Hindawi
Journal of Advanced Transportation
Volume 2020, Article ID 8880390, 17 pages
https://doi.org/10.1155/2020/8880390

Research Article

An Improved Adaptive Parallel Genetic Algorithm for the Airport Gate Assignment Problem

Bingjie Liang[1], **Yongliang Li**[2], **Jun Bi**[1,3], **Cong Ding**[1] and **Xiaomei Zhao**[1]

[1]*School of Traffic and Transportation, Beijing Jiaotong University, Beijing 100044, China*
[2]*Information Science & Technology Department, Beijing Capital International Airport Co., Ltd., Beijing 100621, China*
[3]*Key Laboratory of Transport Industry of Big Data Application Technologies for Comprehensive Transport, Beijing Jiaotong University, Beijing 100044, China*

Correspondence should be addressed to Jun Bi; jbi@bjtu.edu.cn

Received 20 June 2020; Revised 27 November 2020; Accepted 10 December 2020; Published 17 December 2020

Academic Editor: Paola Pellegrini

Gate assignment problem (GAP) is the core issue of airport operation management. However, the limited resources of airport gates and the increase of flight scale result in serious problems for gate allocation. In this paper, to provide decision-making support for large-scale GAPs, a model based on gate assignment rules (e.g., flight type constraints, safe time interval constraints, and adjacency conflict constraints) is built to formulate the problem. An improved adaptive parallel genetic algorithm (APGA) is then designed to solve the model. The algorithm is effective because it introduces the idea of elite strategy and parallel design and can adaptively adjust the crossover probability. Moreover, different instances are presented to demonstrate the proposed algorithm. The calculation results of this algorithm are compared with those of standard genetic algorithm and CPLEX, which show that the proposed algorithm has better performance and takes a shorter computational time. In addition, we verify the stability and practicability of the algorithm by repeated experiments on large-scale flight data.

1. Introduction

During the past twenty years, the number of flights in China has increased significantly. The limited resources of airports have gradually become a bottleneck that limits the development of the aviation industry. Therefore, how to dispatch various resources reasonably has become an intractable problem to be solved. Specifically, one of the most critical problems is gate assignment problem (GAP), which deals with the optimal assignment of flights to gates.

With the rapid growth of the number of flights, limited airport resources have been overwhelmed. Among them, the resource shortage of contact stands (an area adjacent to a terminal building where an aircraft can easily be loaded and unloaded) is particularly prominent. According to the International Air Transport Association (IATA) regulations, 90%–95% of all departing passengers should be boarded via jet bridges [1]. However, in China, only about 70–75% of all departing passengers board through the jet bridges. Therefore, in view of the current flight volume, airport contact stands are very scarce in China.

There are two traditional ways to solve the shortage of airport gate resources. First, directly increase the infrastructure and equipment resources, such as expanding the airport apron. However, the airport's infrastructure cannot be expanded indefinitely; moreover, the expansion of the airport and the investment in hardware equipment require a large amount of capital, time, manpower, land, etc., which are restricted by various factors. Second, optimize flight-gate allocation (i.e., improve the utilization efficiency of airport gates and reduce airport operation costs). At present, the gate assignment of large- and medium-sized airports in China mainly relies on manual allocation, supplemented by a computer system. However, in large hub airports, the flight takeoff and landing processes have the characteristics of short time and high density. According to statistics, more than 70% of all flight delays are caused by improper scheduling of airport resources; 15.45% of flight delays are

caused by ground operation delays and departure delays [2]. The quality of gate assignment depends on the experience of operators, and it is difficult to ensure the optimal allocation of flights to gates. Moreover, manual allocation of airport gates contributes to low efficiency and high cost. With the development of airports, the complexity of GAP has increased exponentially. Therefore, an intelligent allocation method is urgently needed.

Based on the above problems, the contributions of this study are as follows. First, an optimization model is established for large-scale gate assignment problems, which comprehensively considers the factors of the remote stand penalty, the travel distance of passengers, and the fuel consumption of taxiing. The airport allocation personnel can adjust the weight of the corresponding objective according to their own preferences or the airport's allocation strategy so that the allocation plan is more in line with the actual allocation needs. Although there are some literatures that consider multiobjective optimization, there are few literatures considering these three aspects comprehensively. And the objective function of the model is quantified as a cost index, which is beneficial for the airport to compare and evaluate the allocation results from an economic point of view. Second, it is clearly noted that GAP is an NP-hard problem. Therefore, an improved adaptive parallel genetic algorithm is proposed to solve the model. The proposed algorithm fully considers how to obtain more initial feasible solutions. In the iterative process, the optimal solution is retained by using elite strategy, and the computational speed of the algorithm is accelerated by using parallel design. Moreover, a case study is presented to demonstrate the proposed algorithm. With the data from Kunming Changshui International Airport, the results of the proposed algorithm are compared with those of traditional genetic algorithm and CPLEX. It indicates that the computational time of the proposed algorithm is shorter than that of traditional genetic algorithm. And the proposed algorithm is still applicable when CPLEX cannot find the exact solution as the scale of the problem increases.

This paper is organized in the following manner. The literature review of GAPs is presented in Section 2. In Section 3, we formally define the problem and build an optimization model of gate assignment problem based on airport gate allocation rules. In Section 4, we propose an improved adaptive parallel genetic algorithm to solve the large-scale GAPs. Section 5 presents different instances to demonstrate and verify the effectiveness and practicability of the proposed algorithm, and Section 6 gives the conclusion.

2. Literature Review

In recent years, as taking off and landing sorties and passenger flows increase rapidly, to solve the shortage of airport gate resources, GAP has attracted a lot of attention. The methods for solving the GAP can be roughly divided into three categories: mathematical programming methods, computer simulation, and heuristic algorithms. The following is an overview of these three types of methods.

The first is the mathematical programming method. Babić et al. [3] solved the gate assignment model by branch and bound method to minimize the travel distance of passengers. Bihr [4] tried to convert the GAP to a 0-1 linear programming problem with the objective of minimizing passenger travel distance. Although the solution is ideal, it also provides some ideas for practical application in the dynamic airport environment. Yan and Huo [5] proposed a multiobjective model to help airport authorities solve the GAP efficiently and quickly. In order to verify the effectiveness of the model, they used the weighted method, column generation method, simplex method, and branch and bound method to solve the model with the actual data from Chiang Kai-Shek (CKS) Airport. Jaehn [6] considered a special case in which the largest flight/gate departure preference score is the only goal. He proposed a dynamic programming method and used actual data from a European airport to verify the effectiveness of the method. Maharjan and Matis [7] proposed a binary integer multicommodity flow network model to balance the transportation efficiency of shuttle bus and passenger satisfaction, which has been applied in Continental Airlines at George W. Bush Intercontinental Airport in Houston (IAH). Jiang et al. [8] proposed a gate assignment model to minimize the total passenger walking distance and balance the passenger walking distance between different routes. Yu et al. [9] converted the robust GAP to a mixed-integer programming problem, which mainly considers three factors: the robustness of flight schedules, facilities and personnel costs during towing, and passenger satisfaction. However, the number of flights to be allocated in the above literature is less than 50, which is far from the actual number of flights that airports need to allocate. When the number of flights that need to be allocated increases, the mathematical programming method often cannot obtain satisfactory results due to the sharp increase of the problem complexity.

The second common method is computer simulation. Baker [10] introduced a rule-based simulation system and evaluated the impact of different rules on gate utilization. Srihari and Muthukrishnan [11] introduced a knowledge-based expert system to GAPs and performed a sensitivity analysis. Cheng [12] proposed a knowledge-based airport gate assignment system to provide a solution that satisfied static and dynamic conditions in a reasonable computational time. Yan et al. [13] proposed a simulation framework, which analyzed the effects of random flight delays on static gate allocation and evaluated fuel consumption time. Finally, simulation experiments were performed at Chiang Kai-Shek airport to evaluate the effectiveness of the framework. Yan and Tang [14] integrated the disturbance of random factors on gate assignment into the framework. The framework includes three components, a random gate allocation model, a real-time allocation rule, and two penalty adjustment methods. However, computer simulation requires researchers to be proficient in computer programming. Moreover, due to the different influencing factors of each airport, it is difficult to directly migrate the simulation system designed for a specific airport to other airports. If

migration is necessary, the simulation system needs to be redesigned, which leads to a high cost.

