A Rapid Solving Method to Large Airline Disruption Problems Caused by Airports Closure

ZHENG TIAN WU1,4, QING GAO3, BENCHI LI2, CHUANG YIN DANG2, AND FUYUAN HU1

1School of Electronic and Information Engineering, Suzhou University of Science and Technology, Suzhou 215000, China
2Department of Systems Engineering and Engineering Management, City University of Hong Kong, Hong Kong
3Institute for Automatic Control and Complex Systems (AKS), Faculty of Engineering, University of Duisburg-Essen, 47057 Duisburg, Germany
4Virtual Reality Key Laboratory of Intelligent Interaction and Application Technology of Suzhou, Suzhou 215000, China

Zhengtian Wu (wzht8@mail.usts.edu.cn); Qing Gao (gaoqing@iim.ac.cn)

This work was supported in part by RGC of the Government of Hong Kong under Grant 11301014, in part by the National Nature Science Foundation of China under Grant 71771161, Grant 71471091, Grant 71271119, Grant 61672371, Grant 51375323, and Grant 61472267, in part by the Natural Science Fund for colleges and universities in Jiangsu Province under Grant 17KJD110008, in part by the Key Research and Development Plan of Jiangsu Province under Grant BE2017663, in part by the Foundation of Key Laboratory in Science and Technology Development Project of Suzhou under Grant ZZS201609, in part by the Alexander von Humboldt Foundation of Germany, and in part by the Jiangsu Provincial Department of Housing and Urban-Rural Development under Grant 2017zd253.

ABSTRACT Airline disruption is universal phenomenon that block the originally scheduled flights, and the slow recovery scheduling causes a lot of losses to both the airline companies and the passengers. However, it is very difficult to generate a new recovery scheduling in a reasonable time by traditional methods, especially for large dimension airline disruption problem. To deal with this problem rapidly, the airline disruption problem will be reformulated as integer programming and a distributed network, which is based on the fixed-point iterative method for integer programming, will be developed in this paper. In response to the airport closure, an airplane reschedule is constructed by these feasible flight lines. In the implementation of distributed network, a certain number of independent segments are obtained by dividing the solution space. As a feature of this fixed-point method, the number of partial feasible flight lines, which have to be calculated for finding an optimized airplane reschedule, is much fewer compared with the number needed by CPLEX CP Optimizer. This is the first distributed integer programming method to large airline disruption problems caused by airports closure. In the numerical results, the proposed distributed approach shows promising, especially for solving the large dimension airline disruption problem.

INDEX TERMS Airline disruption management, irregular operation, airports closure, integer programming, distributed computation.

I. INTRODUCTION

Generally, a disruption situation which is one of the major challenges that the airline industry currently faces stems from a local urgency event, for example an airport closure, a flight delay, or an aircraft maintenance has problem, although the airlines put considerable effort into scheduling. The disruptions of one flight always have an important impact on the following flights, so lots of flight cancelations and delays will be generated because of the former delay [1]. The European airline has an in-depth investigation about the flights delay and finds that even bad weather condition causes only about 3.6% flight delay, however, the spread of the delayed flights create up to 15.1% delay [2]. In other words, it is more important to formulate the airline disruption problem and solve the formulation for rescheduling rapidly than to research the sources of disruptions.

