A Real-time Active Routing Approach via a Database for Airport Surface Movement

Michal Weiszer[a,*], Jun Chen[a], Paul Stewart[b]

[a]*School of Engineering, University of Lincoln, Brayford Pool, Lincoln, United Kingdom*
[b]*Institute for Innovation in Sustainable Engineering, Derby, United Kingdom*

Abstract

Airports face challenges due to the increasing volume of air traffic and tighter environmental restrictions which result in a need to actively integrate speed profiles into conventional routing and scheduling procedure. However, only until very recently, the research on airport ground movement has started to take into account such a speed profile optimisation problem actively so that not only time efficiency but also fuel saving and decrease in airport emissions can be achieved at the same time. It is envisioned that the realism of planning could also be improved through speed profiles. However, due to the multi-objective nature of the problem and complexity of the investigated models (objective functions), the existing speed profile optimisation approach features high computational demand and is not suitable for an on-line application. In order to make this approach more competitive for real-world application and to meet limits imposed by International Civil Aviation Organization for on-line decision time, this paper introduces a pre-computed database acting as a middleware to effectively separate the planning (routing and scheduling) module and the speed profile generation module. Employing a database not only circumvents duplicative optimisation for the same taxiway segments, but also completely avoids the computation of speed profiles during the on-line decision support owing a great deal to newly proposed database initialization procedures. Moreover, the added layer of database facilitates, in the future, more complex and realistic models to be considered in the speed profile generation module, without sacrificing on-line decision time. The experimental results carried out using data from a major European hub show that the proposed approach is promising in speeding up the search process.

Keywords: Airport operations, Environmental impact, Ground movement, Multi-objective optimisation

*Corresponding author
Email addresses: `mweiszer@lincoln.ac.uk` (Michal Weiszer), `juchen@lincoln.ac.uk` (Jun Chen), `p.stewart1@derby.ac.uk` (Paul Stewart)

1. Introduction

European airports are likely to become bottlenecks in the air transportation system due to the forecast growth in traffic and passenger numbers (EUROCONTROL, 2013). As many airports operate approach maximum capacity, the European Commission (2011) recognized the need for airport capacity to be increased in order to mitigate the growing demand for travel. To increase throughput, large investments in infrastructure of airports have to be made and/or the operation of airports have to be optimised to fully utilise the available resources. From an optimisation point of view, ground movement is one of the key airside operations at the airport as it links other airport operations such as departure sequencing, arrival sequencing and gate/stand allocation and its performance can affect each of these (Atkin et al., 2010b). Therefore, any improvement in ground movement leading to time efficient operation will be of significant importance to airport stakeholders.

Another great challenge faced not only by airports but the whole air transportation sector is environmental impact. The white paper issued by European Commission (2011) outlines very ambitious environmental targets for air transportation. Stricter emissions regulations together with efforts of airlines to reduce fuel costs result in a demand to cut fuel consumption. As shown in (Ravizza et al., 2013b), minimum time taxiing and minimum fuel burn are conflicting objectives, as shorter times normally lead to higher rates and longer periods of acceleration. Ideally, both time and environmental objectives should be minimized simultaneously, in a form of a global optimum.

Previous research on ground movements mostly focus on the taxi time objective with aircraft assumed to taxi with a speed which is constant or within some defined boundaries. Minimisation of the total taxi time is the main goal of the genetic algorithm used by Pesic et al. (2001). A time-space network is employed in (Marín, 2006; Roling and Visser, 2008) to solve the mixed integer linear programming formulations of the problem with the same objective function of the minimum taxi time. A sequential, label-setting the Quickest Path Problem with Time windows (QPPTW) algorithm working on a graph representation is proposed in (Ravizza et al., 2013a). A similar graph-based approach is utilised by Lesire (2010) who devised a modification of the A* algorithm to route and schedule aircraft. The total taxi time is minimised in both graph-based algorithms. In addition to the total taxi time, several researchers also considered other time related objectives. Deviations from the scheduled time of departure or arrival are penalized in (Balakrishnan and Jung, 2007; Smeltink et al., 2004). A genetic algorithm employed by Gotteland et al. (2003) and Deau et al. (2009) minimises the taxi time together with the deviation from assigned slots. Similarly, a weighted sum of objectives including the total taxi time, the delays for arrivals and departures, the number of arrivals and take-offs, the worst routing time and the number of controller's interventions is minimised in (Marín and Codina, 2008). The paper by García et al. (2005) minimises another time related objective: the makespan, i.e. the duration from the first to the last aircraft movement. A mixed integer linear programming formulation by

Clare and Richards (2011) minimises a weighted sum of taxi time and distance related objectives with respect to runway scheduling constraints. As all the aforementioned algorithms do not consider conformity to real-life scenarios, the assumption that the participating aircraft can meet the given time slots without excessive acceleration/deceleration is questionable.

Fuel consumption is only taken into account indirectly in work focusing on the stand holding problem (Atkin et al., 2010a, 2011; Burgain et al., 2009), where the primary aim is to maximise the time an aircraft spends at the stand, with their engines off, rather than taxiing. The main assumption is that a shorter taxi time will result in lower fuel burn. More recently, following a wide adoption of the *4D trajectory* concept (consisting of three spatial dimensions and time as the fourth dimension) in other air transport research, e.g. (Ruiz et al., 2013; Yousefi and Zadeh, 2013; Zúñiga et al., 2013), a few researchers have started to consider a related approach during ground movement. However, for the purpose of ground movement in this paper, not all dimensions are required as aircraft's movement are bounded by taxiways. In this case, it is sufficient to completely define their position in time with routes and speed profiles. Therefore, for consistency and clarity, speed profile is the term used throughout the paper. While the total taxi time remained the main objective of optimisation in previous studies, the speed profile is generated in a post-processing manner with respect to the optimised taxi times. A surface management tool TRACC (Taxi Routing for Aircraft: Creation and Controlling) (Schaper and Gerdes, 2013) employs a genetic algorithm to optimize routes of aircraft. The output specifies the route, speed profile and holding times for each aircraft. A similar system, the Ground-Operation Situation Awareness and Flow Efficiency (GoSafe) system (Cheng and Sweriduk, 2009) utilises dynamic programming for taxi route optimisation. However, it is worth pointing out that none of these methods take into account speed profiles proactively in their planning modules, leading to suboptimal speed profiles in terms of fuel consumption.

A recently published paper by Ravizza et al. (2013b) presents a new concept for the ground movement problem which uses multi-objective optimisation to simultaneously optimise routing, scheduling and speed profiles, with regard to taxi time and fuel consumption. In their approach, the routing and scheduling algorithm (Ravizza et al., 2013a) is combined with the Population Adaptive based Immune Algorithm (PAIA) (Chen and Stewart, 2011; Chen and Mahfouf, 2006) in search of the trade-off between the total taxi time and fuel consumption. However, fuel consumption is represented by a fuel index rather than actual fuel burn, and the final decision is left to controllers' subjective judgement without any quantitative indicators. Based upon (Ravizza et al., 2013b), an Active Routing (AR) framework is proposed in (Chen et al., 2015b,a), aiming at more seamless integration of speed profiles into route and schedule optimisation. A more detailed actual fuel burn modelling and an airport economic optimisation framework are also introduced to facilitate controllers making a more objective decision. The ultimate aim is to produce a more realistic, cost effective, and greener ground movement. Although the aforementioned framework is flexible to include more factors, such as a noise model, in a more holistic way, it suffers from high computational demand. The work proposed in (Weiszer et al., 2014) attempted to speed up the search process using a heuristic procedure for speed profile optimisation. Despite improvement, the computational time of the optimisation framework, as well as the realism of the assumed simplified aircraft dynamic model and fuel consumption model, still prohibits its effective use in a real-time airport decision support environment. Experience of high computational demand in generating speed profiles is also evident in other application fields, such as in car (Mensing et al., 2011, 2014) or train (Li and Lo, 2014) speed profile optimisation. With more complex models, particularly with exact methods such as dynamic programming, speed profile optimisation is generally computationally intensive and not suitable for on-line, real-time optimisation.

A common approach to overcome the burden of high computational demand of complex evaluation functions during optimisation is the application of surrogate models (Forrester and Keane, 2009) to replace the original expensive objective functions by their computationally cheaper approximations to estimate fitness values of solutions. However, several issues may arise when applying surrogate-based optimisation to the airport ground movement problem. As complete speed profiles are required by aircraft to follow, 1) surrogate models would need to estimate not only fitness of individual solutions but also variables defining speed profiles, and 2) the exact model would need to be run on final solutions to provide speed profiles. In the first case, the construction of such a surrogate model may be difficult or even impossible due to the high number of variables defining speed profiles required, and in the second case, real-time use of optimisation is still prohibited by the associated computational overhead.

In the context of the abovementioned issues, the research presented in this paper further extends the proposed ground movement optimisation framework in (Ravizza et al., 2013b; Weiszer et al., 2014; Chen et al., 2015b) to make it fully applicable in real-time airport management systems. The main idea is to use a pre-computed database consisting of optimised speed profiles in order to avoid duplicative optimisation for the same taxiway segments. A similar concept of pre-computation and storage of solutions to speed up the on-line optimisation procedure has been observed in other application domains, for example see (Lewis et al., 2009; Wagner et al., 2013; Sanders and Schultes, 2007). The main difference and novelty of the proposed approach in this paper compared to those methods will be discussed in Section 2.4.

