Reducing Delays by Optimized Runway Assignment

Rainer Kiehne, Air Transport and Airport Research,
German Aerospace Center, Köln, Germany
Michael Kolonko, Institut für Mathematik,
University of Technology Clausthal, Germany

ABSTRACT

During the last decades the limited capacity of runways has become an important source for delays on major airports. Particular safety regulations require asymmetric separation times for aircraft during their landing operations. If there are two or more runways available, aircraft can be assigned to runways in such a way that large separation times are avoided. This increases the actual throughput of the system and reduces additional delays.

We formulate the assignment problem as a mathematical optimization model. Though we cannot find optimal assignment strategies analytically, we can use the model as a framework for the simulation of strategies and as a tool for the heuristic optimization of strategies. We examine two classes of strategies that reduce the waiting times of arriving aircraft in simulations.

In particular, we can show that in realistic simulation scenario of a German airport, one of our strategies performs much better than the manual assignment as it is used by flight operators today. As our strategies are essentially simple look-up tables, they may well be incorporated into future flight assistance systems for airports.

1 INTRODUCTION

During the last decades a significant growth of air traffic has been observed which has not been countered by an adequate increase of runway capacities at airports. Since runway capacity has been identified as a major source of delay, the efficient use of existing runway systems is necessary to meet the demand of commercial aviation.

In this article we present simple strategies to route arriving aircraft to runways in an efficient way such that for a given arrival rate of aircraft the waiting times are minimized or vice versa the arrival rate is maximized with waiting times below a given threshold.

A crucial point is the uncertainty concerning arrival times of approaching aircraft. If all details of future arrivals were known, the problem would be static and an optimal sequence could be determined by mathematical optimization. In reality however, the prediction of air traffic is often imprecise due to en-route delays or arrivals ahead of schedule. Thus

a dynamic scheduling algorithm is needed that assigns incoming aircraft according to the information available at the time of arrival.

In this chapter, we restrict ourselves to a particular routing model: there are two runways available for landing aircraft (we do not consider any starts in the present set-up). Arriving aircraft are assigned to one of the runways as soon as they pass the threshold of the so-called terminal manoeuvring area (TMA) of the airport. We do not assume any knowledge about aircraft that are still beyond TMA, so the main information for the routing decision is the state of the runways. See [3] and the literature cited there for models that take into account additional information. If the assigned runway is not free at the time the aircraft is ready to land, it has to wait in a 'queue' which in this case is rather a loop. Each runway has its own queue, aircraft are not allowed to change the queue or their position within the queue once they are assigned to it. This restriction reflects the fact that assignments have to be fixed some time before the aircraft enters the final approach to ensure a safe trajectory to the runway.

If we look at the runways as servers and at the incoming aircraft as their customers, then the problem can be modeled as a simple queuing system with two parallel servers. The arrival times are random, often it is assumed that they form a Poisson process (see e. g. [7]) but we also consider more realistic arrival patterns in our simulation in Section 5. The service time of an aircraft is the time a trailing aircraft has to wait until it can begin its landing operation. The aim is to find routing strategies that minimizes the average waiting time of a customer in the queue.

The characteristic feature here is that consecutive service times are *not* independent as it usually assumed in queuing systems. An aircraft causes air turbulences that endanger the stability of trailing aircraft. The required separation times between consecutive landings therefore depend on the size and weight of the two aircraft involved and on their order: a heavy aircraft may follow more closely on a light one than the other way round. Hence the service time of a landing aircraft, i. e. the separation time to its successor, also depends on the type of this successor and hence also on the following service time. This dependence of service times cannot be neglected as will be shown below.

We start in Section 2 by formulating a mathematical model for the problem sketched above. It turns out that determining an optimal routing strategy requires the solution of a complex stochastic optimization problem. Though the optimization problem seems intractable, a simulation of the performance of a given strategy is quite simple. We therefore try to find reasonable strategies by a combination of simulation and heuristic search. Using simulation for the evaluation of routing strategies also has the advantage that we can easily include any air traffic management procedures that depend heavily on the airport layout and airport specific regulations.

We identify two classes of simple and transparent strategies for which heuristic search provides good results. In the first case which is detailed in Section 3, the strategies are not allowed to use the full information of the system state, instead they must rely on a rough classification of the state into one of a few categories. Under this restriction good strategies can be derived by local search using ideas from neuro-dynamic programming, see e. g. [6].

In Section 4 we use strategies that are variations of the so-called join-the-shortest-queue principle. These strategies can easily be parameterized and good strategies, i. e. good parameter values can be determined by genetic algorithms and simulated annealing in combination with simulation.

In Section 5 we present results from simulation experiments for these two types of strategies with different scenarios. In a detailed study with realistic 24h data of a German airport we compare our strategies with the routing as it is performed by flight operators today. The results show that even though the mathematical model captures only part of the real problem, the strategies obtained have a significant potential for capacity improvements and delay reductions in real operations.

2 MATHEMATICAL MODELS

In this section we sketch a mathematical model for the the runway assignment problem. For a single runway, a single server queue with dependent service times is an adequate model. Combined into a system of two (or more) runways this leads to a complex stochastic dynamic programming problem. Though we cannot solve this problem analytically to obtain an optimal strategy, we can use it as a framework for the simulation and evaluation of strategies in the Sections below.

A Queuing Model

We start by analyzing a single runway without any routing. As indicated in the Introduction, aircraft are classified into three weight categories: heavy, medium and light. The separation times for consecutive aircraft are given in a 3×3 matrix

$$D := \begin{pmatrix} d(1,1) & d(1,2) & d(1,3) \\ d(2,1) & d(2,2) & d(2,3) \\ d(3,1) & d(3,2) & d(3,3) \end{pmatrix}. \tag{1}$$

Here $d(i,j)$ denotes the separation time an aircraft of type j has to keep to a leading aircraft of type i with $i,j \in \{1,2,3,\} = \{\text{heavy}, \text{medium}, \text{light}\}$.

