Optimization for gate re-assignment

Dong Zhang (zd.pony@gmail.com)
Department of Industrial and Manufacturing Systems Engineering
University of Hong Kong, Hong Kong
Diego Klabjan (d-klabjan@northwestern.edu)
Department of Industrial Engineering and Management Sciences
Northwestern University, Evanston, IL

Abstract: Disruptions such as adverse weather, flight delays and flight cancellations are a frequent occurrence in airport operations. A sophisticated gate assignment plan can be easily disrupted and serious consequences might be caused. Therefore, an efficient gate re-assignment methodology is of great importance for the airline industry. In this paper, we propose *an efficient gate re-assignment methodology* to deal with the disruptions, in which the objective function is to minimize the weighted sum of the total flight delays, the number of gate re-assignment operations and the number of missed passenger connections. Two multi-commodity network flow models are built for the pure gate re-assignment problem and the gate re-assignment problem with connecting passengers. Recognizing the inherent NP hard nature of the gate re-assignment problem, two heuristic algorithms are proposed to solve the models efficiently. The proposed models and algorithms are tested based on real-world data of a large U.S. carrier and computational results reveal that the proposed methodologies can provide high quality solutions within a short computational time.

Key words: gate re-assignment; multi-commodity network flow model; diving heuristic; rolling horizon.

1 Introduction

A good gate assignment plan plays an important role in utilizing gates efficiently and improving passengers' satisfaction. However, disruptions such as adverse weather, flight delays, flight cancellations, etc. are a frequent occurrence and a sophisticated gate assignment plan is easily disrupted. Serious consequences might be caused if an efficient gate re-assignment plan is not available after the disruptions. Therefore, efficient and effective methodologies for the gate re-assignment problem after the disruptions are of great importance to maintain high service quality.

In the U.S. airline industry, gates are leased to airlines and thereby the airlines are responsible for planning gate operations, which is referred to as an airline-specific gate system [1]. Some Asian airports have also adopted the airline-specific gate system. The alternative gate system, used in European and some Asian airports, is the airport-specific gate system, where the gates belong to airports and the airports are responsible for planning the gate operations. In this study, we are focusing on gate operations in the U.S. airline industry, specifically, the gate re-assignment

problem in the airline-specific gate system. In the context of the U.S. airline industry, gate controllers of the airlines are in charge of the gate re-assignment planning. Inputs to gate re-assignment planning are the updated flight schedule and the current gate assignment plan. In the updated flight schedule, flights might be delayed or cancelled, which implies that the current gate assignment plan might not be feasible. Once infeasibility is detected by the gate controllers, a gate re-assignment process is triggered. In current practice, the gate re-assignment process is mostly conducted manually by the experienced gate controllers. They adjust the current gate assignment plan with some computer aided tools, mostly focusing on visualization.

Two main objectives are typically taken into consideration: minimizing total flight delays and minimizing the number of gate re-assignment operations. Flight delays are incurred due to holding either arrival or departure flights of an aircraft. In the gate re-assignment problem, an aircraft corresponds to one arrival flight and one departure flight. If an aircraft is held on the ground waiting for an available gate, holding time is defined as the delay of the arrival flight. If an aircraft is held at a gate to wait for late connecting passengers, holding time is defined as the delay of the departure flight. The total flight delays are defined as the summation of all flights' delays. The number of gate re-assignment operations indicates the number of aircraft not assigned to their initially planned gate.

Furthermore, gate related constraints, such as a gate can only accommodate certain aircraft types and two large aircraft cannot be assigned to two adjacent gates simultaneously, must be satisfied in the gate re-assignment process.

When the number of aircraft involved is low, the gate controllers can efficiently produce a gate re-assignment plan by trying different options. As the number of aircraft involved increases, the number of possible combinations is increasing significantly. It becomes hard to produce a gate re-assignment solution efficiently. Therefore, the airport gate controllers are in need of systematic optimization methodologies to help them efficiently solve the gate re-assignment problem.

In this research, we propose a novel optimization methodology for the gate re-assignment problem with four different options considered: (1) Re-assigning an aircraft to an alternative gate (including the parking area); (2) Delaying an aircraft's gate arrival time in parking to wait for an available gate; (3) Delaying an aircraft's push back time to wait for late transfer passengers; (4) Towing an aircraft to a parking area if there is a long waiting time between the arrival and departure flight. Two multi-commodity network flow models are formulated to consider the pure gate re-assignment problem and the gate re-assignment problem with connecting passengers, respectively. In addition to the two aforementioned objectives (minimizing total flight delays and the number of gate re-assignment operations), minimizing total passenger transfer distance and the number of missed passenger connections are included in the objective function. In both models, each gate is considered as a commodity and thus corresponds to one network. In each gate's network, a feasible sequence of aircraft assigned to it is defined as a flow in the network.

The first model can be efficiently solved by a commercial MIP solver, such as CPLEX. However, it is very challenging to solve the second model efficiently by a commercial solver due to its size and the quick response requirement. Therefore, two heuristic algorithms are proposed to solve the second model. The first heuristic algorithm is a guided diving heuristic algorithm based on general upper bound branching, in which at each iteration, multiple aircraft are selected and each of them is fixed to a gate clique based on the linear relaxation solution. If every aircraft is

already assigned to a gate clique, the restricted mixed integer programming (MIP) model is then directly solved by a commercial MIP solver. In some situations, the length of the re-assignment time window might be very long and the first heuristic algorithm cannot produce a solution within the pre-specified timeframe. The second heuristic algorithm, a variable rolling horizon algorithm, is then proposed to deal with these situations. It decomposes the whole time window into multiple overlapping intervals with each interval corresponding to a problem of smaller size, which can be efficiently solved by the guided diving heuristic algorithm. Computational results revealed that the proposed algorithms can solve all instances within 5 minutes with the largest optimality gap smaller than 13%.

In literature, there are already several studies focusing on the gate re-assignment problem [2-12]. However, few of them considers connecting passengers within their model formulations and solution algorithms. In a hub-and-spoke network, there are many passengers connecting at hub airports. A gate re-assignment plan ignoring connecting passengers has two negative impacts on the connecting passengers: (1) Delay of the arrival flight can make it impossible to catch another departure flight; (2) Transfer time for a passenger can increase causing insufficient time for the passenger to move from the arrival gate to another departure gate. If a connecting passenger misses his or her connection, high cost (compensation cost, passengers' goodwill loss cost, etc.) is incurred for both airports and airlines. Therefore, it is essential to consider connecting passengers in the gate re-assignment problem to minimize the number of missed passenger connections, and thereby, to minimize the total gate re-assignment cost. To the best of our knowledge, Maharjan et al. [7] considered the transfer passengers in their gate re-assignment model. However, there are two limitations of their study. First, it only considered minimization of the passengers' transfer distance but ignored probabilities of missed passenger connections. If a passenger misses his or her connection, the transfer distance is irrelevant; second, a quadratic model was proposed but no algorithm was provided to solve the model. Recognizing the gap between practical requirements in the industry and current gate re-assignment methodologies, we realize that an efficient methodology for the gate re-assignment problem fully considering factors of connecting passengers is highly desired in the industry.

To fill this gap, we propose a novel methodology for the gate re-assignment problem. Our contributions can be summarized as follows:

(1) A more efficient multi-commodity network flow model is built for the gate re-assignment problem;

(2) An extended multi-commodity network flow model is built where minimizing the passenger transfer distance and minimizing the number of missed passenger connections are both considered;

(3) A diving heuristic algorithm and a rolling horizon algorithm are proposed to efficiently solve the proposed model.

The remainder of the paper is organized as follows. In Section 1.1, a literature review related to the gate assignment/re-assignment is provided. In Section 2, a general description of the gate re-assignment problem is described and two multi-commodity models are built. Two heuristic algorithms are proposed in Section 3. In Section 4, extensive computational experiments are reported. Finally, conclusions are offered in Section 5.

1.1 Literature review

In literature, the gate assignment problem has been extensively discussed and it is explored in two ways: the gate assignment planning problem and the gate re-assignment problem. There are abundant solution methodologies proposed for the gate assignment planning problem. We refer interested readers to Babic et al. [13], Mangoubi et al. [14], Vanderstraeten et al. [15], Bihr et al. [16], Wirasinghe et al. [17], Yan et al. [18], Haghani [19], Balot [20-22], Yan et al. [23-25], Ding [26, 27], Seker et al. [28], Kim [29], Narciso et al. [30], Castaing et al. [31], Diepen et al. [32], Dorndorf et al. [33-35], Guepet et al. [36], Kim et al. [37], Kumar et al. [38], Yu et al. [39] and Ravizza et al. [40] for planning gate assignment works. They typically formulate the gate assignment planning problem as a linear or binary quadratic integer programming model with the objective function as minimization of passengers' transfer distance, number of off-gate events, gate idle rate, expected gate blockages, etc. We refer interested readers to two extensive reviews Dorndorf et al. [41] and Bouras et al. [42] for further details.

Compared to the gate assignment planning problem, the gate re-assignment problem differs in four aspects: (1) Flight delay options are considered, i.e., in the gate re-assignment problem, an aircraft might be held on the ground to wait for an available gate or late passengers; (2) The transfer passengers might miss connections; (In the gate assignment planning problem, enough time must be guaranteed for the connecting passengers. However, in the gate re-assignment problem, the connecting passengers might miss their connections because of flight delays and/or re-assigning aircraft to alternative gates) (3) Minimization of the deviation from the initial gate assignment plan is incorporated into the objective function; (4) A short solution computational time is required, i.e., a gate re-assignment plan is usually required to be provided within several minutes after disruptions. These four differences cause the gate re-assignment problems to be far more challenging.

There are a few methodologies proposed for the gate re-assignment problem. These methodologies can be categorized into two groups: (1) Methodologies based on artificial intelligence; (2) Methodologies based on mathematical programming. In earlier days, optimization theories and computing hardware were highly limited and artificial intelligence was a very promising technology for solving the gate re-assignment problem. Gosling [8], Srihari et al. [9] and Jo [10] proposed expert systems to deal with the gate re-assignment problem. Su et al. [11] proposed a knowledge based advisor to assist gate re-assignment decision making. Cheng [12] proposed a knowledge based gate re-assignment system in which mathematical programming was integrated. Gu et al. [2] developed a genetic algorithm to solve the gate re-assignment problems due to flight delays.

As the optimization theories and computing hardware advanced, it becomes feasible to solve the gate re-assignment problem optimally using mathematical programming techniques. Therefore, several methodologies based on the mathematical programming techniques were proposed in recent years. Yan et al. [3] considered the airport gate re-assignment problem following temporary airport closure and a network flow model was proposed for this problem. In their work, re-assigning aircraft to an alternative gate was considered but flight delays were overlooked. Tang et al. [6] considered the flight delay option and proposed another MIP model for the gate re-assignment problem. They also proposed a dynamic decision process in a practical

setting. Tang et al. [5] further extended the model in [6] and proposed a new gate reassignment model considering stochastic flight delays. A more practical re-assignment model was proposed by Yan et al. [4]. In [4], flights are separated into deterministic and stochastic flights based on the time to the anticipated gate arrival. The model is then built to deal with the deterministic flights and the stochastic flights separately. Maharjan et al. [7] proposed a gate re-assignment model based on the traditional deterministic model to consider connecting passengers. In their work, a quadratic integer programming model is formulated and minimizing the passenger's transfer distance was considered in the objective function. No algorithm was proposed to solve the model.

In this study, we propose two novel multi-commodity network flow models for the pure gate re-assignment problem and the gate re-assignment problem with connecting passengers, respectively. Comparing to previous works, our models have the following advantages:

(1) Our first proposed model is a significant improvement of Yan et al. [3]. One major advantage of such a multi-commodity network flow model is that gate constraints (which account for a large percentage of total constraints) can be more concisely formulated and therefore it can be more efficiently solved by a commercial MIP solver compared to other MIP models. The benefits have also been validated in Section 4.5. Although Yan et al. [3] proposed a similar network flow model, the way of formulating flow balance causes the number of decision variables to grow quickly with the model size and therefore offsets advantages provided by the multi-commodity network flow model.

(2) It is the first time in the literature that the gate re-assignment problem with connecting passengers is formulated as a multi-commodity network flow model, which is expected to be much more tractable than the other models considering passenger transfer, i.e., the quadratic model proposed in [7].

2 Gate re-assignment model

2.1 Problem description

Prior to building a mathematical model, characteristics of the gate re-assignment problem are first explored. We start with definitions of several key elements in the gate re-assignment problem.

(1) *Gate re-assignment time window*: The gate re-assignment time window defines a time interval in which gate re-assignment operations are allowed. The start time of the time window is usually set as the first time when the disruptions occur. The length of the time window is set based on the severity level of the disruptions. It is desired that at the end of the time window the initial gate assignment plan is feasible. If a feasible solution can not be obtained within the time window, a large penalty cost is incorporated to minimize the schedule violation after the time window.

(2) *Aircraft's gate arrival time and push-back time*: The aircraft gate arrival time is defined as the time when the aircraft first arrives at a gate and the aircraft push-back time is defined as time when the aircraft is push-backed away from the gate. The example in Figure 1 illustrates an aircraft gate arrival and push-back times. The gate is occupied by the aircraft between the gate

arrival and push-back time.

