The Traffic Flow Management Rerouting Problem in Air Traffic Control: A Dynamic Network Flow Approach

DIMITRIS BERTSIMAS

Sloan School of Management and Operations Research Center, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139

SARAH STOCK PATTERSON

The Fuqua School of Business, Duke University, Durham, North Carolina 27708

We address the problem of determining how to reroute aircraft in the air traffic control system when faced with dynamically changing weather conditions. The overall objective of this problem is the minimization of delay costs. This problem is of primary concern in the European air traffic control system and in particular regions within the US air traffic control system. We present an integrated mathematical programming approach that consists of several methodologies. To address the high dimensionality, we begin by presenting an aggregate model, in which the problem is formulated as a dynamic, multicommodity, integer network flow problem with certain side constraints. Using Lagrangian relaxation, we generate aggregate flows. We decompose the aggregate flows into a collection of flight paths for individual aircraft using a randomized rounding heuristic. This collection of paths is then used in a packing integer programming formulation, the solution of which generates feasible and near-optimal routes for individual flights. The overall Lagrangian Generation Algorithm is used to solve real problems in the southwestern portion of United States. In computational experiments, the solutions returned by our algorithm are within 1% of the corresponding lower bounds.

In the United States, the control of air traffic is centered on 22 regional control centers. These centers receive information from aircraft and ground-based radars on location, altitude, and speed of aircraft, as well as weather information. When the weather conditions are poor, the capacities of some airports and sectors in the National Air Space are forced to drop significantly or even to become zero. Aircraft must then fly alternative routes if they were scheduled to pass through airspace regions of reduced capacity (see Figure 1 for an example).

Currently, the Air Traffic Command Center (ATCC) initiates an iterative process with the Airline Operations Centers (AOC) to reschedule and reroute flights so that the delay costs caused by the weather conditions are kept to a minimum. The ATCC contacts each airline's AOC concerning the necessity of rerouting. Each AOC then determines a set of new flight paths that it would like to use to complete its scheduled flights given the new limited capacity scenario information. This collaborative decision making approach is based on two central tenets as expressed on the website of the Federal Aviation Administration (FAA). First, better information will lead to better decision making and second, tools and procedures need to be in place to enable the ATCC and the National Air Space users to more easily respond to the changing conditions. The FAA further states that the attempt to minimize the effects of the reduced capacity requires the up-to-date information exchange between both the airline and FAA. The impacts of these collaborative decision making concepts have been explored by several authors including ADAMS et al. (1997), JENNY (1997) and MACDONALD (1998). However, there is, as yet, no formal optimization model used

Transportation Science, © 2000 INFORMS
Vol. 34, No. 3, August 2000

0041-1655/00/3403-0239 $05.00
pp. 239–255, 1526-5447 electronic ISSN

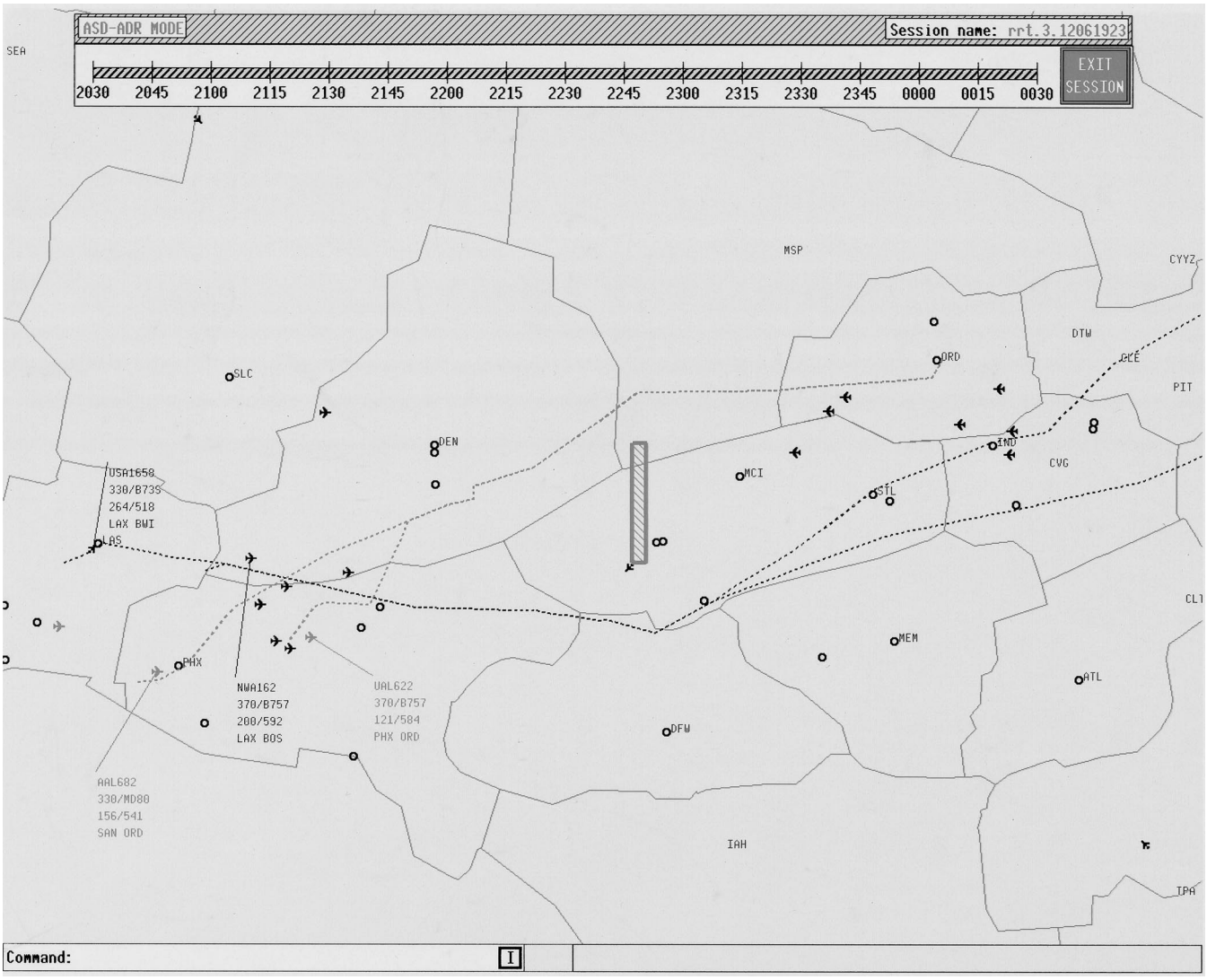

Fig. 1. Alternative routes taken as flights avoid a low capacity region.

at any stage of this collaborative decision making process.

This paper presents a model that we envision being used as a tool for addressing the problem faced by the ATCC and the AOCs of changing conditions, which we have termed the traffic flow management rerouting problem (TFMRP). The problem is to determine how to optimally control aircraft by rerouting, delaying, or adjusting the speed of the aircraft in the air traffic control system to avoid airspace regions that have reduced capacities, primarily due to dynamically changing weather conditions. The overall objective is the minimization of delay costs that include all applicable fuel costs, safety costs, and taxes.

The issue of airspace congestion and, consequently, the possibility of rerouting aircraft, is of considerable concern to the European air traffic control system. In particular, congestion is found predominantly in the airspace rather than at the airports. The European Organization for the Safety of Air Navigation (EUROCONTROL) has projected that air traffic in Europe will grow by a factor of 1.4 by 2002, and has maintained that safety needs to be sustained at the current level. Recent studies performed by the EUROCONTROL Experimental Centre showed that a 5% traffic increase results in a 26% increase in delays. These figures suggest that alleviating delays caused by airspace congestion is, and will continue to be, critical to the operation of the European air traffic control system.

In recent years, there has been considerable research activity concerning the management of air traffic flow using mathematical programming techniques. Earlier work has focused on (a) controlling release times of aircraft in the network (ground-

holding) in a single airport setting (TERRAB and ODONI, 1991; RICHETTA and ODONI, 1993, 1994), and in a multiple airport setting, in which delays propagate through the network (TERRAB and PAULOSE, 1993, VRANAS, BERTSIMAS, and ODONI 1994a, b, ANDREATTA and TIDONA, 1994, BERTSIMAS and STOCK PATTERSON, 1998, ANDREATTA and BRUNETTA, 1998, BRUNETTA, GUASTALLA, and NAVAZIO 1996, and (b) controlling release times and speed adjustments of aircraft while airborne for a network of airports taking into account the capacitated airspace (Bertsimas and Stock Patterson, 1998; HELME, 1994; LINDSAY, BOYD, and BURLINGAME, 1993). For a discussion of the various contributions and a taxonomy of the various problems, see Bertsimas and Stock Patterson (1998) and Andreatta and Brunetta (1998).

The problem of dynamically rerouting aircraft has not been addressed to the best of our knowledge in the literature. We propose an integrated mathematical programming approach that consists of several methodologies and that determines how to adjust the release times of flights into the network, control flight speed once they are airborne, and reroute flights. To address the high dimensionality, we begin by presenting an aggregate model, in which the problem is formulated as a dynamic, multicommodity, integer network flow problem with side constraints. Using Lagrangian relaxation, we generate aggregate flows that are decomposed into a collection of flight paths for individual aircraft using a randomized rounding heuristic. This collection of paths is then used in a packing integer programming formulation, the solution of which generates feasible and near-optimal routes for individual flights. The overall algorithm, termed the Lagrangian Generation Algorithm, is unique from most other Lagrangian techniques in that it is combined with a randomized rounding heuristic. We solve problems with real data in the southwest region of the United States in very short computational times using the Lagrangian Generation Algorithm.

