Advances in the Optimization of Airline Fleet Assignment

RUSSELL A. RUSHMEIER and SPYRIDON A. KONTOGIORGIS

USAir Operations Research Group, 2345 Crystal Drive, Arlington, Virginia 22227

We present an advanced model for the formulation and solution of large scale fleet assignment problems that arise in the scheduling of air transportation. Fleet assignment determines the type of aircraft to operate each flight in a given schedule, subject to a variety of side constraints, due to marketing, operational, maintenance and crew restrictions. We model the problem as mixed-integer multicommodity flow on networks encoding activities linking flight departures. We focus on fully representing flight connection possibilities, while accurately capturing complex operational rules. We also provide a unified framework for the expression of resource constraints via piecewise linear penalties, which permits a profitability-based tradeoff between operational goals and revenue. Computational results on actual schedules show that high quality assignments for one-day problems can be obtained within an hour of computation. The use of the model at USAir results in an annual benefit of at least $15 million.

In planning its operations, an airline must solve scheduling problems that are among the largest in any industry. To produce profitable schedules in short time frames, airlines are increasingly relying on computerized decision support systems. Extensive operations research efforts, abetted by advances in computer hardware and breakthroughs in algorithms, have allowed the replacement of ad hoc heuristics for manual scheduling with automated procedures based on mathematical optimization, thus significantly reducing the turnaround time.

ETSCHMAIER and MATHAISSEL (1985) provide an overview of the airline schedule development process. In most major airlines the development of a monthly flight schedule proceeds along the following steps: 1) market planning—planning the frequency of direct and one-stop service to cities; 2) schedule design—design of the flight timetable in terms of origin, destination, and departure times, including timing of flights at hub airports; 3) fleet assignment—assignment of one of available aircraft types to each flight in the schedule; 4) aircraft routing—selection of the sequence of flights to be flown by each aircraft in the course of a day; 5) crew pairing—assignment of crews to a sequence of flights over a period of 1–4 days; 6) crew blocks—arrangement of crew pairings into monthly blocks of flying for each crew member; 7) staff scheduling—scheduling of non-crew personnel at ground facilities to support the flight schedule.

Mixed-integer programming (MIP) has been used with great success in two of these steps, fleet assignment (ABARA, 1989; HANE et al., 1994) and crew pairing (ANBIL et al., 1991, 1992). These problems are typically approached as a two-stage process: aircraft are scheduled first, with some crew issues implicitly considered, and then all aircraft decisions become fixed and the crews are scheduled. The fleet assignment process seeks to maximize profitability, while providing flyable schedules complying with a large number of marketing and operational requirements. The objective can be achieved both by an increase in expected revenue (for instance, by assigning large aircraft to flight legs with high demand) and by decreasing the expenses associated with flying, such as costs related to fuel, personnel, and maintenance. The solution technique for fleet assignment depends on whether a valid initial assignment exists or not (warm vs. cold start). Local improvement algorithms are typically used in the warm-start case; they improve the current assignment by swapping aircraft types for specific flights, while maintaining aircraft flow balance (TALLURI,
we discuss implementation issues and computational performance on actual schedules. We also discuss effective strategies that curb the size of the problem and the computational effort, while still offering a quality solution. We conclude with a discussion on extensions of the model and related work in progress.

1. THE FLEET ASSIGNMENT MODEL

The COLDSTART fleet assignment process begins once the schedule has been put into a timetable format, with fixed departure times for each flight leg. The flight time on each valid aircraft type and the standards for connect time between an aircraft’s arrival and subsequent departure are provided along with the schedule.

Current fleet assignment methods represent the schedule as a closed-loop space–time network, in which a directed flight arc corresponds to the movement of an aircraft of a particular type along a flight leg. In the conventional model (SOUﬁs, FERLAND, and ROUSSEAU, 1980, BERGE and HOPPERSTAD, 1993, Hane et al., 1994, SUBRAMANIAN et al., 1994) a node at each station is associated with the arrival or departure of a flight leg, for each valid aircraft type. Each node is linked to its successor in time by a directed sit arc, which accounts for aircraft waiting on the ground. A wraparound sit arc, from the last to the first node, accounts for the aircraft overnighting at the station. The optimization problem is formulated as mixed-integer multicommodity flow, in which commodities correspond to fleets and coupling constraints capture the coverage of flights as well as the operational requirements. It has been proven by Gu et al., (1994) that a closely related fleet assignment problem, including only flight coverage and fleet count constraints, is NP-hard.

