Airline Crew Scheduling with Time Windows and Plane Count Constraints

Diego Klabjan
Department of Mechanical and Industrial Engineering
University of Illinois at Urbana-Champaign
Urbana, IL, 61801

Ellis L. Johnson
George L. Nemhauser
School of Industrial and Systems Engineering
Georgia Institute of Technology
Atlanta, GA 30332-0205

Eric Gelman
Srini Ramaswamy
Research and Development
Information Services Division
United Airlines

March 27, 2001

Abstract

Airline planning consists of several problems that are currently solved separately. We address a partial integration of schedule planning, aircraft routing and crew scheduling. In particular, we provide more flexibility for crew scheduling while maintaining the feasibility of aircraft routing by adding plane count constraints to the crew scheduling problem. In addition we assume that the departure times of flights have not yet been fixed and we are allowed to move the departure time of a flight as long as it is within a given time window. We demonstrate that such a model yields solutions to the crew scheduling problem with significantly lower costs than those obtained from the traditional model.

Major US airlines operate up to 2,500 domestic flights per day. Due to the large number of flights, planning is complex and therefore is divided into several stages. Schedule development, i.e. where and when to fly, comes first. Next is *fleet assignment* (FAM), where an assignment of fleets (equipment types) to flights is made in order to maximize potential revenue. After FAM has been solved, the problems that follow decompose by fleet. In aircraft routing, an aircraft is assigned to each flight. Given a fleet and the corresponding plane routes, the next step is *crew scheduling*, which consists of finding crew itineraries or pairings. The last step, called *rostering*, is the assignment of crews to crew itineraries. Some recent literature that presents the individual models is Hane et al. (1995) for fleet assignment, Clarke et al. (1997) for aircraft routing, Barnhart et al. (1999) for crew scheduling, and Gamache and Soumis (1998) for crew rostering. Yu (1998) contains a collection of articles on airline planning and operations.

The five problems, schedule development, FAM, aircraft routing, crew scheduling and rostering, are solved separately. Ideally all five problems should be solved as a single problem, but this is not feasible computationally. Here we take some steps toward an integrated approach. Our goal is to solve the crew scheduling problem, but we assume that crew scheduling is solved before aircraft routing and, in addition, that the flight departure times are not fixed. In order to solve crew scheduling before aircraft routing we add additional constraints to the crew scheduling model, which provide necessary conditions for the aircraft routing problem to be feasible. Each flight has a time window and the final departure time must be within that time window. By assuming that crew scheduling is solved before aircraft routing, we are able to obtain solutions to the modified crew scheduling problem that are significantly better than solutions obtained by using the traditional crew scheduling model. The retimed flights should not have a big impact on the quality of the schedule and the FAM solution since the time windows considered are small.

The rest of the paper is organized as follows. In Section 1 we explain the constraints that have to be added to the crew scheduling problem in order to meet plane count constraints. The time window aspect of pairings is described in Section 2. Section 3 describes the solution methodology. The resulting problem is a set partitioning problem with side constraints and we show the extra steps required to account for the side constraints. Computational results are presented in Section 4. We conclude the introduction with brief descriptions of the fleet assignment, the aircraft routing and the crew scheduling problems.

Major US airlines domestic operations are based on a *hub and spoke* network. High activity airports are called *hubs* and low activity airports are called *spokes*. After the schedule has been built, FAM is solved. FAM has two fundamental sets of constraints: flow conservation and plane count. Flow conservation is represented by a time space network in which there are arcs for each *flight leg* or *segment*, i.e. a nonstop flight. Therefore an arc specifies two *events*, a *departure* and an *arrival*. The constraints that a FAM solution cannot use more aircraft than there exist in a fleet are modeled by introducing ground arcs and the associated variables. A *ground arc* represents a connection between two consecutive events with no flight activity in between. Each fleet has its own set of ground arc variables. The nonnegative ground arc variable counts the number of planes in the fleet on the ground in the time interval defined by the arc. We call the value of such a variable the *ground arc value*. For each fleet the flow conservation constraints state that the number of planes on the ground plus the number of planes arriving must be equal to the number of planes on the ground in the next time interval plus the number of planes departing. The total number of planes in a fleet is the sum of all the ground arc values of those arcs at a specified point in time, e.g. midnight, plus those aircraft that are in the air.

An *aircraft route* is a sequence of flights that are flown by the same aircraft and a *rotation* or a *routing* is a set of aircraft routes that partition all the flights in the schedule. Given a fleet, the *aircraft routing problem* is to find a routing that satisfies the plane count constraints and other constraints mainly related to maintenance, see e.g. Gu et al. (1994) and Clarke et al. (1997). We say that a routing is *plane count feasible* if it satisfies the plane count constraints.

A *plane turn time* is the time needed for a plane to be ready for the next flight after arriving at a gate. We denote by $minTurn$ the minimum plane turn time, which can depend on various factors such as the station and local time, but for simplicity we assume it is a constant. We use a default value $minTurn = 30$ minutes. In the sequel, all times are given in minutes.

A *duty* is a working day of a crew which consists of a sequence of flights and is subject to FAA and company rules. Among other rules, there is a minimum and maximum connection time between two consecutive flights in the duty. A connection within a duty is called a *sit connection*. We denote by $minSit$ the minimum sit connection time. The default value is $minSit = 45$. The minimum sit connection time requirement can be violated only if the crew follows the plane turn, i.e. they do not change planes. The cost of a duty (measured in minutes) is the maximum of three quantities: the flying time, a fraction of the elapsed time, and the duty minimum guaranteed pay.

Crew bases are designated stations where crews must start their first duty and end their last duty. A *pairing* is a sequence of duties, starting and ending at a crew base and with the elapsed time no more than a week. A connection between two duties is called an *overnight connection* or *layover*. We refer to the time of a layover as the *rest*. Similar to sit connection times, there is a lower and an upper bound on the rest. We denote by $minRest$ the minimum allowed rest time ($minRest = 620$ for our data).

The cost of a pairing is also the maximum of three quantities: the sum of the duty costs in the pairing, a fraction of the time away from base and a minimum guaranteed pay times the number of duties. The *excess cost* of a pairing is defined as the cost minus the flying time of the pairing. Note that the excess cost is always nonnegative. The *flight time credit* (FTC) of a pairing is the excess cost times 100 divided by the flying time, i.e. the excess time measured as a percentage of flying time. A pairing is also subject to many FAA rules.

The airline crew scheduling problem is to find a set of pairings that partition all of the segments and minimize excess cost. The daily airline crew scheduling problem is the crew scheduling problem with the assumption that each leg is flown every day of the week. Since in practice, a small number of legs are not operated during weekends, a daily solution needs to be modified somewhat to obtain a weekly solution. The paper deals exclusively with the daily problem. Traditionally a crew scheduling problem is modeled as the set partitioning problem

$$\min\{cx : Ax = 1, x \ \text{ binary}\}, \tag{1}$$

where each variable corresponds to a pairing, $a_{ij} = 1$ if leg i is in pairing j and 0 otherwise, and c_j is the excess cost of pairing j. Note that for the daily problem, a pairing cannot cover a leg more than once since pairings are repeated in the time horizon.

The problem is difficult since the number of pairings, i.e. columns, can be extremely large. The number of pairings varies from about 200,000 for small fleets, to about a billion for medium size fleets and to billions for large fleets. Furthermore since the cost function of a pairing is nonlinear and the legality rules are complex, it is challenging to perform delayed column generation, i.e. generating columns only as they are needed in the optimization algorithm.