In the most basic case, we assume that the arrivals are completely random. More precisely, this means that the arrival times of aircraft at the TMA form a Poisson process. The types of the arriving aircraft are selected according to a given 'type mix' (p_1, p_2, p_3) which gives the average relative proportions of the three types. Here, $(p_1, p_2, p_3) = (0.23, 0.72, 0.05)$ e. g. means that 23% of the aircraft are of type 'heavy', 72% of type 'medium' and only 5% of type 'light'. Under these assumptions, a single runway can be modeled as an $M/SM/1$-queue, where 'SM' stands for 'semi-Markov', a concept that allows for dependent service times as needed in our case, see [8] for more details.

An $M/SM/1$-system is slightly more complicated than the standard $M/G/1$-system with independent service times which is used as a model for a runway e. g. in [4]. In an $M/G/1$-system the average waiting time is easily determined as a function of the arrival

rate and the first two moments of the service time distribution. The corresponding results for $S/SM/1$ are a little more involved, they include the effect the dependence of service times may have on the average (see [8], [1]). In [1] it is shown that the average waiting times obtained from the 'full' $M/SM/1$-models and the simplified $M/G/1$-models may be quite different and that the error becomes arbitrarily large as the arrival rate increases.

Runway Assignment as Markovian Decision Process

We shall now turn to a system with two runways. If the 'local' arrivals of aircraft assigned to a runway form a Poisson process than we have a system of two separate $M/SM/1$-queues. But even if the outer arrival stream of aircraft at the airport (TMA) follows a Poisson process, the local arrival streams of aircraft after assignment will be Poisson only for very special assignment strategies, see [1] for more details on the so-called 'split-strategies'.

In general, an assignment decision influences the situation the next arriving aircraft will see on both runways so that they cannot be treated separately. Runway assignment is a typical *sequential* decision-making problem, it can be modeled using a discrete time Markovian decision process (MDP). We shall briefly describe the main ingredients of an MDP.

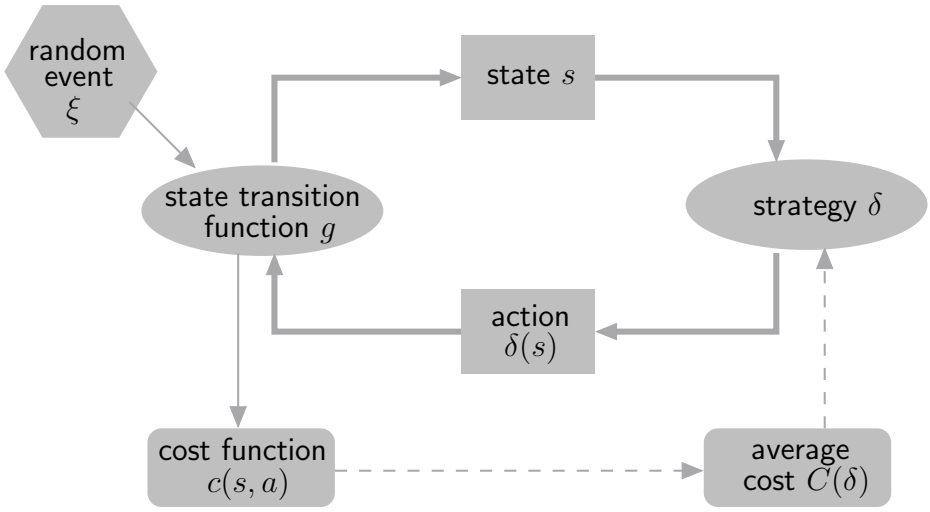

Figure 1: Markov Decision Process

In MDPs, decisions have to be made at certain points in time, in our case these are the arrival times of aircraft, denoted by $T_1, T_2, \ldots$. These may either form a Poisson process or result from an observation of a real arrival stream. The type of the n-th arriving aircraft is denoted by J_n taking on one of the values $1, 2, 3$ for 'heavy','medium or 'light' as described earlier. The time that has elapsed since the last arrival and the type of the incoming aircraft together form a stochastic arrival *event*. If there are two runways named I and II, the possible decisions or *actions* are simply $a = I$ or $a = II$.

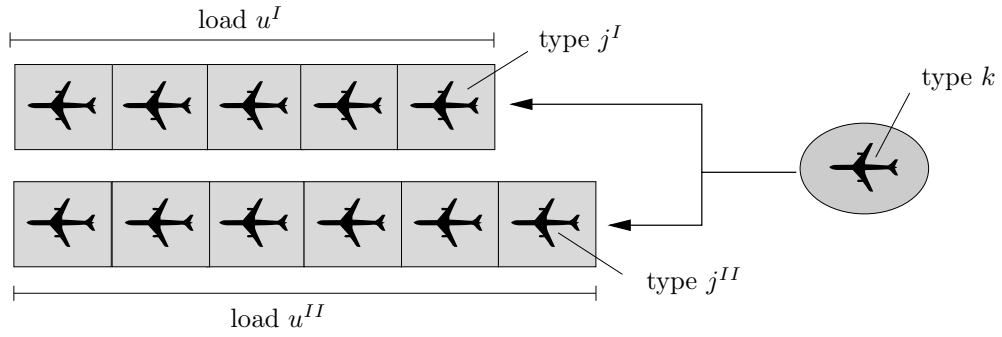

Figure 2: Elements of the system state

The central part of the model is its *state* s. It has to comprise all information available about the future behavior of the system at the moment a decision has to be made, so that the decision may be based solely on the present state of the system. A decision *strategy* is then a function δ that in each state s chooses an action $\delta(s)$. Once an action a has been taken, the system state s changes according to a *transition function* g to the new state $s' = g(s, a, \xi)$ depending on s, a and the *event* ξ that represents the external random influence on the system. This transition incurs *one-step-costs* $c(s, a)$. A strategy δ can be evaluated e. g. by the average costs $C(\delta)$ resulting from applying δ to the system over a long time. This general scheme is depicted in Figure 1.

We must now define a 'state' appropriate for the runway assignment. As we assume that there is no knowledge about aircraft beyond TMA, the state consists of a description of the current load of the runways and the additional demand of the arriving aircraft. The *load* u of a runway at a certain point in time is the remaining waiting time of the aircraft waiting at the tail of its queue. The load u may be negative if there is no aircraft waiting and the last touch-down took place $-u$ time units ago. Beside the loads u^I, u^{II} of the two runways, we also need the types of the aircraft involved in the assignment to be made. These are the types j^I, j^{II} of the two aircraft waiting at the tail of the queue behind one of which the arriving aircraft has to queue and the type k of the arriving aircraft itself, see Figure 2. For ease of presentation we assume for the moment that the aircraft do not need any additional time to proceed from the threshold TMA to the runways, so that if the assigned runway is free landing may start at once.

