Available online at www.sciencedirect.com
ScienceDirect
European Journal of Operational Research 180 (2007) 547–567
EUROPEAN JOURNAL OF OPERATIONAL RESEARCH
www.elsevier.com/locate/ejor

Discrete Optimization

A heuristic approach for airport gate assignments for stochastic flight delays

Shangyao Yan, Ching-Hui Tang *

Department of Civil Engineering, National Central University, Chungli 32054, Taiwan

Received 21 June 2005; accepted 3 May 2006
Available online 19 June 2006

Abstract

To make good flight to gate assignments, not only do all the relevant constraints have to be considered, but stochastic flight delays that occur in actual operations also have to be taken into account. In past research, airport gate assignments and stochastic disturbances have often been handled in the planning and the real-time stages separately, meaning that the interrelationship between these stages, as affected by such delays, has been neglected. In this research, we develop a heuristic approach embedded in a framework designed to help the airport authorities make airport gate assignments that are sensitive to stochastic flight delays. The framework includes three components, a stochastic gate assignment model, a real-time assignment rule, and two penalty adjustment methods. The test results are based on data supplied by a Taiwan international airport, and show that the proposed framework performs better than the current manual assignment process and the traditional deterministic model.

Keywords: Stochastic programming; Gate assignment; Stochastic flight delay; Real-time gate assignment

1. Introduction

Gate flight assignment is an essential feature of an airport's operation. In past research flights were assigned to gates based on the projected flight schedule and fixed parameters. However, in actual operations, stochastic flight delays (such as early or late arrivals and late departures) often occur, causing real-time stochastic disturbances of these gate assignments. Such daily disturbances may reduce the performance of the already planned gate assignment when actually applied to real operations. Therefore, to design a better gate flight assignment plan, not only do the related constraints have to be considered, but also stochastic flight delays that occur in actual operations.

However, a gate assignment plan that does not consider real-time gate assignments (for convenience, we call this *reassignment*) cannot incorporate such disturbances. In the real-time stage, if the two planned *time windows* (a period of time for a flight assigned to a gate), for two consecutive flights, overlap due to stochastic

* Corresponding author. Tel.: +886 3 422 7151x34155; fax: +886 3 425 2960.
E-mail address: s1342015@cc.ncu.edu.tw (C.-H. Tang).

0377-2217/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.ejor.2006.05.002

flight delays, then the airport authority thus needs to reassign the two conflicting flights. Such a reassignment may have an impact on downstream flights. Unfortunately, such downstream disturbances are affected by many complicated factors, including the original gate assignments, the actual flight delays, the reassignment rule, and constraints related to reassigning two conflicting flights, all of which are difficult to estimate ahead of time. Therefore, the interrelationship between the planning and the real-time stages, as affected by stochastic flight delays, has to be taken into account. In other words, if the planned assignments and the reassignments can be integrated in the planning stage, then the two stages can be systematically analyzed together, which would produce a more useful gate assignment plan sensitive to stochastic flight delays.

A number of analytical gate flight assignment models, which were generally formulated as zero–one integer (linear or quadratic) programs, mixed integer programs or network flow problems, have been developed. For example, see Braaksma (1977), Babic et al. (1984), Mangoubi and Mathaisel (1985), Vanderstraetan and Bergeron (1988), Bihr (1990), Zhang et al. (1994), Cheng (1997), Yan and Chang (1998), Haghani and Chen (1998), Bolat (1999, 2000), and Yan and Huo (2001). Their objective functions usually include the minimization of the total passenger waiting time, the total passenger walking distance, the number of off-gate events, the range of unutilized time periods for gates, the variance of idle times at the gates, or a combination of the above. However, in all of the aforementioned models to finalize the planned gate assignments, the proposed flight schedule was fixed, and the common stochastic flight delays that occur in actual daily operations were neglected.

In addition to the above analytical models, some simulation models (Hamzawi, 1986) and expert systems (Gosling, 1990; Su and Srihari, 1993) for aircraft gate assignments have been developed. Gu and Chung (1999) did solve a gate reassignment problem during real-time operations when there were stochastic flight delays, however, they focused on reassigning flights to alternate gates, not solving for a planned gate assignment. Yan et al. (2002) proposed a simulation framework with which to evaluate the interrelationship between the planned and the real-time gate assignments necessary to meet the stochastic flight delays that occur in real operations. However, they did not go further, and solve for the gate assignment plan that would best absorb such stochastic flight disturbances. To the best of the authors' knowledge, no one has integrated both the planning and the real-time stages together to solve for gate assignment plans flexible enough to meet stochastic disturbances.

Stochastic and robust optimization concepts have recently been employed to deal with planning problems under uncertain disturbances in several fields. For examples of stochastic optimization, see Mulvey and Ruszczynski (1995), Cheung and Powell (1996), Du and Hall (1997), and Kenyon and Morton (2003). For examples of robust optimization, see Paraskevopoulos et al. (1991), Escudero et al. (1993), Mulvey et al. (1995), Gutierrez and Kouvelis (1995), Yu and Li (2000), Soteriou and Chase (2000), List et al. (2003), Bertsimas and Sim (2004), and Rosenberger et al. (2004). The objective functions of stochastic optimization models are generally designed to optimize the expected value of all possible scenarios. Mulvey et al. (1995) have used *solution robustness* and *model robustness* concepts to develop robust optimization models. Solution robustness means that the solution remains close to optimal for all possible scenarios, and model robustness means that the solution must remain almost feasible for all scenarios. Generally, unlike stochastic optimization models, robust optimization models are sensitive to variance of the objective value in different scenarios, which is what makes solution robustness. To ensure model robustness of some robust optimization models (e.g., Mulvey et al., 1995), the violation of some constraints is allowed for some scenarios. However, although the aforementioned stochastic and robust optimization models are good in the planning stage, the planned results do not take the real-time stage into account. Problems integrating the planning and real-time stages have not been discussed.

One may wonder if the current optimization methods would be useful for solving integrated problems. In fact, the integration of the planning and real-time stages is a complicated problem, which is more difficult to handle than traditional schemes in which these stages are handled separately. The integration problem is composed of two systems, the planned assignment and the reassignment. In practice, reassignment is usually based on a reassignment rule by the airport authority, which is difficult to formulate in a model due to the complexity of actual operations. As a result, the integrated problem is difficult to formulate in a closed form, making the theoretically optimal solution or a proper lower bound of the integrated problem difficult to explore. It is also difficult to extend the current optimization methods to deal with the integrated problem.

Due to the limitations of the current optimization methods, in this research, we aim to develop a heuristic approach embedded in a framework that can efficiently solve the integrated problem. The framework is designed to be an improvement over the current manual assignment process and the traditional deterministic model, and thus should make airport gate assignments that are sensitive to stochastic flight delays. The framework includes three components, a stochastic flight delay gate assignment model (SFDGAM), a reassignment rule, and two penalty adjustment methods (PAMs). In addition, a simulation-based evaluation method is developed to evaluate the performance of the proposed framework when stochastic flight delays occur. We demonstrate the performance of the framework using numeric tests. It is expected that the integration of the planning and real-time stages will receive increasing attention in planning applications that include stochastic disturbances. It is also expected that the development of the framework and the test results would be useful as a reference for theoretical future studies.

In this research, we only address disturbances that are due to stochastic flight delays that occur in regular operations, rather than other larger types of disturbance, such as aircraft malfunctions, airport closures or unexpected airport congestion. However, the incorporation of such disturbances could be a direction of future research. The remainder of the paper is organized as follows. In Section 2, we introduce the framework. In Section 3, a numerical test, based on CKS Airport (a major airport in Taiwan) operations, is performed to evaluate the framework. Finally, we conclude in Section 4.

2. The framework

The framework includes three parts: the SFDGAM for the planning stage, the reassignment rule for the real-time stage, and the PAMs. Before introducing the three parts, we first outline the solution process.

2.1. The solution process and assumptions

The solution process is as follows. In the planning stage, we first establish the SFDGAM for n flight delay scenarios. To model the interrelationship between the planned assignments and the reassignments as affected by stochastic flight delays, we add penalty values in the SFDGAM. Then using the mathematical programming solver, CPLEX, SFDGAM is solved to obtain the planned gate assignment. Secondly, to simulate reassignments, if an airplane cannot be assigned to its original gate (as obtained from the SFDGAM), then we apply a rule to reassign the airplane to another gate, for all n scenarios. When all n scenarios have been reassigned, we can obtain n reassignment results and estimate the objective value for the reassignments. Thirdly, using these reassignment results, we design two PAMs to revise the penalty value and repeat the above process to obtain a new planned gate assignment. The solution process is designed to iterate the planned assignments and reassignments, with the use of the PAMs, until the stopping criterion is met. The stopping criterion is a preset number of iterations, for which a better objective value for the reassignments than the incumbent one is not found. This incumbent, or the best solution (including the objective value for the reassignments and the planned gate assignments) obtained during the solution process, becomes the final solution. The solution process is indicated in Fig. 1.

The result obtained from the framework is a heuristic solution, designed primarily to help airport authorities get better gate assignments sensitive to stochastic flight delays. In practice, the solution process can be stopped when a satisfactory result is acquired. The stopping rule is adjustable according to their considerations. Note that, in this research, the framework components are designed based on actual CKS Airport operations. If the framework is applied at other airports, then components suitable for their own particular practices can be used or designed using their own gate assignment models, reassignment rules and flight delay distributions, according to their own specific needs.

Before introducing the other components, we need the following information: (1) the number of passengers for each flight, (2) the number of gates and the layout of the terminal area, (3) the daily flight schedule, indicating when each aircraft arrives at and departs from the airport, (4) the aircraft type, here large (specifically, B747s) and wild (other aircraft types), associated with each flight, and (5) the *buffer time*, which is the period of time needed at the gate between two consecutive flights.

