Optimal Sequential Backward Flow Saturation For Cell Transmission Model

Peng Wei
School of Aeronautics and Astronautics
Purdue University
West Lafayette, IN 47907, USA
Email: weip@purdue.edu

Amine Meksoub
Department of Applied Mathematics
Ecole Polytechnique
91120, Palaiseau, France
Email: amine.meksoub@polytechnique.edu

Dengfeng Sun
School of Aeronautics and Astronautics
Purdue University
West Lafayette, IN 47907, USA
Email: dsun@purdue.edu

***Abstract*—The cell transmission model for air traffic flow was built by Sun and Bayen. The problem of managing the flow under sector capacity constraints was formulated as an integer programming problem. However, the corresponding relaxed linear programming in their work failed to give the integer solution for the aircraft count, which is not applicable in practice. In this paper we propose the *Optimal Sequential Backward Flow Saturation* paradigm to find the optimal integer solution. Firstly we introduce the *Backward Flow Saturation* on a single path. Secondly the *Sequential Backward Flow Saturation* is developed to solve a M-path problem given a certain path sequence. Then we show that the path sequence is critical and the Sequential Backward Flow Saturation under an arbitrary path sequence can not guarantee the optimal solution. Finally inspired by the Branch and Bound technique (B&B), we design a decision tree searching to provide the Sequential Backward Flow Saturation the optimal path sequence. The final algorithm *Optimal Sequential Backward Flow Saturation Algorithm* is proved to give the optimal integer solution.**

I. Introduction

The past and the forecast air traffic flow in the US present a sustained growth making it essential to study the *National Airspace System* (NAS), which is a complex physical system consisting of aircraft, control facilities, procedures, navigation and surveillance equipment, analysis equipment, decision support tools, and traffic controllers. The air traffic management is currently attracting intensive research to help controllers handle the increasing complexity of traffic flow in the en route airspace. The development of effective and efficient models in *Traffic Flow Management* (TFM) is highly needed.

Several classes of traffic flow models have emerged from recent studies and current popular TFM schemes mainly focus on ground delay and flight rerouting. Odoni was the first to formulate the TFM problem to design optimal strategies by assigning ground delays to flights [1]. Helme was among the first who included en route capacity restrictions [2]. Lindsay presented a disaggregate 0-1 programming model for deciding ground and airborne holding under airport and airspace capacity constraints [3]. Sridhar proposed an integrated three-step hierarchical method to determine TFM plans consisting of the practical restrictions [4]. Eulerian paradigm [5] developed by Bayen relies on analytical equations (partial differential equations) to find solutions in terms of aircraft count.

In [6] the authors presented a traffic flow model called the *Large-capacity Cell Transmission Model*, in short CTM(L), which is a variation of the model in [7]. They 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 programming containing billions of variables and constraints. It was then relaxed to a *linear programming* (LP) for computational efficiency. However, solving the linear programming by large scale commercial software results in fractional optimal solutions which can not be implemented as en route holding control in practical situation. This is the major motivation of our work.

In this paper, we resolve the CTM(L) problem and guarantee an optimal and integral solution using the Sequential Backward Flow Saturation (SBFS) by successively constructing air traffic flows along each path in a certain path sequence. It is then found that the path sequence is critical when computing the state variables and delay control volumes. Therefore an algorithm inspired by the Branch and Bound (B&B) technique is developed in order to find the optimal path sequence for SBFS. With the optimal path sequence provided, the SBFS is called the Optimal Sequential Backward Flow Saturation (OSBFS) and the optimal integer solution is achieved. Besides presenting the Optimal Sequential Backward Flow Saturation Algorithm, another major contribution of this paper is to show the detailed and explicit optimality proof for the entire procedure.

The rest of the paper is organized as follows: Section II introduces the CTM(L) model and illustrates the optimization problem formulation. Section III presents the Sequential Backward Flow Saturation for calculating the optimal delay control under a given path sequence. In Section IV the B&B decision tree is applied to find the optimal path sequence for the SBFS and an example is demonstrated at the end of this section. Conclusions are made in the fifth section of the paper.

II. Problem Formulation

The air traffic network topology in CTM(L) can be described by an unidirectional graph $G = (E, V)$, in which E is the set of edges or links, and V the set of vertices. Based on G, the authors formulated the optimization problem in [6]. Each path in the network is divided into homogeneous cells by

en route flight time interval. Let $x_i(t)$ be the aircraft count in cell i at time t along one of the flight paths. When no control is actioned, the system's dynamics can be described by:

$$x_{i+1}(t+1) = x_i(t).$$

It is assumed that the above equation holds for all flows, unless the traffic is slown down by delay controls due to sector capacity constraints.