The backbone of our current approach is the dynamic network flow formulation. FORD and FULKERSON (1958) first introduced a dynamic maximum flow problem as a standard network generalized to include traversal times between nodes. For a thorough review of work done on dynamic network flows, see the survey papers of ARONSON (1989), BOOKBINDER and SETHI (1980), and POWELL, JAILLET, and ODONI (1995). These advances are not directly relevant for our problem because our formulation is both multicommodity and integer and involves complicating side constraints.

The paper is structured as follows. In Section 1, we formally introduce the TFMRP for a single airline and present our formulation as a dynamic, multicommodity, integer network flow problem with side constraints. In Section 2, we discuss the multiple airline problem. In Section 3, we describe the Lagrangian Generation Algorithm. In Section 4, we report computational results for the TFMRP based on real data. In Section 5, we include some concluding remarks.

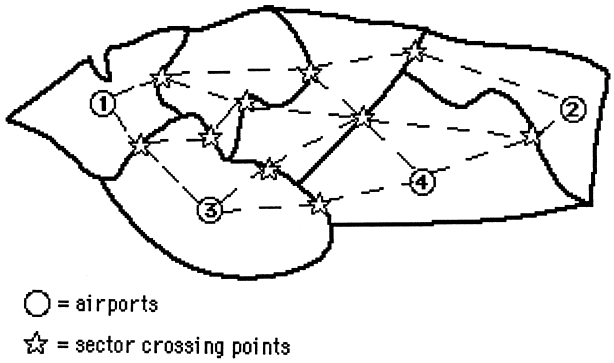

Fig. 2. A network corresponding to four airports and six sectors.

1. THE DYNAMIC MULTICOMMODITY NETWORK FLOW FORMULATION

IN THIS SECTION, we present an integer, multicommodity dynamic network flow model of the TFMRP for a single airline. There are several components to the model. These include the dynamic network, the aggregated flow variables, the non-aggregated variables, and the capacity constraints. We will describe each of these in detail below.

We first describe the dynamic network that models the air traffic system. We create a graph that represents the actual geographical picture of the airport/airspace system. The nodes of the graph represent both airports and sectors. The example in Figure 2 of four airports and six sectors demonstrates how the nodes and arcs of the network are constructed.

The outlined regions depict the sectors, and the stars depict the entrance and exit points of the sectors. We define one node for each sector crossing point. We assume that each sector has a limited number of entrance and exit points. The circles depict the airports. Each airport is represented by four nodes as described below. The arcs connect the entrance and exit points of a sector as well as the sector crossing points and airports. Each arc (i, j) has a corresponding travel time, $t_{i,j}$. To represent delay in the network, we also include self-loops, i.e.,

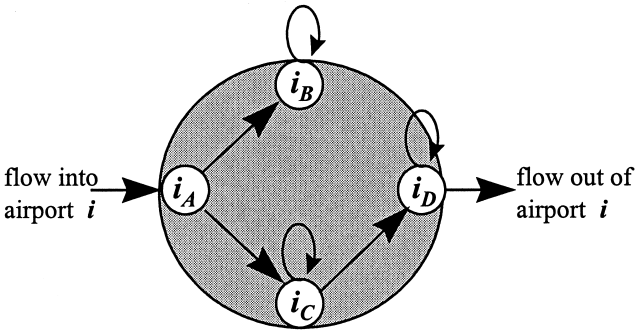

Fig. 3. An airport is modeled with four nodes.

arcs that originate and end at the same node, which have a travel time equal to one.

The commodities in the network are defined as origin–destination pairs of airports. So, if there are A airports and flights between all airports are flown, then there are exactly $A(A - 1)$ commodities. However, if we wanted to distinguish between certain characteristics of flights such as airline or aircraft type, we could do this by breaking the commodities down even further. We will discuss the multiple airline problem in Section 2.

To model airport i, we use four nodes i_A, i_B, i_C, and i_D (see Figure 3). All the incoming flights to airport i first land at node i_A. Each flight can either proceed to node i_B or to node i_C. Node i_B represents the situation in which an aircraft has completed all of its required flights for the time frame under consideration. Consequently, the flow into node i_B is removed from the network for the remaining time. Node i_C represents the situation in which an arriving aircraft must perform at least one more flight during the time frame under consideration. Given that a commodity of a flight is defined by its origin and destination, any given incoming flight will necessarily have a different commodity than any outgoing flight at the same airport. Thus, at node i_C, flow corresponding to arriving flights of a given commodity is balanced with the flow of departing flights of a different commodity. The delay arc at node i_C models the situation in which an aircraft arrives before the continued flight is scheduled to depart. Flow then proceeds from node i_C to node i_D. At node i_D, new aircraft are introduced to the network and all the flights departing from the airport leave from this node. The delay arc at this node represents ground holding of flights.

Let $\mathcal{N} = (\mathcal{S}, \mathcal{E})$ be the network formed from airports and sectors, as described above. The set of commodities is denoted by $\{1, \ldots, A(A - 1)\}$, where A is the number of airports. We discretize the time horizon into a set of time periods, $\mathcal{T} = \{1, \ldots, T\}$. We refer to any particular time period t as the "time t." Note that by "flight," we mean a flight leg between two airports. Throughout this paper we will refer to "continued" flights. A flight is continued if it relies on an aircraft that has just completed a previous flight.

The problem input data are given as follows.

$t_{i,j}$ = minimum travel time along arc (i, j),
$C_i(t)$ = capacity of sector i at time t,
$r(i)$ = turnaround time required to refuel, reload, and clean an aircraft at airport i,
orig(k) = airport of origin for a commodity k flight,
dest(k) = airport of destination for a commodity k flight,
$N(k)$ = set of arcs that a commodity k flight can use,
$\mathcal{F}$ = set of flights,
d_f = scheduled departure time of flight $f \in \mathcal{F}$,
c^{g} = cost of holding a flight on the ground for one unit of time,
c^{a} = average cost of flying an aircraft for one unit of time,
H = total amount of scheduled flying time for all flights,
$\mathcal{C}$ = set of flights that are continued,
T_f = set of feasible departure times for flight f where $f \in \mathcal{C}$,
$k(f)$ = commodity of flight f where $f \in \mathcal{F}$,
$k'(f)$ = commodity of the flight that precedes flight f where $f \in \mathcal{C}$,
$\mathrm{Sup}_k(t)$ = number of flights of commodity k that are scheduled to depart at time t, that are not continued,
$\mathrm{Dem}_k(t)$ = number of flights of commodity k that are scheduled to land at time t, at the latest, and do not continue to a later flight this day.

To reduce the dimensionality of the problem, we aggregate some of the variables over flights. The aggregated variables are defined as:

$x_{i,j}^k(t)$ = number of flights of commodity k that depart from node i at time t and arrive at node j at time $t + t_{ij}$.

Note that these variables are flow variables and not flight variables. So, to recommend flight paths, we must have a method for disaggregating, i.e., for converting these flow variables to flight variables. In Section 3.2, we propose a method for performing the disaggregation through a randomization scheme.

For continued flights, we also introduce non-aggregated flight variables as

$$y_f(t) = \begin{cases} 1, & \text{if the aircraft performing flight } f \in \mathcal{C} \\ & \text{is ready for departure at time } t \in T_f, \\ 0, & \text{otherwise.} \end{cases}$$

We use these non-aggregated flight variables to deal with the transfer of an aircraft from one flight to another. In particular, these variables are used to ensure that the necessary transfer of commodities occurs at the flight level for continued flights. Note that each continued flight is assigned a unique commodity to precede it instead of a unique flight. For each continued flight, $f \in \mathcal{C}$, we know the commodity of flight f, denoted by $k(f)$ and the commodity of the corresponding flight that will precede it, denoted $k'(f)$. So, instead of forcing flight $f \in \mathcal{C}$ to be a continuation of flight f', we ensure that, for every continued flight, there must be aircraft of commodity $k'(f)$ available for flight $f \in \mathcal{C}$ to use.

Using these non-aggregated variables, we create the additional constraints that specify that there must be an aircraft available for each flight $f \in \mathcal{C}$ at some time. In other words, $y_f(t)$ must be equal to 1 for some time in the set of feasible departure times. There are other methods of ensuring this transfer, but they involve adding a large number of side constraints to the formulation. Whereas, by using these variables, only $|\mathcal{C}|$ additional constraints will be required. Without these variables and the associated adjustment to the constraints, continued flights could depart before their scheduled departure times and could use any available aircraft to perform the flight. Note that the set of feasible departure times, T_f for $f \in \mathcal{C}$, begins at the scheduled departure time for flight f and ends several hours later.

Our model does not explicitly allow flights to be cancelled. However, if the optimal schedule contains a flight with a very high delay cost resulting from a very late departure, a long rerouted flight path or both, the airline could opt to cancel this flight. At that point, the schedule could be accepted as is or the model could be rerun to see if further improvements could be attained by reallocating the resources previously used by the now cancelled flight.

The objective of the TFMRP is to minimize the total delay cost of flying all the required flights. Any flight may experience delay resulting from ground holding, decreasing speed while in the air, and selecting a route that is longer than the scheduled route. Moreover, a continued flight may also experience delay if there is no aircraft available for use at its departure time.

