Annals of Operations Research 69(1997)33–46

The aircraft rotation problem

Lloyd Clarke[a], Ellis Johnson[a], George Nemhauser[a] and Zhongxi Zhu[b]

[a]*School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA 30332-0205, USA*

[b]*Metron, Inc., Washington, DC, USA*

Given a set of flights to be flown for a specific aircraft type, with specified maintenance locations, durations, and needed frequency, the *aircraft rotation problem* is to determine the specific route flown by each aircraft. The objective is to maximize the benefit derived from making specific connections. In this paper, we present a mathematical formulation for the aircraft rotation problem (ARP) and discuss its similarity with the asymmetric traveling salesman problem. We solve the ARP by Lagrangian relaxation and subgradient optimization and present computational results on real data from a major airline.

1. Introduction

This paper models and solves the aircraft rotation problem for a commercial passenger airline. The term rotation is widely used in the airline industry and therefore is adopted here as well. The *aircraft rotation problem* is to determine the routes flown by each aircraft in a given fleet. An airline must first solve the *fleet assignment problem*, which is to determine which fleet (aircraft type) is assigned to each flight of a given schedule. A good rotation must be profitable and allow each aircraft to regularly undergo different types of maintenance checks. This routing process is a critical part of the entire planning process.

Profitability is measured by the sum of through values of routing flights through airports. A *through value* is the desirability of one-stop service between a pair of cities, i.e., the revenue that would be expected to be gained from additional passengers who would be attracted to the service because of being able to stay on the same airplane rather than having to change airplanes at the stopover point.

The operational requirement is a single route for each fleet, i.e., each plane in the fleet must fly the same route. The maintenance requirement is regular visits of sufficient length to a maintenance station. Unlike the common practice of fixing the connections during the day and only using overnight connections as options for maintenance routing, our approach uses all connections as options in maintenance routing. Our approach also handles different types of maintenance requirements.

We model the aircraft rotation problem as an asymmetric traveling salesman problem with side constraints and solve the model using Lagrangian relaxation and subgradient optimization. The model and algorithm have been evaluated on real data from a major airline.

This paper is organized as follows. In the remainder of the introduction, we define the aircraft rotation problem and discuss related issues. We provide a survey of related literature. In section 2, we develop the mathematical programming model for the aircraft rotation problem. In section 3, we present preprocessing techniques to simplify the time-line network and we give the details of the Lagrangian relaxation algorithm. In section 4, we present computational results, discuss implementation issues, and present further research opportunities and conclusions.

1.1. The aircraft rotation problem

Assigning the right fleet (aircraft type) to the right set of flight legs is called fleet assignment. An airline usually has a variety of fleets that vary in their seating capacity and operational characteristics. Considering factors such as passenger demands (both point-to-point and continuing service), revenue, seating capacity, fuel costs, crew size, maintenance availability, and gate availability, the airline needs to assign a fleet to each flight leg of the schedule in order to maximize the total potential profit of implementing the schedule. Solving the fleet assignment problem using mathematical programming methods is addressed in [1, 2, 6, 7, 9].

The fleet assignment includes *maintenance constraints* such that the fleet assignment solution may specify a given percentage of the aircraft of a fleet to be at maintenance stations overnight. However, the fleet assignment problem does not influence the spacing of maintenance visits. Adding constraints to do this spacing would make the fleet assignment problem computationally intractable. Therefore, how to space maintenance visits evenly is one of the two basic considerations of the aircraft rotation problem.

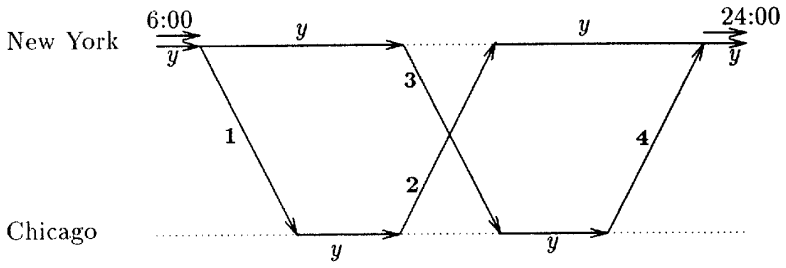

Figure 1. A simple rotation problem.