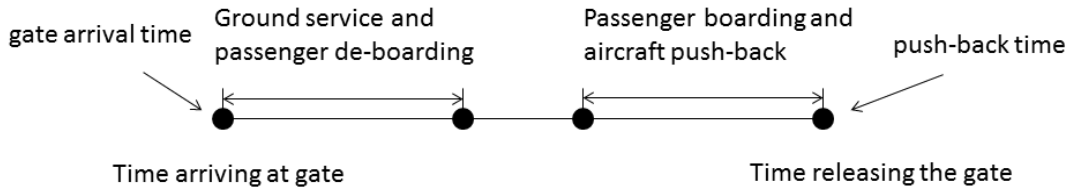

Figure.1 Illustration of the aircraft gate arrival and push-back time

(3) *Involved aircraft in the re-assignment problem*: If an aircraft gate arrival time is located within the re-assignment time window, this aircraft is involved in the re-assignment problem. In the gate re-assignment problem, we only consider these involved aircraft. Aircraft with the gate arrival time before the time window and the push-back time within/after the time window, retain their initial gate assignment plan (made before the time window) and thus they are not involved in the re-assignment problem.

(4) *Gate assignment constraints*: There are three types of gate assignment constraints, which are aircraft constraints, gate constraints and adjacency constraints.

(a) The aircraft constraints restrict that an aircraft type can only be assigned to certain gates. For example, a large aircraft (Boeing 777) cannot be assigned to a gate only for regional jets.

(b) The gate constraints require that in each time point a gate can be occupied by at most one aircraft.

(c) The adjacency constraints require that a specific aircraft pair of specific types cannot be assigned to two adjacent gates simultaneously due to aircraft type characteristics and gate facility limitations.

(5) *Apron gate*: An apron gate is defined as a virtual gate located in the parking area. In some situations, if no gate is available in short time, an aircraft can be assigned to an apron gate in the parking area. A shuttle bus can be arranged to transfer passengers between terminals and the parking area. Since the apron gate will cause inconvenience for both passengers and airlines, the penalty cost should be considered to prevent assigning the aircraft to the apron gate. All of the gate assignment constraints are not applied to the apron gate.

(6) *Flight delays*: There are two types of flight delays in gate assignment operations. First, if an aircraft is held on the ground to wait for an available gate after it lands, then holding time is defined as the delay of its corresponding arrival flight. Second, if an aircraft is held at the gate after its scheduled push-back time to wait for late passengers, then holding time is regarded as the delay of the departure flight. Although the value of the flight delay can be continuous Ideally, discretized values of the flight delay are adopted in our study to built network flow models. Each flight has several different delay options, such as 10 minutes, 20 minutes, 30 minutes, etc.

(7) *Gate re-assignment operations*: In a gate re-assignment solution, an aircraft might not be assigned to its initially scheduled gate, but assigned to an alternative gate. Reassigning the aircraft to an alternative gate is defined as one gate re-assignment operation.

(8) *Gate violations after the re-assignment time window*: It is desirable that there are no gate constraint violations after the time window. However, this might not be possible. One such example is illustrated in Figure 2. Aircraft 1 is involved in the re-assignment problem and delay of the aircraft 1 causes gate violations after the re-assignment time window.

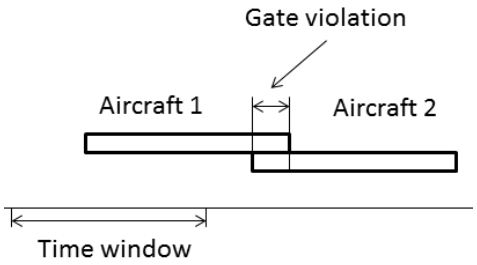

Figure.2 Illustration of gate violations after the re-assignment time window

(9) *Connecting passengers*: In a hub-and-spoke airline network, many passengers connect at hub airports. One connection of a connecting passenger is defined as a pair of arrival and departure flights. If the connection time between the arrival and departure time is not sufficient to transfer from the gate of the arrival flight to the gate of the departure flight, the passenger's connection is missed.

(10) *Objective*: There are four components considered in the objective function of the gate re-assignment problem and a weighted sum of them is minimized. The four components are as follows:

(a) The total flight delays;

(b) The number of gate re-assignment operations;

(c) The number of missed connections; and

(d) The conflicts after the re-assignment time window.

2.2 Multi-commodity network flow model

In literature, the gate re-assignment problem is typically modeled as an assignment model with side constraints [4-6] or a multi-commodity network flow model Yan et al. [3]. The assignment model explores an optimal assignment plan, in which each aircraft is assigned to a gate. Side constraints are incorporated to force that each gate is occupied by at most one aircraft at each time point and adjacency constraints are also satisfied. The multi-commodity network flow model explores a feasible flow for each gate wherein each feasible flow represents the aircraft sequence assigned to this gate. Similarly, gate occupancy and adjacency constraints should be satisfied. Our early computational experiments revealed that either the assignment model or the multi-commodity network flow model achieve a tight lower bound at the root node (solving the root node LP relaxation), and therefore the efficiency of solving the models mainly depends on the efficiency of solving the LP relaxations.

Comparing to the assignment model, one major difference (advantage) of the multi-commodity network flow model is that gate constraints (each gate can only by occupied by at most one aircraft at any time point) can be more concisely formulated as feasibilities of flow balances automatically satisfy gate constraints, while for the assignment model gate constraints are formulated as covering constraints. The number of non-zeros produced by the flow balance constraints in the network flow model is much smaller than those produced by the covering constraints in the assignment model. As the number of non-zeros in an LP model has a significant impact on the efficiency, it implies that the multi-commodity network flow model should be more efficient. This is clearly confirmed in Section 4.

Recognizing the inherent advantage of the multi-commodity network flow model, in this section

we propose a novel enhancement to the multi-commodity network flow model for the gate re-assignment problem in which a different way of formulating the flow balance is adapted to make it much more concise than Yan's model [3] which is of the same type. Computational results produced in Section 4 reveal that our proposed model achieves a much better computational performance.

In our network flow model, each gate is considered as one commodity. The apron gate corresponding to parking is considered as a regular gate exempt from all gate constraints. A sequence of aircraft assigned to a gate is considered as a flow in the gate's network. Figure 3 describes an example of a gate's network. Possible delays of an aircraft are considered by modeling for the aircraft many potential gate arrival and push-back times, e.g. every 5 minutes.

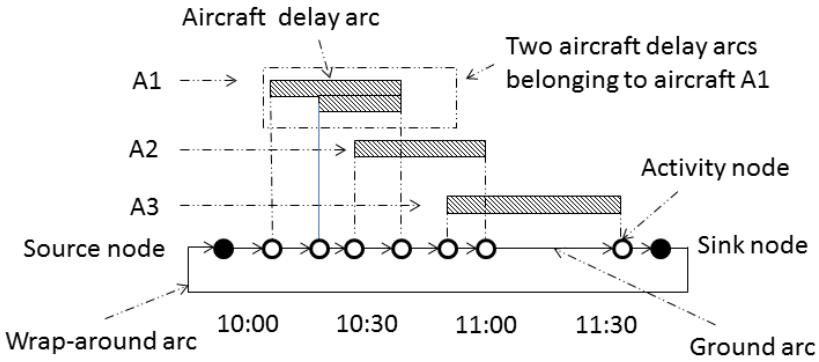

Figure.3 Network flow model and aircraft delay arcs for a gate

There are five essential components in the network for one gate.

(1) *Activity nodes*: An activity node indicates either a gate arrival or push-back activity. Each activity node encodes a gate and time. The gate indicates the location where the activity occurs. If the activity is the arrival activity, the time is the actual gate arrival time; otherwise, the time is the actual push-back time. If two events have the same gate and time, they are regarded as identical activity nodes, i.e, two aircraft arriving at the same time generate only one activity node corresponding to the arrival time in the gate's network. In other words, two (aircraft delay) arcs in the gate's network corresponding to these two aircraft connect to the same activity node.

(2) *Aircraft delay arcs*: An aircraft delay arc uniquely corresponds to one aircraft and one gate. It is defined as a time interval in which the corresponding gate would be occupied by the corresponding aircraft. The start time of the aircraft delay arc indicates the actual gate arrival time and the end time indicates the actual push-back time. Since each flight has several delay options, there are several different options of the gate arrival and push-back times. In a gate's network, an eligible aircraft can create multi flight delay arcs by taking any possible combination of the gate arrival and push-back times. An aircraft is feasible at a gate only if the aircraft type can park at this gate.

(3) *A source and sink node*: The single gate network has exactly one source node and one sink node. The source node indicates the start of gate operations while the sink node indicates the end of the gate operations. The time of the source node should be earlier than the time of any activity node in the network. The time of the sink node should be later than the time of any activity node in the network.

(4) *Ground arcs*: A ground arc is defined to represent a time interval in which the gate is idle. It has three properties: the gate, the start time and the end time. There is a ground arc between any two consecutive activity nodes.

(5) *The wrap-around arc*: The wrap-around arc connects the sink and source nodes. The upper bound on the flow is one since at most one aircraft can be assigned to the gate at each point in time. The purpose of the arc is to keep the flow balance in the gate's network.

A feasible flow in a gate's network consists of a sequence of aircraft delay and ground arcs. One selected aircraft delay arc indicates that the gate is occupied by an aircraft and one selected ground arc indicates that the gate is idle at the underlying times.

The cost of an aircraft delay arc consists of four parts:

(1) The cost of flight delays: An aircraft delay arc corresponds to one arrival and one departure flight. The cost of delays depends on the arrival and departure flights.

(2) The cost of gate re-assignment operations: If the assigned gate of an aircraft is not the initially planned one, the cost of re-assignment operations is incurred.

(3) The cost of gate violations after the re-assignment time window: If the end of an aircraft delay arc violates gate operations planned after the time horizon, the extra cost after the re-assignment time window is incurred based on the length of the violated period.

(4) The cost of assigning aircraft to the apron gate: If an aircraft is assigned to an apron gate, the passenger service is disrupted. Additional operating cost is incurred to transfer passengers between terminals and the parking area.

In the following, we describe the multi-commodity network flow model (**MCGR**) for the gate re-assignment problem. The notation required to build the model is described in Appendix C. In each gate's network, a feasible flow is maintained. Flow balance of the network is guaranteed by keeping the flow balance at each activity node and consistency constraints are incorporated to guarantee that each aircraft is assigned to exactly one gate.

The decision variables used in the model are:

x_e: amount of flow on aircraft delay arc e

y_g: amount of flow on ground arc g

MCGR Model:

$$\text{Min } \sum_{e\in E} c_e \cdot x_e \tag{1}$$

Subject to:

$$\sum_{e\in E_a} x_e = 1 \qquad a \in A \tag{2}$$

$$\sum_{e\in E_{(\pi_1,\rho_1,t)}\cup E_{(\pi_2,\rho_2,t)}} x_e \leq 1 \qquad t \in T,\ (\pi_1,\rho_1,\ \pi_2,\rho_2) \in S \tag{3}$$

$$\sum_{e\in E^n_{input}} x_e + \sum_{g\in G^n_{input}} y_g - \sum_{e\in E^n_{output}} x_e - \sum_{g\in G^n_{output}} y_g = 0 \qquad n \in N \tag{4}$$

$$x_e \in \{0,1\} \qquad y_g \in \{0,1\}$$

Objective (1) minimizes the total cost of selected aircraft delay arcs. Constraints (2) restrict that each aircraft is assigned to exactly one gate and constraints (3) impose the adjacency constraints. Finally, constraints (4) restrict the flow balance at each node.

We finish this part by comparing Yan [3]'s multi-commodity network flow model with ours. To the best of our knowledge, only Yan [3] proposed a multi-commodity network flow model for the gate re-assignment problem in the literature. As aforementioned, Yan et al. [3]'s model did not consider flight delay options, however, it is very easy to extend their model to consider flight delay options by incorporating flight delay arcs (similar to our model) for each flight. The two models differ in the way the flow balance constraints are built. We create ground arcs to build

the flow balance while Yan's model creates connection arcs to build the flow balance. An example reporting the size of the models is illustrated next. Let us suppose there are 200 aircraft and 70 gates involved in the re-assignment problem. Each aircraft can be assigned to any gate and an aircraft can create 7 different aircraft delay arcs in each gate's network. Based on these assumptions, there are 200×7=1,400 aircraft delay arcs and 199×100×7×7=975,100 connection arcs created in Yan's model in the worst case. In total there are 1400×70=98,000 flight delay arcs and 975,100×70=68,257,000 connection arcs created by Yan's model.

In our model, the number of aircraft delay arcs is the same as in Yan's model. However, the number of ground arcs in our model is significantly smaller than the number of connection arcs in Yan's model. In a single gate's network, the number of aircraft delay arcs is 1,400 and then there are at most 2,800 activity nodes created. Considering the source and sink nodes, the number of nodes is 2,802. In total, there are at most 2,802 ground arcs created. Considering all gates, there are at most 98,000 flight delay arcs and 2,802×70=196,140 ground arcs in our model. Apparently, our model is much more concise.

Although in Yan's model, we list an upper bound on the number of decision variables, comparisons of scalability for different instances are reported in Section 4. They further reveal the advantage of our proposed model.

2.3 Multi-commodity network flow model with connecting passengers

We next extend the MCGR model to consider passengers. There are three types of passengers: (1) departing passengers; (2) arriving passengers, and (3) connecting passengers.