Numerous literatures reports different novel approaches for solving this tough problem, i.e. airline disruption problems caused by airports closure. The paper [3] identified the demerits of scheduling at Heathrow airport because of a short airport closure. In this closure, system disruptions assessed consist of flight rerouting, flight delays, flight cancellations, and flight diversions to alternate airports. Additionally, both the influence on fuel consumption and CO2 emissions were analyzed employing the Reorganized Air Traffic Control Mathematical Simulator Plus simulation model and the Advanced Emission Model tool. Then, the relationship between severe weather events associated with climate...
change and airport operations disruption can be assessed by the results and it helps provide base for finding methods to deal with airline disruption problems originated from airports closure. The research [4] analyzed the effectiveness of tourism-oriented airports to arrange departing passengers in case of an unforeseen airport closure due to weather events, terrorist attacks and industrial actions and conducted a case study on the most occupied tourism-oriented island airport in Europe, i.e. Palma de Mallorca Airport (PMI). As a flash point, the paper considered the full passenger itineraries by using air passenger demand data which help to model an airline recovery process in event of an airport closure. For the sake of reducing the impact on stranded tourists, the analysis on policy development is concentrated. The novel methodological process include generating a baseline travel dataset, simulating all the 19 closure scenarios and sequentially relocating the influenced passengers. A framework [5], [6] which is based on a basic model established as a time-space network has been developed to help control schedule perturbations caused by the temporary airports closure. Those models contribute to developing many strategic network models for scheduling and are translated to either pure network flow problems or network flow problems which are then computed employing the network simplex method and a Lagrangian relaxation-based algorithm, respectively. A duty-based formulation [7] is proposed to deal with the crew recovery problem. The approach can reduce the problem size by resolving the disruption within every duty period and hence shortening recovery horizons. In the branch-and-price-based solution method, the master problem is expressed in a set covering formulation and the pricing problem is expressed in a resource constrained shortest path. The novel method was examined on numerous scenarios ranging from single flight delay to airport closure. The conclusions indicate the computational cost is acceptable for operational environment. The paper [8] employs a Method of Inequality-based Multi-objective Genetic Algorithm (MMGA) in which a traditional Genetic Algorithm (GA) and a multi-objective optimization method are combined to generate an efficient multi-fleet aircraft routing algorithm for the schedule disruption of short flights. The combination enables MMGA to address multiple objectives simultaneously and explore the best solution. Although Operations Research (OR) techniques expressed by a precise mathematical model is traditionally employed to deal with the airline schedule disruption management problem, it is difficult to define a precise model for airline operations including numerous factors. However, in the mentioned paper, it is efficient to recover the perturbation relying on Multi-objective Optimization Airline Disruption Management by GA and it is confirmed by the experimental results. Furthermore, the results in the manuscript demonstrate that the method can be extended as a real-time decision support method for relatively practical complex airline operations.

A local search heuristic method [9], which has the ability to find suitable revised flight schedules in a certain good quality within 10 seconds, has been developed. However, the examples which are used in the paper are generated randomly rather than practical airline schedules. An optimization model [10] contained by both a set partitioning problem and a route generation procedure has been presented to reschedule flight routes by minimizing an objective function that is about the cancelation costs. A mixed integer multi-commodity flow model [11] with side constraints has proposed. Under the Dantzig-Wolfe decomposition this model becomes a set packing model. There are two instances tested in this paper, and the larger instance costs much more time than the smaller one. A two-steps framework [12] has been developed to construct a real-time schedule. An introduction to airline disruption management and experiences from project Descartes has been given [13]. A preliminary idea of changing airline disruption problem into the integer programming has been introduced in the paper [14]. Further research about this idea has been developed by the paper [15], [16] and [17]. To obtain more theoretical descriptions on airline disruption scheduling, one can refer to the review paper [18]–[22] and books [23], [24], [25], [26], [27], [28]. For the potential methods which can be used to airline disruption management, one can consult to the papers [29]–[35].

As is known, lots of losses have made due to the slow recovery scheduling. However, in the process of recovery scheduling it is very hard to find all the feasible flight lines in a reasonable time, especially for the large airline disruption problem. Only partial feasible flight lines are necessary. In this research, the airline disruption problem is divided into two subproblems: one is feasible flight lines generation problem and the other is airplanes reschedule problem. After dividing the solution space into several segments, a distributed network which is based on Dang and Ye’s algorithm [36] is developed to obtain feasible flight lines. When the large airplanes reschedule problem is solved by partial feasible flight lines, the comparisons have been made. According to the literature reviews, Liu’s method [8] has the best performance. Therefore, we only compare our method with Liu’s method, as well as our former Dang’s method.

In this paper, the airline disruption problems have been formulated to the integer programming. Therefore, the computational complexity of this approach is the same with the integer programming. As is known, the integer problem is NP-hard and it is very difficult to solve by traditional methods, especially for large dimension problem. Theoretically, Dang and Ye’s methods are very suitable for solving this kind of problems. What is more, a distributed implementation of Dang and Ye’s methods have also been developed to calculate the obtained integer formulation. The flight schedule instances used in this paper comes from the paper [8]. Comparisons between the solutions, which are obtained by partial feasible flight lines which are solved by Dang and Ye’s algorithm and those obtained by Liu’s method [8] for airplanes reschedule problems, will be provided in this manuscript. After more relatively partial feasible flight lines by Dang and Ye’s algorithm are provided, final solutions which have less total delay time than the solutions generated by Liu’s method.
can be shown. Therefore, one can see that the Dang and Ye’s algorithm obviously outperform the other methods.

The rest organization of this paper can be described as follows. Section II gives the introduction of the mathematical formula of this airline disruption problem and dividing the problem into two kinds of subproblems. As a new method for integer programming, Dang and Ye’s iterative method is proposed in Section III. Section IV develops a distributed computation implementation to find feasible flight lines. In Section V, performance comparisons and some discussions are given. Section VI offers a conclusion of this research.

