Transportation Research Part B 53 (2013) 1–16

Contents lists available at SciVerse ScienceDirect

Transportation Research Part B

journal homepage: www.elsevier.com/locate/trb

# Total unimodularity and decomposition method for large-scale air traffic cell transmission model

P. Wei *, Y. Cao, D. Sun

*School of Aeronautics and Astronautics, Purdue University, West Lafayette, IN 47907, USA*

ARTICLE INFO

*Article history:*
Received 27 February 2012
Received in revised form 16 March 2013
Accepted 18 March 2013

*Keywords:*
Air traffic flow management
Cell transmission model
Total unimodularity
Large-scale transportation network

ABSTRACT

In an earlier work, Sun and Bayen built a Large-Capacity Cell Transmission Model for air traffic flow management. They formulated an integer programming problem of minimizing the total travel time of flights in the National Airspace System of the United States subject to sector capacity constraints. The integer program was relaxed to a linear program for computational efficiency. In this paper the authors formulate the optimization problem in a standard linear programming form. We analyze the total unimodular property of the constraint matrix, and prove that the linear programming relaxation generates an optimal integral solution for the original integer program. It is guaranteed to be optimal and integral if solved by a simplex related method. In order to speed up the computation, we apply the Dantzig–Wolfe Decomposition algorithm, which is shown to preserve the total unimodularity of the constraint matrix. Finally, we evaluate the performances of Sun and Bayen's relaxation solved by the interior point method and our decomposition algorithm with large-scale air traffic data.



## 1. Introduction

The *National Airspace System* (NAS) in the United States is a large-scale, nonlinear dynamic system. The airspace is divided into 22 *Air Route Traffic Control Centers* (ARTCCs, or simply, Centers). Each Center is sub-divided into smaller regions, called *Sectors*, with at least one air traffic controller responsible for each of them (Nolan, 2003).

The last few decades have witnessed the tremendous growth of air traffic. Since the function of air traffic controllers is to maintain safe separation between aircraft while guiding them to destinations, an imbalance between the growth of air traffic and the limited airspace capacity arises. So the design of advanced air traffic management schemes is desired to help.

Optimization techniques have been developed to facilitate *Traffic Flow Management* (TFM). Current popular TFM schemes mainly focus on ground delay and/or rerouting flights to accommodate capacitated elements, e.g., en route sectors and airports (Lulli and Odoni, 2007). TFM studies focusing on optimal ground delays have been conducted by many researchers, from both deterministic and probabilistic perspectives (see Odoni (1987), Terrab and Odoni (1993), Gilbo (1993), Vranas et al. (1994), Andreatta et al. (1997), Navazio and Romanin-Jacur (1998), Bertsimas and Patterson (1998), Hoffman and Ball (2000), Dell'Olmo and Lulli (2003), Vossen et al. (2003), Ball and Lulli (2004), Ball et al. (2005), Vossen and Ball (2005, 2006)). Odoni (1987) formulated the TFM problem using a large number of models and algorithms to detect optimal strategies to assign ground delays to flights (see in particular Bertsimas et al. (2008)). Helme (1992) was among the first to include en route capacity restrictions in the TFM problem, which is intuitive to understand but has weak computational performance as was discussed by Bertsimas et al. (2008). Lindsay et al. (1993) formulated a disaggregate deterministic 0–1 integer

* Corresponding author.
*E-mail addresses:* weip@purdue.edu (P. Wei), cao20@purdue.edu (Y. Cao), dsun@purdue.edu (D. Sun).

0191-2615/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.trb.2013.03.004

programming model for deciding ground and airborne holding of individual flights in the presence of both airport and airspace capacity constraints. A deterministic, open-loop integer programming method was formulated to assign departure time and sector occupancy time of each aircraft in the work by Bertsimas and Patterson (1998), but the computational complexity of this model has limited its use to a small number of real-world examples as was shown by Grabbe et al. (2007). To improve the runtime, a method that reduces the number of flights to be optimized was proposed by Rios and Ross (2008); and more recently, a Dantzig–Wolfe Decomposition method was implemented for the Bertsimas and Patterson model (Rios and Ross, 2010), which actually motivated several studies (including the work in this paper) using decomposition methods to solve large-scale TFM problems. The work in Bertsimas and Patterson (1998) was extended to provide a complete representation of all the phases of each flights including rerouting strategies (Bertsimas et al., 2008). In Sherali et al. (2002), a binary integer programming was proposed for a TFM problem, which considers controller workload, airspace safety, and equity among airlines. Subsequently, the binary integer programming was extended to incorporate rerouting in Sherali et al. (2003, 2006). Research that considers equity or market-based traffic management using aggregate models has been conducted by Bloem and Sridhar (2008), Waslander et al. (2008a,b). Sridhar et al. (2002) proposed an integrated three-step hierarchical method for developing deterministic TFM plans consisting of national-level playbook reroutes, miles-in-trail restrictions, and tactical reroutes to alleviate sector-level congestion. Subsequently, Kopardekar and Green (2005) used a deterministic, Center-based system to manually identify congested sectors and compare the trade-offs of implementing altitude capping, local rerouting, departure delays, and time-based metering or miles-in-trail restrictions. Wanke and Greenbaum (2007) proposed a Monte Carlo-based incremental, probabilistic decision making approach for developing en route traffic management controls. More recently, Grabbe et al. (2009) applied a sequential optimization method to manage air traffic flow under the uncertainties in airspace capacity and demand.

Sun and Bayen (2008) presented a traffic flow model called the *Large-Capacity Cell Transmission Model*, in short CTM(L), which is a variation of the air traffic cell transmission model in Menon et al. (2002) and the original cell transmission model in Daganzo (1994, 1995). Sun and Bayen applied it to a problem of minimizing the total travel time of all flights in the NAS of the United States restricted by sector capacity counts, which is an integer program containing billions of variables and constraints. It was then relaxed to a *linear program* (LP) for computational efficiency. Sun et al. (2011) applied the dual decomposition method to solve the large scale linear program in a computationally tractable manner. However, the authors in Sun and Bayen (2008) and Sun et al. (2011) found that solving the linear program by large-scale commercial software with or without decomposition method can possibly result in the fractional optimal solution, which cannot be implemented as en route holding control in practice. Integer solutions should be guaranteed while the optimum is obtained efficiently. This is the major motivation of our work.

In this paper we study the solution space structure of the problem and prove that there exists an optimal integral solution in the linear programming relaxation, which is also the optimal for the original integer program. The solution is guaranteed to be integral when solved by simplex related methods. Therefore we propose the simplex based Dantzig–Wolfe Decomposition to ensure the integral optimum, while achieving a fast computation speed.

The rest of this paper is organized as follows. The second section introduces the CTM(L) model. The third section formulates the integer programming problem in a standard linear programming form and analyzes its total unimodularity. The fourth section explains why the interior point method applied in Sun and Bayen (2008) results in the fractional optimal solution. In Section 5 we apply the Dantzig–Wolfe Decomposition algorithm. Large-scale simulations are performed with historical data. Section 6 concludes the paper.

## 2. CTM(L) and its mathematical formulation

The CTM(L) is based on a network flow model built from the historical *Aircraft Situation Display to Industry* (ASDI) and *Enhanced Traffic Management System* (ETMS) data (Bayen et al., 2006).

### 2.1. Construction of the network

The network flow model is composed of nodes and links. The *nodes* are created as the entry and exit points at the sector boundaries as shown in Fig. 1. For any sectors $s_1$, $s_2$ and $s_3$, if $s_1$ and $s_2$ share a boundary and if $s_2$ and $s_3$ are neighbors, two directed *links* are created: one from node $v_{\{s_1,s_2\}}$ to node $v_{\{s_2,s_3\}}$ and one from node $v_{\{s_3,s_2\}}$ to node $v_{\{s_2,s_1\}}$.

The expected travel time of a flight through a link is computed from ASDI/ETMS data, which is used to determine the length of the link. Each link is divided into several *cells* as time interval units (see Fig. 1). A *path* is defined as a complete flight route connecting one departure airport and one arrival airport, which usually consists of multiple links. Further details on constructing the CTM(L) network are described in Sun and Bayen (2008).