Cells along a path are grouped into several different sectors where each sector is characterized by a capacity which may vary with time. The number of aircraft selected for delay control in cell i at time t is denoted by $u_i(t)$. Then the cell transmission model with delay control is based on the following dynamics:

$$x_{i+1}(t+1) = x_i(t) + u_i(t) - u_{i+1}(t).$$

In order to formulate the complete optimization problem, we define the following variables and parameters:

- The aircraft count state vector $x = [x_1(0), ..., x_{N-1}(0), ..., x_1(T-1), ..., x_{N-1}(T-1)]'$, where N is the number of cells along the path and T is the total planning time. $x_k(t)$ describes the aggregate aircraft count in cell k at time t.
- The delay control vector $u = [u_1(0), ..., u_{N-1}(0), ..., u_1(T-1), ..., u_{N-1}(T-1)]'$. $u_k(t)$ is the the number of aircraft to be delayed in cell k at time t.
- The input $f(t)$ which presents, for $t = 0, ..., T-1$, the entry aircraft count into the path during a unit time interval from t to $t+1$.
- The initial conditions $x_1^0, ..., x_N^0$ which are the initial aircraft counts in each cell on the path.

In this problem, all values, including the states, inputs, control counts, should be integers.

In [6] and [8], the authors formulated the problem using the following equations for each path in the Cell Transmission Model:

- The initial conditions:

$$x_k(0) = x_k^0, \qquad \text{for} \quad k = 2...N-1. \tag{1}$$

- The boundrary conditions:

$$\begin{aligned} &x_1(0) = f(0) + x_1^0, \\ &x_1(t) = f(t) + u_1(t-1), \quad t = 1, 2, ..., T-1. \end{aligned} \tag{2}$$

- The CTM(L) dynamics:

$$\begin{aligned} &x_k(t) = x_{k-1}(t-1) - u_{k-1}(t-1) + u_k(t-1), \\ &k = 2, ..., N-1, \quad t = 1, 2, ..., T-1. \end{aligned} \tag{3}$$

- Notice that the sector capacity constraint may vary at different time periods. In a model with S sectors, let $\nu = [\nu_1(0), ..., \nu_1(T-1), ..., \nu_S(0), ..., \nu_S(T-1)]$ be the vector of capacities which denotes the maximal allowed aircraft count for each sector at time t. In a CTM(L) problem with multiple paths, let $X_i = [x_i, u_i]'$ be the vector of all unknown variables. Where the same matrix form is adopted as in [8] for sector capacity constraints, we have

$$M \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_M \end{bmatrix} \leq \begin{bmatrix} \nu_1 \\ \nu_2 \\ \vdots \\ \nu_S \end{bmatrix}. \tag{4}$$

For each aircraft, the total time to get through the path is divided into time intervals of length τ. Consequently, τ is the time spent by one aircraft in one cell. The objective of the problem is to minimize the total travel time of all the aircraft. The integer formulation of the optimization problem is shown as follow:

$$\begin{aligned} &\min \textstyle\sum_{i=1}^{M} c_i x_i \\ &\text{s.t.} \qquad (1),\ (2),\ (3) \text{ and } (4) \end{aligned} \tag{5}$$

where c_i for $i = 1, ..., M$ is the vector $[\tau, ..., \tau]$. For convenience we set τ to 1 for the rest of the paper.

III. Sequential Backward Flow Saturation

Given a CTM(L) model with S sectors, M paths and N_i cells in each path i, we develop Sequential Backward Flow Saturation to find the optimal integer solution. In this section, we first present *Backward Flow Saturation* for a single path and then an algorithm with M iterations is proposed as Sequential Backward Flow Saturation (SBFS). At the beginning we assume that the solution of this method is not related to the order in which the paths are constructed. The default path sequence is given by the path indices $1, 2, ..., M$. In the next section, we will show that path order is actually critical and we will develop a method to determine the optimal order.

A. Backwards Flow Saturation for a single path

At each iteration of SBFS, i.e., iteration i, we construct the traffic flow on the ith path, which is considered as the only path existed in the current network and we update the sector capacity vector ν with the residual capacities calculated from the previous $i-1$ iterations.

At iteration i the algorithm determines the state vector of the path i

$$x = [x_1^i(0), ..., x_{N-1}^i(0), ..., x_1^i(T-1), ...x_{N-1}^i(T-1)]',$$

and the delay control vector

$$u = [u_1^i(0), ..., u_{N-1}^i(0), ..., u_1^i(T-1), ...u_{N-1}^i(T-1)]'.$$

In detail, we construct the traffic flow on the ith path for $t = 0, ..., T-1$. We start the process from the last cell with the condition $u_{N_i}^i(t) = 0$ for $t \in \{0, ..., T-1\}$. We can see the effect of the delay control $u_{N_i}^i(t)$ in cell N_i at the time instant of $t \leq T-1$ increases the objective function in (5) by $u_{N_i}^i(t)$. In line 7 of Algorithm 1, $x_k^i(t)$ and $u_{k-1}^i(t-1)$ are computed simultaneously based on sector residual capacity at step t using the following equation:

$$\begin{aligned} &x_k^i(t) = x_{k-1}^i(t-1) - u_{k-1}^i(t-1) + u_k^i(t-1) \\ &k = 1...N-1, \ t = 1, ..., T-1, \end{aligned}$$

where $u_k^i(t-1)$ is determined by induction and $x_{k-1}^i(t-1)$ is determined at step $t-1$.