For the departing and arriving passengers, we only need to consider the time for them from a gate to the closest terminal entrance point. The cost of an aircraft delay arc can be easily augmented to consider the traveling cost of the departing and arriving passengers.

It is much trickier to consider connecting passengers. For them, we need to consider both the transfer distances and possible missed connections. The transfer distance of a connecting passenger is decided by the arrival flight's gate and the departure flight's gate. The transfer time of a connecting passenger is decided by the gate arrival time of the arrival flight and the push-back time of the departure flight. If a connecting passenger cannot transfer from the arrival to the departure gate before the push-back time of the departure flight, the connecting passenger misses his or her connection.

The straightforward way to consider the connecting passengers is to model the problem as a quadratic 0-1 integer programming model which hinders the requirement of short computational times. In this section, we propose a multi-commodity network flow model for the gate re-assignment problem in which minimization of passengers' transfer distance and the number of missed passenger connections are considered.

An airport is typically divided into several different terminals. In the following, Y indicates the set of all terminals and it is indexed as γ. Value G_γ indicates the set of all gates in terminal γ. For each terminal γ, we divide gate set G_γ into different gate cliques. A gate clique is defined as a group of gates where the transfer time between any two gates is very short (e.g. 5 minutes).

An example is illustrated in Figure 4. In the terminal, we can divide all gates into three gate

cliques based on their physical location.

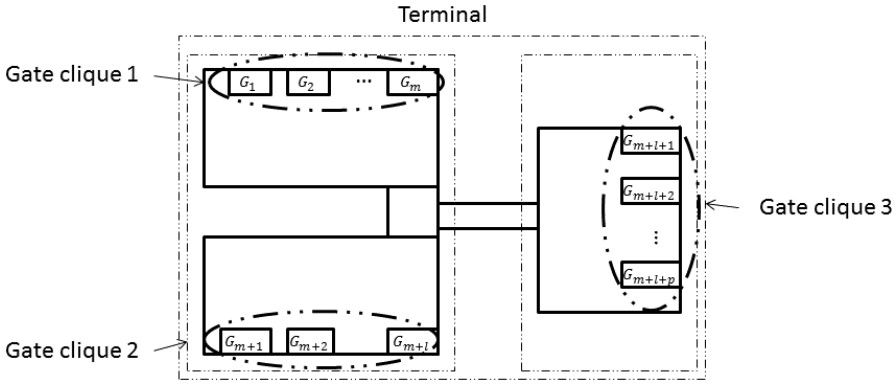

Figure.4 Illustration of gate cliques

In order to build the multi-commodity network flow model, we further make following three assumptions.

Assumption 1: The transfer time between any two gates within the same gate clique is assumed to be same.

Assumption 2: For two gate cliques ($g'_1, g'_2 \dots g'_m$) and ($g''_1, g''_2 \dots g''_n$), the transfer time between any two gates g'_i and g''_j, $1 \le i \le m$, $1 \le j \le n$ is assumed to be same.

Typically, a terminal consists of one or multiple concourses. Gates within one concourse are compact and the transfer time between gates within the same concourse is typically very short. For this situation, a concourse can be regarded as one gate clique. Any two different concourses are connected by a walkway. The transfer time by using the walkway is much longer than the transfer time inside a concourse; it is reasonable to make Assumption 2. In some situations, a concourse might be large and we can divide it into several gate cliques.

Assumption 3: The transfer time between any two gate cliques is known in advance.

Based on these three assumptions, we extend the MCGR model to consider passengers. The extended network flow model is named as the MCGRP model. A *passenger connection* is defined as a pair of arrival and departure flights and there are passengers transferring from the arrival flight to the departure flight. All connecting passengers with the same arrival and departure flights are associated with the same passenger connection.

In the MCGRP model, each passenger connection is considered as a commodity and each identical passenger connection corresponds to one network. An example of the network for a passenger connection is illustrated in Figure 5. There are four essential elements in this network.

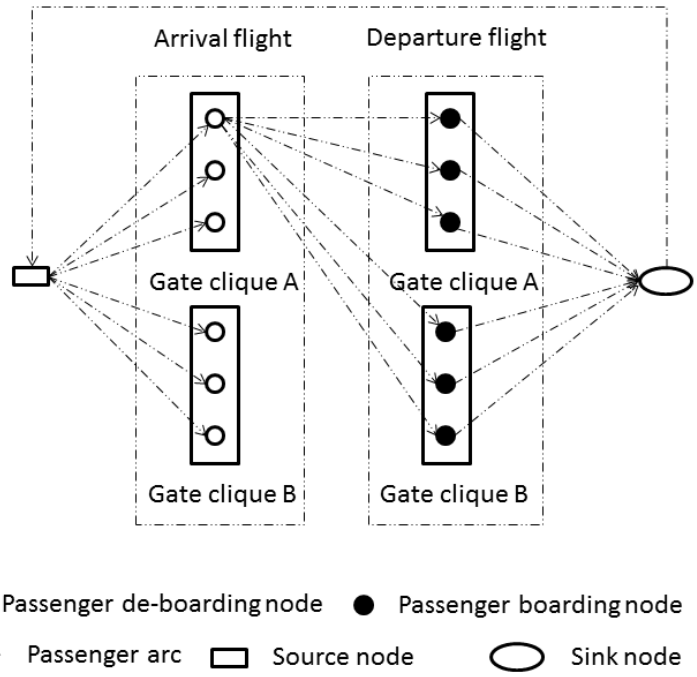

Figure.5 Illustration of the network for one passenger connection

(1) *Passenger de-boarding nodes* are indicating passengers' de-boarding status. A passenger de-boarding node is encoded by a gate clique where the passenger de-boards and the earliest de-boarding time. An example is illustrated in Figure 5. All of the passenger de-boarding nodes are located on the left side. Each rectangle indicates one gate clique. Within each gate clique, there are several nodes indicating different options of passengers' de-boarding time. If the arrival flight of a passenger connection can be assigned to a gate clique, one rectangle corresponding to this gate clique is created on the left side. Within each rectangle, each delay option of the arrival flight creates one de-boarding node. It is noted that the earliest de-boarding time is different from the gate arrival time because that after an aircraft arrives at a gate, it still takes several minutes for the passengers to de-board.

(2) *Passenger boarding nodes* are indicating passengers' boarding status. Similarly, a passenger boarding node contains a gate clique and the latest boarding time. In Figure 5, all of the passenger boarding nodes are located on the right side.

(3) *Passenger arcs* are represented as dashed lines in Figure 5. There are four types of passenger arcs: (a) passenger arcs connecting the source node and passenger de-boarding nodes; (b) passenger arcs connecting passenger de-boarding and boarding nodes; (c) passenger arcs connecting passenger boarding nodes and the sink node; (d) a wrap-around arc connecting the sink and source nodes.

The type (a) passenger arcs are created between the source node and any passenger de-boarding node and selection of a type (a) passenger arc indicates the gate clique and delay option assigned to the arrival flight. Similarly, the type (c) passenger arcs are created between any passenger boarding node and the sink node and selection of a type (c) passenger arc indicates the gate clique and delay option assigned to the departure flight. For any passenger de-boarding node n_d and any passenger boarding node n_b, one type (b) passenger arc is created between them only if there is enough time for passengers to travel from the gate clique of n_d to the gate clique of n_b. A feasible flow in a passenger connection's network contains exactly four arcs. The first arc is a

type (a) passenger arc followed by a type (b) passenger arc, the third arc is a type (c) passenger arc and, finally, the wrap-around arc connects the sink and source nodes. The flow upper bound is one. If no feasible flow is selected in the passenger connection's network, it indicates that the passenger connection is missed.

(4) A *source node* and a *sink node* are two virtual nodes denoting the start and end status of the passenger connection.

In the model, we have to link the gate and passenger connection networks. A feasible flow in one passenger connection's network indicates the corresponding arrival and departure flights' gate assignment decision and delay option. It should be consistent with the decisions in the gates' network.

Let FD_f represent the set of all possible delay options for flight f indexed as fd. Let K represent the set of all gate cliques indexed as k.

Each aircraft delay arc indicates two kinds of information for its arrival and departure flights: (a) the gate clique of the arrival and departure flights and, (b) the delay options of the arrival and departure flights. For any (f, fd, k) pair, let $E_{(f,fd,k)}$ represent the set of all aircraft delay arcs with $e \in E_{(f,fd,k)}$ satisfying three conditions: (a) f is either the arrival or departure flight of e; (b) the delay option of f is fd; (c) the assigned gate of e is located within gate clique k.

Each type (a) or type (c) passenger arc indicates the corresponding arrival or departure flight's gate assignment decision and delay option. Let $E^p_{(\tau,f,fd,k)}$ represent the set of all passenger arcs with $e \in E^p_{(\tau,f,fd,k)}$ satisfying four conditions: (1) e is either type (a) or type (c) passenger arc in passenger connection τ's network; (2) flight f is either the arrival flight or departure flight of this passenger connection; (3) the delay option of f is fd; (4) the assigned gate clique of f in e is k.

For every (f, fd, k), $\omega \in E^p_{(\tau,f,fd,k)}$ can be selected only if at least one $e \in E_{(f,fd,k)}$ has been selected.

We next present the multi-commodity network flow model for the gate re-assignment problem with connecting passengers. The additional notation used in this model is described in Appendix C.

The Model **MCGRP** is based on the **MCGR** model. A network is built for each passenger connection and a passenger connection consists of a group of passengers with identical arrival and departure flights. A feasible flow is explored in each passenger connection's network. Additional constraints are incorporated to build the relationship between the gates' network and passenger connections' network.

The model recognized the following additional variables.

z_ω: binary variable indicating whether passenger arc ω is selected

y_τ: binary variable indicating whether passenger connection τ is missed

The MCGRP model:

$$\text{Min} \ \sum_{e \in E} c_e \cdot x_e + \sum_{\tau \in \text{Conn}} cancel_\tau \cdot y_\tau + \sum_{\omega \in E^p} c^p_\omega \cdot z_\omega \quad (5)$$

Subject to:

Constraints (2)-(4)

$$\sum_{\omega \in E^{p,n}_{input}} z_\omega - \sum_{\omega \in E^{p,n}_{output}} z_\omega = 0 \qquad n \in N^p \quad (6)$$

$$z_{e^{cycle}_\tau} + y_\tau = 1 \qquad \tau \in Conn \quad (7)$$

$$\sum_{e \in E_{(f,fd,k)}} x_e - \sum_{\omega \in E^p_{(\tau,f,fd,k)}} z_\omega \quad \geq \quad 0 \qquad \tau \in Conn,\ f \in F,\ fd \in FD_f\ \ k \in K \qquad (8)$$

$x_e \in \{0,1\}$ $\epsilon_\tau \in \{0,1\}$

Objective (5) is minimizing the summation of aircraft delay arcs cost, missed passenger connection cost and passenger transfer cost. Constraints (6) guarantee the flow balance at each node in each passenger connection's network while constraints (7) restrict that each passenger connection is either made or missed. Constraints (8) build the relationship between gates' network and passenger connections' network.

Finally, we note that model MCGRP can be easily extended to consider crew connections since the basic network structure of a crew connection is the same as a passenger connection.

3 Algorithms

The MCGR model can be solved by a commercial solver, e.g. CPLEX, in seconds. However, it is not tractable to solve the MCGRP model directly by a solver. In this section, we propose two efficient algorithms to solve the MCGRP model. The solver used in our experiments is CPLEX.

3.1 Guided diving heuristic algorithm based on general upper bound branching (GDGUB)

Based on our preliminary computational study, the integrality gap of the LP relaxation is very small. In addition, the number of cuts added by CPLEX at the root node of the branch-and-bound tree is very small. It is thus expected that a diving heuristic algorithm can solve the MCGRP model efficiently.

Traditionally, a diving heuristic algorithm is implemented in a way that decision variables are iteratively fixed [43-46]. In this section, we propose a diving heuristic algorithm in which domains of the decision variables are iteratively decreased based on gate cliques. The guided diving heuristic is executed as follows.

Step 1: All the aircraft are sorted by the initial gate arrival time in ascending order.

Step 2: The linear relaxation of the MCGRP model is solved. The optimal solution indicates the fractional value for each aircraft delay arc. An aircraft can create many aircraft delay arcs in all gates' network. Let Π_a represent the set of all aircraft delay arcs created by aircraft a with $e \in \Pi_a$ satisfying the condition that e has a positive fractional value in the LP relaxation solution.

Step 3: Each aircraft is fixed to a gate clique. As we described before, each aircraft delay arc uniquely corresponds to one gate clique. For an aircraft a, if all aircraft delay arcs in Π_a belong to one gate clique, then aircraft a is fixed to this gate clique; otherwise, aircraft a is fixed to a gate clique with the highest cumulative fractional value. The cumulative fractional value of each gate clique is calculated in the following way. Initially, the cumulative fractional value of each gate clique is set to 0. For every $e \in \Pi_a$, if e belongs to a gate clique k, the cumulative fractional value of k is increased by the value of e in the optimal fractional solution. An example is illustrated in Figure 6. The aircraft has four aircraft delay arcs with positive fractional value. The cumulative fractional value of gate clique 1 is 0.8 while the cumulative fractional value of gate

clique 2 is 0.2. The aircraft 1 is therefore fixed to gate clique 1. Fixing an aircraft to a gate clique indicates that the aircraft can only be assigned to any gate within this gate clique in subsequent steps.