There are also various studies using heuristic methods to solve GAPs. Mangoubi and Mathaisel [15] used a greedy algorithm to find the initial feasible solution of the gate assignment model, the objective of which is to minimize the total travel distance of passengers. Ding et al. [16] studied the overconstrained GAP, intending to minimize the number of flights at remote stands and the total walking time of passengers. They proposed a hybrid algorithm of simulated annealing and tabu search to solve the model. Dorndorf et al. [17, 18] reviewed the development of the GAPs , and for disruption management in flight-gate scheduling, they proposed two methods to incorporate robustness into a flight-gate assignment problem. Dorndorf et al. [19] considered the general case in which an aircraft may be assigned to different gates. They presented a simple transformation of the flight-gate scheduling (FGS) problem to a graph problem, i.e., the clique partitioning problem (CPP). A heuristic based on the ejection chain algorithm by Dorndorf and Pesch [20] was designed to solve this model. They extended their model to minimize the deviations from a reference schedule. They proposed a heuristic with two variants, which iteratively solve subproblems in order to find a solution for the multiple period problem [21]. They also extended their model to focus on the stochastic objectives, which aim at minimizing the expected number of violations of any kind of constraints. An online decision support system was presented to propose recovery actions for resolving constraint violations. Finally, they compared this model to other approaches and different robustness measures based on real-life test data [22]. Şeker and Noyan [23] and Zhao and Cheng [24] considered that the uncertainty inherent in airport traffic might lead to the occupation of gates allocated to specific flights, which may cause flight conflicts and other problems. Therefore, they set up a mixed-integer programming model and introduced random factors into the model. Tabu search and ant colony algorithms were used to obtain a reasonable allocation scheme with better robustness. Liu et al. [25] proposed an optimization model considering operational security, the objective of which was to minimize the deviation of the gate idle time, and a genetic algorithm was used to solve the model. Dell'Orco et al. [26] proposed a new metaheuristic algorithm which was called fuzzy bee colony algorithm that combines bee colony algorithm and fuzzy inference system to minimize the total passenger travel time and the number of remote stands used. Deng et al. [27] proposed an improved adaptive particle swarm optimization algorithm, which takes full advantage of alpha stable distribution and dynamic score calculation. In order to stably escape the local minima and improve the global search ability, the alpha stable distribution theory is used instead of the uniform distribution. Yu et al. [28] designed an adaptive large neighborhood search algorithm (ALNS) to solve the gate assignment model considering traditional cost and robustness. Liu et al. [29] proposed an optimization model for the GAP considering operational safety constraints, the main objective of which is to minimize the dispersion of gate idle time periods. They adopted genetic algorithm to solve this model, and the effectiveness and efficiency of the algorithm were verified via an illustrative example. Mokhtarimousavi et al. [30] mathematically formulated GAP as a three-objective (total passenger walking distance, taxiway conflicts, and costs) problem, which was solved by NSGA-II. Xu and Cai [31] developed an improved GA considering structural properties to avoid GA's prematurity. They compared the results of the proposed algorithm with those of CPLEX to illustrate the effectiveness of the algorithm. However, heuristic algorithms often cannot obtain optimal solutions, and the convergence speed is closely related to the design of the algorithm.

Although several works have investigated GAPs, few studies have given attention to large-scale GAP. GAP is a complex problem, which is affected by a variety of factors. In order to simplify the problem, various studies only considered the basic constraints and omitted some important constraints (e.g., aircraft type constraints, nation constraints, and adjacency conflict constraints), which cannot meet the actual requirements of airport authorities. Moreover, in the case study of most research works, the number of flights used is less than 50, which is far from the actual number of flights in airports. Therefore, this paper proposes an adaptive parallel genetic algorithm to solve the large-scale GAP. In the beginning, the algorithm increases the search speed to expand the search scope and decreases the search speed when it is close to the optimal solution to improve the accuracy of the solution. Then, on this basis, elite strategy and parallel design are introduced to further improve the accuracy and convergence speed. Finally, based on the data from Kunming Changshui International Airport, we present different instances to verify the stability and effectiveness of the algorithm.

3. Problem Formulation and Model Development

The process of airport gate allocation is restricted by various conditions. In this section, we first briefly introduce the basic prerequisites of the GAP. Then, we describe the model's objective function and constraints.

3.1. Basic Prerequisites. Due to the complexity of the practical problems, almost all gate assignment models need to meet certain preconditions. This paper studies the gate assignment model based on the following assumptions.

Prerequisite 1: data information. The relevant data required for the gate assignment include the following:

(i) *Flight Information*. It includes the planned arrival and departure time of flights, the number of passengers, the aircraft type of the flight, etc.

(ii) *Gate Information*. It includes the topological structure of airport gates, the attributes of each gate (e.g., the type of gate, nation attributes, and default boarding gates), the utilization of current gates, etc.

(iii) *Relevant Parameters of Gate Allocation*. It includes the minimum safe time interval of the same gate, the restriction information of adjacent gates, etc.

Prerequisite 2: gate assignment time window. In the actual operation of the airport, the takeoff and landing of a flight is a continuous and dynamic process. Considering the actual requirement of airports, we use the time window to divide the assignment task in the time dimension (i.e., the allocation of gates is optimal within the current time window). This article sets the length of time window to one day.

Prerequisite 3: airport capacity. We assume that the airport's gate capacity can meet the requirements for gate allocation. That is, for a given flight and gate data, there is at least one feasible solution for the GAP.

3.2. Notations. Before setting up the model, we list the definitions of the parameters and variables to be used in the objective function and constraints, as shown below.

(i) Parameters

F: set of flight pairs to be assigned each day, $F = \{1, 2, 3, \ldots, n_f\}$.
G: set of airport gates, $G = \{1, 2, 3, \ldots, n_g\}$.
n_f: total number of flight pairs to be allocated per day, $n_f = |F|$.
n_g: number of airport gates, $n_g = |G|$.
f_i^{type}: the type of aircraft for flight i, $i \in F$.
f_i^{nation}: international and domestic attributes of flight i. If flight i is a domestic flight, then $f_i^{\text{nation}} = 1$; otherwise, $f_i^{\text{nation}} = 0$, $i \in F$.
g_j^{nation}: international and domestic attributes of gate j. If gate j is a domestic flight, then $g_j^{\text{nation}} = 1$; otherwise, $g_j^{\text{nation}} = 0$, $j \in G$.
g_j^{type}: set of flight type that can be accommodated by gate j, $j \in G$.
p_i^a: number of arriving passengers of arrival flight in flight pair i.
p_i^d: number of departing passengers of departure flight in flight pair i.
ETA_i: estimated arrival time of flight i, $i \in F$.
ETD_i: estimated departure time of flight i, $i \in F$.
T_{buffer}: minimum safe time interval of the same stand.
T_{neighbor}: minimum safe time interval between aircraft sliding in and pushing out in adjacent aircraft stands.
g_j^{bridge}: if there is a jet bridge in gate j, then $g_j^{\text{bridge}} = 1$; otherwise, $g_j^{\text{bridge}} = 0$, $j \in G$.
N_{jk}: if gate j is adjacent to gate k, then $N_{jk} = 1$; otherwise, $N_{jk} = 0$, $j, k \in G, j \neq k$.
Q_{jk}: if gate j and gate k are adjacent, then $Q_{jk} = 1$; otherwise, $Q_{jk} = 0$.
S_1: according to ETA_i, ETD_i, and T_{buffer}, find the flights that will cause conflict when assigned to the same gate and then traverse all flights and save the result in the conflict dictionary S_1.
S_2: according to ETA_i, ETD_i, and T_{neighbor}, find the flights that will cause conflict when assigned to adjacent gates and then traverse all flights and save the result in the conflict dictionary S_2.
s_p: average travel speed of passengers.
s_f: average taxi speed of the aircraft.
d_j^a: passenger travel distance from gate j to baggage claim area, $j \in G$.
d_j^d: passenger travel distance from check-in counter to gate j, $j \in G$.
D_j: average distance from the landing runway to the gate j, $j \in G$.
fuel_i: fuel consumption per taxi unit time of flight i (fuel consumption is related to the type of flight i).
C_1: unit penalty cost of a flight parked at a remote stand.
C_2: unit travel time cost of passengers.
C_3: unit cost of aircraft fuel.

(ii) Decision variables

The decision variable $x_{ij} = 1$ if and only if flight i is assigned to gate j; otherwise, $x_{ij} = 0$.

Given the decision variable x_{ij}, the solution of the gate assignment problem could be transformed as a 0-1 binary matrix, as shown in Table 1. A row of the matrix represents a gate, a column of the matrix represents a flight, and the elements in the matrix denote the values of the decision variables.

3.3. Objective Functions. From the actual operation of airport gate assignment, the assignment process involves the interests of many parties. In this paper, we mainly consider three objectives, including the penalty cost of remote stands, the travel time cost of passengers, and the fuel consumption cost of taxiing [17, 32]:

(i) Generally, when an aircraft is parked at a remote stand, it is necessary to coordinate ground service personnel, apron vehicles, platform lift trucks, passenger elevator vehicles, etc. When the aircraft is parked at a contact stand, it only needs a jet bridge. The use of the jet bridge is very convenient for aircraft, crews, and passengers. There is a ground well near the bridge, which is convenient for refueling. It can also save the APU and fuel consumption of the aircraft, and the passengers and crews do not have to struggle. Based on a flight, according to the type of aircraft, the number of passengers, the time of use, and the related costs of supporting equipment, it is estimated that the aircraft needs at least 400–455 yuan to stop at the remote stand and only 200–300 yuan to stop at the contact stand. Therefore, based on such considerations, we will impose penalties on flights allocated to remote stands, as shown in equation (1).

Table 1: The solution form of the gate assignment problem.

Gate	Flight				
	1	2	3	...	n_f
1	0	1	0	...	0
2	0	0	1		0
3	1	0	0		0
...	...				
n_g	0	0	0		1

(ii) Whether for inbound or outbound flights, the convenience and speed of boarding or baggage retrieval is one of the important indicators for passengers to measure the airport service level. Therefore, in order to improve passengers' satisfaction with airport services, airports should assign flights to the nearby stands to reduce passenger travel distance as far as possible. As shown in equation (2), since there are no data of transit passengers, we do not consider the travel distance of transit passengers here.

(iii) As the aviation transport industry is a capital-intensive industry with an average profit margin of only 3%~6%, reducing production costs is of great significance for maintaining the survival and development of enterprises. The cost of aircraft fuel usually accounts for about 30% of all operating costs of airlines. In recent years, due to the continuous rise of international oil prices, airlines are facing increasing pressure on fuel consumption costs year by year. Therefore, reducing fuel consumption as much as possible and saving transportation costs have always been the issues that airlines are most concerned about and strive to solve. In the GAPs, the selection of gates directly determines the ground taxiing distance of the aircraft from the quick exit to the parking gate, which determines the ground taxiing cost of the aircraft. Therefore, the gate assignment scheme has a direct impact on the ground taxiing cost of aircrafts.