There have not been many attempts to integrate planning stages. Barnhart, Lu and Shenoi (1998) present a model that integrates, to some extent, FAM and crew scheduling. The model has a very large number of constraints and therefore is hard to solve. Rexing (1998) presents a FAM model with time windows. His approach significantly differs from ours in the way the columns are generated. He discretizes the time window intervals whereas we generate columns on the fly without discretizing time windows. Another integration of the FAM model and time windows is presented in Desaulniers et al. (1997). They use a set partitioning model with side constraints and solve problems with up to 400 flights. Barnhart et al. (1998) discuss the integration of FAM and aircraft routing by considering strings of flights.

Recently Cordeau et al. (2000) proposed a model that fully integrates crew scheduling and aircraft routing since it produces a feasible crew schedule and a feasible aircraft routing. Their model is solved with branch-and-price, where at each node of the tree the master problem is optimized with Benders decomposition. They report computational results with fleets containing up to 500 flights and a spoke-to-spoke flight network, but it is not clear if the approach is computationally tractable on hub-and-spoke flight networks with many crew bases.

There are also approaches that integrate crew and vehicle scheduling in urban mass transit systems. Haase, Desaulniers and Desrosiers (1998) present a model that minimizes the crew cost and the number of vehicles. Their model is the set partitioning model with side constraints and it is solved with a branch-and-cut-and-price algorithm. Our model is similar except that in our application the number of resources, i.e. aircraft, at any given time in the time horizon is given by FAM. Freling, Huisman and Wagelmans (2000) propose a model that links the crew scheduling formulation with the vehicle scheduling formulation. The model preserves the flow of the vehicles but it does not try to minimize the number of vehicles. Their model resembles the model in Cordeau et al. (2000). Other references on urban mass transit systems can be found in these two papers.

1 Plane Count Constraints

Even though the difference between $minSit$ (45 minutes) and $minTurn$ (30 minutes) is relatively small, judiciously choosing the plane turns can significantly affect the quality of crew scheduling. We performed an experiment on a small fleet consisting of 123 legs. Table 1 shows the effect of the minimum sit connection time on the excess cost. The last column refers to the problem

$minSit$	30	35	40	45	turns
FTC	8.4	8.5	10	12.5	11

Table 1: The impact of the minimum sit time on FTC

with $minSit = 45$ and a given aircraft routing, i.e. the approach used in current methodology. The remaining columns show the objective value if the minimum sit connection time is set to a given number and aircraft routing is neglected. Clearly, it is advantageous to have a minimum sit connection time of 30 minutes. The experiment also indicated that a different routing can significantly reduce the FTC.

The current methodology finds a routing first and then solves the crew scheduling problem. A model that considers crew scheduling as well as aircraft routing would require variables for strings as well as pairings, resulting in a larger formulation. However, since our primary objective is to solve the crew scheduling problem, we will develop a formulation that incorporates the necessary aircraft constraints without using string variables.

Therefore, instead of completely combining the two problems, we solve them sequentially but reverse the order in which they are solved. The advantage of this approach is that the crew cost is high and the impact of a routing on the crew cost can be substantial. Furthermore, the routing problem is primarily a feasibility problem and generally has many feasible solutions. In the remainder of the paper we assume that a routing is not given. Since we do not know the plane turns, any pairing having sit connections shorter than *minSit* can be a feasible pairing assuming that the plane turns are implied by the pairing.

Suppose that the minimum sit connection time equals *minTurn* and we solve the crew scheduling problem under this assumption. Then the pairings in the solution imply some plane turns, namely each connection in a pairing that is shorter than *minSit* forces a plane turn. We call such potential plane turns *forced turns*. Forced turns become part of the input to the routing problem that must be included in feasible routes. Because of the hub and spoke network structure, as long as the number of forced turns is low, it should not be difficult to meet the maintenance requirements. The other remaining significant constraints are the plane count constraints. We show in this section how they are captured in the crew scheduling model.

Example 1. Consider the following scenario shown in Figure 1. Assume that this is the only

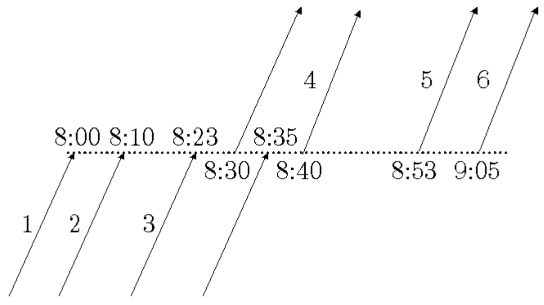

Figure 1: Plane count example

activity at the station and let $minTurn = 30, minSit = 45$. If pairings containing the leg pairs 1-4, 2-5, and 3-6 are in a crew scheduling solution, then they imply 3 forced turns and hence 3 planes on the ground at 8:31. Hence there would have to be one aircraft on the ground at 7:59, and therefore this routing would use more planes than the minimum number. □

1.1 Constraints

The following proposition gives a necessary and sufficient condition for forced turns to be included in a plane count feasible routing.

Proposition 1. *A set of forced turns can be included in a plane count feasible routing if and only if at any point in time the number of planes on the ground imposed by the forced turns is less than or equal to the corresponding ground arc value from the FAM solution.*

Proof. Consider the set of forced turns satisfying the condition in the proposition. Suppose we merge each pair of flights that form a forced turn into a single flight and then adjust the ground arc values accordingly. The new ground arcs and flights still satisfy the flow conservation constraints and the ground arc values are nonnegative. The remaining plane turns can be chosen by a first in first out heuristic. It is easy to see that such a routing is plane count feasible.

Conversely, it can be shown as in Example 1, that if the number of forced turns exceeds the ground arc values, the proposed routing will violate the plane count constraints. □

A FAM solution specifies the number of planes b_g on the ground, for each ground arc $g \in G$ and for each fleet. These ground arcs are defined based on 'ready' times, that is each arrival time is modified by adding the minimum plane turn time to it. For our purposes, it is desirable to define ground arcs based on the original schedule, instead of on the 'ready' times. Given the ground arc values from a FAM solution, it is easy to compute ground arc values based on our definition.

In a daily FAM model where each flight leg is flown every day, the ground arcs G correspond to time intervals within a given 24 hour period. For a ground arc $g \in G$ we use the notation $g + d$ to represent that the ground arc g is shifted by d days in a weekly horizon. We say that a *pairing includes a ground arc* if there is a forced turn within the pairing that contains the time interval represented by the ground arc. Let P be the set of pairings that can be generated from legs in the schedule based on the minimum sit connection time of $minTurn$ minutes. For each $g \in G$, let $P_g \subset P$ be the set of all pairings having a forced turn that includes one of $g, g+1, \ldots, g+6$. Since a ground arc with length greater than or equal to $minSit$ has $P_g = \emptyset$, we only need to consider the subset $G' \subseteq G$ whose elements have length less than $minSit$. Note that a pairing including g and $g + d$ contributes 2 forced turns since it is repeated in the weekly horizon. For each $g \in G'$ and for each $p \in P_g$ define a_{pg} to be the number of times the pairing p includes one of $g, g+1, \ldots, g+6$. The plane count constraints can be written as $\sum_{p \in P_g} a_{pg} x_p \leq b_g$.