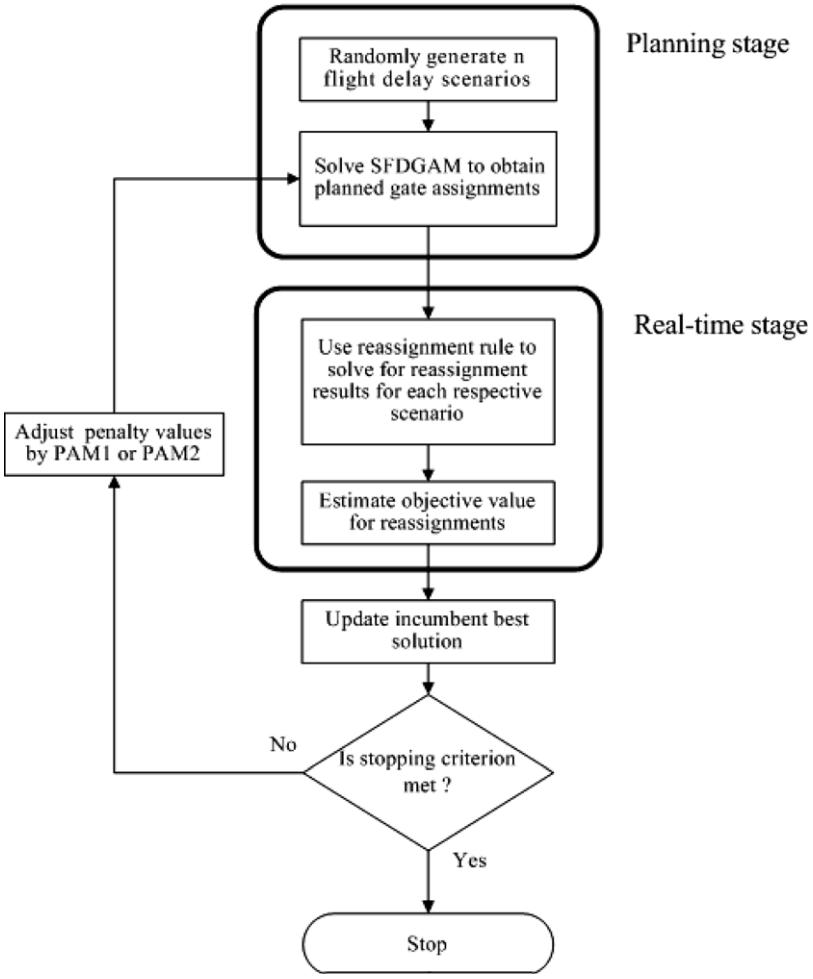

Fig. 1. Solution process.

The assumptions used for the modeling are listed as follows:

(1) At CKS the traditional buffer time is 30 minutes, but this is adjustable to user needs. For simplification, half of the total buffer time is added before and half after the time window used by the flight.
(2) The airline's preference is not considered (Yan and Huo, 2001; Yan et al., 2002). That is, the assignments of flights to the same type of gates are regarded as identical, but this is extendable, which will be discussed after the model formulation.
(3) We use the total passenger waiting time, the expected penalty value and the expected semideviation risk measure (SRM) (Ruszczynski and Shapiro, 2003) for all n scenarios, to build the objective function for the SFDGAM. The penalty value and the SRM will be addressed later. Other performance criteria, for example, the total passenger walking distance, the variance of idle gate times, or a combination of the above, may be used for other applications.
(4) A transferring flight forms a flight pair. In practice, a flight pair is served by a single airplane and is shortly connected, within one hour, so needs to be assigned to the same gate.
(5) The *starting time* of a flight is assumed to be when the gates are ready for the associated aircraft to use and the ending time is when the gates are released from use and ready for another. In other words, the starting time for an arriving or transferring flight is equal to the actual arrival time minus half the buffer time. The starting time for a departing flight is equal to the actual departure time minus the ground service time necessary for checking out the aircraft prior to departure, the fueling and passenger or package boarding, and

half of the buffer time. The *ending time* for an arriving flight is equal to the flight's actual arrival time, plus the passenger or package boarding time, plus half of the buffer time. The ending time for a departing or transferring flight is equal to the flight's actual departure time plus half of the buffer time. Generally, the time window between the starting and ending times is the time block for a particular flight assigned to a gate.

(6) Since currently CKS Airport gate assignments are made daily, the planning period is set to be one day; however this is adjustable.

2.2. The planned gate assignments

We develop the SFDGAM to solve for the planned gate assignments. The major elements in the SFDGAM, including the gate-flow network and the mathematical formulation, are described as follows.

2.2.1. The gate-flow network

The gate-flow network developed in this study to formulate daily gate assignments is shown in Fig. 2. In practice, an airport would usually have several types of gates. For example, at the CKS Airport, there are three types, large, wild and temporary gates. Hence, we must have multiple gate-flow networks, each corresponding to one type of gate. In particular, large and temporary gates can be used by all types of aircraft while wild gates are only available for wild types of aircraft. Altogether, there are five types of arcs: (1) flight arcs, (2) alternate flight arcs, (3) connecting arcs, (4) starting and ending arcs, and (5) cycle arcs. They are defined below.

2.2.1.1. Flight arc. A flight arc represents a flight in the network that is available for the corresponding type of gate. All possible flight arcs are added in the network, provided that the correct type of gate is available. For example, only flight arcs with wild aircraft can be assigned to the wild gate-flow network, while any flight arc (whether with large or wild aircraft) can be assigned to large or temporary gate-flow networks. The head (e.g., node 1) and tail nodes (e.g., node 2) represent the starting and ending times of the associated flight, respectively. The arc cost is set to be zero, indicating the associated flight is assigned on time. The arc flow's upper bound is one, meaning that the flight can be assigned at most once. The arc flow's lower bound is zero, implying that the flight is not assigned to the corresponding type of gate.

2.2.1.2. Alternate flight arc. An alternate flight arc represents the holding of an aircraft on the ground while waiting for a gate. It is used for the airport authority to apply the delay strategy if there is no available gate (especially during peak hours). Several alternate flight arcs are added, with respect to each flight arc, to provide a choice of waiting time. To prevent a long passenger waiting time, which is not practical in actual operations (see Yan and Huo, 2001), we set the maximum allowable waiting time to be 30 minutes. We use a gap of 10 minutes to construct alternate flight arcs in the gate-flow network. For example, in Fig. 2, arcs 3–4, 5–6, and 7–8 are the alternate flight arcs for Flight 1. The arc cost is the number of passengers on the flight, multiplied by the corresponding waiting time. Similar to the flight arc, the arc flow's upper bound is set to be one and the lower bound is set to be zero.

Note that, to maintain the level of service, the CKS Airport does not plan for an aircraft to wait for a temporary gate. Therefore, we do not need to install an alternate flight arc in the temporary gate-flow network. Certainly, this is adjustable for other airports. In addition, because every flight must be assigned to only one gate and one time window, a side constraint should be added, ensuring that the sum of a flight's flight arc and alternate flight arc flows is equal to one.

2.2.1.3. Connecting arc. A connecting arc represents the continuity for a gate serving two different flights. It connects the head node of the flight arc or alternate flight arc to the tail node of another flight arc or alternate flight arc. If the connecting arc holds a flow in the solution, this means that the two connected flights are consecutively assigned to the same gate. Note that based on model robustness, infeasible assignments that violate some scenarios are allowed. That is, in the planning stage, two flights can be concurrently assigned to the same gate. To do this in the SFDGAM we construct a connecting arc even if the ending time (tail node) of one flight's flight arc or alternate flight arc is later than the starting time (head node) of the other flight's (i.e., the two associated time windows overlap). However, if all possible connecting arcs with overlapping time windows for all n scenarios are constructed, then the number of connecting arcs will be too large, making the

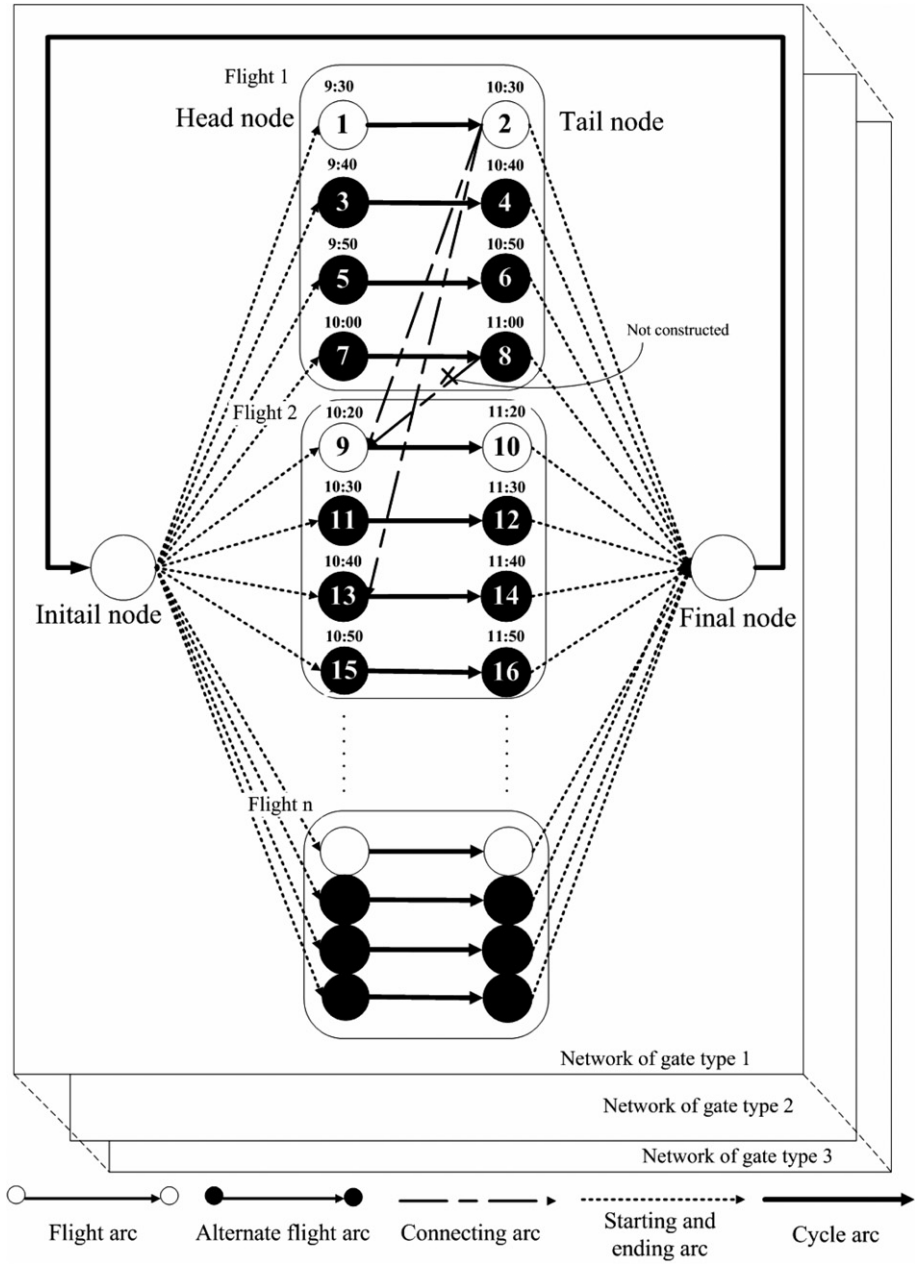

Fig. 2. Gate-flow network.

