Konrad-Zuse-Zentrum
für Informationstechnik Berlin

Takustraße 7
D-14195 Berlin-Dahlem
Germany

RALF BORNDÖRFER IVAN DOVICA IVO NOWAK THOMAS SCHICKINGER

Robust Tail Assignment

This research was funded by Lufthansa Systems Berlin GmbH.

ZIB-Report 10-08 (May 2010)

Robust Tail Assignment*

Ralf Borndörfer** Ivan Dovica§ Ivo Nowak‡ Thomas Schickinger‡

May 27, 2010

Abstract

We propose an efficient column generation method to minimize the probability of delay propagations along aircraft rotations. In this way, delay resistant schedules can be constructed. Computational results for large-scale real-world problems demonstrate substantial punctuality improvements. The method can be generalized to crew and integrated scheduling problems.

1 Introduction

The *tail assignment problem* is one of the classical planning problems in the operation of an airline, see Yu (1997) [20] and Barnhart, Belobaba & Odoni (2003) [3] for an overview on airline optimization in general, and Grönkvist (2005) [12] and the references therein for a recent survey on tail assignment. It deals with the construction of rotations for individual units of aircraft in order to cover the legs of a flight schedule. Tail assignment is not a cost minimization problem; the number of aircraft that is needed to operate a flight schedule is determined in the preceding fleet assignment step. One rather optimizes aircraft maintenances and, in particular, *buffer times* between successive flights of an aircraft in such a way that delays can be absorbed and the operation becomes more "robust". In fact, the ongoing dissemination of optimization technology is cutting at safety margins, such that deviations between the plan and the actual operation have become an increasingly serious problem in the last decade. Eurocontrol has estimated average delay costs at 72 € per minute, see Eurocontrol (2000) [10], Boydell (2005) [6], and Cook, Tanner & Anderson (2004) [8] for detailed analyses.

*This research was funded by Lufthansa Systems Berlin GmbH.

**Zuse-Institute Berlin, Takustr. 7, 14195 Berlin, Germany, Email borndoerfer@zib.de

§Corresponding author, Charles University, Department of Applied Mathematics, Malostranské námestí 25,11800 Prague, Email ivan@kam.mff.cuni.cz

‡Lufthansa Systems Berlin GmbH, Am Salzufer 8, 10587 Berlin, Germany, Email ivo.nowak@lhsystems.com, thomas.schickinger@lhsystems.com

It is therefore necessary to take the risk of disturbances into account and to make delay costs an explicit optimization objective.

The literature has suggested a number of approaches to this *robust tail assignment problem*. The simplest is based on the notion of *key performance indicators* (KPI), which augment or replace the objective of an ordinary tail assignment problem by some term that heuristically measures robustness and favors, e.g., even distributions of ground times, assignments of certain amounts of buffer time before a leg, etc. Examples of more sophisticated KPIs are swap opportunities (Burke et al. (2007) [7]), cancellation cycles (Rosenberger, Johnson & Nemhauser (2004) [17]), and propagated delays emerging from individual arcs AhmadBeygi, Cohn & Lapp (2008) [2], see also AhmadBeygi et al. (2007) [1] and Bian et al. (2003) [5] for KPI analyses of historical aircraft schedules. Comparing the behavior of a KPI schedule to a traditional schedule via simulation shows that such an approach can indeed lead to a substantial reduction of delays. In principle, approaches of *robust optimization* could also be applied, see Bertsimas & Sim (2003) [4] and Stiller (2008) [19] for a recent overview, but probably at a high cost. This seems to be the reason why, as far as we know, such worst case concepts have not yet been applied to concrete tail assignment problems. From a mathematical point of view, the most satisfying solution method for robust tail assignment problems is *stochastic optimization*. In such an approach, one optimizes the expectation of some delay statistic along individual rotations, using a stochastic model of delay generation and propagation. The methods differ in the way in which the delay propagation is computed. Lan (2003) [14]; Lan, Clarke & Barnhart (2006) [15] "sample" the delay in the sense that they simulate a rotation on a number of historical scenarios; the resulting histogram is fitted with a continuous (log-normal) distribution. This fitting is time consuming. Fuhr (2007) [11] computes the delay analytically using a special class of (Erlang-Exp/Exp-Erlang) distributions, which (up to an approximation error) are closed under convolution. The disadvantage of this method is that the starting distributions can not be freely chosen. Kleywegt, Shapiro & Homem-de Mello (2001) [13] propose a general Monte Carlo approach, which approximates expected values by averages of samples. This method can be, but has not been, applied to the tail assignment problem, such that its potential in this setting remains unclear.

We propose in this paper a novel method to derive the probability of delay propagation (PDP) along an aircraft rotation directly by a numerical computation of convolutions from discretized probability density functions. This algorithm is used as a pricing routine in a column generation approach to the robust tail assignment problem. We show that our PDP method is accurate and computationally efficient. In fact, approximation errors of less than 1% in comparison to a simulation can be achieved at low computational costs. In our scenarios, which are already optimized with respect to KPI methods,

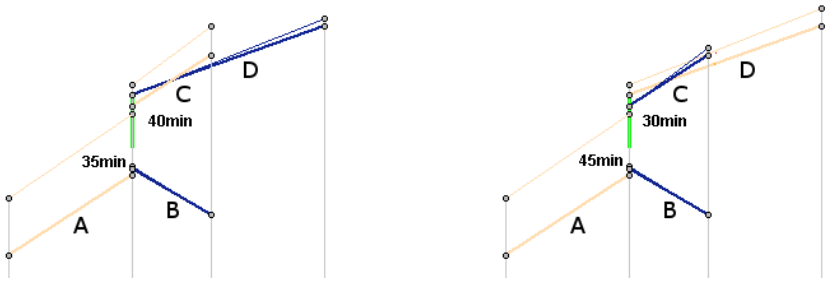

Figure 1: Effect of aircraft routing on delay propagation (thick: plan, thin: day of operation).

PDP optimization achieves additional reductions in delay propagation of up to 5%.

The paper is structured as follows. Section 2 analyzes the propagation of delay along an aircraft rotation. We propose a shortest-path type most robust rotation algorithm to compute a rotation with minimum probability of delay propagation. This algorithm is used as a pricing routine in a column generation approach to the robust tail optimization problem in Section 3. In Section 4, this algorithm is applied to construct robust aircraft rotations for large-scale real-world scenarios covering four months of operation of one subfleet of a European short haul carrier. We compare our PDP approach with a KPI method and prove its validity by directly simulating historical situations.