Our new fleet assignment model (FAM) is also formulated as multicommodity flow, but on an enhanced network which represents accurately the connect time rules in their full complexity. This is combined with a general framework for the resource constraints which better captures the economic tradeoffs. The complete network and the resource constraints are expressed in the AMPL model in terms of basic operational input data.

We now proceed to describe the main features of FAM.

1.1 The Event-Activity Network

In FAM a flight leg is an ordered triple \((o, t, d)\), consisting of the originating station \(o\), the destination \(d\), and the fixed departure time \(t\). Each node in our network is an event representing the start of an
operation \((f, e)\), i.e., a flight \(f\) operated by type \(e\). Restrictions on allowable aircraft for certain flight legs are implemented by excluding the prohibited operations. Each flight activity arc is a triple \((f, g, e)\), which represents operating flight leg \(f\) with aircraft of type \(e\) and being available for connection, on the same type, to flight leg \(g\). Thus each flight activity arc links two operations which can be performed successively in time by the same aircraft. A similar approach is described in Abara (1989), where his turn arcs correspond to our flight activity arcs. We also define sit activity arcs \((f, g, e)\) at a station where all aircraft of type \(e\) available for flight \(f\) are also available for its immediate successor flight \(g\) and all later flights. These arcs between successive nodes are only included at stations where they are guaranteed to reflect the applicable connect rules.

With this representation, FAM is placed in the context of related fixed-schedule time-constrained routing problems (Desrosiers et al., 1993). The input can be thought of as a set of tasks, with fixed starting time and duration, and which can succeed each other in predefined ways, i.e., by obeying the time ordering and the connection rules. The goal is to sequence the tasks, so that each is performed exactly once. FAM is flexible enough to make no assumptions on the ordering logic of the tasks. In contrast, the other ground connect models described employ a network structure which relies on the assumption that in every station every node connects to its successor. As a result, they cannot distinguish between aircraft on the ground, and thus their scope is severely limited.

At USAir for example, much effort has focused into the proper recognition and treatment of connect possibilities, to utilize aircraft most effectively. The resulting connect time standards are a function of both the arriving and departing flight legs. In this case, arriving flights may not connect to all successor departures. Since USAir uses FAM to automate the fleeting of current schedules, as well as for studies of intermediate and long range, it must reflect these detailed timing specifications in its final assignment.

We have constructed FAM so that it can include connection rules of arbitrary complexity. However one must be careful in selecting connection possibilities, in order to keep the networks manageable in size. Abara (1989) addresses this difficulty by limiting the number of possible connections for each flight leg to a fixed number, say five. Another heuristic is to connect only to departures within a suitable time interval after an arrival. Such universal rules do not perform well on real flight networks: at busy stations a flight leg should be allowed to connect to many departures, whereas at small stations there is likely to be only one reasonable connection.

We capture the essential connectivity pattern of the schedule without resorting to such heuristics, by the following approach. We partition the set of operations at each station into connecting complexes: each complex is a balanced subset of the incoming and outgoing flight legs. We allow an incoming leg to connect only to those outgoing legs within the same complex, for which the connection standards are not violated. The concept of connecting complexes generalizes the connection “islands” (Hane et al., 1994) to cases where the ground flow model is not valid. Moreover, it naturally captures fundamental scheduling concepts such as out-and-back flights (hub-to-spoke roundtrips) and hub banks (sets of coordinated arrivals and departures, timed to maximize passenger connection possibilities). In Figure 1 we display portions of a fleet network. The PIT station illustrates the use of sit activities, while ALB exhibits connecting complexes.

In terms of modeling, we have been successful in specifying the defining properties of complexes in terms of AMPL set constructors, so that FAM fully exploits the structure of complexes in the automated generation of the fleet networks. From a performance viewpoint, our computational results in Section 3 demonstrate that the use of connecting complexes reduces substantially the solution time without sacrificing either accuracy or profit.

1.2 Activity-Based Profit

The activity formulation enables us to control and cost the flight connections at the finest level of detail necessary. For each flight leg, our model can provide
makes the model robust enough to be used in all phases of schedule development.

Other separable terms can be easily included in (1), for instance penalties for unfavorable activities, such as changing the aircraft type on a flight relative to a base schedule. A profit coefficient of the form (1) is an engineering compromise between the standard single-flight-leg-based profit, which does not consider connections, and a full origin–destination-based profitability calculation. In Section 4 we discuss model extensions to deal with more general costs that depend on a sequence of flight legs.