The objective function can be written as

$$\sum_{\{k,\,t,\,i=\mathrm{orig}(k)\}} c^{\mathrm{g}} x^k_{i_D,\,i_D}(t) + \sum_{\substack{\{(k,\,t,\,f):f\in\mathcal{C},\\ k(f)=k,\,t\in T_f\}}} (t - d_f)\, y_f(t) + \sum_{\{k,\,i,\,t\}} c^{\mathrm{a}} x^k_{i,i}(t) + \sum_{\{k,\,t,\,(i,j)\in N(k)\}} c^{\mathrm{a}} t_{i,j} x^k_{i,j}(t) - c^{\mathrm{a}} H. \quad (1)$$

The first term represents the cost of ground holding delay. The second term represents the cost of delay incurred by continued flights that were unable to depart on time because there were no available aircraft. The third term represents the cost of air delay due to speed reduction. The fourth term gives the total actual cost of all air travel, and the fifth term is simply a constant representing the total cost of all scheduled air travel. Note that the cost of delay caused by rerouting is obtained when the total cost of all scheduled air travel is subtracted from the total actual cost of air travel.

The constraints are given below.

TFMRP

$$\sum_{\{j:(i,j)\in N(k)\}} x^k_{i,j}(t) - \sum_{\{j:(j,i)\in N(k)\}} x^k_{j,i}(t - t_{j,i}) = 0, \quad \forall i \in \mathcal{S},\ k,\ t, \quad (2)$$

$$\sum_{\{j:(j,\,i_A)\in N(k)\}} x^k_{j,i_A}(t - t_{j,i_A}) - x^k_{i_A,\,i_B}(t) - x^k_{i_A,\,i_C}(t) = 0, \quad \forall k,\ t,\ i = \mathrm{dest}(k), \quad (3)$$

$$x^k_{i_A,\,i_B}(t) + x^k_{i_B,\,i_B}(t-1) - x^k_{i_B,\,i_B}(t) = \mathrm{Dem}_k(t), \quad \forall k,\ t,\ i = \mathrm{dest}(k), \quad (4)$$

$$\sum_{\{f\in\mathcal{C}:k=k'(f)\}} y_f(t) + x^k_{i_C,\,i_C}(t) - x^k_{i_C,\,i_C}(t-1) - x^k_{i_A,\,i_C}(t - r(i)) = 0, \quad \forall k,\ t,\ i = \mathrm{dest}(k), \quad (5)$$

$$\sum_{\{j:(i_D,\,j)\in N(k)\}} x^k_{i_D,\,j}(t) - \sum_{\{f\in\mathcal{C}:k=k(f)\}} y_f(t) - x^k_{i_D,\,i_D}(t-1) = \mathrm{Sup}_k(t), \quad \forall k,\ t,\ i = \mathrm{orig}(k), \quad (6)$$

$$\sum_{k} \sum_{\{j:(i,j)\in N(k)\}} \sum_{\{t':\,t - t_{i,j} < t' \leqslant t\}} x^k_{i,j}(t') \leqslant C_i(t), \quad \forall i,\ t, \quad (7)$$

$$\sum_{\{t\in T_f\}} y_f(t) = 1, \quad \forall f \in \mathcal{C}, \quad (8)$$

$$x^k_{i,j}(t) \geqslant 0, \text{ integer} \quad \forall i, j, k, t, \quad (9)$$

$$y_f(t) \in \{0, 1\}, \quad \forall f, t. \quad (10)$$

Constraint 2 represents dynamic flow conservation for the sectors. There is a constraint for each sector node, $i \in \mathcal{S}$, commodity k, and time t.

The next four constraints represent flow conservation at each of the airport nodes i_A, i_B, i_C, and i_D. Constraint 3 forces flow conservation at node i_A of airport i. At node i_A, we sum over all the nodes that can arrive at airport i from some sector j of commodity k, $\Sigma_{\{j:(j,i_A)\in N(k)\}} x^k_{j,i_A}(t - t_{j,i_A})$. The time index is $t - t_{j,i_A}$ because this is the time that the flow leaves j if it is to arrive at i at time t. This flow must be equal to the flow out of node i_A at time t. This flow goes to either node i_B (where it will be removed from the network) or to node i_C (where it will be transferred to another commodity).

Constraint 4 forces flow conservation at node i_B of airport i. The flow into node i_B at time t equals the flow from node i_A, $x^k_{i_A,i_B}(t)$, plus any flow that is held on the ground from the previous time period, $x^k_{i_B,i_B}(t - 1)$. This must equal to the flow out, which is $\mathrm{Dem}_k(t)$ plus $x^k_{i_B,i_B}(t)$. In other words, if flow arrives at node i_B from node i_A prior to the latest time that it is scheduled to arrive, it lands and then is held on the ground at no cost until the time that the flow can be removed with a non-zero value of $\mathrm{Dem}_k(t)$. Thus, every aircraft is removed from the network at its scheduled latest arrival time.

Constraint 5 forces flow conservation at node i_C. The flow into this node is of commodity k and the flow out of this node is a different commodity. The flow into node i_C only comes from node i_A, giving the term, $\Sigma_{\{k:i=\mathrm{dest}(k)\}} x^k_{i_A,i_C}(t - r(i))$, where $r(i)$ is the turnaround time at airport i. This is the time necessary to refuel and otherwise prepare the aircraft for the next flight. There is a delay arc at node i_C that captures those aircraft that arrive at airport i and are cleaned and refueled before they are needed to fly the next flight. There is no cost for using this delay arc, it simply represents an aircraft waiting at an airport for its next flight. Finally, the flow out of node i_C is given by $\Sigma_{\{f:k=k'(f)\}} y_f(t)$. This captures all the flights that continue from commodity k flights and that may depart at time t.

Constraint 6 forces flow conservation at node i_D. At this node, all the flow leaving i_D to some node j minus the flow into i_D must equal the supply at airport i at time t for commodity k such that $i = \mathrm{orig}(k)$, which is denoted by $\mathrm{Sup}_k(t)$. The flow leaving node i_D is simply given by $\Sigma_{\{j:(i_D,j)\in N(k)\}} x^k_{i_D,j}(t)$. This summation includes ground holding when $j = i_D$. The flow into node i_D is either from node i_C or from ground holding in the previous time period. In the former case, the flow from i_C comes from all the continued flights, which are of commodity $k(f) = k$. The ground holding value from the previous time period is given by $x^k_{i_D,i_D}(t - 1)$.

Constraint 7 captures the capacity restrictions. There is a capacity on the number of aircraft that can be within sector i at time t given by $C_i(t)$. To represent this in terms of the flow variables, we need to sum over all commodities and all arcs that represent travel in sector i of commodity k at time t.

Constraint 8 forces every continued flight to depart at some time, t, within the set of feasible departure times, T_f. The remaining two constraint sets specify that the $x^k_{i,j}(t)$ variables are nonnegative integers, and that the $y_f(t)$ variables are binary.

The above formulation of the TFMRP is a multicommodity, integer variation of the minimum cost dynamic network flow problem. There are some important differences. First, the capacity constraint 7 is bundled over commodities, arcs, and time periods, not just over commodities. Second, the disaggregated variables $y_f(t)$, are not flow variables, and finally there are additional side constraints 8.

2. MODELING THE MULTIPLE AIRLINE PROBLEM

THE MODEL PROPOSED in the previous section can be used to solve the rerouting problem for a single airline. However, if we took the viewpoint of the FAA in which several airlines are occupying the airspace at the same time, then we need to modify the formulation slightly. The reason that the multiple airline problem is not the same concerns the continued flights.

In the formulation of Section 1, we stipulate that a continued flight does not rely on a unique flight, rather the formulation guarantees that an aircraft of the correct commodity will be available before the continued flight can depart. Once we introduce multiple airlines, we need to make sure that not only is the incoming aircraft of the correct commodity, but also that it belongs to the correct airline. For instance, obviously an American Airlines aircraft could not continue a Delta Airlines flight.

To do this, we will need to redefine the commodities such that there is a unique commodity distinguished by each origin–destination pair and by each airline. By redefining the commodities in this manner, we can now ensure that a continued flight will rely on an aircraft whose commodity corresponds to the correct airline. This would increase the number of commodities by a multiple equal to the number of airlines.

We further need to consider the issue of fairness. It may be globally optimal to assign all the delay to a single airline, but of course, this solution is not acceptable. Thus, we would need to ensure that the

delay is allocated in a fair manner across the airlines. This could be accomplished by modifying the packing formulation, discussed in Section 3.3, by adding constraints that guarantee that each airline receive no more than a given percentage of the total delay.

3. THE LAGRANGIAN GENERATION ALGORITHM

IN THIS SECTION, we use Lagrangian relaxation of the formulation **TFMRP**, randomized rounding, and a packing formulation to propose near-optimal solutions for the TFMRP. The overall algorithm is outlined below. We then explain each step in detail.

The motivation for solving the LP relaxation using Lagrangian techniques is to quickly obtain many non-integral solutions which, when used as input to the randomized rounding heuristic outlined below, generate a large set of potential flight routes.

The Lagrangian Generation Algorithm

1. **Lagrangian relaxation of the LP.** Starting with the formulation of **TFMRP**, i.e., the problem of minimizing Eq. 1 subject to the constraints 2–10, we relax the capacity constraints 7 into the objective function with multipliers, λ. We further relax the integrality constraints in 9 and 10, thus solving the relaxed problem as a linear program. Note that the optimal cost of the Lagrangian problem is equal to the cost of the linear programming relaxation of **TFMRP**. We initialize the lower bound by solving the linear programming relaxation of the formulation for the TFMRP. The initial upper bound is infinity.
2. **Solution of the relaxed problem.** We solve the relaxed problem and obtain a potentially fractional solution $y_f(t)$, $x^k_{i,j}(t)$.
3. **Randomized rounding.** We randomly round the variables $y_f(t)$ to zero–one solutions, and randomly decompose the flow into routes for flights. These routes are then added to a list of paths.
4. **Packing formulation.** We formulate and solve an integer packing problem, in which we are attempting to pack the elements of the list of paths into the capacitated airspace system. If a new solution is found, we update the upper bound.
5. **Stopping criterion.** If the upper and lower bound are within a desired accuracy ϵ, we stop.
6. **Update of multipliers.** We update the multipliers λ and go to Step 2.