2 Delay Propagation

On the day of operation, aircraft rotations get disturbed by random events like bad weather, passenger and baggage handling problems, equipment malfunctions, etc. This causes genuine delays of legs, which are inevitable. Such delays are called *primary delays*. After a landing, some of the delay can be absorbed by *ground buffer times*. If no recovery action (like a swap) is taken, which we assume in this paper, the remaining delay is propagated to the succeeding leg. Such a delay, that is propagated from a preceding leg, is called a *secondary delay*. There is nothing we can do about primary delays. But secondary delays, i.e., delay propagation, can be reduced by constructing aircraft rotations in such a way that buffer times are allocated at the "right" connections. In our data, secondary delays amount to approximately 30% of all delays.

Figure 1 shows an example of the influence of aircraft rotations on the propagation of delay. We consider an airport with two incoming legs A

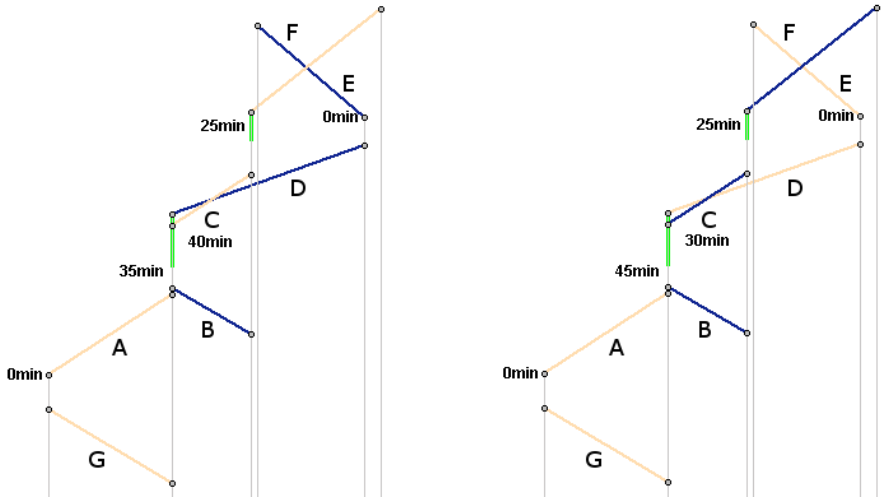

Figure 2: Effect of aircraft routing on delay propagation along an entire rotation.

and B and two outgoing legs C and D. The left side of the picture shows two rotations that connect legs AC and BD in a FIFO manner, resulting in a nearly even distribution of buffer times. However, suppose that leg A tends to be delayed more often than leg B. Then it can make sense to allocate more buffer after leg A by connecting legs AD and BC. Such a solution is shown on the right.

Thinking this example through, it becomes apparent that the propagation of delay from one leg to another does not only depend on the two legs involved, but on the entire rotation. The arrival delay of a leg depends on preceding legs, and the effect of a delay depends on the succeeding legs. Figure 2 illustrates these facts by showing the rotations of Figure 1 in whole. We see that leg A is preceded by leg G, and that there is no ground buffer between these legs. Arrival delays of leg G are therefore fully propagated to leg A, such that the arrival delay of leg A in this rotation may well be much higher than one would expect by considering leg A alone. Similarly, leg C is followed by leg F with 25 minutes of ground buffer and leg D is followed by leg E with no buffer. It may therefore be appropriate to cover legs D and E in a rotation that is unlikely to be delayed, because there is no possibility to reduce propagated or additional primary delay. Such a solution is shown on the right. The example shows that delay propagation has to be computed for an entire aircraft rotation, and can not be decided locally. For this reason, arc-based KPI approaches can not correctly predict the robustness of a rotation.

In the following subsections, we develop a stochastic model for primary

and secondary delays and a robust shortest path framework to compute a rotation with minimum probability of delay propagation.

2.1 Stochastic Model

We propose a delay propagation model that distinguishes gate and block phases in a rotation, see Rosenberger et al. (2002) [16] for more details on aircraft operations and associated models. A *gate phase* represents the ground time of an aircraft at a gate, i.e., the time between the arrival of an incoming leg at a gate and the departure of the succeeding outgoing leg from this gate. A *block phase* represents the taxi-out, en route, and taxi-in time of a leg. The duration of each such phase can deviate from the schedule because of primary delays (note that a gate phase can only be delayed, but a block phase can also be shorter than scheduled due to early arrivals, i.e., a block delay is actually a "block deviation").

Modeling the primary delays by random variables, we derive random variables for the secondary delay along a rotation. The model can be formally described as follows. Consider a rotation $r = (1, \ldots, k)$ with k legs. Denote by $b_{i,i+1}$ the buffer time between legs i and $i+1$, $i = 1, \ldots, k-1$, and by ω the maximal negative deviation of the block time of a leg (clearly, ω is smaller than the scheduled block time). Consider the following random variables:

- G_i: the delay of the gate phase before leg i, $i = 1, \ldots, k$,
- B_i: the deviation from the block time of leg i, $i = 1, \ldots, k$,
- AD_i^r: the arrival delay of leg i in rotation r, $i = 1, \ldots, k$,
- PD_i^r: the delay propagated to leg i in rotation r, $i = 1, \ldots, k$.

We assume that all primary delay variables G_i and B_i are independent. We further assume that $G_i \geq 0$ and $B_i \geq -\omega$, $i = 1, \ldots, k$. We will now argue that the delay propagated to leg i in rotation r is given by the recursion

$$PD_1^r = 0, \quad PD_{i+1}^r = \max\left\{PD_i^r + G_i + B_i - b_{i,i+1}, 0\right\}, \quad i = 2, \ldots, k. \tag{1}$$

To this purpose, we show by induction that the arrival delay and the delay propagated to leg i in rotation r are

$$\begin{aligned} & AD_i^r = PD_i^r + G_i + B_i \geq -\omega, && i = 1, \ldots, k \\ PD_1^r = 0, \quad & PD_i^r = \max\left\{AD_{i-1}^r - b_{i-1,i}, 0\right\}, && i = 2, \ldots, k. \end{aligned}$$

Substituting for AD_i^r gives Equation (1). In fact, since leg 1 is the first leg in rotation r, there is no propagated delay from previous legs, i.e., $PD_1^r = 0$.