1.3 “Soft” Resource Constraints

In addition to maintaining aircraft flow obeying the connection rules, fleet assignments must meet resource requirements determined by the operational capabilities. These are particularly important in cases where certain costs cannot be readily included in the objective, but constraints and penalties are used instead, to avoid assignments that lead to high-cost situations downstream. Of special interest is the proper formulation of constraints which permit only good fleet assignments from a crew-pairing perspective, because of the magnitude of the financial impact. The incorporation of maintenance requirements is also crucial. For recent discussions of models for maintenance in airlines see FEO and BARD (1989), GOPALAN and TALLURI (1995), and CLARKE et al. (1994).

In the literature these maintenance requirements are expressed as linear equality and inequality constraints of a variety of forms. We have chosen instead to express them in the common framework of “soft” constraints, i.e., as penalty terms on the utilization levels of resources by sets of activities. The primary benefit is that, when the target utilization levels cannot be met, the optimization problem still returns a feasible solution. Such a solution furnishes the necessary information to iteratively adjust the schedule and the target levels to achieve a satisfactory assignment. Second, the proper costing of the deviations allows for a profit-maximization-based tradeoff between possibly conflicting requirements and expected revenue.

We illustrate the approach with a typical requirement, crew utilization. A resource variable is introduced for the hours flown for each crew group. A constraint defines the utilization level for each crew group as the sum of flight time over the set of activities flown by aircraft types in the group. The associated penalty function has the piecewise linear convex form illustrated in Figure 2. The user provides the minimum, target, and maximum crew hours,
The "soft" constraint framework is general enough to allow for the modeling of many types of resource constraints. The ones currently in FAM include:

- The number of aircraft of each type used in the fleet.
- The number of aircraft of each type overnighting at a specified group of maintenance stations.
- The number of aircraft of a specified type overnighting at a specified station.
- The number of aircraft of a specified type available on the ground for a certain time length within a time interval at a specified hangar station.
- The hours of flying assigned to each crew group.
- The minimum numbers of daily departures in corresponding aircraft types from each crew base.
- The maximum number of crew groups arriving and departing at certain stations.

Using USAir's Schedule Development Environment (SDE), the schedule planners can switch requirements on and off selectively, and also set the targets and the associated penalty slopes for each requirement. From a performance viewpoint, models with more constraints are generally harder to solve. Therefore, the decision to add constraints depends on the value of the extra precision gained versus the effort required in the solution process.

2. THE OPTIMIZATION PROBLEM

Our MIP model is based on flow through the event-activity network described in Section 1.1. The decision variables are flight activity flow

\[
 z_{(f, g, e)} = \begin{cases} 
 1 & \text{if activity } (f, g, e) \text{ is selected} \\
 0 & \text{otherwise.} 
\end{cases} 
\]  

(2)

Selecting \((f, g, e)\) is taken to mean that an aircraft of type \(e\) is assigned to flight leg \(f\) and is available for the subsequent leg \(g\). For a sit activity \((f, g, e)\) we introduce a variable \(y_{(f, g, e)}\) to measure the number of aircraft of type \(e\) on the ground between the departure of \(f\) and the departure of the subsequent leg \(g\). The \(y\) variables serve only as a bookkeeping mechanism for aircraft ground inventory and must take on integer values for any valid fleet assignment. Therefore we do not explicitly restrict them to be integer-valued. When a distinction as to the type of flow is not important, we will let \(x\) refer to either flight or sit variables.

FAM produces optimization problems with the following block angular structure

\[
 \max \quad p_1(x_1) + \ldots + p_E(x_E) - v(r) 
\]

subject to

\[
\begin{align*}
 N_0 x_1 &= 0 \\
 \ddots & \ddots \\
 N_E x_E &= 0 \\
 C_1 x_1 + \ldots + C_E x_E &= 1 \\
 U_1 x_1 + \ldots + U_E x_E - Ir &= 0 \\
 x_1, \ldots, x_E \text{ integer } &\geq 0, \quad r \geq 0.
\end{align*}
\]  

(3)

Each block of integer variables \(x_i, i = 1, \ldots, E\), in (3) consists of the arcs representing the flight and sit activities for type \(e\). The continuous variables \(r\) measure the utilization of the resources.