For cell k at the time instant of t, the number of aircraft delayed is set to 0 unless this decision violates the sector capacities at $t+1$, which is called the *flow saturation* making the aircraft count saturate the corresponding capacity. To be more precise, all the aircraft in cell k at t are transferred to cell $k+1$ at $t+1$ if $x_{k+1}^i(t+1)$ still respects the capacity constraints in the relation (4).

Therefore, we can go backwards step by step to the first cell by induction and the aircraft count in each cell on path i is then constructed. The procedure along the single path i is called Backward Flow Saturation.

B. The Sequential Backward Flow Saturation Algorithm and its solution feasibility

To solve the problem with M paths, we develop an iterative algorithm which computes the flow path by path with the use of Backward Flow Saturation, which is named Sequential Backward Flow Saturation Algorithm (SBFS).

Algorithm 1 SBFS Algorithm

```
The solution vector X contains the vectors x^1, u^1, ..., x^M, u^M
for i = 1 to M do
    determine [x_0^i(0), ..., x_{N-1}^i(0)] using the initial conditions
    set the control counts u_{N_i}(t) = 0 for t = 0, ..., T-1
    for t = 1 to T-1 do
        for k = N_i, ..., 2 do
            compute x_k^i(t) and u_{k-1}^i(t-1)) based on residual capacity
            update the residual capacity vector ν
        end for
        compute x_1^i(t) using the boundary conditions
    end for
end for
```

From the procedure of Algorithm 1, we can see that the algorithm is ensured to give an integer output. Besides, the state and delay control vectors x^i and u^i satisfy the boundary and initial conditions (1) and (2) and the dynamic equation (3). Additionally, updating sector capacities guarantees that the solution respects constraint (4). Therefore the SBFS gives an integer feasible solution.

C. The proof of optimality

1) The optimality of Backward Flow Saturation: We first prove that Backward Flow Saturation guarantees the optimal solution for a single path, which is equivalent to prove the following theorem.

Theorem 1: In the case of a CTM(L) problem with one path, the solution given by Algorithm 1 is optimal.

To prove Theorem (1), we will use the following lemma.

Lemma 1: For a single path with infinite sector capacities, the optimal solution $x^{opt} = [x, u]$ satisfies $u = 0$ and $x_k(t) = x_{k-1}(t-1)$ $(k = 2...N-1, t = 1, ..., T-1)$, with the intial and boundary constraints as below.

$$\begin{aligned} &x_k(0) = x_k^0, \qquad k = 2...T-1 \\ &x_1(0) = f(0) + x_1^0,\ x_1(t) = f(t), \qquad t = 1...T-1 \end{aligned} \tag{6}$$

Proof: Since $x_k(t) = x_{k-1}(t-1)$ is the consequence of $u = 0$, we only need to prove x^{opt} satisfies $u = 0$, which means no delay control is applied.

We now apply a delay control $u_{i_0}(t_0)$ on top of solution x^{opt}. Let's denote the resulted objective function value as c_f which also can be called the cost c_f. $u_{i_0}(t_0)$ aircraft are added to the aircraft count in cell i_0 instead of being transferred to cell i_0+1 at t_0+1. $x_i(t)$ is the aircraft count in cell i at time t corresponding to the solution x^{opt}. We study the relationship between c_f, the optimal cost c_{opt} and $u_{i_0}(t_0)$. By using the contradiction proof, we assume $c_f < c_{opt}$.

- Case 1 $(T - t_0 + i_0 \le N)$:

$$\begin{aligned} c_f \;=\; & \sum_{i,t\le t_0} x_i(t) + \sum_{i,t_0+1\le t\le T} \Big(\sum_{i<i_0+t-t_0-1} x_i(t) \\ & + \underbrace{u_{i_0}(t_0) + x_i(t)}_{i=i_0+t-t_0-1} + \underbrace{x_{i_0}(t) - u_{i_0}(t_0)}_{i=i_0+t-t_0} + \sum_{i>i_0+t-t_0} x_i(t) \Big). \end{aligned}$$

  After canceling $u_{i_0}(t_0)$ we have:

$$\begin{aligned} c_f \;=\; & \sum_{i,t\le t_0} x_i(t) + \sum_{i,t_0+1\le t\le T} \Big(\sum_{i<i_0+t-t_0-1} x_i(t) \\ & + \underbrace{x_{i_0+t-t_0-1}(t)}_{i=i_0+t-t_0-1} + \underbrace{x_{i_0}(t)}_{i=i_0+t-t_0} + \sum_{i>i_0+t-t_0} x_i(t) \Big). \end{aligned}$$

  It is recognized that the summation of the right hand side of the equation is x^{opt}, thus we have $c_f = c_{opt}$ which is a contradiction.