Hence, the arrival delay of leg 1 in rotation r is the sum of the gate delay G_1 and the block deviation B_1

$$AD_1^r = G_1 + B_1 = PD_1^r + G_1 + B_1 \geq -\omega.$$

Now consider leg $i+1$, $i \geq 1$. Clearly, if the arrival delay AD_i^r is greater than the buffer time $b_{i,i+1}$, then $b_{i,i+1}$ minutes of delay will be absorbed by the ground buffer and only the remaining delay will propagate to leg $i+1$. Otherwise no delay will propagate to leg $i+1$. On the other hand, early arrival of the aircraft does not propagate to the next leg. Therefore, the delay propagated to leg $i+1$ in rotation r is

$$PD_{i+1}^r = \max\left\{AD_i^r - b_{i,i+1}, 0\right\}.$$

The arrival delay of leg $i+1$ in rotation r is the sum of the propagated delay from leg i and the primary gate delay G_i and the block deviation B_i

$$AD_{i+1}^r = PD_i^r + G_{i+1} + B_{i+1} \geq -\omega.$$

The random variables AD_i^r and PD_i^r can be used in various ways to define a quantitative measure of the *robustness* of a rotation r. We focus on the *probability of delay propagation* (PDP) $Pr[PD_i^r > 0]$ and use the "total probability of delay propagation" in rotation r defined as

$$d_r = \sum_{i \in r} Pr[PD_i^r > 0] \tag{2}$$

as a measure of robustness, i.e., we measure robustness in terms of (total) PDP (a rotation is more robust if its total PDP is smaller). We have defined the PDP for a leg in a rotation, and the total PDP for a rotation; for simplicity of notation, let us also speak of the *PDP of a rotation* in lieu of the total PDP. We remark that we have also tried other measures such as the "total expected propagated delay " (EPD), the "total expected arrival delay" (EAD), etc. These objectives produce similar results.

2.2 Minimizing the Probability of Delay Propagation

We consider the following graph theoretic framework to identify a rotation of minimal PDP. Let $G = (V, A)$ be an acyclic digraph, whose vertices V correspond to the legs of a flight schedule and whose arcs $e = (u, v)$ correspond to feasible connections of these legs. (Namely, a connection between legs u and v is feasible if the arrival airport of leg u is the same as the departure airport of leg v and if the scheduled departure time of leg v is greater than the scheduled arrival time of leg u plus some minimum ground time.) Associated with each leg v are two random variables $G_v \geq 0$ and B_v,

representing ground delays and block time deviations, and associated with each arc (u, v) is some integer value $b_{(u,v)}$, representing the buffer time in the connection of legs u and v. Then the arrival delay AD_v^r at leg v, the propagated delay PD_v^r to leg v, and the PDP d_r are well defined for every path in D, respectively, cf. Section 2.1. Given two legs s and t, the *Most Robust Rotation Problem* (MoRoRoP) is to find the most robust st-path in D, i.e., the st-path of smallest PDP.

Algorithm 1: Most Robust Rotation Algorithm.

Input : Digraph $D = (V, A)$ with random node variables $G_v \geq 0$ and B_v and arc buffers $b_{(u,v)} \geq 0$, two nodes s and t
Output: PDP of a most robust st-path in D

```
1  L_s := {(0,0)}
2  L_v := ∅ for all v ∈ V \ {s}
3  sort V topologically as v_1, ..., v_j = s, ..., v_k = t, ..., v_n
4  for i = j to k − 1 do
5      u := v_i
6      for all (u, v) ∈ E do
7          for (d_u, PD_u) ∈ L_u do
8              PD_v := max(PD_u + G_u + B_u − b_uv, 0)
9              d_uv := Pr[PD_v > 0]
10             L_v := L_v ∪ {(d_u + d_uv, PD_v)}
11         end
12         remove all dominated labels from L_v
13     end
14 end
15 return min_{(d_t, PD_t) ∈ L_t} d_t
```

Algorithm 1 solves the most robust rotation problem. It assigns sets L_v of labels (c_v, PD_v) to the nodes, that store the PDP of the current path to node v and the propagated delay. Labels are "pushed" along a topological node order; this guarantees finiteness. To save some running time over pure enumeration, labels are pruned by dominance. Namely, (c_1, PD_1) is dominated by label (c_2, PD_2) if $c_2 \leq c_1$ and $F_{PD_1}(x) \leq F_{PD_2}(x)$ for $\forall x \in \mathbb{R}$, where F_X denotes the cumulative distribution function of a random variable X.

2.3 Discretization

An implementation of the most robust rotation Algorithm 1 requires an encoding of the random variables G_v, B_v, and PD_v. These variables are summed up (Line 8), a nonnegative constant is subtracted (Line 8), the positive part is taken (Line 8), probabilities are computed (Line 9), and

dominance is checked (Line 12). All these operations can be performed using probability density functions, and we propose to use a discretization of the densities for numerical computations. Namely, we approximate the probability density function f_X of a random variable X by a step function $\tilde{f}_X^k$ with constant step size k and a finite number of non-zero segments

$$\tilde{f}_X(x) = \begin{cases} \alpha_i, & \text{for } x \in]k(i-1), ki], \quad i = l_X, \ldots, u_X, \\ 0, & \text{else,} \end{cases} \tag{3}$$

where l_X and u_X are integer lower and upper step bounds and

$$\alpha_i = \frac{\int_{k(i-1)}^{ki} f_X(x)dx}{k} = \frac{F_X(ki) - F_X(k(i-1))}{k}, \quad i = l_X, \ldots, u_X.$$

To approximate the primary delay variables G_i and B_i, we choose a step size k and a small real number $\epsilon > 0$. For $X = B_i$, we set $l_X = -\lfloor \omega/k \rfloor$, u_X such that $Pr[B_i > u_X k] < \epsilon$, and normalize. For $X = G_i$, we set $l_X = 0$, u_X such that $Pr[G_i > u_X k] < \epsilon$, and normalize. Starting from this initialization, we compute approximations to the derived distributions as follows.