To explain a rotation, we consider the example in figure 1. This simple time-line network shows 4 flights assigned to a specific fleet. The aircraft rotation problem

orders these flights. For the flights in figure 1, one possible rotation is to cover flight 1, flight 2, flight 3, flight 4, then repeat the sequence. We represent this simply by $1-2\circ3-4\circ1$. Two other possible rotations are $1-2\circ1\diamond3-4\circ3$ and $1-4\circ3\circ2\circ1$. Each of these rotations has different characteristics which we will explain now along with the symbols that describe the rotations.

The schedules used in this paper are daily. The flights that are shown in figure 1 are repeated on subsequent days. For rotation $1-2\circ3-4\circ1$, on day 1 aircraft *A* takes flights 1, 2, then spends the night in NY, while aircraft *B* takes flights 3, 4 and also overnights in NY. On day 2, aircraft *A* takes flights 3 and 4 and aircraft *B* takes flights 1 and 2. This completes the rotation. This rotation covers the 4 flights with two aircraft, both overnighting in NY. The symbol $\circ$ indicates an overnight stay between flights. The number of overnights is the number of aircraft needed to complete the rotation. The other feature of this rotation is that the aircraft used to satisfy this schedule cover each flight. Both aircraft *A* and *B* cover all four flights in the same order. The last feature is that planes are only on the ground when there is a *y* arc between flights.

The rotation $1-2\circ1\diamond3-4\circ3$ indicates a different flying sequence. Aircraft *A* covers flights 1,2, overnights in NY and then repeats flights 1,2 the next day. Aircraft *B* covers flights 3,4, overnights in NY and repeats the same flights the next day. This rotation uses the same number of planes as the previous rotation and also follows the ground arcs. What causes the rotations to differ is the choice made overnight in NY when there are 2 planes on the ground. When the flights covered by one aircraft are separate from the flights covered by another aircraft in the fleet, we follow the airline terminology of calling this a *broken rotation*. The symbol $\diamond$ indicates where coverage breaks into different sets of aircraft. We refer to these breaks as "continuity breaks" since they break the continuous flow of an aircraft around a cycle that contains all of the flight legs. There can be several continuity breaks in a rotation and within each break there can be many overnights. This paper follows the practice of an airline that forbids continuity breaks. We do not consider a solution to be a rotation unless there are no continuity breaks. A *locked rotation* is a broken rotation where the separate sequences do not have any locations in common. To fix a locked rotation requires a new solution from fleet assignment. Locked rotations are unacceptable to almost all airlines.

The rotation $1-4\circ3\circ2\circ1$ requires 3 planes to cover the same 4 flights that the other rotations covered with only 2 planes. The reason for this is that flight 1 is followed by flight 4. This requires that an aircraft remain on the ground all day when *y* arcs do not indicate this. Violating the ground arcs given by the fleet assignment problem may result in the need for additional planes. The fleet assignment problem assigns planes to flights and ground stays to satisfy many different criteria including fleet size constraints. In the process of determining a fleet assignment, an airline may have a final solution that has many intangible and desirable aspects. It is important that the rotation problem does not change the nature of the fleet assignment by changing when planes are on the ground.

Unlike the simple example presented in figure 1, real aircraft rotation problems of major domestic carriers have many connection decisions that have to be made because of the underlying hub-and-spoke structure of the schedule. These rich connecting opportunities during the day at hubs, result in exponentially many possible rotations. The aircraft rotation problem seeks the rotation that yields the largest total through value and satisfies maintenance constraints.