- Case 2 $(T - t_0 + i_0 > N)$:

$$\begin{aligned} c_f \;=\; & \sum_{i,t\le t_0} x_i(t) + \sum_{i,t_0+1\le t\le T} \Big(\sum_{i<i_0+t-t_0-1} x_i(t) \\ & + \underbrace{u_{i_0}(t_0) + x_i(t)}_{i=i_0+t-t_0-1} + \underbrace{x_{i_0}(t) - u_{i_0}(t_0)}_{i=i_0+t-t_0} + \sum_{i>i_0+t-t_0} x_i(t) \Big) \\ & + \underbrace{\left(\sum_{i<i_0+t-t_0-1} x_i(t) \right) + u_{i_0}(t_0) + x_{i_0+t-t_0-1}(t)}_{t=t_0+1-i_0+N} \\ & + \sum_{i,t_0+2-i_0+N\le t\le T} \sum_i x_i(t). \end{aligned}$$

  After cancelling $u_{i_0}(t_0)$ we get the following expression:

$$\begin{aligned} c_f \;=\; & \sum_{i,t\le t_0} x_i(t) + \sum_{i,t_0+1\le t\le T} \Big(\sum_{i<i_0+t-t_0-1} x_i(t) \\ & + \underbrace{x_{i_0+t-t_0-1}(t)}_{i=i_0+t-t_0-1} + \underbrace{x_{i_0}(t)}_{i=i_0+t-t_0} + \sum_{i>i_0+t-t_0} x_i(t) \Big) \\ & + \underbrace{\left(\sum_{i<i_0+t-t_0-1} x_i(t) \right) + u_{i_0}(t_0) + x_{i_0+t-t_0-1}(t)}_{t=t_0+1-i_0+N} \\ & + \sum_{i,t_0+2-i_0+N\le t\le T} \sum_i x_i(t). \end{aligned}$$

  Therefore the result cost c_f takes the following form:

$$c_f = c_{opt} + u_{i_0}(t_0) > c_{opt}.$$

So Case 2 is also a contradiction. In summary if there is a delay control, the resulted objective function value is always worse than or equal to the optimal solution. ■

Remarks:

- When there is no capacity constraint, a feasible solution with a delay control vector $u > 0$ is not optimal.
- From the proof we can also see when we add a delay control to any feasible solution, the result cost will be increased.
- The solution in this particular case can be computed in polynomial time. The problem of minimizing the aircraft count is reduced to a backward (from cell N_i to cell 1) flow saturation procedure based on:

$$x_{i+1}(t+1) = x_i(t).$$

Proof of Theorem (1):

Proof: With the second remark of Theorem 1, we have seen that adding a delay control $u > 0$ to a feasible solution x increases the objective function value. Thus, under the sector capacity constraint, the optimal solution should correspond to the minimal delay control u. x^{opt} is the optimal solution of the CTM(L) problem and x^{bfs} is the solution given by the Backward Flow Saturation (BFS). Suppose our solution x^{bfs} is not optimal, then the control delay $u^{opt} < u^{bfs}$.

With the proof by contradiction, let us assume that $\exists t, i, s.t. u_i^{opt}(t) < u_i^{bfs}(t)$. We consider

$$t_m = inf\{0 \le t \le T-1 | \exists\ i\ 0 \le i \le N\ u_i^{opt}(t) < u_i^{bfs}(t)\},$$

$$i_m = sup\{0 \le i \le N | u_i^{opt}(t_m) \le u_i^{bfs}(t_m)\},$$

then $u_{i_m}^{bfs}(t_m)$ must be positive.

If $i_m = N$, we have $u_N^{bfs}(t) = 0$, the control count of $u_{i_m}^{opt}(t_m)$ can not be fewer than the BFS solution $u_{i_m}^{bfs}(t_m)$ at cell N.