Summation. The summations in Line 8 of Algorithm 1 involve independent random variables. Given integrable probability density functions f_X and f_Y of two independent random variables X and Y, the probability density function f_Z of their sum $Z = X+Y$ can be computed by the convolution

$$f_Z(t) = \int_{-\infty}^{\infty} f_X(t-x) f_Y(x) dx.$$

The convolution of two step functions $f'_Z = \tilde{f}_X + \tilde{f}_Y$ with identical step size k (but not necessarily identical upper and lower step bounds l_X, l_Y, and u_X, u_Y) is a piecewise affine function with segment step points in the set

$$\{l_Z, \ldots, u_Z\} = \{l_X + l_Y, \ldots, u_X + u_Y\}.$$

The function values at the end points of the line segments are

$$f'_Z(ik) = \sum_{j=\max\{l_X, i-l_Y\}}^{\min\{u_X, i-l_Y\}} k \tilde{f}_X(jk) \tilde{f}_Y((i-j)k), \quad i = l_X + l_Y, \ldots, u_X + u_Y.$$

Averages of these values can be used to again approximate f'_Z by a step function with step size k

$$\tilde{f}_Z(x) = \begin{cases} \beta_i, & \text{for } x \in]k(i-1), ki], \quad i = l_Z, \ldots, u_Z, \\ 0, & \text{else,} \end{cases}$$

where

$$\beta_i = \frac{f'_Z(k(i-1)) + f'_Z(ki)}{2}, \quad i = l_Z, \ldots, u_Z.$$

Note that $\int_{-\infty}^{\infty} \tilde{f}(x) = 1$ for $f = \tilde{f}_X, \tilde{f}_Y, \tilde{f}_Z$. Executing this procedure in Algorithm 1, step functions with increasing numbers of nonzero segments build up.

Nonnegative Constant. The summations in Line 8 of Algorithm 1 also involve the subtraction of a nonnegative constant. Let $\tilde{f}_X$ be a step function (3) with step size k and lower and upper step bounds l_X and u_X and let b be a nonnegative constant. For $b = k$, the probability density function of $Y = X - b = X - k$ is

$$\tilde{f}_Y(x) = \begin{cases} \alpha_{i+1}, & \text{for } x \in]k(i-1), ki], \quad i = l_Y, \ldots, u_Y, \\ 0, & \text{else,} \end{cases}$$

where $l_Y = l_X - 1$ and $u_Y = u_X - 1$. For $0 < b < k$ the probability density function of $Y = X - b$ is a step function with step size k and "shifted segments". It can be approximated by a step function with step size k and segment end points in $k\mathbb{Z}$ as

$$\tilde{f}_Y(x) = \begin{cases} \alpha_i \frac{k-b}{k}, & \text{for } x \in]k(i-1), ki], \quad i = u_X \\ \frac{\alpha_{i+1} b + \alpha_i (k-b)}{k}, & \text{for } x \in]k(i-1), ki], \quad i = l_X, \ldots, u_X - 1, \\ \alpha_{i+1} \frac{b}{k}, & \text{for } x \in]k(i-1), ki], \quad i = l_X - 1, \\ 0, & \text{else,} \end{cases}$$

i.e., $l_Y = l_X - 1$ and $u_Y = u_X$. The remaining cases follow.

Positive Part. Line 8 of Algorithm 1 requires taking the positive part of a random variable. Consider $W = \max(Z, 0)$ where Z is a random variable with discretized probability density function (3) with step size k and step bounds l_Z, u_Z. If $Z \geq 0$, $f_W = f_Z$, so assume $Z \ngeq 0$, i.e., $l_Z < 0$. Then f_W can be approximated as

$$\tilde{f}_W(x) = \begin{cases} \sum_{j=l_Z}^{0} \alpha_j, & \text{for } x \in] - k(i-1), ki], \quad i = 0, \\ \alpha_i, & \text{for } x \in]k(i-1), ki], \quad i = 1, \ldots, u_Z, \\ 0, & \text{else,} \end{cases}$$

i.e., $l_W = 0$ and $u_W = \max\{0, u_Z\}$. We remark that we actually use a different density function in our implementation that involves a point distribution at zero; for simplicity of exposition, we omit the details here.

Probability. The probability that a random variable is positive must be computed in Line 9 of Algorithm 1. If $\tilde{f}_X$ is a discretized probability density function (3) with step size k and lower and upper step bounds l_X, u_X, this probability is

$$Pr[X > 0] = k \sum_{i=1}^{u_X} \alpha_i.$$

Dominance. Comparing the distribution functions of two random variables with discretized probability density functions $\tilde{f}_X$ and $\tilde{f}_Y$ in Line 12 of Algorithm 1 can be done by checking $F_X \geq F_Y$ as

$$\sum_{i=\min\{l_X,l_Y\}}^{j} \tilde{f}_X(ik) \geq \sum_{i=\min\{l_X,l_Y\}}^{j} \tilde{f}_Y(ik), \quad j = \min\{l_X, l_Y\}, \ldots, \max\{u_X, u_Y\}.$$

***

Approximating the density functions of random variables as described in this section throughout the course of Algorithm 1 results in the computation of an "approximately most robust rotation" with an "approximate PDP" of $\tilde{d}_r$. The computational complexity of these computations depends on the development of the span $u_X - l_X$ of the step bounds. If convolutions are computed and if constants are subtracted, the span increases, if positive parts are taken, the span decreases. We heuristically decrease the span further by cutting off segments in the tail of the distribution as long as their total probability is less than some small constant. In our computations, the span then typically increases along the first three or four arcs, and remains constant afterwards. Clearly, the precision also deteriorates with increasing path length. In practice, however, the accuracy is high, as we will see in the computational Section 4.

3 Robust Tail Assignment

The tail assignment problem is about the construction of rotations for a set of individual aircraft in order to cover a set of flights. The problem is solved on a daily basis before the day of operation in order to adjust short- to long-term maintenances and other operational requirements. More details can be found in the thesis of Grönkvist (2005) [12].

3.1 Integer Programming Model

The tail assignment problem can be modeled as a set partitioning problem with base constraints, see again Grönkvist (2005) [12].

$$\min \sum_{k} \sum_{r \in R_k} d_r x_r^k \tag{4}$$

$$\sum_{k} \sum_{r: l \in r, r \in R_k} x_r = 1 \qquad \forall l \in L \tag{5}$$

$$\sum_{k} \sum_{p \in R_k} a_{bp} x_p^k \leq r_b \qquad \forall b \in B \tag{6}$$

$$\sum_{j \in R_k} x_j^k = 1 \qquad \forall k \tag{7}$$

$$x_r^k \in \{0,1\} \qquad \forall k, \forall r \in R_k. \tag{8}$$

Here, R_k is the set of feasible rotations for aircraft k. Constraint (5) guarantees that every leg is covered by exactly one rotation. Constraint (6) ensures that the base constraints (which model, e.g., maintenance capacities) are satisfied. The objective maximizes robustness in the sense of minimizing the probability of delay propagation. Using discretization, the objective is approximated as $\sum_k \sum_{r \in R_k} \tilde{d}_r x_r^k$.