model difficult to solve and unreasonable to put into practice. Therefore, we use the average starting and ending times of the n scenarios as the criterion to construct the connecting arcs. In other words, a connecting arc for the two flight arcs or alternate flight arcs can be constructed if the average ending time of the one is later than the average starting time of the other, but within 30 minutes. In addition, according to CKS Airport staff experience, the gap between two flights sequentially assigned to the same gate rarely exceeds 4.5 hours. A connecting arc can also be constructed for two flight arcs or alternate flight arcs if the average ending time of one is earlier than the average starting time of the other, but within 4.5 hours. These values are adjustable. The arc cost is equal to zero, the arc upper bound is one and the arc lower bound is zero.

To better understand the construction of connecting arcs in the SFDGAM, suppose that there are two flights, Flight 1 and Flight 2, as shown in Fig. 2. Using the flight delay distributions for Flight 1 and Flight

2, we generate departing and arriving times for n scenarios. Using the starting and ending times introduced above, we can then calculate the average starting and ending times of the n scenarios. For simplicity, we also assume that: for Flight 1 the average starting and ending times for n scenarios are 9:30 and 10:30, respectively; for Flight 2 they are 10:20 and 11:20. Thus, flight arcs 1–2 and 9–10 only overlap by 10 minutes (10:30–10:20), far less than 30 minutes, so connecting arc 2–9 can be constructed. Since alternate flight arc 13–14 indicates that Flight 2 has been delayed by 20 minutes, there is no overlap between flight arc 1–2 and alternate flight arc 13-14 (10:30–10:40). Therefore, connecting arc 2–13 can be constructed. Since alternate flight arc 7–8 indicates that Flight 1 has been delayed by 30 minutes, the overlap time between alternate flight arc 7–8 and flight arc 9–10 is 40 minutes (11:00–10:20), hence, connecting arc 8–9 cannot be included in the network.

2.2.1.4. Starting and ending arcs. A starting arc connects the initial node to the head node of a flight arc or alternate flight arc; indicating the beginning of the daily gate assignments. An ending arc connects the tail node of a flight arc or alternate flight arc to the final node, and denotes the end of the day's assignments. The starting and ending arc costs are both zero. The upper bounds for the two arcs are both one and the lower bounds are both zero.

2.2.1.5. Cycle arc. A cycle arc connects the final node in each network to the initial node. The arc flow represents the number of each type of gate used in the network. The arc cost is equal to zero. The upper bound is the available number of gates in each network and the lower bound is zero.

2.2.2. The model formulation

Before introducing the model formulation, we first represent the penalty value and the SRM used in the SFDASM. We add a penalty value to each connecting arc for each flight delay scenario in the SFDGAM. That is, each connecting arc has n penalty values with respect to the n flight delay scenarios. We use the designed penalty value to interrelate the SFDASM and the reassignments of the n scenarios. Thus, we can integrate the planned assignments and the reassignments together, to solve for a good stochastically sensitive assignment result that can absorb flight delays. The notations and symbols used in the SFDGAM are

Decision variables

x_{ij}^k the arc (i,j) flow in the kth gate-flow network

Parameters and sets

c_{ij}^k the arc (i,j) cost in the kth gate-flow network
$u_{ij}^{s,k}$ the penalty value for the connecting arc (i,j) in the kth gate-flow network for scenario s
U^s the total penalty value for scenario s
$E[U^s]$ the expected penalty value of all scenarios
p^s the probability for scenario s
z^s the objective value for scenario s
$E[z^s]$ the expected objective value of all scenarios
w the weighting vector for the SRM, which can be determined by the user according to his or her own considerations
ZP the objective value of the SFDGAM
g^k the number of available gates in the kth gate-flow network
CA^k the set of all cycle arcs in the kth gate-flow network
K the set of all gate types
N^k, A^k the set of all nodes and arcs in the kth gate-flow network
F_t the set of all flight arcs and alternate flight arcs for flight t
AF the set of all flights.
Ω the set of all scenarios. Note that the aforementioned n scenarios selected in the SFDGAM form a sub-set of Ω
QA^k the set of all connecting arcs in the kth gate-flow network

The total penalty value for scenario s can be written as follows:

$$U^s = \sum_{k\in K}\sum_{ij\in \mathrm{QA}^k} u_{ij}^{s,k}x_{ij}^k. \tag{1}$$

We use the SRM for solution robustness. Other approaches to solution robustness that better meet the user needs could also be used. The SRM represents the expected excess over the expected objective value $E[z^s]$, as follows:

$$\mathrm{SRM} = E[(z^s - E[z^s])_+] = E[\max(0, z^s - E[z^s])]. \tag{2}$$

In this research, the objective value for scenario s can be represented as

$$z^s = \sum_{k\in K}\sum_{ij\in A^k} c_{ij}^k x_{ij}^k + U^s = \sum_{k\in K}\sum_{ij\in A^k} c_{ij}^k x_{ij}^k + \sum_{k\in K}\sum_{ij\in \mathrm{QA}^k} u_{ij}^{s,k}x_{ij}^k. \tag{3}$$

The first term $\sum_{k\in K}\sum_{ij\in A^k} c_{ij}^k x_{ij}^k$ is the same for each scenario, while the second term $\sum_{k\in K}\sum_{ij\in \mathrm{QA}^k} u_{ij}^{s,k}x_{ij}^k$ varies with different scenarios. Therefore, referring to Eq. (2), the SRM is

$$\mathrm{SRM} = E[\max(0, U^s - E[U^s])] = \sum_{s\in\Omega} p^s \left(\max\left(0, \sum_{k\in K}\sum_{ij\in \mathrm{QA}^k} u_{ij}^{s,k}x_{ij}^k - \sum_{s\in\Omega} p^s \sum_{k\in K}\sum_{ij\in \mathrm{QA}^k} u_{ij}^{s,k}x_{ij}^k\right)\right). \tag{4}$$

The objective function of the SFDGAM is then

$$\begin{aligned}\min \quad \mathrm{ZP} &= \sum_{k\in K}\sum_{ij\in A^k} c_{ij}^k x_{ij}^k + E[U^s] + w \times \mathrm{SRM} \\ &= \sum_{k\in K}\sum_{ij\in A^k} c_{ij}^k x_{ij}^k + \sum_{s\in\Omega} p^s \sum_{k\in K}\sum_{ij\in \mathrm{QA}^k} u_{ij}^{s,k}x_{ij}^k + w \\ &\quad \times \sum_{s\in\Omega} p^s \left(\max\left(0, \sum_{k\in K}\sum_{ij\in \mathrm{QA}^k} u_{ij}^{s,k}x_{ij}^k - \sum_{s\in\Omega} p^s \sum_{k\in K}\sum_{ij\in \mathrm{QA}^k} u_{ij}^{s,k}x_{ij}^k\right)\right).\end{aligned} \tag{5}$$

For minimization problems, the objective function (5) can then be modified as follows:

$$\min \quad \mathrm{ZP} = \sum_{k\in K}\sum_{ij\in A^k} c_{ij}^k x_{ij}^k + \sum_{s\in\Omega} p^s \sum_{k\in K}\sum_{ij\in \mathrm{QA}^k} u_{ij}^{s,k}x_{ij}^k + w\sum_{s\in\Omega} p^s h^s \tag{6}$$

$$\text{s.t.} \quad \sum_{k\in K}\sum_{ij\in \mathrm{QA}^k} u_{ij}^{s,k}x_{ij}^k - h^s \leqslant \sum_{s\in\Omega} p^s \sum_{k\in K}\sum_{ij\in \mathrm{QA}^k} u_{ij}^{s,k}x_{ij}^k \quad \forall s \in \Omega, \tag{7}$$

$$h^s \geqslant 0 \quad \forall s \in \Omega. \tag{8}$$

The SFDGAM can now be formulated as follows:

$$\min \quad \mathrm{ZP} = \sum_{k\in K}\sum_{ij\in A^k} c_{ij}^k x_{ij}^k + \sum_{s\in\Omega} p^s \sum_{k\in K}\sum_{ij\in \mathrm{QA}^k} u_{ij}^{s,k}x_{ij}^k + w\sum_{s\in\Omega} p^s h^s \tag{9}$$

$$\text{s.t.} \quad \sum_{j\in N^k} x_{ij}^k - \sum_{r\in N^k} x_{ri}^k = 0 \quad \forall i \in N^k,\ \forall k \in K, \tag{10}$$

$$\sum_{k\in K}\sum_{ij\in F_t} x_{ij}^k = 1 \quad \forall t \in \mathrm{AF}, \tag{11}$$

$$0 \leqslant x_{ij}^k \leqslant g^k \quad \forall (i,j) \in \mathrm{CA}^k,\ \forall k \in K, \tag{12}$$

$$\sum_{k\in K}\sum_{ij\in \mathrm{QA}^k} u_{ij}^{s,k}x_{ij}^k - h^s \leqslant \sum_{s\in\Omega} p^s \sum_{k\in K}\sum_{ij\in \mathrm{QA}^k} u_{ij}^{s,k}x_{ij}^k \quad \forall s \in \Omega, \tag{13}$$

$$h^s \geqslant 0 \quad \forall s \in \Omega, \tag{14}$$

$$x_{ij}^k \in Z_+ \quad \forall (i,j) \in \mathrm{CA}^k,\ \forall k \in K, \tag{15}$$

$$x_{ij}^k = 0,1 \quad \forall (i,j) \in A^k - \mathrm{CA}^k,\ \forall k \in K. \tag{16}$$

The SFDGAM can be formulated as an integer multiple commodity network flow problem. The objective function (9) denotes the minimization of the total passenger waiting time, the expected penalty value for all n scenarios, and the expected SRM for all n scenarios multiplied by the weighting vector w. Constraint (10) is the flow conservation constraint at every node in each network. Constraint (11) denotes that every flight is assigned to only one gate and one time window. Constraint (12) ensures that the number of gates used in each network does not exceed its available number of gates. Constraints (13) and (14) are used to calculate the SRM. Constraint (15) ensures that the cycle arc flows are integers. Constraint (16) indicates that, except for the cycle arcs, all other arc flows are either zero or one.