Hence the state of the system at the time of the arrival of an aircraft can be summarized as

$$s \;=\; (u^I, j^I, u^{II}, j^{II}; k), \tag{2}$$

see Figure 2 for an example. Now assume, that the n-th aircraft arriving at time T_n is of type $J_n = k$ and sees the state $s_n = (u^I, j^I, u^{II}, j^{II}; k)$ upon its arrival. If it is assigned to runway I then its waiting time would be

$$[u^I + d(j^I, k)]^+$$

where $[t]^+ = \max\{t, 0\}$ is the positive part of a real number t. To see why this is the waiting time, note that the new aircraft first has to wait for u^I time units until the aircraft (of type j^I) waiting in front of it begins its landing. In addition, it has to keep the separation time $d(j^I, k)$ to this predecessor resulting in $u^I + d(j^I, k)$. If this expression is positive it is the additional waiting time of the newly assigned aircraft in the queue. If it is negative or 0 , which may happen if the predecessor has landed a long time ago and u^I is negative, then the additional waiting time of the aircraft is zero, it can start its landing operation immediately. A similar argument applies for runway II, so that we can define the waiting time of the arriving aircraft of type k when it is assigned to runway $a \in \{I, II\}$ as the one-step-costs

$$c(s, a) \;:=\; \begin{cases} [u^I + d(j^I, k)]^+ & \text{if } a = I \\ [u^{II} + d(j^{II}, k)]^+ & \text{if } a = II \end{cases} \;=\; [u^a + d(j^a, k)]^+. \tag{3}$$

Assume that the assignment was to runway I and that the next aircraft is of type $J_{n+1} = l$ arriving t time units later at $T_{n+1} := T_n + t$. Then the next arrival event is $\xi = (t, l)$ and the $n+1$-st aircraft would see the load $[u^I + d(j^I, k)]^+ - t$ on runway I and load $u^{II} - t$ on runway II as t time units have elapsed. Waiting at the end of queue I is the newly assigned aircraft of type k, so that the new state, seen by the $n + 1$-st aircraft immediately before its own assignment is

$$s_{n+1} \;:=\; ([u^I + d(j^I, k)]^+ - t, k, u^{II} - t, j^{II}; l).$$

Generally, the transition function from state $s = (u^I, j^I, u^{II}, j^{II}; k)$ when action a was applied and the arrival event $\xi = (t, l)$ occurred is

$$g(s, a, (t, l)) \;:=\; \begin{cases} ([u^I + d(j^I, k)]^+ - t, k, u^{II} - t, j^{II}; l) & \text{if } a = I \\ (u^I - t, j^I, [u^{II} + d(j^{II}, k)]^+ - t, k; l) & \text{if } a = II \end{cases}. \tag{4}$$

Depending on the distribution of the arrival events, the sequence of states $s_1, s_2, \ldots$ becomes a stochastic process. If a strategy δ is applied, the n-th aircraft has waiting time $c(s_n, \delta(s_n))$ and we can define the expected average waiting time as

$$C(\delta) \;:=\; \mathbf{E}\left(\lim_{N\to\infty} \frac{1}{N} \sum_{n=1}^{N} c(s_n, \delta(s_n)) \right). \tag{5}$$

Finding a strategy δ that minimizes $C(\cdot)$ for a given event distribution (e.g. for a Poisson stream with a given type mix) requires the solution of a complex stochastic dynamic optimization problem, see [5] and [2] for more details about this. Nevertheless, the model sketched above can easily be used for simulation, see Section 5 below. We only need a source for the arrival events, then the state transitions and the determination of the waiting time can be performed as given in (4), (3).

Let us now discuss some simple strategies δ that could be used to assign the aircraft. First, the so-called round-robin strategy (also called 'stagger approach') simply switches between runway I and II, so that all aircraft with an even number are on one runway and all

with an odd number on the other. This strategy does not take into account the present state of the runway but simply tries to balance the loads. This can be done more efficiently by strategies of the type 'join-the-least-load' (JLL) (also called join-the-shortest-queue (JSQ)). In its most basic form, this strategy assigns an arriving aircraft to the shorter queue, i.e. to runway I if $u^I \leq u^{II}$ for the present state $s = (u^i, j^I, u^{II}, j^{II}; k)$. As the types of the aircraft involved are known, it is more reasonable to route to runway I if

$$[u^I + d(j^I, k)]^+ \ \leq \ [u^{II} + d(j^{II}, k)]^+ \tag{6}$$

i. e., if the waiting time on runway I would be less then on runway II. In terms of the MDP, this means that we choose that action (runway) a that minimizes the one-step-costs $c(s, a)$. Such a strategy does not take into account any long-term effects. In certain situations it may e.g. be reasonable to impose a larger waiting time on the present aircraft to keep a runway free for a heavy aircraft that may arrive next with high probability. Optimal solutions to the MDP balance the short-term and long-term effects of actions but, as was mentioned above, they are difficult to obtain due to the complexity of the state space. In the next two Sections we therefore study simpler strategies that do some balancing and perform much better than round-robin and JLL.

Before doing so let us shortly discuss a possible extension to our model. We can allow for an additional time τ that an aircraft needs to proceed from TMA to the runways. Then, the earliest landing time of an aircraft is its arrival time plus τ. In reality, τ may depend on the aircraft, the runway it is assigned to and sometimes even on the size of the waiting queue of the runway it is not assigned to but which it must circumfly. If we restrict ourselves to the simple case where τ depends only on the arriving aircraft we may include τ into the arrival event (t, l, τ). We can then extend our model to the important case where aircraft may arrive at an airport from different directions, so that they have trajectories of different lengths to their runways or queues.

The waiting time on runway I in (3) then changes to $[u^I - \tau + d(j^I, k)]^+$ as $u^I - \tau$ is the load the aircraft sees when it arrives at the runway. Its landing time is

$$[u^I - \tau + d(j^I, k)]^+ + \tau \ = \ \max\{\tau, [u^I + d(j^I, k)]^+\}.$$

Hence the load the next aircraft sees at its arrival as part of the next state s_{n+1} is $\max\{\tau, [u^I + d(j^I, k)]^+\} - t$ on runway I and $u^{II} - t$ on runway II.