### 2.2. Dynamics

The CTM(L) model is reduced to a linear time-invariant dynamical system. The air traffic flow on link $i$ can be depicted as the *Link Level Model* (Sun and Bayen, 2008):

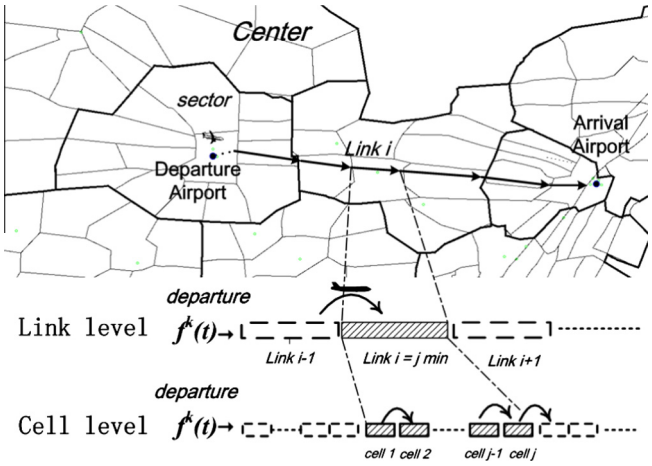


**Fig. 1.** Construct the CTM(L) network.

$$\begin{aligned} x_i(t+1) &= A_i x_i(t) + B_1^i u_i(t) + B_2^i f_i(t), \\ y(t) &= \widetilde{C}_i' x_i(t), \end{aligned} \tag{1}$$

where $x_i(t) = \left[x_i^1(t), \ldots, x_i^{N_i}(t)\right]$ is the state vector whose elements represent the corresponding aircraft counts in each cell of link $i$ at time $t$, and $N_i$ is the number of cells along link $i$. The forcing scalar input $f_i(t)$ denotes the entry count into link $i$ during time $t$, and the vector $u_i(t)$ represents airborne holding control. The output $y(t)$ is the aircraft count in a user-specified set of cells at time $t$. $\widetilde{C}_i$ is the user-specified index vector. $A_i$ is an $N_i \times N_i$ nilpotent matrix with 1's on its super-diagonal. $B_2^i = [1; 0; \ldots; 0]$ is the forcing vector with $N_i$ elements, and $B_1^i$ is the $N_i \times N_i$ holding pattern matrix, in which all the non-zero elements are 1's on the diagonal and $-1$'s on the super-diagonal.

Based on the Link Level Model, it is easy to extend it to the *Path Level Model* and build a *Sector Level Model* by integrating all the paths in a sector, e.g. the matrices $A_i$ of different paths in Eq. (1) are put in diagonal blocked matrix $A$ and the vectors $x_i$ are cascaded as $x$. The NAS-wide model can also be cast in the same procedure.

## 3. Problem formulation and totally unimodular property

### 3.1. Integer programming formulation

#### 3.1.1. Path Level Model

According to Sun and Bayen (2008), for a single path with $N$ cells, the initial condition of the model is

$$x_k(0) = x_k^0, \quad k = 0, 1, \ldots, N-1, \tag{2}$$

the boundary conditions are

$$\begin{aligned} x_0(0) &= f(0) + x_0^0, \\ x_0(t) &= f(t) + u_0(t-1), \quad t = 1, 2, \ldots, T-1, \end{aligned} \tag{3}$$

and the dynamics are

$$\begin{aligned} &x_k(t) = x_{k-1}(t-1) - u_{k-1}(t-1) + u_k(t-1), \\ &k = 1, 2, \ldots, N-1, \quad t = 1, 2, \ldots, T-1, \end{aligned} \tag{4}$$

where $T$ is the planning horizon.

We cascade all the $x_k(t)$ into a vector $x$ in the sequence as below

$$x = [x_0(0), \ldots, x_{N-1}(0), x_0(1), \ldots, x_{N-1}(1), \ldots, x_0(T-1), \ldots, x_{N-1}(T-1)]'. \tag{5}$$

Similarly, vector $u$ is created of the same length $NT$ as $x$:

$$u = [u_0(0), \ldots, u_{N-1}(T-1)]'. \tag{6}$$

We integrate the initial states (2) and boundary states (3) into a vector $f$ of the length $NT$:

$$f = [f(0) + x_0(0), x_1(0), \ldots, x_{N-1}(0), f(1), 0, \ldots, 0, \ldots, f(T-1), 0, \ldots, 0]'. \tag{7}$$

Finally, an equality form is generated by combining Eqs. (5)–(7):

$$x = Px + Qu + f, \tag{8}$$

where matrices $P$ and $Q$ are both of dimension $NT \times NT$.

The matrix $P$ is

$$P_{(NT\times NT)} = \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ P_o & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & P_o & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & P_o & \mathbf{0} \end{pmatrix}, \tag{9}$$

where $\mathbf{0}$ and $P_o$ are both $N \times N$ matrices. The matrix $P_o$ is

$$P_{o(N\times N)} = \begin{pmatrix} 0 & 0 & 0 & \dots & 0 & 0 \\ 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & \dots & 1 & 0 \end{pmatrix}. \tag{10}$$

The matrix $Q$ is

$$Q_{(NT\times NT)} = \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ Q_o & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & Q_o & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & Q_o & \mathbf{0} \end{pmatrix}, \tag{11}$$

where $\mathbf{0}$ and $Q_o$ are both $N \times N$ matrices. The matrix $Q_o$ is

$$Q_{o(N\times N)} = \begin{pmatrix} 1 & 0 & 0 & \dots & 0 & 0 \\ -1 & 1 & 0 & \dots & 0 & 0 \\ 0 & -1 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 1 & 0 \\ 0 & 0 & 0 & \dots & -1 & 1 \end{pmatrix}. \tag{12}$$

Considering both states $x$ and holding controls $u$ are unknown variables, we transform Eq. (8) to

$$[I - P, -Q]\begin{bmatrix} x \\ u \end{bmatrix} = f, \tag{13}$$

where $[x; u]$ is a vector of variables. Eq. (13) is the dynamic constraint of the model. Restricted by practical physics rules, the problem has the other three constraints as follows.

*Hold Constraint*: in each cell $k$ at time $t$, the number of aircraft to be held is fewer than the current aircraft counts:

$$u \leqslant x. \tag{14}$$

Eq. (14) is incorporated into Eq. (13) in an inequality form as the *Dynamics Constraint* for a single path:

$$\begin{bmatrix} I - P & -Q \\ P - I & Q \\ -I & I \end{bmatrix}\begin{bmatrix} x \\ u \end{bmatrix} \leqslant \begin{bmatrix} f \\ -f \\ \mathbf{0} \end{bmatrix}. \tag{15}$$

*Non-negative Constraint*: $x$ and $u$ should be non-negative:

$$x \geqslant 0, \quad u \geqslant 0. \tag{16}$$

*Integral Constraint*: $x$ and $u$ should be integer vectors:

$$x, u \in \mathbb{I}^{NT}, \tag{17}$$

where $\mathbb{I}^{NT}$ is the integer vector domain of dimension $NT$.

### 3.1.2. Decoupled Sector Level Model

For the dynamics constraint in Eq. (15) of each path $i$ with $N_i$ cells, we denote the $3N_iT \times 2N_iT$ matrix as $A_i$, the $2N_iT$ vector consisting of $x$ and $u$ as $x_i$, and the $3N_iT$ vector on right-hand side as $f_i$. Thus we obtain the decoupled all-path dynamics constraints as

$$\begin{bmatrix} A_1 & & & \\ & A_2 & & \\ & & \ddots & \\ & & & A_M \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_M \end{bmatrix} \leqslant \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_M \end{bmatrix}, \tag{18}$$

where the matrix size is $\left(3\sum_{i=1}^{M} N_iT\right) \times \left(2\sum_{i=1}^{M} N_iT\right)$, vector $x$ has a length of $2\sum_{i=1}^{M} N_iT$, and the length of $f$ is $3\sum_{i=1}^{M} N_iT$. Eq. (18) describes the internal dynamics constraints for all paths. All paths are decoupled.