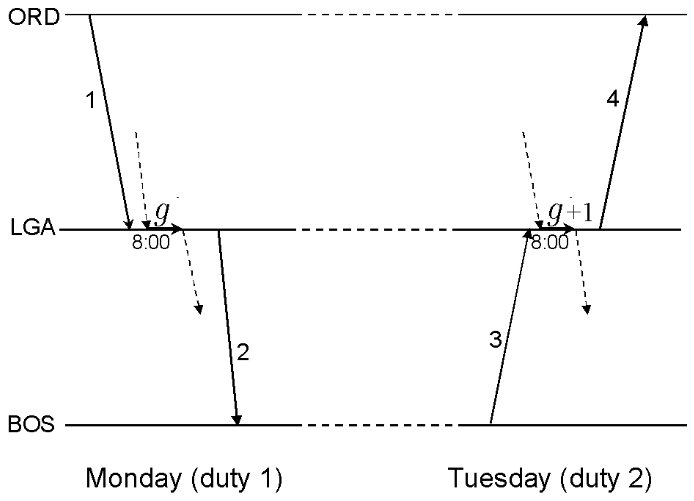

Figure 2: A pairing including a ground arc g and $g + 1$

Example 2. The 2 duty pairing p shown in Figure 2 and consisting of legs 1,2,3,4, includes ground arc g on Monday and ground arc $g + 1$ on Tuesday, therefore $a_{pg} = 2$. □

The new model, which we call the *crew scheduling model with plane count constraints* (CSPC) can be formulated as

$$\min \sum_{p\in P} c_p x_p$$

$$\sum_{p\in P^i} x_p = 1 \qquad \text{for each leg } i \tag{2}$$

$$\sum_{p\in P_g} a_{pg} x_p \le b_g \qquad \text{for each } g \in G' \tag{3}$$

$$x \ \text{ binary,}$$

where P^i is the set of all pairings covering the leg i and $x_p = 1$ if pairing p is selected.

The constraints (2) are the usual set partitioning constraints. We call the constraints (3) the *plane count constraints*. Note that if $b_g = 0$, we can remove all the pairings in P_g from P and the inequality becomes redundant. A solution to this problem provides a crew schedule. The forced turns implied by the solution need to be included in a feasible routing if that is possible. Experience indicates that the forced turns implied by the crew scheduling solution generally do not eliminate all feasible routings.

Example 3. Consider the scenario from Example 1. The resulting plane count constraint derived from the ground arc [8:30,8:35] is

$$\sum_{\substack{p\in P^1 \\ p\in P^4\cup P^5\cup P^6}} x_p + \sum_{\substack{p\in P^2 \\ p\in P^4\cup P^5\cup P^6}} x_p + \sum_{\substack{p\in P^3 \\ p\in P^4\cup P^5\cup P^6}} x_p \le 2\,,$$

assuming that each of the pairings does not include any other 'copy' of the ground arc. □

It can be shown that in FAM the only ground arc variables needed are those that correspond to an outgoing flight followed by an incoming flight, see Hane et al. (1995). For example, ground arc variables corresponding to two incoming flights can be aggregated. Next we state the same result for the plane count constraints (3). A ground arc $g \in G'$ is *essential* if it corresponds to an outgoing flight followed by an incoming flight. Let (at_i, dt_i) be the (arrival, departure) time of leg i.

Theorem 1. *The plane count constraints corresponding to non-essential ground arcs are redundant in the linear programming relaxation of CSPC.*

Proof. Let x be a vector satisfying constraints (2), and (3) for essential ground arcs. We show that x satisfies (3) for all ground arcs in G'. Consider a constraint $\sum_{p\in P_g} a_{pg}x_p \le b_g$ corresponding to a ground arc $g \in G'$ that is not essential. Let s be the station of g. Let $\bar{l}_1$ and $\bar{l}_2$ be the two legs defining g and assume that the activity (arrival or departure) of $\bar{l}_1$ is earlier than the activity of $\bar{l}_2$. We need to distinguish two cases depending on the status of ground arc g.

Case 1.) The arrival station of leg $\bar{l}_1$ is s.

Let $l_1, \ldots, l_r$ be all the legs whose arrival station is s, whose arrival time is earlier than the arrival time of $\bar{l}_1$, and are covered by pairings in P_g. Assume without loss of generality that $at_{l_1} \le at_{l_2} \le \ldots \le at_{l_r}$. We have $\sum_{p\in P^{l_i}} x_p = 1$ for $i = 1, \ldots, r$ since x satisfies (2).

If $r \leq b_g$, then

$$\sum_{p \in P_g} a_{pg} x_p \leq \sum_{i=1}^{r} \sum_{p \in P^{l_i}} x_p \leq r \leq b_g$$

since each $p \in P_g$ is in exactly a_{pg} pairings from $\cup_{i=1}^{r} P^{l_i}$. Note that a pairing cannot cover the same leg twice.

Now suppose that $r > b_g$. In addition assume that in the time interval $\left[at_{l_1}, at_{\bar{l}_1}\right]$ there are no legs departing from s. Let u be the ground arc value of the ground arc g_1 immediately preceeding the leg l_1, see Figure 3. If k legs arrive at s in the interval $\left[at_{l_1}, at_{\bar{l}_1}\right]$, then $r \leq k$ and $u + k = b_g$. Then $u = b_g - k \leq b_g - r < 0$, which is a contradiction since ground arc values are nonnegative.

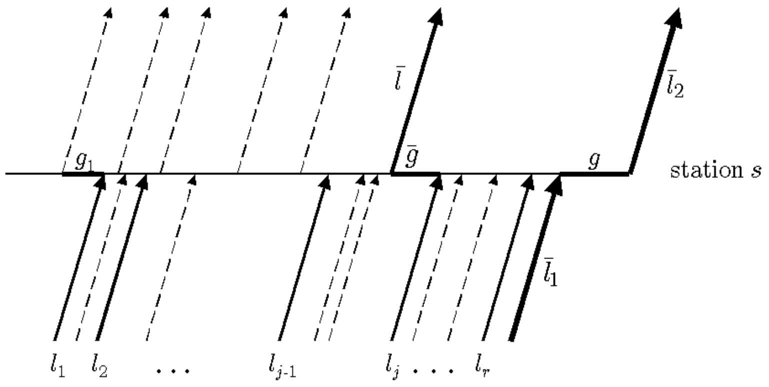

Figure 3: Timeline for the proof of Theorem 1

Hence there is a leg $\bar{l}$ departing from s and $dt_{\bar{l}}$ is in the interval $\left[at_{l_1}, at_{\bar{l}_1}\right]$. Among all such legs we select the leg $\bar{l}$ that has the latest departure time, i.e. there are no legs departing from s in the interval $\left[dt_{\bar{l}}, at_{\bar{l}_1}\right]$. If $dt_{\bar{l}} = at_{\bar{l}_1}$ and the departure station of $\bar{l}_2$ is s, then we can apply case 2 below by taking $\bar{l}$ and $\bar{l}_2$ as the legs that define g. If $dt_{\bar{l}} = at_{\bar{l}_1}$ and the arrival station of $\bar{l}_2$ is s, then g is essential, which is a contradiction. Hence we can assume that $dt_{\bar{l}} < at_{\bar{l}_1}$. Let $\bar{g}$ be the ground arc immediately following $\bar{l}$. Then $\bar{g}$ is essential and hence x satisfies the constraint

$$\sum_{p \in P_{\bar{g}}} a_{p\bar{g}} x_p \leq b_{\bar{g}} \, . \tag{4}$$

Let $j, 1 \leq j \leq r$ be such that $at_{l_{j-1}} \leq dt_{\bar{l}} < at_{l_j}$ and let $P_g = \bar{P}_g \cup \bar{\bar{P}}_g$, where $\bar{P}_g$ is the set of pairings covering at least one leg whose arrival time is before $dt_{\bar{l}}$ and $\bar{\bar{P}}_g$ is the set of pairings covering legs only in $\{l_j, \ldots, l_r\}$. If such a j does not exist, then the set $\bar{P}_g$ is empty. Then it is clear that $\bar{P}_g \subseteq P_{\bar{g}}$. Let c be the total number of legs with arrival time in $\left[dt_{\bar{l}}, at_{\bar{l}_1}\right]$. Then $b_{\bar{g}} + c = b_g$ and $r - j + 1 \leq c$.