This paper presents a valid and realistic approach towards on-line decision support for airport ground movement due to the following:

1. In contrast to previous research (Ravizza et al., 2013b; Weiszer et al., 2014) which considered whole routes for taxiing aircraft, in this paper we deconstruct the original problem into a set of independent subproblems for individual taxiway segments and by doing so take advantage of using stored speed profiles in a database instead of costly on-line optimisation,
2. a database of optimised speed profiles is initialized with a limited set of solutions such that the combination of speed profiles retrieved from the database can represent any possible route on the airport.

The addition of such a database liberates the routing and scheduling module from speed profile generation during the on-line search. The saving in computational time meets the limits imposed by the International Civil Aviation Organization (ICAO) for on-line decision time within the Advanced Surface Movement Guidance and Control Systems (ICAO, 2004). Since the database is initialized before the actual route planning, the proposed AR framework can be extended in future to incorporate a more complex and realistic aircraft dynamic model, fuel consumption model, and other airport environmental related model, such as noise and pollution models, without increasing on-line decision time.

The subsequent parts of this paper are organised as follows: Section 2 introduces the Active Routing concept including the related routing, scheduling and speed profile optimisation subproblems. The combined solution method and the proposed real-time approach utilising a database is described in Section 3. The computational experiments with the algorithm are carried out on instances from Zürich Airport in Section 4. Finally, conclusions are drawn in Section 5.

2. Active routing (AR)

2.1. A Multi-objective and Multi-component (MOMC) Approach

The optimisation framework for the ground movement problem proposed in (Ravizza et al., 2013b; Chen et al., 2015a) introduces a new concept, which in the light of previous research, can be called *Active Routing (AR)*. The name is an acknowledgement of the fact that optimised speed profiles are seamlessly embedded in the search of the optimal routes and schedules and multi-faceted needs of stakeholders are proactively considered. Furthermore due to the adoption of multi-objective framework, many objectives could be taken into account at the same time in speed profile generation. In addition to multi-objective optimisation, an economic decision making model selects the routes and speed profiles which will be assigned for the aircraft to follow. The results from AR will be given to the guidance function for a more realistic guidance and control. The AR concept is illustrated in Fig. 1.

From an optimisation perspective, the aforementioned concept consists of two interdependent optimisation problems, which are also known as multi-component optimisation problems (Bonyadi et al., 2013):

- the routing and scheduling problem,
- the speed profile optimisation problem.

The AR concept as proposed in (Ravizza et al., 2013b; Chen et al., 2015a), minimises two objective functions:

- g_1: total taxi time (s),
- g_2: fuel consumption (kg).

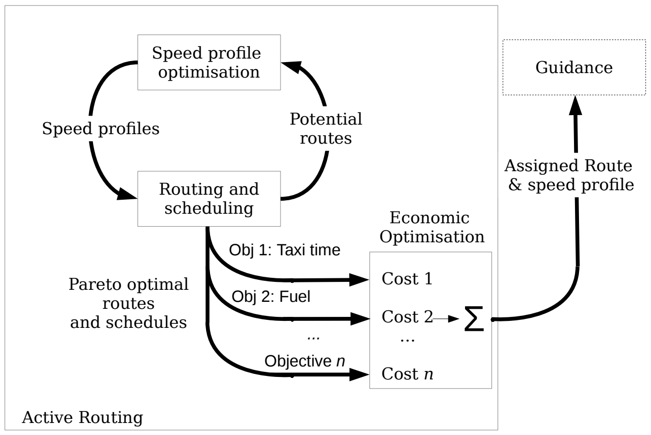


Figure 1: Active Routing concept for the ground movement problem.

2.2. Routing and scheduling problem

The objective of the routing and scheduling problem is to route aircraft from source to destination locations in a time and fuel efficient manner, respecting routes and schedules of other aircraft while preventing conflicts between them. The airport surface is represented as a directed graph $G = (V, E)$, where the edges $e \in E$ represent the taxiways and the vertices $n \in V$ represent the taxiway crossings, intermediate points and sources/destinations such as gates, stands and runway exit points. For the sake of simplicity and without the loss of generality, all edges of the taxiway network used in this paper are bidirectional. Aircraft are considered to occupy edges and only one aircraft can travel along one edge at a time so that a minimum safety distance from all other aircraft is ensured. The period when the edge is not used by any other aircraft is also called a *time window*.

In this paper, the k-Quickest Path Problem with Time Windows (k-QPPTW) (Ravizza et al., 2013b) is used to solve this problem, however it is also possible to employ other routing and scheduling heuristic methods such as (Mandow and De La Cruz, 2010; Guan et al., 2013; Xue-Jun et al., 2013). The algorithm sequentially routes aircraft according to their pushback/landing time, respecting time windows corresponding to edges. The k-QPPTW algorithm generates a set of k-best solutions with regard to minimum taxi time, based on the maximum allowed speed. These potential routes are passed as an input to the speed profile optimisation problem which is described in detail in Section 2.3.

2.3. Speed profile optimisation problem

The objective of the speed profile optimisation problem is to identify a set of Pareto optimal unimpeded speed profiles for a given route, simultaneously minimising total taxi time given in seconds (objective g_1) and related fuel consumption in kilograms (objective g_2). In order to keep the problem tractable, the route of an aircraft is further divided into larger segments, each consisting of multiple edges. Instead of searching for speed profiles for individual edges, a speed profile is generated over segments. This effectively restricts the search space by allowing the linear piece-wise speed profile described later in this section. The division into segments facilitates speed profile optimisation as, for each segment type, a typical taxiing behaviour can be identified according to the initial speed v_0 of an aircraft, end speed v_4 and the maximum permitted speed v_{max} as given in Table 1. Fig. 2 illustrates the definitions of different segment types. Considering edges of the route iteratively in the direction of the taxiing aircraft, edge e is considered as part of a straight segment if the angle between the edge e and its predecessor edge $e-1$ is less than 30 degrees (edges between 1–4, 6–9). Otherwise, the edge is marked as part of a turning segment (edges between 4–5, 5–6). Consecutive edges of the same segment type are grouped together to form one segment. If a straight segment is ended by a runway exit/gate, it is marked as a straight holding/parking segment. Similarly, if a straight segment begins with a runway exit/gate, it is marked as a straight breakaway segment.

Table 1: Classification of segments according to the initial speed v_0 of an aircraft, end speed v_4 and the maximum permitted speed v_{max} in knots.

Type	v_0 (kn)	v_4 (kn)	v_{max} (kn)
straight breakaway	0	10	30
straight holding/parking	10	0	30
straight	10	10	30
turning	10	10	10

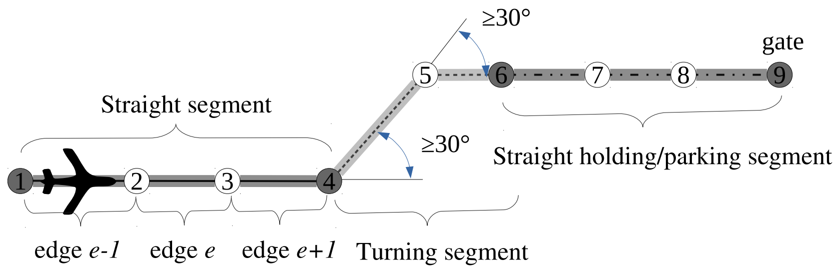


Figure 2: An example of a taxiway, consisting of edges. Multiple edges form segments of different types (straight, turning, straight holding/parking), depending on the topology and taxi direction.

As we are only interested in unimpeded aircraft taxiing within each segment and leave the interaction of aircraft to the routing and scheduling function, the speed profile optimisation problem for each segment can be solved independently without considering other segments, thus turning the original problem of speed profile optimisation for a given route into a set of smaller problems.

In order to further reduce the complexity of the speed profile optimisation problem, each straight segment of the taxiway is divided into four parts, corresponding to four different aircraft taxiing phases, i.e. acceleration, travelling at constant speed, braking and rapid braking, representing a typical taxiing behaviour as illustrated in Fig. 3. The first phase is the acceleration phase in which an aircraft maintains a constant acceleration rate a_1 over the distance d_1, thus increasing its speed from the initial speed v_0 at the start of the segment to v_1. During the second phase, an aircraft will traverse at the constant speed v_1 until the end of the second phase d_2 is reached. In the third and the fourth phases, an aircraft will decelerate from the speed v_1 to the speed v_4 at the end of the segment. The last two phases have different deceleration rates where, a_4 is equal to the maximum deceleration rate which enables the speed to be quickly reduced to v_4. With regards to the third phase, the deceleration rate a_3 will be uniquely determined by a_4 and d_4, since v_3 can be derived backwards given a_4, v_4, d_4 and the length of the third phase is equal to $d_3 = d - d_1 - d_2 - d_4$.

