potable water vehicles, sewage disposal vehicles, tankers and catering trucks. The problem can be described as there are n flights waiting for services in a period of time with known information, and the vehicle type must match the aircraft type when the vehicle serves the flight. The problem is to find the optimal vehicle scheduling to minimize the number of vehicles used and balance the workload of each vehicle, while satisfying the time window and service demand of each flight. Some other assumptions of the model are as follows.

(1) Virtual flights are introduced for flights that require multiple ferry vehicles. Two virtual flights j and k are added if real flight i requires 3 ferry vehicles. Real flight i and virtual flights j and k require one ferry vehicle respectively.

(2) The vehicles all depart from the depot before starting their assigned tasks and return to the depot after completing their tasks.

(3) The start time of each service must fall within the time window of the flight, and the vehicle will have to wait if it arrives earlier than the earliest start time allowed.

(4) The service capacities of the same type of vehicles are the same. Resources should be replenished when the remaining resources of the vehicle are not enough to serve the next flight.

3.2 Model construction

In this paper, Skill CVRPTW model is established with the goal of minimizing the number of vehicles used and balancing the workload of each vehicle. The sets, parameters and decision variables involved in this model are shown in Table IV.

Table IV: Sets, parameters, and decision variables in the model.

Sets	
F	The set of flights, $F = \{1, 2, \ldots, n\}$.
F_1	$F_1 = F \cup \{0\} \cup \{n+2\}$, where 0 represents depot, and $n+2$ represents the resupply station.
F_2	$F_2 = F \cup \{n+1\} \cup \{n+2\}$, where $n+1$ represents depot.
F_3	$F_3 = F \cup \{n+2\}$
M	The set of vehicle types
K^m	The set of m-type vehicles
Parameters	
$[e_i, l_i]$	The time window of flight i, $i \in F$
q_i	The demand of flight i, $i \in F$
s_i	The service duration required for flight i, $i \in F_3$
t_{ij}	The travel time of the vehicle from the parking position of flight i to the parking position of flight j, $i, j \in F_3$
u_i^m	$u_i^m = \begin{cases} 1, \text{flight } i \text{ can be served by } m-\text{type vehicle} \\ 0, \text{otherwise} \end{cases}, i \in F_3, m \in M$
C	The capacity of vehicles
Decision variables	
x_{ij}^{km}	$x_{ij}^{km} = \begin{cases} 1, m-\text{type vehicle } k \text{ serves flights } i \text{ and } j \text{ consecutively} \\ 0, \text{otherwise} \end{cases}, \forall i \in F_1, j \in F_2, k \in K^m, m \in M$
st_i^{km}	The time when the m-type vehicle k start to serve flight i, $i \in F_3, k \in K^m, m \in M$
cs_i^{km}	The amount of resources of m-type vehicle k before serving flight i, $i \in F_3, k \in K^m, m \in M$
ce_i^{km}	The amount of resources of m-type vehicle k after serving flight i, $i \in F_3, k \in K^m, m \in M$

According to the above assumptions and notations, a bi-objective scheduling optimization model for the ground support vehicles can be constructed as follows.

$$\min \sum_{j \in F} \sum_{k \in K^m} \sum_{m \in M} x_{0j}^{km} \tag{1}$$

$$\min \left\{ \max_{k \in \cup_{m \in M} K^m} \left\{ \sum_{i \in F} \sum_{j \in F_2, i \neq j} x_{ij}^{km} \right\} - \min_{k \in \cup_{m \in M} K^m} \left\{ \sum_{i \in F} \sum_{j \in F_2, i \neq j} x_{ij}^{km} \right\} \right\} \tag{2}$$

$$\sum_{j \in F_2, j \neq i} \sum_{k \in K^m} \sum_{m \in M} x_{ij}^{km} = 1, \forall i \in F \tag{3}$$

$$\sum_{j\in F_2, j\neq i} x_{ij}^{km} \leq u_i^m, \forall i \in F, k \in K^m, m \in M \tag{4}$$

$$\sum_{j\in F} x_{0j}^{km} = \sum_{j\in F} x_{j,n+1}^{km} \leq 1, \forall k \in K^m, m \in M \tag{5}$$

$$\sum_{i\in F_1, i\neq j} x_{ij}^{km} = \sum_{h\in F_2, h\neq j} x_{jh}^{km}, \forall j \in F_3, k \in K^m, m \in M \tag{6}$$