Let $p \in \bar{P}_g$ and $a_{pg} = \tilde{a}_{pg} + \tilde{\tilde{a}}_{pg}$, where $\tilde{a}_{pg}$ is the number of times the pairing p includes either one of $g, g+1, \ldots, g+6$ and the sit connection involves a leg from $\{l_1, \ldots, l_{j-1}\}$. Then $\tilde{a}_{pg} \leq a_{p\bar{g}}$.

It follows that

$$\begin{aligned}\sum_{p\in P_g} a_{pg}x_p &= \sum_{p\in \bar{P}_g} \tilde{a}_{pg}x_p + \sum_{p\in \bar{P}_g} \tilde{\tilde{a}}_{pg}x_p + \sum_{p\in \bar{\bar{P}}_g} a_{pg}x_p \\ &\leq \sum_{p\in P_{\bar{g}}} a_{p\bar{g}}x_p + \sum_{p\in \bar{P}_g} \tilde{\tilde{a}}_{pg}x_p + \sum_{p\in \bar{\bar{P}}_g} a_{pg}x_p \\ &\leq b_{\bar{g}} + \sum_{i=j}^{r} \sum_{p\in P^{l_i}} x_p \leq b_{\bar{g}} + r - j + 1 = b_g - c + r - j + 1 \leq b_g\ .\end{aligned}$$

The first inequality follows from $\bar{P}_g \subseteq P_{\bar{g}}$ and $\tilde{a}_{pg} \leq a_{p\bar{g}}$ for all pairings $p \in \bar{P}_g$. The second inequality holds since a pairing $p \in \bar{P}_g$ is in $\tilde{\tilde{a}}_{ap}$ pairings from $\cup_{i=j}^{r} P^{l_j}$, a pairing $p \in \bar{\bar{P}}_g$ is in a_{pg} pairings from $\cup_{i=j}^{r} P^{l_j}$, and from (4).

Case 2.) The departure station of $\bar{l}_1$ and $\bar{l}_2$ is s.

Let $l_1, \ldots, l_r$ be all the legs whose departure station is s, whose departure time is later than the departure time of $\bar{l}_2$, and are covered by pairings in P_g. Assume without loss of generality that $dt_{l_r} \geq dt_{l_{r-1}} \geq \ldots \geq dt_{l_1}$.

Let $\bar{l}$ be the first leg arriving at the station s and $at_{\bar{l}} \geq dt_{\bar{l}_2}$. Let $\bar{g}$ be the ground arc immediately preceeding $\bar{l}$. Now we can apply similar arguments as those in the first case to complete the proof. □

As a result of Theorem 1, the number of necessary plane count constraints can be significantly reduced to only those that correspond to essential ground arcs. So their addition to the standard crew scheduling model should not cause significant computational difficulties.

2 Time Windows

Here we assume that the schedule is not yet fixed, in the sense that we are allowed to make very small changes in the departure time of each leg. For obvious reasons it would not make sense to consider 'big' changes in departure times. For the remainder of the paper let $2w$ be the size of the time window in minutes. Namely, the revised departure time of a leg i must be in the time interval $[dt_i - w, dt_i + w]$. The *offset* of leg i is the revised departure time minus the original departure time dt_i. A typical value for w is 5 or 10 minutes. We assume a default value $w = 5$. For simplicity we assume that the window size does not depend on the leg index but the approach can be easily generalized to handle such a dependency.

The output of the *crew scheduling problem with time windows* is a set of departure offsets and a set of pairings that partition the legs and are feasible based on the retimed schedule. The flexibility in departure times should allow pairings that are infeasible based on the original schedule to become feasible. For example, if two legs are separated by 20 minutes, they can be part of a pairing if the departure time of each one of them is adjusted by 5 minutes in the final retimed schedule. Therefore, we expect better objective values to be mostly due to the increased number of feasible pairings rather than to the change in the cost of a pairing if its legs are perturbed. In addition to capturing more short sit connections, shorter layover times can increase the number of possible pairings as well. Many pairings that are disregarded because they violate the 8-in-24 rule might become feasible if we retime the legs. Additional pairings also can be captured by extending the maximum sit connection time by $2w$, but we do not address this possibility here since such a duty would have a high cost and hence it is unlikely that it would be part of a good solution. However, the techniques presented can be easily extended to allow this extension.

We define a *duty* as a sequence of flights that satisfy all the FAA and company rules based on the original schedule and the modified pairing feasibility parameters $minSit = minSit - 2w$ and the maximum duty elapsed time is increased by $2w$.

A *feasible pairing with respect to a given feasibility rule* is a sequence of duties, starting and ending at a crew base, together with offsets of the legs such that the given feasibility rule is satisfied with respect to the departure times defined by the offsets. In what follows a *pairing* is a feasible pairing with respect to all of the feasibility rules, specifically the minimum and the maximum sit and rest connection times, the maximum duty elapsed time, and the 8-in-24 rule, and a single set of offsets for all of the feasibility rules. Assume we modify the following pairing feasibility parameters: $minSit = minSit - 2w$, $minRest = minRest - 2w$, the maximum duty elapsed time is increased by $2w$, and the minimum allowed compensatory rest is reduced by $2w$ (see Section 2.2 for the definition of the compensatory rest). A *potential pairing* is a sequence of duties, starting and ending at a crew base, such that all of the feasibility rules are satisfied based on the original schedule and the modified pairing feasibility parameters. Note that every pairing is also a potential pairing, but the converse is not true. A duty consisting of two consecutive connections of 20 and 25 minutes cannot be part of a pairing since there is no way to retime the 3 involved legs to meet the minimum sit connection requirement of 30 minutes, however it can be part of a potential pairing. On the other hand, a duty with two consecutive 20 and 30 minute connections can be part of a pairing since we can retime the three legs to have the sit connection times longer than 30 minutes.

We generate potential pairings and during the generation we compute new departure times of legs in a potential pairing such that at the end we produce a pairing. If a partial potential pairing cannot be extended, it is pruned. With the parameters given above, every pairing can be generated. Because of the hub-and-spoke flight network structure and several crew bases for large fleets, all of the pairings cannot be generated in a reasonable amount of time. Instead, we generate subsets of random pairings as proposed in Klabjan et al. (1999).

There are two possible approaches to pairing generation: one generates pairings directly from legs and the other generates duties first and then pairings are constructed from duties. For a more comprehensive discussion of pairing generation see Klabjan (1999). Here we generate pairings from duties by depth-first search.

2.1 Generating Feasible Pairings

We first show how to generate feasible pairings with respect to connection times and duty elapsed times. For the time being we ignore the 8-in-24 rule.

For a potential pairing having l legs, let c_i be the connection time between the ith and the $(i+1)$th leg in the original schedule, $i = 1, \dots, l-1$. Note that $c_i \geq minTurn - 2w$ or $c_i \geq minRest - 2w$ depending on the type of connection. Define $m_i, i = 1, \dots, l-1$ to be $minTurn$ if the connection i is a sit connection and $minRest$ otherwise.