Considering the processing of the follow-up algorithm and the different preferences of different personnel, we make a weighted summation of these three objectives. As shown in equation (4), α_1, α_2, and α_3 are the preference coefficients of allocation personnel, which are used to indicate the importance of different objectives.

$$\min Z_1 = C_1 \sum_{j=1}^{n_g} \sum_{i=1}^{n_f} x_{ij}\left(1 - g_j^{\text{bridge}}\right), \tag{1}$$

$$\min Z_2 = C_2 \frac{\sum_{j=1}^{n_g} \sum_{i=1}^{n_f} \left(x_{ij} p_i^a d_j^a + x_{ij} p_i^d d_j^d\right)}{s_p}, \tag{2}$$

$$\min Z_3 = C_3 \frac{\sum_{j=1}^{n_g} \sum_{i=1}^{n_f} x_{ij} \text{fuel}_i D_j}{s_f}, \tag{3}$$

$$\min Z = \alpha_1 Z_1 + \alpha_2 Z_2 + \alpha_3 Z_3. \tag{4}$$

3.4. Constraints. Due to the differences in airport positioning, service area, and scheduling rules, there are certain differences in the constraints of different airports. However, there are a lot of allocation rules that almost all airports need to follow when allocating gates, only slightly different in specific details. According to the actual business rules of large hub airports, the aircraft stand allocation model has the following constraints:

(i) Uniqueness constraint: an aircraft must and can only park at one stand.

$$\sum_{j=1}^{n_g} x_{ij} = 1, \quad \forall i \in F. \tag{5}$$

(ii) Conflict constraint of the same gate: there shall be a safe time interval between two consecutive aircraft assigned to the same gate to guarantee the safe departure of the former aircraft and the safe entry of the latter.

$$x_{ij} + x_{lj} \le 1, \quad \forall i, l \in S_1, \forall j \in G. \tag{6}$$

(iii) Conflict constraint of adjacent gates: the aircraft parked on the adjacent aircraft stand cannot enter and leave the aircraft stand at the same time. There should be a certain time interval when the aircraft slides in and out.

$$x_{ij} + x_{lk} \le 1, \quad \forall i, l \in S_2, \forall Q_{jk} = 1. \tag{7}$$

(iv) Aircraft type restriction: large aircraft stands can park all types of aircraft, while small aircraft stands can only park corresponding small aircraft.

$$f_i^{\text{type}} \in g_j^{\text{type}}, \quad \forall x_{ij} = 1, \forall i \in F, \forall j \in G. \tag{8}$$

(v) International and domestic attribute constraints: as flights are divided into international flights and domestic flights, gates are also divided into corresponding attributes. According to the relevant regulations of the airport, international flights can only park in international gates, while domestic flights need to stop in domestic gates.

$$f_i^{\text{nation}} = g_j^{\text{nation}}, \quad \forall x_{ij} = 1, \forall i \in F, \forall j \in G. \tag{9}$$

4. Design of Improved Adaptive Parallel Genetic Algorithm

The model established in this paper is a mixed-integer programming model, and the large-scale GAP involves hundreds of flights, which is difficult to be solved by the traditional optimization algorithm in polynomial time. Therefore, we consider using the genetic algorithm to solve

the model. Through an in-depth analysis of the model, the traditional genetic algorithm is improved in this section. We design an improved adaptive parallel genetic algorithm (APGA) that considers global and local search capabilities.

4.1. Encoding Approach. Chromosomes are encoded by digital code. In order to explain the encoding mode of chromosomes better, sample data are used for illustration. Sort the flight pair data according to the estimated departure time, as shown in Table 2.

Each flight pair is then assigned to a gate, as shown in Table 3. Gate no. in the table is only used to explain the design of chromosome encoding and does not represent the actual allocation.

According to the above encoding principles, the chromosome encoding sequence is shown in Figure 1. The length of the chromosome is the number of flights, that is, the length of the chromosome is determined by the number of paired flights. The gene loci on the chromosome represent the flights to be assigned, and the numbers on the loci indicate the gate no. For example, paired flight no. 0 is assigned to gate no. 0; paired flight no. 1 is allocated to gate no. 2, and so on.

4.2. Generation of Initial Population Based on CSP. In the process of gate assignment, flights are mutually restricted. Whether a flight can be parked is restricted by the idle status of the gate, and the availability of the gate is determined by the assigned flights before and after. Therefore, in the process of chromosome encoding, the value of a gene is affected by the value of the gene before it. In addition, the set of possible gate allocations for each flight is subject to other restrictions such as aircraft type and the international/domestic attributes of gates, and so on. These effects are determined by the constraints in the gate assignment model. Only when the genetic code of the chromosome meets the constraints can the chromosome represent a feasible solution.

In the process of generating the initial population and each generation of population, if a chromosome is generated randomly, the probability of the chromosome being a feasible solution is very small, which leads to a sharp reduction in the efficiency of the algorithm. Therefore, how to generate a feasible solution to the GAP is crucial.

4.2.1. CSP Problem. Constraint satisfaction problem (CSP) is an important branch of artificial intelligence research for many years. Many problems can be modeled as CSP, such as vision and resource allocation, temporal reasoning, and so on.

The CSP can be defined as a tuple X, D, C, where $X = \{x_1, x_2, \ldots, x_n\}$ is a set of variables, $D = \{D_1, D_2, \ldots, D_n\}$ is a nonempty set of definition fields for each variable, and $C = \{C_1, C_2, \ldots, C_n\}$ is a set of constraints. x_i can choose a suitable value from the nonempty D_i. The solution of CSP is to find a set of values that satisfy the constraints to solve a given problem.

Table 2: Flight pair information sorted by estimated departure time.

Paired flight no.	Flight no.	Estimated time of arrival	Estimated time of departure	Aircraft type
0	B1378	00:00	00:10	C
1	B2835	00:00	00:15	C
2	B6832	00:05	01:10	C
3	XU997	01:15	2:15	C
4	9MAGD	01:25	02:30	D
5	HSCBA	02:00	02:55	C
6	B7992	00:00	06:20	C
...	...	...	...	...

Table 3: An example of flight pair assignment.

Paired flight no.	Gate no.
0	0
1	2
2	6
3	4
4	7
5	3
6	4
...	...

4.2.2. Initial Population. The generation of the initial population is based on the idea of CSP. In the GAP, X represents a daily flight set, x_i represents the gate that flight i assigned to (i represents the flight number in the model, $i \in F$), D_i represents a gate set that flight i can be assigned to, and $C = C_1, C_2, \ldots, C_n$ are the constraints of allocation. The main steps are as follows:

Step 1. Initialization of paired flights and stands: according to the departure time of the paired flights, all flights are sorted from morning to night, and n_f pairs are numbered as 0, 1, 2, ..., $n_f - 1$. Similarly, the n_g stands are also numbered, with the contact stands first and the remote stands next. According to the arrival time and departure time of each flight pair, for flight pair i:

(i) In the case of the same gate, we need to find the set of flight pairs that conflict with flight pair i and save it to the dictionary of the same gate conflict.

(ii) In the case of adjacent gates, we need to find the set of all flight pairs that conflict with flight pair i and save it to the dictionary of adjacent gates conflict. By traversing all flight pairs, we can get two conflict dictionaries: one is the dictionary of the same gate conflict and the other is the dictionary of adjacent gates conflict. The form of the dictionary is like {flight pair i: the set of flight pairs that conflict with flight pair i, such as {j, k, ...}, ...}.

Step 2. Since the optional gate set D_i of the flight i to be assigned is affected by the assigned flight pairs, the optional gate set D_i is constantly changing according to the assigned flights. We need to judge the flight pairs allocated before:

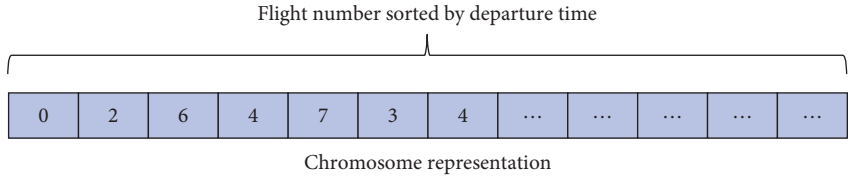

Figure 1: Schematic diagram of chromosome encoding.

(i) We find flight pairs that conflict with flight i which is numbered less than i in the dictionary of the same gate conflict. According to the conflict flight number, we get the number of the gate where the conflict flight is assigned. Then, we delete the corresponding gate number in the set D_i.

(ii) We find flight pairs that conflict with flight i which is numbered less than i in the dictionary of adjacent gates conflict. According to the conflict flight number, we get the number of the stand where the conflict flight is assigned. Then, we delete the corresponding gate and its adjacent gates in the set D_i.

Step 3. Randomly select a gate j from the optional gate set D_i and assign flight i to the gate j. If the set D_i is an empty set, then discard the chromosome and repeat Steps 2 and 3.

Step 4. If the population size is M, we repeat the above process until the number of chromosomes generated reaches M.

4.3. Fitness Calculation. In order to ensure the convergence speed of the algorithm, the fitness is calculated according to the following rules:

(i) Constraint conflict: if the allocation scheme conflicts with the constraints during the calculation, the fitness is 0.

(ii) No constraint conflicts: in general, fitness function is transformed from objective function. Since the objective function is to find the minimum value, we use the bound construction method to construct its upper bound. As shown in equation (10), the function value is always greater than 0 and increases with the decrease of the objective function, which meets the requirements of the fitness function.

$$\text{fitness} = C_{\max} - Z, \tag{10}$$

where $C_{\max}$ is the maximum estimate of the objective function Z.

4.4. Selection Method. Roulette wheel selection strategy is one of the most basic selection strategies, which is used in this study. The probability that everyone in the population being selected is proportional to the value of the corresponding fitness function of the individual. The fitness values of all individuals in the population are accumulated and normalized, and finally random numbers are generated to select the individuals corresponding to the region where the random numbers are located, like a rotating roulette in a casino. The basic idea is that the higher the fitness, the greater the probability that the individual will be selected. Since the roulette wheel selection strategy is simple and easy to use, it is a selection operation often used in genetic algorithms [33].