However, for a runway system with high workload, a continuous occupancy of the runways without significant idle times can be expected. Thus it can be assumed that for most of the aircraft the earliest possible landing time is not of particular importance for the actual landing time. We shall therefore drop this feature from further discussions.

3 OPTIMAL STRATEGIES IN A REDUCED STATE SPACE

Formally, the state space S for a model as sketched in Section 2 is very large, it contains all real numbers u^I, u^{II} as possible loads of the runways. To derive optimal strategies becomes

a very difficult task in this situation. Also, from a practical point of view, it seems unlikely that a routing strategy δ that takes on only two values can make full use of the detailed information contained in the state $s = (u^I, j^I, u^{II}, j^{II}; k)$.

Instead, it seems reasonable to assume that the routing decision for an incoming aircraft will mainly depend on the *difference* of the loads on the two runways and not so much on their absolute values. Hence one could replace the original state $s = (u^I, j^I, u^{II}, j^{II}; k)$ by $(\Delta, j^I, j^{II}; k)$ where $\Delta = u^I - u^{II}$. Moreover, one would expect that a good strategy does not change its values (I or II) arbitrarily often as Δ varies. Hence, we may fix a few threshold values $x_1, \ldots, x_n$ and restrict ourselves to strategies $\hat{\delta}$ that change their values only at the thresholds. See Table 1 for an example with four thresholds.

Δ	$\Delta \le -120$	$-120 < \Delta \le -90$	$-90 < \Delta \le 90$	$90 < \Delta \le 120$	$120 < \Delta$
level d	0	1	2	3	4

Table 1: A simple threshold strategy with $n = 4$ symmetric thresholds $-120, -90, 90, 120$.

Restricting the class of strategies may also be viewed as replacing the original state s by a reduced state $\hat{s} := (d, j^I, j^{II}; k)$ where $d = 0, 1, \ldots, n$ denotes the *level* of the load difference determined by $x_d < u^I - u^{II} \le x_{d+1}$ for $1 \le d \le n-1$, $d = 0$ for $u^I - u^{II} \le x_1$ and $d = n$ for $x_n < u^I - u^{II}$, see Table 1. Note that our reduced state space $\hat{S}$ has only finitely many states, e.g. with four thresholds and three types of aircraft we have $5 \times 3^3 = 135$ reduced states in $\hat{S}$.

Our model has now become a special case of a so-called partially observable Markovian decision process (POMDP). These are generally used for systems in which the exact identification of the system state is impossible and only 'observations' of the true state are available to the decision maker. The structure of a POMDP is outlined in figure 3.

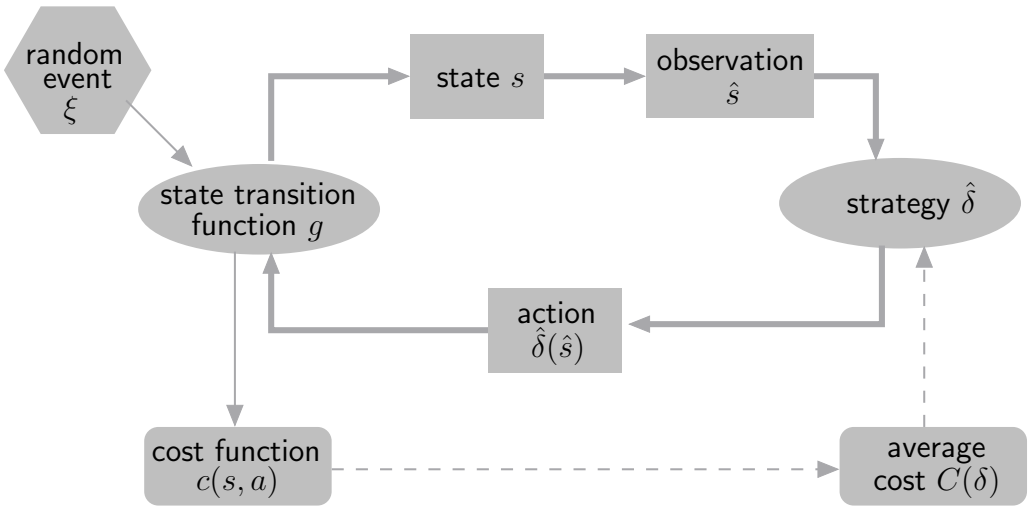

Figure 3: Partially Observable Markov Decision Process

Note that the restricted state space of observations is used for decision making only, the dynamics of the system still rely on the complete state s. In particular, when simulating the performance of a strategy $\hat{\delta}$, state transitions are as given in (4) but the action applied in state s is $\hat{\delta}(\hat{s})$.

The advantage of this approach is that we may use the reduced state space for a simplified search for good strategies $\hat{\delta}$, whereas the quality of $\hat{\delta}$ is measured against the dynamic system with complete information. Of course, we still have to decide how the state space is reduced, i.e. we have to decide on the number and on the values of the thresholds to be used.

Representation and Optimization of Threshold Strategies

The finite state space $\hat{S} = \{\hat{s}_1, \ldots, \hat{s}_N\}$ allows for a representation of a strategy $\hat{\delta}$ as a vector of length N. Its i-th entry is I if $\hat{\delta}(\hat{s}_i) = I$ and II if $\hat{\delta}(\hat{s}_i) = II$ for $i = 1, \ldots, N$. This vector is used as a look-up table to determine the action $\hat{\delta}(\hat{s})$ for each reduced state $\hat{s}$, see Table 2 for an example. Though the reduced state space is finite, the number of possible strategies is 2^N which still may be very large. Finding a good strategy $\hat{\delta}$ therefore remains a difficult task.

observation $\hat{s} = (d, j^I, j^{II}; k)$	runway $\hat{\delta}(\hat{s})$
(0,1,1;1)	I
(1,1,1;1)	I
(2,1,1;1)	II
(3,1,1;1)	II
(4,1,1;1)	II
(0,2,1;1)	I
(1,2,1;1)	I
(2,2,1;1)	I
(3,2,1;1)	II
$\vdots$	$\vdots$

Table 2: An example of a threshold strategy $\hat{\delta}$ with thresholds as in Table 1.