II. PROBLEM FORMULATION

In this paper, one can consider a situation that a thunderstorm happens, and it causes that one or more airports are temporarily closed. All flights, which are served by the closed airports, are suspended until the airports reopen again. A minimum impact from the thunderstorm can be obtained after reschedule of the airline by reassigning of all the flight legs to all the airplanes. In order to obtain an airplanes reschedule which can minimize deviation from the original schedule and the loss of disruption, the procedure of solving this airline disruption problem is divided into two subproblems: one is feasible flight lines generation problem and the other is airplanes reschedule problem.

The notations used in the formulation can be introduced as follows.

Indices

\( i, j, k \) the indices of flight

\( t \) the index station

Sets

\( F \) the set of all the flights

\( S \) the set of all the stations

Parameters

\( t_{d_i} \) duration time of a flight \( i \)

\( t_{t_i} \) turnaround time for a flight \( i \)

\( T_j \) total time from the departure of flight \( j \) to the departure curfew time of a station

\( d_{it} \) = 1, if flight \( i \) leaves the station \( t \); = 0, if not

\( a_{it} \) = 1, if flight \( i \) reaches the station \( t \); = 0, if not

\( e_{ijk} \) = 1, if flight \( k \)'s destination station and flight \( j \)'s original station are the same station, and flight \( k \)'s original station and flight \( j \)'s destination station are the same station; = -1, if flight \( i \)'s destination station and flight \( k \)'s original station are the same station, or flight \( i \)'s destination station and flight \( k \)'s destination station are the same station, or flight \( i \)'s original station and flight \( k \)'s original station are the same station, or flight \( i \)'s original station and flight \( k \)'s destination station are the same station; = 0, otherwise

\( e_{nj} \) = -1, if flight \( j \)'s destination station is the station \( t \), or flight \( j \)'s original station is the station \( t \); = 0, otherwise

Variables

\( x_{it} \) = 1 if the \( i \) which is flight leg is contained among a feasible flight line; = 0, if not

\( s_t \) = 1 if the feasible flight line’s starting point is the station \( t \); = 0, if not

\( k_j \) = 1 if the feasible flight line’s destination point is the station \( t \); = 0, if not

The mathematical formulation of feasible flight line generation

\[ \exists x_{it}, s_t, k_j \in \{0, 1\}, \quad \forall i \in F, \quad \forall t \in S, \]  

subject to

(1a)

\( \text{(limitation on flight time)} \)

\[ \sum_{i \in F} x_{it}(t_{d_i} + t_{t_i}) \leq T_j, \quad \forall j \in F, \]  

(1b)

\( \text{(node conservation)} \)

\[ \sum_{i \in F} x_{it}a_{it} = \sum_{i \in F} x_{it}d_{it} + s_t - k_j = 0, \quad \forall t \in S, \]  

(1c)

\( \text{(flow from source node)} \)

\[ \sum_{i \in F} x_{it}d_{it} \geq s_t, \quad \forall t \in S, \]  

(1d)

\( \text{(flow to sink node)} \)

\[ \sum_{i \in F} x_{it}a_{it} \geq k_j, \quad \forall t \in S, \]  

(1e)

\( \text{(source node cover)} \)

\[ \sum_{t \in S} s_t = 1, \]  

(1f)

\( \text{(sink node cover)} \)

\[ \sum_{t \in S} k_j = 1, \]  

(1g)

\( \text{(subroute elimination)} \)

\[ \sum_{i \in F} x_{it}e_{ijk} + \sum_{t \in S} s_t e_{nit} + \sum_{t \in S} k_j e_{nt} \geq 1, \quad \forall j \in F, \quad \forall k \in (j + 1, \cdots, \text{Card}(F) - 1). \]  

(1h)

More details of the formulation and the feasible transformation pseudo code can be found in the previous work [16]. However, each of these two paper focuses on different disruption. This manuscript focuses on disruption problems caused by airports closure, while the other paper pays more attention on disruption problem caused by groundings. What is more, the new method has obvious advantage on large dimension disruption problems caused by airports closure.

The other subproblem is formulated as a resource assignment problem which is very hard to solve as introduced in the research [37], [38]. However, after a certain number of partial feasible flight lines of the formulation (1) obtained using the distributed fixed-point method in Section III, this airplanes rescheduled problem could be worked out easily.

The formulation for the airline rescheduled subproblem can be introduced as follows. To simplify the problem, the assumption is adopted that all airplanes are derived from the same fleet.
The notations, which are used in the formulation, can be described as follows.