Example 4. Consider the potential pairing in Figure 4 depicted in bold. The connection times are listed next to the connections. Legs (3,4,5) can be retimed to make feasible connections and the same is true for legs (2,3,4), however we can not retime legs (2,3,4,5) to form feasible connections. To see this, we can attempt to 'stretch' the connections starting with leg 2. We can move it 5 minutes earlier (dashed flight legs in the figure) and then try to make the next connection as short as possible. We proceed in this manner until we reach leg 5, which would have to be moved by 6 minutes, thus violating the time window. A formal reason for not being able to retime legs (2,3,4,5)

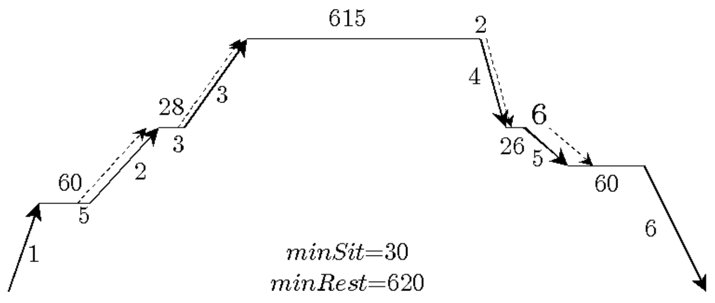

Figure 4: An example of a potential pairing that cannot be retimed

is that the connection time deficit $\sum_{i=2}^{4}(c_i - m_i) = -11$ cannot be compensated for by moving the departure time of the second leg 5 minutes earlier and the departure time of the fifth leg 5 minutes later. No matter how large the connection times are between the legs 1,2 and 5,6, we can not retime the whole potential pairing. The potential pairing in the figure is not a pairing. □

We start with a proposition that addresses the connection times issue. The proposition is presented in a more general setting, namely window sizes depend on the leg index, which will be needed later.

Proposition 2. *Let the sequence of legs in a pairing be given by $(1, 2, \ldots, l)$ and assume that each leg i has a window size w_i and that $c_i \geq m_i - w_i - w_{i+1}$ for each index i. The potential pairing is a feasible pairing with respect to connection times if and only if*

$$\sum_{j=s}^{i}(c_j - m_j) + w_s + w_{i+1} \geq 0 \tag{5}$$

for all $1 \leq s \leq i \leq l-1$.

Proof. We first prove the necessity of (5). Assume that each leg has an offset x_j such that the feasible pairing satisfies connection time requirements based on the offsets, i.e. the departure time of the leg j is $dt_j + x_j$, $-w_j \leq x_j \leq w_j$. Then $m_j \leq c_j + x_{j+1} - x_j$ for each $1 \leq j \leq l-1$. Note that the right hand side of the inequality is the connection time of the pairing and hence by definition is larger than m_j. Summing the inequalities from s to i and using the time window bounds, we get the claim.

The sufficiency is proved algorithmically by constructing leg offsets such that the new departure times are as early as possible and they yield a feasible pairing with respect to connection times.

Algorithm 1 computes a set of offsets x. We claim that the given offsets satisfy the time window restrictions and the minimum connection time requirements.

It is easy to see that the computed connection time is always greater than or equal to m_{i-1}. We still need to show that x_i is within the time window using the assumption given in the proposition. Clearly, $x_i \geq -w_i$. By induction it follows that either $x_i = -w_i$ or there is an index $s, 1 \leq s \leq i-1$ such that $x_i = \sum_{j=s}^{i-1}(m_j - c_j) - w_s$. In the first case, $x_i \leq w_i$. In the second case, the claim follows directly from the assumption in the proposition. □

Algorithm 1: Feasible Connection Time Pairing

```
1: Let $x_1 = -w_1$.
2: for $i = 2$ to $k$ do
3:     if $c_{i-1} - x_{i-1} \geq m_{i-1} + w_i$ then
4:         $x_i = -w_i$
5:     else
6:         $x_i = m_{i-1} + x_{i-1} - c_{i-1}$
7:     end if
8: end for
```

Note that the proof of the proposition also establishes a linear time algorithm for computing the offsets or detecting infeasibility. An infeasibility occurs whenever a computed offset is not within the time window.

We have already indicated that the duties are generated first. Consider a duty d having k legs and a potential pairing containing the duty. No matter what the offsets of the first and the last legs of the duty in the pairing are, the inequalities $\sum_{j=s}^{i}(c_j - minTurn) + 2w \geq 0$ must hold for all $2 \leq s \leq i \leq k-2$. So all the duties violating one of these inequalities must be removed.

Algorithm 1 is a fast procedure for generating feasible pairings with respect to the connection time requirements, however such pairings do not necessarily satisfy the maximum duty elapsed time bounds (or the 8-in-24 rule). If duty elapsed time bound was violated, new offsets would have to be computed, adding to the already computational intensive pairing generation. Instead we compute the offsets of a duty that we are attempting to append to a partial pairing in such a way that the maximum duty elapsed time is not violated (if possible). The key idea is to push the departure times of the new duty as early as possible but still be within the time window.

We first derive the explicit formula for the offset of the last leg in a duty given an offset of the first leg of the duty and assuming Algorithm 1 is applied. With each duty having k legs we define the following quantities:

$$\bar{\alpha}_d = \min_{j=1,\ldots,k-2} \sum_{i=1}^{j}(c_i - minTurn) + w$$

$$\bar{\beta}_d = \min_{j=2,\ldots,k-1} \sum_{i=j}^{k-1}(c_i - minTurn) + w$$

$$\gamma_d = \sum_{i=1}^{k-1}(c_i - minTurn)\,.$$

Observe that if the offset of the first leg of the duty is x and the offset of the last leg is y, then

$$x \leq \bar{\alpha}_d, y \geq -\bar{\beta}_d, x - y \leq \gamma_d\,. \tag{6}$$

These conditions follow from Proposition 2 if we assume that $w_0 = 0, w_k = 0$, the departure time of the first leg l_1 in the duty is $dt_{l_1} + x$, and that the departure time of the last leg l_k in the duty is $dt_{l_k} + y$. Since any feasible offset must satisfy $x \leq w$ and $y \geq -w$, we define $\alpha_d = min(\bar{\alpha}_d, w)$ and $\beta_d = max(-\bar{\beta}_d, -w)$.

Proposition 3. *If the offset of the first leg of the duty is x, then the offset of the last leg in the duty is*

$$y = max(\beta_d, x - \gamma_d) \; , \tag{7}$$

if Algorithm 1 is used.

Proof. It is easy to see that

$$y = \begin{cases} -w & \text{or} \\ \sum_{i=j}^{k-1}(c_i - minTurn) + w & \text{for an index } j, 2 \leq j \leq k-1, \text{ or} \\ x - \gamma_d \; . \end{cases}$$

From (6) we know that $y \geq max(\beta_d, x - \gamma_d)$. Combining the two observations yields the claim. □

Now we are ready to describe the generation of feasible pairings with respect to connection times and duty elapsed times. We assume that the maximum duty elapsed time is a constant $maxElapse$, for a more general maximum duty elapsed time function see Klabjan (1999).

The pairing generation routine only keeps track of the offsets of the first and the last leg in a duty. Assume that we have a partial pairing consisting of duties $d_1, \ldots, d_{j-1}$ and we want to append a duty d. Let $(x_1, y_1), \ldots, (x_{j-1}, y_{j-1})$ be the computed offsets. We want to derive the offsets (x_j, y_j) such that the partial pairing extended with the duty d satisfies the minimum and the maximum sit and rest connection times, and the maximum duty elapsed time based on the computed offsets.

We would like to avoid backtracking when recomputing the new offsets of the already appended duties. In order to achieve this, we generate the utmost left pairing. A pairing is the *utmost left pairing* if it is a pairing based on offsets $x_1, \ldots, x_j$, and for any time t and index $i, i \leq j$, the pairing with offsets $x_1, \ldots, x_{i-1}, x_i - t, x_{i+1}, \ldots, x_j$ violates either the maximum duty elapsed time bound or a minimum connection time limit. Hence as soon as we move a departure time one time unit earlier, the pairing violates one of the two feasibility rules.