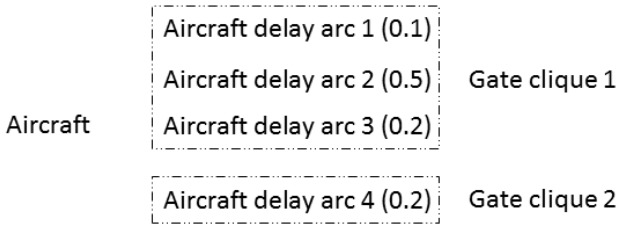

Figure.6 One fractional solution for an Aircraft

Step 4: To guarantee good solution quality, in each iteration, a limited number of aircraft are fixed to gate cliques. Let n_{fix} to be the number of fixed aircraft in this iteration. In the beginning, n_{fix} is set to 0 and the aircraft are fixed in the pre-defined order specified in Step 1. For an aircraft a, if all aircraft delay arcs in Π_a do not belong to one gate clique, n_{fix} is increased by 1. If the n_{fix} is larger than a certain limit, this iteration ends.

Step 5: If each aircraft is fixed to one gate clique, a restricted integer programming model is formulated in which aircraft delay arcs violating fixation decisions made in previous steps are removed. CPLEX is called to solve the restricted integer programming model; else, we re-start with step 1.

The pseudo-code of the algorithm GDGUB is described in Appendix D.

3.2 Variable rolling horizon algorithm (VRH)

If the length of the re-assignment time window is short, such as 3 hours, the GDGUB algorithm can efficiently solve the MCGRP model. However, in some situations, gate controllers might set a longer length of the time window, such as 9 hours. The GDGUB algorithm cannot provide a solution within several minutes for such cases. In this section, we propose a rolling horizon algorithm in which the gate re-assignment problem is divided into several sub-problems. The model size for each sub problem is small enough so that the GDGUB algorithm can efficiently solve it.

A rolling horizon algorithm is typically designed by dividing the timeline into several intervals. The idea of rolling horizon has already been applied to solve various large-scale problems [47-50]. One sub problem is formulated for each interval. The length of each interval has a significant impact on the solution quality and computational performance. If the length of each interval is large, a good solution quality can be achieved since each sub problem can have a good global view. However, the computation time might be long since the model size of each sub problem is large. A good rolling horizon can better trade-off between the computation time and the solution quality.

In this section, we propose a rolling horizon algorithm (VRH) with variable length of intervals. The algorithm is proposed based on following two observations:

(1) Passenger connections are not uniformly distributed over the timeline. For example, in a peaked flight schedule, most of the passenger connections are concentrated in a certain time interval. In a de-peaked flight schedule, the passenger connections are more evenly distributed across the timeline.

(2) In the MCGRP model, the number of decision variables and constraints created in the passenger connections' network is much larger than the number of decision variables and constraints created in the gates' network. Therefore, the number of passenger connections has a significant impact on the computational performance.

In our algorithm, the length of each interval is decided so that the passenger connections are uniformly distributed among all of the intervals. One benefit of such a design is that a longer length can be set for an interval with relatively fewer passenger connections within it. Compared to the design of uniformly dividing the timeline, our design can achieve a better trade-off between the solution quality and computation time. To further improve the solution quality, two consecutive intervals are overlapped. It indicates that only part of a solution is fixed after one sub problem is solved. In our study, the starting time of the next horizon is one hour later than the starting time of the current horizon, although the length of each horizon is three hours.

Figure 7 illustrates how to divide the timeline. The timeline is decomposed to four intervals. The number of passenger connections in each interval is roughly the same. Any two consecutive iterations are overlapped. The sub problem in each interval is solved by the GDGUB algorithm.

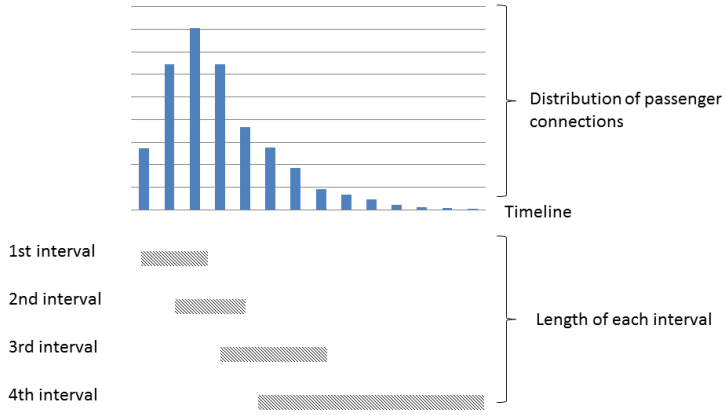

Figure.7 Illustration of dividing the timeline

The pseudo-code of the VRH algorithm is described in Appendix D.

4. Experiments

To test the performance of the proposed models and algorithms, we perform several numerical experiments based on real world data of a major airline at one of the largest U.S. airports. The algorithms are tested on an Intel core X5660 CPU with 32 gigabytes memory. The MIP solver in our study is ILOG CPLEX 12.5.

4.1 Description of data

The airport has 4 terminals and each terminal has multiple concourses. The airline occupies two terminals. For ease of exposition, we call them terminal 1 and terminal 2 in the following context. The map of terminal 1 and terminal 2 is illustrated in Figure 8. In concourse B and C, there are 22 gates and 28 gates assigned to the airline, respectively. In concourse E and F, there are 5 and 27 gates assigned to the airline, respectively. The airline provided operational data for three days in

January of 2015. There are 1101, 1054 and 1064 flights departing and arriving at the airport in these three days, respectively. We were also provided with planning data, and all engineering and business requirements.

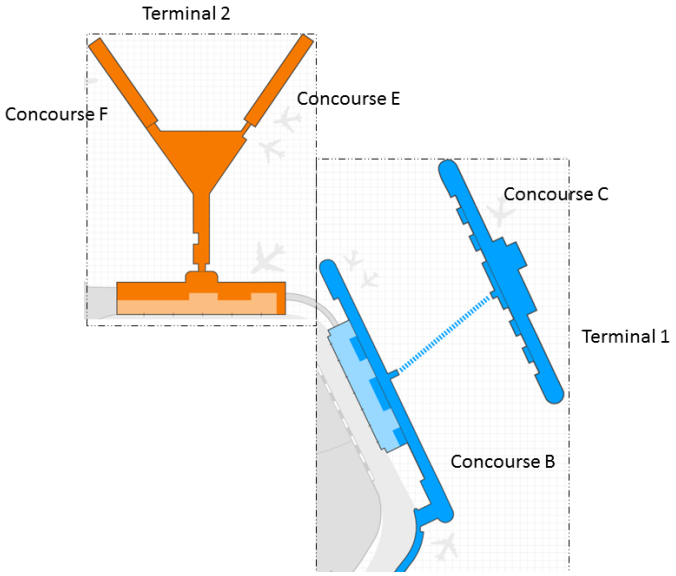

Figure.8 Illustration of the airport map

In these three days, severe weather caused different disruption scenarios. The airline provided several snapshots of the flight schedule and gate assignment plan at different time points over these three days. Each snapshot indicates the current flight schedule and gate assignment plan. The detailed information of all snapshots is given in Table 8 in Appendix A.

After consulting with gate controllers, three different options were recommended for the length of the re-assignment time window, 3 hours, 6 hours and 9 hours. Although we test these three time windows for all instances in our experiments, in practice, the 3 hour-time window is typically adopted for small-scale disruptions, the 6 hour-time window for medium-scale disruptions and the 9 hour-time window for large-scale events. It is noted that these three options are not applicable to each instance. For example, the 7^{th} instance in scenario 1 starts at 21:20, only the 3h time window is meaningful because it already reaches the end of the day and thus very low activity at the airport. All of the applicable time window options are described in Table 9 in Appendix A.

To evaluate the solution quality, several key performance indicators are proposed in Table 1.

Table 1 Key performance indicators

KPI	Description
TFD	Total flight delays in minutes
NDF	The number of delayed flights
NGR	The number of gate re-assignments
NKM	The number of crew missing connections
NPM	The number of passenger missing connections
TRC	The total re-assignment cost (objective value of the model)
LOPT	The optimal value of the linear relaxation of the MCGRP model
OPTGAP	The gap between TRC and LOPT
CPU	The computation time calculated in seconds

Detailed settings of all parameters are described in Table 2. If a flight is held on the ground for

one hour, increments of the crew salary and flight attendant salary are approximated as 250 USD and 150 USD, respectively. The passenger goodwill loss for all passengers associated with the same flight is approximated as 800 USD/Hour. The cost of a flight delay consists of the salary increment and passenger goodwill loss. Therefore, the cost of each minute of a flight delay is approximated as 20 USD. If one crew misses its connection, a reserve crew is utilized. The crew salary is increased by 750 USD for a three hours workload increment. In addition, 250 USD operating cost is incurred which yields the total of USD 1,000. If a passenger misses a connection, the compensation cost is approximated as 100 USD. The goodwill loss is approximated as 100 USD. In total, the cost of a missed passenger connection is approximated as 200 USD. If a flight is assigned to the apron gate, additional operating cost (such as a shuttle bus) is incurred to transfer the passengers between the parking area and a terminal. In addition, passengers' inconvenience and goodwill loss are incurred. Therefore, a large cost of USD 2,000 is assigned to prevent assigning a flight to the apron gate. If there are gate violations occurring after the re-assignment window, high penalty cost is assigned because the future gate assignment plan is disrupted. In the experiment, we set the cost of a gate violation after re-assignment to 1,000 USD.

Table 2 Parameters setting

Parameters	Values
The cost of each minute of a flight delay	20 USD
The cost of a missed crew connection	1,000 USD
The cost of a missed passenger connection	200 USD
The cost of reassigning a flight	150 USD
The cost of assigning a flight to the apron gate	2,000 USD
The cost of a gate violation after the re-assignment window	1,000 USD

The cost of a gate re-assignment is a step function. If a flight close to the beginning of the time window is re-assigned, the larger cost is set since more passenger inconvenience is caused. The cost consists of two parts: the basic operating cost and the inconvenience cost to all passengers traveling on the corresponding flight. The basic operating cost is estimated at 40 USD. The passenger inconvenience cost is only applicable for departure flights due to a gate change. Table 3 lists five cost values. The lead time indicates the gap between the start of the re-assignment time window and the flight's departure time.

Table 3 Passenger inconvenience cost for departure flights

Lead time	1h	2h	3h	4h	5h	6h
The passenger inconvenience cost (USD)	260	160	120	80	40	0

4.2 Computational results

The data provided by the airline contains crew information, however due to sensitivity, no information on passenger transfer traffic was provided. For this reason, we simulate passenger connections. Two different scenarios of passenger connections are simulated. The first scenario is based on a peaked flight schedule while the second is based on a de-peaked flight schedule. In a peaked schedule, flight arrivals typically occur first, followed by a brief period of aircraft inactivity and then flight departures occur. The period of aircraft inactivity between arrival and departure banks allows transfer passenger to connect. The peaked scenario is illustrated in Figure 13 in

Appendix B. In a de-peaked flight schedule, the set of incoming and outgoing flights are interspersed, which is illustrated in Figure 14 in Appendix B. The connection time of all of the passenger connections is between 20 and 80 minutes (assuming longer connection while occurring, they are not time sensitive). Among all instances, the maximum number of involved aircraft is 688, the maximum number of passenger connections is 1,558 and the maximum number of crew connections is 163.

Based on the architecture of terminals 1 and 2, three gate cliques are defined. The concourses B and C are defined as two gate cliques. The concourses E and F are much close together and they are combined into one gate clique.

All instances are solved using the VRH algorithm. The parameters of the VRH are $l_{overlap} = 60\ Minutes$ and $N_t = 600$. The solution by our model and the operations solution are compared to evaluate our solution. The latter reflects the gate reassignments produced by gate controllers. They are very experienced and use sophisticated visualization techniques combined with trial-and-error to design reassignments.

Figures 9 shows comparisons between our solution and the operations solution. In Figure 9, there are three box-plots with each one representing the gap for one key performance indicator. The gap is $\frac{operations\ solution - our\ solution}{operations\ solution}$. A positive gap value indicates our method is better than the operations method. The box plot is based on the different instances for each scenario. All instances are abbreviated as t=y or dt=y, where d represents a de-peaked flight schedule and t=y represents the length of the time window is y. All computational results are described in Tables 10-15 in Appendix E.

In general, our solution has the absolute advantage compared to the operations solution with respect to the three most important key performance indicators. Here are the key findings.

(1) For TFD, the average gap between our solution and the operations solution increases as the length of the time window increases. This is expected since the problem scale is not very large when the length of the time window is 3 hours, and thus a good solution can be explored by manually adjusting the initial gate assignment plan. When the length of the time window increases, our method can still provide a high quality solution due to strong exploration capability embedded in our method. However, the solution quality of the operations method deteriorates since it is hard to find a good solution purely by experienced gate controllers.

(2) The average gap of NPM increases as the length of the time window increases. As the length of the time window increases, it is hard for gate controllers to consider all of the objectives. Assigning each aircraft to a gate has a higher priority and passenger considerations are neglected to some extent. Therefore, more passenger connections might be missed as the length of time window increases.

(3) The average gap of NGR decreases as the length of the time window increases. One possible reason is as follows. As the length of time window increases, our method explores solutions with more reassigning operations to decrease total flight delays and the number of missed passenger connections. While, for the operations method, the value of NGR has roughly linear relationship with the length of the time window. In most instances, the number of NGR of our solution is smaller than the value of the operations solution. Therefore, the gap of NGR decreases as the length of time window increases.