To explain the use of through values, we present the example in figure 2. Here, Atlanta has two inbound flights, one from Birmingham and the other from Boston, and two outbound flights to Miami and New York. We can route these two inbound

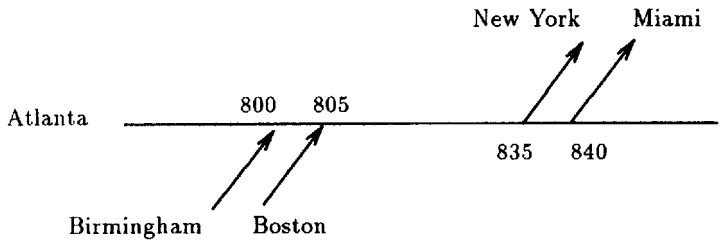

Figure 2. Through values.

flights and two outbound flights in two different ways. Routing A is Birmingham-Atlanta-New York and Boston-Atlanta-Miami. Routing B is Birmingham-Atlanta-Miami and Boston-Atlanta-New York. In routing B, Boston to NY via Atlanta has no marketing advantage and therefore has a through value of zero. Suppose that the through value of Birmingham-Atlanta-New York is a, Boston-Atlanta-Miami is b, Birmingham-Atlanta-Miami is c, and Boston-Atlanta-New York is zero. Then routing A is better if $a + b > c$.

The through value of a connection also depends on the ground time between its inbound flight and its outbound flight. Airlines usually assign zero through values when the ground time is longer than one and a half hours because a long connecting time does not attract passengers.

Every airline has its own maintenance policy which complies with the FAA rules. We have adopted the following from a major US airline. We assume that a *routine* check takes four hours and needs to be performed every three days. An *avionics* check takes five hours and needs to be performed every four days. An avionics check covers the requirements of a routine check with additional items included. Due to equipment requirements, stations may be able to perform both routine and avionics checks, only routine or neither. We will use the generic term *service* to mean a routine and/or avionics check.

A service break is the length of time between two consecutive service checks of the same type along the rotation. We measure this length of time by taking the difference between the departure time of the flight leaving immediately after the last

service and the arrival time of the flight arriving immediately before the next service. We use the longest service break to measure the quality of a given rotation. We call this longest break the *service index* of the rotation. Specifically, we call the longest routine check break the *routine index* of the rotation and the longest avionics check break the *avionics index*. If this service index is under the specified period, then all service breaks are under the specified period. In terms of maintenance, the smaller this service index, the better the rotation is.

Using the routine and avionics indices defined earlier, we can now define feasible and optimal rotations. The aircraft rotation problem is to maximize the total through value subject to operational and maintenance constraints. We say a rotation is *feasible* if there are no continuity breaks and its service index is under the specified service period for each type of service performed. If there exists a feasible rotation, then an *optimal* solution to the aircraft rotation problem is the rotation with the largest total through value among all feasible rotations.

1.2. Previous work

Previous work on the aircraft rotation problem has focused on the maintenance routing portion of the problem without directly considering through values and is restricted to only one type of frequent maintenance requirement. A common practice is to fix the connections during the day and only use overnight connections as options for maintenance routing. This practice significantly reduces the options to form rotations, especially for the hub-and-spoke structures. In addition, heuristics have been the main solution approach. Thus, the literature is very limited on both models and methodologies for solving the aircraft rotation problem.

Feo and Bard [3] study the maintenance location problem which involves finding the minimum number of maintenance stations required to meet the specified 4-day requirement for a proposed flight schedule. They assume that the interim stops of flights during the day are unimportant. Using the one-day routings between overnight cities as input, they formulate the maintenance base location problem as a minimum cost, multicommodity network flow problem with integer restrictions on the variables. As such, each plane represents a separate commodity, and each arc has an upper and lower capacity of one. Because the size of the formulation is too large to optimize, they give a two-phase heuristic.