**Indices**
- $i$: the indices of flight
- $j$: the indices of route
- $t$: the index of station

**Sets**
- $F$: the set of all flights
- $S$: the set of all stations
- $P$: the set of all feasible flight lines

**Parameters**
- $x_{ij}$ = 1, if $i$, which stands for a flight, is contained in the feasible flight line $j$; = 0, if not
- $s_{ij}$ = 1, if the feasible flight line $j$'s starting point is the station $t$; = 0, if not
- $k_{ij}$ = 1, if the feasible flight line $j$'s destination point is the station $t$; = 0, if not
- $d_{ij}$: the feasible flight line $j$'s delay time
- $h_t$: number of airplane needed to leave the starting point $t$
- $g_t$: number of airplane needed to arrive at the sink point $t$
- $TN$: the sum number of all airplanes

**Variables**
- $y_j$ = 1, if the feasible flight line $j$ is rescheduled to an airplane; = 0, if not
- $z_i$ = 1 if the flight $i$ is canceled; = 0, if not

The mathematical formulation of airplane reschedule

\[
\begin{align*}
\text{minimize} & \quad \sum_{j \in P} dt_{ij}y_j \quad (2a) \\
\text{subject to} & \\
& (\text{flight cover}) \quad \sum_{j \in P} x_{ij}y_j + z_i = 1, \quad \forall i \in F, \quad (2b) \\
& (\text{airplanes balance of source stations}) \quad \sum_{j \in P} s_{ij}y_j = h_t, \quad \forall t \in S, \quad (2c) \\
& (\text{airplanes balance of sink stations}) \quad \sum_{j \in P} k_{ij}y_j = g_t, \quad \forall t \in S, \quad (2d) \\
& (\text{total available airplanes}) \quad \sum_{j \in P} y_j = TN, \quad (2e) \\
& (\text{binary assignment}) \quad y_j \in \{0, 1\}, \quad \forall j \in P, \quad (2f) \\
& (\text{binary assignment}) \quad z_i \in \{0, 1\}, \quad \forall i \in F. \quad (2g)
\end{align*}
\]

For the details of the explanation for this formulation, one can also reference the paper [16], while all the variables are on the conditions of airports closure. Traditionally, CPLEX Optimizers Concert Technology is applied to calculate the formulation (2) of this kind, whose solutions are combined with feasible flight lines. Sometimes more than one optimum solutions exist. However, the number of preliminary flight lines are different among the optima. One can see that the more preliminary flight lines contained in a combination of the feasible flight lines, the better. This is because more preliminary flight lines in the combination imply that the reschedule has less deviation from the original schedule. In this paper a distributed fixed-point iterative integer programming is developed to obtain the combination which contains the maximum number of preliminary flight lines.

### III. METHODOLOGY

As known, the CPLEX can solve both the formulation (1) and the formulation (2). However, for the airline disruption problem with large dimension it is impossible to find all the formulation (1)'s feasible flight lines using CPLEX CP Optimizer tool. Let’s explain this through an instance. In Section V, an instance includes 140 legs of flight. If each flight line’s length is limited between 10 and 13 legs of flight in the original arrangement, then the upper limit number of feasible flight lines of the formulation (1) will be $C_{140}^{10} + C_{140}^{11} + C_{140}^{12} + C_{140}^{13} = 7.98 \times 10^{17}$. The upper time-limit of finding all the feasible flight lines of the formulation (1) for this instance through CPLEX CP Optimizer tool can be obtained by a simple calculation. Generally, it costs about 21762ms to obtain 100000 feasible flight lines using the CPLEX CP Optimizer tool, and the average computing time to obtain a feasible flight line is more than 0.21762ms because more time will be cost to obtain a feasible flight line when the search tree goes deeper. In order to find all the feasible flight lines which is about $7.98 \times 10^{17}$, it will cost more than 17362360779542.113s which is equal about 5505568.423years in the worst case using CPLEX CP Optimizer.

To solve the above issue, a distributed implementation of Dang and Ye’s algorithm is developed. This algorithm has a good performance to find feasible flight lines through dividing the solution space of the formulation (1) into a certain number of segments. This distributed method works well even when the dimension of the problem is very large. This Dang and Ye’s algorithm, which has been developed in the research [36], is the improvement of Dang’s iterative method [39]. Numerical calculation has show that the former method requires fewer number of linear programming and costs less time to find an integer solution than the later method. Firstly, a simple example will be given to illustrate Dang and Ye’s algorithm.