2.2.3. The extension and the deterministic flight delay gate assignment model (DFDGAM)

If the airport authority must consider airline preference for the assignment of flights to gates or the total passenger walking distance for the objective function, then the relative location of assigned gates needs to be considered. To do this, the SFDGAM needs be extended to multiple gate-flow networks, each corresponding to a specific gate. Related constraints should also be modified accordingly. In particular, the upper bound of the cycle arc in each network is modified to be one. As well, the arc cost should be modified according to the use's specific needs. For example, if an airline prefers to use certain gates, then a smaller cost can be set for its flight and alternate flight arcs in the corresponding networks. As well, if neither the airport nor the airline prefers that a flight be assigned to a temporary gate, then a larger cost is set for the temporary gate flight arcs.

The major differences between the SFDGAM and the DFDGAM are the construction of the connecting arc, the penalty value $u_{ij}^{s,k}$ and the SRM. In the DFDGAM, the two flights cannot be concurrently assigned to the same gate. In other words, we can only construct a connecting arc for the two flight arcs or alternate flight arcs, if the average ending time of the one is earlier but within 4.5 hours of the average starting time of the other. In addition, both the penalty value $u_{ij}^{s,k}$ and the SRM are not considered in the DFDGAM. We use only the total passenger waiting time as an objective for the DFDGAM. As well, in the DFDGAM constraints (13) and (14) from the SFDGAM are not included. The detailed model formulation of the DFDGAM is shown in Appendix.

The connecting arcs in the SFDGAM will include the ones in the DFDGAM. Other types of arcs are the same for both the SFDGAM and the DFDGAM. Hence, with the SFDGAM, if the penalty value for the connecting arcs that cannot be constructed in the DFDGAM is set to be a very large value (or infinite), and the penalty value for the other connecting arcs is zero, then, for minimization problems, the SFDGAM terms $\sum_{k\in K}\sum_{ij\in \mathrm{QA}^k} u_{ij}^{s,k} x_{ij}^k$, $\sum_{s\in\Omega} p^s \sum_{k\in K}\sum_{ij\in \mathrm{QA}^k} u_{ij}^{s,k} x_{ij}^k$, and h^s will become zero. Thus, the SFDGAM solution is equal to the DFDGAM solution. Therefore, the DFDGAM solution can serve as an initial solution in the solution process. In other words, in theory the proposed framework can always improve the DFDGAM solution.

2.3. The reassignment

To evaluate the actual performance of the planned results in real-time operations, we perform n reassignments with the same n scenarios mentioned in Section 2.2. According to actual CKS Airport practices, the airport authority does not consider a buffer time when they reassign the flight to an alternate gate or have it wait for its original gate. Therefore, for the same n scenarios, in the real-time stage, the buffer time for each flight is excluded. The rule used here is now outlined.

We follow chronologically, finding a disrupted aircraft that cannot be assigned to the planned gate. Then we check whether the disrupted aircraft can be held without exceeding the maximum allowable waiting time. If the answer is yes, then we hold the disrupted aircraft until the original gate is available for use. In this case, if this disrupted aircraft overlaps the time window of a later incoming aircraft, then the later incoming aircraft now becomes disrupted. If the answer is no, then the disrupted aircraft is reassigned to an alternate gate. According to CKS Airport staff experience, the maximum allowable waiting time is 30 minutes. In addition, when a disrupted aircraft needs to be reassigned to an alternate gate, it should first be determined whether a wild (only for wild types of aircraft) or a large gate (for all types of aircraft) has an idle time window that does not overlap the time allowed to a later incoming aircraft. If the answer is yes, then we reassign the disrupted aircraft to the wild or large gate, which has the longest time available. If the answer is no, then we check whether there is a temporary gate with an available time window. If the answer is yes, then we reassign the

disrupted aircraft to the temporary gate which has the largest idle time window available. Otherwise, we find the wild or large gate with the shortest conflicting time, that overlaps the time of a later incoming aircraft, and then reassign the disrupted aircraft to this (note that, in this case, the later incoming aircraft now also becomes disrupted). Then, chronologically, for this disrupted aircraft, the same rule must be used to reassign it. The process is not stopped until all disrupted aircraft has been reassigned. The reassignment process is indicated in Fig. 3.

It should be mentioned that according to actual CKS Airport practices, the airport authority can use temporary gates to alleviate a shortage of gates during the peak flow. Under normal conditions flight will not be cancelled but will be assigned to another gate. Therefore, the cancellation strategy needs not be considered in our reassignment rule, but this can be modified for other airports. In addition, in this research, we only address disturbances that occur due to daily stochastic flight delays. In practice, as the airport authority receives the flight delay information in advance, they usually have enough lead time to reassign flights to alternate gate. Therefore, an uncertain lead time is not considered here. Other types of disturbances, such as unexpected

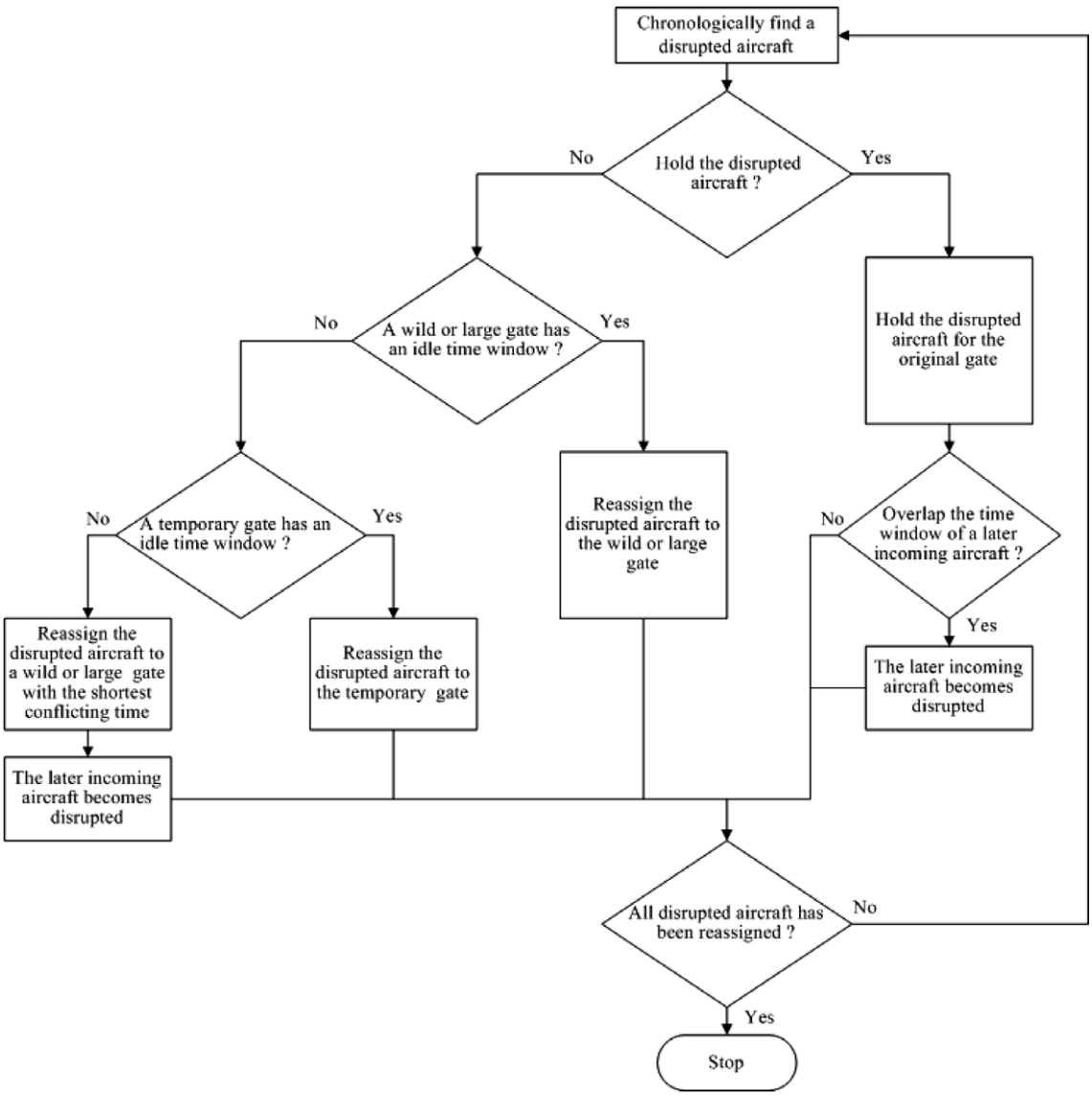

Fig. 3. Reassignment rule process.

aircraft malfunctions, may also cause uncertain lead times. Thus, a stochastic lead time needs to be considered in the reassignment process, which could be a topic of future research.