In the following subsections, we describe each of the steps of the algorithm.

3.1 Lagrangian Techniques

During Step 2 of the algorithm, we solve an uncapacitated multicommodity dynamic network flow problem as a linear program. Using the network flow solver of CPLEX, we solve the problem quickly (see Section 4). The main motivation of this step is to generate attractive routes for flights.

We update the multipliers using the iterative approach of EVERETT (1963) as follows. We represent the capacity constraints 7 as $Ax \leq b$. Let a_j be the jth row of the matrix A, and let b_j be the right-hand side value for this row. Let x^k be the vector of solutions at iteration k. Let λ^k_j be the Lagrange multiplier for the jth constraint at iteration k, which is determined using the following rule.

$$\text{If} \quad a_j' x^k > b_j \quad \text{then} \quad \lambda^{k+1}_j = (1 + \delta^k_j)\lambda^k_j.$$

$$\text{If} \quad a_j' x^k \leq b_j \quad \text{then} \quad \lambda^{k+1}_j = (1 - \delta^k_j)\lambda^k_j.$$

The parameters δ^k_j are updated using the following rule.

$$\text{If} \quad (a_j' x^k - b_j)(a_j' x^{k-1} - b_j) > 0, \quad \text{then} \quad \delta^{k+1}_j = \epsilon_1 \delta^k_j.$$

$$\text{If} \quad (a_j' x^k - b_j)(a_j' x^{k-1} - b_j) < 0, \quad \text{then} \quad \delta^{k+1}_j = \epsilon_2 \delta^k_j.$$

$$\text{If} \quad (a_j' x^k - b_j)(a_j' x^{k-1} - b_j) = 0, \quad \text{then} \quad \delta^{k+1}_j = \delta^k_j.$$

The values of ϵ_1 and ϵ_2 are fixed parameters where $\epsilon_1 > 1$ and $\epsilon_2 < 1$.

The motivation for this method is as follows. If $a_j' x > b_j$, then the solution x uses too much of the available amount of the jth resource; thus, we increase the Lagrange multiplier to penalize the violation more. In this method, the Lagrange multiplier would be increased by a factor of $(1 + \delta^k_j)$. Likewise, if $a_j' x \leq b_j$ then the solution x uses a feasible amount of the jth resource; thus, we decrease the Lagrange multiplier. It is decreased by the factor $(1 - \delta^k_j)$ as shown above. The amount of increase or decrease at each iteration is determined by δ^k_j, called the step size, which is controlled at each iteration.

The values of δ^k_j are updated in the following manner. If a constraint is not satisfied iteration after iteration, then the step size is gradually increased based on the assumption that the value of λ^k_j may still be quite far from its optimal value. If the constraint fluctuates between feasibility and infeasibility, then the step size is reduced substantially based on the assumption that λ^k_j has come close to its optimal value. It is interesting to note that updating the Lagrange multipliers depends only upon whether or not the constraint was satisfied, not on the magnitude of the difference.

3.2 Randomized Rounding Heuristic

The objective of this step is to generate a rich set of paths for individual flights from the aggregated flow solutions. The motivation for using randomization is to generate a broader set of solutions. After completing the Lagrangian relaxation step of the Lagrangian Generation Algorithm, we have a potentially fractional solution $y_f(t)$, $x^k_{i,j}(t)$.

Basically, this heuristic randomly walks through the network for every flight looking for a positive flow path. Starting at the departure airport of flight f and at the departure time of flight f, the heuristic randomly picks the next arc that has a positive flow on it. If this turns out to be a self-loop, then it remains at the airport for another time period and then makes another random decision about where to move in the next time period. Perhaps the next step places it at a new sector after it has completed the travel time for that arc. It then, once again, will randomly pick another arc that has some positive flow on it. It will walk through the network in this manner until it reaches its destination. With this in mind, we will now describe the process in full detail.

We will create a list of paths as follows:

$$\mathbf{P} = \{p_1, \ldots, p_N\},$$

where $p_i = \{(s_i(0), t_i(0)), \ldots, (s_i(n_i), t_i(n_i))\}$, where $s_i(m)$ is the mth element of path p_i, and $t_i(m)$ is the time that the flight arrives at $s_i(m)$, $m = 0, \ldots, n_i$. We define N as the total number of paths in the list and $n_i + 1$ as the total number of elements in path p_i.

We select the first element in each path $s_i(0)$ in a deterministic manner. For every noncontinued flight, we can create a path in which $s_i(0)$ is equal to node i_D of the departure airport and $t_i(0)$ is equal to the scheduled departure time. For every continued flight, we select the earliest time such that $y_f(t)$ is non-zero, and set this equal to one. Now we can create a path for each continued flight where $s_i(0)$ is equal to node i_D of the departure airport, and $t_i(0)$ is equal to the time at which $y_f(t)$ is equal to one.

To build the rest of each path, we step through the network for every flight beginning at the node given by $s_i(0)$ at the time $t_i(0)$. We will refer to the commodity of our flight as commodity k. Next, we randomly select from all arcs emanating from $s_i(0)$ that have a positive flow value, i.e., $x^k_{s_i(0),\hat{j}}(t_i(0)) > 0$. This may include the possibility of selecting the arc that represents ground delay. Let the arc that we randomly select be denoted by $(s_i(0), \hat{j})$. We set

$$s_i(1) = \hat{j} \quad \text{and} \quad t_i(1) = t_i(0) + t_{s_i(0),\hat{j}}.$$

We then need to decrease the flow value on the variable, $x^k_{s_i(0),\hat{j}}(t_i(0))$, by one.

We continue in this manner until we reach the node representing node i_A of the destination airport. Because these flows respect the flow conservation constraints in the Lagrangian relaxation, there will always be flow out of a node that the heuristic reaches.

There are a number of ways that we could set the probabilities used to select paths. Currently, we simply assign an equal probability to each node that has a positive flow. The rationale for this is simply to place a higher probability on obtaining alternative paths. Another possible method of randomizing would be to assign each arc a probability based on the flow on the arc. In particular, we could assign a probability P_j to arc (a, j) defined by

$$P_j = \frac{x^k_{a,j}(t)}{\sum_{\{j:(a,j)\in N(k)\}} x^k_{a,j}(t)}.$$

We could then use these probabilities to determine which arc to select at each step.

After this heuristic is completed, we have a set of path and time specifications that are added to a list of paths and used as input data for the integer programming packing problem. Note that the heuristic only produces paths that satisfy flow conservation constraints. In particular, the capacity constraints 7, and the airport constraints that handle continued flights 5, may not be satisfied. The path and time specifications will satisfy constraints 2, 3, 4, 6, 8, 9, and 10. By not forcing these paths to satisfy all the constraints in **TFMRP**, we create, after a few iterations, a list of paths that has more flexibility. The integer programming packing formulation will then select from this list a combination of paths that will satisfy all the constraints in **TFMRP**.

3.3 The Integer Programming Packing Formulation

The goal of the packing problem is to pack the paths generated by the randomized rounding heuristic into the air traffic system, so that all the necessary flights occur at correct (though not necessarily on time) times, and so that the capacity constraints are satisfied.

A path, $p_i = (s_i(0), t_i(0)), \ldots, (s_i(n_i), t_i(n_i))$, specifies the elements and the times of a given route. The elements are both airports and sectors. Obviously the first element, $s_i(0)$ is the departure airport and the last element $s_i(n_i)$ is the arrival airport. However, ground holding is represented in these paths. So, if path p_i includes g units of ground hold-

ing, then the elements $s_i(m)$, $m = 0, \ldots, g-1$, all represent the departure airport, and the times $t_i(m)$, $m = 0, \ldots, g-1$, are each separated by one time unit. Thus, time $t_i(0)$ is not the actual departure time, rather it is the time at which an aircraft becomes available to perform the flight represented by path p_i. If this is a flight that is not continued, then $t_i(0)$ will be the same as the scheduled departure time. If the path represents a flight that is continued, then $t_i(0)$ may be later than the scheduled departure time, because it is possible that there was no aircraft available for a continued flight to use at its departure time.

The decision variables in the packing formulation are

$$z_{f,i} = \begin{cases} 1, & \text{if path/time pair } p_i \\ & \text{is used to fly flight } f \in \mathcal{F}, \\ 0, & \text{otherwise.} \end{cases}$$

Let Z be the set of feasible combinations of path/time pairs and flights.

$$Z = \{(f, i): f \in \mathcal{C},\ p_i(0) = \text{orig}(k(f)),\ p_i(n_i) = \text{dest}(k(f)),\ t_i(0) \geq d_f\}$$
$$\cup \{(f, i): f \in \mathcal{F}\backslash\mathcal{C},\ p_i(0) = \text{orig}(k(f)),\ p_i(n_i) = \text{dest}(k(f)),\ t_i(0) = d_f\}.$$

The objective of the packing problem is to minimize the cost of delay in the air and the ground holding cost of departing after the scheduled departure time. Let g once again be the number of ground holding units associated with path p_i. Then, the delay cost of path p_i is given by

$$c_i = c^a[t_i(n_i) - t_i(g)] + c^g[t_i(g-1) - t_i(0)],$$

which includes both the amount of time spent ground holding and flying.