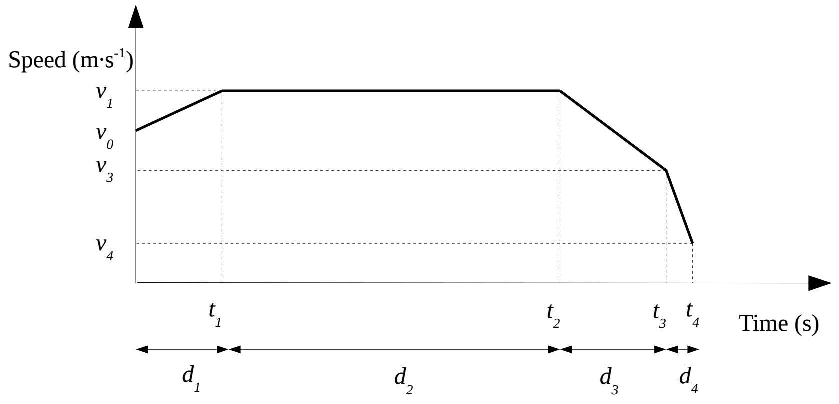


Figure 3: An example of a speed profile with four phases.

For turning segments we assume that the aircraft will have a constant speed v_{turn}. The maximum speed on straight taxiways $v_{straight}$ is restricted to 30 knots (15.43 m·s^{-1}) and turning speed v_{turn} is set to 10 knots (5.14 m·s^{-1}). Furthermore, the maximum acceleration and deceleration rate a_{max} is set to 0.98 m·s^{-2} for passenger comfort (Chen and Stewart, 2011).

Consequently, there are four free variables a_1, d_1, d_2, d_4 which completely define a unique speed profile over a segment. However, variables a_1, d_1, d_2, d_4 have to satisfy physical constraints in order to be feasible. The constraints are determined in a sequential manner where once a constraint has been calculated it serves as an input for the next computation:

1. Firstly, the upper and lower bounds for a_1 are defined as follows. The upper bound a_1^u equals to the maximum acceleration a_{max}. The lower bound a_1^l corresponds to a situation where an aircraft constantly accelerates over the segment and must ensure that at the end of the segment v_4 can be

reached:

$$a_1^l = \frac{v_4^2 - v_0^2}{2 \cdot d}. \tag{1}$$

2. Secondly, the bounds for the distance of the first phase d_1 are determined after a_1 has been fixed. The lower bound d_1^l must satisfy Eq. 2 to reach the end speed v_4 of the edge.

$$d_1^l = \frac{v_4^2 - v_0^2}{2 \cdot a_1} \tag{2}$$

   d_1^u represents one of the extreme situations where aircraft can only accelerate within such a distance, beyond which its taxi speed may exceed the maximum allowed speed $v_{straight}$, or the remaining distance may not be long enough for aircraft to reduce its taxi speed to v_4 even with a_{max}.
3. Once d_1 has been fixed, v_1 is also fixed and can be used to determine d_2^u and d_2^l:

$$d_2^u = d - d_1 - \frac{v_1^2 - v_4^2}{2 \cdot a_{max}}, \tag{3}$$

$$d_2^l = d - d_1 - \frac{v_1^2 - v_4^2}{2 \cdot a_{min}^d}. \tag{4}$$

   Where a_{min}^d is defined in Eq. 5 and represents the situation where there is only one deceleration phase with a small deceleration rate a_{min}^d and consequently, aircraft have to decelerate earlier.

$$a_{min}^d = \frac{v_1^2 - v_4^2}{2 \cdot (d - d_1)} \tag{5}$$

   The upper bound of d_2 represents the situation where d_3 does not exist and aircraft have to decelerate with a_{max}.
4. Finally, after determining d_2 within its feasible bounds d_4^u is calculated according to Eq. 6 which refers to the situation when d_3 does not exist.The lower bound d_4^l is set to 0.

$$d_4^u = \frac{v_1^2 - v_4^2}{2 \cdot a_{max}} \tag{6}$$

Once the feasible values of decision variables have been defined, a unique speed profile of which an example is shown in Fig. 3, can be determined. Based on the derived speed profile, taxi time (objective g_1) and fuel consumption (objective g_2) are calculated. In this paper, the method based on the ICAO database (Chen et al., 2015b) is used to model fuel consumption, which consists of physics-based equations taking into account the acceleration force and rolling resistance to calculate thrust. The calculated thrust and aircraft engines will then be mapped into corresponding fuel flows according to the ICAO database.

Algorithm 1 Heuristic for speed profile optimisation.

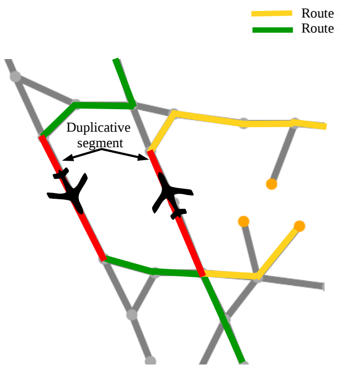
```
1:  $p = 0$;
2:  for $v = v_{turn}$ to $v_{straight}$ step $s$ do
3:      generate speed profile with $v_{max} = v$;
4:      $p = p + 1$;
5:  end for
6:  for weight $w_1 = 0$ to 1 step $\frac{1}{p}$ do
7:      $w_2 = 1 - w_1$;
8:      assign utility $w_1 \cdot g_1 + w_2 \cdot g_2$ to every speed profile generated in line 3;
9:      select speed profile with the minimum utility;
10:     assign parameters $a_1, d_1, d_2, d_4$ to one solution on the Pareto front;
11: end for
12: return  set of $p$ solutions approximating the Pareto front;
```

Therefore, by searching for the values of a_1, d_1, d_2, d_4, one can explore different speed profiles with different taxi time and fuel consumption.

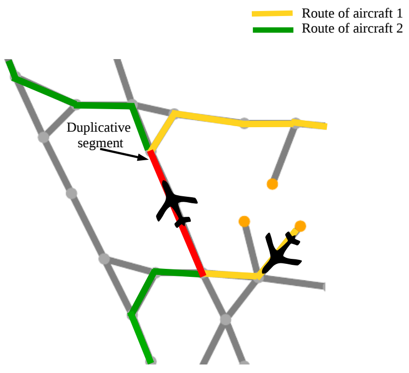
To perform the search, different search algorithms can be employed such as PAIA (Chen and Stewart, 2011; Chen and Mahfouf, 2006). However, as documented in (Weiszer et al., 2014; Chen et al., 2015b), PAIA required a comparatively large computation time. In this paper, a further simplified heuristic (Weiszer et al., 2014) is used for the speed profile optimisation. The heuristic is based on observations noted during experiments with PAIA and constrains the search space by determining some of the original decision variables a_1, d_1, d_2, d_4 beforehand. Firstly, the decision variable a_1 is fixed to 0.98 m·s^{-2}. Secondly, the distance d_2 during which the aircraft travels at constant speed v_1 is maximised, since braking will not save fuel, but will increase traversing time. With maximised d_2 the rapid braking distance d_4 using deceleration $a_{max} = 0.98$ m·s^{-2} to slow down from v_1 to v_4 is set to d_4^u and can be easily calculated using Eq. 6. The only decision variable left undecided is the acceleration distance d_1 which affects the maximum speed v_1 that can be achieved over the segment. The maximum speed v_1 affects the fuel consumption as well as the time needed to traverse the segment. The remaining task is to search for the optimal values of v_1 and hence d_1.

The search for a trade-off is performed as described by Algorithm 1. The heuristic starts by iteratively generating speed profiles for the input segment with the maximum speed v_1 set to a value from v_1= 5.14 m·s^{-1} (10 knots) to v_1= 15.43 m·s^{-1} (30 knots), with step s. In total, p solutions are generated.

In order to construct the Pareto front for the segment, the subroutine (lines 6–11) iteratively selects weights for p iterations in total. The solutions generated in line 3 are ranked according to utility obtained by linear combination of weighted taxi time (objective g_1) and fuel consumption (objective g_2). The solution with the best (i.e. minimum) utility is selected and in line 10 the parameters a_1, d_1, d_2, d_4 are assigned to one solution on the Pareto front.



(a)



(b)

Figure 4: Examples of duplicative segments: (a) The same segment is a part of different routes; (b) Different routes contain a segment with the same characteristics.

2.4. Problems of the MOMC Approach

As one can see from Section 2.1 and 2.3, a number of difficulties arise when such an approach is applied in a real-world environment:

1. Duplicative optimisation of speed profiles: as can be seen in Fig. 1 the speed profile is optimised each time an aircraft is being scheduled which may result in a time consuming redundant optimisation for the same routes. On the other hand, since routes are further divided into segments as described in Section 2.3, it may result in a duplicative speed profile optimisation for segments with the same properties:
   (a) as indicated in Fig. 4a, different routes forof different aircraft may contain the same segments, thus making the speed profile optimisation problem duplicative for these segments,
   (b) as illustrated in Fig. 4b, different segments of the taxiway may have the same properties, i.e. type and length, therefore speed profile optimisation for segments with the same properties is duplicated.

   As shown in (Chen et al., 2015b), by using a generic optimisation algorithm, such as PAIA, long computational times were observed. Moreover, a tailored and simplified heuristic devised in (Weiszer et al., 2014), even seeded as an initial population in PAIA, did not improve the performance of the generic optimisation algorithm significantly. Even the heuristic method itself used alone was still not suitable for on-line optimisation. Therefore, further speed-up by removing redundant searches is required in order to facilitate real-time optimisation.
2. Restrictions on complexity of included models: as computational time is crucial for on-line application of the AR framework on the airport, the current approach is constrained by using a simplified aircraft dynamic model and piece-wise linear speed profile. However, a more detailed complex aircraft dynamic model should be able to give a more realistic and accurate

fuel estimation. How to include more detailed models while still keeping the problem tractable is the biggest challenge facing an AR approach.