(4) The trend of TRC is similar to the trend of NGR as the total re-assignment cost depends on the

number of re-assignment operations. More re-assignment operations indicate a higher total re-assignment cost.

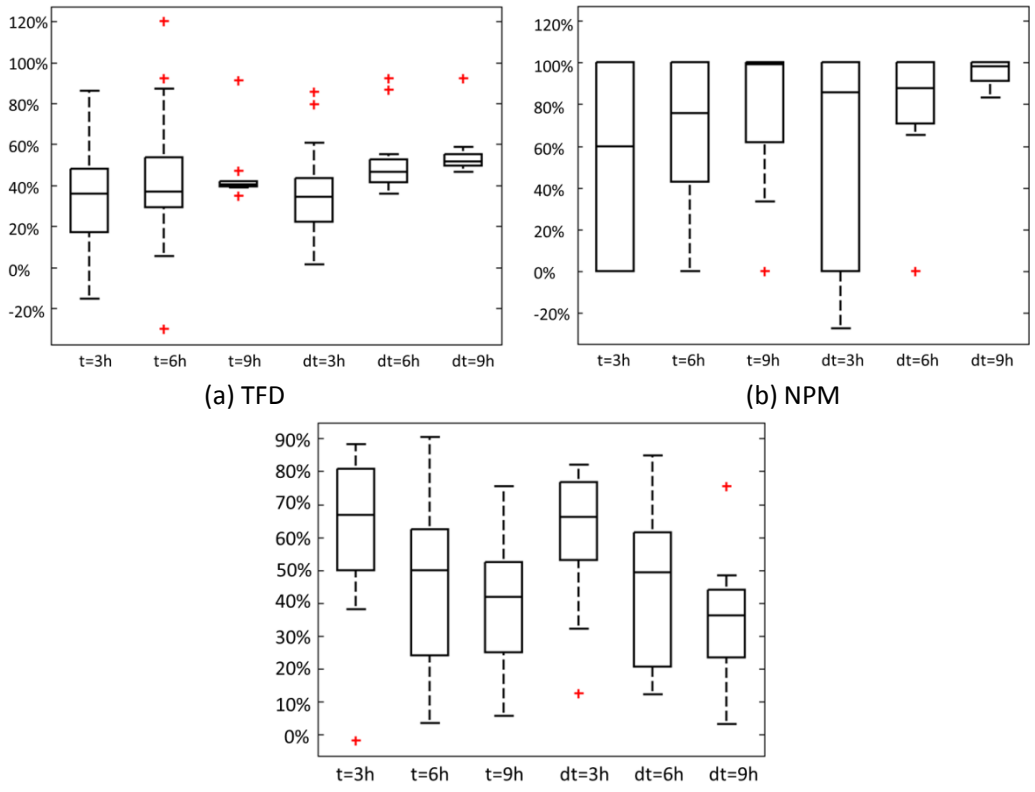

Figure.9 Comparison between our method and the operations method

Computational performances of our method are summarized in Table 4. For the scenarios in which passenger connections are simulated in the peaked flight schedule, the performance of our algorithm is very good when the length of the re-assignment time window is 3 hours. Among 21 instances, our algorithm can get an optimal solution in 15 instances. The largest gap is 6.34% and the average gap is 0.7%. The longest computing time is 22.16 seconds. When the length of the time window is 6 hours, the largest gap is 8.57% and the average gap is 3.91%. The longest computing time is 88.83 seconds. When the length of the time window is 9 hours, the largest gap is 12.02% and the average gap is 7.00%. The longest computing time is 232.57 seconds. The results are consistent with our intuition. As the length of time window increases, the algorithm's performance deteriorates since more sub-problems are solved in the VRH algorithm. For the de-peaked flight schedule, the trend is similar.

Note that Maharjan et al. [7] proposed a quadratic 0-1 model to solve the gate re-assignment problem with connecting passengers. It is well known that quadratic 0-1 models are very difficult to solve. For a large-scale problem, it is nearly impossible to provide a solution within 1 hour. Our methodology has an obvious advantage over quadratic 0-1 models due to all instances being solved within minutes. For this reason we do not present comparative results between our method and the quadratic 0-1 model.

Table 4 Summary of computational results

		Average CPU seconds	Maximum CPU seconds	Average Gap	Maximum Gap
peaked	scenario 1	9.15	22.16	1.06%	6.34%

t=3h	scenario 2	8.74	12.76	0.56%	3.17%
	scenario 3	5.42	10.50	0.57%	4.14%
peaked	scenario 1	54.20	78.10	3.87%	6.97%
t=6h	scenario 2	60.03	88.83	4.84%	8.57%
	scenario 3	32.67	77.21	3.00%	7.31%
peaked	scenario 1	175.67	232.57	6.57%	7.27%
t=9h	scenario 2	125.30	216.69	7.25%	12.02%
	scenario 3	81.04	139.20	6.89%	9.82%
de-peaked	scenario 1	9.37	17.10	1.58%	5.34%
t=3h	scenario 2	9.30	11.74	0.14%	0.50%
	scenario 3	5.71	9.85	0.40%	1.95%
de-peaked	scenario 1	57.28	67.65	3.05%	5.03%
t=6h	scenario 2	65.06	92.22	4.94%	7.65%
	scenario 3	27.61	50.55	2.10%	7.59%
de-peaked	scenario 1	101.03	103.54	8.48%	8.81%
t=9h	scenario 2	217.05	287.12	6.64%	12.61%
	scenario 3	127.73	266.32	7.20%	11.24%

4.3 Setting parameters' value

In Section 4.2, values of the parameters, in particular cost values, are set based on approximations. Likewise, in practice, the gate controllers might not be very sure about how to set parameter values. In this section, we propose a method based on the *Data Envelop Analysis* (**DEA**) to help the gate controllers select several potential parameter value settings. There are two inputs of this method: (1) several typical instances; and (2) several possible values for each parameter.

Let P represent the set of all the parameters which is indexed as p. Let I represent the set of all key performance indicators which is indexed as i. The method is conducted as follows.

(1) Let Θ represent the set of all possible reasonable parameters' value settings. For every $\theta \in \Theta$, we solve the MCGRP model and the key performance indicators of the solution is denoted as s^{θ}.

(2) For every $\theta \in \Theta$, a linear programming model is built to calculate the efficient coefficient of θ. Model **DEA** is the standard DEA model.

Model DEA:

$\text{Min } t_{\theta_0}$ (9)

s.t.

$\sum_{\theta \in \Theta/\theta_0} \lambda_\theta \cdot s^\theta - t_{\theta_0} \cdot s^{\theta_0} \geq 0$ (10)

$\sum_{\theta \in \Theta/\theta_0} \lambda_\theta \cdot \theta - \theta_0 \leq 0$ (11)

$\lambda_\theta \geq 0, \quad t_{\theta_0}$ unconstrained

(3) All of the value settings are sorted by the efficient coefficient value t_{θ_0} in ascending order. Several value settings at the front of the list are recommended to the gate controllers. Once a disruption scenario occurs in the future, the gate controllers can try these value settings and choose a solution based on the situation at hand. Table 5 illustrates the results of the DEA model

for one scenario. The first 10 best parameters value settings can be selected by choosing 10 parameters value settings with the smallest efficient coefficient value. Based on these parameter value settings, several promising solutions can be produced.

Table 5 Results of DEA model

Efficient coefficient value	Unit reassignment cost (USD)	Flight delay cost/Minute (USD)	Gate violation cost after window (USD)	Unit crew miss connection cost (USD)	Unit passenger miss connection cost (USD)
0.6284	150	20	2000	1000	200
0.6315	150	20	3000	1000	300
0.6322	100	30	2000	1200	200
0.6322	150	30	2000	1200	300
0.6409	150	20	2000	1200	200
0.6468	100	30	3000	1200	200
0.6469	150	20	3000	1200	200
0.6523	150	20	2000	1200	300
0.6529	150	20	3000	1200	300
0.6534	100	20	3000	1200	300

4.4 Sensitivity analysis

To explore the impact of different parameter values on the solution, sensitivity analysis studies are conducted. We conduct three experiments. In the first experiment, three different values of the cost of each minute of a flight delay are tested. All other parameter values are fixed. The trend of solutions is illustrated in Figure 10. It is obvious that the total flight delays decrease as the unit flight delay cost increases. However, more gate re-assignment operations are required to accommodate all the aircraft within the recovery time window. In addition, more transfer passenger connections are missed because of insufficient connection time.

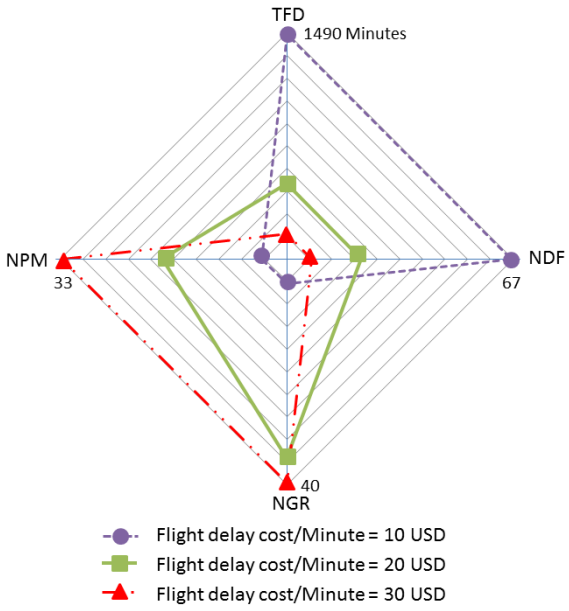

Figure.10 Illustration of sensitivity analysis 1

In the second sensitivity analysis, we try to explore the impact of the cost of a unit gate

re-assignment operation on solutions. The cost of the unit gate re-assignment operation is defined as a step function. Notation 0.5X indicates the cost of unit gate re-assignment operation is set as 0.5 times the original cost. The trend of solutions is illustrated in Figure 11. The number of gate re-assignment operations decreases as the cost of a unit gate re-assignment operation increases. In the meanwhile, the total flight delays increase to accommodate more aircraft. We also observe that the number of passenger connections missed slightly decreases.

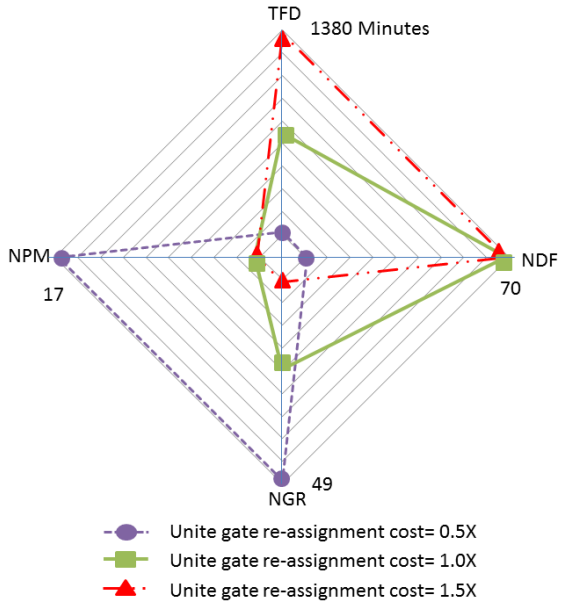

Figure.11 Illustration of sensitivity analysis 2

In the third sensitivity analysis, we try to explore the impact of the unit cost of missed passenger connections on solutions. As illustrated in Figure 12, the trend is similar to the previous two experiments. As the cost increases, the total number of missed passenger connection decreases while the values of other indicators increase on average.

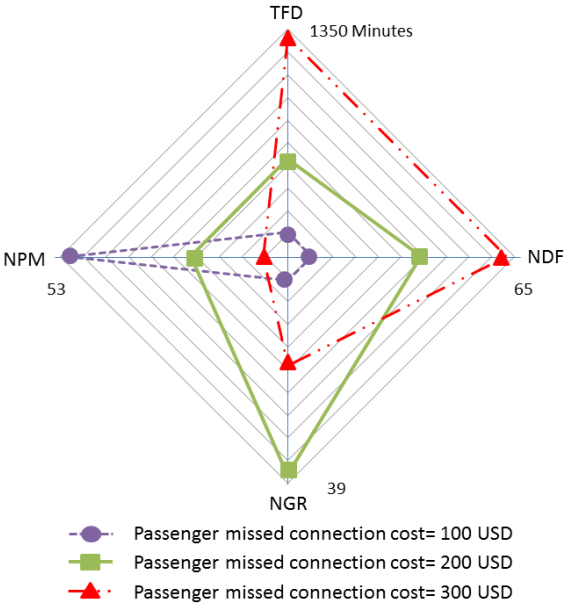

Figure.12 Illustration of sensitivity analysis 3

4.5 Comparison between the MCGR model and recent studies

In literature, the gate re-assignment problem is typically modeled as an assignment model with side constraints [4-6] or a multi-commodity network model [3]. In this section, experimental results with respect to model performance are illustrated. We set the time limit as 5 minutes based on the gate re-assignment requirement in practice. No passengers are included since they require quadratic constraints. All models are directly solved by CPLEX to optimality. Experimental results reveal that the optimal solution cannot be achieved within five minutes by solving the **YModel** from Yan et al. [3] except for one instance. However, an optimal solution can be achieved within one minute by solving our model. Table 6 illustrates comparisons with respect to the number of columns and rows. It clearly reveals that the model size of the YModel is much larger than our MCGR model. For a model of a such large scale, it is virtually impossible to get the optimal solution within minutes.