3.2 Column Generation

We use a column generation algorithm to solve the tail assignment problem. Such an algorithm solves the linear programming relaxation of the problem, the so-called *master problem*, in an iterative way. In each iteration, a subset of columns is considered and the associated *reduced master LP* is solved. Then the *pricing problem* to identify new columns with negative reduced cost is solved. These columns are added to the reduced master problem. This process is repeated until no more improving columns exist. At this point, the master LP is solved. An integer solution is constructed by a rounding heuristic in a second phase.

Associating dual variables π_l, $l \in L$, with the leg covering Constraints (5), μ_b, $b \in B$, with the base Constraints (6), and ν_k, $k \in K$, with the aircraft Constraints (8), the pricing problem to construct for aircraft k a rotation $r \in R_k$ of negative reduced cost can be stated as

$$\min_{r \in R^k} \overline{d}_r = \min_{r \in R^k} d_r - \sum_{l \in r} \pi_l + \sum_{b \in B} a_{br} \mu_b - \nu_k.$$

Assuming furthermore that the consumption of base resources on a rotation for aircraft k can be expressed in terms of base resource consumptions $a_l^{b,k}$

Algorithm 2: Resource Constrained Most Robust Rotation Algorithm.

Input : Digraph $D = (V, A)$ with random node variables $G_v \geq 0$ and B_v, arc buffers $b_{(u,v)} \geq 0$ and base resource consumptions $a^{b,k}_{(u,v)}$, rotation feasibility oracle, aircraft k, two nodes s and t

Output: Most negative reduced cost of rotation $r \in R_k$ for aircraft k

```
L_s := {(−ν_k, 0)}
L_v := ∅ for all v ∈ V \ {s}
sort V topologically as v_1, ..., v_j = s, ..., v_k = t, ..., v_n
for i = j to k − 1 do
    u := v_i
    for all (u, v) ∈ E do
        for (d_u, PD_u) ∈ L_u do
            updatePathResourceConsumption(u,v,(d_u, PD_u))
            if resourceConsumptionNotValid() then
                next
            end
            PD_uv := max(PD_u + G_v + B_v − b_uv, 0)
            d_uv := Pr[PD_v > 0] − π_v + Σ_{b∈B} μ_b a_v^{b,k}
            L_v := L_v ∪ {(d_u + d_uv, PD_uv)}
        end
        remove all dominated labels from L_v
    end
end
return min_{(d_t,PD_t)∈L_t} d_t
```

on legs l, the pricing problem becomes

$$\min_{r \in R^k} \overline{d}_r = \min_{r \in R^k} d_r - \sum_{l \in r} \pi_l + \sum_{l \in r} \sum_{b \in B} \mu_b a_l^{b,k} - \nu_k. \tag{9}$$

For typical tail assignment problems, this is a *resource constrained most robust rotation problem*. Algorithm 2 shows pseudocode of a corresponding resource constrained most robust rotation algorithm. The algorithm handles feasibility rules for aircraft rotations by means of a feasibility oracle. Using discretization, we are looking for minimal approximate reduced costs

$$\min_{r \in R^k} \overline{\tilde{d}}_r = \min_{r \in R^k} \tilde{d}_r - \sum_{l \in r} \pi_l + \sum_{l \in r} \sum_{b \in B} \mu_b a_l^{b,k} - \nu_k$$

in our implementation.

4 Computational Results

We report in this section on computational results for the solution of real-world large-scale tail assignment problems with our PDP method. Our code extends the pricing algorithm of the commercial state-of-the-art tail assignment optimizer TAIL xOPT from the NETLINE planning system of Lufthansa Systems Berlin GmbH (LSB) (see Schickinger (2008) [18]) in order to handle the probability of delay propagation as a robustness objective. In other words, we use TAIL xOPT with the pricing Algorithm 2 for our computations; we will refer to this method as PDP. Simulations were done using a fast, event-based engine Dammer (2010) [9].

We first discuss our test instances and, in particular, the concrete form of our stochastic model for our data. We then compare our PDP approach to a traditional KPI method, and investigate its accuracy and efficiency. Finally, we benchmark our approach on historic situations in order to show that our model indeed "captures reality".

All computations were made on a 64-bit desktop PC with two Intel(R) Core(TM) 2 processors with 3.0 GHz and 8 GB of RAM memory, running openSuse Linux 11.2. Our code is implemented in `C++` and was compiled using `g++` 3.4.

4.1 Data

LSB provided historical data on the operation of a fleet of a European short haul carrier. It consists of a logfile on the operation of all 350,000 flights that were flown by this fleet over a period of 28 months in the airline's hub-and-spoke network (with two hubs), see Figure 3 for an illustration. The data contains information about departure and arrival times, and about departure delays including the IATA codes. Using these codes, we identified primary and secondary delays. The primary delays were analyzed to build the stochastic model.

Gate Delays. The two constitutive factors are the probability that a delay occurs and, subordinately, the length of the delay. In fact, our analysis shows that the length of a gate delay is independent of the airport and the time of day, while the frequency of delay occurrences depends on the airport and the time of day. We therefore derived the probability of a gate delay for each airport and hour of the day as the ratio of the number of departing flights with primary gate delays over the number of all departing flights. The length of gate delays was modeled in terms of a universal log-normal distribution that was fitted over all airports, and a power-law distribution

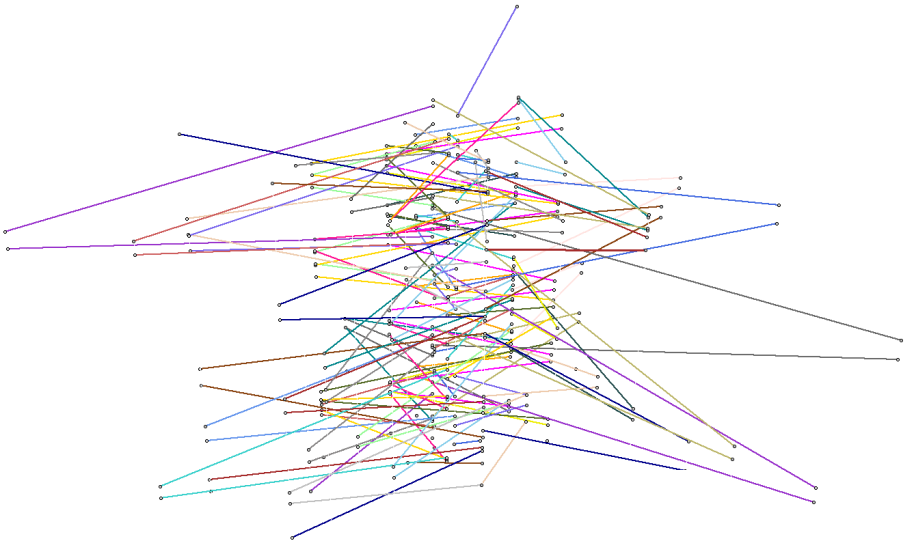

Figure 3: Flight network for one day of operation of a European short haul carrier (time goes up).