We have used local search for an iterative improvement of the strategies. Local search starts with an arbitrary solution and then randomly picks candidate solutions from the neighbourhood of the present solution. If the candidate is better, it becomes the new solution, otherwise a new candidate is picked.

If strategies are represented as vectors with entries I and II, a natural neighbourhood of $\hat{\delta}$ for the local search is the set of all strategies (vectors) that differ in exactly one position from $\hat{\delta}$. This means that we have to compare $\hat{\delta}$ with a candidate strategy $\hat{\delta}'$ that chooses a different runway for exactly one reduced state $\hat{s}'$.

The strategies are compared with respect to the average waiting times. These waiting times are estimated by the following procedure: a certain number H of arriving aircraft are simulated and their average waiting time over time horizon H is recorded. This is repeated several times until we obtain a reliable estimator of the average waiting time for the strategy used. As the two strategies to be compared differ only in a single reduced state $\hat{s}'$ all waiting times are equal for the two strategies until the reduced state $\hat{s}'$ is hit for the first time. Therefore, we only need to simulate the system after it has gone through $\hat{s}'$. The simplest way to achieve this would be to start the simulation of the runway system in the reduced state $\hat{s}'$.

However, the problem here is that there is a huge class $S' \subset S$ of full states that may underly the observation of the reduced state $\hat{s}' = (d, j^I, j^{II}; k)$. In fact, these are all states s' that have load differences on the level d. Starting with observation $\hat{s}'$ therefore means to select one of these states as starting state s' of the runway simulation system. However, if the results are to representative, this choice of s' must correspond to the frequency of the occurrence of s' during operation or simulation. This is not known and will generally depend on the strategies used.

Nevertheless, we can simulate this unknown distribution by starting the simulation system in an arbitrary but fixed state $s_0 \in S$, e.g. with empty runways. The system is run and its reduced states $\hat{s}$ are observed until the particular reduced state $\hat{s}'$ occurs for the first time. Now, the average waiting time w_1^H of the next H aircraft is determined. After that, simulation is continued without recording of waiting times until $\hat{s}'$ appears again starting another observation sequence of H aircraft resulting in average waiting time w_2^H. In this way average waiting times $w_1^H, \ldots, w_n^H$ with time horizon H are sampled until the estimator

$$\frac{1}{n}(w_1^H + \cdots + w_n^H)$$

has a prescribed accuracy determined from the 95% confidence interval. If the particular reduced state $\hat{s}'$ reappears before a sequence of H observations is finished, it is treated as an arbitrary state in order to keep the sampled sequences non-overlapping and independent.

Subsequently, the system is started again in state s_0 this time using the candidate strategy $\hat{\delta}'$. The estimated average waiting times over the finite horizon H for the two strategies are then compared and $\hat{\delta}'$ is either accepted as new solution or rejected.

We cannot guarantee that the state process starting in s_0 will ever hit the reduced state $\hat{s}'$ or that it will return to it later. If $\hat{s}'$ has not (re-)occurred after a fixed number of steps (aircraft), we consider $\hat{s}'$ as irrelevant for the present strategy and pick a new candidate from its neighbourhood.

The characteristics of the solutions which are generated using the local search algorithm depend crucially on the simulation horizon H. For small values of H, the resulting strategies are greedy and short-sighted similar to join-the-least-load. In the extreme case where $H = 1$, a change of the routing decision is accepted only if it improves the waiting time of the next arriving aircraft. In our experiments we found best results with horizons H between 5 and 10.

4 GENERALIZED JLL-STRATEGIES

We shall now examine the join-the-least-load strategy and variants of it more closely.

In its simplest form, JLL routes an arriving aircraft to runway I if $u^I \leq u^{II}$ in the present state $s = (u^I, j^I, u^{II}, j^{II}; k)$. If the waiting times are taken into account as in (6) we see that routing to runway I occurs if either $[u^I + d(j^I, k)]^+ = 0$, i.e. runway I is free, or if it is not free and

$$u^I \ \leq \ u^{II} + d(j^{II}, k) - d(j^I, k).$$

If we take (u^I, u^{II}) as a point in the two-dimensional plane, then these two strategies can be visualized as in Figure 4.

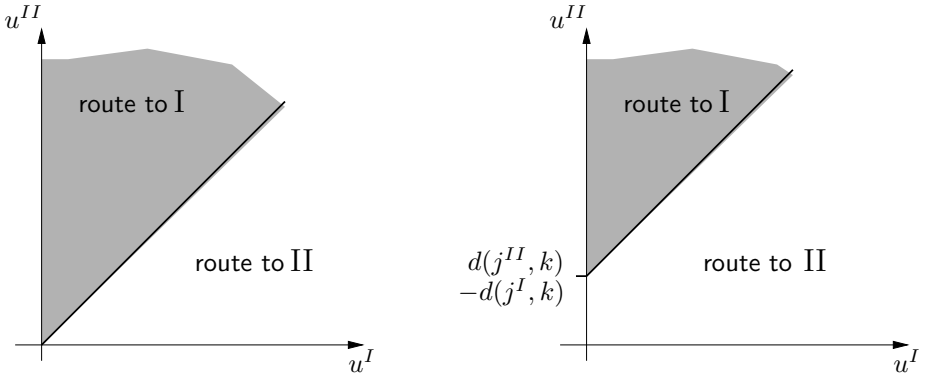

Figure 4: Simple JLL-strategies are characterized by a line parallel to the diagonal

A more general class of strategies 'JLL++' is obtained if one allows any straight line $A + Bx$ with coefficients A, B as a separator between the 'route to I' and 'route to II' regions as indicated in Figure 5. Moreover, these coefficients $A = A(j^I, j^{II}, k)$ and $B = B(j^I, j^{II}, k)$ may depend on the three types j^I, j^{II}, k of aircraft contained in state s.

Finding good strategies in this class JLL++ means to determine $(3 \times 3 \times 3)$-matrices A and B such that the decision rule $\delta_{A,B}$ has low average waiting time where $\delta_{A,B}$ is defined by

$$\delta_{A,B}(s) \ = \ I \quad \Longleftrightarrow \quad u^I \ \leq \ A(j^I, j^{II}, k) + B(j^I, j^{II}, k) \cdot u^{II} \tag{7}$$

with $s = (u^I, j^I, u^{II}, j^{II}; k)$. The average waiting time $C(\delta_{A,B})$ as defined in (5) has to be estimated for each pair A, B from simulation.