In this section, we also compare the MCGR model with the assignment model [4-6]. The assignment model proposed in [4-6] is represented as **TModel**. Experimental results reveal that the optimal solution for all instances can be achived within 5 minutes by solving either our model or Tmodel. Therefore, we will compare them from a computational perspective. Four indices are used to evaluate these two models' computational efficiency: no. of columns, no. of rows, no. of non-zeros in decision variables and CPU seconds. For each index i, MCGR_i indicates the value of index i for the solution of the model MCGR. Similarly, we define $\mathrm{TModel}_{\ i}$ for the TModel model. Value $\frac{(\mathrm{TModel}_{\ i}-\mathrm{MCGR}_i)}{\mathrm{TModel}_{\ i}}$ defines the gap. If it is a positive value, it indicates the model MCGR has a smaller value of index i.

The comparison between the two models is illustrated in Table 7. The average difference between two models in each scenario is illustrated in each row. The results reveal that the number of decision variables and constraints in the MCGR model are slightly greater than the number in TModel. However, the number of non-zeros in the MCGR model is much smaller than the number in TModel. The computational efficiency of a model is highly dependent on the number of non-zeros, therefore, the MCGR model achieves on average 71.39% CPU seconds improvement compared to the TModel model.

Table 6 Comparison between the MCGR and YModel models

		No. of columns		No. of rows		CPU seconds			
		MCGR	YModel	MCGR	YModel	MCGR		YModel	
		Average	Average	Average	Average	Max	Average	Max	Average
t=3h	Scenario 1	86,734	6,493,902	16,892	127,205	3.82	2.22	>300	>300
	Scenario 2	89,245	7,271,012	18,997	144,220	4.53	2.88	>300	>300
	Scenario 3	90,039	6,706,343	18,965	124,996	3.89	2.56	>300	>300
t=6h	Scenario 1	251,157	100,054,813	36,716	256,814	17.00	12.43	>300	>300
	Scenario 2	247,893	173,799,700	39,243	427,186	16.46	12.20	>300	>300
	Scenario 3	234,622	96,246,346	37,691	289,056	35.87	15.70	>300	>300
t=9h	Scenario 1	425,326	232,090,070	54,668	346,245	45.79	31.70	>300	>300
	Scenario 2	402,889	623,738,361	58,811	702,495	34.72	26.63	>300	>300
	Scenario 3	375,833	237,026,175	55,049	394,990	45.95	38.59	>300	>300

Table 7 Comparison between the MCGR and TModel models

		No. of Cols	No. of Rows	No. of Non-zeros	CPU improvement
t=3h	scenario 1	-23.65%	-1.02%	86.75%	74.95%
	scenario 2	-20.47%	-0.88%	86.10%	78.11%
	scenario 3	-21.45%	-0.90%	85.43%	78.71%
t=6h	scenario 1	-12.42%	-0.46%	84.00%	71.08%
	scenario 2	-13.27%	-0.43%	84.76%	75.88%
	scenario 3	-14.13%	-0.45%	83.99%	69.67%
t=9h	scenario 1	-10.29%	-0.31%	83.43%	64.89%
	scenario 2	-11.98%	-0.28%	83.32%	72.24%
	scenario 3	-12.13%	-0.30%	83.25%	56.97%

5 Conclusions

In this study, we first build a more efficient multi-commodity network flow model for the gate re-assignment problem. Based on this model, we further propose a novel multi-commodity network flow model in which passengers are considered. To the best of our knowledge, this is the first multi-commodity network flow model for the gate re-assignment problem considering connecting passengers. To efficiently solve the models, a guided diving heuristic algorithm and a rolling horizon algorithm are proposed. The models and algorithms are tested based on the operational data from a large U.S. airline at a big airport. We simulate two different scenarios of transfer passenger connections. Several instances with difference length of the re-assignment time window are tested. Experimental results reveal that our algorithm can achieve a very good performance when the length of the time window is 3 hours. When the length of the time window increases, the optimality gap deteriorates. However, the average optimal gap is still less than 8%. The solution quality is very good for all time windows with respect to the operations solutions.

Four possible extensions can make this research more interesting. First, in the current research, we are considering the gate re-assignment problem for the U.S. airline industry. The gate re-assignment problem in the European or Asian airline industry is quite different because gate operations are planned by airports. It indicates that the problem scope becomes much larger. More efficient algorithms are needed to efficiently solve the gate re-assignment problem in the European or Asian airline industry. Second, it is possible to extend our models to solve the gate assignment planning problem considering passenger's transfer distance. Traditionally, the gate assignment planning problem considering passenger's transfer distance is built as a quadratic 0-1 integer programming model. Our research provides a possible way to model this problem as a multi-commodity network flow model. Third, in the current research, we only consider the gate re-assignment problem. It might be beneficial to consider the airport ground traffic control and/or runway sequencing problems simultaneously with the gate re-assignment problem. Fourth, since the objective of the current model is a weighted sum of several components, it is possible to propose alternative multi-objective optimization methods to achieve a better

trade-off between the different objectives.

References:

1 Tang, C.H., and Wang, W.C.: 'Airport gate assignments for airline-specific gates', Journal of Air Transport Management, 2013, 30, pp. 10-16

2 Gu, Y., and Chung, C.A.: 'Genetic algorithm approach to aircraft gate reassignment problem', Journal of Transportation Engineering, 1999, 125, (5), pp. 384-389

3 Yan, S., Chen, C.Y., and Tang, C.-H.: 'Airport gate reassignment following temporary airport closures', Transportmetrica, 2009, 5, (1), pp. 25-41

4 Yan, S., Tang, C.H., and Hou, Y.Z.: 'Airport gate reassignments considering deterministic and stochastic flight departure/arrival times', Journal of Advanced Transportation, 2011, 45, (4), pp. 304-320

5 Tang, C.H.: 'A gate reassignment model for the Taiwan Taoyuan Airport under temporary gate shortages and stochastic flight delays', Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on, 2011, 41, (4), pp. 637-650

6 Tang, C.H., Yan, S., and Hou, Y.Z.: 'A gate reassignment framework for real time flight delays', 4OR, 2010, 8, (3), pp. 299-318

7 Maharjan, B., and Matis, T.I.: 'An optimization model for gate reassignment in response to flight delays', Journal of Air Transport Management, 2011, 17, (4), pp. 256-261

8 Gosling, G.D.: 'Design of an expert system for aircraft gate assignment', Transportation Research Part A: General, 1990, 24, (1), pp. 59-69

9 Srihari, K., and Muthukrishnan, R.: 'An expert system methodology for aircraft-gate assignment', Computers & Industrial Engineering, 1991, 21, (1), pp. 101-105

10 Jo, G.S., Jung, J.J., and Yang, C.Y.: 'Expert system for scheduling in an airline gate allocation', Expert systems with applications, 1997, 13, (4), pp. 275-282

11 Su, Y., and Srihari, K.: 'A knowledge based aircraft-gate assignment advisor', Computers & industrial engineering, 1993, 25, (1), pp. 123-126

12 Cheng, Y.: 'A knowledge-based airport gate assignment system integrated with mathematical programming', Computers & Industrial Engineering, 1997, 32, (4), pp. 837-852

13 Babic, O., Teodorovic, D., and Tošic, V.: 'Aircraft stand assignment to minimize walking', Journal of Transportation Engineering, 1984, 110, (1), pp. 55-66

14 Mangoubi, R., and Mathaisel, D.F.: 'Optimizing gate assignments at airport terminals', Transportation Science, 1985, 19, (2), pp. 173-188

15 Vanderstraeten, G., and Bergeron, M.: 'Automatic assignment of aircraft to gates at a terminal', Computers & industrial engineering, 1988, 14, (1), pp. 15-25

16 Bihr, R.A.: 'A conceptual solution to the aircraft gate assignment problem using 0, 1 linear programming', Computers & Industrial Engineering, 1990, 19, (1), pp. 280-284

17 Wirasinghe, S., and Bandara, S.: 'Airport gate position estimation for minimum total costs—approximate closed form solution', Transportation Research Part B: Methodological, 1990, 24, (4), pp. 287-297

18 Yan, S., and Chang, C.M.: 'A network model for gate assignment', Journal of Advanced Transportation, 1998, 32, (2), pp. 176-189

19 Haghani, A., and Chen, M.C.: 'Optimizing gate assignments at airport terminals', Transportation

Research Part A: Policy and Practice, 1998, 32, (6), pp. 437-454
20 Bolat, A.: 'Procedures for providing robust gate assignments for arriving aircrafts', European Journal of Operational Research, 2000, 120, (1), pp. 63-80
21 Bolat, A.: 'Models and a genetic algorithm for static aircraft-gate assignment problem', Journal of the Operational Research Society, 2001, pp. 1107-1120
22 Bolat, A.: 'Assigning arriving flights at an airport to the available gates', Journal of the Operational Research Society, 1999, pp. 23-34
23 Yan, S., and Huo, C.M.: 'Optimization of multiple objective gate assignments', Transportation Research Part A: Policy and Practice, 2001, 35, (5), pp. 413-432
24 Yan, S., Shieh, C.Y., and Chen, M.: 'A simulation framework for evaluating airport gate assignments', Transportation Research Part A: Policy and Practice, 2002, 36, (10), pp. 885-898
25 Yan, S., and Tang, C.H.: 'A heuristic approach for airport gate assignments for stochastic flight delays', European Journal of Operational Research, 2007, 180, (2), pp. 547-567
26 Ding, H., Lim, A., Rodrigues, B., and Zhu, Y.: 'New heuristics for over-constrained flight to gate assignments', Journal of the Operational Research Society, 2004, 55, (7), pp. 760-768
27 Ding, H., Lim, A., Rodrigues, B., and Zhu, Y.: 'The over-constrained airport gate assignment problem', Computers & Operations Research, 2005, 32, (7), pp. 1867-1880
28 Şeker, M., and Noyan, N.: 'Stochastic optimization models for the airport gate assignment problem', Transportation Research Part E: Logistics and Transportation Review, 2012, 48, (2), pp. 438-459
29 Kim, S.H.: 'Airport Control Through Intelligent Gate Assignment', Georgia Institute of Technology, 2013
30 Narciso, M.E., and Piera, M.A.: 'Robust gate assignment procedures from an airport management perspective', Omega, 2015, 50, pp. 82-95
31 Castaing, J., Mukherjee, I., Cohn, A., Hurwitz, L., Nguyen, A., and Müller, J.J.: 'Reducing airport gate blockage in passenger aviation: Models and analysis', Computers & Operations Research, 2016, 65, pp. 189-199
32 Diepen, G., Van den Akker, J., Hoogeveen, J.a., and Smeltink, J.: 'Finding a robust assignment of flights to gates at Amsterdam Airport Schiphol', Journal of Scheduling, 2012, 15, (6), pp. 703-715
33 Dorndorf, U., Jaehn, F., and Pesch, E.: 'Modelling robust flight-gate scheduling as a clique partitioning problem', Transportation Science, 2008, 42, (3), pp. 292-301
34 Dorndorf, U., Jaehn, F., and Pesch, E.: 'Flight gate scheduling with respect to a reference schedule', Annals of Operations Research, 2012, 194, (1), pp. 177-187
35 Dorndorf, U., Jaehn, F., and Pesch, E.: 'Flight gate assignment and recovery strategies with stochastic arrival and departure times', OR Spectrum, 2016, pp. 1-29
36 Guépet, J., Acuna-Agost, R., Briant, O., and Gayon, J.P.: 'Exact and heuristic approaches to the airport stand allocation problem', European Journal of Operational Research, 2015, 246, (2), pp. 597-608
37 Kim, S.H., Feron, E., and Clarke, J.P.: 'Gate assignment to minimize passenger transit time and aircraft taxi time', Journal of Guidance, Control, and Dynamics, 2013, 36, (2), pp. 467-475
38 Prem Kumar, V., and Bierlaire, M.: 'Multi‐objective airport gate assignment problem in planning and operations', Journal of advanced transportation, 2014, 48, (7), pp. 902-926
39 Yu, C., Zhang, D., and Lau, H.: 'MIP-based heuristics for solving robust gate assignment problems', Computers & Industrial Engineering, 2016, 93, pp. 171-191

40 Ravizza, S., Atkin, J.A., and Burke, E.K.: 'A more realistic approach for airport ground movement optimisation with stand holding', Journal of Scheduling, 2014, 17, (5), pp. 507-520

41 Dorndorf, U., Drexl, A., Nikulin, Y., and Pesch, E.: 'Flight gate scheduling: State-of-the-art and recent developments', Omega, 2007, 35, (3), pp. 326-334

42 Bouras, A., Ghaleb, M.A., Suryahatmaja, U.S., and Salem, A.M.: 'The airport gate assignment problem: a survey', The Scientific World Journal, 2014, 2014

43 Grötschel, M., Borndörfer, R., and Löbel, A.: 'Duty scheduling in public transit': 'Mathematics—Key Technology for the Future' (Springer, 2003), pp. 653-674