### 3.1.3. Coupled network level model

In the real air traffic network, multiple paths usually pass through one certain sector with a capacity constraint. To be more precise, air traffic controllers even set different sector capacity constraints to one sector at different time periods. The *Sector Count Constraint* is given by

$$M \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_M \end{bmatrix} \leqslant \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_S \end{bmatrix}, \tag{19}$$

where

$$v_j = [v_j(0); v_j(1); \ldots; v_j(T-1)]. \tag{20}$$

$v_j(t)$ is the sector capacity for the $j$th sector at time period $t$. $v = [v_1; v_2; \ldots; v_S]$ is $TS \times 1$ and $x$ has a length of $2\sum_{i=1}^{M} N_iT$. The matrix $M$ has a dimension of $TS \times 2\sum_{i=1}^{M} N_iT$, mapping aircraft counts from paths to sectors.

$M$ consists of blocks like $M_{ji}$ mapping aircraft counts from path $i$ to sector $j$, explained as follows.

$$M = \begin{bmatrix} M_{11} & M_{12} & \ldots & M_{1M} \\ M_{21} & M_{22} & \ldots & M_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ M_{S1} & M_{S2} & \ldots & M_{SM} \end{bmatrix}, \tag{21}$$

where the structure of block $M_{ji}$ is

$$M_{ji} = \left[\begin{array}{cccc|c} s'_{ji} & \mathbf{0}'_\alpha & \ldots & \mathbf{0}'_\alpha & \\ \mathbf{0}'_\alpha & s'_{ji} & \ldots & \mathbf{0}'_\alpha & \\ \vdots & \vdots & \ddots & \vdots & \mathbf{0}_\beta \\ \mathbf{0}'_\alpha & \mathbf{0}'_\alpha & \ldots & s'_{ji} & \end{array}\right] := \left[M^l_{ji} \middle| \mathbf{0}_\beta\right]. \tag{22}$$

Inside $M_{ji}$, for each path $i$ with $N_i$ cells, we use a vector $s_{ji}$ of length $N_i$ to denote which cells on path $i$ lie in sector $j$:

$$s_{ji}(k) = \begin{cases} 1, & \text{if the } k\text{th cell of path } i \text{ is in sector } j; \\ 0, & \text{otherwise.} \end{cases} \tag{23}$$

In this paper, the sector boundaries define all the sectors as convex polygons. As a result, when a path intersects a sector, there is only one segment lying inside this sector as shown in Fig. 2a. However, in practice the sectors are not necessarily convex, so there may be two or even more segments from a path falling inside one sector as shown in Fig. 2b. For example, the vector $s'_{ji}$ for Fig. 2a should be [0,0,1,1,1,1,1,1,1,1,1,1,0], which contains only one consecutive 1's series and the vector $s'_{ji}$ for Fig. 2b is [0,0,1,1,1,0,0,1,1,0], which contains two consecutive 1's series.

In Eq. (22) $\mathbf{0}'_\alpha$ is a zero row vector and $\mathbf{0}_\beta$ is a $T \times N_iT$ all-zero matrix. The matrix $M^l_{ji}$ to the left of $\mathbf{0}_\beta$ is also $T \times N_iT$. The diagonal blocks consist of the same row vector $s'_{ji}$ because the relationship that cell $k$ in path $i$ belongs to sector $j$ does not change with time.

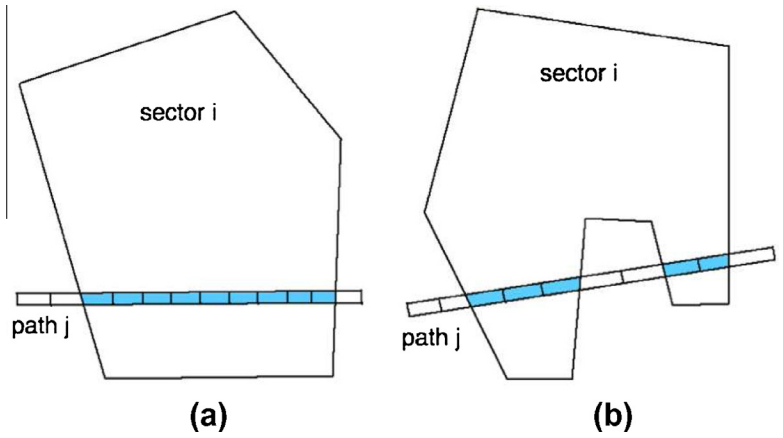


**Fig. 2.** (a) Path $j$ intersects with the convex sector $i$ and (b) Path $j$ intersects with the non-convex sector $i$.

We define $M_i^l$ as follow:

$$M_i^l = \begin{bmatrix} M_{1i}^l \\ M_{2i}^l \\ \vdots \\ M_{Si}^l \end{bmatrix}. \tag{24}$$

Another feature inside matrix $M$ is that the sum of each column in matrix $M_i^l$ is 1, because at each time step $t$, the $k$th cell of path $i$ can only belong to one sector.

*3.1.4. Integer programming formulation*

We formulate the optimization problem as follows. First we denote

$$A = \begin{bmatrix} A_1 & & & \\ & A_2 & & \\ & & \ddots & \\ & & & A_M \\ M_{11} & M_{12} & \dots & M_{1M} \\ M_{21} & M_{22} & \dots & M_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ M_{S1} & M_{S2} & \dots & M_{SM} \end{bmatrix}, \quad b = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_M \\ v_1 \\ v_2 \\ \vdots \\ v_S \end{bmatrix}, \tag{25}$$

and

$$c = [c_1; \mathbf{0}_1; c_2; \mathbf{0}_2; \dots; c_M; \mathbf{0}_M], \tag{26}$$

where $c_i$ is all-one and $\mathbf{0}_i$ is all-zero. $c$ and $x$ are both vectors of length $2\sum_{i=1}^{M} N_i T$.

From (16) and (17), we know that $x$ is required to be non-negative and integral. According to the physics rules of 2, 3 and 20, vector $b$ is also integral. $b \in \mathbb{I}^{\left(3\sum_{i=1}^{M} N_i + S\right)}$.

In summary, the original problem is formulated as an integer program as follow:

$$\begin{aligned} \min \quad & c'x, \\ \text{s.t.} \quad & Ax \leqslant b, \\ & x \geqslant 0, \text{ and } x \in \mathbb{I}^{2T\sum_{i=1}^{M} N_i}. \end{aligned} \tag{27}$$

*3.2. Standard linear programming form and total unimodularity*

Solving the integer program in formulation (27) is extremely time consuming and sometimes impossible for a large-scale problem in air traffic management. Sun and Bayen (2008) relaxed (27) to a linear program to achieve better computational efficiency. In this paper it is written in the *Standard Linear Programming Form* (Chvatal, 1983):

$$\begin{aligned} \min \quad & c'x, \\ \text{s.t.} \quad & Ax \leqslant b, \\ & x \geqslant 0. \end{aligned} \tag{28}$$

However, in general the linear relaxation (28) results in fractional solutions (Sun and Bayen, 2008). In order to maintain the computational efficiency and obtain the integral solution, the total unimodularity of the linear relaxation (28) is studied in this work.

*3.2.1. Total unimodularity and integral optimum*

**Theorem 1.** *If A is totally unimodular and the problem (28) is feasible, there exists at least one integral optimum for formulation (28), which can be found by simplex method.*

**Proof.** The proof contains two parts. First, according to the Hoffman and Kruskal's theorem described by Schrijver (1998), if $A$ in formulation (28) is *totally unimodular* with the fact that vector $b$ is integral, the corner points (extreme points) of the feasible polyhedron $\{x|Ax \leqslant b, x \geqslant 0\}$ defined in (28) are integral. Second, recall that the simplex method generates the optimal solution by pivoting from one extreme point to another adjacent extreme point around the feasible polyhedron. The simplex method must provide an integral optimal solution for formulation (28) when $A$ is totally unimodular. □

It is evident that when the optimal solution to the relaxed linear program (28) is integral, this solution is also an optimal solution to the integer program (27). So the key point is to prove $A$ is totally unimodular.