The other important measure at service or system performance in the real-time stage is inconsistent gate assignments. In practice, if a disrupted aircraft cannot be assigned to its planned gate, then the reassignment is considered inconsistent. At the CKS Airport, the handling of such reassignments in real operation is typically time constrained. For most airport authorities, the disturbances that arise from reassignments must be considered during the real-time stage. However, the consequences of such inconsistencies are not easy to evaluate, as they are related to such subjective factors as the feelings of passengers, inconvenience to airport and airlines staffs, the efficient use and flexible adjustment of ground service personnel, and of course airport service levels. Therefore, for simplicity, we use a *disturbance time*, which is multiplied by the number of passengers on the associated disturbed flight, as an *inconsistent time*, for measuring disturbances that occur when the disturbed flight is reassigned to an alternate gate. The disturbance time, based on CKS staff suggestions, can be set to 30 minutes per inconsistent flight assignment. Note that this primarily helps to describe a disturbance due to an inconsistent flight assignment. The measure for evaluating such an inconsistency is adjustable in other applications. For example, the user could just set a fixed inconsistent time for an inconsistent assignment, or set a variable one by taking into consideration the terminal configuration, the distance between gates, and other such factors. Finally, the notations and symbols used in the real-time stage are listed below:

zr^s	the objective value for scenario s after reassignments
wt^s	the passenger waiting time for scenario s after reassignments
it^s	the inconsistent time for scenario s due to inconsistent reassignments
ZR	the objective value for the reassignment
AWT	the average passenger waiting time of n scenarios after reassignments
AIT	the average inconsistent time of n scenarios due to inconsistent reassignments

Altogether, we consider both the total passenger waiting time and the total inconsistent time due to inconsistent reassignments. Thus, in the real-time stage, the objective value for each scenario can be calculated as

$$\mathrm{zr}^s = \mathrm{wt}^s + \mathrm{it}^s. \tag{17}$$

Then, similar to what was done in Section 2.2, we apply Eq. (2) to estimate the semideviation risk measure after the reassignment (SRMR), as follows:

$$\mathrm{SRMR} = E[\max(0, \mathrm{zr}^s - E[\mathrm{zr}^s])] = \sum_{s\in\Omega} p^s \left(\max\left(0, \mathrm{zr}^s - \sum_{s\in\Omega} p^s \mathrm{zr}^s\right)\right). \tag{18}$$

As a result, ZR is

$$\begin{aligned}\mathrm{ZR} &= \mathrm{AWT} + \mathrm{AIT} + w \times \mathrm{SRMR} = \sum_{s\in\Omega} p^s \mathrm{wt}^s + \sum_{s\in\Omega} p^s \mathrm{it}^s + w \sum_{s\in\Omega} p^s \left(\max\left(0, \mathrm{zr}^s - \sum_{s\in\Omega} p^s \mathrm{zr}^s\right)\right) \\ &= \sum_{s\in\Omega} p^s \mathrm{zr}^s + w \sum_{s\in\Omega} p^s \left(\max\left(0, \mathrm{zr}^s - \sum_{s\in\Omega} p^s \mathrm{zr}^s\right)\right).\end{aligned} \tag{19}$$

2.4. The PAMs

In each iteration, after solving the SFDGAM and performing the reassignment, two developed PAMs are used to repeatedly adjust the penalty values, $u_{ij}^{s,k}$s. Note that other methods can be designed and tested, as a future research topic. A discussion of the two PAMs is as follows.

2.4.1. PAM1

In PAM1 we utilize the sum of the passenger waiting time and the inconsistent time obtained from the previous iteration, for a cumulative penalty value, $u_{ij}^{s,k}$, in each iteration. To illustrate the calculation, we assume

that connecting arc 1–2 connects two flights, Flight 1 and Flight 2, with 150 and 120 passengers, respectively. Suppose that arc 1–2 contains a flow (i.e., one) in the planned assignments, meaning that both Flight 1 and Flight 2 have been consecutively assigned to the same gate. For simplicity, let us consider three scenarios, 1–3. Also assume that after the reassignments, in scenario 1, Flight 2 waits for 10 minutes; in scenario 2, Flight 1 is reassigned to an alternate gate; and in scenario 3, both flights are assigned to the planned gate without waiting or reassignments. Therefore, in scenario 1, the connecting arc 1–2 has a passenger waiting time of 1200 ($10 * 120$) and in scenario 2, the connecting arc 1–2 has an inconsistent time of 4500 ($30 * 150$). In scenario 3, the passenger waiting time and the inconsistent time are both 0. Thus, the penalty values for connecting arc 1–2 in the three scenarios are 1200, 4500, and 0, respectively. Let m represent the mth iteration. Let $dw_{ij}^{s,k}$ and $di_{ij}^{s,k}$, respectively, denote the passenger waiting time and the inconsistent time with respect to connecting arc (i,j) in the kth gate-flow network for scenario s. The penalty values for each iteration calculated at PAM1 are shown as follows:

$$(u_{ij}^{s,k})^{m+1} = (u_{ij}^{s,k})^m + (dw_{ij}^{s,k} + di_{ij}^{s,k})^m. \tag{20}$$

2.4.2. PAM2

Yan and Young (1996) used a modified sub-gradient method to search for good Lagrangian multipliers when applying Lagrangian relaxation to network flow problems with side constraints. Since those Lagrangian multipliers, which correspond to the side constraints, are somewhat similar to the penalty values used in this research, we therefore modify their sub-gradient method to develop PAM2. The aforementioned authors calculated the violation values for the side constraints after the Lagrangian problem was solved. Hence, we have to estimate the violation values for the connecting arcs. To do this, we define the violation value for each connecting arc, in each scenario. In particular, if the connecting arc holds a flow in the planned assignments (obtained from the SFDGAM), then this means that these two connected flights will be assigned consecutively to the same gate. For a scenario, after the reassignment, if the passenger waiting time or the inconsistent time for the connecting arc is larger than zero, this means that in the real-time stage, one of the two connected flights must either wait for their planned gate or be reassigned to an alternate gate; in other words the connecting arc in this scenario is then considered to be violated. The violation value for the connecting arc in this scenario is set to be the connecting arc's flow (i.e., one). Oppositely, if the passenger waiting time or the inconsistent time for the connecting arc is equal to zero, it means that in the real-time stage, both connected flights can be assigned to the planned gate without waiting or reassignment; in other words the connecting arc in this scenario is then not considered to be violated. The violation value for the connecting arc in this scenario is set to be zero. Taking the same example in PAM1, based on the violation definition, the violation values for connecting arc 1–2 for the three scenarios are 1, 1, and 0, respectively.

Finally, the penalty values can be calculated by Eqs. (21)–(25). Where $dv_{ij}^{s,k}$ denotes the violation value for connecting arc (i,j) in the kth gate-flow network for scenario s. $d_{ij}^{s,k}$ is the direction for modifying $u_{ij}^{s,k}$, b^m is useful for modifying $dv_{ij}^{s,k}$ in the mth iteration, t^m is the step size in the mth iteration, and λ is a parameter for modifying t^m. After extensive testing, we found that the results were the best when $\lambda = 1.5$.

$$(dv_{ij}^{s,k})^m = \begin{cases} 1 & \text{if } dw_{ij}^{s,k} > 0 \text{ or } di_{ij}^{s,k} > 0, \\ 0 & \text{if } dw_{ij}^{s,k} = 0 \text{ and } di_{ij}^{s,k} = 0, \end{cases} \quad \forall (i,j) \in \mathrm{QA}^k, \ \forall k \in K, \ \forall s \in \Omega \tag{21}$$

$$b^m = \max\left\{0, \frac{-\sum (dv_{ij}^{s,k})^m (d_{ij}^{s,k})^{m-1}}{\|(d_{ij}^{s,k})^{m-1}\|^2}\right\}, \tag{22}$$

$$(d_{ij}^{s,k})^m = (dv_{ij}^{s,k})^m + b^m (d_{ij}^{s,k})^{m-1} \quad \forall (i,j) \in \mathrm{QA}^k, \ \forall k \in K, \ \forall s \in \Omega, \tag{23}$$

$$t^m = \frac{\lambda |\mathrm{ZR}^m - \mathrm{ZP}^m|}{\|dv_{ij}^{s,k}\|^2}, \quad 0 < \lambda \leqslant 2, \tag{24}$$

$$(u_{ij}^{s,k})^{m+1} = (u_{ij}^{s,k})^m + t^m (d_{ij}^{s,k})^m \quad \forall (i,j) \in \mathrm{QA}^k, \ \forall k \in K, \ \forall s \in \Omega. \tag{25}$$

3. Numerical tests

To test how well the framework may be applied in the real world, we performed numerical tests based on CKS Airport operations. We used the C computer language to write the necessary programs, coupled with the CPLEX 8.1 mathematical programming solver, to solve the problem. The tests were performed on an Intel P4 2G with 2 GB RAM in the environment of Microsoft Windows XP. In addition, we use a simulation-based evaluation method to compare the performance of the current CKS manual assignment process, the DFDGAM, and the framework. We randomly generate different flight delays to the n scenarios used in the framework. Then, using the different flight delay scenarios, we perform reassignments of the planned gate assignments obtained from the current CKS manual assignment process, the DFDGAM, and the framework. Finally, we perform a comparison of their results with different flight delay scenarios.

3.1. Data analyses

The timetable used for the test was based on the operating day, September 24, 2004. It included 172 flights. Fourteen aircraft types were included, which could be classified into two types for gate assignment purposes, large and wild. There were 49 arrival flights (five served by large aircraft), 50 departure flights (eight served by large aircraft), and 73 transferring flights (24 served by large aircraft) in a single day's operations. There were 29 gates in operation, of which eight were temporary. Seven of the 29 gates were wild type gates, while the others were large type gates. In addition, according to the airport recommendation, the weighting vector w and the disturbance time were set to be 2 and 30 minutes, respectively. The flight delay distributions were obtained from the actual CKS airport flight delay. Specifically, 88.39% of the departing flight delays ranged from 0 to 20 minutes while for arriving flights, 75.56% of the delays ranged from -9 (early) to 20 minutes. For ease of testing, the probability for each scenario was set to be the same, however, this is adjustable. The other inputs were primarily taken from Yan and Huo (2001) and from actual operating data.