Kabbani and Patty [8] study the maintenance routing problem for American Airlines where each aircraft needs to be checked every three days. They formulate the problem as a set partitioning model where a column represents a possible week-long routing and a row a flight. They develop pseudo-costs to penalize routings with unfavorable characteristics such as violation of connection times, maintenance violations, and continuity breaks. However, this formulation becomes too large to solve for the MD-88 fleet. Thus, they separate the problem into two subproblems. First, they solve for the appropriate “over-the-day” routings (fixed sequences of flight legs

leaving in the morning and arriving at night). Second, they solve for the connections among these routings. If the subproblem approach does not find an appropriate solution, they then try to generate solutions by swapping either flights at spoke stations or the routings. Their least desirable option is to change daily connections at hubs because of the customer service implications.

Gopalan and Talluri [5] study the maintenance routing problem for US Air, where each aircraft needs to have a routine check every three days and a "balance-check" once in a while. They consider the problem for a daily schedule under the term "static infinite horizon model". Their maintenance routing procedure starts with a set of sequences of flight legs. Each of these sequences is constructed by fixing all connections during the day at non-overnight stations using simple rules such as first-in-first-out or last-in-first-out. As the result of this daily connection fixing process, the resultant network contains only overnight stations and the fixed sequences between these stations. They then route the maintenance on this reduced network to try to meet the three-day maintenance requirement. The independent process of fixing daily connections before the actual maintenance routing can easily create locked rotations. In this case, they try to "unlock" the rotation by swapping either flights or even fleets and, at the same time, try to improve the maintenance routing.

In practice, it appears to be the case that very little has been done in the airline industry to build decision support systems for the rotation problem. Moreover, software that is currently used focuses entirely on maintenance feasibility and does not consider through values.

2. Modeling the aircraft rotation problem

The aircraft rotation problem is to maximize the through value of connecting flights, subject to no continuity breaks and acceptable maintenance. This can be alternatively stated in the following manner. A directed graph $\mathcal{N}$ is called Eulerian if each node has in-degree equal to out-degree and it is connected. An Euler tour of an Eulerian digraph is a cycle that includes all the arcs (in our case, flight and ground arcs) exactly once. We want to find an Eulerian tour that maximizes value and satisfies maintenance constraints. Given an Eulerian digraph $\mathcal{N}$, an Euler tour is represented by a sequence of arcs of A as $i_1, i_2, \ldots, i_{|A|}$ such that i_k and i_{k+1} are adjacent for $k = 1, \ldots, |A|$, where $i_{|A|+1} = i_1$. If arc i and arc j are adjacent, we say that there is a *connection* from i to j. Such a connection is denoted by $i \to j$.

To formulate Euler tours, we let x_{ij} denote the decision variable for the connection from $i \in A$ to $j \in A$ such that $h(i) = t(j)$ and $i \neq j$, where $h(i)$ is the head node of arc $i \in A$ and $t(i)$ is the tail node. Hence,

$$x_{ij} = \begin{cases} 1 & \text{if } i \to j, \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

An Euler tour in $\mathcal{N}(V, A)$ traverses every arc of $\mathcal{N}$ exactly once. Therefore, for every arc $i \in A$ there is exactly one connection $j \in A \setminus \{i\}$, i.e.,

$$\sum_{j:h(i)=t(j),\, j\neq i} x_{ij} = 1 \quad \text{for all } i \in A. \tag{2}$$

Similarly, for every arc $j \in A$ there is exactly one connection $i \in A \setminus \{j\}$, i.e.,

$$\sum_{i:h(i)=t(j),\, i\neq j} x_{ij} = 1 \quad \text{for all } j \in A. \tag{3}$$

The continuity break constraints, which we now call subtour elimination constraints, can be stated as

$$\sum_{i\in S,\, j\in A\setminus S \text{ s.t. } h(i)=t(j)} x_{ij} \geq 1 \quad \text{for all } S \subset A \text{ with } 2 \leq |S| \leq |A| - 2. \tag{4}$$

Constraints (1)–(4) completely define Euler tours.