A polytope $P$ will be defined to explain this method. Let $P = \{x \in \mathbb{R}^2 | Ax \leq b \}$ with

\[
A = \begin{pmatrix} -17 & 2 \\ 6 & 5 \\ -3 & -3 \end{pmatrix} \quad \text{and} \quad b = \begin{pmatrix} -8 \\ 4 \\ 7 \end{pmatrix}
\] (3)
and let a lattice \( D(P) = \{x \in \mathbb{Z}^2 | x^l \leq x \leq x^u\} \) which can be described in Figure 1, the problem intend to find a \( x(x \in P \cap D(P)) \) or prove that there is no such point exists.

The idea to calculate this problem is to state an increasing-mapping. This mapping is defined form the lattice into itself. All the integer solutions, which are outside the polytope \( P \), are mapped to the first integer point in this polytope. This first integer point is not bigger than them which are in the lexicographical order or \( x^l \). All integer solutions, which are inside the \( P \), are fixed points by this procedure. After a certain number of iterations, this procedure either proves no such integer point exists or obtains an integer solution in \( P \) for any given initial integer point. After a simple improvement, all the integer solutions in the polytope could be found sequentially [36]. This method has been widely used in many applications, such as market split problem [40], Nash equilibria problem [41] and so on. The research [36], [39] has given the details of this method. It is stated that the more preliminary flight lines contained in a combination of the feasible flight lines, the better. One feature of the Dang and Ye’s algorithm for integer programming is that the integer solution near the starting point will be found firstly. This feature could be explained by the Figure 1. Under this feature, more preliminary flight lines contained in a combination of the feasible flight lines will be obtained.

At the beginning of this distributed implementation, the solution space is split up to a certain number of parts, and the computing integer points is conducted simultaneously in each part by Dang and Ye’s algorithm. Since the divided parts are self-governed to each other, the computing procedure for integer points in each segment could be executed simultaneously in each computer and processor. Two division methods have been developed to solve airline disruption problems [16].

One division method, which have a good performance, is dividing the solution space into a certain number of segments which contain nearly equal integer points in \( D(P) \). For a large dimension problem, if a sufficient number of computers and processors are provided, the solution space could be divided into subsegments small enough, each of subsegments can be solved in a reasonable time. If the computing of some subsegment lasts too long or there is no more processors can be given, another division method has been developed. This cutting method is designed especially for airline disruption problems. In this method, the preliminary flight lines are taken as the initial points, subsegments are divided by 2 bound points which are among two consecutive initial points by the lexicographical order. Then the cluster is stated around each initial point in each segment for computing, and a parameter is applied to control the feasible flight lines’ number. The details of the two division methods have been presented in the previous work [16].

IV. IMPLEMENTATION OF THE DISTRIBUTED NETWORK

Each subsegment is divided through the split method in the paper [16] is self-governed to each other. Therefore, the computing in each part could be done synchronously from the start point to the end point by each independently computing processor. In order to take greater advantage of hardware, the communication network among the individual computers is developed by Message Passing Interface (MPI), the a parallel computing among the processors in a same computer is conducted by OpenMP. The flow diagram of the distributed computing process can be described in Figure 3.

The pseudo code of developing the distributed computing network by OpenMP and MPI can be described as Figure 4.
for the rank of an individual computer which begins from 0 and is kept as a variable `computerid`. A independent file will be created as configuration file which contains all the computer’s names. The master computer which takes charge of controlling the procedure of all computing is defined in the first part of this configuration file, and the master computer’s `computerid` is equal to 0. The other slave ones are assigned a certain number from one to the sum number of slave computers. This configuration file has to be copied into each participating computer, as well as the execution file of the distributed computing programming. All execution files in every computer have to be executed synchronously using MPI order. At the beginning of execution procedure, the number of processors in each computer feedbacks to a variable `SystemInfo.dwNumberOfProcessors`. Firstly the number of each slave’s processors is sent from the individual slave to the master computer, then the sum number of processors of all the computers can be obtained. In the master computer, the solution space is divided by one of the two division methods as mentioned in the previous work [16] on account of the sum number of processors. After that, each part’s beginning point $x_{s_i}$ and the last point $x_{e_i}$ are sent to the corresponding slave computer from master computer. If a slave computer has more than one processors, there is a parallel computation will be carried out by OpenMP through a compiler directive “#pragma omp parallel” which will split up the loop iterations. If the number of parts assigned to a computer is equal to the number the computer’s processors, computing on all the parts could be done synchronously too. If the number of parts assigned to a computer is more than the number of this computer’s processors, computing on the surplus parts could begin at the moment of any of its processors completes its work. As soon as the computing on all the parts in an individual computer completes, the feasible flight lines obtained will be transmitted to the master computer in order to calculate the formulation (2).