3. Restrictions on the number of objectives: at most two objectives have been considered so far within the optimisation framework. The stakeholders may be interested in other objectives as well, for example noise, emissions or cost. However, an increase in the number of objectives will inevitably lead to higher overall complexity and longer computational times.
4. Restrictions on speed profile representation: the involvement of more complex aircraft dynamic and fuel consumption models will unavoidably give rise to non-linear speed profiles; although more realistic and may produce better solutions, similarly as above, this will result in increased complexity of the optimisation framework.

In light of the above discussion, a method which can avoid on-line computationally expensive optimisation of speed profiles is required. Caching of segments and associated optimal speed profiles into a database could successfully speed up the search, since the algorithm would use only cached solutions instead of costly on-line optimisation. However, two questions arise: a) how to determine which segments should be pre-computed and stored? b) do we have to store all possible routes? Ideally, if one could identify a limited number of fundamental building blocks, each of them representing certain type and length (properties) of a representative segment, then any route on the airport could be recreated by combining some of these building blocks. The requirement for memorising the resulting building blocks falls into a common and well-established memoization technique which stores results of expensive evaluation functions for re-use during on-line computation (Michie, 1968). In the following analysis, several implementations of such a technique are discussed and compared with the proposed implementation in this paper. The novelties of the proposed approach are also highlighted.

Wagner et al. Wagner et al. (2013) used a database of previously computed and memoized solutions to speed up the optimisation of wind turbine layouts. The database is initially empty and is gradually populated during the search. If a newly encountered turbine layout is the same as the previously evaluated one, the evaluation is retrieved from the database. Otherwise, the evaluation of the previously found similar layout is used as a building block and only the difference in the layout is calculated. Multi-objective optimisation of RFID antennas presented in (Lewis et al., 2009) employs a similar approach, in which a database of previously evaluated solutions is filled during the search. Memoization for image processing (Khalvati et al., 2011) divides an image and stores fixed-sized building blocks of pixels so that they can be reused during on-line image processing. However, it should be noted, that each time a new block of pixels is evaluated, the algorithm still needs to perform calculations on-line. Approaches described above represent one type of application where a stringent real-time decision is not required. Furthermore, the pre-computed database does not have to be complete. Here, “complete” is defined in the sense that no on-line calculation is needed. Any newly encountered scenarios will be calculated on-line. In the airport ground movement application, the above mentioned implementations would be infeasible, since if a more detailed aircraft movement model is employed, executing such on-line calculation even once would make the decision support system untractable for real-time decision making. If only some dummy routes (the k-shortest routes) are examined to initialise the database with the segments included in the dummy routes, the database will be incomplete even if k takes a big value. Additionally, the chance of encountering new segments which are not stored in the database is still high. This is due to the nature of the airport ground movement, as aircraft may take longer routes due to the shortest routes being occupied by other aircraft, especially during higher traffic periods. For example, in case of Zürich Airport described in Section 4, the number of new segments which are not stored in the database initialized by solving more than 1400 routes increased from 18 for scenario with 100 aircraft considered to 56 for a scenario with 150 aircraft. With even more traffic, this situation will deteriorate significantly. Although the heuristic used in this paper to evaluate these newly encountered segments is computationally realtively fast, it is only a simplified solution method as discussed in Section 2.3. In the case of more complex aircraft model and nonlinear speed profiles, more general optimisation methods, such as PAIA, have to be employed, which in the simplest case (simplified aircraft model and piece-wise linear speed profiles) may take several minutes on average to converge. Therefore, evaluation of even a single segment on-line could compromise the on-line capability of the decision support system. The attempt to build up a complete database by enumerating of all possible routes between the source and destination nodes to discover all segments has proved intractable. For a graph with n nodes, in theory, there can be up to $n!$ routes.

Another application of storing previously computed results to speed up the on-line search, is the pre-processing of the input graph for finding the shortest path routes (Sanders and Schultes, 2007). Shortest path methods utilising pre-computation include highway hierarchies (Sanders and Schultes, 2005), contraction hierarchies (Geisberger et al., 2008) or landmarks (Goldberg and Harrelson, 2005). In these methods, nodes are selected based on certain criteria (e.g. in case of highway hierarchies higher level nodes typically lying on a highway) for which a shortcut is pre-computed and stored as illustrated in Fig. 5. Storing the results of those pre-computations then accelerates on-line queries when searching for the shortest path route. Shortcuts for the multi-objective shortest path problem were generated in (Delling and Wagner, 2009). However, it is worth pointing out that in case of multiple objectives, how to work out a reasonable number of shortcuts and guarantee that all shortcuts are explored at the same time remain unclear.

In order to address the abovementioned issues, we propose a new approach in this paper, which is guaranteed to build up a complete database. Furthermore, the creation of such a complete database does not require ther search of all possible routes, and can be solved in a tractable time frame. The idea is to identify a complete set of building blocks. Then, speed profile optimisation is run for this limited number of building blocks. These building blocks together

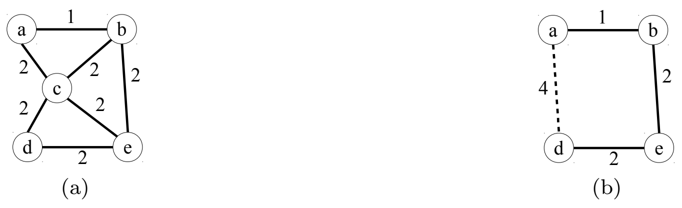


Figure 5: Contraction hierarchies preprocess the original graph (a) by removing certain nodes and introducing pre-computed shortcuts (dashed line). The final graph (b) is more sparse and therefore easier to search. (Adapted from (Sturtevant and Geisberger, 2010).)

with their corresponding optimal speed profiles are memoized into the database, from which any route on the airport can be recreated. The proposed method will be very appealing to the air transportation industry since the database is complete. This will facilitate the promotion of trajectory-based airport ground operations (SESAR, 2012; Joint Planning and Development Office, 2010; Cheng and Sweriduk, 2009; Schaper and Gerdes, 2013) as costly on-line speed optimisation can be completely avoided. In the following, details of the AR approach and how the pre-computed database is generated and used in the AR framework to enhance its on-line decision capability are described.

3. Pre-computed Database Approach for Real Time Active Routing

3.1. Real-time Active Routing procedure

This section provides a detailed description of the proposed approach for real-time AR based on a pre-computed database. As illustrated in Fig. 6, the original concept of AR is extended to incorporate a database consisting of speed profiles.

In effect, this database functions as a middleware to effectively separate the planning module and the speed profile generation module. The database is filled off-line *a-priori*, with optimised speed profiles for a complete set of building blocks as described later in Section 3.3. During on-line decision making, speed profiles for building blocks are retrieved from the database, combined into a complete route and transferred to the planning module. This way, many duplicative search processes can be prevented. Furthermore, the fact that the database is initialized off-line rather than on-line enables the implementation of complex and computationally expensive models, with increased number of objectives, and/or adoption of more realistic speed profile representation without increasing the computational time during the on-line decision making process.

Algorithm 2 details the integrated procedure for the AR approach, while more discussion about database formation can be found in the following section. Similarly as the approach introduced in (Ravizza et al., 2013b), Algorithm 2 approximates the global Pareto front by generating l points on the Pareto front. In each iteration (lines 4–13) the whole set of aircraft is scheduled using the k-QPPTW algorithm and one point of the Pareto front is generated. Note, that

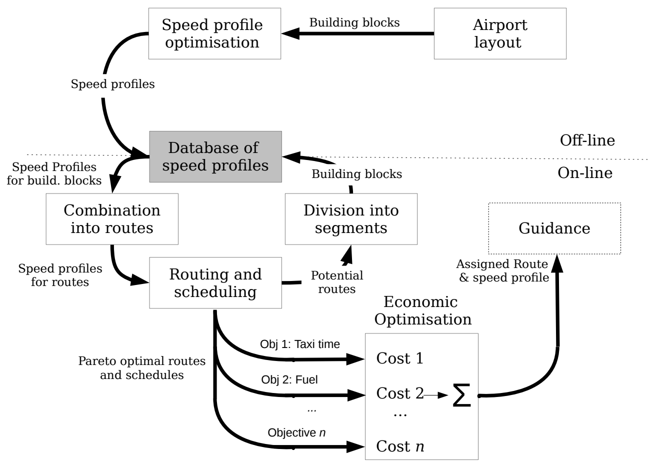


Figure 6: A real time Active Routing approach based on an integrated procedure using a pre-computed database.

all aircraft in one iteration follow speed profiles with the same ratio between objectives g_1, g_2, generated for the k-shortest routes. As the parameter i is incrementally increased (line 3), the algorithm finds points on the Pareto front gradually changing from the most time-efficient solution to the most fuel-efficient solution.