Assuming that $i_m < N$, we notice that for $t < t_m$ we have $u_i^{opt}(t) = u_i^{bfs}(t)$ and by using boundary conditions and induction:

$$x_i^{opt}(t) = x_i^{bfs}(t), \qquad \text{for} \quad i \in \{1, ..., N\} \tag{7}$$

we can extend (7) to $t = t_m$ using the relation:

$$x_k(t_m) = x_{k-1}(t_m - 1) - u_{k-1}(t_m - 1) + u_k(t_m - 1), \quad k = 1...N-1. \tag{8}$$

Thus we get for time t before t_m

$$x_i^{opt}(t) = x_i^{bfs}(t) \qquad \text{for} \quad i \in \{1, ..., N\},\ t \le t_m. \tag{9}$$

To determine $u_{i_m}^{bfs}(t_m)$, we consider the relation:

$$x_{i_m+1}^{bfs}(t_m + 1) = x_{i_m}^{bfs}(t_m) - u_{i_m}^{bfs}(t_m) + u_{i_m+1}^{bfs}(t_m), \tag{10}$$

where $x_{i_m}^{bfs}(t_m) = x_{i_m}^{opt}(t_m)$, according to (9), $u_{i_m+1}^{bfs}(t_m) = u_{i_m+1}^{opt}(t_m)$, and the definition of i_m. In Section III.A we explain that if $x_{i_m}^{bfs}(t_m) + u_{i_{max}+1}^{bfs}(t_{min})$ is greater than the capacity ν of the sector containing cell $i_m + 1$ then $u_{i_m}^{bfs}(t_m) = x_{i_m}^{bfs}(t_m) + u_{i_m+1}^{bfs}(t_m) - \nu$. Otherwise, $u_{i_m}^{bfs}(t_m)$ is set to zero.

For case $x_{i_m}^{bfs}(t_m) + u_{i_m+1}^{bfs}(t_m) \le \nu$ we have

$$u_{i_m}^{bfs}(t_m) = 0 = u_{i_m}^{opt}(t_m),$$

which contradicts with the definition of i_m and t_m.

For case $x_{i_m}^{bfs}(t_m) + u_{i_m+1}^{bfs}(t_m) > \nu$, since $u_{i_m}^{bfs}(t_m) > u_{i_m}^{opt}(t_m)$, applying relation (10) to solution x^{opt} gives:

$$\begin{aligned} x_{i_m+1}^{opt}(t_m+1) &= x_{i_m}^{opt}(t_m) + u_{i_m+1}^{opt}(t_m) - u_{i_m}^{opt}(t_m) \\ &= x_{i_m}^{bfs}(t_m) + u_{i_m+1}^{bfs}(t_m) - u_{i_m}^{opt}(t_m) \\ &> x_{i_m}^{bfs}(t_m) + u_{i_m+1}^{bfs}(t_m) - u_{i_m}^{bfs}(t_m) \\ &> x_{i_m}^{bfs}(t_m) + u_{i_m+1}^{bfs}(t_m) \\ &\quad -(x_{i_m}^{bfs}(t_m) + u_{i_m+1}^{bfs}(t_m) - \nu). \end{aligned}$$

Thus, we obtain:

$$x_{i_m+1}^{opt}(t_m + 1) > \nu.$$

The optimal solution violates the capacity constraint at cell $i_m + 1$ at $t = t_m + 1$ which is the contradiction with the feasibility of x^{opt}.

In summary, the BFS delay control $u^{bfs} \le u^{opt}$, which means the BFS solution is optimal. ■

D. Sequential Backward Flow Saturation

Now we consider $M > 1$. To solve a M-path problem, we sequentially examine the paths $1, ..., M$, construct the state and delay control vectors and update sectors capacities. The process is the following:

$$\begin{aligned} X_p^0 &= argmin\ c' X_p, \\ & X_p \in \mathbb{Z}^{2T \sum_{i=1}^{M} N_i} \\ & AX_p = f \\ & MX_p \le \nu_p \end{aligned}$$

where $p \in \{1, ..., M\}$, ν_p is the residual sector capacities after constructing the paths $1, ...p-1$. Initially we have $\nu_1 = \nu$. The vector X_p corresponds to the unknown state and control vectors of path p, so the other coordinates are set to zero.

$$X_p = [0, ..., 0, ..., \underbrace{x_p, u_p}_{p-th\ path}, 0, ..., 0...]'.$$

If the path sequence did not effect the optimality of the final solution, the SBFS would give the optimal solution. However, the default sequence $1, ..., M$ or other arbitrary path sequences can not provide optimal solution in general. The main reason is because even if in the objective function (5) the different paths have the same cost c_i, they are generally subject to different capacity constraints. In the following section a method based on the Branch and Bound technique is developed to explore the tree of the possible path sequences and find the optimal sequence which will be combined with the SBFS to provide the optimal and integral solution.

IV. Find the Optimal Path Sequence

A. The Branch and Bound method

Branch and Bound (B&B) is an algorithm that is often implemented for finding the optimal solutions in integer optimization problems [9]–[11]. In general, the B&B method is used when the domain of possible candidates is too large.