The objective function is as follows:

$$\sum_{i=1}^{N} c_i \sum_{\{f:(f,i)\in Z\}} z_{f,i} + c^g\left[\sum_{i=1}^{N} t_i(0) \sum_{\{f:(f,i)\in Z\}} z_{f,i} - \sum_{\{f\in\mathcal{F}\}} d_f\right] - c^a H.$$

The first term captures the cost resulting from ground holding delay and from air travel. When combined with the final term, which is a fixed cost of scheduled air travel, we capture the cost of delay resulting from ground holding, decreasing speed while in the air, and selecting a route that is longer than the scheduled route. The only remaining delay occurs if there is no aircraft available for a continued flight to use at its departure time. To capture this delay, we sum over all the times at which aircraft become available to perform the flight represented by path p_i and subtract the sum of all the scheduled departure times.

The constraint set is given by the following equations.

PP

$$\sum_{\substack{\{(f,i)\in Z:\exists m | j = s_i(m),\\ t_i(m)\leq t < t_i(m+1)\}}} z_{f,i} \leq C_j(t), \quad \forall j \in \mathcal{S},\ t, \tag{11}$$

$$\sum_{\{i:(f,i)\in Z\}} z_{f,i} = 1, \quad \forall f \in \mathcal{F}, \tag{12}$$

$$\sum_{\substack{\{(f,i)\in Z: f\in\mathcal{C},\\ k'(f)=k,\ t_i(0)\leq t\}}} z_{f,i} \leq \sum_{\substack{\{(f',i')\in Z: k(f')=k,\\ t_{i'}(n_{i'})+r(s_{i'}(n_{i'}))\leq t\}}} z_{f',i'}, \quad \forall k,\ t, \tag{13}$$

$$z_{f,i} \in \{0, 1\}, \quad \forall i = 1, \ldots, N.$$

Constraints 11 represent the capacity constraints. They stipulate that, for every sector j and every time t, the sum over all the flights that are in this sector at time t must be less than or equal to the sector capacity. A flight is within a sector at time t if it entered the sector before time t, $t_i(m) \leq t$, and has not yet entered the next sector in its path before time t, $t < t_i(m+1)$. Constraints 12 ensure that each flight will be assigned to exactly one route. Constraints 13 guarantee that a continued flight will not depart before a suitable aircraft has arrived for it to use. The left-hand side of constraints 13 represents the number of continued flights whose preceding flight is of commodity k, $k'(f) = k$, and whose possible departure time is less than or equal to t. This number must be less than the number of flights of commodity k, $k(f') = k$, that arrive before time t minus the turnaround time, $r(s_{i'}(n_{i'}))$. The final constraint set simply forces these variables to be binary.

4. COMPUTATIONAL RESULTS FOR THE LAGRANGIAN GENERATION ALGORITHM

IN THIS SECTION, we report on the computational performance of the Lagrangian Generation Algorithm. In Sections 4.1, 4.2, and 4.3, the Lagrangian Generation Algorithm is applied to solve three instances of the air traffic flow management rerouting problem. These instances model different weather fronts passing through a portion of southwestern United States. This region consists of four airports, located at Denver (DEN), Phoenix (PHX), Las Vegas (LAS), and Salt Lake City (SLC). There are 42 sec-

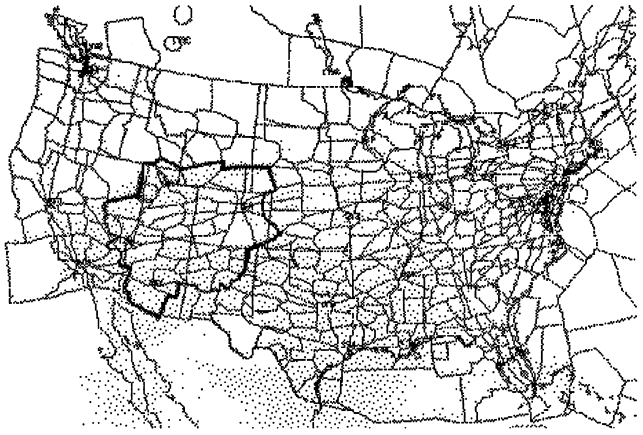

Fig. 4. Sector map of the US with southwest region shown.

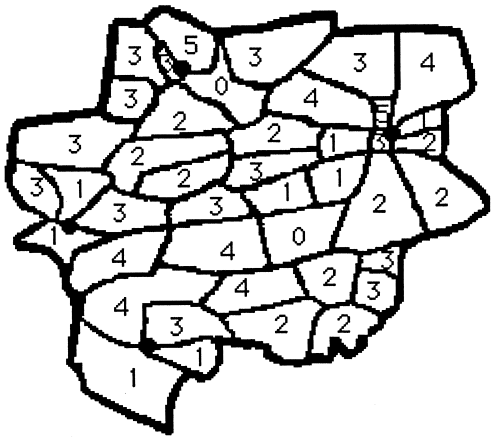

Fig. 5. Weather Scenario I at 8 A.M. representing normal operating conditions before the weather front has hit this region.

tors that lie in the vicinity of these four airports as shown in Figure 4. This data, provided by the FAA, includes the travel times for the sectors and the necessary turnaround times. Each of the three instances has a different capacity scenario determined by the movement of a weather front through the region. All of the computation were performed on a Sun Sparc 20 workstation with 48MB.

4.1 Computations for Weather Scenario I

For this instance, flight schedules for 71 flights among the four airports shown in Figure 4 are extracted from a dataset provided by the FAA that covers an 8-hour time frame with 5-minute time intervals. We simulated a weather front passing from the northeast corner of this region to the southwest corner. The sector capacities were generated probabilistically according to a uniform distribution.

During normal weather conditions, the sector capacities were generated using a uniform distribution with a mean determined by the size of the sector and a standard deviation of one. Figure 5 shows the sector capacities during normal weather conditions. At the cusp of the storm, the capacities were generated using a uniform distribution with a mean of zero. All of the resulting negative values were set to zero. As the weather front gradually passes through, the mean slowly increases up to the mean for normal weather conditions. Figure 6 shows the different weather scenarios that evolve over time for this capacity scenario as the weather front passes through the region. The shaded areas show the cusp of weather front with the numbers corresponding to sector capacities.

To achieve a lower bound on the solution, we solve the LP relaxation of the multicommodity dynamic network formulation, which consists of 24,509 constraints and 61,912 variables. Finding a solution to the LP requires 181 seconds. The solution is highly non-integral with an objective value of 2498.5.

We ran the Lagrangian Generation Algorithm setting the parameter values as $\epsilon_1 = 2$ and $\epsilon_2 = 0.33$. The starting values for λ_j^0 and δ_j^0 were set to 10 and 0.8, respectively. We did not perform extensive trials to determine the best starting values for λ and δ, however, a few different settings were tried. The results presented reflect the stated starting conditions, which converged in the shortest number of iterations. The Lagrangian relaxation solves a network problem with 15,279 nodes, 54,427 arcs and 26 of the side constraints 8. The size of the integer programming packing problem grows at each iteration as the size of the list of paths increases. At the final iteration, the formulation consists of 2209 constraints and 145 variables, which reduces to 424 constraints and 125 variables by using some presolving routines in CPLEX. Table I tracks the performance of the Lagrangian Generation Algorithm as it steps through the algorithm.

The total amount of time needed to solve this problem, including the subproblem times for the Lagrangian relaxation and the integer programming packing problem given in Table I, was 330 seconds. The solution value found, 2509, is within 0.4% of the lower bound. The total delay associated with this solution is composed of 810 minutes of ground delay, 15 minutes of airholding delay, 200 minutes of rerouting delay, and 215 minutes of delay caused by late incoming aircraft for continued flights.

Numerous routes were used for each particular commodity. As an example, we will consider one of these commodities, Las Vegas to Phoenix, in detail. Three different routes were used over the course of the day to fly flights of this commodity. Two routes