$$ce_i^{km} = C, \forall i \in \{0, n+2\}, k \in K^m, m \in M \tag{7}$$

$$M\left(1 - \sum_{j\in F_2, j\neq i} x_{ij}^{km}\right) + cs_i^{km} \geq q_i, \forall i \in F, k \in K^m, m \in M \tag{8}$$

$$ce_i^{km} = cs_i^{km} - q_i \cdot \sum_{j\in F_2, j\neq i} x_{ij}^{km}, \forall i \in F, k \in K^m, m \in M \tag{9}$$

$$cs_j^{km} \leq ce_i^{km} + M(1 - x_{ij}^{km}), \forall i \in F_1, j \in F_2, i \neq j, k \in K^m, m \in M \tag{10}$$

$$cs_j^{km} \geq ce_i^{km} - M(1 - x_{ij}^{km}), \forall i \in F_1, j \in F_2, i \neq j, k \in K^m, m \in M \tag{11}$$

$$e_i \cdot \sum_{j\in F_2, j\neq i} x_{ij}^{km} \leq st_i^{km} \leq l_i \cdot \sum_{j\in F_2, j\neq i} x_{ij}^{km}, \forall i \in F, k \in K^m, m \in M \tag{12}$$

$$st_i^{km} + s_i + t_{ij} \leq st_j^{km} + M\left(1 - x_{ij}^{km}\right), \forall i, j \in F_3, i \neq j, k \in K^m, m \in M \tag{13}$$

$$0 \leq cs_i^{km}, ce_i^{km} \leq C, \forall i \in F_3, k \in K^m, m \in M \tag{14}$$

$$x_{ij}^{km} \in \{0,1\}, \forall i \in F_1, j \in F_2, k \in K^m, m \in M \tag{15}$$

Eqs. (1) and (2) are the objective functions to minimize the number of vehicles used and balance the workload of each vehicle. The constraint (3) indicates that flight i can only be served by one vehicle. The constraint (4) indicates that the vehicle can serve the flight only if the vehicle type matches the aircraft type. The constraint (5) indicates that each vehicle can be dispatched at most once. Eq. (6) is flow balance constraint. The constraint (7) indicates that the vehicle is fully loaded when it departs from the resupply station if the vehicle goes to resupply resources. The constraint (8) means that if the vehicle serves flight i, its resources before serving flight i must meet the demand of this flight. The constraint (9) represents the relationship between the remaining resources of the vehicle before and after serving flight i. The constraints (10) and (11) together represent the relationship between the amount of remaining resources of the same vehicle when it serves two flights consecutively. The constraint (12) means that the service start time of flight i should be within its time window. The constraint (13) represents the time relationship when the same vehicle serves two flights consecutively. The constraints (14) and (15) represent the value range of variables.

When the dispatched vehicles are ferry vehicles, the number of virtual flights introduced for flight i is set as f_i. Then, the set of arrival flights to be served is $NT_1 = \{1, \dots, n, n+1, \dots, n + \sum_{i=1}^{n} f_i\}$, the set of departure flights to be served is $NT_2 = \left\{n + 1 + \sum_{i=1}^{n} f_i, \dots, n + m + \sum_{i=1}^{n} f_i, n + m + 1 + \sum_{i=1}^{n} f_i, \dots, n + m + \sum_{i=1}^{n} f_i + \sum_{i=n+1+\sum_{i=1}^{n} f_i}^{n+m+\sum_{i=1}^{n} f_i} f_i\right\}$, and the set of all flights to be served is $NT_1 \cup NT_2$ when there are n real arrival flights and m real departure flights.

The tractor needs to arrive at the parking position before the departure of the flight and wait for the off-blocks to start the towing service due to its special characteristics, so its real start time is the time of off-blocks. Since the duration of the towing service is known, the end time of the towing for flights can be calculated. At this point, constraint (13) is replaced by

$$et_i^{km} + t_{ij} \le st_j^{km} + M\left(1 - x_{ij}^{km}\right), \forall i, j \in F_3, i \neq j, k \in K^m, m \in M, \tag{16}$$

where the et_i^{km} is the end time of towing service for flight i.