To find good parameters A, B we used genetic algorithms, see e. g. [9] for a general outline of these algorithms. Genetic algorithms use a set of solutions, the *population*, which is iteratively improved by applying operations like crossover and mutation.

In our case, the population consists of pairs of matrices (A, B) as the *individuals*. A starting population is made up from randomly chosen matrices. A *crossover* creates a new pair (A^+, B^+) from its two 'parents' $(A_1, B_1), (A_2, B_2)$ selected from the present population. A^+ could result from A_1 by exchanging part of the rows and columns with A_2 or by interpolation between A_1 and A_2 taken as points in the $3 \times 3 \times 3$-dimensional space.

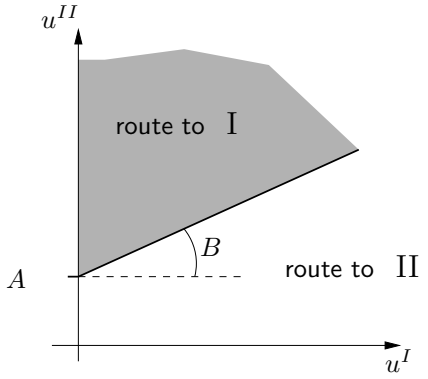

Figure 5: A generalized JLL-strategy, parameterized by $A = A(j^I, j^{II}, k)$ and $B = B(j^I, j^{II}, k)$.

Similarly, B^+ is made up from B_1, B_2. The result (A^+, B^+) hopefully inherits some of the good properties of its parents. It is then *mutated* by randomly changing some of the values in the matrices, typically by a small amount only.

In this way, a number of new offspring individuals are produced enlarging the present population. Then the 'fittest' individuals from this extended population are selected to form the next population. Here 'fitness' of an individual (A, B) is measured by the simulated average waiting time of the corresponding strategy $\delta_{A,B}$. The selection may be such that the best strategies only survive, but it is often more reasonable to randomize the selection and pick the individuals with a probability proportional to their average waiting times.

		$A^*(j^I, j^{II}, 1)$			$A^*(j^I, j^{II}, 2)$			$A^*(j^I, j^{II}, 3)$		
		j^{II}			j^{II}			j^{II}		
		1	2	3	1	2	3	1	2	3
j^I	1	-0.423	-0.307	0.814	-0.953	0.489	-0.864	-0.253	-12.576	-45.014
	2	0.569	-0.808	-0.433	0.697	0.81	-0.107	12.527	-0.179	-31.96
	3	0.121	0.658	-0.827	-0.591	0.917	-0.617	44.515	31.935	0.22

		$B^*(j^I, j^{II}, 1)$			$B^*(j^I, j^{II}, 2)$			$B^*(j^I, j^{II}, 3)$		
		j^{II}			j^{II}			j^{II}		
		1	2	3	1	2	3	1	2	3
j^I	1	0.472	1.742	1.129	1.803	0.056	0.511	1.042	1.479	1.129
	2	0.928	0.695	1.3	1.682	0.304	0.885	1.755	0.957	1.076
	3	0.511	0.142	0.019	0.661	0.472	1.862	0.69	0.499	1.138

Table 3: A solution (A^*, B^*) for the generalized JLL++ strategy found by genetic algorithms.

After several such 'generations', the population typically contains good solutions. The process can be improved if from time to time local optimization is applied to single indi-

viduals. This stops the genetic evolution for some time, but afterwards a new, often much improved individual is given back into the population. Its good 'genetic material' then improves the quality of the population during the next few generations.

Table 3 shows matrices (A^*, B^*) obtained from optimization with genetic algorithms. The corresponding strategy δ_{A^*,B^*} works as follows: if e.g. an arriving aircraft of type $k = 1$ ='heavy' finds an aircraft of type $j^I = 3$ = 'light' at the end of queue I and $j^{II} = 2$ = 'medium' at queue II then it would be routed to runway I if the load u^I on this runway is less than or equal to

$$A(3,2,1) + B(3,2,1) \cdot u^{II} \;=\; 0.658 + 0.142 \cdot u^{II}.$$

5 SIMULATION RESULTS

The simulation and optimization system we used were developed at UT Clausthal. The structure of the simulation tool is shown in Figure 6, it follows the structure of the Markov decision process as depicted in Figure 1. The arrival stream generator produces the random arrival events. They can be a truly random with inter-arrival times and types of aircraft selected according to some distribution but the events can also be derived from real arrival data of some airport. The central part of the system maintains the state of the runways and queues as formalized in (2). It uses a strategy δ for the routing decisions. Finally, the waiting times are recorded and averages and other statistical information from the simulation run are gathered.

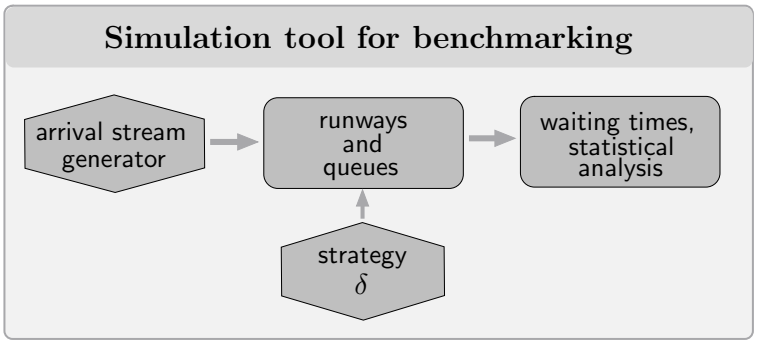

Figure 6: The structure of the simulation tool

In this set-up, the tool is used for the simulation and benchmarking of given strategies. It may also be used as a subsystem for the heuristic optimization of strategies as indicated in Figure 7. Here the output of a simulation run (or part of it) is used for the evaluation of strategies during the iterative improvement by local search or genetic algorithms as described in the previous chapters.