Let v_{ij} be the through value of flight j following flight i. If either i or j is a ground arc, then v_{ij} has a value of zero. Our objective is

$$\max \sum_{(i,j)\in C} v_{ij} x_{ij},$$

where $C = \{i, j \in A \text{ s.t. } h(i) = t(j),\ i \neq j\}$. We now have the problem of finding an Euler tour of maximum value. We refer to this as the *through value problem* since it does not include maintenance constraints. Note that the through value problem is an asymmetric maximization traveling salesman problem (ATSP). To see this, we construct a new digraph $\mathcal{D}$, called the *line graph* of $\mathcal{N}$, where the arcs of $\mathcal{N}$ become nodes of $\mathcal{D}$. Two nodes i and j of $\mathcal{D}$ are joined by a directed arc (i, j), if in $\mathcal{N}$ i is an "in" arc and j is an "out" arc of a common node. The arc (i, j) in $\mathcal{D}$ is assigned the value v_{ij}. Then there is a one-to-one correspondence between Euler cycles in $\mathcal{N}$ and Hamiltonian cycles in $\mathcal{D}$. We will continue to use the Euler tour terminology because the tradition in the airline industry is to model flights with arcs, as in the time line network.

In order to complete the formulation of the aircraft rotation problem as the constrained Euler tour problem, we need to develop maintenance constraints. A *service violation* path is a path with a length in time longer than the specified service period. Such paths must be eliminated from the solution. It suffices to eliminate explicitly only those service violation paths which are not properly contained in any other service violation paths since the longer ones will be implicitly eliminated. Let $\mathcal{P}^k$ be the set of all minimal violation paths for maintenance type k and K the set of maintenance types. Let P' denote the set of arcs of $P \in \mathcal{P}^k$ excluding the last arc and let $f(i)$ denote the follower of $i \in P'$. Then constraints

$$\sum_{i \in P',\, j \in A \setminus f(i) \text{ s.t. } h(i) = t(j)} x_{ij} \geq 1 \quad \forall P \in \mathcal{P}^k, k \in K \tag{5}$$

eliminate all service violation paths. This is true because at least one arc must leave an intermediate node of the path.

We can now state the aircraft rotation problem as

(ARP) maximize $\sum_{(i,j) \in C} v_{ij} x_{ij}$

subject to (1) – (5).

3. Algorithm

3.1. Preprocessing

We use *arc aggregation* and *node aggregation* to simplify any structure of the network which provides no choice in forming Euler tours.

Any node with its in-degree equal to one does not contribute to the formation of tours. We aggregate the in-arc and out-arc into a super-arc and therefore eliminate the node between them. After such arc aggregation, all nodes have in-degree of at least two. Figure 3 illustrates the arc-wise aggregated network of the time-line network

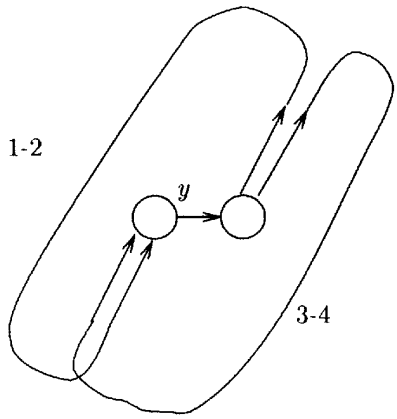

Figure 3. Arc aggregation.

given in figure 1. The two super-arcs are the two sequences of flight legs 1–2 and 3–4. Note that this aggregation may remove a constant from the objective function. In real problems that are significantly larger, the super-arcs will likely be composed of many flight segments.

Through values are based on flight to flight connections. The incoming flights in figure 3 connect to a ground arc. To make this situation clearer, we can remove y and consolidate the two nodes. The resulting network is shown in figure 4.

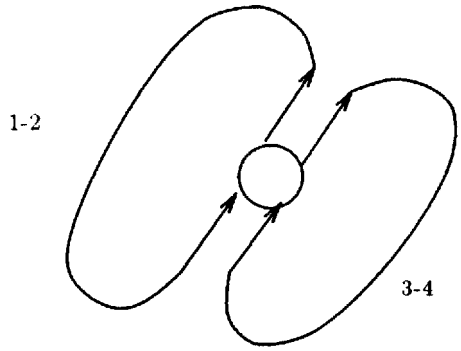

Figure 4. Node aggregation.