In addition to computing the new offsets (x_j, y_j) in such a way that the new partial pairing satisfies the feasibility rules we need to preserve the utmost left property. We assume that the current partial pairing is the utmost left one. Let m_{j-1} be either the minimum rest time or the minimum compensatory rest depending on the 8-in-24 violation flag in the previous step (see the next section for the discussion of the compensatory rest). Define

$$\delta_j = \begin{cases} -w & \text{if } c_{j-1} - y_{j-1} \geq m_{j-1} + w \\ m_{j-1} + y_{j-1} - c_{j-1} & \text{otherwise.} \end{cases}$$

Note that δ_j is determined by performing one step of Algorithm 1. Combining the above definition and the inequalities (6), the new offsets have to satisfy the inequalities

$$\delta_j \leq x_j \leq \alpha_d \tag{8}$$

$$\beta_d \leq y_j \leq w \tag{9}$$

$$x_j - y_j \leq \gamma_d \; . \tag{10}$$

The above inequalities guarantee that the new partial pairing based on the offsets will have connection times that are bigger than the required minimum. We still need to take care of the maximum duty elapsed time and the utmost left property. Assume that e_d is the elapsed time of the duty d based on the original schedule and let $\tilde{e}_d$ be the elapsed time of the retimed duty.

Then it is clear that $\tilde{e}_d = e_d - x_j + y_j$. The elapsed time $\tilde{e}_d$ has to be smaller than or equal to $maxElapse$. Hence we get an additional inequality

$$\hat{e}_d \leq x_j - y_j \; , \tag{11}$$

where $\hat{e}_d = e_d - maxElapse$. The new offsets have to satisfy the system of inequalities (8)-(11), denoted by Q.

We claim that if the system Q is infeasible, then we cannot append the duty d. Suppose there were a set of offsets $(\tilde{x}_1, \tilde{y}_1), \ldots , (\tilde{x}_j, \tilde{y}_j)$ such that the partial pairing $\{d_1, \ldots , d_{j-1}, d\}$ satisfies the feasibility rules. Then the offsets $\tilde{x}_j$ and $\tilde{y}_j$ have to satisfy (9), (10), and (11). Since $\tilde{x}_j \leq \alpha_d$ it must be the case that $\tilde{x}_j < \delta_j$. Due to the definition of δ_j it follows that $\tilde{y}_{j-1} < y_{j-1}$. But this contradicts the utmost left property of the partial pairing and the computed offsets.

Assume now that the system Q is feasible. We can explicitly compute a solution to the system that minimizes x_j using Fourier-Motzkin elimination (see e.g. Schrijver (1986)) given by

$$x_j = \max\left(\delta_j, \beta_d + \hat{e}_d\right) \tag{12}$$

$$y_j = \max\left(\beta_d, x - \gamma_d\right) \; . \tag{13}$$

This solution has the smallest x_j and the corresponding y_j is the one listed in Proposition 3. If we start with the offset x_j and apply Algorithm 1, then the resulting y_j is given by (13). Clearly the algorithm produces the utmost left sequence of offsets. Hence the values given by (12) and (13) maintain the property of being the utmost left.

To summarize, we compute the values x_j and y_j from the formulas (12) and (13) and then check inequalities (8)-(11). If at least one is violated, then the duty d is discarded. Else we append the duty d and impose the corresponding offsets (x_j, y_j).

In the US, the FAA requires that pairings satisfy the 8-in-24 rule, which says that if in a 24 hour time window there is more than 8 hours of flying, then the next rest, called a *compensatory rest*, must be longer than a given limit. Different flight departure times can cause a violation of the rule and therefore care has to be taken when time windows are present. The treatment of the 8-in-24 rule and time windows can be done efficiently as described in Klabjan (1999).

2.2 8-in-24 Rule and Time Windows

The 8-in-24 rule is imposed by the FAA. Consider two consecutive duties d_1 and d_2 in a pairing. If the flying time in d_1 and d_2 in any 24 hour period is more than 8 hours, there is an 8-in-24 violation. The rule says that if there is an 8-in-24 violation, then the layover following the duty d_2, called a *compensatory rest*, must be longer than a given limit, typically around 850 minutes. The airlines usually make the rule more stringent so that they have more flexibility in operations.

Our goal is to capture all pairings that are feasible based on the original schedule. Suppose in the time windows model, we stretch the 24 hour window into a $24 \cdot 60 + 2w$ minutes time window. A pairing satisfying the 8-in-24 rule based on such a window in the original schedule, is 8-in-24 rule feasible for all retimings of legs. Similarly, if there is a violation based on the $24 \cdot 60 - 2w$ time window, then there is a violation for any retiming of legs. These two cases determine the connection time requirement for the next layover.

For the remaining case, where there is a violation based on the $24 \cdot 60 + 2w$ time window and there is no violation based on the $24 \cdot 60 - 2w$ time window, some retimings may have a violation while others may not. Since we do not want to prune any pairing, we treat this case as if there is no violation, but a special flag is set. If the flag has not been set during the generation, then such a pairing is 8-in-24 feasible for any retiming.

If the layover connection time based on the computed offsets (using the algorithm from the previous section) is longer than the minimum required compensatory rest, then we can reset the flag to the default value. In this case, regardless of the 8-in-24 violation status in the previous generation step, the current overnight connection time is long enough.

On average the flag is set for about 20% of the pairings. Given a flagged pairing we first check if it is feasible based on the offsets computed during the generation. In most cases it is feasible. If the computed offsets are not feasible, then we first try to stretch the overnight rest following a violation. If it cannot be made longer than the minimum compensatory rest, then we try to get rid of the 8-in-24 violation.

Suppose that duties d_1 and d_2 have offsets (x_1, y_1) and (x_2, y_2), respectively, and there is an 8-in-24 violation based on these offsets. At this point we have to compute new offsets that remove the violation. If such offsets do not exist, then the pairing is discarded. In order to reduce the flying time in a 24 hour time window, we have to either push the departure time of the first leg in the window earlier or the arrival time of the last leg in the window later. Since the first leg in the window is always in the duty d_1 and the pairing is utmost left feasible, we cannot move its departure time earlier. Hence, we have to change the departure time of the last leg in the window, which is always in d_2. It is easy to see that when checking the 8-in-24 violation it suffices to consider only 24 hour time windows that start at departure times of legs in the duty d_1.

The idea is to compute new lower bounds for the leg offsets in the duty d_2 and then to recompute the offsets using the algorithm from the previous section. For computing new time windows consider Figure 5. The departure time of each leg in the figure is assumed to be already retimed with respect to the offsets computed by the algorithm from the previous section. Let f be the flying time in the 24 hour time window depicted in the figure and let $\tau = f - 8 \cdot 60$. Since there is a 8-in-24 violation in the time window, $\tau > 0$.

Scenario 1: The end of the 24 hour time window is during a flight l in the duty d_2, see Figure 5. If $\tau > h_j$, the pairing is discarded. Otherwise the departure time of the leg l has to be moved later by at least τ.

Scenario 2: The end of the 24 hour time window is during the sit connection following a leg l, see Figure 5. The departure time of the leg l has to be moved later by at least $\tau + g_j$.

We compute the new time windows by iterating the procedure for all legs j in the duty d_1. Based on these new time windows for legs in the duty d_2, we apply the algorithm from the previous section. If the new offsets do not exist, the pairing is discarded.