In order to make more processors can work together to enhance the efficiency of computing, the number of subsegments is matching to the number of processors. On one hand the efficiency of the proposed distributed computation is high. On the other hand, the feasible flight lines found using the division approach in the paper [16] outperform those performance found using CPLEX CP Optimizer when the feasible flight lines are used for solving the formulation (2). In addition to this, solutions using feasible flight routes calculated by Dang and Ye’s algorithm through the second division method in paper [16] have much less total delay time. Numerical result comparisons are given in Section V.

V. NUMERICAL RESULTS
In this paper, there are two flight schedule instances which are from Taiwanese Domestic Airlines. The instance are MD-90 fleet and DH-8 fleet which are firstly used in the paper [8]. The constraints and assumptions used in paper [8] are also
Distributed computation implementation

begin
MPI_Init(argc, argv);
MPI_Comm_rank(MPI_COMM, computerid)
GetSystemInfo(SystemInfo)
if(computerid == 0)
begin
receive the parameter
SystemInfo.dwNumberOfProcessors from the computers if the computerid ≠ 0
split up the solution space into certain number of subsegment which are on account of the sum number of all processors and calculate the numerical value of \( x^r \) and \( x^s \) defining each subpart
send \( x^r \) and \( x^s \) to each computer with computerid ≠ 0
#pragma omp parallel
for(i=0;i< SystemInfo.dwNumberOfProcessors;i++)
begin
calculate the feasible flight lines of the formulation (1) in the rest of parts using Dang and Ye’s algorithm or CPLEX CP Optimizer tool
receive flight lines of feasible from each computer with computerid ≠ 0
end
else
begin
send the parameter
SystemInfo.dwNumberOfProcessors to the computer with computerid = 0
receive \( x^r \) and \( x^s \) from the computer with computerid = 0
#pragma omp parallel
for(i=0;i< SystemInfo.dwNumberOfProcessors;i++)
begin
calculate feasible flight lines of formulation (1) in received parts through Dang and Ye’s algorithm or CPLEX CP Optimizer
send feasible flight lines to the computer with computerid = 0
end
end

FIGURE 4. Distributed computing network pseudo code.

adopted in this paper. MD-90 fleet makes up of 7 airplanes in the same fleet that serve 70 flights between 6 cities a day, and it works in a hub-and-spoke system. DH-8 fleet consists of 12 airplanes in the same fleet which serve 140 flights between 11 cities a day, and it is operated in a combined point-to-point and hub-and-spoke system. Details of the two schedule instances can be obtained in the paper [8]. This paper intends to get a feasible solution which has less total delay time when the stations TSA and TXG are restarted after a 60 minutes temporary closure happens.

In this paper, a distributed computing network is built by MPI to find the feasible flight lines formulation (1). This network hardware makes up of 3 different individual computers. One computer has 16 threads and the other two have 2 threads. The C++ is used for coding all programs, and the CPLEX Concert Technology with 12.6.1 version is used for solving the linear programming which has to be solved in Dang and Ye’s algorithm. Firstly, the solution space of the formulation (1) will be divided into several parts through the initial seeds cluster division method, and the feasible flight lines of the formulation (1) are obtained by CPLEX CP Optimizer tool and [36]’s algorithm separately in every parts. All the feasible flight lines are applied to generate a solution of the formulation (2) with the situation that stations TSA and TXG are temporarily closed for one hour because of a thunderstorm. CPLEX Optimizers’s Concert Technology is applied to solve the formulation (2). After the optimal values of the formulation (2) has been calculated through the partial feasible flight lines found by CPLEX CP Optimizer tool and Dang’s algorithm respectively, a comparison is presented. After the optimal values of the formulation (2) has been calculated through the partial feasible flight lines found by Dang and Ye’s algorithm and Liu’s method respectively, another comparison will be given too.

In the paper [14], Figure 5(a) shows the capabilities of the partial feasible flight lines of the formulation (1) obtained through CPLEX CP Optimizer and Dang’s algorithm to solve the formulation (2). When the cluster size is increasing, the total delay time of a solution solving by Dang’s algorithm is decreasing in MD-90 fleet. In other words, the more partial feasible flight lines obtained by Dang’s algorithm, the less total delay time will be got. However, it is not the same for the method of CPLEX CP Optimizer. For the DH-8 fleet, the similar result as the MD-90 could be obtained. For explanation of this result, one can consult the Figure 5(b).