**Lemma 1.** *If matrix A is full row (column) rank, the total unimodularity of A is preserved under the three elementary row (column) operations listed in* Table 1.

Lemma 1 is obtained by combining both Theorem 19.5 and (43)(ii) in Schrijver (1998).

*3.2.2. Total unimodularity of matrix A*

**Theorem 2.** *The matrix A in (28) is totally unimodular.*

**Proof.** Since there is no sufficient condition or lemma which can directly prove a matrix is total unimodular, the elementary row and column operations are used to transform the original matrix $A$ to a recognized or proved total unimodular format.

We start with performing elementary column operations inside each blocked column of matrix $A$ as shown in (25). For simlicity of illustration, the $M$th blocked column is shown as example and note that there are $(M-1)$ extra zero-blocks on top of this column omitted in the following example (29):

$$\begin{bmatrix} A_M \\ M_{1M} \\ M_{2M} \\ \vdots \\ M_{SM} \end{bmatrix} = \begin{bmatrix} I-P & -Q \\ P-I & Q \\ -I & I \\ M^l_{1M} & \mathbf{0}_\beta \\ M^l_{2M} & \mathbf{0}_\beta \\ \vdots & \vdots \\ M^l_{SM} & \mathbf{0}_\beta \end{bmatrix}, \tag{29}$$

where the form of $A_M$ and $M_{jM}$ can be found in (15) and (22) respectively.

Through a series of elementary column operations, we have transformed the upper part $A_M$ into several smaller blocks $I$, $-I$ and $\mathbf{0}$. Eq. (29) is changed into the following format (the detailed derivation from (29) and (30) can be found in the appendix):

$$\begin{bmatrix} I & \mathbf{0} \\ -I & \mathbf{0} \\ \mathbf{0} & I \\ L_{1M} & R_{1M} \\ L_{2M} & R_{2M} \\ \vdots & \vdots \\ L_{SM} & R_{SM} \end{bmatrix}. \tag{30}$$

**Table 1**
Elementary row (column) operations.

| |
|---|
| 1. Exchanging two rows (columns) |
| 2. Multiplying a row (column) by $-1$ |
| 3. Adding a row (column) to another row (column) |

From the first three blocked rows above the horizontal line in (30) we know that this matrix is full column rank.

Similar elementary column operations can be performed in other $(M-1)$ blocked columns of matrix $A$ in (25) and the resulted matrix is shown in (31):

$$\begin{bmatrix} I & \mathbf{0} & & & & & \\ -I & \mathbf{0} & & & & & \\ \mathbf{0} & I & & & & & \\ & & I & \mathbf{0} & & & \\ & & -I & \mathbf{0} & & & \\ & & \mathbf{0} & I & & & \\ & & & & \ddots & & \\ & & & & & I & \mathbf{0} \\ & & & & & -I & \mathbf{0} \\ & & & & & \mathbf{0} & I \\ L_{11} & R_{11} & L_{12} & R_{12} & \dots & L_{1M} & R_{1M} \\ L_{21} & R_{21} & L_{22} & R_{22} & \dots & L_{2M} & R_{2M} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ L_{S1} & R_{S1} & L_{S2} & R_{S2} & \dots & L_{SM} & R_{SM} \end{bmatrix}. \tag{31}$$

The upper part of (31) tells that the matrix (31) after the elementary column operations is full column rank. The elementary column operations do not change the column rank of a matrix, so the matrix $A$ before these operations is also full column rank. Thus according to Lemma 1, the elementary column operations we have performed can preserve the total unimodularity of matrix $A$. The problem becomes to prove (31) is totally unimodular.

Moreover, based on (43)(v) of Schrijver (1998), if the lower part of (31) is totally unimodular, then the whole matrix (31) is also totally unimodular. Now we only need to show that (32) is totally unimodular.

$$\begin{bmatrix} L_{11} & R_{11} & L_{12} & R_{12} & \dots & L_{1M} & R_{1M} \\ L_{21} & R_{21} & L_{22} & R_{22} & \dots & L_{2M} & R_{2M} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ L_{S1} & R_{S1} & L_{S2} & R_{S2} & \dots & L_{SM} & R_{SM} \end{bmatrix}, \tag{32}$$

where $L_{ji}$ is a lower triangle blocked matrix with every non-zero block as $s'_{ji}$ and the non-zero block $t'_{ji}$ fills the lower triangle positions of matrix $R_{ji}$ below the main diagonal as in (33) and (34).

$$L_{ji} = \begin{bmatrix} s'_{ji} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ s'_{ji} & s'_{ji} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ s'_{ji} & s'_{ji} & s'_{ji} & \dots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ s'_{ji} & s'_{ji} & s'_{ji} & \dots & s'_{ji} & \mathbf{0} \\ s'_{ji} & s'_{ji} & s'_{ji} & \dots & s'_{ji} & s'_{ji} \end{bmatrix}, \tag{33}$$

$$R_{ji} = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ t'_{ji} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ t'_{ji} & t'_{ji} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ t'_{ji} & t'_{ji} & t'_{ji} & \dots & \mathbf{0} & \mathbf{0} \\ t'_{ji} & t'_{ji} & t'_{ji} & \dots & t'_{ji} & \mathbf{0} \end{bmatrix}. \tag{34}$$

In (32) we assume that every sector contains at least one cell from a path. If there is a sector containing no cells, the corresponding blocked row can be deleted according to (43)(v) in Schrijver (1998). In that case we actually do not need to include this sector in our model.

We perform elementary row operations to (32) and get (35):

$$\begin{bmatrix} \widetilde{L}_{11} & \widetilde{R}_{11} & \widetilde{L}_{12} & \widetilde{R}_{12} & \dots & \widetilde{L}_{1M} & \widetilde{R}_{1M} \\ \widetilde{L}_{21} & \widetilde{R}_{21} & \widetilde{L}_{22} & \widetilde{R}_{22} & \dots & \widetilde{L}_{2M} & \widetilde{R}_{2M} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \widetilde{L}_{S1} & \widetilde{R}_{S1} & \widetilde{L}_{S2} & \widetilde{R}_{S2} & \dots & \widetilde{L}_{SM} & \widetilde{R}_{SM} \end{bmatrix}, \tag{35}$$

in which matrices $\widetilde{L}_{ji}$ and $\widetilde{R}_{ji}$ are:

$$\widetilde{L}_{ji} = \begin{bmatrix} s'_{ji} & & & & & & \\ & s'_{ji} & & & & & \\ & & s'_{ji} & & & & \\ & & & \ddots & & & \\ & & & & s'_{ji} & & \\ & & & & & s'_{ji} & \end{bmatrix}, \tag{36}$$

$$\widetilde{R}_{ji} = \begin{bmatrix} \mathbf{0} & & & & & & \\ t'_{ji} & \mathbf{0} & & & & & \\ & t'_{ji} & \mathbf{0} & & & & \\ & & \ddots & \ddots & & & \\ & & & t'_{ji} & \mathbf{0} & & \\ & & & & t'_{ji} & \mathbf{0} & \end{bmatrix}. \tag{37}$$

As we have assumed previously there must be at least one 1 appearing at certain cell of $s'_{ji}$ in every row of a certain $\widetilde{L}_{ji}$ of each blocked row. If $\widetilde{L}_i$ is defined as:

$$\widetilde{L}_i = \begin{bmatrix} \widetilde{L}_{1i} \\ \widetilde{L}_{2i} \\ \vdots \\ \widetilde{L}_{Si} \end{bmatrix}, \tag{38}$$

the sum of each column in $\widetilde{L}_i$ is 1, which means that the 1 appearing at a certain cell of $s'_{ji}$ cannot show up in another row, in other words, every row is independent to each other. Since (35) is full row rank, the elementary row operations we have performed also preserve the total unimodularity. Our next concern is whether (35) is totally unimodular.