4.5. Crossover Operator

4.5.1. Adaptive Cross Probability Design. The selection of the crossover probability P_c has a direct impact on the convergence speed of the algorithm. The larger P_c is, the faster the new individuals are generated. However, when P_c is too large, the structure of the chromosome with greater fitness is more likely to be destroyed. Too small P_c will lead to a slow search process and reduce the search efficiency of the algorithm. Therefore, this paper uses an adaptive crossover probability. The calculation formula of P_c is as follows:

$$P_c = \begin{cases} P_{c1} - \dfrac{(P_{c1} - P_{c2})(f' - f_{\text{avg}})}{f_{\max} - f_{\text{avg}}}, & f' \geq f_{\text{avg}}, \\ P_{c1}, & f' < f_{\text{avg}}, \end{cases} \tag{11}$$

where f' is the larger fitness value of the two individuals to be crossed and f_{avg} is the average fitness value of each generation. When $f' < f_{\text{avg}}$, a fixed crossover probability is used; when $f' \geq f_{\text{avg}}$, an adaptive crossover probability is used. According to formula (11), the individuals whose fitness is lower than the average fitness of the population belong to the poor individuals in the population. Therefore, a large crossover probability is adopted to carry out a crossover operation to promote gene recombination and gene variation. Individuals whose fitness is higher than the average fitness of the population belong to the better individuals in the population, and the corresponding crossover probability is calculated according to equation (11). The larger the individual fitness value, the smaller the crossover probability value. When the individual fitness value is the largest, the crossover probability is the smallest. However, due to the initial stage of population evolution, the better individuals in the population are not definitely the global optimal solution. If this adaptive crossover probability adjustment method is used, it is easy for the algorithm to fall into a local optimum and mature prematurely.

Therefore, in the first n generations of the genetic algorithm, a large fixed cross probability is used to perform the crossover operation to expand the search range and maintain the population diversity. However, due to the slow

convergence speed of the simple genetic algorithm, in order to increase the convergence speed of the algorithm, this paper adopts an adaptive strategy to adjust the crossover probability in the later iteration of the algorithm.

4.5.2. Design of Chromosome Crossover Method. This paper uses a two-point crossover operator. First, pairwise operations are performed on individuals in the population. Then, according to the contemporary individual crossover probability, we determine whether to perform crossover operations. For the two parent chromosomes that need to be crossed, the two crossing points of the crossing operation are randomly generated. It is assumed that the selected crossing points are points 1 and 2. Finally, the two random crossing points (point 1 and point 2) exchanged their gate number. The specific steps are shown in Figure 2.

4.6. Mutation. Since the crossover operations on the parent chromosomes in the population are randomly selected, the offspring individuals generated after crossover are likely to become infeasible solutions because they do not meet the constraints of the model established in this paper. Therefore, these chromosomes are mutated by mutation operation to make them feasible solutions. The operation process is as follows: for the crossed offspring chromosomes, all the genes are checked sequentially. If the gene satisfies the constraints in the model in this paper, the value of the gene at that position does not change. If it is not satisfied, the flight corresponding to the position gene is reassigned according to the method of single chromosome generation in the initial population generation, and the allocation result is reflected in the chromosome code.

Take the offspring 1 in Figure 2 after the crossover as an example. As shown in Figure 3, the crossover starting point is 3, and the coding of the third and subsequent genes is checked in order. The third flight is assigned to gate no. 4. Since no other flight has been assigned to stop at gate no. 4 before, the gene location meets the constraint conditions and retains the gene value of the location. Next, check the fifth flight, which also meets the constraint conditions in the model, until the seventh flight. Judging from the conflict dictionary of the same gate in the initial population generation, there is a conflict with the fifth flight stopped at gate no. 3, so the flight needs to be reassigned. According to the conflict dictionary of the same gate and adjacent gate, the set of optional gates can be obtained, and then a gate is randomly selected from the set of optional gates to the flight. The randomly selected gate in Figure 3 is gate no. 9.

4.7. Elite Strategy. In order to prevent the loss of the optimal individuals of the current population in the next generation, which leads to the failure of the genetic algorithm to converge to the global optimal solution, de Jong [34] proposed the elitism strategy, also known as the elitist reservation strategy, in his doctoral dissertation. The best individuals that have appeared in the population so far in the evolution process are copied directly to the next generation without genetic manipulation, and they will generally replace the worst individuals in the next generation. Elite retention strategy improves the global convergence ability of the standard genetic algorithm.

4.8. Parallel Design. The general genetic algorithm is a single population design. In the early stage of population evolution, it tends to the direction of objective function optimization. However, when the algorithm is iterated to the later stage, the algorithm's optimal individual in the population hardly changes, resulting in the poor global search ability. Moreover, the traditional genetic algorithm tends to suffer from premature convergence for many problems and easily falls into a local optimum. Based on the analysis of the causes of these phenomena, we adopt a multipopulation design similar to the multi-island genetic algorithm. Multi-island genetic algorithm is a parallel genetic algorithm based on population grouping, which is developed from traditional genetic algorithm. The difference between this algorithm and traditional genetic algorithm is that the multi-island genetic algorithm divides the entire population into several subgroups and isolates the subgroups from each other on different "islands." The "migration" operation is carried out at a certain time interval to enable the exchange of information among the "islands." In response to these shortcomings, the introduction of multiple swarm strategies can not only maintain population diversity but also reduce the possibility of immature, which is conducive to the quality of solutions.

The difference from the traditional parallel island-based genetic algorithm is that we have meticulously divided the role of each population, and each subpopulation has its own specific role. Furthermore, the idea of adaptive parameter adjustment is integrated, which not only maintains the stability of the evolutionary process but also maintains the diversity of individuals. As shown in Figure 4, the characteristics of development subpopulations are that their crossover probability is relatively small, so it is easier for GA to maintain the stability of individuals, and it is easier to find excellent individuals in a local range (in a certain hyperplane). The development subpopulations play a prominent role in protecting good individuals. On the contrary, large cross probability of the probe subpopulation make it easier for APGA to detect new hyperplanes, thereby increasing the probability of detecting the optimal individuals. The role of this subpopulation is to continuously provide new hyperplanes to overcome premature convergence. The crossover probability of the probe and development subpopulation is between the two groups. Its main function is to take into account the local and global and make up for the deficiency of the development subpopulation and probe subpopulation. The last subpopulation is the reserved subpopulation, which has no individuals at first, and is composed of excellent individuals selected in the evolution of the first three types of populations. Its function is to preserve the outstanding individuals that evolve in the first three types of populations, and at the same time, it is also evolving, and its crossover probability is relatively small, whose purpose is to maintain the stability and diversity of individuals. The first three types of populations evolve in parallel according to their own evolutionary strategies. At the same time, in order to maintain the diversity of individual distribution, individuals must migrate between these three types of populations so that

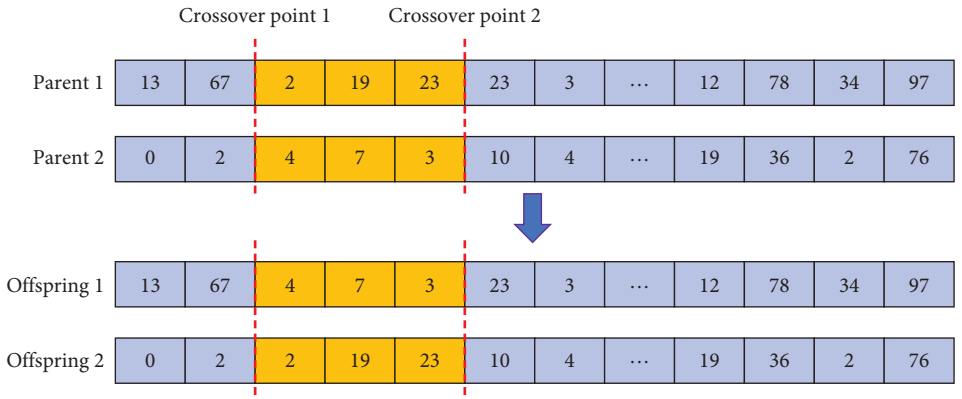

Figure 2: Schematic diagram of the chromosome crossover process.

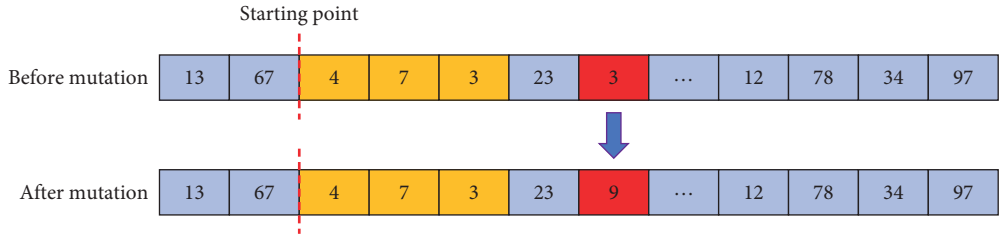

Figure 3: Schematic diagram of chromosome mutation operation.