The objective function trades off the profit \(p\) of assigning fleets with the penalty \(v\) for violations from preplanned resource utilization targets. In order to have a computationally tractable problem and to stay within the validity limits of our data from the real business operations, we chose to approximate \(p_e(x_e)\), the profit for each fleet \(e\), by a function separable over the activities, which is zero for a sit arc and linear for a flight arc. Thus

\[
p_e(x_e) = \sum_{(f, g, e)} p_{(f, g, e)} \delta_{(f, g, e)} 
\]  

(4)

where each coefficient \(p_{(f, g, e)}\) is of the form (1).

The penalty component \(v(r)\) is of the form \(\sum_i v_i(r_i)\), where \(r_i\) is the utilization level for resource \(i\) and
each \( u_i \) is a piecewise linear convex function. AMPL translates the composite piecewise linear concave objective into an equivalent linear one, by replacing each piecewise linear term with a combination of bounded auxiliary variables. The coefficients of these variables correspond to slopes and their bounds correspond to distances between breakpoints. This type of transformation is described in more detail in Fourer and Gay (1995).

In the matrix of constraints, each block \( N_e \) is the node-arc incidence matrix for the event-activity network for type \( e \), with rows corresponding to operations and columns corresponding to activities. It captures the aircraft flow balance constraints, which are of the circulation type

\[
\sum_g x_{(f,g,e)} - \sum_h x_{(h,f,e)} = 0, \quad \forall (f,e).
\]  

(5)

We note that the networks can vary widely across the aircraft types, because of flight restrictions such as range limitations, curfews at stations, etc.

In certain cases, such as when specifying one-stop service, we want to ensure that, for a given pair of flight legs \( (f,g) \), the first is followed by the second on the same type. This is accomplished by a side constraint of the form

\[
z_{(f,g,e)} = \sum_h z_{(g,h,e)}
\]  

(6)

in which \( e \) ranges over all types capable of flying \( f \) followed by \( g \), and \( h \) ranges over all successor flight legs of \( g \) on type \( e \).

The matrices \( C_1, \ldots, C_E \) specify the flight coverage constraints, which ensure that, for each flight leg \( f \), exactly one aircraft type is assigned to it. They have the form

\[
\sum_f z_{(f,g,e)} = 1, \quad \forall g.
\]  

(7)

Because each \( z_{(f,g,e)} \) is binary, exactly one of them will have value 1 at a solution. Each matrix \( C_e \) thus consists of columns of the identity, corresponding to flight activities, and of zero columns, corresponding to sit activities.

The matrices \( U_1, \ldots, U_E \) specify the resource constraints. A set of activities is related to a utilization level by an equality constraint of the form

\[
r_i = \sum_{a \in S_i} u_{ia} x_a
\]  

(8)

where \( i \) is the resource to be controlled, \( r_i \) is the level, \( S_i \) is the set of activities that consume the resource and \( u_{ia} \) is the unit consumption by activity \( a \). This equation, along with the target, range, and slopes of the associated penalty function, are sufficient to convey these constraints. Thus, if a (flight or sit) activity on type \( e \) utilizes a resource, then the corresponding entry of the matrix \( U_e \) is the associated (positive) unit consumption, and 0 otherwise.

3. COMPUTATIONAL RESULTS

The OR Group of USAir has been providing automated decision support to USAir’s schedule development department since 1993. This latest fleet assignment model has been integrated into the SDE, which consists of a graphical user interface, a database and a suite of software tools to support the creation, editing, and evaluation of airline schedules. Fleet assignments are currently produced by the following steps.

1. The planners use SDE to prepare the schedule, specify the operational rules, and select the revenue and cost components which define an instance of the fleet assignment problem.
2. An AMPL process reads the model and the SDE extracts data files, constructs the flight network and the resource constraints, and formulates the optimization problem.
3. AMPL spawns a CPLEX process to solve the linear programming (LP) relaxation of the problem.
4. A rounding heuristic is applied in AMPL to fix the assignments of flight legs that have near-integer assignments in the LP solution.
5. AMPL spawns a new CPLEX process to assign the remaining flight legs, through a branch-and-bound search controlled by parameters input by the planners.
6. The planners view and evaluate the resulting assignment in SDE. They can then refine operational rules and resource targets, and iterate.

The above process is used to fleet a variety of schedules: long-range, intermediate, and current. A recent planning effort, involving seven schedules, offered us the opportunity to analyze the computational performance of the process. All runs were done on an IBM RS-6000/590 workstation with 256 MB of RAM, using AMPL Version 19950802 and CPLEX version 3.0. Table I displays the characteristics of the flight networks derived from the schedules. Differences between schedules are due to the addition and deletion of flights, as well as to changes in departure times and flight times. At the four hub cities, which are the busiest stations, ground flow logic applies.