Because Dang and Ye’s algorithm is the improvement of Dang’s iterative method, the Dang and Ye’s algorithm certainly performance better than the CPLEX CP Optimizer. Both of the two methods could be the core for integer programming in the distributed computing network. The numerical comparison of these two methods will be given in Table 3.

The solutions, which calculated by the partial feasible flight lines obtained by Dang and Ye’s algorithm in some cluster sizes and a pareto optimal set found in [8], is presented in Table 1 and Table 2. The solutions of MD-90 fleet are shown in the Table 1 while the solutions of DH-8 fleet are presented in the Table 2. A multi-objective genetic approach has been developed in the research [8] to the airline disruption problem. In the paper [8], each population stands for an airline schedule which is consisted of the flight line of each airplane and the size of population is 100. The pareto optimal sets in Liu’s method is got after 50000 generations
The total computational time of calculating the airline disruption problem should contain both the computational time of solving the formulation (1) and the computational time of solving the formulation (2). Considering the original schedules are usually determined some days before the execution of schedules [18]. The partial feasible flight lines for formulation (1) could be began to obtained once the original schedules are ready. There is no need to wait till a disruption happened that the feasible flight routes are started to generate. Once a disruption occurred, partial feasible flight lines get ready to obtain a solution of formulation (2). So computing time of the formulation (1) can be ignored in the total computing time of calculating these airline disruption problems. Only the computing time of the formulation (2) is counted. Therefore, for the computational time, the duration of this method in Table 1 and Table 2 is the computing time of the formulation (2), and it is much less than the duration of Liu’s method which can obtain the solutions in minutes. It is very important for practical application. If the dimension of the problem adds and the number of slave computers increases, the advantages are more obvious.

Since Dang and Ye’s algorithm is the improvement of Dang’s iterative method, a comparison between the two method when they are used to solve the problem (1) is given in Table 3. Partial feasible flight lines obtained using the two approaches are all the same. However, there are differences in the cost of the computing time and the number of linear programming. The number of the linear programming which are used in Dang and Ye’s approach to find a feasible flight line is about 1/6 of that used in Dang’s approach. The computing time of Dang and Ye’s approach to obtain a feasible flight line is about 1/4 of the computing time of Dang’s approach.

The initial points of Dang and Ye’s algorithm for generation is the starting point of flight lines. So the feasible flight lines in the lexicographical order could be got sequentially by Dang and Ye’s algorithm. Because the variables $x_i$’s in the formulation (1) stand for the legs of flight that are rescheduled to an increasing order of leaving time, the solutions found from the preliminary flight line earlier are not more deflected from the preliminary flight line than the ones found later. Besides, there is no feasible flight line again among any two feasible flight lines found continuously in the lexicographical order.
TABLE 2. Comparison between Dang and Ye’s algorithm with Liu’s method for DH-8 fleet.

<table>
<thead>
<tr>
<th>Method</th>
<th>Cluster size</th>
<th>Feasible flight routes size</th>
<th>Total delay times/min</th>
<th>Delay over 30 minutes (Flights)</th>
<th>Delay flight</th>
<th>Flight swaps (Flights)</th>
<th>Duration/ms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dang and Ye</td>
<td>10</td>
<td>56</td>
<td>640</td>
<td>9</td>
<td>23</td>
<td>5</td>
<td>32</td>
</tr>
<tr>
<td></td>
<td>18000</td>
<td>17121</td>
<td>650</td>
<td>9</td>
<td>23</td>
<td>5</td>
<td>7023</td>
</tr>
<tr>
<td>Liu</td>
<td>100 pop. 50000 gen.</td>
<td>640</td>
<td>9</td>
<td>22</td>
<td>4</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

TABLE 3. Comparison between Dang and Dang-Ye.

<table>
<thead>
<tr>
<th>Method</th>
<th>Number of linear programming per solution</th>
<th>Duration per solution /ms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dang</td>
<td>96.73</td>
<td>139.74</td>
</tr>
<tr>
<td>Dang-Ye</td>
<td>16.33</td>
<td>32.39</td>
</tr>
</tbody>
</table>

Changes of new feasible flight lines are alterations of the legs of flight in the last part of the primary flight lines at the first of computing by Dang and Ye’s algorithm. The variations of legs of flight spread from the last part of the primary flight lines to the front part of flight lines with computing goes ahead. Thus, the feasible flight lines found later are more deflected from the preliminary flight lines. From comparisons one can see that most of the solutions which have less total delay time of the formulation (2) are obtained through the feasible flight lines which are around each preliminary flight line in the lexicographical order. Solutions which have much less total delay time could be found through the feasible flight lines which are relatively far from each preliminary flight line in the lexicographical order. However, these solutions are more deflected from the original airplane schedule. With a properly cluster size, a solution of the formulation (2) can be obtained which is very close to the optimum and less deflected from the preliminary flight lines by the partial feasible flight lines which are got by Dang and Ye’s algorithm combined with the initial seeds cluster division. If it is necessary, a set of solutions, which have different total delay time and other properties, could be obtained through the partial feasible flight lines obtained using Dang and Ye’s algorithm in various cluster size for the users to make decision.