Note that to compute τ, h and g it suffices to scan the legs in the duty just once by starting with the first leg. Since Algorithm 1 scans the legs of a duty in the same order, all computations for finding new offsets can be performed in a single pass through all the legs in the duties d_1 and d_2.

For small fleets the exact handling of the 8-in-24 rule described in this section is necessary. However, for bigger fleets, we discard the pairing if it is not 8-in-24 feasible based on the computed offsets by applying the algorithm from the previous section.

Some airlines still use an old 8-in-24 rule which additionally requires that the rest between two duties having an 8-in-24 violation (*included rest*) has to be at least a given limit. Such a modification can be easily incorporated into the algorithm.

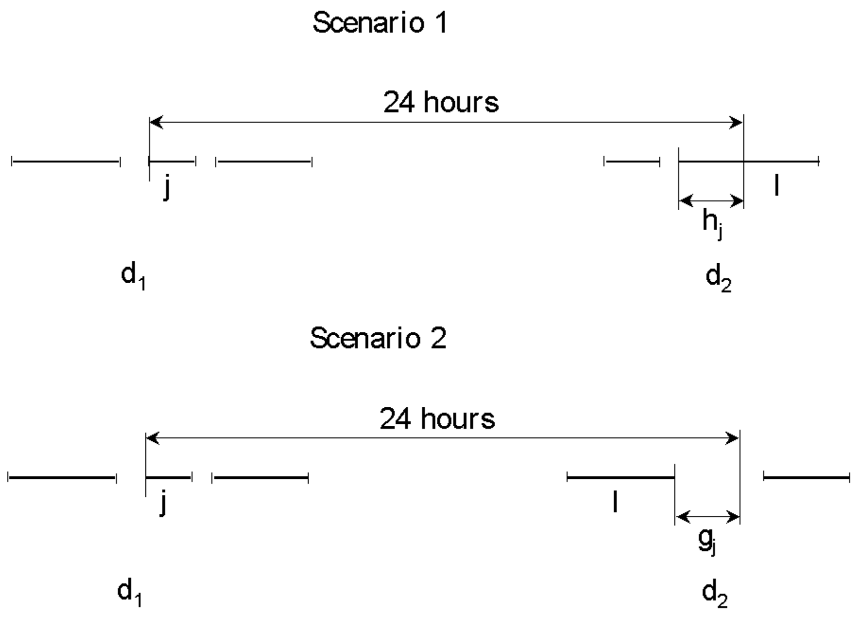

Figure 5: 8-in-24 rule and time windows

3 Solution Methodology

We outline the overall methodology of integrating the plane count constraints and time windows into the crew scheduling model.

1. We generate potential pairings based on the original schedule but with some pairing feasibility parameters modified. Namely, the minimum sit and layover time is decreased by $2w$, the maximum duty elapsed time is increased by $2w$, and the minimum compensatory rest is reduced by $2w$. Since we include the plane count constraints, the minimum sit connection time is the minimum plane turn time. We use the algorithms from Section 2 within the generation routine for obtaining pairings.

With each generated pairing, we get a sequence of leg offsets such that the pairing is feasible on the retimed legs. Even though a pairing may have more than one retiming we consider only one, namely the one given by the generation routine. We do not try to find a retiming of legs that produces the lowest cost pairing since this is a time consuming operation and it would not bring substantial additional savings.

2. Next we solve the crew scheduling model with plane count constraints by considering only the generated pairings. Each pairing in the solution implies a set of departure time offsets.

Because the leg offsets can change the set of ground arcs, capturing all the plane count constraints exactly is hard. The approach described below approximates the plane count constraints since it may not find all of the pairings contained in a ground arc of length less than $2w$. We use the set of ground arcs from the FAM solution and there is a plane count constraint for each essential ground arc of length less than $minSit + 2w$. We need to redefine when a pairing includes a ground arc. Consider a pairing implying the offsets x of the legs in the pairing. The pairing includes a ground arc g defined by legs $\hat{l}_1$ and $\hat{l}_2$ if there is a sit connection in the pairing, defined by legs $\tilde{l}_1$ and $\tilde{l}_2$, such that $dt_{\tilde{l}_2} + x_{\tilde{l}_2} - at_{\tilde{l}_1} - x_{\tilde{l}_1} < minSit$ and $at_{\tilde{l}_1} + x_{\tilde{l}_1} \leq dt_{\hat{l}_1} + w, dt_{\tilde{l}_2} + x_{\tilde{l}_2} \geq at_{\hat{l}_2} - w$, see Figure 6. The first condition states that the sit connection implies a forced turn and the last two say that the pairing includes the ground arc even if the legs $\hat{l}_1, \hat{l}_2$ defining g are moved as close together as possible. With this definition, we capture exactly the plane count constraints for

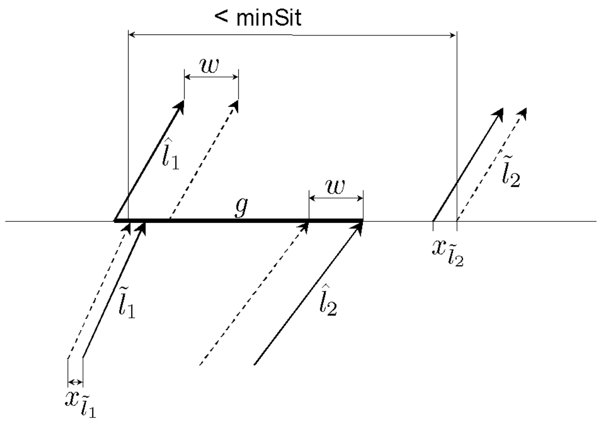

Figure 6: The new definition of inclusion

ground arcs of length greater than $2w$. However if the length is less than $2w$, then some pairings might be left out of P_g.

3. The plane count given by the pairing solution can be increased due to the approximate handling of some of the plane count constraints. The increased plane count can only occur if in the solution a leg defining an essential ground arc is swapped in time with an incoming flight. If the solution implies a bigger plane count, then we attempt to retime the schedule again, this time only using pairings from the solution.

Suppose that the arrival time of leg i is before the departure time of leg j in the original schedule and that in the retimed schedule the order of the two times is reversed and it yields a higher plane count. We have to push the arrival time of leg i earlier or the departure time of leg j later. The former is not possible due to the utmost left property of pairings. Hence the departure time of leg j, or some other leg k, has to be pushed forward, past the new arrival time of leg i. Note that leg j does not need to be the first leg following leg i. For example, if $at_i < dt_j < dt_k$ and retiming of leg j fails, we can try to retime leg k.

Experiments have shown that there are not many stations with an increased plane count. Even when there was an increased plane count, the above procedure was able to retime the legs. The smaller window size $w = 5$ never yielded an increased plane count.

4. If the plane count cannot be adjusted with local changes in the departure times, then we would add a constraint forbidding the two involved pairings to be selected simultaneously. The problem is then reoptimized. In our experiments this was never observed.

The LP based branch-and-bound methodology for solving the crew scheduling problem with time windows and plane count constraints, namely steps 1 and 2 above, closely follows the algorithm presented in Klabjan et al. (1999). It is not discussed here.

4 Proof of Concept

All computational experiments were performed on 4 fleets, 2 small ones with 100-200 legs and 2 larger ones with 300-450 legs. Cases 1,2,3,4 refer to the 4 fleets with case 1 corresponding to the smallest fleet and case 4 to the largest. The number of crew bases varies from 3 to 5. The number of pairings, i.e. variables, for the first two problems is approximately half a million. However, due to the hub-and-spoke flight network and several crew bases, this number is several billion for the last two problems. We used the same feasibility rules and cost function as the airline. The only approximation to the real data is the minimum plane turn times, where we used a constant value of 30 minutes since the real values (depending on the time and station) were not available.