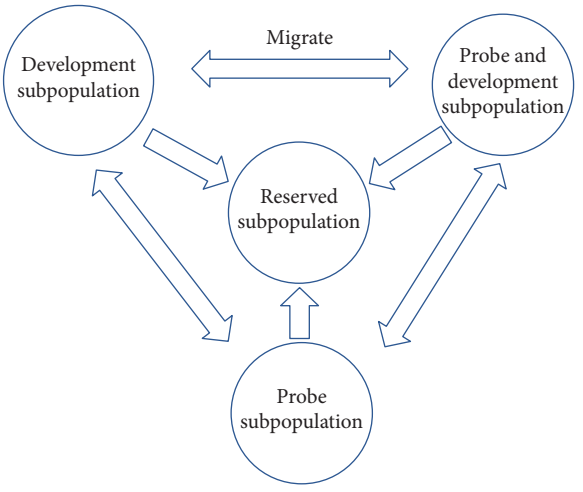

Figure 4: Schematic diagram of the parallel design.

they can absorb the advantages of other populations and overcome individual trends. It is also necessary to regularly select the best individuals from the first three types of subpopulations and classify them into the fourth type of subpopulations so as to protect excellent individuals from damage and to absorb the advantages of the first three types of subpopulations and maintain the diversity of individuals.

4.9. Algorithm Flow. The main flow of the algorithm is as follows:

Step 1. Chromosome encoding: flights are encoded according to the departure time. The gene positions on chromosomes represent flights, and the gene values represent the assigned airport gates.

Step 2. Parallel design: relevant parameters of the subpopulation were set, and each subpopulation carried out migration between the populations after m times of isolation, and the optimal individuals were copied to the reserved subpopulation. The maximum number of migrations is set to n.

Step 3. Generation of initial populations: the idea of constraint satisfaction problem is introduced, and a feasible solution that satisfies the constraints is generated based on the flight conflict dictionary of the same gate conflict and the adjacent gate conflict.

Step 4. Fitness calculation: calculate the fitness of each individual based on equation (10).

Step 5. Selection operator: roulette is adopted and elite strategy is introduced to retain the best individuals.

Step 6. Crossover operator: two different crossing points are randomly generated to cross-pair segments between chromosomes.

Step 7. Mutation operator: because the random crossover produces the infeasible solution, the mutation is used to turn it back into a feasible solution.

Step 8. Evolution: repeat Steps 4–7 so that the population keeps evolving.

Step 9. Termination condition: the iteration reaches the maximum number of migrations n or the population will automatically jump out of the loop if it does not evolve for k consecutive times.

Step 10. Output: output the optimal population and fitness of the allocation scheme.

5. Case Study

5.1. Background. In this section, to verify the effectiveness and practicability of the model and algorithm proposed in this paper, the actual data from Kunming Changshui International Airport, one of China's top ten airports, are used for the case study. Its basic information is shown in Table 4.

Table 4: Basic information of Kunming Changshui International Airport.

Attribute name	Content
Name	Kunming Changshui International Airport
Airport code	ICAO:ZPPP; IATA:KMG
Navigation date	June 28, 2012
Airport type	4F civil transportation airport
Terminal area	548,300 square meters
Number of gates	110
Passenger throughput	44.73 million person-times (2017)
Cargo throughput	419,000 tons (2017)
Takeoff and landing flights	305,300 sorties (2017)

Test environment: Intel (R) Core (TM) i7-8750H CPU 2.20 GHz 2.21 GHz computer.

Table 5: Basic information of airport gates.

Gate_no	Mdl	Nation	Bridge
0	C	I	1
1	C	I	1
2	C	I	1
3	D	I	1
4	D	I	1
5	D	I	1
6	D	I	1
7	D	I	1
8	F	I	1
9	E	I	1
10	E	I	1
11	D	I	1
12	D	I	1
13	C	D	1
14	C	D	1
15	C	D	1
16	C	D	1
17	C	D	1
...			
193	C	D	0
194	E	D	0
195	C	D	0
196	C	D	0
197	E	D	0

5.2. Raw Data Overview. In this section, we briefly introduce the raw data. The data used in the instances of the following sections are all a subset of the raw data. The raw data used in the case study are from Kunming Changshui International Airport on November 23, 2019. Table 5 presents the basic information of airport gates. The first column "Gate_no" represents the number of the gate, and "Mdl" represents the largest aircraft type that the gate can accommodate. The value of nation is "I," indicating that the gate can stop international flights, and "D," indicating that the gate can stop domestic flights. The column "Bridge" indicates whether there is a jet bridge in the gate. If the value is 1, it means there is a jet bridge, and if the value is 0, it means there is no jet bridge. The number of contact stands equals 65, and the number of remote stands equals 133. About one-third of the stands are contact stands, including 13 international stands.

According to the flight data of Kunming Changshui International Airport on November 23, the total number of flights is 381. The data samples are shown in Table 6. As the flight no. of inbound and outbound flights may change, the aircraft number is used to represent the paired flight in the table, and the nation represents the international and domestic attributes of inbound and outbound flights. "Atime" indicates the arrival time of the flight, and "Dtime" indicates the departure time. Mdl represents the type of flight. The last two columns "Apassenger" and "Dpassenger" indicate the number of passengers arriving and departing from the airport. According to the different types of aircrafts, the fuel consumption per unit time of taxiing is also different [32], as shown in Table 7. Table 8 shows the value of parameter d_j^a corresponding to different gates, indicating the distance from different gates to baggage claim area. The first column "Gate_no" indicates the gate number, and the second column "$d_j^a(m)$" indicates the distance. Table 9 shows the value of parameter d_j^d corresponding to different gates, indicating the distance from check-in counter to different gates. The first column "Gate_no" indicates the gate number, and the second column "$d_j^d(m)$" indicates the distance. Table 10 shows the value of parameter $D_j(m)$ corresponding to different gates, indicating the distance from landing runway to different gates. The first column "Gate_no" indicates the gate number, and the second column "$D_j(m)$" indicates the distance.

The statistics of the selected flights are divided into different periods below, and the distribution of inbound and outbound flights in each period is calculated. As shown in

Table 6: Flight pair data samples of Kunming Changshui International Airport on November 23, 2019.

Flight no.	Nation	Mdl	Atime	Dtime	Apassenger	Dpassenger
XU997	I/I	C	2019-11-23 1:15	2019-11-23 2:15	79	167
9MRAG	I/I	C	2019-11-23 1:25	2019-11-23 2:30	120	120
HSBBB	I/I	C	2019-11-23 2:00	2019-11-23 2:55	120	120
PKLAM	I/I	C	2019-11-23 2:10	2019-11-23 3:10	120	140
PKGTF	I/I	C	2019-11-23 2:45	2019-11-23 3:45	120	157
B6176	D/D	C	2019-11-23 0:35	2019-11-23 6:25	127	112
B307U	D/D	C	2019-11-23 0:10	2019-11-23 6:30	150	89
B1330	D/D	C	2019-11-23 0:10	2019-11-23 6:40	139	134
B1593	D/D	C	2019-11-23 0:30	2019-11-23 7:00	174	189
B6943	D/D	C	2019-11-23 0:05	2019-11-23 7:05	159	176
...						
B6743	D/D	C	2019-11-23 22:15	2019-11-23 23:20	152	164
B6956	D/D	C	2019-11-23 22:15	2019-11-23 23:20	148	142
B5825	D/D	C	2019-11-23 22:20	2019-11-23 23:20	142	138
B6016	D/D	C	2019-11-23 22:15	2019-11-23 23:30	87	159
B6728	D/D	C	2019-11-23 22:25	2019-11-23 23:30	163	173
B5475	D/D	C	2019-11-23 22:25	2019-11-23 23:30	150	146
B5267	D/D	C	2019-11-23 21:45	2019-11-23 23:40	91	121
B1459	D/I	C	2019-11-23 21:20	2019-11-23 23:45	123	109
B5823	D/D	C	2019-11-23 22:55	2019-11-23 23:50	132	126

Table 7: Fuel consumption per unit time of taxiing for different types of aircraft.

Mdl	Fuel consumption per unit time of taxiing (kg/min)
C	11.5
D	16
E	25
F	35

Table 8: Passenger travel distance from gate j to baggage claim area.

Gate_no	$d_j^a(m)$
0	550
1	704
2	575
3	690
...	...
192	2607
193	2342
194	1497
195	2449
196	1407
197	2922

Figure 5, a great number of flights enter the airport from 0:00 to 1:00, and there are almost no flights leaving the airport from 5:00 to 10:00. The remaining flights are distributed between 10:00 and 19:00.

5.3. Parameters. The quality of the parameter setting is closely related to the reliability and efficiency of the proposed algorithm. The traditional genetic algorithm has four parameters that need to be set in advance, including population size, termination evolution algebra, crossover probability, and mutation probability. As for the adaptive parallel genetic algorithm we proposed, we consider using a response surface methodology (RSM) to adjust the parameters. The experimental random error is considered in the response surface analysis. RSM is an effective method to solve practical problems by fitting complex unknown functional relations in a small area with a simple polynomial model. And the prediction model is continuous.

Table 9: Passenger travel distance from check-in counter to gate j.

Gate_no	$d_j^d(m)$
0	357
1	460
2	392
3	457
4	453
...	...
193	2252
194	1422
195	2360
196	1309
197	2835

Table 10: Average distance from the landing runway to the gate j.