In all schedules the following resource requirements were present: aircraft count, crew utilization, and maintenance overnights (both as a specified
TABLE I
Characteristics of the Test Problems

<table>
<thead>
<tr>
<th>Month</th>
<th>Fleets</th>
<th>Stations</th>
<th>Flight Logs</th>
<th>Operations</th>
<th>Flight Activities</th>
<th>Sit Activities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jan</td>
<td>8</td>
<td>99</td>
<td>1,609</td>
<td>11,596</td>
<td>37,145</td>
<td>5,028</td>
</tr>
<tr>
<td>Feb</td>
<td>8</td>
<td>96</td>
<td>1,602</td>
<td>11,348</td>
<td>32,026</td>
<td>4,865</td>
</tr>
<tr>
<td>Apr</td>
<td>8</td>
<td>97</td>
<td>1,611</td>
<td>11,492</td>
<td>40,979</td>
<td>5,035</td>
</tr>
<tr>
<td>Jun</td>
<td>8</td>
<td>99</td>
<td>1,617</td>
<td>11,470</td>
<td>31,830</td>
<td>5,025</td>
</tr>
<tr>
<td>Sep</td>
<td>8</td>
<td>98</td>
<td>1,620</td>
<td>11,522</td>
<td>36,945</td>
<td>5,064</td>
</tr>
<tr>
<td>Nov</td>
<td>8</td>
<td>98</td>
<td>1,620</td>
<td>11,544</td>
<td>34,649</td>
<td>5,035</td>
</tr>
<tr>
<td>Dec</td>
<td>8</td>
<td>98</td>
<td>1,610</td>
<td>11,480</td>
<td>34,658</td>
<td>5,011</td>
</tr>
</tbody>
</table>

It takes AMPL 2–4 minutes to read the data and formulate each optimization problem. Their dimensions are shown in Table II. About 85% of the constraints are network constraints, of the form (5). The coupling constraints account for 12% of the total and correspond to the flight coverage requirement (7). The network side constraints consist of the one-stop restrictions (6) and the resource utilization constraints (8), which for these schedules decompose per aircraft type. About 87% of the columns are the binary flight arcs. The remainder is made of the sit arcs at the hubs and the auxiliary variables modeling the piecewise linear terms in the objective.

To reduce the size of the problem we invoke CPLEX's presolver (preprocessor). The presolver applies techniques such as variable substitution, aggregation of nodes, and elimination of dependent rows (Brearley, Mitra, and Williams, 1977). In all cases, it takes no more than a minute to presolve. The LP sizes after presolving are shown in Columns 2 and 3 of Table III. Their low density, about 0.04%, is characteristic of network structures.

The optimal objective values in Table III are with respect to the variable profit of the fleet assignment, defined to be the optimal value after deducting $\sum r \min_{g,e} p_{(f,g,e)}$, the value of the least profitable assignment that meets the resource allocation targets. Similarly, all the LP-MIP objective gaps we will display will be with respect to variable profit.

To solve the LP relaxation we initially let the CPLEX optimizer select an algorithm based on the structure of the problem. It opted for primal simplex

TABLE II
Sizes of the AMPL-Generated Optimization Problems

<table>
<thead>
<tr>
<th>Month</th>
<th>Network (nodes)</th>
<th>Side</th>
<th>Coupling</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jan</td>
<td>11,596</td>
<td>151</td>
<td>1,609</td>
<td>13,356</td>
</tr>
<tr>
<td>Feb</td>
<td>11,352</td>
<td>656</td>
<td>1,602</td>
<td>13,610</td>
</tr>
<tr>
<td>Apr</td>
<td>11,492</td>
<td>610</td>
<td>1,611</td>
<td>13,713</td>
</tr>
<tr>
<td>Jun</td>
<td>11,470</td>
<td>596</td>
<td>1,617</td>
<td>13,883</td>
</tr>
<tr>
<td>Sep</td>
<td>11,522</td>
<td>589</td>
<td>1,620</td>
<td>13,731</td>
</tr>
<tr>
<td>Nov</td>
<td>11,544</td>
<td>580</td>
<td>1,620</td>
<td>13,744</td>
</tr>
<tr>
<td>Dec</td>
<td>11,480</td>
<td>582</td>
<td>1,610</td>
<td>13,672</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Month</th>
<th>Network (arcs)</th>
<th>Resource</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jan</td>
<td>42,173</td>
<td>206</td>
<td>42,379</td>
</tr>
<tr>
<td>Feb</td>
<td>36,688</td>
<td>203</td>
<td>36,891</td>
</tr>
<tr>
<td>Apr</td>
<td>45,508</td>
<td>206</td>
<td>46,014</td>
</tr>
<tr>
<td>Jun</td>
<td>36,660</td>
<td>195</td>
<td>36,855</td>
</tr>
<tr>
<td>Sep</td>
<td>41,812</td>
<td>197</td>
<td>42,009</td>
</tr>
<tr>
<td>Nov</td>
<td>39,498</td>
<td>195</td>
<td>39,693</td>
</tr>
<tr>
<td>Dec</td>
<td>39,475</td>
<td>194</td>
<td>39,669</td>
</tr>
</tbody>
</table>