Table 2 summarizes the solution qualities represented by FTC (percentage of excess cost above flying) and the number of forced turns. FTC generally decreases with larger fleet size in a hub-and-spoke network since larger fleets yield many more connection opportunities. The "CS" column refers to the traditional crew scheduling model. All the time windows variants have plane count constraints. The column "$w = 0$" stands for the crew scheduling problem with plane count constraints but without time windows. For the biggest fleet we did not perform the time window variants since the solution with $w = 0$ has FTC of almost zero. The flexibility with respect to forced turns improves the FTC substantially, typically by a factor of two. Time windows improve the solution by an additional 25%.

The solutions with plane count constraints generally have more forced turns. A larger number of forced turns and the freedom to select them explain the improved FTC. Increased window size also gives more potential forced turns and therefore solutions with larger time windows use more forced turns. Note also that for robustness reasons a larger number of forced turns is desirable. If a crew does not follow the plane turn, then a disruption of a flight can occur either because of a 'late' plane or crew. Some airlines even give an artificial bonus to pairings with plane turns by reducing their cost.

Cases	FTC				# forced turns			
	CS	$w = 0$	$w = 5$	$w = 10$	CS	$w = 0$	$w = 5$	$w = 10$
1	3.94	2.21	1.71	1.35	2	9	10	10
2	3.12	1.85	1.54	1.06	11	11	12	17
3	2.86	1.40	0.88	0.88	17	59	70	70
4	0.31	0.08	-	-	66	142	-	-

Table 2: Solution qualities

The relative values of the IP/LP gaps defined by $\frac{100(\text{IP obj} - \text{LP obj})}{\text{LP obj}}$ are listed in Table 3. The gaps are larger than for traditional airline crew scheduling problems since the additional plane count constraints typically yield a larger number of fractional variables in the LP relaxations, which makes it harder to find good integer solutions.

The number of plane count constraints is shown in Table 4. We considered only constraints corresponding to essential ground arcs with a positive right hand side. There is no need to use row generation since there are not many constraints. Further in the integer programming phase of the algorithm, in which only a subset of the pairings is considered, almost all of the plane count constraints are redundant.

All computational experiments were performed on a cluster of PCs by using the parallel algo-

Cases	CS	$w = 0$	$w = 5$	$w = 10$
1	11	4	13	3
2	16	14	10	61
3	91	203	234	513
4	422	686	-	-

Table 3: $\frac{100(\text{IP obj}-\text{LP obj})}{\text{LP obj}}$

Cases	# plane count cons.			# plane count cons. for IP		
	$w = 0$	$w = 5$	$w = 10$	$w = 0$	$w = 5$	$w = 10$
1	16	17	18	0	1	1
2	18	19	20	0	2	3
3	75	84	88	0	1	9
4	59	-	-	0	-	-

Table 4: Number of plane count constraints

rithm from Klabjan et al. (1999). The cluster consists of 48 300MHz Dual Pentium IIs linked via 100 MB point-to-point Fast Ethernet. Table 5 gives the computational times. The first two fleets are computationally easy; the execution times are less than an hour. However, the remaining two fleets require 10 to 15 hours. We estimate that an hour is due to extra computation to account for time windows in the pairing generation routine. If a problem is being solved for different values of w, computational time could be reduced by using a warm start since a feasible solution with a time window w is also feasible with a time window $\hat{w} \geq w$.

The results clearly demonstrate that by solving crew scheduling with the addition of plane count constraints before solving aircraft routing, and by considering small time windows for modifying fleet scheduling, it is possible to reduce crew cost substantially. For hub-and-spoke network systems, this may be a good compromise between current practice and a fully integrated model.

Cases	LP				IP			
	CS	$w = 0$	$w = 5$	$w = 10$	CS	$w = 0$	$w = 5$	$w = 10$
1	0.1	0.1	0.1	0.1	0.2	0.2	0.2	0.2
2	0.3	0.3	0.4	0.4	0.5	0.5	0.6	0.6
3	7	7.7	8.4	9	4.1	4.1	4.2	6
4.1	9	9.5	-	-	5.2	5.2	-	-

Table 5: CPU times in hours

5 Acknowledgments

This work was supported by NSF grant DMI-9700285 and United Airlines. Intel Corporation funded the parallel computing environment and ILOG provided the linear programming solver used in computational experiments.

References

Barnhart, C., Boland, N., Clarke, L., Johnson, E., Nehmauser, G. and Shenoi, R. 1998. Flight String Model for Aircraft Fleeting and Routing. *Transportation Science*, **32**, 208–220.

Barnhart, C., Johnson, E., Nemhauser, G. and Vance, P. 1999. Crew Scheduling. In *Handbook of Transportation Science.* R. W. Hall (editor). Kluwer Scientific Publishers, 493–521.

Barnhart, C., Lu, F. and Shenoi, R. 1998. Integrated Airline Schedule Planning. In *Operations Research in the Airline Industry.* G. Yu (editor). Kluwer Academic Publishers, 384–403.

Clarke, L., Johnson, E., Nemhauser, G. and Zhu, Z. 1997. The Aircraft Rotation Problem. In *Annals of OR: Mathematics of Industrial Systems II.* R. E. Burkard, T. Ibaraki and M. Queyranne (editors). Baltzer Science Publishers, 33–46.

Cordeau, J., Desrosiers, J., Soumis, F. and Stojković, G. 2000. Benders Decomposition for Simultaneous Aircraft Routing and Crew Scheduling, *Technical Report G-2000-37*, Cahiers du GERAD.

Desaulniers, G., Desrosiers, J., Dumas, Y., Solomon, M. M. and Soumis, F. 1997. Daily Aircraft Routing and Scheduling. *Management Science*, **43**, 841–855.

Freling, R., Huisman, D. and Wagelmans, A. 2000. Models and Algorithms for Integration of Vehicle and Crew Scheduling, *Technical Report 2000-10/A*, Erasmus University.

Gamache, M. and Soumis, F. 1998. A Method for Optimally Solving the Rostering Problem. In *Operations Research in the Airline Industry.* G. Yu (editor). Kluwer Academic Publishers, 124–157.

Gu, Z., Johnson, E., Nemhauser, G. and Wang, Y. 1994. Some Properties of the Fleet Assignment Problem. *Operations Research Letters*, **15**, 59–71.

Haase, K., Desaulniers, G. and Desrosiers, J. 1998. Simultaneous Vehicle and Crew Scheduling in Urban Mass Transit Systems, *Technical Report G-98-58*, Cahiers du GERAD.

Hane, C., Barnhart, C., Johnson, E., Marsten, R., Nemhauser, G. and Sigismondi, G. 1995. The Fleet Assignment Problem: Solving a Large-Scale Integer Program. *Mathematical Programming*, **70**, 211–232.

Klabjan, D. 1999. *Topics in Airline Crew Scheduling and Large Scale Optimization.* Ph.D. Dissertation, Georgia Institute of Technology.

Klabjan, D., Johnson, E., Nemhauser, G., Gelman, E. and Ramaswamy, S. 1999. Solving Large Airline Crew Scheduling Problems: Random Pairing Generation and Strong Branching, *Technical Report TLI/LEC-99-11*, Georgia Institute of Technology. To appear in *Computational Optimization and Applications.*

Rexing, B. 1998. *Fleet Assignment with Time Windows*, Master's Thesis, Massachusetts Institute of Technology.

Schrijver, A. 1986. *Theory of Linear and Integer Programming.* John Wiley & Sons Ltd.

Yu, G. (editor) 1998. *Operations Research in the Airline Industry*. Kluwer Academic Publishers.