for delays of more than 60 minutes in order to account for the heavy tails that we observed. Cutting off delays greater than or equal to 220 minutes, the probability density function $f_{G_i}(x)$ for a gate delay of leg i that results from these considerations is (modulo some normalization)

$$f_{G_i}(x) = \begin{cases} p(a_i, t_i) Ln(x, \mu, \sigma), & 0 < x <= 60, \\ p(a_i, t_i) Pl(x, \alpha), & 60 < x < 220, \\ 1 - p(a_i, t_i), & x = 0, \\ 0, & \text{else.} \end{cases}$$

Here, $Ln(\mu, \sigma)$ denotes the probability density function of a log-normal distribution with mean μ and standard deviation σ, $Pl(\alpha)$ denotes the probability density function of a power-law distribution with parameter α, and $p(a_i, t_i)$ is a probability of delay occurrence that depends on the departure airport $a(i)$, and the hour of departure $t(i)$ of leg i.

Block Delays. Our analysis shows that block delays (and early arrivals) depend only on the scheduled block duration, see Figure 4. This at first sight surprising finding can be explained as a result of regular flight time adjustments by the airline in order to match scheduled and observed flight times. Modeling block delays by a log-logistic distribution, the probability

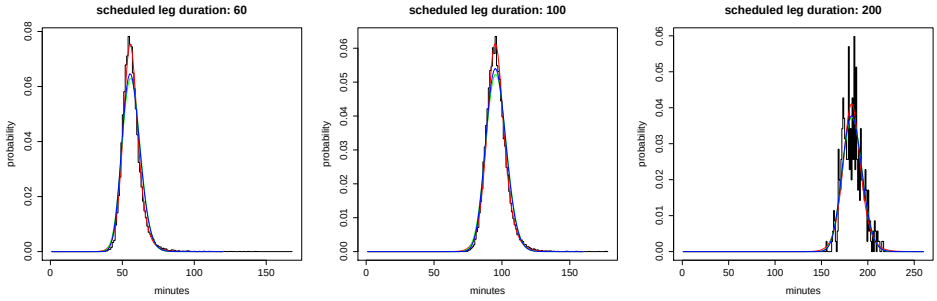

Figure 4: Historic block delay distributions for legs with scheduled leg durations of 60, 100, and 200 minutes.

	min			max			avg		
	legs	acft	time	legs	acft	time	legs	acft	time
January	44	12	3840	105	17	8830	88	15	7447
February	94	15	8295	118	17	10065	109	16	9339
March	94	15	7900	121	17	10390	110	16.3	9483
April	93	15	7080	118	18	9750	103	16	8648

Table 1: Statistics on tail assignment instances. Maximum, minimum, and average number of legs, aircraft, and flight time per day.

density function $f_{B_i}(x)$ for block delays of leg i becomes

$$f_{B_i}(x) = Llg\big(x, \alpha(\ell_i), \beta(\ell_i)\big), \quad x \in R.$$

Here, $Llg\big(x, \alpha(\ell_i), \beta(\ell_i)\big)$ is the probability density function of a log-logistic distribution with parameters $\alpha(\ell_i)$ and $\beta(\ell_i)$ that depend on the scheduled duration ℓ_i of leg i.

Scenarios. We test our method on four scenarios that each cover one month of data, namely, January, February, March, and April 2007. Table 1 lists some statistics on these scenarios. Each of them consists of several days. In the upcoming computations, these days will always be optimized individually, i.e., we run the optimizer for about 30 one day instances of a monthly scenario.

4.2 Maximizing Robustness

We compare our PDP approach in this section with a KPI method that uses no stochastic information. This method, further referred to as ORC (Original Robustness Costs), rewards ground buffer times between two successive

		ORC			PDP			savings	
		PDP	EAD	CPU	PDP	EAD	CPU	PDP	EAD
Month	(#)		[min]	[s]		[min]	[s]		[min]
January	(26)	414.51	28488	28	395.46	28085	66	19.05	403
February	(22)	540.48	31870	31	530.42	31652	89	10.06	218
March	(21)	516.69	30363	31	507.91	30174	75	8.78	189
April	(27)	465.48	34453	42	449.16	34159	71	16.51	294

Table 2: Optimizing the robustness of airline rotations using a KPI method (ORC) and a stochastic optimization method (PDP).

legs in a rotation up to a maximum value of 15 minutes. This is a standard practitioner's approach; it aims at distributing ground buffers of reasonable length in a homogenous way.

Table 2 summarizes the results of our experiments. The first column of the table identifies the scenario. Because of some data inconsistencies we were not able to optimize every day of every month; the numbers in brackets in the second column give the number of optimized days. The columns labeled PDP show the probability of delay propagation over all rotations in all schedules that have been constructed for a scenario using the respective optimization method. For the ORC schedules, these numbers have been computed a posteriori, for the PDP schedules, they constitute the objective. In all cases, a discretization step size of one minute was used. The columns labeled EAP give the total expected arrival delay minutes. Arrival delay causes disruptions of passenger connections and excess working time for crews; this is some measure of increase in operation costs. The columns labeled CPU list the average computation time of one instance of a scenario in CPU seconds. The last two columns of the table summarize the expected improvements of PDP over ORC. With up to 5%, they are substantial.

Figure 5 gives a more detailed impression of these improvements. It plots the arrival delays measured in 500 simulation runs of an ORC and a PDP schedule for one particular day of operation. Points on the right of the diagonal represent simulation runs in which PDP outperformed ORC, points on the left of the diagonal represent points where ORC outperformed PDP. This happens in only 21% of all cases. This means that PDP is not only better than ORC on average, but also in the vast majority of cases. Note also that the superiority grows with the number of disruptions. In an average simulation, the PDP schedule saves 29 minutes of arrival delay over the ORC solution, and 62 minutes in simulations where both schedules suffer at least 1000 minutes of arrival delay.