44 Caprara, A., Lancia, G., and Ng, S.K.: 'Sorting permutations by reversals through branch-and-price', INFORMS journal on computing, 2001, 13, (3), pp. 224-244

45 Johnson, E.L., Nemhauser, G.L., and Savelsbergh, M.W.: 'Progress in linear programming-based algorithms for integer programming: An exposition', Informs journal on computing, 2000, 12, (1), pp. 2-23

46 Depuy, G.W., Savelsbergh, M.W., Ammons, J.C., and McGinnis, L.F.: 'An integer programming heuristic for component allocation in printed circuit card assembly systems', Journal of Heuristics, 2001, 7, (4), pp. 351-369

47 Bard, J.F., Huang, L., Jaillet, P., and Dror, M.: 'A decomposition approach to the inventory routing problem with satellite facilities', Transportation science, 1998, 32, (2), pp. 189-203

48 Jaillet, P., Bard, J.F., Huang, L., and Dror, M.: 'Delivery cost approximations for inventory routing problems in a rolling horizon framework', Transportation Science, 2002, 36, (3), pp. 292-300

49 Bostel, N., Dejax, P., Guez, P., and Tricoire, F.: 'Multiperiod planning and routing on a rolling horizon for field force optimization logistics': 'The vehicle routing problem: latest advances and new challenges' (Springer, 2008), pp. 503-525

50 Cordeau, J.F., Dell'Amico, M., Falavigna, S., and Iori, M.: 'A rolling horizon algorithm for auto-carrier transportation', Transportation Research Part B: Methodological, 2015, 76, pp. 68-80

Appendix A:

Table 8 Description of snapshots

Scenario	Time points of snapshots
Scenario 1	12:20, 13:50, 16:20, 17:50, 19:50, 20:20, 21:20
Scenario 2	10:50, 11:50, 12:50, 13:50, 14:50, 16:20
Scenario 3	11:20, 12:50, 13:20, 14:50, 16:20, 17:50, 19:50. 20:50

Table 9 Description of recovery time windows in each instance

		start time	Length of the re-assignment time window
scenario 1	instance 1	12:20	3h, 6h, 9h
	instance 2	13:50	3h, 6h, 9h
	instance 3	16:20	3h, 6h
	instance 4	17:50	3h, 6h
	instance 5	19:50	3h
	instance 6	20:20	3h
	instance 7	21:20	3h

scenario 2	instance 1	10:50	3h, 6h, 9h
	instance 2	11:50	3h, 6h, 9h
	instance 3	12:50	3h, 6h, 9h
	instance 4	13:50	3h, 6h, 9h
	instance 5	14:50	3h, 6h, 9h
	instance 6	16:20	3h, 6h
scenario 3	instance 1	11:20	3h, 6h, 9h
	instance 2	12:50	3h, 6h, 9h
	instance 3	13:20	3h, 6h, 9h
	instance 4	14:50	3h, 6h, 9h
	instance 5	16:20	3h, 6h
	instance 6	17:50	3h, 6h
	instance 7	19:50	3h
	instance 8	20:50	3h

Appendix B:

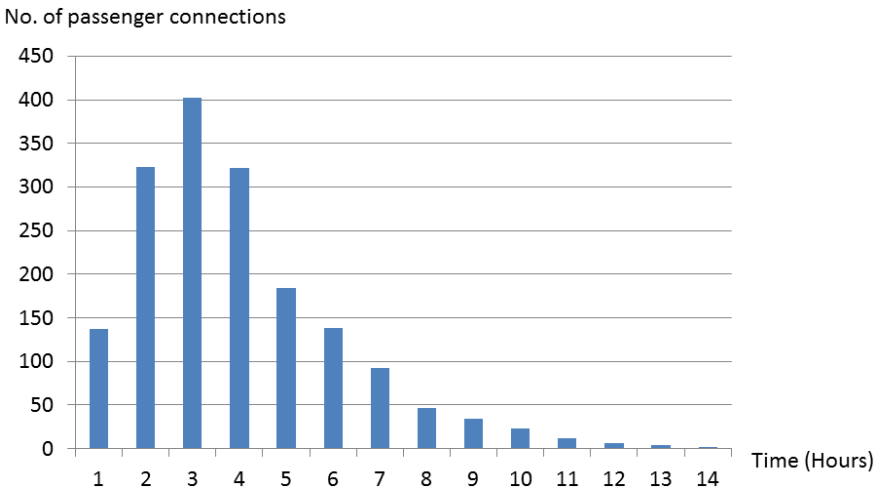

Figure.13 Distribution of passenger connections in the peaked flight schedule

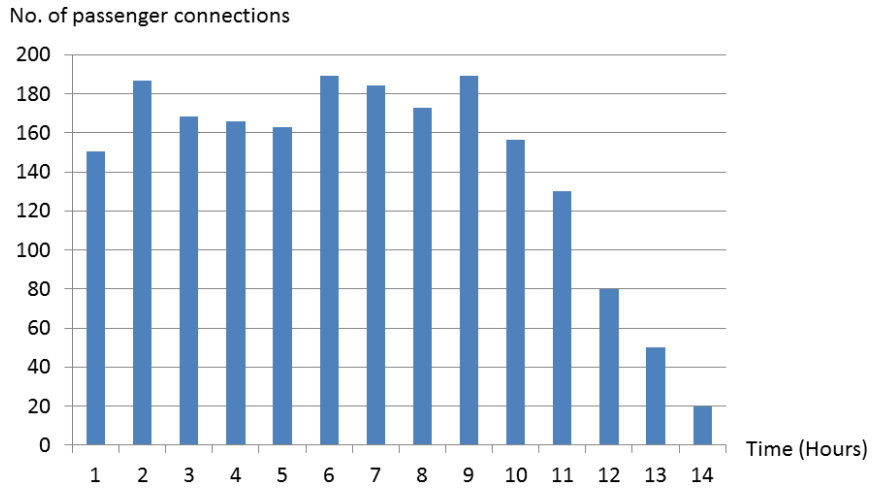

Figure.14 Distribution of passenger connections in the de-peaked flight schedule

Appendix C:

Notations for MCGR:

Π: set of all aircraft equipment types indexed as π

P: set of all gates indexed as ρ

A: set of all aircraft indexed as a

E: set of all aircraft delay arcs indexed as e

E_a: set of all aircraft delay arcs corresponding to aircraft a

N: set of all nodes (including activity, source and sink nodes) indexed as n

G: set of all ground arcs (including wrap-around arcs) indexed as g

E^n_{input}: set of all aircraft delay arcs with node n as tail

E^n_{output}: set of all aircraft delay arcs with node n as head

G^n_{input}: set of all ground arcs with node n as tail

G^n_{output}: set of all ground arcs with node n as head

S: set of all adjacency constraints indexed as s

Each adjacency constraint s is represented as $(\pi_1, \rho_1, \pi_2, \rho_2)$. It indicates that two aircraft of equipment types π_1 and π_2 cannot be assigned to gates ρ_1 and ρ_2 simultaneously.

T: set of all time points indexed as t

$E_{(\pi,\rho,t)}$: set of all aircraft delay arcs in gate ρ's network pertaining to aircraft of type π and including time t.

c_e: cost of aircraft delay arc e

Notations for MCGRP:

F: set of all flights indexed as f

$Conn$: set of all passenger connections indexed as τ

N^p: set of all nodes (including all the passenger de-boarding nodes and boarding nodes, the source and sink nodes) indexed as n

E^p: set of all passenger arcs indexed as ω

K: set of all gate cliques indexed as k

FD_f: set of all delay options for flight f indexed as fd

$E^{p,n}_{input}$: set of all input passenger arcs of node n

$E^{p,n}_{input}$: set of all output passenger arcs of node n

$E^p_{(\tau,f,fd,k)}$: set of all passenger delay arcs corresponding to pair (τ, f, fd, k)

$E_{(f,fd,k)}$: set of all aircraft delay arcs corresponding to pair (f, fd, k)

c^p_ω: cost of passenger arc ω

$cancel_\tau$: cancellation cost of passenger connection τ (there may be many passengers involved in connection τ; $cancel_\tau$ is the summation of the cancellation cost of all these passengers).

e^{cycle}_τ: wrap-around arc in the passenger connection τ's network

Appendix D:

The pseudo-code of the algorithm CGGUB is described as follows.

Algorithm GDGUB
Parameters:

n_{fix}: the number of fixed aircraft in each iteration

$Limit_{fix}$: the limit of fixed aircraft in each iteration

Initialization:

1: Set the fixed gate clique of each aircraft as null

2: Sort all aircraft by the gate arrival time in ascending order

Loop:

3: $n_{fix} = 0$

4: Solve the linear relaxation of the MCGRP model

5: For each aircraft a, calculate the cumulative fractional value of each gate clique.

6: If a gate clique k has the cumulative fractional value of 1

7: Fix aircraft a to gate clique k

8: else

9: Fix aircraft a to gate clique k with the largest cumulative fractional value

10: $n_{fix} = n_{fix} + 1$

11: end If

12: If $n_{fix} > Limit_{fix}$

13: break the for loop.

14: end If

15: End For

16: For each aircraft

17: Remove all aircraft delay arcs violating fixing decisions

18: End For

19: If each aircraft is assigned to one gate clique

20: Solve the restricted integer programming model by CPLEX

21: else

22: Go to step 3.

23: End If

The pseudo-code of the algorithm VRH is described as follows.

Algorithm VRH

Parameters:

$l_{overlap}$: the overlapped length between two consecutive intervals

N_t: the limit of passenger connections in each interval

t_s: the start time of the current interval

t_e: the end time of the current interval

TC: it indicates whether the termination condition is satisfied

Initialization:

1: Set t_s as the start time of the re-assignment time window

2: TC = false

Loop:

3: While TC is false

4: Set t_e as the latest time so the number of passenger connections in [t_s, t_e] is less then N_t

5: Solve the sub-problem in the interval [t_s, t_e] using the GDGUB algorithm

6: Update the status of aircraft and gate based on the solution of the sub problem

7: If t_e is the end time of the re-assignment time window
8: TC =true
9: Else
10: $t_s = t_s + l_{overlap}$
11: End If
12: End While

Appendix E:

Tables 10-15 describe detailed comparison results. Columns TFD, NDF, NGR, DAR, NKM, NPM and TRC show gaps between our and operations solutions for these key performance indicators. The gap is calculated as $\frac{operations\ solution - our\ solution}{operations\ solution}$. The column CPU shows the computation time in seconds of our solutions. The column OPTGAP shows the optimality gaps of our solutions. For all key performance indicators, positive values reflect that our solution is better. The negative values are in gray.

Table 10 Computational results in the peaked schedule (length of time window = 3 hours)

		TFD	NDF	NGR	DAR	NKM	NPM	TRC	CPU	OPTGAP
Scenario 1	Instance 1	47.86%	19.57%	-1.79%	97.83%	0.00%	100.00%	65.84%	22.164	6.34%
	Instance 2	57.30%	18.97%	56.36%	100.00%	0.00%	100.00%	84.40%	8.746	0.00%
	Instance 3	30.63%	10.61%	46.75%	100.00%	0.00%	60.00%	60.31%	17.049	1.09%
	Instance 4	14.44%	-20.45%	65.79%	100.00%	0.00%	100.00%	76.62%	8.768	0.00%
	Instance 5	2.50%	-420.00%	85.07%	0.00%	0.00%	0.00%	21.86%	3.847	0.00%
	Instance 6	11.43%	-220.00%	86.89%	0.00%	0.00%	0.00%	8.32%	2.308	0.00%
	Instance 7	35.71%	-100.00%	84.85%	0.00%	0.00%	0.00%	44.94%	1.149	0.00%
	Average	28.55%	-101.62%	60.56%	56.83%	0.00%	51.43%	51.76%	9.147	1.06%
Scenario 2	Instance 1	31.79%	15.79%	50.75%	0.00%	0.00%	100.00%	36.15%	12.76	0.00%
	Instance 2	36.07%	10.14%	44.29%	100.00%	0.00%	27.78%	83.69%	9.668	0.00%
	Instance 3	21.47%	-6.06%	39.06%	100.00%	0.00%	80.52%	76.05%	6.123	3.17%
	Instance 4	42.01%	17.81%	38.10%	100.00%	0.00%	85.71%	75.00%	9.662	0.00%
	Instance 5	12.32%	-1.45%	51.79%	100.00%	0.00%	50.00%	65.91%	9.988	0.19%
	Instance 6	-15.52%	-44.44%	66.04%	100.00%	0.00%	100.00%	72.81%	4.217	0.00%
	Average	21.36%	-1.37%	48.34%	83.33%	0.00%	74.00%	68.27%	8.736	0.56%
Scenario 3	Instance 1	43.35%	25.84%	69.44%	0.00%	0.00%	60.00%	41.73%	9.421	0.00%
	Instance 2	18.26%	7.55%	80.83%	0.00%	0.00%	100.00%	13.87%	7.049	0.00%
	Instance 3	29.01%	7.46%	66.67%	100.00%	0.00%	81.82%	25.97%	10.497	4.14%
	Instance 4	85.92%	81.48%	66.91%	100.00%	0.00%	100.00%	45.81%	5.601	0.00%
	Instance 5	49.02%	50.00%	80.74%	0.00%	100.00%	0.00%	35.07%	4.81	0.45%
	Instance 6	42.86%	48.28%	88.14%	100.00%	66.67%	0.00%	53.65%	3.106	0.00%
	Instance 7	71.88%	53.85%	76.74%	0.00%	100.00%	0.00%	57.14%	2.177	0.00%
	Instance 8	53.57%	64.29%	76.71%	0.00%	0.00%	0.00%	28.99%	0.679	0.00%
	Average	49.23%	42.34%	75.77%	37.50%	33.33%	42.73%	37.78%	5.417	0.57%
	Total average	34.38%	-18.13%	62.86%	57.04%	12.70%	54.56%	51.15%	7.609	0.73%