The B&B method proceeds as follows: at any point during the solution process, the status of the solution is described by a pool of unexplored subset [12]. Initially only one subset exists, namely the complete solution space, and the best solution value is set to be $+\infty$. The unexplored subspaces are represented as nodes in a dynamically maintained search tree, which initially only contains the root, and each iteration of a classical B&B algorithm processes one such node. The root node corresponds to the original problem to be solved, and each other node corresponds to a subproblem of the original problem.

More precisely, let us consider the following combinatorial problem developed in [13]:

$$\begin{aligned} & min f(x) \\ \text{s.t.} \quad & x \in X \end{aligned} \tag{11}$$

where X is finite and $f : X \longrightarrow \mathbb{R}$. The set X can be described as the leaves of a decision tree. Given a node Q of the tree, the children of Q are subproblems derived from Q through imposing a single new constraint for each subproblem. For each node s in the decision tree we define the conditional cost v:

$$v(s) = min \ f(s')$$
$$s' \text{ is a child node of } s$$

Computing the conditional cost is as complicated as the original problem (11). Particularly, when s is the root node the conditional cost coincides with the original objective function. Consequently, we simplify the optimization problem by introducing a lower bound $b(s)$ for the conditional cost at node s:

$$b(s) \leq v(s).$$

1) The algorithm: Starting from the root, the algorithm explores the children in the decision tree. Let m be the minimal cost found and initialize m with $+\infty$. When an internal node s is explored for the first time, the bound $b(s)$ is calculated. If $b(s) \geq m$, we do not explore the child nodes of s as the cost generated from s is not better than the cost of the best feasible solution found. Hence, we go back to the parent node of s and select another child node. If $b(s) < m$, it is possible that a leaf node originated from s contains a feasible solution improving m: in this case we continue the search by going to an unexplored child node of s. In terms of tree traversal we use a depth-first search. When a leaf node x is reached, the cost $f(x)$ is calculated: if $f(x) < m$, the best feasible solution found is x, we thus update m as $f(x)$. We continue the search by going back to the parent node of the current leaf.

The algorithm visits every leaf at most once. The worst case is when the bound b does not eliminate any eligible node: the algorithm consists of the enumeration of all the feasible nodes of (11).

2) The lower bound: The bounding function is the key component of any B&B algorithm. Ideally the value of a bounding function for a given subproblem should be equal to the value of the best feasible solution of the problem, but since obtaining this value is usually NP-hard, the goal is to approach the solution using only a limited amount of computational effort (i.e. in polynomial time). However, the more time spent on calculating the bound, the better the bound value usually is. It is normally considered beneficial to use as strong of a bounding function as possible in order to keep the size of the search tree small.

Normally there are two standard ways of converting the NP-hard problem of solving a subproblem into a P problem of determining a lower bound for the objective function. One of them is to use relaxation which consists of leaving out some of the constraints of the original problem and thereby enlarging the set of the feasible solutions. The optimal solution of the relaxed subproblem provides a smaller lower bound for the original function because it is performed over a larger set of values. The other way is to maintain the feasible region and modify the objective function ensuring that for all feasible solutions of the modified function have values less than or equal to the original function.

In order to find the optimal path sequence for SBFS, we develop our B&B algorithm with a lower bound computed with a variant of the first method. The feasible region is enlarged by stretching the capacity constraints for some paths in $1, ..., M$.

B. The optimal path sequence for SBFS

The B&B tree is a representation of all possible permutations of the path set $\{1, ..., M\}$. It contains $M!$ leaf nodes. An internal node located at depth k ($1 \leq k \leq M$) arranges k paths from the set $\{1, ..., M\}$. Thus, at depth k we have $M(M-1)...(M-(k-1))$ nodes, where each node is the first k paths of a possible order $p_1, ..., p_M$ and the node can be represented by $(p_1, ..., p_k)$.

Notice that the conditional cost at an internal node $(p_1, ..., p_s)$ is:

$$\begin{aligned} & v(p_1, ..., p_s) = \textstyle\sum_{i=1}^{s} \sum_{k=0}^{N_{p_i}} \sum_{t=0}^{T} x_k^{p_i}(t)^{bfs} \\ & \qquad + \min \textstyle\sum_{p \in S} \sum_{k,t} x_k^p(t), \\ \text{s.t.} \quad & (1), (2), \text{ and } (3) \\ & Mx \leq \nu_s \end{aligned}$$

where $S = \{1, ..., M\} \backslash \{p_1, ..., p_s\}$. The paths $p_1, ..., p_s$ are constructed using the SBFS algorithm demonstrated in Section III. The vector of residual capacities ν_s represents the new sector capacity constraints imposed on the unbuilt paths. We propose the following lower bound b:

$$b(p_1, ..., p_s) = \sum_{i=1}^{s} \sum_{k=0}^{N_{p_i}} \sum_{t=0}^{T} x_k^{p_i}(t)^{bfs} + \sum_{p \in S} \sum_{k=0}^{N_{p_i}} \sum_{t=0}^{T} x_k^{p}(t)^{bfs}.$$