Algorithm 2 Integrated procedure combining routing and scheduling, speed profile optimisation employing a database.

```
 1: Initialize database;
 2: Sort aircraft by their pushback/landing time;
 3: for $i = 1$ to $l$ do
 4:     for all aircraft $a$ do
 5:         Generate the shortest $k$ routes using the $k$-QPPTW algorithm;
 6:         for $k$ of aircraft $a$ do
 7:             Divide route $k$ into segments, for each segment retrieve $m$ speed
                profiles from the database;
 8:             Combine optimal speed profiles for each segment to cover the whole
                route $k$;
 9:         end for
10:         Generate the combined Pareto-front for the source-destination pair of
            aircraft $a$;
11:         Discretise this Pareto front into $l$ roughly equally spaced solutions;
12:         Select the $i$-th solution and reserve the relevant route for aircraft $a$;
13:     end for
14:     Save the accumulated values for all aircraft for both objective functions
        for the global Pareto front;
15: end for
```

The aircraft are considered sequentially according to their pushback/landing time (line 2). Previously assigned routes and speed profiles do not change whenever a new aircraft is taken into consideration. For each aircraft a, the k-best routes are generated based on their taxi times assuming constant speed $v_{straight}$ and v_{turn} for straight and turning edges, respectively (line 5). Individual edges of each route are assigned time windows which represent periods when the edge is not occupied by any other aircraft. Then, for each aircraft a the following loop is executed in lines 6–9. Each route k is divided into segments as explained in Section 2.3. For each segment, m speed profiles complying with time window constraints for aircraft a are retrieved from the database consisting of pre-configured speed profiles in line 7.

The optimal speed profiles for individual segments are combined together, i.e. the most time efficient speed profiles are selected for all segments, gradually changing to the most fuel efficient speed profiles, to form speed profiles for the whole route k in line 8.

The subroutine in line 10 combines the different Pareto fronts for k routes, each consisting of m solutions, and by selecting non-dominated solutions it produces the global Pareto front for the given source-destination pair of aircraft r. The resulting Pareto front is discretised into l roughly equally spaced solutions in line 11. The discretisation is based on Euclidean distance between solutions in the objective space. Eq. 7 defines the distance s between solutions x, y for each q of h objectives:

$$s(x,y) = \sqrt{\sum_{q=1}^{h}(x_q - y_q)^2} \tag{7}$$

The distance s is computed for every neighbouring pair of points $P(p), P(p+1)$ on the Pareto front P, with $|P|$ representing the total number of solutions on the Pareto front. In order to have more representative discretised solutions, only those solutions with a distance D to their neighbouring solutions are chosen, where D is defined in Eq. 8.

$$D = \frac{\sum_{p=1}^{|P|-1} s(P(p+1), P(p))}{l} \tag{8}$$

Fig. 7 illustrates this procedure and finally the i-th discretised solution is chosen in line 12. The route, together with the corresponding speed profile, is used to schedule aircraft a. Note, that although in one iteration the value of i is the same for all aircraft, each aircraft a has a different speed profile depending on its route and underlying segments.

The inner loop (lines 4–13) is repeated until all aircraft from the dataset have been routed. The total taxi time and the total fuel consumption is accumulated to generate a single solution on the global Pareto front (line 14). As can be seen from Algorithm 2, the most time consuming part, i.e. speed profile generation is now replaced by a pre-configured database. The database formation and speed profile retrieval executed in Algorithm 2, line 7 is described in the next section.

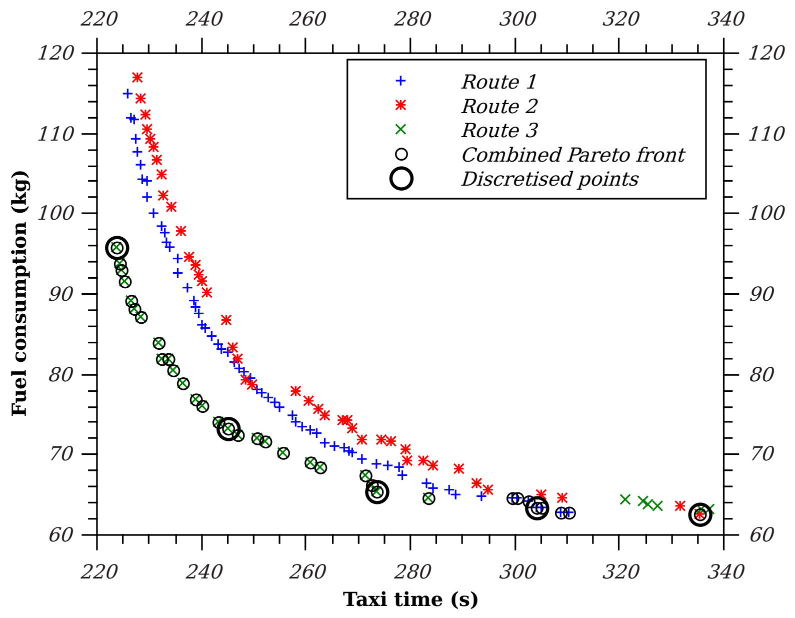


Figure 7: Combined Pareto front from 3 different routes for one aircraft and $l = 5$ discretised points.

3.2. Database formation and speed profile retrieval

As can be seen in Line 7, Algorithm 2, m Pareto-optimal solutions are required for the speed profile optimisation problem for a given route of aircraft a. The decision variables a_1, d_1, d_2, d_4 determining the speed profiles are retrieved from the database. The data within the database is indexed in a B-tree structure (Comer, 1979). In this tree-like structure, data is sorted into hierarchical levels using the following keys: weight class, segment type, segment length, discretisation, as illustrated in Fig. 8. This structure facilitates an efficient search, as only the relevant branch has to be traversed to read the data.

Firstly, as each aircraft has a unique fuel consumption which results in different optimal speed profiles, in order to keep the problem tractable, aircraft have been classified into 3 *weight_class* groups: light, medium and heavy, according to their wake vortex separation requirements (ICAO, 2007). Next, the segments are characterised by their *type* (refer to Table 1) and *length*. For aircraft traversing turning segments, a constant speed is assumed and their taxi time and fuel consumption (objectives g_1, g_2) are always constant. As a result, the turning segments are not included in the database. Other segments (straight, straight holding and straight breakaway) are included in the database and characterised by their length. In order to keep the database size tractable, the length of segments is rounded to the nearest metre thus limiting the number of possible

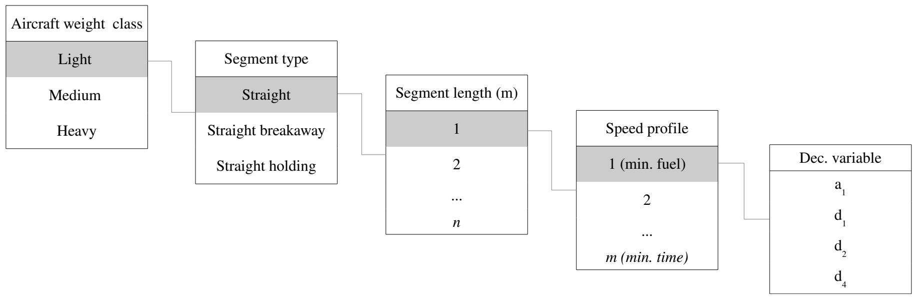


Figure 8: Structure of the database.

entries in the database. A single database entry contains m discretised solutions from the Pareto front for the given segment ranging from the most fuel efficient solution to the most time efficient one. Finally, each Pareto optimal solution is described by a set of corresponding decision variables a_1, d_1, d_2, d_4.

The complete solution for the whole route is constructed as a set of decision variables a_1, d_1, d_2, d_4 for each segment of the whole route. The solution is checked for potential violations of time windows of already routed aircraft. The infeasibility of solutions is discussed in more detail in Section 3.4. Once the complete database is constructed according to the abovementioned structure, it will be used any time when optimal speed profiles for a given route are queried by Algorithm 2. Details about database initialization are discussed in the next section.

3.3. Database Initialization

In the light of discussion in Section 2.4, a database initialization method is proposed to identify all distinctive building blocks from which any route on the airport can be recreated. Firstly, preprocessing of the graph G representing the taxiways is performed and segments between certain nodes are generated. Then, optimal speed profiles are found for these segments, and memoized in the database for retrieval during on-line search.