Since the column sum of every $\widetilde{L}_i$ is 1, according to (43)v in Schrijver (1998), the problem is equivalent to proving (39) is totally unimodular.

$$\begin{bmatrix} \widetilde{R}_{11} & \widetilde{R}_{12} & \dots & \widetilde{R}_{1M} \\ \widetilde{R}_{21} & \widetilde{R}_{22} & \dots & \widetilde{R}_{2M} \\ \vdots & \vdots & \vdots & \vdots \\ \widetilde{R}_{S1} & \widetilde{R}_{S2} & \dots & \widetilde{R}_{SM} \end{bmatrix}. \tag{39}$$

Appendix A shows that $t'_{ji} = s'_{ji}Q_o$. Based on the definition of $Q_o$ in Eq. (12), we know that $t'_{ji}$ in (34) and (37) records what kind of 0–1 changes in every cell in the corresponding $s'_{ji}$. For example, for $s'_{ji} = [0, 0, 1, 1, 1, 0]$, the corresponding $t'_{ji} = [0, -1, 0, 0, 1, 0]$. The $-1$ at the 2nd cell of $t'_{ji}$ represents a change from 0 to 1 between the 2nd cell and the 3rd cell in $s'_{ji}$. The 1 at the 5th cell of $t'_{ji}$ represents a change from 1 to 0 between the 5th cell and the 6th cell in $s'_{ji}$.

Consider the $i$th blocked column of matrix in (39), it describes all the 0–1 changing information of the $N_i$ cells of the $i$th path in all $S$ sectors. When there is a $-1$ at a certain cell of a certain $t'_{ji}$ in a certain row, there must be one and only one 1 at the same cell in a different row. Recall Eqs. (21)–(23). When we find a $s'_{ji} = [\dots 1, 1, 1, 1, 0, 0, 0 \dots]$, there must exist a $s'_{ki} = [\dots 0, 0, 0, 0, 1, 1, 1 \dots]$ in the same column in Eq. (21) because that is where path $i$ is crossing sector $j$ and sector $k$. Furthermore, there will be a corresponding $t'_{ji} = [\dots 0, 0, 0, 1, 0, 0, 0 \dots]$ and a $t'_{ki} = [\dots 0, 0, 0, -1, 0, 0, 0 \dots]$ in Eq. (39). In other words, 1 and $-1$ must show up in pair in each column. Based on (43)v in Schrijver (1998), the modified matrix (39) only contains the columns having exactly one 1/$-1$ pair after deleting all-zero columns in (39), where no 0–1 change

happens, and removing the last column of each blocked column in (39), where there is only one non-zero item 1 in a certain row.

According to (18) of Schrijver (1998), the matrix of which each column contains exactly one 1 and exactly one $-1$ is totally unimodular.

In summary, since all the operations we have performed can preserve the total unimodularity of a matrix, and the final transformed matrix has been proved to be totally unimodular, the original matrix $A$ is totally unimodular.

Since matrix $A$ is totally unimodular and vector $b$ is integral, when (28) is feasible, there must exist an integral optimum for the LP relaxation in (28), which is also the optimal solution for the integer program in (27).

## 4. The form of the fractional solution

Sun and Bayen (2008) applied the interior point method to solve the problem, which does not guarantee to obtain an integral optimal solution. The major reason is not because of the low computational round-up error or the inappropriate step-length but because of the multiple optimal solutions of the LP relaxation. More accurately, there are multiple optimal extreme point solutions in this problem. These multiple optimal extreme points will form an *Optimal Polyhedron* which is the subset of the *Feasible Polyhedron* defined by the constraints.

For instance, in a 2$D$ case as shown in Fig. 3, the feasible polyhedron $P_{\text{fs}}$ is defined by five extreme points $A$, $B$, $C$, $D$ and $E$. Suppose under a certain objective function, the two extreme points $A$ and $B$ are both the optimal solutions denoted as $opt_1$ and $opt_2$. The line segment $\overline{AB}$ connecting these two optimal extreme points is called the optimal polyhedron $P_{\text{opt}}$ because all the points along $\overline{AB}$ are also optimal as any point of $P_{\text{opt}}$ can be written as the linear combination of the two optimal extreme points $\rho \cdot opt_1 + (1-\rho)\cdot opt_2$.

Generally $P_{\text{fs}}$ is the feasible polyhedron defined by constraints while the optimal polyhedron $P_{\text{opt}}$ is resulted by multiple optimal extreme points. Any solution in $P_{\text{opt}}$ is a linear combination of the optimal extreme point solutions and can be written as $\sum_i^{|opt|}\rho_i \cdot opt_i$, with $\sum_i^{|opt|}\rho_i = 1$, where $|opt|$ is the number of optimal extreme point solutions.

Although every optimal vertex solution $opt_i$ is integral because of total unimodularity, the linear combination of them cannot be guaranteed integral. There are fractional optimal solutions inside $P_{\text{opt}}$. Since the interior point method starts inside $P_{\text{fs}}$ and walks toward the boundaries of $P_{\text{fs}}$ instead of the extreme points, an inner point of the subset $P_{\text{opt}}$ is usually achieved. That is why the interior point method gives out the fractional optimum in Sun and Bayen (2008).

## 5. The Dantzig–Wolfe Decomposition on large-scale study

We decide to choose simplex related methods to find optimal extreme point solutions for large-scale TFM problems. In order to speed up it by taking advantage of $A$'s sparsity and the *block-angular structure* (Chvatal, 1983), we exploit the *Dantzig–Wolfe Decomposition* (DWD) method (Dantzig and Wolfe, 1961). Based on the property of simplex method (Chvatal, 1983), the DWD method guarantees the integral optimum.

### 5.1. Rearrangement for Dantzig–Wolfe Decomposition

Formulation (25) is rearranged into the canonical form for DWD. We group blocks $M_{1i}, M_{2i}, \ldots, M_{Si}$ in column $i$ into a single block called $M_i$ for $i = 1,2,\ldots,M$, and flip the positions of $M_i$'s and the dynamics constraints $A_i$'s. The new constraint matrix $A_{\text{DW}}$ and vector $b_{\text{DW}}$ are

$$A_{\text{DW}} = \begin{bmatrix} M_1 & M_2 & \ldots & M_M \\ A_1 & & & \\ & A_2 & & \\ & & \ddots & \\ & & & A_M \end{bmatrix}, \quad b_{\text{DW}} = \begin{bmatrix} v \\ f_1 \\ f_2 \\ \vdots \\ f_M \end{bmatrix}, \tag{40}$$

where the grouped sector counts capacity is $v = [v_1; v_2; \ldots; v_S]$. The cost vector $c_{\text{DW}}$ in (26) can be written as

$$c_{\text{DW}} = [c_{\text{DW}_1}; c_{\text{DW}_2}; \ldots; c_{\text{DW}_M}], \tag{41}$$

where $c_{\text{DW}_i} = [c_i; \mathbf{0}_i]$, with $c_i$ and $\mathbf{0}_i$ from (26).

This rearranged formulation can be solved by the DW Decomposition, which transforms the original problem into a *master problem* and its *subproblems* (Chvatal, 1983).

### 5.2. The Dantzig–Wolfe Decomposition algorithm

For the $i$th subproblem as below:

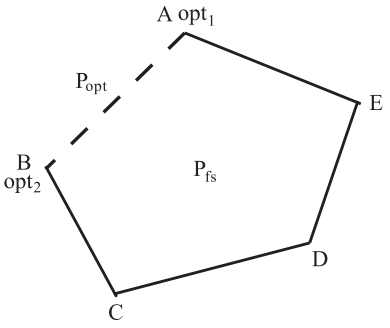


**Fig. 3.** The feasible and optimal polyhedra in a 2-dimension example.

$$\begin{aligned} &\min \quad c'_{\mathrm{DW}_i} x_i, \\ &\text{s.t.} \quad A_i x_i \leqslant f_i, \end{aligned} \tag{42}$$

there are $V_i$ extreme points $x_i^{(j)}, j = 1, 2, \ldots, V_i$. We denote $P_{ij} = M_i x_i^{(j)}$ and $c_{ij} = c'_{\mathrm{DW}_i} x_i^{(j)}$.