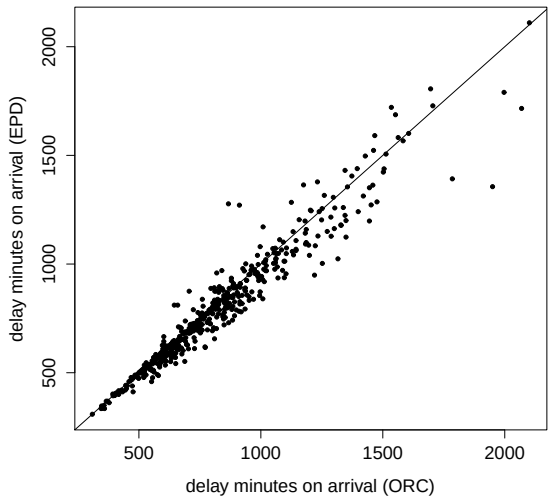

Figure 5: Comparing average arrival delay minutes for 500 disruption simulations of an ORC and an PDP schedule.

4.3 Complexity

The accuracy and the running time of the PDP method can be controlled in terms of the step size parameter. The smaller the step size, the higher the accuracy, but also the computational effort. We now investigate this tradeoff, as well as the accuracy per se. To determine the approximation error, we estimate the "true" probability of delay propagation of a schedule by averaging over the results of 100,000 runs of a simulator that generates random disruptions according to our stochastic model, see Dammer (2010) [9].

We start with an experiment that is designed to give an impression of the speed of the convolution algorithm. In order to eliminate influences from different execution paths of the column generation algorithm, we simply use the PDP algorithm to compute the probability of delay propagation for two (not optimized) schedules SC1 and SC2, i.e., we take the rotations of a schedule and compute their PDPs. Changing the discretization step size, the results then differ only with respect to accuracy and computation time. As usual, both instances cover a single day of operation. Instance SC1 involves 100 legs flown by 16 aircraft; SC2 has 103 legs covered by 21 aircraft. The rotations involve up to 8 legs.

Table 3 shows the effect of six different step sizes ranging from 0.1 to 4.0 minutes. The first column in both subtables gives the name of the instance,

	step size [m]	CPU [s]	PDP	error [%]		step size [m]	CPU [s]	PDP	error [%]
SC1	0.1	15.4	25.0586	0.11	SC2	0.1	12.8	21.3792	0.09
SC1	0.5	1.0	25.0672	0.15	SC2	0.5	1.0	21.3858	0.12
SC1	1.0	0.5	25.0917	0.25	SC2	1.0	0.6	21.4047	0.21
SC1	2.0	0.4	25.2227	0.77	SC2	2.0	0.5	21.5145	0.72
SC1	3.0	0.4	25.4775	1.79	SC2	3.0	0.4	21.7354	1.76
SC1	4.0	0.3	25.7657	2.94	SC2	4.0	0.4	21.9386	2.70
SIM			25.0303		SIM			21.3605	

Table 3: Investigating the influence of discretization step sizes on running time and accuracy of convolution computations.

	step size [m]	PDP (opt.)	time [s]	PDP (sim.)		step size [m]	PDP (opt.)	time [s]	PDP (sim.)
SC1	0.1	19.7268	4450	19.7469	SC2	0.1	18.9141	616	18.8499
SC1	0.5	19.7362	231	19.7382	SC2	0.5	18.878	17	18.8071
SC1	1.0	19.7450	70	19.7239	SC2	1.0	18.8955	19	18.8088
SC1	2.0	19.8693	45	19.7313	SC2	2.0	19.0195	8	18.8459
SC1	3.0	20.0651	29	19.7239	SC2	3.0	19.2033	7	18.8208
SC1	4.0	20.3353	31	19.7562	SC2	4.0	19.3789	11	18.8252

Table 4: Investigating the influence of discretization step sizes on running time and accuracy of tail optimization.

the column labeled step size [m] and CPU [s] gives exactly this. The PDP column gives the objective as computed by our algorithm, except for the last row, which gives the averaged result of 100,000 simulation runs. The last column gives the approximation error as the relative difference between the values computed by the PDP algorithm and the simulation. It can be seen that the error is in general very small, even a relatively coarse discretization step size of two minutes gives very accurate results. Moreover, even coarser discretizations do not pay off with respect to running times. A discretization step size of one minute, which is the standard setting that we have used for computations throughout this paper, seems to be perfectly adequate.

Table 4 investigates the effects of varying step sizes on a run of the PDP optimization algorithm for the same scenarios. Throughout the column generation, many rotations are constructed and many delay distributions are computed, such that the running times, listed in the columns labeled CPU [s], are much higher than those in Table 3. The columns labeled PDP (opt.) give the PDP value computed by the optimizer. This value is compared

with the average delay propagation over 10,000 simulation runs for such a solution, listed in the columns labeled PDP (sim.). It can be seen that the solution quality of the PDP approach is insensitive to the discretization step size, i.e., even with very coarse discretizations, the method computes very good solutions. In other words, the important point is to take delay propagation into account; if one does, the quality of the discretization does not influence the quality of the solution much. However, the objective values computed by the optimizer become increasingly bad estimates of the "true" objective for coarser discretizations. Again, a discretization step size of one minute appears to be a good setting.

leg	duration	buffer	PDP	step size 1.0		step size 4.0	
	[m]	[m]	(sim.)	PDP	error [%]	PDP	error [%]
1	60	25	0	0	0	0	0
2	100	10	0.0478	0.0470	1.67	0.0483	1.05
3	115	20	0.2144	0.2136	0.37	0.2213	3.22
4	40	0	0.0957	0.0949	0.84	0.0985	2.93
5	45	10	0.5521	0.5514	0.13	0.5608	1.58
6	75	20	0.3365	0.3367	0.06	0.3486	3.60
7	90	15	0.1959	0.1955	0.20	0.2035	3.88
8	70	-	0.1428	0.1432	0.28	0.1489	4.27

leg	duration	buffer	PDP	step size 1.0		step size 4.0	
	[m]	[m]	(sim.)	PDP	error [%]	PDP	error [%]
1	95	0	0	0	0	0	0
2	95	0	0.3177	0.3176	0.03	0.3267	2.83
3	65	0	0.4596	0.4591	0.11	0.4711	2.50
4	60	-	0.5746	0.5743	0.05	0.5873	2.21

leg	duration	buffer	PDP	step size 1.0		step size 4.0	
	[m]	[m]	(sim.)	PDP	error [%]	PDP	error [%]
1	95	55	0	0	0	0	0
2	100	35	0.0110	0.0109	0.91	0.0110	0
3	95	0	0.0195	0.0194	0.51	0.0195	1.53
4	85	35	0.4402	0.4395	0.16	0.4402	2.20
5	120	-	0.0600	0.0589	1.84	0.0600	2.17

Table 5: Probability of delay propagation for individual legs in rotations.