3.2. Test results

Before performing the numerical tests, we evaluated a suitable number of scenarios used in the framework and the evaluation method. We tested different numbers of scenarios. We found that, after 40 scenarios, the assignment results did not change much. This will also be discussed in Section 3.3. For ease of testing, the number of scenarios in the framework and the evaluation method were both set to be 40, but this is adjustable. Five different cases, each with different 40 scenarios, were tested. For each case, the average flight delay for 40 scenarios was used as input to build and solve the DFDGAM. Then, using the DFDGAM assignment results, we applied the reassignment rule, to solve for reassignment results, for each scenario. To set the stopping criterion we evaluated a suitable number of iterations, for which a better solution than the incumbent one is not found during the solution process. The test results indicated that, for the framework, eight iterations were the best.

In addition, to understand the change in the objective values and assignments during the solution process of the framework, we estimate the rate of change of the ZR (RC_{ZR}) and the rate of change of the assignments (RC_A), for each iteration. In particular, RC_{ZR} indicates the gap (an absolute value) between the ZR in the current iteration and the one in the previous iteration. RC_A indicates the rate of change between the current and the previous iterations for flight arc, alternate flight arc, and connecting arc flows. It equals to the number of flight, alternate flight, and connecting arcs whose flows are changed between the current and the previous iterations, divided by the total number of flight, alternate flight, and connecting arcs. Due to the fact that the results for the frameworks with both PAMs were similar, only the results for the framework with PAM2 are discussed here. As shown in Fig. 4, the values of RC_{ZR} for all five cases tended to decrease as the number of iterations increased. After the stopping criterion was met, the RC_{ZR} fluctuations for the five cases were all within 5%. In particular, for case 3, the RC_{ZR} fluctuation was within only 3% after the iteration 36. Similar results were also found in RC_A, for the all five cases that is, the RC_A fluctuations were all about 20%, when the framework was stopped. These results show that the framework solutions gradually converged.

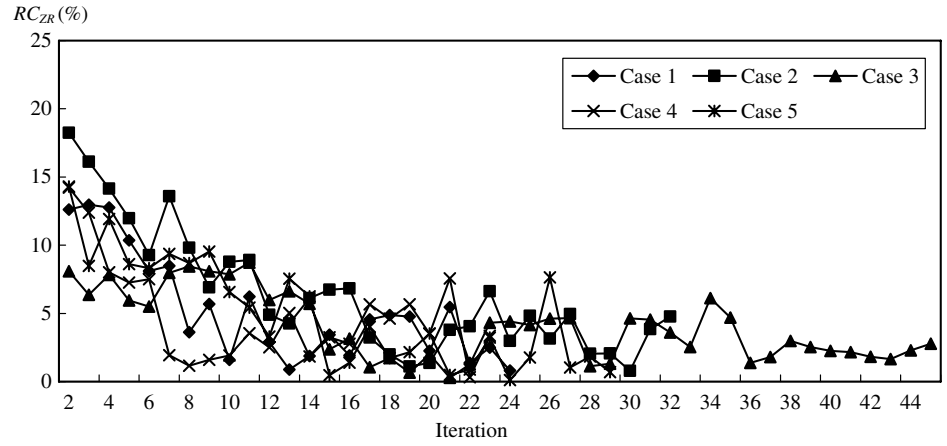

Fig. 4. RC_{ZR} results (framework with PAM2).

For ease of comparing the framework's performance with the DFDGAM's, the improvement percentage of ZR (I_{ZR}), the improvement percentage of AWT (I_{AWT}), the improvement percentage of AIT (I_{AIT}), the improvement percentage of SRMR (I_{SRMR}) in Table 1 are defined:

$$I_{ZR}\ (\%) = \frac{|\text{ZR of framework} - \text{ZR of DFDGAM}|}{\text{ZR of DFDGAM}}, \tag{26}$$

$$I_{AWT}\ (\%) = \frac{|\text{AWT of framework} - \text{AWT of DFDGAM}|}{\text{AWT of DFDGAM}}, \tag{27}$$

$$I_{AIT}\ (\%) = \frac{|\text{AIT of framework} - \text{AIT of DFDGAM}|}{\text{AIT of DFDGAM}}, \tag{28}$$

$$I_{SRMR}\ (\%) = \frac{|\text{SRMR of framework} - \text{SRMR of DFDGAM}|}{\text{SRMR of DFDGAM}}. \tag{29}$$

As shown in Table 1, the values of I_{ZR}, I_{AWT}, I_{AIT}, and I_{SRMR} of all five cases were significantly positive, showing that the frameworks with the two PAMs out-performed the DFDGAM. In particular, the average I_{AWT}, the average I_{AIT}, and the average I_{SRMR} of the framework with PAM1 were 23.24% and 18.22%, and 30.78%, respectively. These results indicate that, compared with the DFDGAM, the framework can not only reduce the passenger waiting time and inconsistent time, but will also yield a more "robust" solution, for all 40 scenarios.

Table 1
Test results

	Case					
	1	2	3	4	5	Average
Framework with PAM1						
I_{ZR} (%)	16.93	21.96	38.89	17.46	28.80	24.81
I_{AWT} (%)	7.84	23.78	41.10	16.28	27.21	23.24
I_{AIT} (%)	22.03	13.59	16.01	13.60	25.86	18.22
I_{SRMR} (%)	27.08	27.10	56.44	24.86	35.07	34.11
Solution time (minutes)	176.6	117.2	149.2	197.8	103.4	148.8
Framework with PAM2						
I_{ZR} (%)	17.81	17.31	25.56	21.74	21.61	20.81
I_{AWT} (%)	6.25	9.80	7.66	11.29	2.53	7.51
I_{AIT} (%)	34.87	27.22	33.55	25.36	35.39	31.28
I_{SRMR} (%)	13.30	22.72	43.06	37.10	37.70	30.78
Solution time (minutes)	44.3	56.3	126.6	93.1	82.7	80.6

By comparing the two PAMs, we found that the framework with PAM1 generally yielded better values of ZR than did the framework with PAM2. However, the former was more time-consuming to solve than the latter. The average solution times of the five cases for the framework with PAM1 and PAM2 were 148.8 and 80.6 minutes, respectively. These results show that the framework with PAM1 is superior in terms of solution quality; but the framework with PAM2 is superior in terms of computational efficiency. Consequently the most suitable choice of framework would depend upon each user analyses in practice. In addition, the shortest time of all five cases, (case 1 of the framework with PAM2) and the longest time (case 4 of the framework with PAM1) were about 44 and 197 minutes, respectively, which are efficient in the planning stage.

Using the above evaluation method, we compared the performance of the current manual assignment process, the DFDGAM, and the frameworks with both PAMs. For ease of comparison the ZR, AWT, AIT, and SRMR of the evaluation method were called as simulated ZR, AWT, AIT, and SRMR, respectively. I_{ZR}, I_{AWT}, I_{AIT}, and I_{SRMR} in Table 2 are defined similar to Eqs. (26)–(29), with the simulated ZR, AWT, AIT, and SRMR, replacing ZR, AWT, AIT, and SRMR, respectively. As shown in Table 2, in comparison with the current manual assignment process, the average values of I_{ZR} for the frameworks with the two PAMs were about 52.80% and 50.20%, respectively. As well, the average values of I_{AWT}, I_{AIT}, and I_{SRMR} varied from 40.21% to 59.89%. These results show that frameworks with both PAMs offered a significant improvement over the current manual assignment process, and would thus be helpful to enhance current airport operations. Moreover, compared with the DFDGAM, similar to the aforementioned results, the frameworks with both PAMs also significantly out-performed the DFDGAM after the evaluation method. This indicates that the superiority of the frameworks with both PAMs will not be affected after their planned results are applied in real operations.

3.3. Sensitivity analysis

To understand the performance of the framework in different situations, we performed sensitivity analyses of the following factors: (1) the number of scenarios, (2) the buffer time, (3) the disturbance time, and (4) the weighting vector w, with the same data as for the above five test cases.

Table 2
Evaluation results

	Case					
	1	2	3	4	5	Average
Framework with PAM1 vs. Current assignment						
I_{ZR} (%)	48.50	54.17	54.07	52.00	55.24	52.80
I_{AWT} (%)	44.50	58.82	42.50	54.78	52.49	50.62
I_{AIT} (%)	54.85	53.10	66.10	48.40	58.56	56.20
I_{SRMR} (%)	47.17	42.38	43.16	50.92	55.98	47.92
Framework with PAM2 vs. Current assignment						
I_{ZR} (%)	51.03	48.56	48.54	52.32	50.56	50.20
I_{AWT} (%)	47.93	51.31	29.51	51.87	44.05	44.93
I_{AIT} (%)	57.66	55.85	67.57	56.50	61.89	59.89
I_{SRMR} (%)	46.67	29.16	32.91	47.02	45.29	40.21
Framework with PAM1 vs. DFDGAM						
I_{ZR} (%)	17.34	23.74	33.41	20.04	26.42	24.19
I_{AWT} (%)	0.39	29.91	33.08	18.11	23.99	21.10
I_{AIT} (%)	28.80	12.47	16.50	13.12	15.79	17.33
I_{SRMR} (%)	32.40	22.37	51.68	32.22	44.87	36.71
Framework with PAM2 vs. DFDGAM						
I_{ZR} (%)	21.39	14.41	25.38	20.57	18.73	20.10
I_{AWT} (%)	6.54	17.12	17.96	12.85	10.49	12.99
I_{AIT} (%)	33.23	17.61	20.11	26.75	22.56	24.05
I_{SRMR} (%)	31.77	4.56	42.97	26.84	31.47	27.52

3.3.1. The number of scenarios

We tested 10 situations, with 2, 5, 8, 10, 20, 30, 40, 50, 60, and 70 scenarios. To save space, we only show the results for the framework with PAM1. The results for the framework with PAM2 were similar. In addition, to measure the difference between the ZR and the simulated ZR in each situation, we also estimated the evaluation gap (EG), which is defined as follows:

$$\text{EG}\ (\%) = \frac{|\text{ZR} - \text{simulated ZR}|}{\text{ZR}}. \tag{30}$$