The $c_{\mathrm{MP}}$, $A_{\mathrm{MP}}$, $b_{\mathrm{MP}}$ of the master problem (Chvatal, 1983) are:

$$c_{\mathrm{MP}} = [c_{11} \ldots c_{1V_1} c_{21} \ldots c_{2V_2} \ldots c_{M1} \ldots c_{MV_M}]',$$

$$A_{\mathrm{MP}} = \begin{bmatrix} P_1 & P_2 & \ldots & P_M \\ \mathbf{1}'_{\gamma 1} & & & \\ & \mathbf{1}'_{\gamma 2} & & \\ & & \ddots & \\ & & & \mathbf{1}'_{\gamma M} \end{bmatrix}, \quad b_{\mathrm{MP}} = \begin{bmatrix} v \\ 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}, \tag{43}$$

where $P_i = [P_{i1}, \ldots, P_{iV_i}]$ and $\mathbf{1}'_{\gamma i} = [1, \ldots, 1]$ with the length of $V_i$.

Simplex related methods are used in the DWD method (Dantzig and Wolfe, 1961). We adopt the classical simplex method and the interior point method with crossover (Anonyms, 2010) as two kernel solvers in our DWD implementation. In these two simplex related methods, there is a vector of "prices" $[\pi', \tilde{\pi}']$, where $\pi$ is of length $TS$ and $\tilde{\pi}$ is of $M$. Each item of the price vector is associated with one constraint in formulation (43).

Starting from an initial basis, the iterative DWD process begins, where the master problem transfers the price vector to subproblems while the subproblems provide the entering basic vector which has the minimum negative reduced cost. When we have a new basis, we update the price vector and transfer it to the subproblems again. The iteration will be terminated when the master problem converges. The iteractions between the master problem and its subproblems are described in Fig. 4.

The *Initial Basis Generation* (IBG) is performed to construct the initial basis. For each path, let the aircraft count in each cell flow to its next cell, which will obtain the initial basic vectors. Multiple optima were found in Section 4. However, in every iteration of the decomposition algorithm, only one optimal solution of each subproblem is used to update the basis. In order to obtain the integral optimal solution, the optimal extreme points in subproblems must be reached by a simplex related method solver. In this work we adopt the classical simplex method and the interior point method with crossover provided by CPLEX (Anonyms, 2010). The absolute value of the minimum negative reduced cost is compared to a given convergence threshold $\delta_c$ as the stopping criterion. When it is less than $\delta_c$, the algorithm is considered converged. The complete DW Decomposition is given in Algorithm 1.

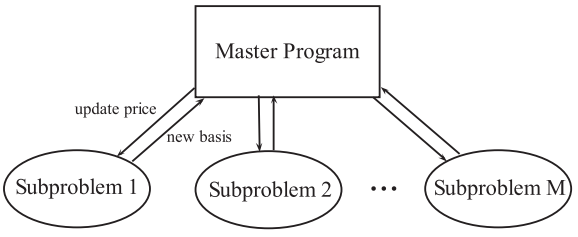


**Fig. 4.** Interactions in DW decomposition.

**Algorithm 1.** DW Decomposition Algorithm

```
1: Run Initial Basis Generation
2: while not converged do
3:    Solve the master problem
4:    Update the price vector based on current basis
5:    for i = 1 to M do
6:       Plug π and π̃ into each subproblem i
7:       Find the optimal solution of subproblem i by a simplex related method
8:       if the minimum negative reduced cost < −δ_c then
9:          The basis is updated by the corresponding column
10:      end if
11:   end for
12:   if the basis is not updated then
13:      Converged
14:   end if
15: end while
```

### 5.3. Large-scale simulation

The nationwide air traffic data on May 24, 25, 26, 2010 is utilized to evaluate the performances of Sun and Bayen's interior point method without decomposition and Wei, Cao and Sun's DW Decomposition method with different optimization configurations. A workstation is equipped with a 2.8 GHz 8-processors INTEL i7 CPU and 32G RAM for the single workstation experiments. The other three workstations have slightly different CPUs and memory sizes. The historical flight trajectories with 1 min updating rate from Aircraft Situation Display to Industry (ASDI) and Enhanced Traffic Management System (ETMS) (Volpe National Transportation Center, 2005) are loaded to build the Large-Capacity Cell Transmission Model as described in Sun and Bayen (2008). With our optimization tool and parallel computing setup presented in Sun et al. (2011), Cao and Sun (2011, 2012), the CPLEX (Anonyms, 2010) is used to implement both the interior point method and the DW Decomposition method in C++. The large-scale nationwide simulations have been performed with 1-h and 2-h planning horizons for each day. The air traffic during two peak periods, which are from 18:00 to 19:00 and from 18:00 to 20:00 eastern time, are optimized by a single workstation or multiple workstations. 3419 flight paths (3419 subproblems) are identified in the decomposition method. The convergence threshold $\delta_c$ is set to $1 \times 10^{-6}$.

By default the CPLEX starts a "crossover procedure" after the interior point method in order to obtain an optimal extreme point (basic solution) from the interior point method solution. However, the crossover procedure slows down the optimization, especially for large-scale cases. Sun and Bayen (2008) turned off the crossover procedure to increase the optimization speed. In this section the crossover option in CPLEX is switched on and off to study its influence on the interior point method. The experiments of nine different optimization configurations are performed, which are listed in Table 2. Configurations 1–5 are all single workstation experiments and Configurations 6–9 are parallel computing experiments with 2 or 4 workstations. Configuration 1 is the same experiment setup as the one in Sun and Bayen (2008), where no decomposition technique is applied and the crossover is turned off. In this experiment the optimal solution is fractional. Configuration 2 is implemented to show the effect of crossover procedure on the large-scale interior point method. The integral optimal solution is guaranteed by crossover procedure in this experiment. Configuration 3 is expected to have the fastest optimization speed in a single workstation experiment using the decomposition technique without crossover. However, the integral solution is not guaranteed. The DW Decomposition method is implemented in both Configuration 4 and Configuration 5 on a single workstation. The interior point method with crossover procedure is applied in the DW Decomposition in Configuration 4 and the simplex method solver is applied in Configuration 5. Both simplex related methods can guarantee the integral optimal solution. In Configurations 6 and 7 the parallel computing framework (Cao and Sun, 2012) is implemented on 2 workstations with

**Table 2**
Optimization configurations in the large-scale simulation.

| Index | Configuration |
|---|---|
| Configuration 1 | Interior point method without crossover. No decomposition |
| Configuration 2 | Interior point method with crossover. No decomposition |
| Configuration 3 | Interior point method without crossover is applied in Algorithm 1 |
| Configuration 4 | Interior point method with crossover is applied in Algorithm 1 |
| Configuration 5 | Simplex method is applied in Algorithm 1 |
| Configuration 6 | 2-Workstation parallel computing with Configuration 4 |
| Configuration 7 | 2-Workstation parallel computing with Configuration 5 |
| Configuration 8 | 4-Workstation parallel computing with Configuration 4 |
| Configuration 9 | 4-Workstation parallel computing with Configuration 5 |

**Table 3**
Computation times on large-scale case (min).

| Configuration Index | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| May 24 1-h planning | 14.93 | 17.32 | 13.16 | 25.08 | 28.14 | 16.24 | 18.89 | 9.51 | 11.67 |
| May 25 1-h planning | 15.18 | 18.17 | 13.53 | 24.64 | 28.73 | 16.65 | 19.03 | 9.63 | 11.75 |
| May 26 1-h planning | 15.35 | 17.76 | 13.42 | 24.82 | 28.78 | 16.71 | 19.29 | 9.86 | 12.22 |
| May 24 2-h planning | 41.47 | 48.97 | 38.29 | 69.21 | 75.04 | 38.82 | 42.32 | 21.82 | 28.34 |
| May 25 2-h planning | 41.26 | 48.43 | 37.84 | 69.47 | 75.25 | 39.11 | 42.38 | 21.93 | 28.18 |
| May 26 2-h planning | 42.14 | 49.24 | 38.77 | 69.94 | 76.13 | 39.54 | 42.64 | 22.17 | 28.92 |

the same configurations as Configurations 4 and 5 respectively. Configurations 8 and 9 are the 4-workstation parallel computing implementation with the same configurations as Configurations 4 and 5 respectively. Computation times of the nine configurations are listed in Table 3.