4. EGA-LNS

GA is widely used in solving vehicle routing problems and other optimization problems [17, 18] due to its good global search and parallel search ability. According to the characteristics of the model in this paper, we design the EGA-LNS: firstly, we use the greedy algorithm to generate the initial population to improve its quality, and secondly, we introduce the destroy-repair operation in the large neighbourhood search algorithm into the EGA to further improve the local search ability.

4.1 Algorithm design

(1) Coding and population initialization

The integer encoding method is adopted, and the chromosome length is set as the number of flights to be served. The greedy algorithm is used to generate the initial population, and the specific steps are as follows, where A is initially an empty set.

Step 1. Sort all flights in ascending order of the earliest service start time to obtain the set F_1.

Step 2. Add the first flight i in set F_1 to set A, and arrange a vehicle that meets the requirements of flight i to serve it. The service start time of flight i is $st_i = e_i$ and update $F_1 = F_1 \backslash \{i\}$.

Step 3. Calculate the service end time et_i of the vehicle serving flight i and the remaining resources after serving flight i. Judge in turn whether the vehicle type, the service end time and the amount of remaining resources satisfy the demand for the next flight j. If they do, update $A = A \cup \{j\}$, $F_1 = F_1 \backslash \{j\}$; otherwise, skip flight j and find the first flight k in set F_1 that satisfies the above three conditions, then update $A = A \cup \{k\}$, $F_1 = F_1 \backslash \{k\}$.

Step 4. Repeat *Step 3* until all flights in F_1 are traversed. Obtain the updated sets A and F_1.

Step 5. If $F_1 = \emptyset$, all flights have been added to set A and set A can be converted to a chromosome $chrom$; otherwise, go to *Step 2*.

Step 6. Half of the individuals in the initial population are identical to $chrom$, and the other half are randomly generated.

(2) Fitness function

This paper converts the bi-objective optimization problem into a single-objective one to solve. The objective function is set as:

$$f(x) = \varphi \times \text{the number of vehicles used} + \text{the workload balance of the vehicles} \tag{17}$$

The value of φ can be chosen according to the length of planning period and the number of flights to be served. The fitness function is:

$$fitness(x) = \frac{1}{f(x)} \tag{18}$$

The initialized elite chromosome is the individual with the highest fitness value in the population.

(3) Selection, crossover and mutation operators

In this paper, a roulette wheel selection strategy is adopted. The crossover operator adopts the multi-point crossover. Firstly, two crossover points are randomly generated. Secondly, Offspring 2 inherits the gene segment between the two crossover points of Parent 1, while Offspring 1 inherits the gene segment between the two crossover points of Parent 2. Finally, delete the same integers in the chromosome of Parent 1 as in the existing gene segment of Offspring 1 (inherited from Parent 2), and put the remaining gene segment after the existing gene segment of Offspring 1 in the original order to form a complete Offspring 1 chromosome. Produce Offspring 2 in the same way that Offspring 1 was produced. A schematic diagram of this process is shown in Fig. 3.

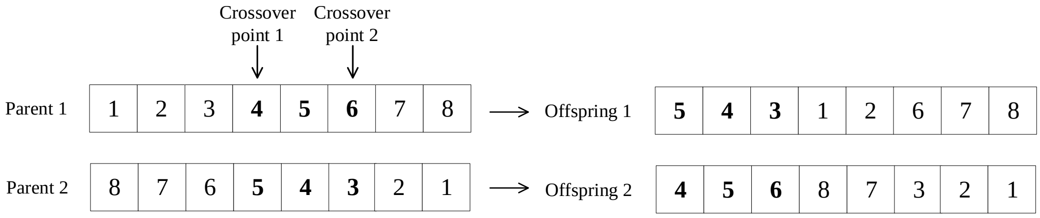

Figure 3: Crossover operation.

The mutation operator adopts the two-point exchange operator, that is, randomly select two mutation points and exchange genes at the two mutation points, as shown in Fig. 4.

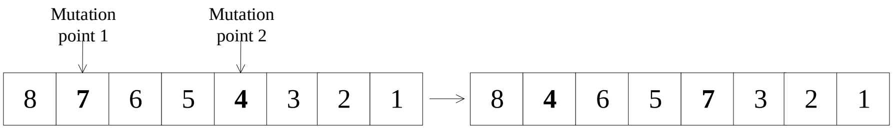

Figure 4: Mutation operation.