Gate_no	$D_j(m)$
0	4062
1	3320
2	5290
3	2215
4	3206
...	...
193	3744
194	4511
195	5260
196	2670
197	4925

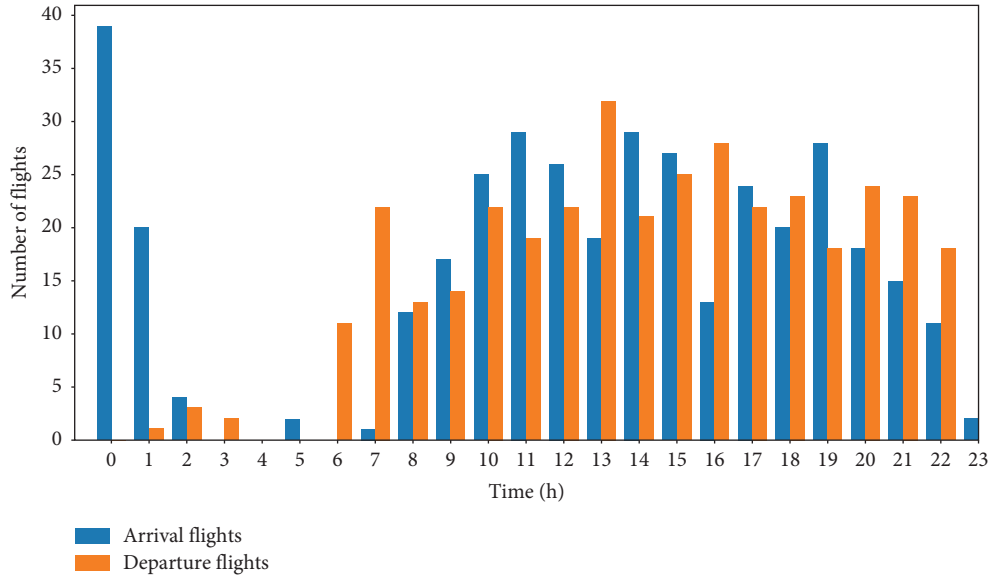

Figure 5: Schematic diagram of flight arrival and departure time.

Compared with the orthogonal experiment, its advantage is in the process of optimizing the experimental conditions, it can continuously analyze all levels of the experiment, while the orthogonal experiment can only analyze the isolated experimental points. RSM is also used in parameter optimization of genetic algorithm, which can greatly reduce the number of experiments required for parameter optimization [35]. Before parameter adjustment, we consider determining the parameters with less influence to reduce the scale and improve the efficiency of parameter adjustment. The parameters to be adjusted are analyzed as follows:

(i) According to the adaptive crossover probability formula (11), the parameters include P_{c1} and P_{c2}, where P_{c2} mainly plays an auxiliary role in adaptive adjustment. To improve the efficiency of parameter adjustment, we set it to 0.2 according to experience and only adjust P_{c1}.

(ii) In the process of mutation after crossover, if an individual is an infeasible solution, the mutation operation is performed to change it into a feasible solution; otherwise, no mutation is performed. Therefore, there is no need to adjust the mutation probability.

(iii) For the population size, it is difficult to set a good value empirically, so we adjust the population size from 20 to 200.

(iv) In Section 4.8, the parallel structure is designed as three subpopulations, and their respective roles are analyzed. Therefore, no parameter adjustment is required.

(v) After the algorithm runs for a period, the optimal solution is no longer improved. After that, the number of iterations only affects the efficiency of the algorithm. According to experiments, the number of iterations to obtain the optimal solution is less than 100. Just in case, we adjust the number of termination iterations from 50 to 500.

Next, we need to determine the dataset for parameter adjustment. We use equal probability sampling to randomly select 100 flights from the raw data of Yunnan Kunming Airport on November 23, 2019, for parameter adjustment. Yunnan Kunming Airport has 198 gates, with more than 300 flights passing through the airport every day. Since the number of selected flights is 100, to simulate the actual situation of the gate assignment, we limit the allocation of flights to the first 65 gates and regard the 35 contact stands numbered 30–64 as remote stands. In addition, the parameters of the model need to be determined. We set the passenger's average travel speed s_p as 1.25 m/s [36], the taxi speed of the aircraft s_f as 20 km/h, C_1 as 200 yuan per flight, C_2 as 50 yuan/h, and C_3 as 7000 yuan/ton [32]. After that, we use the RSM to find the optimal parameters.

Table 11 shows the low and high levels of population size, crossover rate, and generation number. Table 12 shows the design of experiments, detailing the real values of population size, crossover rate, and generation number for each experiment and the sequence of each experiment. The last column "Result" represents the objective function value of the model, and the specific calculation process is shown in equation (4). This article treats the three goals Z_1, Z_2, and Z_3 as equally important, that is, α_1, α_2, and α_3 are all set to 1. In order to eliminate the influence of accidental factors, we conducted five repeated experiments under the same parameter setting and finally filled the average value of the five results into the table.

The polynomial equation model is obtained by fitting the experimental data through regression analysis, as shown in the following equation:

Table 11: Levels and values of the proposed GA parameters.

Parameter	Symbol	Low level	High level
Population size	P_s	20	200
Crossover rate	P_{c1}	0.4	0.95
Generation number	G_N	50	1000

Table 12: Experiment design.

Experiment no.	Run order	P_{c1}	P_s	G_N	Result (yuan)
1	1	0.675	110	525	200524
2	2	0.675	110	525	202492
3	3	0.95	110	50	272876
4	4	0.675	20	50	332529
5	5	0.4	110	1000	198007
6	6	0.675	20	1000	207099
7	7	0.95	110	1000	197055
8	8	0.4	110	50	264583
9	9	0.95	20	525	215530
10	10	0.4	20	525	217211
11	11	0.675	110	525	201413
12	12	0.4	200	525	197447
13	13	0.675	200	1000	200480
14	14	0.675	110	525	200305
15	15	0.675	200	50	274609
16	16	0.95	200	525	198958
17	17	0.675	110	525	200742

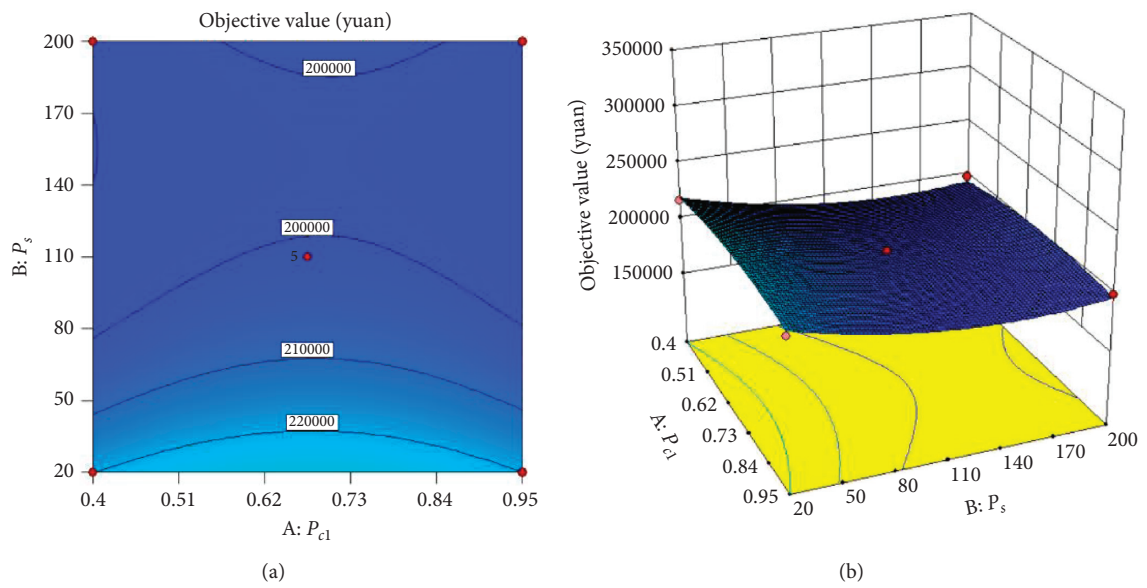

Figure 6: The response surface plot and the corresponding contour plot showing the effects of crossover rate and population size on fitness.

$$\begin{aligned}\text{Fitness} = {} & 2.99625 * 10^5 + 1.37153 * 10^5 * P_{c1} \\ & - 682.51075 * P_s - 293.53704 * G_N \\ & + 32.24242 * P_{c1} * P_s - 17.69378 * P_{c1} * Gn \\ & + 0.30001 * Ps * Gn \\ & - 94926.94214 * {P_{c1}}^2 + 1.65064 * {P_s}^2 \\ & + 0.17380 * Gn^2,\end{aligned} \tag{12}$$

where P_{c1} is the crossover rate, P_s is the population size, and G_N is the generation number.

As can be seen from Figure 6, the response surface is a three-dimensional plane, so the interactions between population size, crossover probability, and generation number have a negligible effect on fitness. According to the response surface analysis, the minimum predicted value of fitness is 181470, which is taken when the population size, crossover rate, and generation number are 126, 0.95, and 1000, respectively.

Table 13: Comparison of results.

Instance	Flight	Gate		CPLEX		GA			APGA		
		Contact stands	Remote stands	Result (yuan)	Computation time (s)	Result (yuan)	Computation time (s)	Gap (%)	Result (yuan)	Computation time (s)	Gap (%)
1	20	7	5	42800	4.36	44334	52.28	3.58	43247	27.73	1.04
2	40	10	8	88666	10.14	92354	111.07	4.16	90116	47.42	1.64
3	60	13	11	137267	29.75	145291	171.76	5.85	141517	70.95	3.10
4	80	16	14	194862	70.42	205806	224.04	5.62	197295	91.43	1.25
5	100	21	19	232502	195.78	250419	271.98	7.71	234823	115.55	1.00
6	120	24	22	271627	522.24	311468	324.92	14.67	271859	138.72	0.09
7	140	28	26	328878	1225.16	348203	357.66	5.88	336940	141.20	2.45
8	160	32	30	354699	2765.36	378952	428.26	6.84	357426	163.64	0.77
9	180	36	34	None	None	436359	464.89	None	422863	184.43	None
10	200	40	38	None	None	483526	543.20	None	469921	204.65	None

5.4. Multiscenario Design. In multiscenario design, we removed 100 flights for parameter adjustment from the original data, leaving 281 flights. Then, we use the equal probability sampling method to randomly generate 10 sets of data from the remaining flight sets as instances to verify the performance of the algorithm. Each dataset contains different amounts of flights and gates, which are used to verify the performance of APGA, GA, and CPLEX on different data scales. Based on the results in Section 5.3, the parameters of the proposed algorithm are set as follows: population size is 126, termination evolution algebra is 1000, P_{c1} is 0.95, and P_{c2} is 0.2. We compare it with the traditional genetic algorithm and CPLEX. The results are shown in Table 13.