For a single path $p \in S$, the solution $[x^p, u^p]$ is computed using Backward Flow Saturation with the updated capacity

vector ν_s. As in Section III we write the corresponding minimization problem:

$$\begin{array}{ll} & \min \sum_{k,t} x_k^p(t) \\ \text{s.t.} & (1),(2), \text{ and } (3) \\ & M * [0,...,0,.....,\underbrace{x^p,u^p}_{\text{path } p},0,...,0.....] \leq \nu_s \end{array}$$

Theorem 2: For any internal node $(p_1,...,p_s)$, the Sequential Backward Flow Saturation gives the optimal solution.

Proof:

To prove the theorem, it is equal to prove the inequality:

$$\begin{array}{ll} & \min \sum_{p\in S}\sum_{k,t} x_k^p(t) \;\geq\; \sum_{p\in S}\sum_{k=0}^{N_{p_i}}\sum_{t=0}^{T} x_k^p(t)^{bfs} \\ \text{s.t.} & (1),(2), \text{ and } (3), \\ & Mx \leq \nu_s. \end{array} \tag{12}$$

For $p \in S$, the state vector $x_k^p(t)^{bfs}$ is the BFS solution of the CTM(L) problem with only one path p, the vector of capacities is set to ν_s. Hence, the BFS solution $x_k^p(t)^{bfs}$ is also the solution of the minimization problem:

$$\begin{array}{ll} & \min \sum_{k,t} x_k^p(t) \\ \text{s.t.} & (1),(2), \text{ and } (3) \\ & M * [0,...,0,.....,\underbrace{x^p,u^p}_{\text{path } p},0,...,0.....] \leq \nu_s \end{array}$$

Thus $x_k^p(t)^{bfs}$ is the solution of an optimization problem with larger constraints than in the term left in (12). We get the following result:

$$\begin{array}{ll} \min \sum_{p\in S}\sum_{k,t} x_k^p(t) \;\geq\; \sum_{p\in S} \min \sum_{k,t} x_k^p(t). \\ s.t.\ (1),\ (2),\ \text{and}\ (3) \quad s.t.\ (1),\ (2),\ \text{and}\ (3) \\ Mx \leq \nu_s \qquad\qquad M * [0,...,0,...,\underbrace{x^p,u^p}_{\text{path } p},0,...,0] \leq \nu_s \end{array}$$

And finally the expression:

$$\begin{array}{ll} & \min \sum_{p\in S}\sum_{k,t} x_k^p(t) \;\geq\; \sum_{p\in S}\sum_{k=0}^{N_{p_i}}\sum_{t=0}^{T} x_k^p(t)^{bfs} \\ \text{s.t.} & (1),(2), \text{ and } (3) \\ & Mx \leq \nu_s \end{array}$$

In summary, the Sequential Backward Flow Saturation gives the optimal solution for any internal node in the B&B search tree. ■

Since we can achieve the optimal solution for every internal node, the B&B tree exploring can provide the optimal path sequence for the SBFS and together they can find the optimal solution for the problem.

C. A small size example

We apply the B&B algorithm with the lower bound developed previously to the model represented in Fig. 1. The total planning time is $T = 4$. The capacity count is 1 for all sectors and at any time t. We consider that the initial aircraft count is set to zero for all cells.

The tree exploring is shown in Fig. 2. We start from node (1). The branching step begins with choosing a child node of (1), for example the node $(1,2)$. Since we have not obtained a feasible point yet, $m = +\infty$ and $b(1,2) < m$ is automatically satisfied. Thus, we do not calculate $b(1,2)$. When the node $(1,2,3)$, which is a leaf node, is attained, we have a feasible

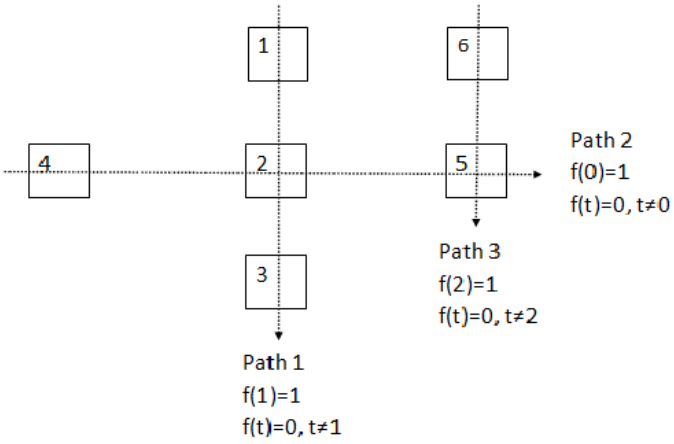

Fig. 1. Small scale model example.