(4) Local search

In order to improve the local search ability of the algorithm, the destroy operator based on correlation removal (Shaw [19]) and the repair operator based on optimal greedy insertion in the large neighbourhood search algorithm are selected for local search. The calculation method of correlation R_{ij} in the correlation removal is as follows.

$$R_{ij} = \alpha_1 \cdot t_{ij} + \alpha_2 \cdot \left(\left|e_i - e_j\right| + \left|l_i - l_j\right|\right) + \alpha_3 \cdot \left(\left|q_i - q_j\right|\right) + \alpha_4 \tag{19}$$

The first term is the travel time between two flights, and weight α_1 is set to 6; the second term is the time window difference between two flights, and α_2 is set to 5; the third term is the demand difference between two flights, and α_3 is set to 4; the last term is whether two flights are served by the same vehicle, if yes, $\alpha_4 = 0$, otherwise, $\alpha_4 = 1$. The smaller the value of R_{ij}, the greater the correlation between two flights. The removal steps are as follows.

Step 1. Randomly select a flight i from the flight set and add it to the removed flight set $Remove$. The remaining flight set is $Left$.

Step 2. Randomly select a flight from $Remove$ and calculate the correlation R_{ij} of the remaining flights with that flight.

Step 3. Select the flight with the largest correlation, that is, the smallest R_{ij}, to add to set $Remove$ and update set $Left$.

Step 4. Check the number of removed flights $remove_{num}$. If $remove_{num} =$ the number of flights to be served / 10 , the removal process is complete, otherwise, go to *Step 2*.

In the optimal greedy insertion, select the first flight in $Remove$, traverse the available insertion positions in $Left$ and select the insertion position with the smallest objective value, then update sets $Remove$ and $Left$. Repeat the above steps until set $Remove = \emptyset$.

4.2 Algorithm steps

(1) Initialize the algorithm and population parameters, including population size $PopSize$, crossover probability Pc, mutation probability Pm and the maximum number of iterations $MAXGEN$.

(2) Generate $PopSize$ initial feasible solutions using greedy algorithm and random method.

(3) Calculate the fitness values of the individuals in the population.

(4) The individual with the highest fitness value is labelled as the elite chromosome. Individuals are selected using a roulette wheel selection strategy.

(5) The crossover and mutation operations are performed on individuals in the population sequentially with crossover probability Pc and mutation probability Pm.

(6) The correlation removal operator is performed on a certain number of individuals, and repair these individuals with the optimal greedy insertion operator.

(7) Remove duplicate individuals and replenish the population with random generated chromosomes.

(8) Update the elite chromosome. If there is no individual better than the elite chromosome in this population, replace the worst individual with the elite chromosome, otherwise update the elite chromosome.

(9) Repeat steps (3)-(8) until the maximum number of iterations $MAXGEN$ of the algorithm is reached.

The corresponding algorithm flowchart is shown in Fig. 5.

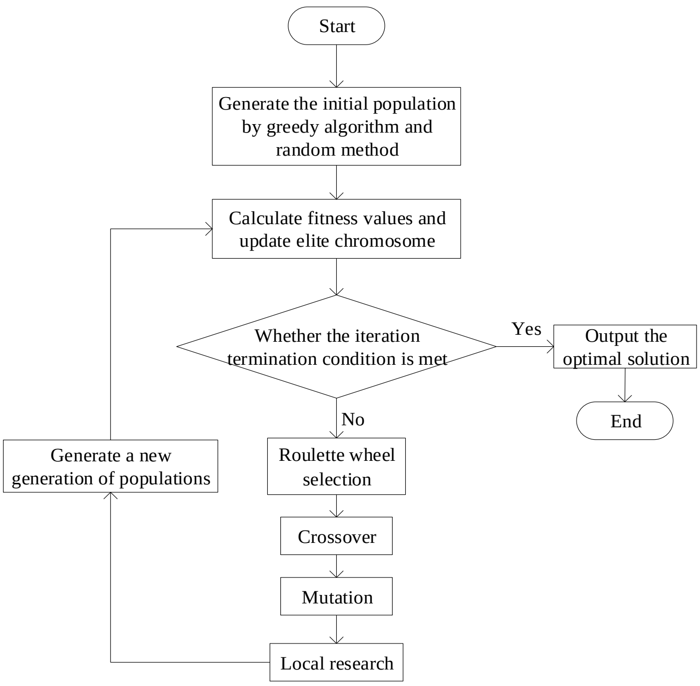

Figure 5: The flowchart of EGA-LNS.