Table 11 Computational results in the peaked schedule (length of time window = 6 hours)

		TFD	NDF	NGR	DAR	NKM	NPM	TRC	CPU	OPTGAP
Scenario 1	Instance 1	43.02%	24.14%	23.90%	0.00%	0.00%	92.31%	32.15%	78.10	3.29%
	Instance 2	36.91%	15.97%	37.02%	0.00%	0.00%	100.00%	34.86%	58.62	6.97%
	Instance 3	25.90%	0.00%	49.76%	0.00%	0.00%	100.00%	27.01%	62.66	4.01%
	Instance 4	5.21%	-32.65%	64.88%	0.00%	0.00%	100.00%	18.06%	17.42	1.21%
	Average	27.76%	1.86%	43.89%	0.00%	0.00%	98.08%	28.02%	54.198	3.87%
Scenario 2	Instance 1	34.92%	17.11%	16.89%	0.00%	100.00%	76.92%	30.92%	71.22	7.13%
	Instance 2	37.76%	15.58%	13.61%	0.00%	0.00%	75.00%	32.09%	74.97	8.57%
	Instance 3	30.63%	-1.53%	3.57%	0.00%	100.00%	55.56%	21.89%	88.83	8.39%
	Instance 4	31.60%	10.38%	24.11%	0.00%	0.00%	55.56%	14.61%	44.25	3.72%
	Instance 5	18.97%	-14.12%	26.76%	0.00%	0.00%	40.00%	12.84%	52.96	0.87%
	Instance 6	-30.65%	-61.76%	56.85%	0.00%	100.00%	0.00%	0.88%	27.95	0.35%
	Average	20.54%	-5.72%	23.63%	0.00%	50.00%	50.51%	18.87%	60.031	4.84%
Scenario 3	Instance 1	47.59%	21.43%	50.00%	100.00%	0.00%	45.45%	44.83%	59.35	4.60%
	Instance 2	35.59%	14.17%	60.15%	100.00%	100.00%	100.00%	53.20%	29.85	7.31%
	Instance 3	41.77%	24.14%	52.79%	0.00%	0.00%	93.33%	42.27%	77.21	5.67%
	Instance 4	92.25%	92.31%	72.62%	0.00%	100.00%	100.00%	80.97%	13.41	0.00%
	Instance 5	87.21%	81.36%	77.57%	0.00%	60.00%	0.00%	70.18%	10.80	0.42%
	Instance 6	70.97%	69.39%	90.36%	0.00%	80.00%	0.00%	71.00%	5.37	0.00%
	Average	62.56%	50.46%	67.25%	33.33%	56.67%	56.46%	60.41%	32.662	3.00%
	Total average	38.10%	17.24%	45.05%	12.50%	40.00%	64.63%	36.74%	48.310	3.91%

Table 12 Computational results in the peaked schedule (length of time window = 9 hours)

		TFD	NDF	NGR	DAR	NKM	NPM	TRC	CPU	OPTGAP
Scenario 1	Instance 1	39.59%	22.06%	35.52%	0.00%	33.33%	100.00%	40.79%	232.57	7.27%
	Instance 2	41.95%	21.99%	43.55%	0.00%	100.00%	100.00%	46.26%	118.76	5.86%
	Average	40.77%	22.02%	39.53%	0.00%	66.67%	100.00%	43.52%	175.665	6.57%
Scenario 2	Instance 1	35.01%	5.85%	5.56%	0.00%	100.00%	80.00%	30.09%	216.69	12.02%
	Instance 2	40.00%	10.12%	21.61%	100.00%	0.00%	0.00%	30.96%	64.54	11.91%
	Instance 3	39.21%	4.67%	21.09%	0.00%	100.00%	33.33%	29.41%	165.06	7.44%
	Instance 4	47.25%	22.05%	41.89%	0.00%	0.00%	55.56%	42.85%	85.36	3.66%
	Instance 5	38.96%	15.65%	52.48%	0.00%	0.00%	100.00%	43.01%	94.85	1.23%
	Average	40.09%	11.67%	28.53%	20.00%	40.00%	53.78%	35.26%	125.300	7.25%
Scenario 3	Instance 1	-93.58%	-175.44%	38.87%	0.00%	100.00%	99.16%	59.06%	139.20	9.79%
	Instance 2	41.01%	28.08%	58.99%	0.00%	100.00%	100.00%	52.01%	55.77	7.94%
	Instance 3	40.63%	25.00%	51.62%	0.00%	0.00%	94.29%	46.46%	112.76	9.82%
	Instance 4	91.45%	91.59%	75.35%	0.00%	100.00%	100.00%	81.01%	16.44	0.00%
	Average	19.88%	-7.69%	56.21%	0.00%	75.00%	98.36%	59.64%	81.042	6.89%
	Total average	32.86%	6.51%	40.59%	9.09%	57.58%	78.39%	45.63%	118.363	7.00%

Table 13 Computational results in the de-peaked schedule (length of time window = 3 hours)

		TFD	NDF	NGR	DAR	NKM	NPM	TRC	CPU	OPTGAP
Scenario 1	Instance 1	45.79%	26.60%	12.50%	66.67%	0.00%	100.00%	37.64%	14.92	0.91%
	Instance 2	30.43%	10.64%	54.55%	100.00%	0.00%	0.00%	6.17%	8.08	1.12%
	Instance 3	35.84%	21.62%	46.75%	100.00%	0.00%	40.00%	30.20%	17.10	5.34%

	Instance 4	19.67%	10.45%	63.16%	0.00%	0.00%	100.00%	24.92%	14.50	1.60%
	Instance 5	22.78%	6.98%	73.13%	0.00%	0.00%	100.00%	24.63%	6.07	2.09%
	Instance 6	17.39%	8.33%	81.97%	0.00%	0.00%	100.00%	16.16%	3.28	0.00%
	Instance 7	25.93%	-25.00%	66.67%	0.00%	0.00%	0.00%	22.70%	1.65	0.00%
	Average	28.26%	8.52%	56.96%	38.10%	0.00%	62.86%	23.20%	9.369	1.58%
Scenario 2	Instance 1	24.03%	15.94%	46.27%	0.00%	0.00%	100.00%	25.30%	10.57	0.15%
	Instance 2	42.50%	17.65%	57.14%	0.00%	0.00%	85.71%	35.31%	10.63	0.18%
	Instance 3	29.81%	12.50%	50.00%	100.00%	0.00%	-27.27%	16.74%	6.85	0.00%
	Instance 4	34.21%	18.06%	53.97%	0.00%	0.00%	71.43%	26.25%	11.74	0.50%
	Instance 5	19.58%	-2.90%	32.14%	0.00%	0.00%	70.00%	13.92%	9.75	0.00%
	Instance 6	17.78%	-2.50%	66.04%	100.00%	0.00%	100.00%	28.34%	6.27	0.00%
	Average	27.99%	9.79%	50.93%	33.33%	0.00%	66.65%	24.31%	9.302	0.14%
Scenario 3	Instance 1	34.69%	20.48%	68.52%	0.00%	0.00%	14.29%	33.46%	8.31	1.95%
	Instance 2	35.83%	21.43%	77.50%	0.00%	0.00%	100.00%	22.55%	7.06	0.00%
	Instance 3	34.67%	13.85%	65.04%	100.00%	0.00%	100.00%	30.28%	9.85	0.83%
	Instance 4	85.51%	82.14%	66.91%	100.00%	0.00%	100.00%	47.39%	6.81	0.00%
	Instance 5	57.38%	55.17%	80.74%	0.00%	100.00%	0.00%	40.03%	5.63	0.45%
	Instance 6	1.59%	-10.34%	80.51%	100.00%	66.67%	0.00%	30.48%	4.75	0.00%
	Instance 7	79.37%	75.00%	76.74%	0.00%	100.00%	0.00%	64.64%	2.42	0.00%
	Instance 8	60.61%	68.75%	76.71%	0.00%	0.00%	100.00%	35.04%	0.82	0.00%
	Average	48.70%	40.81%	74.08%	37.50%	33.33%	51.79%	37.98%	5.707	0.40%
	Total average	35.97%	21.18%	61.76%	36.51%	12.70%	59.72%	29.15%	7.955	0.72%

Table 14 Computational results in the de-peaked schedule (length of time window = 6 hours)

		TFD	NDF	NGR	DAR	NKM	NPM	TRC	CPU	OPTGAP
Scenario 1	Instance 1	46.78%	28.16%	30.24%	0.00%	0.00%	84.85%	35.77%	66.09	3.76%
	Instance 2	45.68%	29.46%	42.79%	0.00%	0.00%	90.91%	43.60%	55.42	3.10%
	Instance 3	49.48%	38.36%	48.31%	0.00%	0.00%	75.00%	45.60%	67.65	5.03%
	Instance 4	36.08%	21.57%	62.50%	0.00%	0.00%	100.00%	35.36%	39.95	0.32%
	Average	44.51%	29.39%	45.96%	0.00%	0.00%	87.69%	40.08%	57.277	3.05%
Scenario 2	Instance 1	46.31%	25.47%	12.16%	0.00%	0.00%	100.00%	38.76%	92.22	7.65%
	Instance 2	53.67%	27.33%	16.33%	0.00%	0.00%	100.00%	47.09%	58.09	5.57%
	Instance 3	42.77%	18.87%	13.57%	100.00%	100.00%	84.62%	37.35%	62.27	7.29%
	Instance 4	48.75%	23.61%	20.57%	0.00%	0.00%	77.78%	34.09%	67.43	4.43%
	Instance 5	40.84%	12.33%	20.42%	0.00%	0.00%	100.00%	34.43%	64.28	4.18%
	Instance 6	37.19%	20.21%	50.00%	0.00%	100.00%	66.67%	35.46%	46.09	0.51%
	Average	44.92%	21.30%	22.18%	16.67%	33.33%	88.18%	37.86%	65.061	4.94%
Scenario 3	Instance 1	42.07%	19.05%	56.91%	100.00%	0.00%	65.22%	44.48%	50.55	0.90%
	Instance 2	51.71%	28.99%	60.54%	100.00%	100.00%	100.00%	61.95%	43.57	3.71%
	Instance 3	55.37%	35.23%	51.67%	0.00%	0.00%	94.34%	53.32%	39.69	7.59%
	Instance 4	92.17%	92.73%	72.62%	100.00%	100.00%	100.00%	84.21%	14.54	0.00%
	Instance 5	86.59%	81.36%	77.57%	0.00%	66.67%	0.00%	69.37%	11.43	0.42%
	Instance 6	36.73%	25.58%	84.94%	0.00%	80.00%	0.00%	46.90%	5.90	0.00%
	Average	60.77%	47.15%	67.38%	50.00%	57.78%	59.93%	60.04%	27.614	2.10%
	Total average	50.76%	33.02%	45.07%	25.00%	34.17%	77.46%	46.73%	49.072	3.40%

Table 15 Computational results in the de-peaked schedule (length of time window = 9 hours)

		TFD	NDF	NGR	DAR	NKM	NPM	TRC	CPU	OPTGAP
Scenario 1	Instance 1	47.74%	30.92%	25.14%	0.00%	33.33%	96.30%	46.62%	98.51	8.81%
	Instance 2	49.53%	32.96%	35.16%	0.00%	100.00%	95.45%	52.71%	103.54	8.14%
	Average	48.64%	31.94%	30.15%	0.00%	66.67%	95.88%	49.66%	101.025	8.48%
Scenario 2	Instance 1	46.60%	25.00%	3.17%	100.00%	100.00%	100.00%	42.47%	268.24	12.61%
	Instance 2	51.68%	25.11%	9.52%	0.00%	0.00%	89.47%	43.77%	182.89	7.74%
	Instance 3	50.00%	23.94%	22.91%	0.00%	100.00%	87.50%	44.17%	287.12	4.21%
	Instance 4	58.93%	35.57%	36.15%	0.00%	0.00%	83.33%	53.70%	226.54	3.65%
	Instance 5	51.72%	28.89%	45.04%	0.00%	0.00%	100.00%	51.01%	120.45	5.01%
	Average	51.79%	27.70%	23.36%	20.00%	40.00%	92.06%	47.02%	217.046	6.64%
Scenario 3	Instance 1	-123.76%	-220.75%	41.29%	0.00%	100.00%	98.39%	52.74%	127.06	6.75%
	Instance 2	55.40%	34.52%	48.31%	0.00%	100.00%	100.00%	60.09%	100.45	10.81%
	Instance 3	55.36%	37.96%	41.30%	0.00%	0.00%	100.00%	56.21%	266.32	11.24%
	Instance 4	92.46%	92.56%	75.35%	0.00%	100.00%	100.00%	85.33%	17.08	0.00%
	Average	19.87%	-13.93%	51.56%	0.00%	75.00%	99.60%	63.59%	127.726	7.20%
	Total average	39.61%	13.33%	34.85%	9.09%	57.58%	95.49%	53.53%	163.471	7.18%