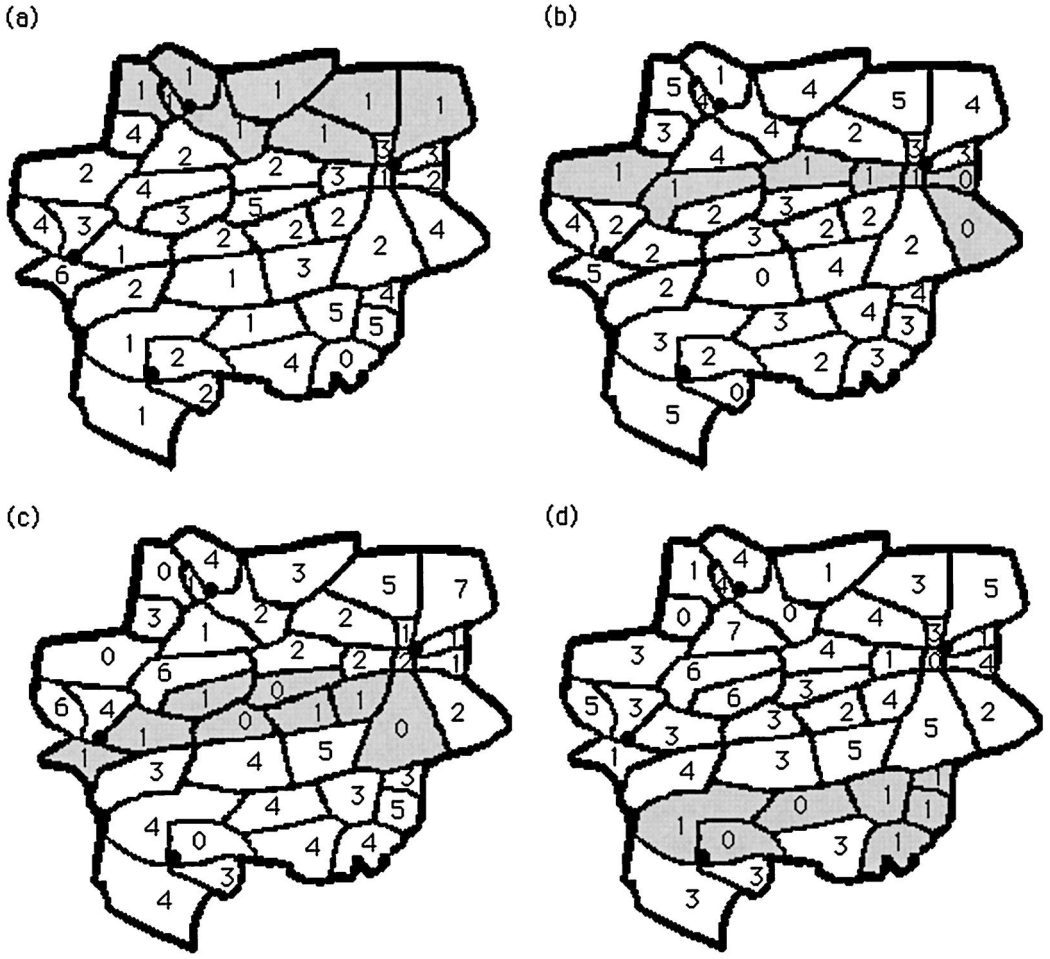

Fig. 6. Weather Scenario I at: (*a*) 9:45 A.M. (*b*) 10:35 A.M. (*c*) 11:25 A.M. (*d*) 12:15 P.M.

are shown and labeled in Figure 7. Most of the time when this commodity is flown, route 1 is used and ground delay alone is assigned to avoid any anticipated capacity problems while in the air. However, we will examine two flights that use route 2. The important times for these two flights are depicted in Figure 8 and explained below.

TABLE I
Computational Results for Weather Scenario I

	Lagrangian			Packing		
Iteration	Time	Objective Value	Number Infeasible	Presolve Time	IP Time	Objective Value
0	25.01	−113062	89	0.03	–	Inf.
1	24.22	−12572	65	0.03	–	Inf.
2	24.70	1321.46	77	0.07	–	Inf.
3	24.42	2073.94	54	0.12	–	Inf.
4	24.58	2248.13	64	0.18	–	Inf.
5	24.40	2324.27	68	0.18	–	Inf.
6	24.65	2286.42	61	0.18	–	Inf.
7	24.67	2377.25	66	0.18	15.26	2509

All times in seconds.

At 11:50 A.M. a flight is scheduled to depart from Las Vegas. This flight is scheduled to arrive at Phoenix one hour later. The flight is held on the ground for 45 minutes and actually departs at 12:35 P.M. It reaches Phoenix at 2:10 P.M. making the total amount of time spent traveling equal to 1 hour and

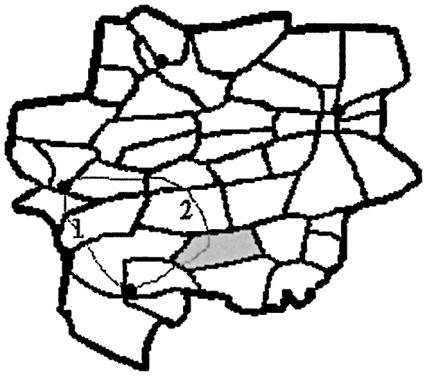

Fig. 7. Two routes used at different times to fly from Las Vegas to Phoenix.

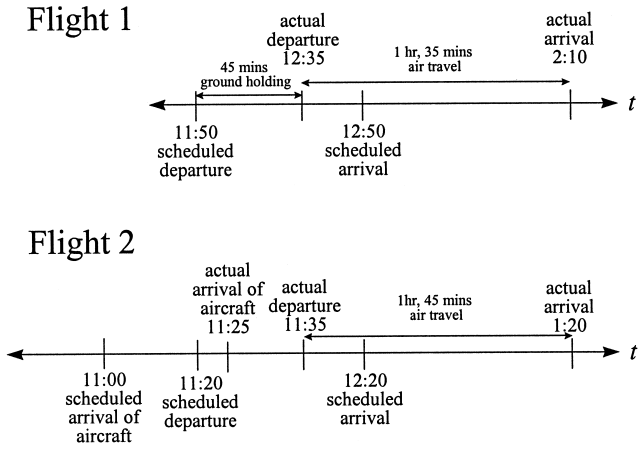

Fig. 8. Time lines of two flights that use route 2 on Figure 7.

35 minutes, which is 35 minutes more than scheduled. The total delay of 80 minutes resulted from 45 minutes of ground holding, and 35 minutes of rerouting delay.

At 11:20 A.M., a continued flight is scheduled to depart from Las Vegas. It is relying on an aircraft that is scheduled to arrive at Las Vegas at 11:00 A.M. However, this incoming flight experiences delay and does not arrive at Las Vegas until 11:25 A.M. The aircraft is immediately refueled and boarded, and the continued flight actually departs at time 11:35 A.M., 15 minutes after the scheduled departure time. There is no ground holding assigned to this flight. The flight travels along route 2 and is further delayed in terms of airspeed reduction. When the flight reaches the shaded sector in Figure 7, its speed is reduced such that the travel time through this sector is increased by 10 minutes. It reaches Phoenix at 1:20 P.M., a total of 1 hour after the scheduled arrival time. This total delay of 60 minutes resulted from 10 minutes of airspeed reduction, 35 minutes of rerouting delay, and 15 minutes of departure delay due to the late incoming aircraft.

4.2 Computations for Weather Scenario II

For this instance, the same flight schedules for the 71 flights between the 4 airports shown in Figure 4 were used. However, to simulate the weather front passing from the northeast corner of this region to the southwest corner, the sector capacities were set deterministically.

During normal weather conditions, we fixed the capacities according to the size of the sector. At the cusp of the storm, the sector capacities were set to zero. One hour later, as the storm front moves along, those sectors that had zero capacity during the last hour, now have a slightly increased capacity of one. The sector capacities would continue to increase hourly until they have resumed the level of normal weather conditions. Figure 9 shows the different sector capacities that evolve over time for the second capacity scenario as the weather front passes through the region. Figures 9*a* through *f* show the sector capacities between the times 8:20 to 9:20 A.M., 9:25 to 10:25 A.M., 10:30 to 11:30 A.M., 11:35 A.M. to 12:35 P.M., 12:40 to 1:40 P.M., and 1:45 to 2:45 P.M., respectively. The shaded areas show the cusp of the weather front where the sector capacities are zero. The front moves along gradually, spending one hour before each progression.

To achieve a lower bound on the solution, we solve the LP relaxation of the multicommodity dynamic network formulation. The size of the formulation is not affected by the change in the weather capacity scenario. Thus, the number of constraints and variables is the same as specified for weather scenario I. Solving the LP requires 59 seconds and gives a solution that is highly non-integral with an objective value of 2387.

We ran the Lagrangian Generation Algorithm setting the parameter values as $\epsilon_1 = 2$ and $\epsilon_2 = 0.33$. The starting values for λ_j^0 and δ_j^0 were set to 10 and 0.8, respectively. Table II tracks the performance of the Lagrangian Generation Algorithm as it steps through the algorithm. The problem sizes are the same as for weather scenario I.

Solving this problem took 116 seconds, which includes the subproblem times for the Lagrangian relaxation and the integer programming packing problem given in Table II. The solution value found, 2418, is within 1.2% of the lower bound. The total delay associated with this solution is composed of 495 minutes of ground delay, no airholding delay, 225 minutes of rerouting delay, and 55 minutes of delay caused by late incoming aircraft for continued flights.

We ran the algorithm for 100 iterations, without the ϵ stopping condition, to see if we could find a solution that is even better than the one found above. The results from the first 15 iterations are given in the Table III. The remaining 85 iterations did not find a solution with a lower solution value less than the value at the fifteenth iteration, and were thus not including in the table. Within 7 iterations we generate the best solution that we were able to find. This value is 2389, which is within 0.08% of the lower bound of 2387.