Table 13 shows the comparison results for 10 instances. The first column in the table indicates the serial number of the instance, the second column indicates the number of flights that need to be allocated, the third column mainly indicates the number of contact stands and remote stands used in the instances, and the remaining columns represent the calculation results of CPLEX, traditional genetic algorithm, and APGA, respectively. The "Gap" column of GA and APGA represents the gap percentage from CPLEX best value. Firstly, we can see that the proposed algorithm has higher efficiency and better results than the traditional genetic algorithm. The gap of GA is mostly above 5%, while the gap of APGA is mostly below 3%. For different instances, APGA performs better than GA. Secondly, compared with the calculation results of CPLEX, the results obtained by the proposed algorithm are remarkably close to the optimal solution and can meet the actual demands of the airport authorities. Thirdly, when the number of flights is small, we can see that the efficiency of CPLEX is significantly higher. With the increase of the number of flights to be allocated, the solution time of CPLEX increases sharply. When the number of flights is 160, the time required to solve the problem has reached 2765.36 seconds. When the number of flights reaches 180, CPLEX can no longer get results in a limited time. However, the APGA algorithm we designed can still get satisfactory results in a limited time, so APGA has better performance for large-scale GAPs.

We select the result of instance 10 with 200 flights for further analysis. First, its convergence curve is shown in Figure 7. It can be seen from the figure that as the iterative process continues, the convergence rate of algorithm gradually slows down, and the fitness of the optimal individual of the population is continuously optimized. After 500 iterations, the curve tends to be stable, and the optimal solution did not change significantly. Second, Figure 8 shows the Gantt diagram of the result. The horizontal axis of the Gantt chart represents time, while the vertical axis of the Gantt chart represents the number of aircraft stands. Each rectangle in this figure represents an assigned flight pair (aircraft). And the aircraft number which is marked on the corresponding flight pair is used to indicate the allocated aircraft. We can see that there are few planes parked on gates numbered 0–12. This is because these gates are all international gates where domestic flights cannot park. In addition, we find that there are some unused gates in the Gantt chart. By looking at the data files, we find that the D_j values corresponding to these gates are relatively large. For example, the D_j values of gates 15–19 are 5939 m, 5275 m, 5503 m, 5751 m, and 5226 m, respectively, so the algorithm is not inclined to use these gates when there are other gates that can stop flights.

From the experimental results of CPLEX, it can be seen that the traditional mathematical programming method is not suitable for more than 120 flights. Compared with heuristic methods, the time consumption cost is too high, and as the scale of the problem increases, the result cannot be obtained. In order to further verify that the APGA algorithm is applicable to large-scale GAPs and to demonstrate the stability of the algorithm for different datasets with the same number of flights, we randomly generated 10 different flight datasets from the flight data on November 23, 2019, and each set of data contains 200 flights. We determined that the number of contact stands for testing is 40, and the number of remote stands is 38. We conducted five repeated experiments on different datasets to reduce the influence of random factors.

The results of repeated tests of different instances are shown in Table 14. The first column of the table indicates the serial number of instances, and the column "Repeated experiments" are the results of 5 repeated experiments. The

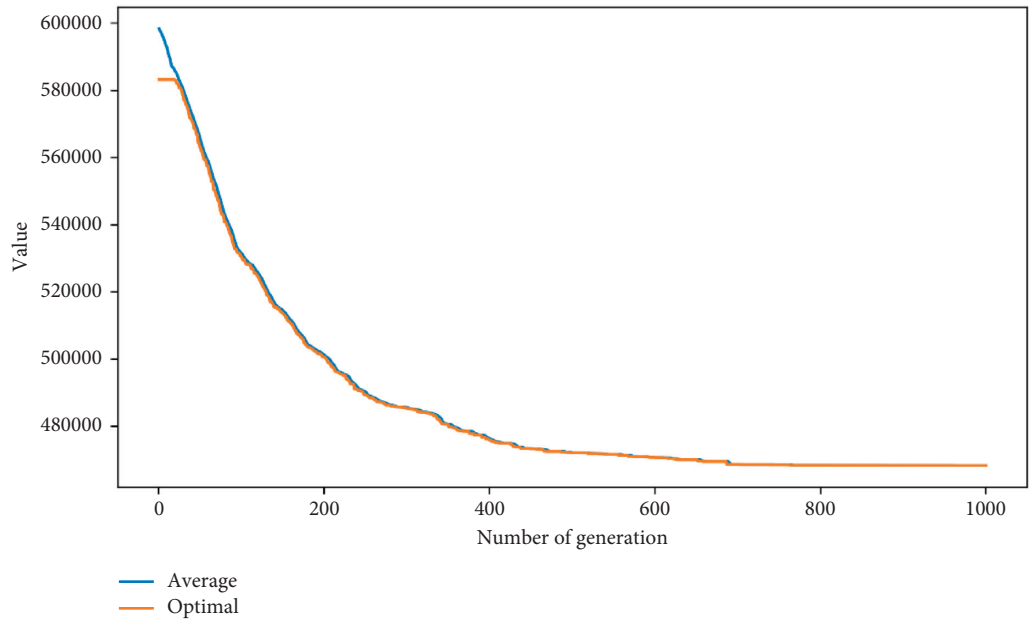

Figure 7: Schematic diagram of convergence process of 200 flights.

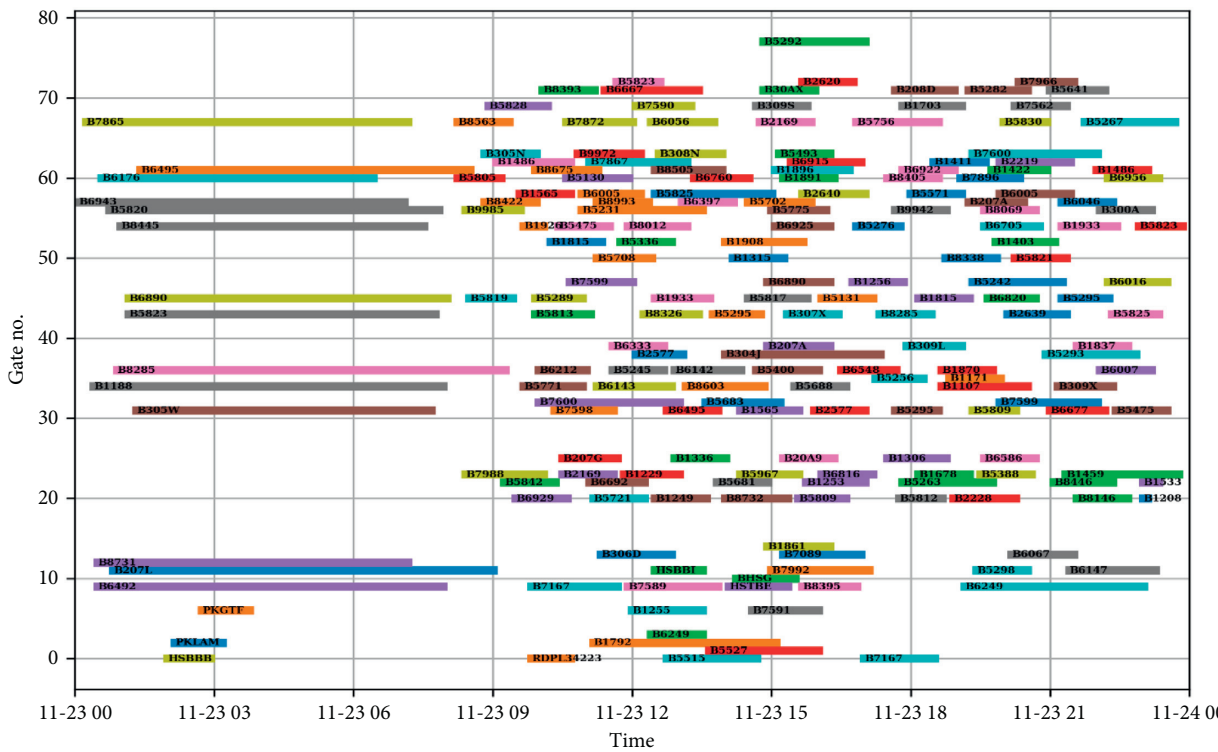

Figure 8: Gantt chart for the result of 200 flights.

corresponding result value of each experiment is the value of the objective function Z, and the specific calculation process is shown in equation (4). The remaining columns are statistical indicators of repeated experimental results, mainly including maximum, mean, minimum, standard deviation, and coefficient of variation. By observing the maximum, minimum, and standard deviation in the table, we can find that for different datasets, the final objective function values are basically between 46000 and 47500, and the standard deviations are all below 1000. Further analysis of CV (coefficient of variation) shows that for datasets with different mean values, the CV values are all below 0.21%, which indicates that the algorithm can maintain high stability. It shows that in the case of large-scale GAPs, the proposed algorithm can not only reduce computation time but also have high stability and practicability.

Table 14: Results of repeated experiments.