TABLE III
LP Relaxation Results with CPLEX 3.0

<table>
<thead>
<tr>
<th>Month</th>
<th>Presolved Size</th>
<th>Optimal Value</th>
<th>Dual Simplex Minutes</th>
<th>Iterations</th>
<th>Barrier Minutes</th>
<th>Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jan</td>
<td>8,774</td>
<td>36,895</td>
<td>2,643,116</td>
<td>44</td>
<td>49,023</td>
<td>56</td>
</tr>
<tr>
<td>Feb</td>
<td>8,457</td>
<td>31,732</td>
<td>2,198,710</td>
<td>42</td>
<td>53,402</td>
<td>39</td>
</tr>
<tr>
<td>Apr</td>
<td>8,484</td>
<td>40,621</td>
<td>2,182,518</td>
<td>60</td>
<td>68,211</td>
<td>43</td>
</tr>
<tr>
<td>Jun</td>
<td>8,456</td>
<td>31,599</td>
<td>2,168,222</td>
<td>39</td>
<td>48,816</td>
<td>38</td>
</tr>
<tr>
<td>Sep</td>
<td>8,526</td>
<td>36,681</td>
<td>2,195,710</td>
<td>59</td>
<td>67,896</td>
<td>41</td>
</tr>
<tr>
<td>Nov</td>
<td>8,444</td>
<td>34,378</td>
<td>2,175,653</td>
<td>50</td>
<td>57,780</td>
<td>33</td>
</tr>
<tr>
<td>Dec</td>
<td>8,441</td>
<td>34,364</td>
<td>2,202,687</td>
<td>52</td>
<td>61,559</td>
<td>35</td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td></td>
<td></td>
<td>49</td>
<td>58,098</td>
<td>40</td>
</tr>
</tbody>
</table>

Copyright © 2001 All Rights Reserved
TABLE IV

<table>
<thead>
<tr>
<th>Month</th>
<th>Presolved Size</th>
<th>First Integer</th>
<th>Best Integer in 2 CPU hrs</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Rows</td>
<td>Cols</td>
<td>% Gap</td>
</tr>
<tr>
<td>Jan</td>
<td>4,492</td>
<td>10,578</td>
<td>2.51</td>
</tr>
<tr>
<td>Feb</td>
<td>3,670</td>
<td>7,441</td>
<td>6.29</td>
</tr>
<tr>
<td>Apr</td>
<td>4,122</td>
<td>9,025</td>
<td>5.61</td>
</tr>
<tr>
<td>Jun</td>
<td>3,591</td>
<td>6,623</td>
<td>1.20</td>
</tr>
<tr>
<td>Sep</td>
<td>4,529</td>
<td>10,627</td>
<td>1.72</td>
</tr>
<tr>
<td>Nov</td>
<td>4,474</td>
<td>10,117</td>
<td>0.87</td>
</tr>
<tr>
<td>Dec</td>
<td>4,128</td>
<td>8,988</td>
<td>4.34</td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td></td>
<td>3.22</td>
</tr>
</tbody>
</table>

with devex pricing (HARRIS, 1973), starting from the advanced basis obtained by solving the extracted pure network problem. This did not prove effective: it required almost half a million iterations, taking 11 hours. This was unusual, because the CPLEX software has been successful in selecting good strategies for a wide range of difficult problems.