The problem presented in figure 1 has been fully preprocessed to arrive at figure 4. The choices left are to connect incoming flight 4 (head of super-arc 3–4) to outgoing flight 1 (tail of super-arc 1–2) or to outgoing flight 3 (tail of super-arc 3–4). Incoming flight 2 would be connected to the other outgoing flight. The through values of super-arc connections are determined by the last flight for an incoming super-arc and the first flight of an outgoing super-arc.

Part of the role of preprocessing is to create one super-node at each city, where choices are available, by removing all the ground arcs. This leaves only flight-to-flight connection choices to be made. This process must be done in such a way that the ground arcs are honored without their explicit existence. For example, in figure 5,

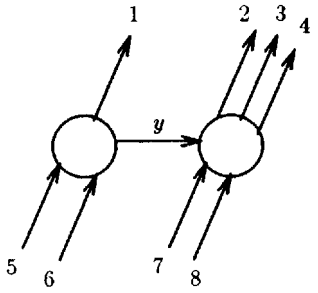

Figure 5. Time reversal from 7 or 8 to 1.

flight 7 or 8 cannot connect to flight 1. When the super-node is created, the connections 7–1 and 8–1 are given through values of negative infinity. All other through values remain unchanged. This essentially forbids these connections in the assignment of incoming flights to outgoing flights.

As a result of arc and node aggregation, service opportunities can be contained in the aggregated arcs or nodes. After all preprocessing is done, what remains is several arcs and at most one node for each city in the network. Tests on data from a

major airline have shown that these aggregation techniques reduce the size of the networks by a factor of 4 to 100. Note that the aggregations keep the Eulerian property of the network. From now on, without loss of generality, we assume that $\mathcal{N}$ is Eulerian and both arc-wise and node-wise aggregated.

3.2. Lagrangian relaxation

ARP with constraint sets (4) and (5) omitted is an assignment problem that actually decomposes into separate assignment problems for each city for which choices are available. Thus, our approach is to dualize constraint sets (4) and (5) to obtain the Lagrangian relaxation given by

$$Z_{LR}(\lambda,\mu) = \max_{x} \sum_{(i,j)\in C} v_{ij}x_{ij} + \sum_{t\in T}\lambda_t\left(\sum_{(i,j)\in C(t)} x_{ij} - 1\right) + \sum_{v\in V}\mu_v\left(\sum_{(i.j)\in C(v)} x_{ij} - 1\right)$$

$$\text{subject to} \quad (1) - (3),$$

where T is the set of subtour elimination constraints and V is the set of service violation constraints. $C(t)$ and $C(v)$ are the set of connections in the tth subtour constraint and vth violation constraint, respectively. The Lagrangian dual is

$$(LD) \qquad Z_D = \min_{\lambda,\mu\geq 0} Z_{LR}(\lambda,\mu).$$

We solve *LD* using conventional subgradient optimization, see e.g. Fisher [4], except that since (4) and (5) contain an exponential number of constraints, we only dualize violated constraints as they are identified. In particular, let $\overline{T}\subseteq T$, $\overline{V}\subseteq V$ and $LD(\overline{T},\overline{V})$ be the Lagrangian dual over the constraints given by $\overline{T}$ and $\overline{V}$.

Initially, we solve $LD(\varnothing,\varnothing)$. Now suppose we have solved $LD(\overline{T},\overline{V})$. If all constraints in $T\backslash\overline{T}$ and $V\backslash\overline{V}$ are satisfied, we are done. Otherwise we find violated constraints and add them to the Lagrangian and again apply subgradient optimization. Actually, we have found that adding only one new violated constraint per run of the subgradient method performs well in terms of keeping the total number of dualized constraints relatively low. We add maintenance constraints before subtour elimination constraints and give preference to short paths or subtours.