Numerous routes were used for each particular commodity. In Figure 10, we look at a few of the routes used to fly between Phoenix and Salt Lake City. Route 1 travels from Salt Lake City heading toward Phoenix. This flight is scheduled to depart at 9:30 A.M., but is held on the ground until 11:30

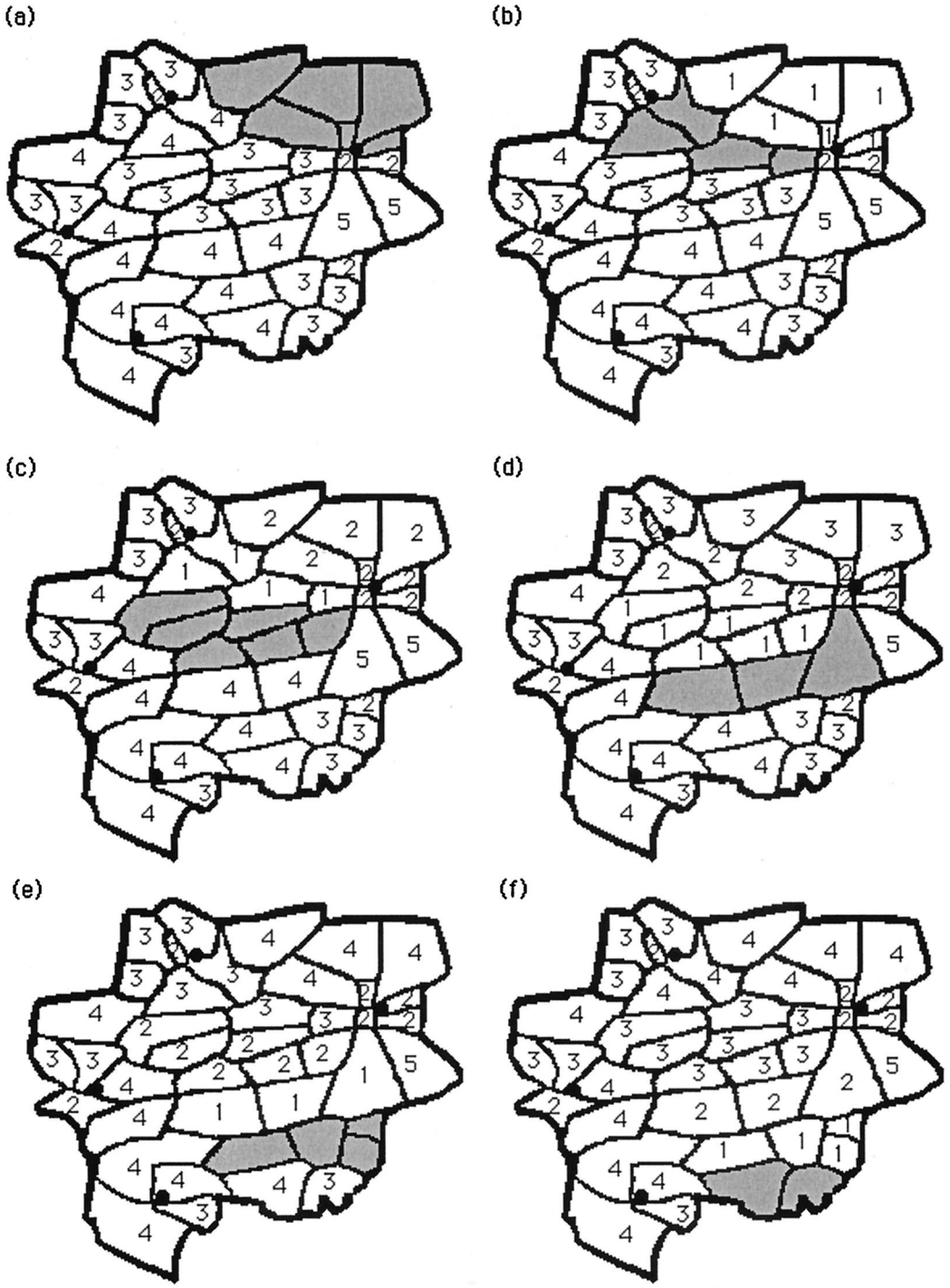

Fig. 9. Weather Scenario II: (*a*) 8:20–9:20 A.M., (*b*) 9:25–10:25 A.M., (*c*) 10:30–11:30 A.M., (*d*) 11:35 A.M.–12:35 P.M., (*e*) 12:40–1:40 P.M., (*f*) 1:45–2:45 P.M.

A.M., incurring a 2-hour ground delay. At that point, the flight departs from Salt Lake City and follows a reasonably direct route, route 1, to Phoenix, basically trailing the storm front.

Routes 2 and 3 go north from Phoenix to Salt Lake City. The flight that travels along route 2 is scheduled to depart at 9:25 A.M., but is held on the ground for 1 hour. It then departs and follows a circuitous route due to the limited sector capacities. The flight that travels along route 3 is scheduled to depart at 11:30 A.M. and only suffers a 20-minute ground delay. This flight passes through its route immedi-

TABLE II
Computational Results for Weather Scenario II

	Lagrangian			Packing		
Iteration	Time	Objective Time	Number Infeasible	Presolve Time	IP Time	Objective Time
0	24.72	−107072	72	0.12	–	Inf.
1	24.63	−11148	60	0.10	–	Inf.
2	24.72	1618.38	20	0.42	17.51	2418

All times in seconds.

ately after the weather front. The capacities are still quite limited, however, forcing this flight to significantly deviate from the shortest path. To appreciate how this affects travel time, we note that route 1 requires 1 hour and 55 minutes of flying time, route 2 requires 2 hours and 25 minutes of flying time, and route 3 requires 2 hours and 35 minutes of flying time.

TABLE III
Computational Results for Weather Scenario II for 15 Iterations

	Lagrangian			Packing		
Iteration	Time	Objective Value	Number Infeasible	Presolve Time	IP Time	Objective Value
0	24.72	−107072	72	0.12	–	Inf.
1	24.63	−11148	60	0.10	–	Inf.
2	24.72	1618.38	20	0.42	17.51	2418
3	26.18	2104.20	25	0.48	17.42	2408
4	25.98	2231.75	32	0.53	18.02	2396
5	25.98	2295.00	18	0.53	19.10	2393
6	25.98	2317.74	19	0.57	19.35	2392
7	25.43	2345.69	21	0.62	18.46	2389
8	25.88	2327.27	21	0.65	19.56	2389
9	25.82	2360.46	22	0.65	19.89	2389
10	25.88	2304.59	22	0.68	19.99	2389
11	25.78	2367.44	10	0.72	20.03	2389
12	26.55	2375.27	12	0.68	20.36	2389
13	25.93	2378.53	14	0.72	21.58	2389
14	26.45	2379.18	21	0.67	21.60	2389

All times in seconds.

Fig. 10. Three routes used to fly between Salt Lake City and Phoenix.

4.3 Computations for Weather Scenario III

For the third weather scenario that we tested, we increased the number of flights to 200 and scaled sector capacities during normal weather accordingly. To simulate the weather front passing from the northeast to the southwest corner, we set the sector capacities deterministically.

During normal weather conditions, we fixed the sector capacities according to the size of the sector. At the cusp of the storm, the sector capacities were set to zero. The front once again gradually moves along spending forty minutes, before each progression. As the cusp of the storm front moves through the region, the available sector capacity increases by two each forty minutes. So this weather front moves more quickly and does not leave such bad conditions behind it, as does the previous weather scenarios. Had we kept the capacity levels at that of either of the two previous scenarios, then the problem would have been infeasible, meaning that there would have been no way to complete all of the 200 flights during the time frame without cancelling some flights. Figure 11 shows the different weather scenarios that evolve over time for the third capacity scenario as the weather front passes through the region. The shaded areas show the cusp of the weather front in which the sector capacities are set to zero.

To achieve a lower bound on the solution, we solve the LP relaxation of the multicommodity dynamic network formulation, which consists of 25,881 constraints and 66,489 variables. Solving the LP requires 86 seconds and gives a solution that is highly non-integral with an objective value of 6513.5.

We again ran the Lagrangian Generation Algorithm with the same starting values as before. Table IV tracks the performance of the Lagrangian Generation Algorithm as it steps through the algorithm. The Lagrangian relaxation solves a network problem with 16,219 nodes, 57,294 arcs, and 138 of the side constraints 8. The size of the integer programming packing problem grows at each iteration as the size of the list of paths increases. At the final iteration, the formulation consists of 2394 constraints and 1197 variables, which reduces to 879 constraints and 1078 variables by using some presolving routines in CPLEX.

Solving this problem took 169 seconds, which includes the subproblem times for the Lagrangian relaxation and the integer programming packing problem given in Table IV. The solution value found, 6574, is within 0.92% of the lower bound. The total delay associated with this solution is composed of 670 minutes of ground delay, no airholding delay, 290 minutes of rerouting delay and 370 minutes of