After experimentation we discovered that the best strategies were the dual simplex and the primal–dual barrier method. This agrees with the published work of Subramanian et al. (1994) and Hane et al. (1994). For both algorithms we add a small random perturbation to the objective function, in order to curb degeneracy. The use of steepest-edge pricing (GOLDFARB and REID, 1977) and CPLEX’s perturbation strategy, which expands the bounds on variables, improved the performance of dual simplex. For the barrier method the choice of minimum local fill-in ordering and a reduced complementarity tolerance at termination of $10^{-5}$ improved overall runtimes. We note that in the current AMPL–CPLEX interface the barrier method is always followed by crossover to a basis. As can be seen from columns 4 and 6 of Table III, both methods produce LP solutions in no more than one hour of elapsed time, with the barrier method being, on aggregate, 17% faster. Even with the algorithmic tuning, the iteration counts are quite high, especially for dual simplex, where they are 6 to 8 times the number of rows, instead of the usual 1 to 4 (section 3.7 in SHAMIR, 1987). This reflects the “tightness” of the schedules, as they are constructed with high utilization for the aircraft.

Following the solution of the relaxation we make certain assignments in order to reduce the size of the MIP. We fix at one flight activity variables with values no less than 0.99 and fix at zero all other eligible activities for that flight leg. A call to the presolver carries out the logical implications. About 70% of the variables are fixed in this way. The resulting problem is the root LP solved in the subsequent branch-and-bound process. Both the dual simplex and the barrier method perform well on this LP. Despite the large number of variables fixed, we found that the optimal value of the root LP agrees to 9 digits with that of the LP relaxation.

Once the root node has been solved, CPLEX’s branch-and-bound algorithm is employed to find an integer point. We currently provide two options to our users: either terminate the search at the first integer feasible point, or search for the best integer point in a time interval they set. By default CPLEX uses depth-first search to find the first integer point, and a breadth-first-based search strategy thereafter. Because these problems tend to have integer feasible points deep in the search tree, we found the breadth-first-based strategy ineffective and opted for depth-first search overall. To solve the LPs at the nodes we employ dual simplex. We also let CPLEX generate clique and cover cuts as needed and use special branching rules for “special ordered sets of type 3,” such as in the flight coverage constraints.

Problem sizes and MIP search times are shown in Tables IV and V, for the dual simplex and the barrier root solvers, respectively. The input to the MIP process in both cases is the solution of the LP relaxation obtained by the same algorithm, as modified by the fixing step. The slight size differences between some of the input problems to dual simplex and barrier are caused by fixing and then presolving on different optima of the LP relaxation. The density of the presolved root LP is about twice that of the presolved LP relaxation. It takes 4–5 minutes to solve the root LP with the dual simplex and 7–8 minutes with the barrier method, both from cold start. For dual simplex a partial warm start is possible, by using the parts of the optimal LP relaxation basis that were not affected by fixing and presolving.

From the tables we observe that LP-MIP gaps in the range 0.3–6% of variable profit were found after two hours of computing time. (With respect to the full objective, the average gap was 0.37% for dual simplex and 1.33% for barrier.) The results obtained
by the combination of dual simplex and fixing can be used without reservation, whereas those of the barrier approach fluctuate above acceptable tolerances. Due to this aspect of the barrier method, we refrained from evaluating it further.

To explore the ability to solve these problems to optimality, we continued our experimentation using only the dual simplex method. We found out that, to close the gap further, significant additional computing effort and memory space are required, as the following examples indicate. On the February problem, CPLEX’s breadth-based default search strategy, after examining 8234 nodes, which took 10 hours and in excess of 200 MB of memory to store the tree, produced an optimal gap of 0.06%. On several of the other problems the default strategy failed, as it used up all available memory without producing a good integer point. Trading time for space, we ran a depth-first search strategy on the December problem: after 1,335,000 nodes, which took 31 CPU days and about 30 MB to store the tree, the gap was 0.28%.

The hardness of the problems can be partially attributed to the resource constraints, because their removal produces smaller gaps faster. As an illustration, we fletched the February schedule with the resource constraints removed; the first integer point, with gap 0.16%, was produced after 40 minutes total (including solving the LP relaxation); an optimal integer point, with gap 0.03%, was found after 88 additional minutes. In comparison, as shown in Table IV, with resource constraints, the gap after two search hours was 6%. Similar observations are made by (Clarke et al., 1994).

To better assess the beneficial impact of the fixing heuristic, we ran several schedules without it and observed a severe increase in computing time. On the November problem, for example, the first integer point without fixing was found after 5 hours and 314 nodes and had a gap of 1.07%. In comparison, using fixing, as shown in Tables III and IV, the first integer point, with a gap of 0.87%, was obtained after a total of 63 minutes and only 84 nodes.