In Table 3, Configuration 1 and Configuration 3 fail to offer the integral and optimal solution. Their computation times are only listed as benchmarks. Configuration 2 shows that the crossover takes considerable time to generate an integral solution from the interior point method solution. Configuration 4 and Configuration 5 are two DW Decomposition optimization configurations for the large-scale computation on a single workstation. Their optimal solutions are guaranteed to be integral. The computation times of Configuration 4 and Configuration 5 are both longer than the one of only applying the interior point method with crossover in Configuration 2. In Configuration 4 the crossover procedure is executed in every iteration of the decomposition method, which slows down the optimization. The simplex method is always slower than the interior point method with/without the crossover. The DW decomposition with parallel computing further accelerates the optimization speed. The 2-workstation and 4-workstation parallel computing implementations in Configurations 6–9 outperform the computation speed of Configuration 2. The results show that the DW decomposition method with parallel computing can find the integral optimal solution of the CTM(L) problem efficiently, i.e., to solve a nationwide large-scale problem with 1-h planning horizon takes about 10 min with 4 workstations and it takes about 21 min to solve a large-scale problem with 2-h planning horizon.

## 6. Conclusion

In this paper, the CTM(L) is introduced and an integer programming optimization problem is formulated. We prove that there exists an integral optimal solution for the corresponding LP relaxation because of its total unimodularity and this solution is also optimal for the integer program. We demonstrate that the simplex related methods guarantee the integral optimum and apply the Dantzig–Wolfe Decomposition algorithm which takes advantage of matrix $A$'s special block structure to speed up the computation. The large-scale experiments are performed to evaluate interior point method and DW Decomposition. The results show that the DW Decomposition with parallel computing can obtain the integral optimal solution with high computational efficiency.

## Appendix A

In this section we show how to obtain (30) from (29), why $L_{ji}$ and $R_{ji}$ are lower triangular and what the structures of $s'_{ji}$ and $t'_{ji}$ are. For the ease of demonstration, a small size example is used for derivation.

$$\begin{bmatrix} I-P & -Q \\ P-I & Q \\ -I & I \\ M^I_{1M} & \mathbf{0}_\beta \\ M^I_{2M} & \mathbf{0}_\beta \\ \vdots & \vdots \\ M^I_{SM} & \mathbf{0}_\beta \end{bmatrix} = \begin{bmatrix} I & & & & 0 & & & \\ -P_o & I & & & -Q_o & 0 & & \\ & -P_o & I & & & -Q_o & 0 & \\ & & -P_o & I & & & -Q_o & 0 \\ -I & & & & 0 & & & \\ P_o & -I & & & Q_o & 0 & & \\ & P_o & -I & & & Q_o & 0 & \\ & & P_o & -I & & & Q_o & 0 \\ -I & & & & I & & & \\ & -I & & & & I & & \\ & & -I & & & & I & \\ & & & -I & & & & I \\ s'_{1M} & & & & 0 & 0 & 0 & 0 \\ & s'_{1M} & & & 0 & 0 & 0 & 0 \\ & & s'_{1M} & & 0 & 0 & 0 & 0 \\ & & & s'_{1M} & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}, \tag{44}$$

where vector $s'_{1M}$ is defined in (22) and (23).

By the definition of $P_o$ and $Q_o$ in (10) and (12), we obtain $P_o + Q_o = I$. In Eq. (44), add the 5th column to the 1st column, add the 6th column to the 2nd column, add the 7th column to the 3rd column, add the 8th column to the 4th column, we have:

$$\begin{bmatrix}
I & & & & 0 & & & \\
-I & I & & & -Q_o & 0 & & \\
 & -I & I & & & -Q_o & 0 & \\
 & & -I & I & & & -Q_o & 0 \\
-I & & & & 0 & & & \\
I & -I & & & Q_o & 0 & & \\
 & I & -I & & & Q_o & 0 & \\
 & & I & -I & & & Q_o & 0 \\
0 & & & & I & & & \\
 & 0 & & & & I & & \\
 & & 0 & & & & I & \\
 & & & 0 & & & & I \\
s'_{1M} & & & & 0 & 0 & 0 & 0 \\
 & s'_{1M} & & & 0 & 0 & 0 & 0 \\
 & & s'_{1M} & & 0 & 0 & 0 & 0 \\
 & & & s'_{1M} & 0 & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots
\end{bmatrix}. \tag{45}$$

Based on Eq. (45), add the 4th column to the 3rd column, then add the resulted 3rd column to the 2nd column, then add the resulted 2nd column to the 1st column, we have:

$$\begin{bmatrix}
I & & & & 0 & & & \\
 & I & & & -Q_o & 0 & & \\
 & & I & & & -Q_o & 0 & \\
 & & & I & & & -Q_o & 0 \\
-I & & & & 0 & & & \\
 & -I & & & Q_o & 0 & & \\
 & & -I & & & Q_o & 0 & \\
 & & & -I & & & Q_o & 0 \\
0 & & & & I & & & \\
 & 0 & & & & I & & \\
 & & 0 & & & & I & \\
 & & & 0 & & & & I \\
s'_{1M} & & & & 0 & 0 & 0 & 0 \\
s'_{1M} & s'_{1M} & & & 0 & 0 & 0 & 0 \\
s'_{1M} & s'_{1M} & s'_{1M} & & 0 & 0 & 0 & 0 \\
s'_{1M} & s'_{1M} & s'_{1M} & s'_{1M} & 0 & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots
\end{bmatrix}. \tag{46}$$

According to Eq. (46), the 4th column right multiplied by $Q_o$ is added to the 7th column, the 3rd column right multiplied by $Q_o$ is added to the 6th column, the 2nd column right multiplied by $Q_o$ is added to the 5th column. Now we have:

$$\begin{bmatrix}
I & & & & & 0 & & & \\
& I & & & & & 0 & & \\
& & I & & & & & 0 & \\
& & & I & & & & & 0 \\
-I & & & & & 0 & & & \\
& -I & & & & & 0 & & \\
& & -I & & & & & 0 & \\
& & & -I & & & & & 0 \\
0 & & & & & I & & & \\
& 0 & & & & & I & & \\
& & 0 & & & & & I & \\
& & & 0 & & & & & I \\
s'_{1M} & & & & & 0 & 0 & 0 & 0 \\
s'_{1M} & s'_{1M} & & & & t'_{1M} & 0 & 0 & 0 \\
s'_{1M} & s'_{1M} & s'_{1M} & & & t'_{1M} & t'_{1M} & 0 & 0 \\
s'_{1M} & s'_{1M} & s'_{1M} & s'_{1M} & & t'_{1M} & t'_{1M} & t'_{1M} & 0 \\
\vdots & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & \vdots
\end{bmatrix}, \tag{47}$$

where the row vector $t'_{1M}$ is the result of the row vector $s'_{1M}$ right multiplied by $Q_o$ ($t'_{1M} = s'_{1M}Q_o$).

From Eq. (47), we can tell that $L_{1M}$ and $R_{1M}$ are lower triangular. Similarly, we know that all of the $L_{ji}$'s and $R_{ji}$'s are lower triangular.

## References


Andreatta, G., Brunetta, L., Guastalla, G., 1997. Multi-airport ground holding problem: a heuristic approach based on priority rules. In: Bianco, L., Dell'Olmo, P., Odoni, A. (Eds.), Modeling and Simulation in Air Traffic Management. Springer, Berlin Heidelberg, Berlin, Germany.