To initialize the database, Algorithm 3 is executed on the graph G. As described in Section 2.3, any of two adjacent edges e and $e + 1$ with an angle greater or equal to 30 degrees can become a turning segment depending on the direction in which edges are traversed. These "knee edges" are identified and together with gate/runway edges (edges with a node being a gate or runway exit) are put into set K in line 1. Edges in K are the potential ends of the segments, where a straight segment can change to a turning segment, or in case of gate/runway edges terminate. Then, shortcuts, i.e. possible straight segments are generated in line 2 − 17. It should be noted, that although shortcuts are created in the proposed approach, the intermediate nodes are not necessarily removed, in contrast to other graph-preprocessing techniques such as contraction hierarchies (Geisberger et al., 2008). This will depend on whether the nodes belong to knee edges, whereas in the contraction hierarchy, as shown in Fig. 5,

Algorithm 3 Initialization procedure for building a complete database.

```
1: $K = \{e, e+1 \colon e \in E \wedge angle(e, e+1) \geq 30\} \cup \{e \colon e \in E \wedge e \text{ is gate/runway edge } \}$;
2: for all $e \in K$ do
3:   Find all straight routes $r$ from $e$ to $f \in K$;
4:   for all routes $r$ do
5:     if $e$ is a knee edge then
6:       $length = sum(e+1, e+2, \ldots, f)$;
7:     else
8:       $length = sum(e, e+1, \ldots, f)$
9:     end if
10:    if segment with $length$ not in $database(weight\_class, type, length)$ then
11:      for all $weight\_class$ do
12:        search for variables $a_1, d_1, d_2, d_4$ and objectives $g_1$, $g_2$ using the speed profile generation algorithm;
13:        save $g_1$, $g_2$ and variables $a_1, d_1, d_2, d_4$ to $database(weight\_class, type, length)$;
14:      end for
15:    end if
16:  end for
17: end for
```

the contracted node will be inevitably removed regardless of being a part of a knee edge. As this would result in an incomplete set of building blocks, in our approach the shortcuts are generated as follows: Starting with the edge e, a depth-first search finds all straight routes r from e to f, i.e. successive edges which have an angle less than 30 degrees between them until another edge $f \in K$ is reached in line 3. It is worth mentioning that multiple edges $f \in K$ may be reached from e, and multiple straight routes r between e and f may exist, and all are considered in lines 4 – 16. The length of a possible straight segment is a sum of lengths of edges $e+1$ to f if e is a knee edge (line 6). If e is a gate/runway edge, then the length of a possible straight segment is a sum of lengths of edges e to f (line 8). If the segment with this length is not in the database, it is solved by the speed profile generation algorithm for all weight class categories in line 12 and saved into the database. Iterating through all edges $e \in K$ in lines 2 – 17 therefore guarantees that all possible segments are generated.

Fig. 9 illustrates this approach. For an aircraft taxiing from node 1 to 9 in Fig. 9a, edge e constitutes a turning segment. Edges $e+1$ to f then form a straight segment. In the next iteration, illustrated in Fig. 9b, we suppose an aircraft taxiing in the opposite direction, from node 9 to 1, and a similar situation is solved. Therefore, all possible divisions of edges between nodes 2–8 into straight and turning segments are considered. In the case of the gate/runway edge shown in Fig. 9c, edge e will be included in the straight segment.

This method is exhaustive in its nature, however the requirement of a straight

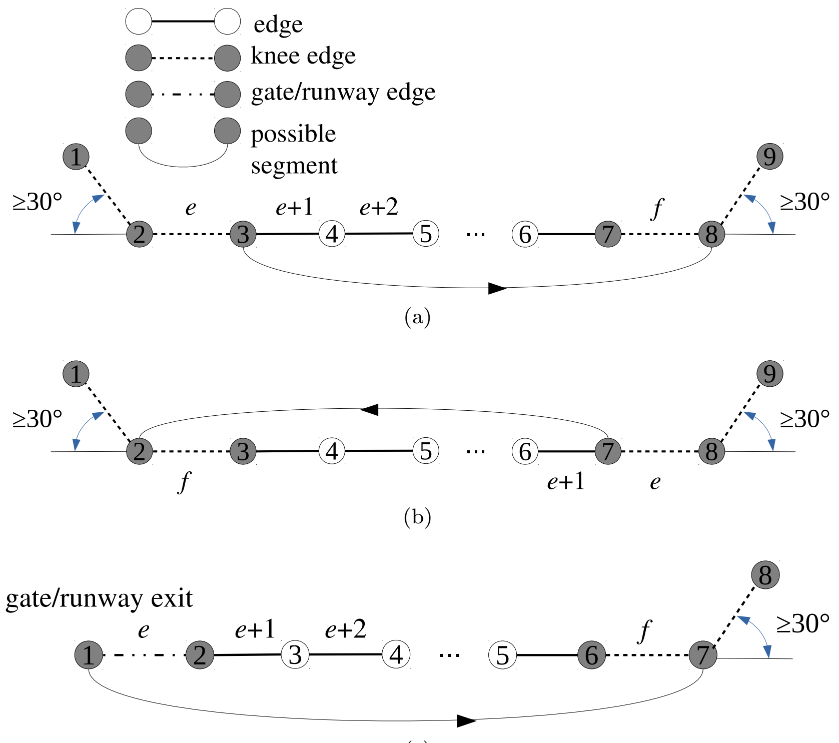


Figure 9: Calculated distances between edges represent possible straight segments; arrow on the segment indicates the direction in which the segment will be traversed by aircraft.

segment between edges $e, f \in K$ provides an effective pruning condition and for reasonable sized graphs the search is fast, as later demonstrated in Section 4.3.

3.4. Resolution of Infeasibility

The database contains only Pareto optimal solutions for unimpeded aircraft which does not consider any time window constraints. Therefore, during the routing and scheduling, some or even all solutions retrieved from the database may be infeasible due to current time window violations. If only some solutions are infeasible, the remaining feasible solutions are used. However, if all solutions retrieved from the database are discarded, the only feasible solutions may be the previously suboptimal solutions which are not in the database.

Fig. 10 shows an example of Pareto front for an aircraft which is subject to a time constraint for a specific turning segment as indicated in Fig. 11. The solutions initially retrieved from the database are all infeasible due to time window violation. In order to generate feasible solutions: 1) an extra holding (buffer) time can be added to departure time of the aircraft to "shift" the solutions out of the infeasible region as shown in Fig. 10, or 2) a speed profile optimisation which actively takes constraints into account (e.g. PAIA (Chen

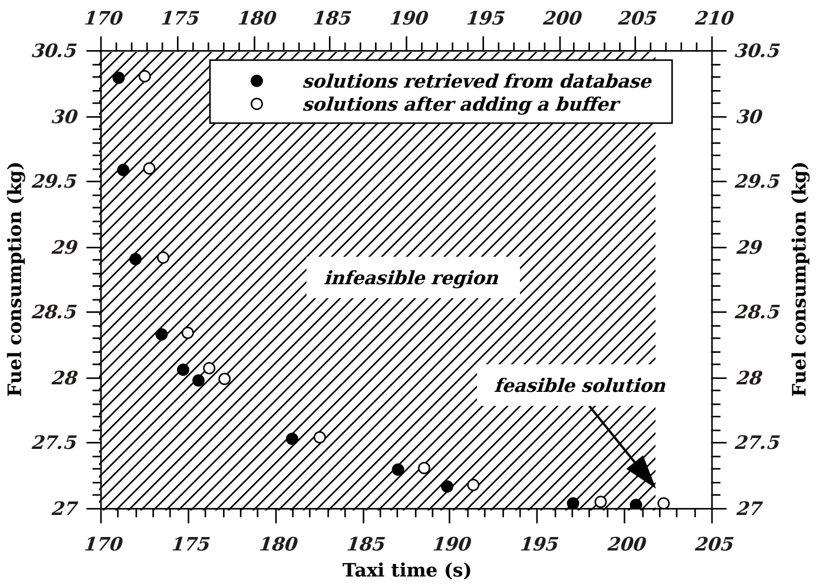


Figure 10: Resolution to infeasibility of solutions is achieved by adding a time buffer to shift the solutions out of the infeasible region.

et al., 2015b)) can be performed as shown in Fig. 11b. In our algorithm, the first method is applied and the smallest possible holding time i.e. time by which the time window is violated is added to obtain a feasible solution in Algorithm 2, line 7.

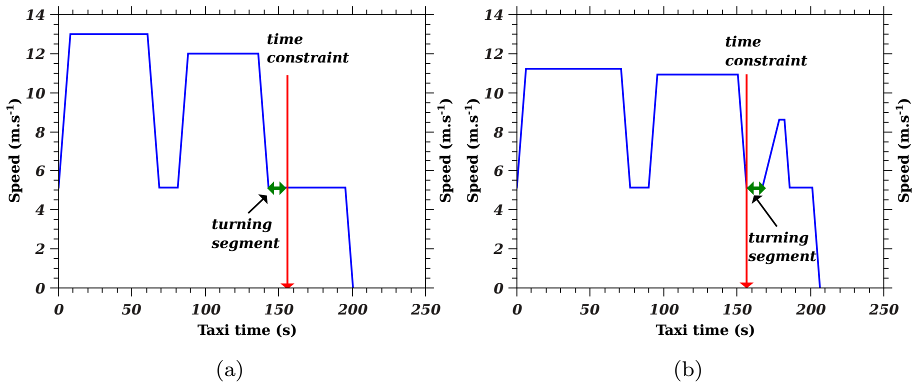


Figure 11: Speed profile of aircraft 446 with time constraint. The aircraft can enter the turning segment only after time constraint indicated by an arrow: (a) Speed profile retrieved from database violating the time constraint; (b) Feasible speed profile generated by PAIA.

4. Experimental results and discussions

4.1. Dataset description

The algorithm was tested on a dataset of real arrival and departure flights on Zürich Airport (ZRH) which is the largest airport in Switzerland. The airport has 3 runways, 8 runway exits used in daily operation and there are 88 gates or stands in total. The layout of ZRH is depicted in Fig. 12.