A final experiment illustrated the computational benefits of using connection complexes: for the November problem we retained the ground flow logic where it applies but allowed all possible connections at the other stations, instead of connecting only within complexes. The generated problem had 124,435 variables, out of which 119,391 were the binary flight arcs, i.e., 3.5 times as many as in the version with complexes. To solve the LP relaxation, it took the dual simplex method 172,224 iterations, requiring 446 minutes; the barrier method took 51 iterations and 190 minutes. The value of the optimal objective for both algorithms agreed with that for the version with complexes, in nine significant digits. Thus, no profit was gained by the additional connection options. Instead, the solution times increased 6- and 9-fold, respectively. After fixing and presolving, the MIP had 4848 rows and 24,374 columns, i.e., more than twice the number of variables for the version with complexes. This resulted in markedly inferior results in the MIP phase: the search started from the optimal LP basis for dual simplex produced a gap of 17.96% after 2 CPU hours and 800 nodes. Starting the search from the optimal crossover LP basis for the barrier method fared even worse: after 2 CPU hours and 551 nodes, the gap was 160.8%.

4. CONCLUSIONS AND FUTURE DIRECTIONS

WE HAVE PRESENTED FAM, a second-generation fleet assignment model, that has been integrated into USAir’s Schedule Development Environment. The model is complete in its ability to capture the detailed connection and resource requirements faced by large airlines. At the same time the model is robust, returning useful answers in the face of the inconsistencies typical of the schedule design pro-
cess. Moreover, a comprehensive economic tradeoff is achieved through a unified treatment of resource constraints by incorporating penalties in the objective function. Care has been taken in the design of the model, to provide these capabilities without compromising computational performance: as demonstrated in Section 3, FAM finds near optimal fleet assignments for real problems in about an hour of computing time. Because of these features, the model is currently in wide use by schedule planners at USAir.

One track of our ongoing work is to increase the scope of the model with respect to the length of the decision horizon. Within the framework of single-flight-leg-based cost, the one-day model we described can be linearly extended to a multi-day model; our first target in this direction is a weekend model which uses different schedules for Friday, Saturday, and Sunday. Extending the planning horizon allows for a greater degree of optimization, for instance in scheduling flights around holidays and in determining charter leasing on weekends. From a computational standpoint, this extension translates to a significant growth in the size of the optimization problem, which increases the LP solution time and makes possible exponential growth in the size of the branch-and-bound tree. To solve the LP relaxation efficiently, we are considering warm starts based on the solution of the one-day problem, as well as specialized decomposition algorithms for block-angular optimization (De Leone et al., 1994). For the branch-and-bound phase we are considering problem-specific branching rules coupled with parallel searches (Rushmeier and Nemhauser, 1993).

Derivatives of FAM can support decision making in the resource planning and schedule development activities surrounding fleet assignment. We are developing such models to determine the optimal size and composition of fleets with respect to aircraft ownership and crew costs, to examine the impact of retiring or adding aircraft and to assist in the transition from one schedule to another. The foundation of FAM and its derivatives is a core model which describes the flight connection logic and the constraints imposed by the operational restrictions and the allocation of resources. In building such a family of models, the example of AMPL demonstrates that a modeling language holds considerable advantages over a regular programming language: its flexibility and compactness enable the modification and extension of the core model at a high level and in a relatively short time.

As the sophistication of the schedule planning process improves with respect to forecast quality and profit estimates, there is a need for models that use as a unit of planning a sequence of connected flight legs. Consideration of up to a day's flying for each aircraft (line of flying) allows fleet assignment to capture more accurately costs such as crew penalties. A line of flying-based FAM leads to a set covering problem with an enormous number of columns, because of the density and connection patterns of large flight networks. Continued algorithmic developments, such as sophisticated column generation (Anbil et al., 1992), shortest-path-based decomposition (Lavoie, Minoux, and Odié, 1988 and Barnhart et al., 1991), and branch-and-cut (Hoffman and Padberg, 1992) hold promise for success in solving a line-of-flying-based fleet assignment, as they have been in large crew scheduling problems. Parallel computing will be instrumental in dealing effectively with a problem of such size and complexity.

REFERENCES


T. Feo and J. Bard, "Flight Scheduling and Maintenance


(Received, February 1996; revisions received: June 1996; accepted July 1996)