Ball, M., Hoffman, R., Lovell, D., Mukherjee, A., 2005. Response mechanisms for dynamic air traffic flow management. In: Proceedings of the 6th USA/Europe ATM 2005 R&D Seminar. Baltimore, MD.

Ball, M., Lulli, G., 2004. Ground delay programs: optimizing over included flight set based on distance. Air Traffic Control Quarterly 12 (1), 1–25.

Bayen, A., Raffard, R., Tomlin, C., 2006. Adjoint-based control of a new Eulerian network model of air traffic flow. IEEE Transactions on Control Systems Technology 14 (5), 804–818.

Bertsimas, D., Lulli, G., Odoni, A., 2008. The air traffic flow management problem: an integer optimization approach. In: Lodi, A., Panconesi, A., Rinaldi, G. (Eds.), Integer Programming and Combinatorial Optimization. No. 5035 in Lecture Notes in Computer Sciences. Springer, Berlin Heidelberg.

Bertsimas, D., Patterson, S., 1998. The air traffic flow management problem with enroute capacities. Operations Research 46 (3), 406–422.

Bloem, M., Sridhar, B., 2008. Optimally and equitably distributing delays with the aggregate flow model. In: IEEE/AIAA 27th Digital Avionics Systems Conference (DASC). St. Paul, Minnesota, pp. 3.D.4-1–3.D.4-14.

Cao, Y., Sun, D., 2011. A link transmission model for air traffic flow management. AIAA Journal of Guidance, Control and Dynamics 34 (5), 1342–1351.

Cao, Y., Sun, D., 2012. A parallel computing framework for large-scale air traffic flow optimization. IEEE Transactions on Intelligent Transportation Systems 13 (4), 1855–1864.

Chvatal, V., 1983. Linear Programming. W.H. Freeman.

Daganzo, C.F., 1994. The cell transmission model: a dynamic representation of highway traffic consistent with the hydrodynamic theory. Transportation Research Part B 28 (4), 269–287.

Daganzo, C.F., 1995. The cell transmission model, part ii: Network traffic. Transportation Research Part B 29 (2), 79–93.

Dantzig, G., Wolfe, P., 1961. The decomposition algorithm for linear programs. Econometrica 29 (4), 767–778.

Dell'Olmo, P., Lulli, G., 2003. A dynamic programming approach for the airport capacity allocation problem. IMA Journal of Management Mathematics 14 (3), 235–249.

Gilbo, E., 1993. Airport capacity: representation, estimation, optimization. IEEE Transactions on Control Systems Technology 1 (3), 144–154.

Grabbe, S., Sridhar, B., Mukherjee, A., 2007. Central east pacific flight scheduling. In: AIAA Conference on Guidance, Navigation, and Control Conference and Exhibit. Hilton Head, SC, AIAA Paper 2007-6447.

Grabbe, S., Sridhar, B., Mukherjee, A., 2009. Sequential traffic flow optimization with tactical flight control heuristics. AIAA Journal of Guidance, Control, and Dynamics 32 (3), 810–820.

Helme, M., 1992. Reducing air traffic delay in a space-time network. In: IEEE International Conference on Systems, Man and Cybernetics, vol. 1, pp. 236–242.

Hoffman, R., Ball, M., 2000. A comparison of formulations for the single airport ground holding problem with banking constraints. Operations Research 48 (4), 578–590.

ILOG CPLEX 11.0 User's Manual, 2010. <http://www.decf.berkeley.edu/help/apps/ampl/cplex-doc/> (retrieved 06.06.10).

Kopardekar, P., Green, S., 2005. Airspace restriction planner for sector congestion management. In: AIAA 5th ATIO and 16th Lighter-Than-Air Sys Tech. and Balloon Systems Conferences. Arlington, Virginia, AIAA Paper 2005-7435.

Lindsay, K., Boyd, E., Burlingame, R., 1993. Traffic flow management modeling with the time assignment model. Air Traffic Control Quarterly 1 (3), 255–276.

Lulli, G., Odoni, A., 2007. The european air traffic flow management problem. Transportation Science 41 (4), 431–443.

Menon, P., Sweriduk, G., Bilimoria, K., 2002. A new approach to modeling, analysis and control of air traffic flow. In: Proceedings of AIAA Guidance, Navigation and Control Conference. Monterey, CA.

Navazio, L., Romanin-Jacur, G., 1998. The multiple connections multi-airport ground holding problem: models and algorithms. Transportation Science 32 (3), 268–276.

Nolan, M., 2003. Fundamentals of Air Traffic Control, fourth ed. Brooks Cole, Reading, MA.

Odoni, A., 1987. The flow management problem in air traffic control. In: Flow Control of Congested Networks. Springer Verlag, Berlin, Germany.

Rios, J., Ross, K., 2008. Solving high-fidelity, large-scale traffic flow management problems in reduced time. In: AIAA Aviation Technology, Integration and Operations Conference. Anchorage, Alaska, AIAA Paper 2008-8910.

Rios, J., Ross, K., 2010. Massively parallel dantzig-wolfe decomposition applied to traffic flow scheduling. Journal of Aerospace Computing, Information, and Communication 7 (1), 32–45.
Schrijver, A., 1998. Theory of Linear and Integer Programming. Wiley.
Sherali, H., Smith, J., Trani, A., 2002. An airspace planning model for selecting flights plans under workload, safety, and equity considerations. Transportation Science 36 (4), 378–397.
Sherali, H., Staats, R., Trani, A., 2003. An airspace planning and collaborative decision making model: part I – probabilistic conflicts, workload, and equity consideration. Transportation Science 37 (4), 434–456.
Sherali, H., Staats, R., Trani, A., 2006. An airspace planning and collaborative decision making model: part II – cost model, data considerations, and computation. Transportation Science 40 (2), 147–164.
Sridhar, B., Chatterji, G., Grabbe, S., Sheth, K., 2002. Integration of traffic flow management decisions. In: AIAA Conference on Guidance, Navigation, and Control Conference and Exhibit. Monterey, CA.
Sun, D., Bayen, A., 2008. Multicommodity Eulerian–Lagrangian large-capacity cell transmission model for en route traffic. AIAA Journal of Guidance, Control, and Dynamics 31 (2), 616–628.
Sun, D., Clinet, A., Bayen, A., 2011. A dual decomposition method for sector capacity constrained traffic flow optimization. Transportation Research Part B 45 (6), 880–902.
Terrab, M., Odoni, A.R., 1993. Strategic flow management for air traffic control. Operations Research 41 (1), 138–152.
Volpe National Transportation Center, 2005. Enhanced Traffic Management System (ETMS). Tech. Rep., U.S. Department of Transportation, Cambridge, MA.
Vossen, T., Ball, M., 2005. Optimization and mediated bartering models for ground delay programs. Naval Research Logistics 53 (1), 75–90.
Vossen, T., Ball, M., 2006. Slot trading opportunities in collaborative ground delay programs. Transportation Science 40 (1), 29–43.
Vossen, T., Ball, M., Hoffman, R., Wambsganss, M., 2003. A general approach to equity in traffic flow management and its application to mitigating exemption bias in ground delay programs. In: Proceedings of the 5th USA/Europe ATM 2003 R&D Seminar. Budapest, Hungary.
Vranas, P., Bertsimas, D., Odoni, A.R., 1994. The multi-airport ground holding problem in air traffic control. Operations Research 42 (2), 249–261.
Wanke, C., Greenbaum, D., 2007. Incremental probabilistic decision making for en route traffic management. Air Traffic Control Quarterly 15 (4), 299–319.
Waslander, S., Raffard, R., Tomlin, C., 2008a. Market-based air traffic flow control with competing airlines. AIAA Journal of Guidance, Control and Dynamics 31 (1), 148–161.
Waslander, S., Roy, K., Johari, R., Tomlin, C., 2008b. Lump sum markets for air traffic flow control with competitive airlines. IEEE Special Issue on Aviation Information Systems 96 (12), 2113–2130.