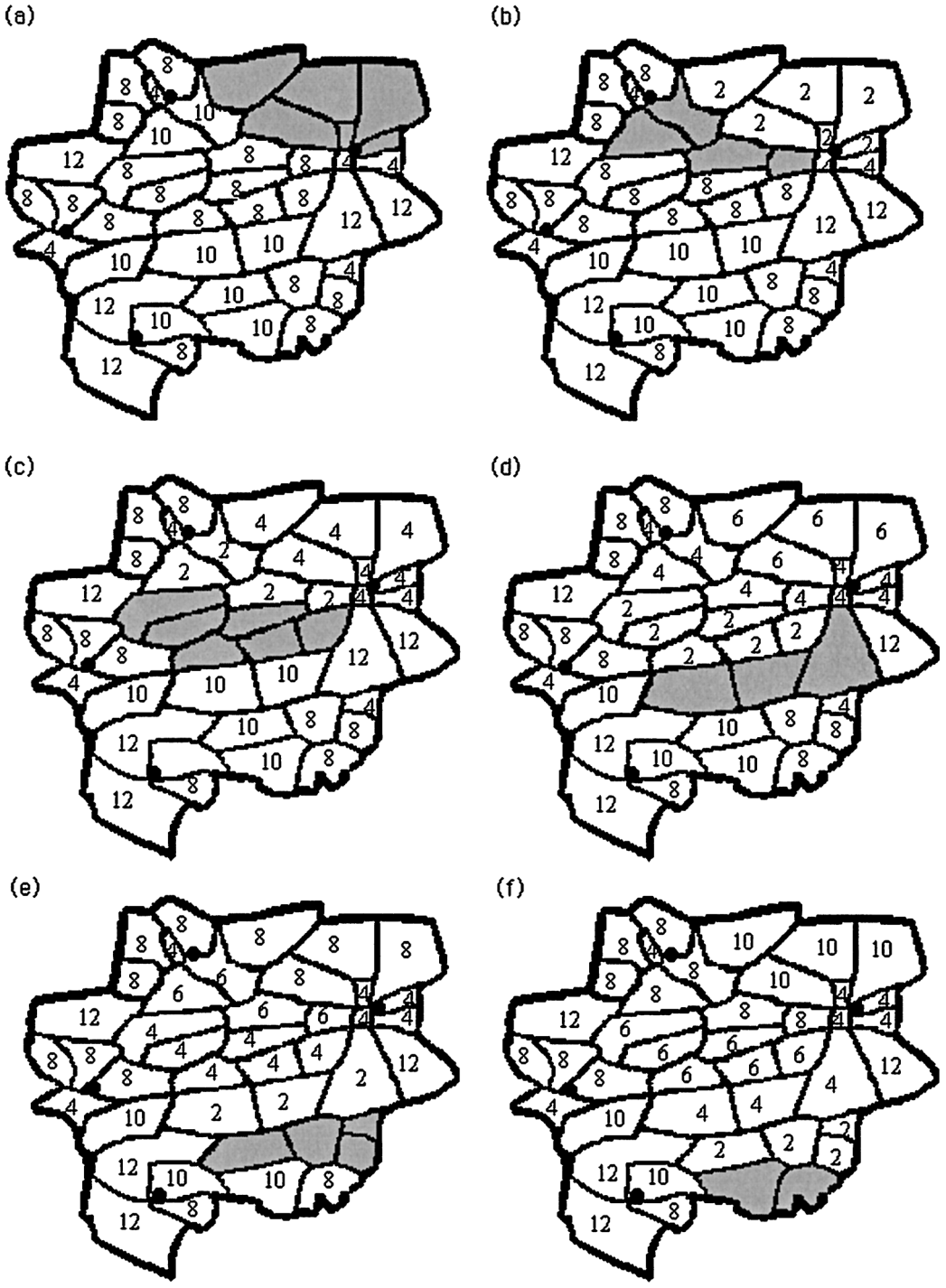

Fig. 11. Weather Scenario III for 200 flights: (*a*) 8:20–9:00 A.M., (*b*) 9:05–9:45 A.M., (*c*) 9:50–10:30 A.M., (*d*) 10:35–11:15 A.M., (*e*) 11:20 A.M.–12:00 P.M., (*f*) 12:05–12:45 P.M.

delay caused by late incoming aircraft for continued flights.

Once again, we ran this problem for more iterations to see if we could obtain a better solution, even though we already found a solution that is well within the tolerance. Table V gives results that show all the solutions that we generated. Of these, the best solution, 6520, is within 0.09% of the lower bound, 6513.5.

5. CONCLUSIONS

THE AIR TRAFFIC Flow Management Rerouting Problem determines how to reroute flights through

TABLE IV
Computational Results for 200 Flight Dataset

	Lagrangian			Packing		
Iteration	Time	Objective Value	Number Infeasible	Presolve Time	IP Time	Objective Value
0	36.52	−283953	91	0.19	–	Inf.
1	36.49	−26652	54	0.26	–	Inf.
2	36.31	5772.76	39	0.81	–	Inf.
3	36.56	6152.06	44	0.83	18.11	6574

All times in seconds.

different flight paths to reach their destinations if the current routes pass through a region that is unusable as a result of poor weather conditions. This is the first research that has taken a global look at rerouting. Our approach determines the best routes for the aircraft to follow, as well as the amount of ground holding and the amount of speed adjustment, while taking into consideration that the entire national airspace system is operating under capacity restrictions. We modeled the problem as a dynamic network flow formulation with additional constraints and presented an integrated mathematical programming approach utilizing several methodologies. The computational results suggest that this approach is capable of efficiently solving real problems for a portion of the national airspace system.

In the course of this research, we obtained some general insights that may have wider applicability. The algorithmic design of the Lagrangian Generation Algorithm could be used in other problem contexts. The idea of extracting solutions using randomization and combining these solutions using integer programming may be useful in other problems as well. Although we have presented our formulations in the context of air traffic control, we envision other applications of our models in any area in which goods are dynamically flowing through a system with several types of capacitated elements such as manufacturing, telecommunications, and ground transportation systems.

TABLE V
Computational Results for 200-Flight Dataset for 12 Iterations

	Lagrangian			Packing		
Iteration	Time	Objective Value	Number Infeasible	Presolve Time	IP Time	Objective Value
0	36.10	−283953	91	0.19	–	Inf.
1	36.16	−26652	54	0.26	–	Inf.
2	36.14	5772.76	39	0.81	–	Inf.
3	36.26	6152.06	44	0.83	18.11	6574
4	36.08	6291.80	42	0.92	18.78	6545
5	36.03	6327.60	39	0.94	18.63	6529
6	35.97	6307.54	42	1.00	19.00	6527
7	35.83	6370.08	37	1.04	19.12	6523
8	36.51	6453.82	36	1.08	19.58	6520
9	36.26	6458.95	47	1.09	19.64	6520
10	35.90	6319.08	41	1.10	20.01	6520
11	36.22	6380.18	36	1.10	20.82	6520

All times given in seconds.

ACKNOWLEDGMENT

THIS RESEARCH WAS supported in part by grants from Draper Laboratory and National Aeronautics and Space Administration. We are grateful to the referees for their careful reading of the paper and for their thoughtful discussion and comments.

REFERENCES

M. ADAMS, S. KOLITZ, J. MILNER, AND A. ODONI, "Evolutionary concepts for Decentralized Air Traffic Flow Management," *Air Traffic Control Quart.* **4,** 281–306 (1997).

G. ANDREATTA AND L. BRUNETTA, "Multiairport Ground Holding Problem: A Computational Evaluation Of Exact Algorithms," *Opns. Res.* **46,** 57–64 (1998).

G. ANDREATTA, A. R. ODONI, AND O. RICHETTA, "Models for the Ground-Holding Problem," in *Large-Scale Computation and Information Processing in Air Traffic Control*, L. Bianco and A. R. Odoni (eds), 125–168, Springer-Verlag, Berlin (1993).

G. ANDREATTA AND G. TIDONA, "A New Formulation for the Multi-Airport Ground-Holding Problem," Internal Report No. 3, Department of Pure and Applied Mathematics, University of Padova, Italy, 1994.

J. E. ARONSON, "A Survey of Dynamic Network Flows," *Ann. Opns. Res.* **20,** 1–66 (1989).

D. BERTSIMAS AND S. STOCK PATTERSON, "The Air Traffic Flow Management Problem with Enroute Capacities," *Opns. Res.* **46,** 406–422 (1998).

J. H. BOOKBINDER AND S. P. SETHI, "The Dynamic Transportation Problem: A Survey," *Naval Res. Logist. Quart.* **27,** 65–88 (1980).

L. BRUNETTA, G. GUASTALLA, AND L. NAVAZIO, "A New Approach for Solving the Multi-Airport Ground Holding Problem," *Ann. Opns. Res.* **81,** 271–287 (1996).

H. EVERETT, "Generalized Lagrange Multiplier Method for Solving Problems of Optimum Allocation of Resources," *Opns. Res.* **3,** 399–417 (1963).

L. R. FORD AND D. R. FULKERSON, *Flows in Networks*, Princeton University Press, Princeton, New Jersey, 1958.

M. HELME, "Reducing Air Traffic Delay in a Space–Time Network," Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics, 236–242, 1994.

M. JENNY (ed.), "Foreword," *Air Traffic Control Quart.* **5,** 129–132 (1997).

K. S. LINDSAY, E. A. BOYD, AND R. BURLINGAME, "Traffic Flow Management Modeling with the Time Assign-

ment Model," *Air Traffic Control Quart.* **1,** 255–276 (1993).

L. MACDONALD, "Collaborative Decision Making in Aviation," *J. Air Traffic Control* **40,** 12–17 (1998).

G. NEMHAUSER AND L. WOLSEY, *Integer and Combinatorial Optimization*, John Wiley & Sons, New York, 1988.

A. R. ODONI, "The Flow Management Problem in Air Traffic Control," in *Flow Control of Congested Networks*, A. R. Odoni and G. Szego (eds), 269–288, Springer-Verlag, Berlin, 1987.

W. B. POWELL, P. JAILLET, AND A. R. ODONI, "Stochastic and Dynamic Networks and Routing," in *Handbooks in Operations Research and Management Science: Networks*, M. O. Ball, T. L. Magnanti, C. L. Monma, and G. L. Nemhauser (eds), Elsevier Science Publishers B.V., 1995.

G. E. PUGH, Value-Driven Methods of Decision and Control, draft of a textbook, 3–5, 1993.

O. RICHETTA AND A. R. ODONI, "Solving Optimally the Static Ground-Holding Policy Problem in Air Traffic Control," *Transp. Sci.* **27,** 228–238 (1993).

O. RICHETTA AND A. R. ODONI, "Dynamic Solution to the Ground-Holding Policy Problem in Air Traffic Control," *Transp. Res.* **28a,** 167–185 (1994).

M. TERRAB AND A. R. ODONI, "Strategic Flow Control on an Air Traffic Network," *Opns. Res.* **41,** 138–152 (1991).

M. TERRAB AND S. PAULOSE, Dynamic Strategic and Tactical Air Traffic Flow Control, Technical Report, Rensselaer Polytechnic Institute, Troy, NY, 1993.

P. VRANAS, D. BERTSIMAS, AND A. R. ODONI, "The Multi-Airport Ground-Holding Problem in Air Traffic Control," *Opns. Res.* **42,** 249–261 (1994a).

P. VRANAS, D. BERTSIMAS, AND A. R. ODONI, "Dynamic Ground-Holding Policies for a Network of Airports," *Transp. Sci.* **28,** 275–291 (1994b).

(Received: August 1998; revisions received: August 1999; accepted: August 1999)