As shown in Fig. 5, when the number of scenarios increased, the values of ZR for all five cases increased. However, after 40 scenarios, the values of ZR did not change much, showing that the solution results stabilized after 40 scenarios. In addition, as shown in Fig. 6, when the number of scenarios was less than 40, the values of EG rose sharply, the average values of EG increased from 21.03% (30 scenarios) to 1198.89% (2 scenarios). After 40 scenarios the average values of EG only varied from 4.68% to 6.87%. These results indicate that a smaller number of scenarios would produce a larger difference between the values of ZR and the values of simulated ZR, although a smaller number of scenarios yielded better values of ZR than did a larger number of scenarios. Moreover, we found that after 40 scenarios, the solution times significantly increased. In particular, for 40, 50, 60, and 70 scenarios, the average solution times were 148.8, 376.1, 338.0, and 394.5 minutes, respectively. All in all, a set of 40 scenarios was found to be the most suitable for representing the events, in

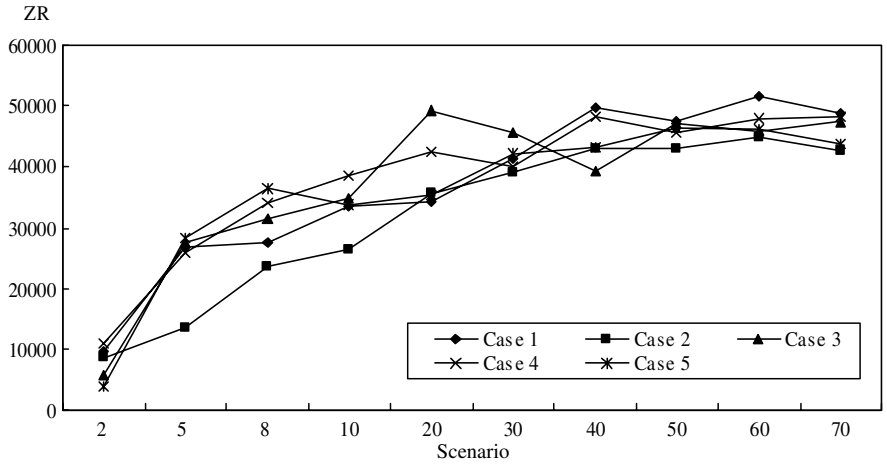

Fig. 5. Values of ZR with a changing number of scenarios (framework with PAM1).

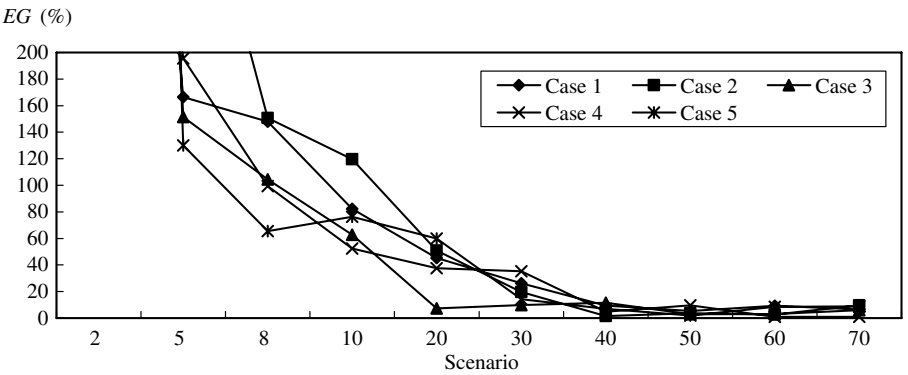

Fig. 6. Values of EG with a changing number of scenarios (framework with PAM1).

terms of solution accuracy and computational efficiency. Based on this, the airport authority could perform similar tests on its own operations to determine a suitable number of scenarios.

3.3.2. The buffer time

We set buffer times from 10 to 50 minutes. Since the results for the frameworks with the two PAMs were similar, only the results for the framework with PAM2 are discussed here. The average values of ZR and the average values of simulated ZR for the five cases are shown in Fig. 7. We found that when the buffer time was 30 minutes, both the average values of ZR and the average values of simulated ZR were the smallest. When the buffer time was larger or less than 30 minutes, both the average values of ZR and the average values of simulated ZR tended to increase. These results indicate that, based on current CKS operations, the framework would have the best performance, with a 30-minute buffer time. In addition, we found that as the buffer time decreased, the solution time increased. In particular, when the buffer time decreased from 50 to 10 minutes, the solution time increased from 52.5 to 305.7 minutes. This is probably because a short buffer time increases the number of connecting arcs, meaning that a problem size with a shorter buffer time is larger than that with a longer one.

3.3.3. The disturbance time

We tested five scenarios, using 10–50-minute disturbance times. To save space, we only show the values of I_{ZR} for the ZR and the simulated ZR, for the framework with PAM2, which were similar to the framework with PAM1 results. As can been seen in Table 3, for most of the five cases, the values of I_{ZR} for both the ZR and the simulated ZR increased as the disturbance time increased. Their average values of I_{ZR} were increased from 15.07% to 26.24% and from 14.03% to 22.46%, respectively. We can see from these results that the framework performed well irregardless of the disturbance time. As for the solution time, all five cases were solvable within 210 minutes irregardless of the different disturbance times, showing that the disturbance time did not affect the computational efficiency of the framework.

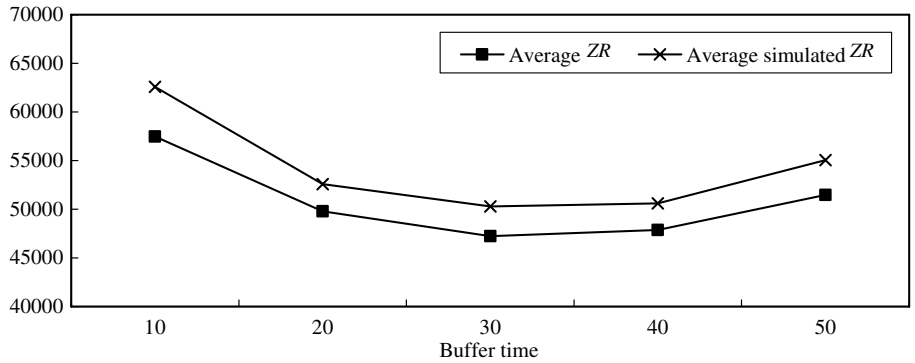

Fig. 7. Average values of ZR and simulated ZR with changing buffer times (framework with PAM2).

Table 3
Values of I_{ZR} with changing disturbance times (framework with PAM2)

Disturbance time	I_{ZR} (%) for ZR Case						I_{ZR} (%) for simulated ZR Case					
	1	2	3	4	5	Average	1	2	3	4	5	Average
10	12.24	18.27	19.17	10.62	15.03	15.07	11.39	18.75	18.11	7.90	14.02	14.03
20	12.23	20.27	19.57	14.83	23.15	18.01	12.27	16.94	21.56	11.75	19.64	16.43
30	17.81	17.31	25.56	21.74	21.61	20.81	21.39	14.41	25.38	20.57	18.73	20.10
40	18.97	25.34	26.76	23.25	30.35	24.93	19.82	19.48	21.47	19.86	24.55	21.04
50	25.43	28.33	24.03	24.29	29.12	26.24	24.19	23.96	20.78	22.21	21.15	22.46

3.3.4. The weighting vector w

We tested five w values, from 1 to 5. To save space, only the results for the framework with PAM2 are discussed. The values of I_{ZR} for the ZR and the simulated ZR are shown in Table 4. We found that when the w value increased, the values of I_{ZR} changed without any specific trend. However, in all five cases, the values of I_{ZR} for both the ZR and the simulated ZR were significant. Their average values of I_{ZR} varied from 17.41% to 22.00% and from 14.08% to 20.10%, respectively. These results indicate that the framework out-performed the DFDGAM, irregardless of the w values. In addition, we found that as the w values increased, the required solution times varied without any specific trend. The longest time, for case 4 with a w value 3, was still within 280 minutes, which is acceptable for the planned gate assignment problems. In other words, the computational efficiency of the framework was not much affected by the w values.

3.4. The flight delay patterns

To evaluate the influence of different flight delay patterns on the DFDGAM and the frameworks with the two PAMs, we tested various flight delay patterns based on Yan et al. (2002). They designed 11 flight delay patterns, which were denoted as scenarios 0–10. Scenario 0 is the original delay and the other scenarios are classified into three types of delays, low (scenarios 1–3), medium (scenarios 4–7), and high (scenarios 8–10) delays. For ease of testing, we selected four flight delay patterns, scenarios 0, 1, 5, and 10, with respect to four delay types (including the original delay). The procedure is not described in detail here, because it is not the focus in this research. The interested reader can refer to Yan et al. (2002) for a more detailed description. Nevertheless, it should be mentioned that an airport authority could generate, design and test other types of delay distributions suitable to its own operations.