Instances	Repeated experiments					Min	Mean	Max	Standard deviation	Coefficient of variation (%)
	1	2	3	4	5					
1	473368	472216	471553	471016	473011	471016	472233	473368	979.45	0.21
2	467090	469041	468431	467084	467486	467084	467826	469041	873.23	0.19
3	466083	467888	466372	466732	466804	466083	466776	467888	686.06	0.15
4	467818	466947	465739	467786	467005	465739	467059	467818	845.94	0.18
5	466347	467341	465296	467755	466343	465296	466616	467755	963.36	0.21
6	469341	468463	468388	467527	469455	467527	468635	469455	788.84	0.17
7	463768	462837	464477	463782	463544	462837	463682	464477	587.90	0.13
8	472016	473555	472281	472936	471434	471434	472444	473555	822.53	0.17
9	461351	460935	460981	461921	462282	460935	461494	462282	591.48	0.13
10	464262	462783	463428	464093	463054	462783	463524	464262	641.79	0.14

6. Conclusions

Based on the analysis of the research status of GAPs, we propose a gate assignment model based on the actual assignment rules of airports, which comprehensively considers the factors of the remote stand penalty, the travel distance of passengers, and the fuel consumption of taxiing. First, we formulate actual allocation rules into model constraints, such as flight type constraints, safe time interval constraints, nation constraints, adjacency conflict constraints, and so on. Next, an improved adaptive parallel genetic algorithm is proposed to solve the large-scale GAPs. Then, we generate different sample datasets from the real data of Kunming Changshui International Airport. The calculation results of the proposed algorithm are compared with those of standard genetic algorithm and CPLEX, which show that the proposed algorithm has better performance and takes a shorter computational time. In addition, we repeat tests on a large dataset of 200 flights to show the stability and practicability of the algorithm. Moreover, even if the size of the GAP becomes larger and the required iteration cannot be completed within the required time, we can also stop the algorithm and obtain an approximate solution.

Notably, compared with the actual process of gate assignment, the factors considered in the model are relatively simplified, and only the factors that are critical in the actual process are considered. This paper assumes that the capacity of the airport can meet the requirements of gate allocation. However, during the peak hours of airport operation, airport resources are tight, and there may not be an allocation result that allows all flights to be assigned. The algorithm still wants to allocate all flights, so it cannot get a feasible solution. Therefore, built upon the proposed algorithm, new assignment strategies will be introduced in future research.

Data Availability

The data used in this paper were supplied by Yunnan TravelSky Airport Technology Co., Ltd., under license and so cannot be made freely available.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was sponsored by the Key Laboratory of Airport Cluster Intelligent Operation of China (KLAGIO20180801).

References

[1] E. Piazza, "Increasing airport efficiency: injecting new technology," *IEEE Intelligent Systems*, vol. 17, no. 3, pp. 10–13, 2002.

[2] Z. Ma and D. Cui, "Optimizing airport flight delays," *Journal-Tsinghua University*, vol. 44, no. 4, pp. 474–477, 2004.

[3] O. Babić, D. Teodorović, and V. Tošić, "Aircraft stand assignment to minimize walking," *Journal of Transportation Engineering*, vol. 110, no. 1, pp. 55–66, 1984.

[4] R. A. Bihr, "A conceptual solution to the aircraft gate assignment problem using 0, 1 linear programming," *Computers & Industrial Engineering*, vol. 19, no. 1–4, pp. 280–284, 1990.

[5] S. Yan and C.-M. Huo, "Optimization of multiple objective gate assignments," *Transportation Research Part A: Policy and Practice*, vol. 35, no. 5, pp. 413–432, 2001.

[6] F. Jaehn, "Solving the flight gate assignment problem using dynamic programming," *Zeitschrift für Betriebswirtschaft*, vol. 80, no. 10, pp. 1027–1039, 2010.

[7] B. Maharjan and T. I. Matis, "Multi-commodity flow network model of the flight gate assignment problem," *Computers & Industrial Engineering*, vol. 63, no. 4, pp. 1135–1144, 2012.

[8] Y. Jiang, L. Zeng, and Y. Luo, "Multiobjective gate assignment based on passenger walking distance and fairness," *Mathematical Problems in Engineering*, vol. 2013, Article ID 361031, 7 pages, 2013.

[9] C. Yu, D. Zhang, and H. Y. K. Lau, "MIP-based heuristics for solving robust gate assignment problems," *Computers & Industrial Engineering*, vol. 93, pp. 171–191, 2016.

[10] V. J. Baker, "Pitching a tent in the native village; Malinowski and participant observation," *Journal of the Humanities and Social Sciences of Southeast Asia*, vol. 143, no. 1, pp. 14–24, 1987.

[11] K. Srihari and R. Muthukrishnan, "An expert system methodology for aircraft-gate assignment," *Computers & Industrial Engineering*, vol. 21, no. 1–4, pp. 101–105, 1991.

[12] Y. Cheng, "A knowledge-based airport gate assignment system integrated with mathematical programming," *Computers & Industrial Engineering*, vol. 32, no. 4, pp. 837–852, 1997.

[13] S. Yan, C.-Y. Shieh, and M. Chen, "A simulation framework for evaluating airport gate assignments," *Transportation*

Research Part A: Policy and Practice, vol. 36, no. 10, pp. 885–898, 2002.

[14] S. Yan and C.-H. Tang, "A heuristic approach for airport gate assignments for stochastic flight delays," *European Journal of Operational Research*, vol. 180, no. 2, pp. 547–567, 2007.

[15] R. S. Mangoubi and D. F. X. Mathaisel, "Optimizing gate assignments at airport terminals," *Transportation Science*, vol. 19, no. 2, pp. 173–188, 1985.

[16] H. Ding, A. Lim, B. Rodrigues, and Y. Zhu, "The over-constrained airport gate assignment problem," *Computers & Operations Research*, vol. 32, no. 7, pp. 1867–1880, 2005.

[17] U. Dorndorf, A. Drexl, Y. Nikulin, and E. Pesch, "Flight gate scheduling: state-of-the-art and recent developments," *Omega*, vol. 35, no. 3, pp. 326–334, 2007.

[18] U. Dorndorf, F. Jaehn, C. Lin, H. Ma, and E. Pesch, "Disruption management in flight gate scheduling," *Statistica Neerlandica*, vol. 61, no. 1, pp. 92–114, 2007.

[19] U. Dorndorf, F. Jaehn, and E. Pesch, "Modelling robust flight-gate scheduling as a clique partitioning problem," *Transportation Science*, vol. 42, no. 3, pp. 292–301, 2008.

[20] U. Dorndorf and E. Pesch, "Fast clustering algorithms," *ORSA Journal on Computing*, vol. 6, no. 2, pp. 141–153, 1994.

[21] U. Dorndorf, F. Jaehn, and E. Pesch, "Flight gate scheduling with respect to a reference schedule," *Annals of Operations Research*, vol. 194, no. 1, pp. 177–187, 2012.

[22] U. Dorndorf, F. Jaehn, and E. Pesch, "Flight gate assignment and recovery strategies with stochastic arrival and departure times," *OR Spectrum*, vol. 39, no. 1, pp. 65–93, 2017.

[23] M. Şeker and N. Noyan, "Stochastic optimization models for the airport gate assignment problem," *Transportation Research Part E: Logistics and Transportation Review*, vol. 48, no. 2, pp. 438–459, 2012.

[24] H. Zhao and L. Cheng, "Ant colony algorithm and simulation for robust airport gate assignment," *Mathematical Problems in Engineering*, vol. 2014, Article ID 804310, 7 pages, 2014.

[25] S. Liu, W. Chen, and J. Liu, "Optimizing airport gate assignment with operational safety constraints," in *Proceedings of the 2014 20th International Conference on Automation and Computing (ICAC)*, X. Luo, Ed., pp. 61–66, IEEE, Cranfield, UK, September 2014.

[26] M. Dell'Orco, M. Marinelli, and M. G. Altieri, "Solving the gate assignment problem through the fuzzy bee colony optimization," *Transportation Research Part C: Emerging Technologies*, vol. 80, pp. 424–438, 2017.

[27] W. Deng, H. Zhao, X. Yang, J. Xiong, M. Sun, and B. Li, "Study on an improved adaptive PSO algorithm for solving multi-objective gate assignment," *Applied Soft Computing*, vol. 59, pp. 288–302, 2017.

[28] C. Yu, D. Zhang, and H. Y. K. Lau, "An adaptive large neighborhood search heuristic for solving a robust gate assignment problem," *Expert Systems with Applications*, vol. 84, pp. 143–154, 2017.

[29] S. Liu, W.-H. Chen, and J. Liu, "Robust assignment of airport gates with operational safety constraints," *International Journal of Automation and Computing*, vol. 13, no. 1, pp. 31–41, 2016.

[30] S. Mokhtarimousavi, D. Talebi, and H. Asgari, "A non-dominated sorting genetic algorithm approach for optimization of multi-objective airport gate assignment problem," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2672, no. 23, pp. 59–70, 2018.

[31] R. Xu and K. Cai, "Solving airport gate assignment problem using an improved genetic algorithm with dynamic topology," in *Recent Developments in Intelligent Computing, Communication and Devices*, S. Patnaik and V. Jain, Eds., vol. 752, pp. 877–884, Springer Singapore, Singapore, 2019.

[32] J. Xiong and C. Zhang, "Airport gate assignment with airplane taxiing cost analysis," *Journal of Transportation Systems Engineering and Information Technology*, vol. 10, no. 3, pp. 165–170, 2010.

[33] D. E. Goldberg and J. H. Holland, "Genetic algorithms and machine learning," *Machine Learning*, vol. 3, pp. 95–99, 1988.

[34] K. A. de Jong, *"Analysis of the behavior of a class of genetic adaptive systems"* Ph.D. thesis, University of Michigan, Ann Arbor, MI, USA, 1975.

[35] I. Kucukkoc, A. D. Karaoglan, and R. Yaman, "Using response surface design to determine the optimal parameters of genetic algorithm and a case study," *International Journal of Production Research*, vol. 51, no. 17, pp. 5039–5054, 2013.

[36] Y. L. Wang and C. F. Wang, "Aircraft stands pre-assignment based on passenger traffic characteristics and cross entropy," *Journal of Transportation Systems Engineering and Information Technology*, vol. 16, no. 4, pp. 211–216, 2016.