Finally, in the process of solving $LD(\overline{T},\overline{V})$, the solution of the assignment problem with fixed multipliers may yield a feasible (no subtours), but not provably optimal, solution. The best of these solutions found so far is saved as the incumbent. When the process terminates, the incumbent solution is output. Thus, the procedure does not guarantee finding an optimal solution, although it could do so by being imbedded in a branch-and-bound scheme.

4. Computational results

We present computational results for both the through value and the aircraft rotation problems. The through value problem does not include any service constraints. The rotation problem includes through value and service constraints with a 3-day routine and 4-day avionics requirement. The test data is from the flight schedule of a major US airline. All computations were done on an IBM RS/6000 model 550 workstation running AIX 3.1. All implementation was done in the C programming language.

Table 1 displays the characteristics of the eleven test case aircraft rotation problems. The two columns under "Original" display the sizes of the original time-line networks given by the fleet assignment solution. The first two columns under

Table 1

Characteristics of the test problems.

	Original		Aggregated			Time (sec.)
Fleet	Arcs	Nodes	Arcs	Nodes	0–1 Vars.	Preprocessing
f1	195	154	30	12	84	16.1
f2	1292	638	170	37	1078	24.1
f3	192	143	32	13	102	16.5
f4	3818	1095	361	55	5937	39.3
f5	141	129	12	6	24	15.9
f6	2246	1002	307	66	2271	32.1
f7	1342	651	210	52	1278	24.2
f8	149	122	27	12	69	15.9
f9	228	169	38	14	116	16.3
f10	151	115	27	11	71	15.8
f11	43	39	4	2	8	15.1

"Aggregated" display the sizes of the aggregated networks and the next column gives the number of 0–1 variables in the aggregated formulation. The last column gives the total time spent on all the necessary initial processing, including reading in data and aggregating the network.

Table 2 gives the computational results of the Lagrangian relaxation method for the through value problem. The solution objective function value (maximum through value) is in the second column. The third column gives the resultant routine and avionics indices from the Lagrangian solutions even though routine and avionics requirements are not considered in solving these through value problems. Since the Lagrangian solutions of f3, f5, f8, f9, f10 and f11 satisfy the 3-day routine and 4-day avionics requirements, these problems are completely solved as both through value problems and aircraft rotation problems. and will not be discussed further.

Table 2

Results for the through value problem.

Fleet	*T* value	*R*/*A*	Sbtrs	Subgradient steps	Time (sec)
f1	1,470	2.7/5.6	1	1	0.4
f2	14,029	4.7/9.6	2	1,1	0.6
f3	2,038	2.5/3.6	1	1	0.2
f4	46,615	7.6/22.6	5	40,1,1,1,1	24.7
f5	4,261	2.6/3.5	0	0	0.1
f6	45,575	2.7/6.7	10	53,1,1,56,1,1,1,1,1,211	49.9
f7	47,183	3.7/5.4	3	45,1,1	3.8
f8	4,153	1.7/1.7	3	1,1,1	0.2
f9	6,066	1.6/1.6	2	1,1	0.1
f10	8,111	1.6/3.5	0	0	0.1
f11	2,020	1.5/1.5	0	0	0.1

The fourth and fifth columns give the number of subtour elimination constraints generated and the corresponding number of iterations for solving the restricted Lagrangian dual. The last column gives the CPU times in seconds. Each of the solutions was proven optimal.

Table 3 gives the computational results of the Lagrangian relaxation method for the aircraft rotation problem for the fleets in table 2 that did not yield a feasible service solution. The third column gives the percentage deviation from the through value problem. The sixth the number of maintenance violation elimination constraints, and the seventh the specified periods for routine and avionics checks used to run the Lagrangian relaxation method.

Table 3

Results for the aircraft rotation problem.