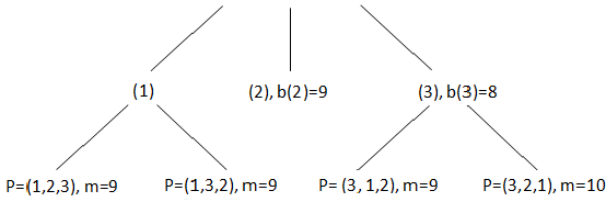

Fig. 2. The decision tree corresponding to the model of Fig. 1.

point $(1,2,3)$ and a cost $m = 9$. The parent node is revisited and we explore another leaf node $(1,3,2)$ which has a cost of 9 equal to m. The tree traversal is carried on by visiting node (2): the lower bound is $b(2) = 9$. Notice that $b(2) = m$, thus we cut the sub-tree starting from (2). The next node is (3) with a cost of $b(3) = 8 < m$. The first descendant of node (3) is explored: $(3,1)$ leads to the leaf node $(3,1,2)$ with the cost $m = 9$. We go to the second descendant of (3): $(3,2)$ and the last leaf node $(3,2,1)$ which corresponds to the cost $10 > m$. The tree exploring generates $(1,2,3)$ as the optimal path order and an optimal cost of $m = 9$.

V. Conclusion

In this paper, an integer programming problem based on CTM(L) [6] is introduced. To guarantee the optimal and integral solution, the Sequential Backward Flow Saturation is developed path by path with a given path sequence. The algorithm is performed to successively construct the traffic flows and to determine the vectors of aircraft counts and delay control volumes. Then it is noticed that the path sequence is critical to search the optimal solution. The SBFS under an arbitrary path order can not provide the optimal solution. Therefore an algorithm based on the Branch and Bound technique is developed in order to find the optimal path sequence for the SBFS. Together, the integration of SBFS and B&B guarantee the optimal solution. Moreover, the authors provide a complete proof for the optimality of this integrated Optimal Sequential Backward Flow Saturation paradigm.

ACKNOWLEDGEMENTS

This material is based upon work supported by the National Science Foundation under Grant No. 1035532. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] A. Odoni, "The flow management problem in air traffic control," in *Flow Control of Congested Networks*. Berlin, Germany: Springer Verlag, 1987.

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

[3] K. Lindsay, E. Boyd, and R. Burlingame, "Traffic flow management modeling with the time assignment model," *Air Traffic Control Quarterly*, vol. 1, no. 3, pp. 255–276, 1993.

[4] B. Sridhar, G. Chatterji, S. Grabbe, and K. Sheth, "Integration of traffic flow management decisions," in *AIAA Conference on Guidance, Navigation, and Control Conference and Exhibit*, Monterey, CA, August 2002.

[5] A. M. Bayen, R. Raffard, and C. J. Tomlin, "Eulerian network model of air traffic flow in congested areas," in *Proceedings of the American Control Conference*, 2004.

[6] D. Sun and A. Bayen, "Multicommodity eulerian-lagrangian large-capacity cell transmission model for en route traffic," *AIAA Journal of Guidance, Control, and Dynamics*, vol. 31, no. 2, pp. 616–628, 2008.

[7] P. Menon, G. Sweriduk, and K. Bilimoria, "A new approach to modeling, analysis and control of air traffic flow," in *Proceedings of AIAA Guidance, Navigation and Control Conference*, Monterey, CA, 2002.

[8] P. Wei and D. Sun, "Total unimodularity and degeneracy-aware dantzig-wolfe decomposition for large-capacity cell transmission model," in *the 50th IEEE Conference on Decision and Control and European Control Conference*, Orlando, FL, Dec 2011.

[9] *Mathematical Programming*, 1988, vol. 42, no. 1-3, ch. Simulation Tool for the Performance of Parallel Branch and Bound Algorithms.

[10] J. Clausen, *Branch and bound algorithms principles and examples*. department of comp sc., university of Copenhagen, 1999.

[11] J. Clausen and M. Perregaard, "Large quadratic assignment problems in parallel," *Computational Optimization and Applications*, vol. 8, p. 111127, 1994.

[12] B. Gendron and T. G. Crainic, "Parallel branch-and-bound algorithms: Survey and synthesis," *Operations Research*, vol. 42, pp. 1042–1066, 1994.

[13] S. Gaubert and F. Bonnans, *Recherche Operationnelle: aspects mathmatiques et applications*. Ecole Polytechnique, 2009.