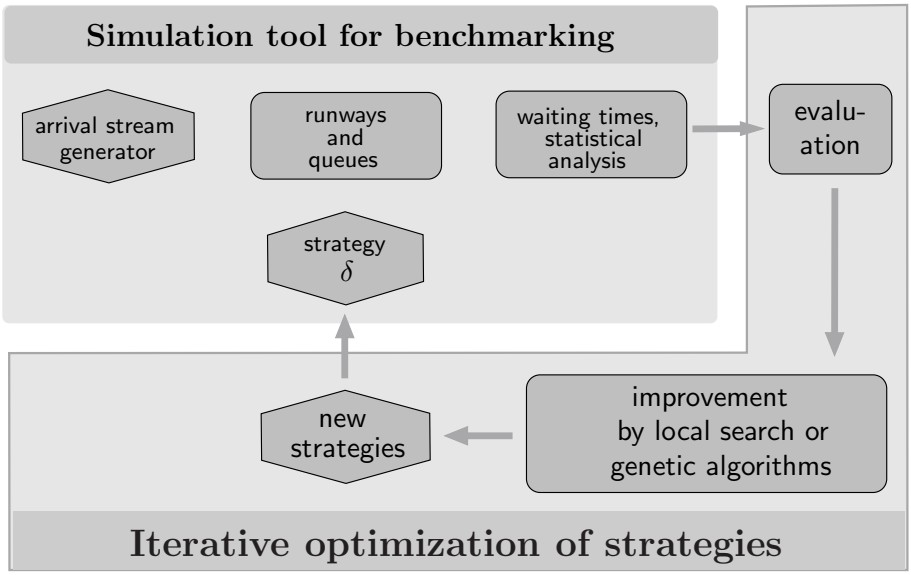

Figure 7: The interaction between optimization and simulation

Scenario Parameter

For the optimization and the comparison of different strategies we used a Poisson arrival stream where the inter-arrival times are independent and identically distributed according to an exponential distribution with different arrival rates λ. The weight class of arriving aircraft is determined randomly such that the arrival stream consists of 23% heavy, 72% medium and 5% of light aircraft. These are realistic data for larger airports.

The separation times that successive aircraft have to keep are determined by air traffic regulations. Table 4 shows data that were published for a major German hub. As the

		trailing		
		heavy	medium	light
leading	heavy	4	5	6
	medium	2,5	2,5	5
	light	2,5	2,5	2,5

Table 4: Separation distances in nautical miles

distances are typically given in nautical miles, we have to transform them into seconds. If we assume an average speed of 145kn for heavy aircraft, 140kn for medium sized and 130kn for light aircraft, the separation time at the runway threshold can be approximated by the times that are given in Table 5. These values were used for the separation matrix D as given in (1).

		trailing		
		heavy	medium	light
leading	heavy	124	153	191
	medium	87	89	163
	light	87	89	94

Table 5: Aircraft separation in seconds

Comparison of Different Strategies

We compared three different strategies. The first one is a threshold strategy of the reduced type described in Section 3. It was determined using the local search algorithm with a Poisson arrival stream with arrival rate $\lambda = 69.23$ aircraft per hour. The second strategy is the generalized JLL strategy $\delta_{A^*B^*}$ as given in Table 3. It was obtained from optimization by genetic algorithm using a Poisson stream with $\lambda = 67.35$. Finally, we included as an additional benchmark the simple JLL strategy as described in (6).

We fed these strategies into our simulation tool and estimated the average waiting time for different arrival rates varying from about 45 aircraft per hour up to more than 70. For each arrival rate, the simulation was carried on until the relative error of the estimated values as determined by the 95% confidence interval was below 5%. Results are shown in Figure 8. Typically, the simulated average waiting times first grow moderately with increasing arrival rate. When the capacity of the system is reached the waiting times explode, the system becomes congested. The actual capacity, i.e. the arrival rate that still can be handled by the system, obviously depends on the routing strategy used.

Figure 8 shows that JLL performs well for low arrival rates where aircraft often find empty runways. In this situation the threshold strategy creates larger waiting times. This picture changes as the arrival rate gets beyond 65 aircraft per hour. Here both our optimized strategies are much better than JLL. In particular, the generalized JLL strategy seems to be very good under all arrival rates.

However, these are only simulations with an artificial scenario in which arrival rates and type mix are assumed to be constant during the simulation run. In realistic arrival data both arrival rates and type mix change drastically during a day. In Figure 9, the upper oscillating curve shows the typical pattern of the arrival rate over 24 hours.

We therefore performed an additional simulation study with the simulation tool Simmod™ that is often used for airport capacity analysis with more realistic scenarios. The data used originate from a major German airport, they also include starting aircraft that use the same runways as the arriving ones. Instead of JLL, we used as a benchmark an example of an actual manual routing as it was performed by a flight guidance. Figure 9 shows the additional delay for arriving aircraft as it occurred under the manual routing and under a strategy of the reduced threshold type which was optimized by local search with these arrival data. The waiting times here are moving averages over intervals of 15 minutes.

In this study, the threshold strategy was superior to the manual routing in practically