VI. CONCLUSIONS
In this paper, it has developed a procedure for calculating airline disruption problem. In this procedure, the problem has been divided into two subproblems: one is solving feasible flight lines generation subproblem and the other is airplanes reschedule subproblem. The main focus of this paper is the feasible flight lines generation subproblem. If the dimension of airline disruption problem is large, in a reasonable time it is impossible to calculate all the feasible flight lines. Therefore, the solution space of this subproblem has been divided into a certain number of segments which can be calculated by distributed Dang and Ye’s algorithm simultaneously and independently. From the computing experiments of two flight schedules given, it is shown that Dang and Ye’s algorithm obtains much fewer partial feasible flight lines for formulation (2) to get a better solution than CPLEX CP Optimizer. Besides, solutions which have less total delay time than those solved by Liu’s method could be shown by relatively more feasible flight lines provided. The solution of the formulation (2) is more and more close to the optimum of the formulation (2) if the cluster size is increasing. However, the flight lines of the solution are more deflected from the preliminary flight lines. In a reasonable time, for large dimension airline disruption problems which are impossible to calculate all the feasible solutions, a properly chosen cluster size could obtain a solution of the formulation (2) which is at least very close to the optimum and less deflected from the preliminary flight lines by partial feasible flight lines which are got by Dang and Ye’ method combined with the initial seeds cluster division. The numerical results have shown that this distributed method proposed is promising.

In the future work, disruptions caused by both airport closures and airplane groundings will be considered together. One can believe that Dang and Ye’s algorithm will have a good performance in practice.

COMPETING INTERESTS
The authors declare that they have no competing interests.

AUTHOR’S CONTRIBUTIONS
All authors read and approved the final manuscript.

ACKNOWLEDGEMENTS
The authors are very grateful to the editor and reviewers for their valuable comments and suggestions. Zhengtian Wu and Qing Gao contributed equally to this work.

REFERENCES


QING GAO received the B.Sc. degree from the University of Science and Technology of China, China, in 2008, and double Ph.D. degrees from the University of Science and Technology of China and the City University of Hong Kong in 2014. He currently holds a post-doctoral position with the Institute for Automatic Control and Complex Systems (AKS), Faculty of Engineering, University of Duisburg-Essen, Duisburg, Germany. He has demonstrated outstanding research track record in the area of system engineering. In the last five years as a System Engineering Researcher in various roles, he has had over 18 journal articles published in international high quality journals, such as the IEEE Transactions and Automatica, which all have high impact factors and are highly ranked in the area of control theory and applications. So far his publications have been cited by over 400 times by scholars from over ten countries, such as the U.K., the USA, China, Germany, and France, and his h-index on Google scholar is 11.

BENCHI LI received the B.Sc. degree in mechanical manufacturing and automation from the University of Science and Technology of China, China, in 2007, and double Ph.D. degrees in operations research from the University of Science and Technology of China and City University of Hong Kong in 2013. He is currently a Researcher with China National Petroleum, China. His research interests include approximation algorithm and distributed computation.

CHUANGYIN DANG received the B.Sc. degree in computational mathematics from Shanxi University, China, in 1983, the M.Sc. degree in applied mathematics from XiDian University, China, in 1986, and the Ph.D. degree (cum laude) in operations research/economics from Tilburg University, The Netherlands, in 1991. He is currently a Professor with the Department of Systems Engineering and Engineering Management, City University of Hong Kong, China. He has published over 100 papers in journals and his research interests include computational optimization, logistics, supply chain management, and modeling in economics.

FUYUAN HU is pursuing the Ph.D. degree with Northwestern Polytechnical University, and is a Visiting Ph.D. Student with the City University of Hong Kong. He was a Post-Doctoral Researcher with Vrije Universiteit Brussel, Belgium. He is currently an Associate Professor in computer vision and machine learning with the Suzhou University of Science and Technology. His research interests include graphical models, structured learning, and tracking.

* * *