5. NUMERICAL EXAMPLES

We select the transit flights of Beijing Capital International Airport on October 1st, 2017 to construct STNs and decouple them. There were 524 transit flights in a total of 727 flights landed. The standard time required for each service is based on the numerical results obtained through the operation time model simulation [20]. The decoupled planning time windows of deboarding and boarding, towing and watering services are shown in Table V. The planning time windows for other services are omitted.

Table V: The decoupled time windows of services.

Flight number	Start time of deboarding	Start time of boarding	Start time of towing	End time of towing	Start time of watering
MU5195/MU525	[545,547]	[584.35,592]	[605,607]	617	[556.5,569.75]
OS63/OS64	[555,557]	[654.5,657]	[675,677]	687	[575.5,590.5]
…	…	…	…	…	…
SC4677/SC4660	[1335,1337]	[1371,1377]	[1390,1392]	1402	[1335,1355]
CA1739/CA1740	[1360,1362]	[1396.85,1402]	[1415,1417]	1427	[1360,1380]

Note: 00:00 is the time reference point, and its corresponding value is 0.

According to the time windows obtained above, the service requirements of vehicles for each time period are counted, and the peak period is selected for scheduling. The obtained peak periods for ferry vehicles, tractors and potable water vehicles are 22:00 to 24:00, 17:00 to 19:00 and 00:00 to 02:00, respectively. The number of ferry vehicles required for different aircraft type categories is B-1, C-2, D-3, E-4, F-4. The small tractors can serve B and C aircraft type categories, the medium tractors can serve C, D and E aircraft type categories, and the large tractors can serve E and F aircraft type categories. It is tested that the scheduling scheme with a fewer number of vehicles used can be preferentially selected when φ = 10. All experiments are run by Matlab R2020a on a computer with Windows 10, Intel$^{(R)}$ Core $^{(TM)}$ i5-8265U CPU @ 1.60 GHz, and 8.00 GB of RAM (7.85 GB available), and related parameters of EGA-LNS are set as follows: population size $PopSize$ = 100, crossover probability Pc = 0.9, and mutation probability Pm = 0.05. The three data sets are run 10 times using GA and EGA-LNS at maximum iterations $MAXGEN$ of 50, 100, 150, and 200 respectively. The optimal solutions among all results are recorded and the average running time is calculated (Table VI).

Table VI: Comparison of optimization results of GA and EGA-LNS.

Vehicle	$MAXGEN$	Solving method							
		GA				EGA-LNS			
		Obj. 1	Obj. 2		Runtime (s)	Obj. 1	Obj. 2		Runtime (s)
Ferry vehicle	50	53	2	[1, 3]	13.7	52	2	[1, 3]	274
	100	52	2	[1, 3]	17	52	2	[1, 3]	506
	150	52	2	[1, 3]	21.79	52	2	[1, 3]	749
	200	52	2	[1, 3]	25	52	2	[1, 3]	976
Tractor	50	7	7	[1, 8]	3.9	7	3	[3, 6]	69
	100	7	7	[1, 8]	4.8	7	3	[3, 6]	98
	150	7	6	[2, 8]	5.8	7	3	[3, 6]	147
	200	7	6	[2, 8]	6.8	7	3	[3, 6]	196
Potable water vehicle	50	16	5	[1, 6]	5.9	15	2	[3, 5]	87
	100	16	4	[2, 6]	6.24	15	2	[3, 5]	166
	150	16	4	[2, 6]	8.2	15	2	[3, 5]	266
	200	15	4	[2, 6]	11.95	15	2	[3, 5]	300

It can be seen from Table VI that GA and EGA-LNS can converge to the same optimal solution within 50 generations. However, for the tractor and the potable water vehicle scheduling problems, the solutions obtained by EGA-LNS are better than GA. In the scheduling of the tractors and the potable water vehicles, EGA-LNS can still converge to the optimal solution within 50 generations, while the solution obtained by GA after 200 iterations is still inferior to EGA-LNS. There is still no improvement when GA is used to solve the selected cases and the value of $MAXGEN$ is set to 2000, indicating that GA is caught in the local optimum, while EGA-LNS with the destroy-repair operator can jump out of the local optimum and further balance the workload among vehicles. Taking the tractor scheduling problem as an example, the convergence curve with EGA-LNS at $PopSize$ = 100 and $MAXGEN$ = 200 is as follows.