We finally take a closer look at the probability of delay propagation on individual rotations. Table 5 reports PDP values on the individual legs of

three rotations that were computed in the optimization of instance SC2 for Table 3 with discretization step sizes of one and four minutes. The columns of this table list the legs in the three rotations considered, scheduled block times in minutes, ground buffer times in minutes, simulated PDP values, and optimized PDP values as well as the relative error for discretization step sizes of one and four minutes, respectively. The solution quality is again very good. For a discretization step size of one minute, the PDP is forecasted with an error less than 1% on each individual legs, except on legs with very small PDPs in the order of the computational precision. Such errors are irrelevant in practice. Note also that the approximation error does not grow with the length of the rotation. For a step size of four minutes, the results for individual legs become less reliable. Again, a discretization step of one minute suggests itself.

4.4 Proof of the concept

All hitherto presented results depend on the correctness of our stochastic optimization model. In order to verify the superiority of a PDP approach over an ORC approach, we have run a direct simulation on the basis of historic data, i.e., we applied primary delays as recorded in our data to schedules that were optimized with the PDP and the ORC method. This evaluation does not depend on the stochastic model. In our experiment, each day in each scenario was optimized with respect to ORC and PDP, and then evaluated with respect to the historic disturbances as observed on the respective days of operation.

		ORC		PDP		savings	
Month		DP [#]	AM [m]	DP [#]	AM [m]	DP [#]	AM [m]
January	(26)	559	48670	531	47051	28	1619
February	(22)	554	26301	536	25934	18	367
March	(21)	662	39048	637	38743	25	305
April	(27)	466	27044	465	27160	1	-116

Table 6: Directly testing ORC and PDP schedules on historic data.

Table 6 shows the results. The columns of this table give the name of the scenario and the number of days that were evaluated, the columns labeled DP and AM give the number of delay propagations and the arrival delay in minutes that would have resulted in the historic situation for an ORC and a PDP optimized schedule, and the savings. The PDP schedules would have clearly outperformed the ORC schedules.

5 Conclusion

We have developed a novel stochastic optimization approach in order to construct robust aircraft rotations in the sense of minimizing the total probability of delay propagation along aircraft rotations. The method is fast, accurate, and produces substantially more robust schedules than traditional KPI approaches with respect to evaluation criteria such as total PDP, PDP on individual legs, stochastic and historic simulation. The PDP method is easily plugged into any column generation framework, and therefore generalizes to crew scheduling, crew rostering, and other problems that can be solved with set partitioning approaches.

References

[1] S. AhmadBeygi, A. Cohn, Y. Guan & P. Belobaba. Analysis of the potential for delay propagation in passenger airline networks. Technical report, Sloan Industry StudiesWorking Paper Series, 2007.

[2] S. AhmadBeygi, A. Cohn & M. Lapp. *Decreasing airline delay propagation by re-allocating scheduled slack*. In *AGIFORS*, 2008.

[3] C. Barnhart, P. Belobaba & R. Odoni. *Applications of operations research in the air transport industry*. Transporation Science **37**:368 – 391, 2003.

[4] I. Bertsimas & M. Sim. *Robust discrete optimization and network flows*. Mathematical Programming Series B **98**:49–71, 2003.

[5] F. Bian, E. Burke, S. Jain, G. Kendall, G. Koole, J. L. Silva, J. Mulder, M. Paelinck, C. Reeves, I. Rusdi & M. Suleman. *Measuring the robustness of airline*, 2003. URL citeseer.ist.psu.edu/730896.html.

[6] G. Boydell. Standard inputs for eurocontrol cost benefit analyses. Technical report, EUROCONTROL, 2005.

[7] E. K. Burke, P. De Causmaecker, G. De Maere, J. Mulder, M. Paelinck & G. Vanden Berghe. *A multi-objective approach for robust airline scheduling*. Preprint, 2007.

[8] A. Cook, G. Tanner & S. Anderson. Evaluating the true cost to airlines of one minute of airborne or ground delay. Technical report, EUROCONTROL, 2004.

[9] R. Dammer. Integrierte und robuste planung von flugzeugen und personal im luftverkehr. Master's thesis, TU Berlin, 2010. to appear.

[10] Eurocontrol. Costs of air transport delay in europe. Technical report, EUROCONTROL, 2000.

[11] B. Fuhr. *Robust flight scheduling - an analytic approach to performance evaluation and optimization*. In *AGIFORS*, 2007.

[12] M. Grönkvist. *The Tail Assignment Problem*. PhD thesis, Göteborg university, 2005.

[13] A. J. Kleywegt, A. Shapiro & T. Homem-de Mello. *The sample average approximation method for stochastic discrete optimization.* SIAM Journal on Optimization **12**:479–502, 2001.

[14] S. Lan. *Planning for robust airline operations : Optimizing aircraft routings and flight departure times to achieve minimum passenger disruptions*. PhD thesis, Massachusetts Institute of Technology, Dept. of Civil and Environmental Engineering., 2003.

[15] S. Lan, J.-P. Clarke & C. Barnhart. *Planning for robust airline operations: Optimizing aircraft routings and flight departure times to minimize passenger disruptions*. Transportation Science **40**(1):15–28, February 2006.

[16] J. M. Rosenberger, A. J. Schaefer, D. Goldsman, E. L. Johnson, A. J. Kleywegt & G. L. Nemhauser. *A stochastic model of airline operations*. Transportation Science **36**(4):357–377, 2002.

[17] J. M. Rosenberger, E. L. Johnson & G. L. Nemhauser. *A robust fleet-assignment model with hub isolation and short cycles*. Transportation Science **38**(3):357–368, 2004. ISSN 1526-5447.

[18] T. Schickinger. *The potential of tail assignment optimization*. In *AGIFORS Airline Operations*, 2008.

[19] S. Stiller. *Extending Concepts of Reliability*. PhD thesis, TU Berlin, 2008.

[20] G. Yu. *Operations Research in the Airline Industry*. Kluwer Academic Publishers, 1997.