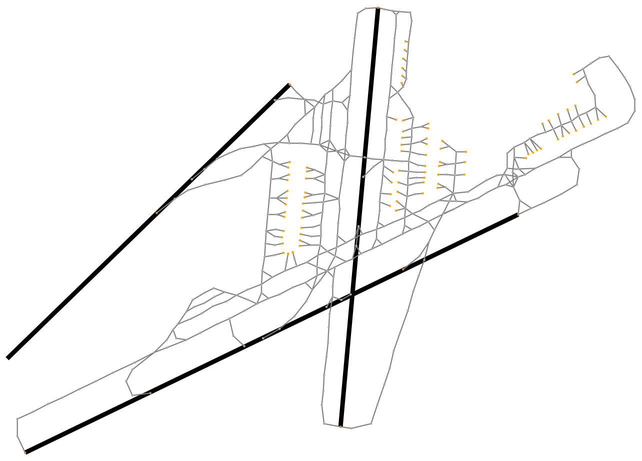


Figure 12: Layout of ZRH with taxiways.

Table 2 summarises data including flights recorded on 14th October 2014, which have been divided into 3 instances representing high, medium and low traffic conditions. Each instance consists of flights departing or landing in the corresponding hour (i.e. for ZRH 6, between 6:00 and 7:00, for ZRH 8, between 8:00 and 9:00, etc.). The dataset was constructed from data published on the airport website including arrival and departure times and gates whereas runway exits for each flight were determined by Flightradar24 AB (2014). Data with anonymised flights used in this work are published with this paper. For interested readers, we recommend (Brownlee et al., 2014) for more information about generating and processing freely-available datasets, including airport layout, for the ground movement problem. In order to study the scale-up performance of the proposed algorithm when it is facing more traffic and under pressure, instances with randomly generated flights named T75, T100, T125, T150, indicating the number of flights were created. Each such instance consists of flights departing or landing within one hour with randomly assigned times and gates/runways. As the focus of this paper is on the ground movement problem, it has to be noted that neither runway capacity nor overlaps of gate allocations were considered.

Table 2: Summary of data instances utilised in computational experiments.

	Instance						
	ZRH 6	ZRH 8	ZRH 10	T75	T100	T125	T150
Number of aircraft	26	48	37	75	100	125	150
Arrivals	12	33	20	43	46	64	74
Departures	14	15	17	32	54	61	76

As discussed in Section 3.2, aircraft have been divided into 3 categories according to their wake vortex separation requirements. For each category, a representative aircraft is designated and its specifications are used during the calculation. The specifications are summarized in Table 3. The real instances ZRH 6–ZRH 10 used in this paper consist of mostly medium category aircraft and a few heavy aircraft used in long-haul flights, whereas for instances T75–T150 the distribution of weight categories is random.

Table 3: Specifications of the representative aircraft.

	Learjet 35A	Airbus A320	Airbus A333
Take-off weight	8300 kg	78000 kg	230000 kg
Engines	TFE731-2-2B	CMF56-5-A1	CF6-80E1A2
Number of engines	2	2	2
Rated output F_o	2×15.6 kN	2×111.2 kN	2×287 kN
Rolling resistance	1221 N	11.48 kN	33.84 kN
Fuel flow at 7% F_o	0.024 kg·s^{-1}	0.101 kg·s^{-1}	0.228 kg·s^{-1}
Fuel flow at 30% F_o	0.067 kg·s^{-1}	0.291 kg·s^{-1}	0.724 kg·s^{-1}

4.2. Experimental settings

The routing and scheduling part of the algorithm has been programmed in Java and the speed optimisation part is written in the MATLAB programming language. The computational experiments have been performed on a computer with an Intel i3-2120 processor and 8 GiB of RAM, running Windows 7.

A setting $l = 5$ (line 3 in Algorithm 2) has been used to generate five roughly equally spaced solutions. The parameter defining the number of routes generated by k-QPPTW (line 5 in Algorithm 2) is set to $k = 3$. The values of parameters l, k have been chosen according to initial experiments which showed a good ability to approximate the global Pareto front and are the same as in work by Ravizza et al. (2013b).

4.3. Database initialization

The complete database is created by enumerating all possible straight segments as explained in Section 3.3. The graph for ZRH airport consists of 399 knee nodes and the enumeration lasted around 4 seconds. The results in Table 4 show that despite the relatively large number of nodes the number of found segments (building blocks) is reasonable.

Table 4: Summary of the complete database initialization process.

Segment	Database
Straight	734
Straight breakaway	79
Straight holding	54
$\sum$	867

The complete database was initialized by directly solving 867 segments, without any overhead from routing, which lasted 673 seconds. The size of the complete database is around 1 MiB for 3 weight class categories. Additional categories or aircraft types would proportionally increase the size of the database.

4.4. Computational results

After the database has been initialized as described in Section 3.3, Algorithm 2 has been run. During the experiments, as expected, many aircraft routes overlapped as illustrated in Fig. 13. Table 5 compares the computation of one solution on the global Pareto front without and with the database. In the first case, the number of segments which have to undergo speed profile optimisation is much higher than the number of segments retrieved from the database in the second case. Furthermore, retrieved segments from the database were pre-computed offline.

In order to show the effect of the proposed approach on computational time, the algorithm has been tested with and without the database in the speed optimisation. In cases without the database, the heuristic for speed profile optimisation and PAIA has been used as a solution method in line 7 of Algorithm 2 with the same settings as in (Ravizza et al., 2013b). The running time of algorithms to generate global Pareto front for each instance (for example see Fig. 15) is given in Fig. 14. As can be seen from the results, employing the database of pre-computed solutions reduced the computational time compared to other approaches including the simplified heuristic.

In real-time application of airport management systems, it is likely that aircraft will be considered sequentially, therefore the computational time per single aircraft is an important factor. ICAO specifies that the on-line decision time within the Advanced Surface Movement Guidance and Control Systems should not exceed 10 seconds (ICAO, 2004). As shown in Table 6, the computational

Table 5: The number of segments under investigation during computation of one solution on the global Pareto front without and with the database.

	Computed without DB						
Segments	ZRH 6	ZRH 8	ZRH 10	T75	T100	T125	T150
Straight	312	651	500	875	1123	1345	1757
Straight breakaway	42	48	51	96	162	183	228
Straight holding	78	144	111	225	300	375	450
$\sum$	432	843	662	1196	1585	1903	2435
	Retrieved from complete DB						
	ZRH 6	ZRH 8	ZRH 10	T75	T100	T125	T150
Straight	48	51	60	148	168	178	201
Straight breakaway	9	8	12	17	17	22	20
Straight holding	12	16	14	29	33	34	38
$\sum$	69	65	86	194	218	234	259

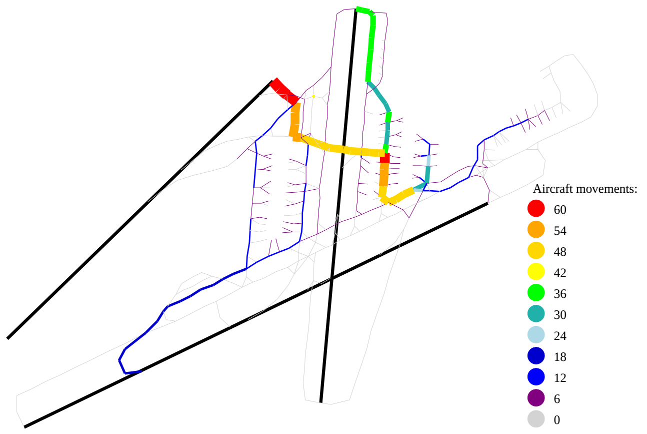


Figure 13: The number of aircraft movements on individual edges for combined flights from instances ZRH 6, 8, 10. The high number of aircraft movements on edges enables the o take advantageous use of the pre-computed database of speed profiles to circumvent repetitive search.

time per single aircraft increased with more traffic, particularly for instances T75–T150. The increase in computational time is linked to the number of infeasibilities, as reported later in this section, due to the fact that the algorithm is restarted every time no feasible solutions exist in line 7 of Algorithm 2. However, the results indicate that the proposed approach is applicable in an on-line decision support system even under high traffic conditions represented by instances T75–T150.

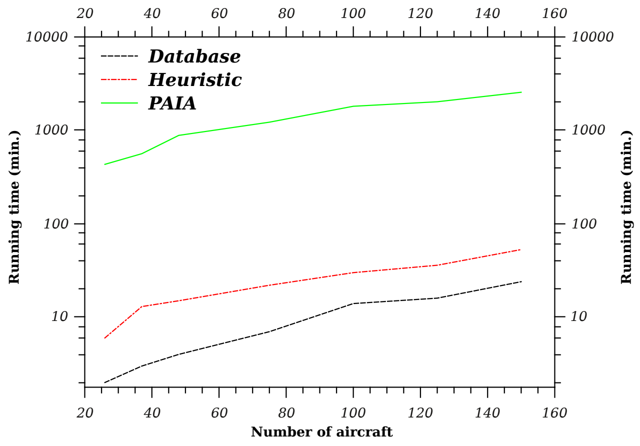


Figure 14: Running time of algorithms (min.) as a function of number of aircraft.