Since the results for the frameworks with the two PAMs were similar, to save space, in Table 5 we only show the results for the framework with PAM2. We found that for all five cases, when the flight delay was heightened to the high delay pattern, the values of I_{ZR} for the ZR tended to decrease. The average values of I_{ZR} for the ZR decreased from 20.41% to 7.86%. Similar results were also found for the values of I_{ZR} for the simulated ZR. However, the values of I_{ZR} were all positive irregardless of the delay patterns, showing

Table 4
Values of I_{ZR} with changing w values (framework with PAM2)

w	I_{ZR} (%) for ZR Case						I_{ZR} (%) for simulated ZR Case					
	1	2	3	4	5	Average	1	2	3	4	5	Average
1	14.00	22.45	19.43	18.03	13.44	17.47	12.52	16.11	16.64	18.51	11.36	15.03
2	17.81	17.31	25.56	21.74	21.61	20.81	21.39	14.41	25.38	20.57	18.73	20.10
3	15.98	18.31	16.28	12.88	25.21	17.73	19.23	13.75	11.36	9.04	20.35	14.75
4	20.10	23.28	25.10	17.90	23.64	22.00	20.82	23.58	18.02	16.16	18.77	19.47
5	13.52	14.57	22.73	18.10	20.13	17.81	17.17	8.69	13.46	16.41	14.67	14.08

Table 5
Values of I_{ZR} with different flight delay patterns (framework with PAM2)

Flight delay patterns	I_{ZR} (%) for ZR Case						I_{ZR} (%) for simulated ZR Case					
	1	2	3	4	5	Average	1	2	3	4	5	Average
Original	17.81	17.31	25.56	21.74	21.61	20.81	21.39	14.41	25.38	20.57	18.73	20.10
Low delay	15.59	27.73	9.71	25.45	23.56	20.41	20.64	25.88	6.66	21.81	23.71	19.74
Medium delay	13.50	10.13	6.99	11.39	12.14	10.83	13.37	10.39	5.57	11.59	12.20	10.63
High delay	11.63	6.47	9.38	5.00	6.83	7.86	11.69	7.01	8.94	5.83	5.77	7.85

that the values of ZR were better for the framework than for the DFDGAM, irregardless of the flight delay patterns. This also indicates that the framework is more useful than the DFDGAM for solving gate assignment plans in different types of stochastic flight delay situations, although the framework's effectiveness is lower for the higher flight delay pattern.

4. Discussion and conclusions

In this research, we developed a heuristic approach embedded in a framework that integrated the planning and real-time stages, to solve gate assignment problems under stochastic flight delays. The framework included three components, the SFDGAM, the reassignment rule, and two PAMs. Numerical tests, utilizing data from a Taiwan international airport's operations, were performed to evaluate the framework. During the testing process, several sensitivity analyses were also performed, coupled with the application of different flight delay patterns, to demonstrate the flexibility of this framework. Some findings regarding the development of the framework and the test results are summarized and discussed.

In practice, the reassignment process is usually performed based on a reassignment rule by the airport authority, making integrated problems difficult to formulate in a closed form. The theoretically optimal solution or a proper lower bound of the integrated problem is thus difficult to explore. Since it is difficult to extend the current optimization methods to deal with the integrated problem, we aim to develop a heuristic approach embedded in a framework that can efficiently solve the integrated problem. The test results show that the framework allowed for significant improvements over the current manual assignment process and the DFDGAM in stochastic flight delay situations.

Although the framework with PAM1 and PAM2 were shown to be good in terms of solution quality and computational efficiency, the most suitable PAM choice for the framework may be considered according to the user's analyses in practice. In addition, the best stopping criterion is: if a better solution than the incumbent one is not found after eight iterations, then the solution process may be stopped. Based on the data from the current CKS Airport operations, the longest solution time, 197 minutes, is required for the framework, which is efficient in the planning stage.

We found that after 40 scenarios, the solution results trended to stabilize, but the solution times significantly increased. This indicates that in terms of solution accuracy and computational efficiency a set of 40 scenarios was the most suitable. The sensitivity analysis of the buffer time for current CKS Airport operations indicates that a 30-minute buffer time offers the best performance. The solution times increase when the buffer time is shorter, probably because a problem size with a shorter buffer time is larger than that with a longer one. In addition, the sensitivity analysis of the disturbance time shows that both the solution quality and computational efficiency of the framework perform well irregardless of the disturbance time. Similar results were also found for the analyses of the weighting vector w and the flight delay pattern. That is, the performance will not be affected by these problem parameters.

Finally, in future we hope to explore an integrated model or framework with a closed form, so that related optimization theories or issues can be investigated. Nevertheless, the heuristic approach and the test results discussed in this research should be useful as a reference for the development of such an integrated model or framework.

Acknowledgements

This research was supported by a grant (NSC-93-2211-E-008-022) from the National Science Council of Taiwan. We thank the CKS Airport personnel for providing the test data and their valuable opinions. We also thank an anonymous referee for his/her valuable suggestions on the presentation of the paper.

Appendix. The model formulation of the DFDGAM

The DFDGAM is given by

$$\min \quad \sum_{k\in K}\sum_{ij\in A^k} c_{ij}^k x_{ij}^k \tag{A.1}$$

$$\text{s.t.} \quad \sum_{j\in N^k} x_{ij}^k - \sum_{r\in N^k} x_{ri}^k = 0 \quad \forall i \in N^k,\ \forall k \in K, \tag{A.2}$$

$$\sum_{k\in K}\sum_{ij\in F_t} x_{ij}^k = 1 \quad \forall t \in \mathrm{AF}, \tag{A.3}$$

$$0 \leqslant x_{ij}^k \leqslant g^k \quad \forall (i,j) \in \mathrm{CA}^k,\ \forall k \in K, \tag{A.4}$$

$$x_{ij}^k \in Z_+ \quad \forall (i,j) \in \mathrm{CA}^k,\ \forall k \in K, \tag{A.5}$$

$$x_{ij}^k = 0,1 \quad \forall (i,j) \in A^k - \mathrm{CA}^k,\ \forall k \in K. \tag{A.6}$$

The objective of this model is to simultaneously "flow" all arcs that cycle within each network at a minimum cost. Since only the alternate flight arcs have a positive cost (i.e., the passenger waiting time), this objective is equivalent to minimizing the total passenger waiting time. The other constraints are the same as those in the SFDGAM.

References

Babic, O., Teodorovic, D., Tosic, V., 1984. Aircraft stand assignment to minimize walking. Journal of Transportation Engineering 110, 55–66.

Bertsimas, D., Sim, M., 2004. The price of robustness. Operation Research 52, 35–53.

Bihr, R.A., 1990. A conceptual solution to the aircraft gate assignment problem using 0, 1 linear programming. Computers and Industrial Engineering 19, 280–284.

Bolat, A., 1999. Assigning arriving flights at an airport to the available gates. Journal of the Operational Research Society 50, 23–34.

Bolat, A., 2000. Procedures for providing robust gate assignments for arriving aircrafts. European Journal of Operational Research 120, 63–80.

Braaksma, J.P., 1977. Reducing walking distance at existing airports. Airport Forum, 135–145.

Cheng, Y., 1997. A knowledge-based airport gate assignment system integrated with mathematical programming. Computers and Industrial Engineering 32, 837–852.

Cheung, R.K.M., Powell, W.B., 1996. Models and algorithms for distribution problems with uncertain demands. Transportation Science 30, 43–59.

Du, Y., Hall, R., 1997. Fleet sizing and empty equipment redistribution for center-terminal transportation networks. Management Science 43, 145–157.

Escudero, L.F., Kamesam, P.V., King, A.J., Wets, R.J.-B., 1993. Production planning via scenario modeling. Annals of Operations Research 43, 311–335.

Gosling, G.D., 1990. Design of an expert system for aircraft gate assignment. Transportation Research A 24, 59–69.

Gu, Y., Chung, C.A., 1999. Genetic algorithm approach to aircraft gate reassignment problem. Journal of Transportation Engineering 125, 384–389.

Gutierrez, G.J., Kouvelis, P., 1995. A robustness approach to international sourcing. Annals of Operations Research 59, 165–193.

Haghani, A., Chen, M.C., 1998. Optimizing gate assignments at airport terminals. Transportation Research A 32, 437–454.

Hamzawi, S.G., 1986. Management and planning of airport gate capacity: A microcomputer-based gate assignment simulation model. Transportation Planning and Technology 11, 189–202.

Kenyon, A.S., Morton, D.P., 2003. Stochastic vehicle routing with random travel times. Transportation Science 37, 69–82.

List, G.F., Wood, B., Nozick, L.K., Turnquist, M.A., Jones, D.A., Kjeldgaard, E.A., Lawton, C.R., 2003. Robust optimization for fleet planning under uncertainty. Transportation Research E 39, 209–227.

Mangoubi, R.S., Mathaisel, D.F.X., 1985. Optimizing gate assignment at airport terminals. Transportation Science 19, 173–188.

Mulvey, J.M., Ruszczynski, A., 1995. A new scenario decomposition method for large-scale stochastic optimization. Operations Research 43, 477–490.

Mulvey, J.M., Vanderbei, R.J., Zenios, S.A., 1995. Robust optimization of large-scale systems. Operations Research 43, 254–281.

Paraskevopoulos, D., Karakitsos, E., Rustem, B., 1991. Robust capacity planning under uncertainty. Management Science 37, 787–800.

Rosenberger, J.M., Johnson, E.L., Nemhauser, G.L., 2004. A robust fleet-assignment model with hub isolation and short cycles. Transportation Science 38, 357–368.

Ruszczynski, A., Shapiro, A., 2003. Stochastic Programming. Elsevier, Amsterdam.

Soteriou, A.C., Chase, R.B., 2000. A robust optimization approach for improving service quality. Manufacturing & Service Operations Management 2, 264–286.

Su, Y.Y., Srihari, K., 1993. A knowledge based aircraft-gate assignment advisor. Computers and Industrial Engineering 25, 123–126.

Vanderstraetan, G., Bergeron, M., 1988. Automatic assignment of aircraft to gates at a terminal. Computers and Industrial Engineering 14, 15–25.

Yan, S., Chang, C.M., 1998. A network model for gate assignment. Journal of Advanced Transportation 32, 176–189.
Yan, S., Huo, C.M., 2001. Optimization of multiple objective gate assignments. Transportation Research A 35, 413–432.
Yan, S., Young, H.F., 1996. A decision support framework for multi-fleet routing and multi-stop flight scheduling. Transportation Research A 30, 379–398.
Yan, S., Shieh, C.W., Chen, M., 2002. A simulation framework for evaluating airport gate assignments. Transportation Research A 36, 885–898.
Yu, C., Li, H., 2000. A robust optimization model for stochastic logistic problems. International Journal of Production Economics 64, 385–397.
Zhang, S.X., Cesarone, J., Miller, F.G., 1994. A comparative study of an aircraft assignment problem at a large aircraft. International Journal of Industrial Engineering 1, 203–212.