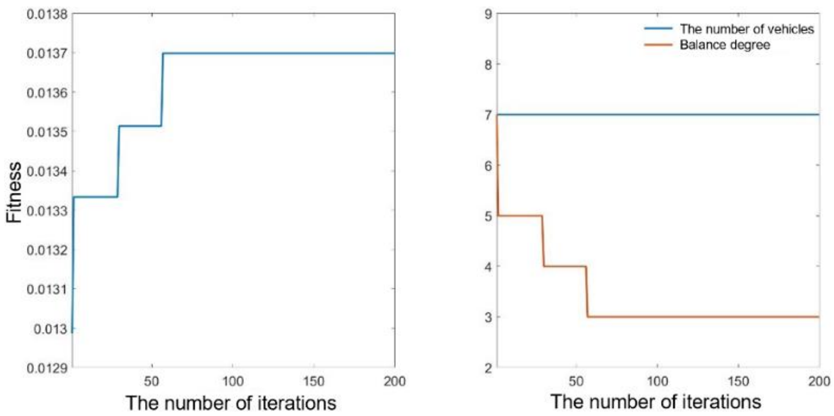

Figure 6: Convergence curve when solving the tractor scheduling problem.

In addition, the Cplex solver is also used to further verify the effectiveness of EGA-LNS. Using the idea of hierarchical sequence, the objective (1) is solved first and then it is added as a constraint to the model to solve objective (2), and the results are shown in Table VII.

Table VII: Comparison of optimization results of EGA-LNS and Cplex.

Solving method	**Objective**	**Vehicle**		
		Ferry vehicle	Tractor	Potable water vehicle
EGA-LNS	The number of vehicles	52	7	16
	Balance degree	2 [1,3]	3 [3,6]	1 [3,4]
Cplex	The number of vehicles	Out of memory	7	Out of memory
	Balance degree		3 [3,6]	

For the small-scale examples such as tractors, the exact solutions obtained by EGA-LNS and Cplex are the same. For the larger-scale examples such as ferry vehicles and potable water vehicles, EGA-LNS still works, but Cplex cannot solve due to memory overflow.

6. CONCLUSION

This paper proposes and solves the Skill CVRPTW for various airport ground support vehicles. The STN and TDP are used to obtain the time windows of each ground support service for transit flights to ensure that each service meets the time and sequence constraints. This study provides an airport service scheme with the least number of vehicles and the most balanced workload between vehicles.

ACKNOWLEDGEMENT

This research was supported in part by the National Natural Science Foundation of China (72071122, 72134004), the Natural Science Foundation of Shandong Province (ZR2020MG002), the Social Science Planning Research Project of Shandong Province (20CGLJ11).

REFERENCES

[1] Feng, M. D.; Xiao, X.; Zhou, H. (2021). Research on the multi-type joint scheduling model of airport ground support vehicles, *Journal of Wuhan University of Technology (Transportation Science & Engineering)*, *http://kns.cnki.net/kcms/detail/42.1824.U.20211028.1745.017.html*, 10 pages, online first

[2] Wang, S.; Che, Y.; Zhao, H.; Lim, A. (2021). Accurate tracking, collision detection, and optimal scheduling of airport ground support equipment, *IEEE Internet of Things Journal*, Vol. 8, No. 1, 572-584, doi:10.1109/JIOT.2020.3004874