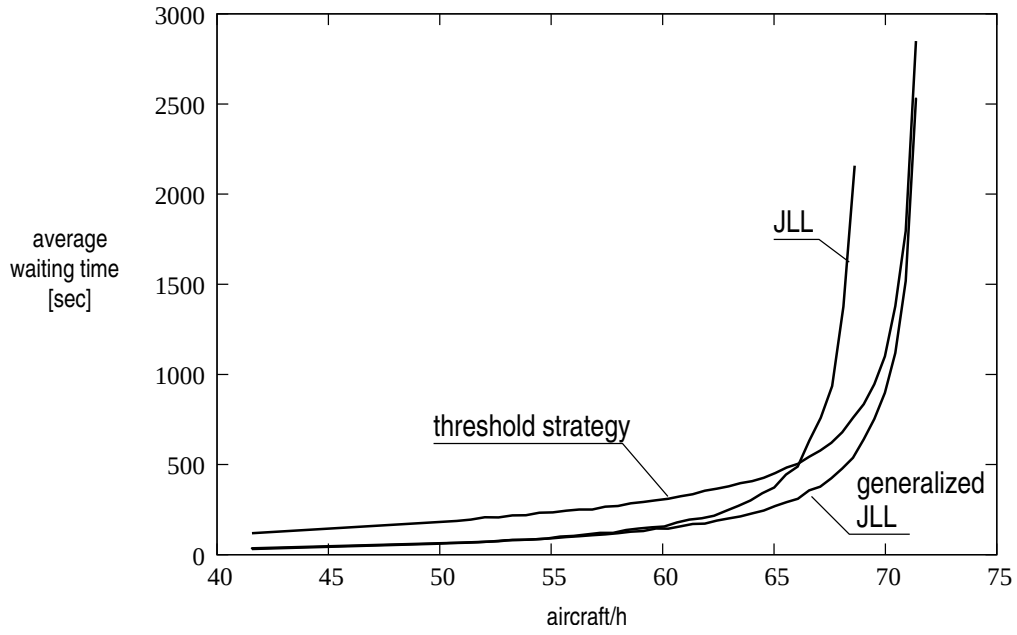

Figure 8: Average waiting times for increasing arrival rates

all situations, the reduction of delay is significant. Not shown in the Figure are delays for departures which were included in the Simmod^{TM} scenario. It turned out that these could also be reduced if landing aircraft were assigned by the threshold strategy.

At the time of writing, the corresponding results for the generalized JLL strategy were not yet available.

6 CONCLUSION

We have have derived two types of dynamic strategies for the assignment of arriving aircraft to runways. Both offer significant potential for the reduction of delays occurring in heavy traffic situations. Due to their simple structure both type of strategies can be evaluated in real time and could be included into arrival management tools and decision support systems for air traffic management.

We restricted ourselves to an assignment of single aircraft in a first-come first-served fashion. These assignments can be made at an early stage of the arrival process so that air traffic management has enough time to calculate a safe trajectory to the assigned runway. Furthermore first-come first-served methods are considered as fair by the parties involved, especially the airlines, which is important for the acceptance of a computer based arrival management.

The simulation based approach used for the determination of our strategies also allows for an easy inclusion of additional constraints, e. g. dependencies between runways or ad-

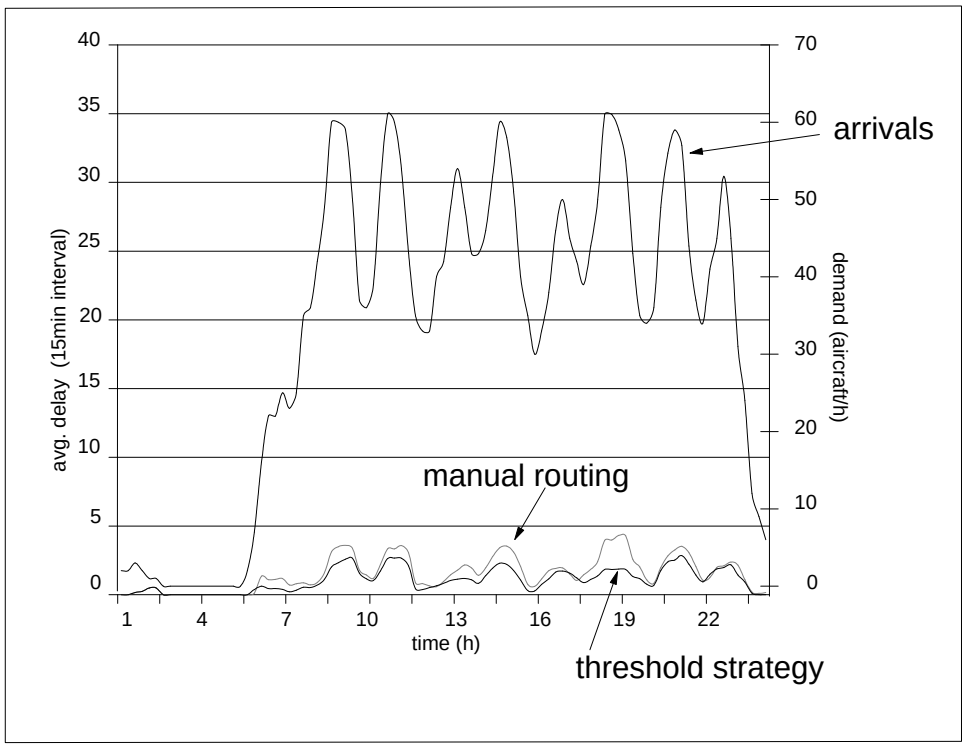

Figure 9: Comparison of delays under a manual routing and the routing according to a threshold strategy.

ditional weight classes.

Our future research will concentrate on strategies that adapt themselves to the changing arrival patterns as they appear in real operations. Further simulation studies under realistic scenarios have to be carried out.

References

[1] Bäuerle, N., Engelhardt-Funke, O. and Kolonko M. (2007) On the Waiting Time of Arriving Aircrafts and the Capacity of Airports with One or Two Runways.*Europ. J. Operational Research*, **177**, 1180-1196.

[2] Bäuerle, N., Engelhardt-Funke, O. and Kolonko, M. (2004) Routing of Aircrafts to Two Runways: Monotonicity of Optimal Controls. *Probability in the Engineering and Informational Sciences* **18**, 533-560.

[3] Beasley, J.E., Krishnamoorty, M., Sharaiha, Y.M. and Abramson, D. (2000) Scheduling Aircraft Landings - The Static Case. *Transportation Science* **34**, p 180-197.

[4] Bolender, M.A. and Slater, G.L. (2000) Evaluation of Scheduling Methods for Multiple Runways. *Journal of Aircrafts* **37**, 410 - 416.

[5] Bertsekas, D. (1987) *Dynamic Programming: Deterministic and Stochastic Models.* Prentice Hall, Englewood Cliffs, N.J.

[6] Bertsekas, D. and Tsitsiklis, J. N. (1996) *Neuro-Dynamic Programming.* Prentice Hall, Englewood Cliffs, N.J.

[7] Horonjeff, R. and MCKelvey, F.X. (1994) *Planning and Design of Airports.* McGraw-Hill, 4. ed., Boston.

[8] Neuts, M.F. (1989) *Structured Stochastic Matrices of M/G/1 Type and Their Applications* Marcel Dekker, New York and Basel.

[9] Reeves, C. R. and Rowe, J. E. (2003) *Genetic Algorithms – Principles and Perspectives*, Kluwer Academic Publishers, Boston.