Fleet	*T* value	%Opt.	*R*/*A*	Sbtrs	Mtc	*R*/*A* bounds	Time
f1	1,455	1.03	2.6/3.4	8	2	3.0/4.0	1.4
f2	13,218	6.14	3.6/5.8	87	85	6.0/6.5	55.9
f4	46,031	1.27	3.8/8.7	119	653	6.0/9.2	2264
f6	45,489	0.19	2.8/4.6	6	8	3.5/4.7	6.0
f7	47,183	0	2.6/3.9	6	1	3.0/4.0	1.4

For f1, a 1% drop in through value gets a maintenance feasible rotation. This solution was later proved optimal by the branch-and-bound algorithm in [10]. The branch-and-bound approach is computationally much more intensive, which is the reason for presenting only the Lagrangian relaxation heuristic here. For problem f7,

no drop in through value is needed to gain maintenance feasibility, i.e., for this problem there is an alternative optimal solution to the through value problem that is maintenance feasible.

For f6, the Lagrangian solution obtained the maximum through value within 0.19% of optimality and came close to satisfying the avionics requirement. Problems f2 and f4 remain very difficult for obtaining feasibility in the avionics requirement.

The actual routine period p_r and avionics period p_a used are given under "R/A" in the ninth column. For f2, f4, and f6, if p_r and p_a were set close to the required values 3 and 4 days, the Lagrangian relaxation method would generate thousands of constraints, most of which were service violation constraints and which did not converge to a feasible solution. In order to make the method converge to feasible solutions, we needed to relax p_r and p_a to higher values and try to gradually bring them down to get the "boundary" solvable solutions. Between the routine requirement and the avionics requirement, we found that avionics was the bottleneck for getting good rotations. Therefore, we tried to push p_a as low as possible. For f2, f4, and f6, these lowest p_a values were found empirically to be 6.5, 9.2, and 4.7 days.

The Lagrangian relaxation approach has worked well, although it is not quite capable of solving the very difficult problems f2 and f4. Furthermore, we could not make additional progress on these problems with branch-and-bound. Other approaches are necessary for the very difficult rotation problems. One possiblity that we are currently investigating is a column generation approach that uses sequences of flights from one maintenance station to another. Each of these sequences is generated to be maintenance feasible. A master problem determines how to put these sequences together.

Acknowledgements

This work was partially supported by AFORS and NSF grants DDM-9115768 and SES-9122674. We thank Delta Air Lines Inc., for their assistance in explaining the problem and providing data. We also thank the referees for suggestions that helped us improve the presentation.

References

[1] J. Abara, Applying integer programming to the fleet assignment problem, Interfaces 19(1989) 20–28.
[2] L.W. Clarke, C.A. Hane, E.L. Johnson and G.L. Nemhauser, Maintenance and crew considerations in fleet assignment, Transportation Science 30(1996)249–260.
[3] T.A. Feo and J.F. Bard, Flight scheduling and maintenance base planning, Management Science 35(1989)1415–1432.
[4] M.L. Fisher, An applications oriented guide to Lagrangian relaxation, Interfaces 15(1985)10–21.
[5] R. Gopalan and K.T. Talluri, The aircraft maintenance routing problem, Operations Research, to appear.

[6] Z. Gu, E.L. Johnson, G.L. Nemhauser and Y. Wang, Some properties of the fleet assignment problem, Operations Research Letters 15(1994)59–71.
[7] C.A. Hane, C. Barnhart, E.L. Johnson, R.E. Marsten, G.L. Nemhauser and G. Sigismondi, The fleet assignment problem: Solving a large-scale integer program, Mathematical Programming 70(1995) 211–232.
[8] N. M. Kabbani and B. W. Patty, Aircraft routing at American airlines, *Proceedings of the 32nd Annual Symposium of AGIFORS*, Budapest, Hungary, 1992.
[9] R. Subramanian, R.P. Scheff, Jr., J.D. Quillinan, D.S. Wiper and R.E. Marsten, Coldstart: Fleet assignment at Delta Air Lines, Interfaces 24(1994)104–120.
[10] Z. Zhu, The aircraft rotation problem, Ph.D. Thesis, Georgia Institute of Technology, 1994.