[3] Zhao, P. X.; Han, X.; Wan, D. (2021). Evaluation of the airport ferry vehicle scheduling based on network maximum flow model, *Omega*, Vol. 99, Paper 102178, 11 pages, doi:10.1016/j.omega.2019.102178
[4] Cheung, A.; Ip, W. H.; Lu, D.; Lai, C. L. (2005). An aircraft service scheduling model using genetic algorithms, *Journal of Manufacturing Technology Management*, Vol. 16, No. 1, 109-119, doi:10.1108/17410380510574112
[5] Liu, Y.; Wu, J.; Tang, J.; Wang, W.; Wang, X. (2022). Scheduling optimisation of multi-type special vehicles in an airport, *Transportmetrica B: Transport Dynamics*, Vol. 10, No. 1, 954-970, doi:10.1080/21680566.2021.1983484
[6] Zeng, Z.; Jia, L. (2020). Research on airport special vehicle scheduling problem, *2020 International Conference on Virtual Reality and Intelligent Systems*, 1076-1078, doi:10.1109/ICVRIS51417.2020.00262
[7] Feng, X.; Ren, Z. Y. (2016). Collaborative scheduling of fuelling vehicle and ferry vehicle based on genetic algorithm, *Journal of Transportation Systems Engineering and Information Technology*, Vol. 16, No. 2, 155-163, doi:10.16097/j.cnki.1009-6744.2016.02.024
[8] Zhao, P. X.; Gao, W. Q.; Han, X.; Luo, W. H. (2019). Bi-objective collaborative scheduling optimization of airport ferry vehicle and tractor, *International Journal of Simulation Modelling*, Vol. 18, No. 2, 355-365, doi:10.2507/IJSIMM18(2)CO9
[9] Padrón, S.; Guimarans, D.; Ramos, J. J.; Fitouri-Trabelsi, S. (2016). A bi-objective approach for scheduling ground-handling vehicles in airports, *Computers & Operations Research*, Vol. 71, 34-53, doi:10.1016/j.cor.2015.12.010
[10] Feng, X.; Zuo, H.; Sun, Q.; Sun, H. (2021). Research on collaborative scheduling of aircraft ground service vehicles based on simple temporal network, *2021 IEEE 3rd International Conference on Civil Aviation Safety and Information Technology*, 263-269, doi:10.1109/ICCASIT53235.2021.9633418
[11] Chen, B.; Liu, Y.; Yang, Y. (2022). Airport flight transit support time collaborative planning modeling based on STN, *Journal of System Simulation*, Vol. 34, No. 6, 1196-1207, doi:10.16182/j.issn1004731x.joss.20-1057
[12] Kunnapapdeelert, S.; Thawnern, C. (2021). Capacitated vehicle routing problem for Thailand's steel industry via saving algorithms, *Journal of System and Management Sciences*, Vol. 11, No. 2, 171-181, doi:10.33168/JSMS.2021.0211
[13] Liu, M. L.; Yao, X. Z.; Huang, J. Y.; Zhang, C. (2022). Optimization of unmanned vehicle scheduling and order allocation, *International Journal of Simulation Modelling*, Vol. 21, No. 3, 477-488, doi:10.2507/IJSIMM21-3-613
[14] Cappanera, P.; Gouveia, L.; Scutellà, M. G. (2011). The skill vehicle routing problem, Pahl, J.; Reiners, T.; Voß, S. (Eds.), *Network Optimization, INOC 2011, Lecture Notes in Computer Science*, Vol. 6701, Springer, Berlin, 354-364, doi:10.1007/978-3-642-21527-8_40
[15] Dechter, R.; Meiri, I.; Pearl, J. (1991). Temporal constraint networks, *Artificial Intelligence*, Vol. 49, No. 1-3, 61-95, doi:10.1016/0004-3702(91)90006-6
[16] Hunsberger, L. (2002). *Group Decision Making and Temporal Reasoning*, PhD Thesis, Harvard University, Cambridge
[17] Huang, Y.; Jin, C.; Huang, S. (2021). 3D-container loading problem with a distribution plan based on hybrid quantum genetic algorithm, *Economic Computation and Economic Cybernetics Studies and Research*, Vol. 55, No. 4, 117-132, doi:10.24818/18423264/55.4.21.08
[18] Ding, F.; Liu, S.; Li, X. (2022). Pareto optimality of centralized procurement based on genetic algorithm, *Technical Gazette*, Vol. 29, No. 6, 2058-2066, doi:10.17559/TV-20220723180901
[19] Shaw, P. (1998). Using constraint programming and local search methods to solve vehicle routing problems, Maher, M.; Puget, J. F. (Eds.), *Principles and Practice of Constraint Programming – CP98, Lecture Notes in Computer Science*, Vol. 1520, Springer, Berlin, 417-431, doi:10.1007/3-540-49481-2_30
[20] Yang, Y. L. (2020). *Airport Ground Support Modelling and Time Optimization for Flight Turnaround*, Master Thesis, Civil Aviation University of China, Tianjin, doi:10.27627/d.cnki.gzmhy.2020.000147