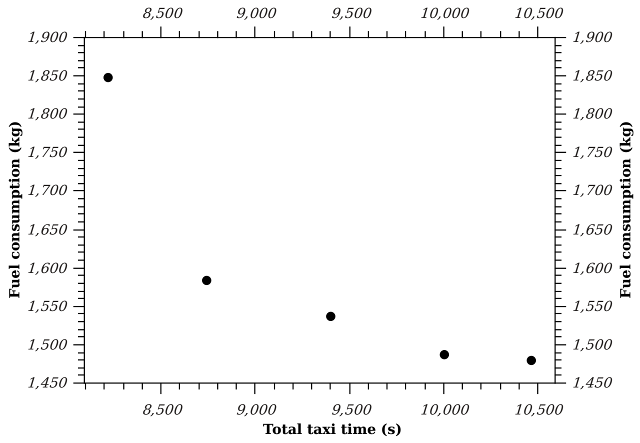


Figure 15: The global Pareto front for ZRH 8 instance.

Table 6: Computational time per single aircraft using the proposed approach (s).

	Instance						
	ZRH 6	ZRH 8	ZRH 10	T75	T100	T125	T150
Time per aircraft (s)	1.03	0.88	1.03	1.09	1.63	1.56	1.95

As mentioned in Section 3.4, in some cases all solution on the Pareto front can be infeasible due to the time window constraints. The results in Table 7 suggest that infeasibilities occurred only during higher traffic in instances and on average from around 20 seconds to 45 seconds of buffer time per infeasible taxi schedule was needed to be added to the original departure time. The total buffer time increased notably with test instances with high traffic levels and follows similar observations in other studies (Ravizza et al., 2013a).

Table 7: Number of infeasibilities which are resolved by adding buffer time.

	Instance						
	ZRH 6	ZRH 8	ZRH 10	T75	T100	T125	T150
Occurrences	0	8	14	66	139	148	290
Total buffer time (s)	0	165	569	2227	4737	5921	13181

5. Conclusions

This paper focuses on the real-time application of AR for the ground movement problem to simultaneously minimise taxi time and fuel consumption of taxiing aircraft. A new procedure based on a database consisting of efficient speed profiles was introduced. The database contains previously computed speed profiles for individual segments of the taxiway which circumvents the repetitive optimisation of these profiles, resulting in a faster algorithm. Moreover, this approach effectively separates the speed profile optimisation from the routing and scheduling phase of the algorithm. As a result, the pre-computed database can incorporate more realistic speed profiles created through a complex and more precise optimisation procedure without compromising computational time during the real-time application of the algorithm. The computational experiments conducted on real-world data from a major European hub indicate that the proposed approach can be fully applicable in real-time airport management systems.

Future research in this area could exploit the advantage of the proposed approach to replace the speed optimisation procedure by a more detailed model, thus removing the simplifications of the discretised speed profile or constant speed during turns. Indeed, this is an essential requirement, as speed profiles have to consider characteristics of the jet engine such as time to spool up/down and passenger comfort related to rate of change of acceleration known as jerk. The introduction of the database in this paper makes such consideration feasible in future research. Also, more environmentally related objectives can be added e.g. considering exhaust emissions or engine noise. The current implementation of integrated procedure combining routing, scheduling and speed profile optimisation does not enable the investigation of different combination of speed profiles, e.g. one aircraft follows the most time efficient speed profile while another aircraft uses the most fuel efficient one. Such combination could produce a more detailed global Pareto front or solutions with less buffer time. Finally, in the current implementation, the routes and speed profiles of already scheduled aircraft does not change when a new aircraft enters, which under unprecedented events may compromise the throughput of an airport. Therefore, an approach able to react dynamically may be needed and requires further investigation.

Acknowledgements

This work is supported in part by the Engineering and Physical Sciences Research Council (EPSRC) under Grant EP/H004424/1.

References

Atkin, J. A., Burke, E. K., Greenwood, J. S., 2010a. TSAT allocation at London Heathrow: the relationship between slot compliance, throughput and equity. Public Transport 2 (3), 173–198.

Atkin, J. A., Burke, E. K., Greenwood, J. S., 2011. A comparison of two methods for reducing take-off delay at London Heathrow airport. Journal of Scheduling 14 (5), 409–421.

Atkin, J. A., Burke, E. K., Ravizza, S., 2010b. The airport ground movement problem: Past and current research and future directions. Proceedings of the 4th International Conference on Research in Air Transportation (ICRAT), Budapest, Hungary, 131–138.

Balakrishnan, H., Jung, Y., 2007. A framework for coordinated surface operations planning at Dallas-Fort Worth International Airport. In: Proceedings of the AIAA Guidance, Navigation, and Control Conference, Hilton Head, SC, USA.

Bonyadi, M., Michalewicz, Z., Barone, L., 2013. The travelling thief problem: The first step in the transition from theoretical problems to realistic problems. In: Evolutionary Computation (CEC), 2013 IEEE Congress on. pp. 1037–1044.

Brownlee, A. E., Atkin, J. A., Woodward, J. R., Benlic, U., Burke, E. K., 2014. Airport Ground Movement: Real World Data Sets and Approaches to Handling Uncertainty. In: Proceedings of the 10th International Conference on Practice and Theory of Automated Timetabling (PATAT 2014). pp. 462–464.

Burgain, P., Feron, E., Clarke, J., 2009. Collaborative virtual queue: Benefit analysis of a collaborative decision making concept applied to congested airport departure operations. Air Traffic Control Quarterly 17 (2), 195–222.

Chen, J., Mahfouf, M., 2006. A Population Adaptive Based Immune Algorithm for Solving Multi-objective Optimization Problems. In: Bersini, H., Carneiro, J. (Eds.), Artificial Immune Systems. Vol. 4163 of Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 280–293.
URL `http://dx.doi.org/10.1007/11823940_22`

Chen, J., Stewart, P., 2011. Planning aircraft taxiing trajectories via a multi-ojective immune optimisation. In: Natural Computation (ICNC), 2011 Seventh International Conference on. Vol. 4. pp. 2235–2240.

Chen, J., Weiszer, M., Locatelli, G., Ravizza, S., Atkin, J. A., Stewart, P., Burke, E., 2015a. Towards a More Realistic, Cost Effective and Greener Ground Movement through Active Routing: Part 2-A Multi-Objective Shortest Path Approach.

Chen, J., Weiszer, M., Stewart, P., Shabani, M., 2015b. Towards a More Realistic, Cost Effective and Greener Ground Movement through Active Routing: Part 1-Optimal Speed Profile Generation.

Cheng, V. H. L., Sweriduk, G., Aug 2009. Trajectory design for aircraft taxi automation to benefit trajectory-based operations. In: Asian Control Conference, 2009. ASCC 2009. 7th. pp. 99–104.

Clare, G., Richards, A., Dec 2011. Optimization of Taxiway Routing and Runway Scheduling. Intelligent Transportation Systems, IEEE Transactions on 12 (4), 1000–1013.

Comer, D., Jun. 1979. Ubiquitous B-Tree. ACM Comput. Surv. 11 (2), 121–137. URL http://doi.acm.org/10.1145/356770.356776

Deau, R., Gotteland, J., Durand, N., 2009. Airport surface management and runways scheduling. In: Proceedings of the 8th USA/Europe Air Traffic Management Research and Development Seminar, Napa, CA, USA.

Delling, D., Wagner, D., 2009. Pareto paths with SHARC. In: Experimental Algorithms. Springer, pp. 125–136.

EUROCONTROL, 2013. Challenges of Growth 2013: The Effect of Air Traffic Network Congestion in 2035.
URL http://www.eurocontrol.int/sites/default/files/content/documents/official-documents/reports/201306-challenges-of-growth-2013-summary-report.pdf

European Commission, 2011. Roadmap to a Single European Transport Area – Towards a competitive and resource efficient transport system. White paper.

Flightradar24 AB, 2014. flightradar24. http://www.flightradar24.com, accessed: 2014-10-14.

Forrester, A. I., Keane, A. J., 2009. Recent advances in surrogate-based optimization. Progress in Aerospace Sciences 45 (1), 50–79.

García, J., Berlanga, A., Molina, J., Casar, J., 2005. Optimization of airport ground operations integrating genetic and dynamic flow management algorithms. AI Communications 18 (2), 143–164.

Geisberger, R., Sanders, P., Schultes, D., Delling, D., 2008. Contraction hierarchies: Faster and simpler hierarchical routing in road networks. In: Experimental Algorithms. Springer, pp. 319–333.

Goldberg, A. V., Harrelson, C., 2005. Computing the shortest path: A search meets graph theory. In: Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms. Society for Industrial and Applied Mathematics, pp. 156–165.

Gotteland, J., Durand, N., Alliot, J., 2003. Handling CFMU slots in busy airports. In: Proceedings of the 5th USA/Europe Air Traffic Management Research and Development Seminar, Budapest, Hungary.

Guan, X.-M., Zhang, X.-J., Zhu, Y., Hwang, I., Sun, D.-F., 2013. An Efficient Routing Strategy On Spatial Scale-Free Networks. International Journal of Modern Physics C.

ICAO, 2004. Advanced Surface Movement Guidance and Control Systems (A-SMGCS) Manual.
URL `http://www.icao.int/Meetings/anconf12/Document`

ICAO, 2007. Air Traffic Management: Procedures for Air Navigation Services. 15th edition. Doc 4444 ATM/501.

Joint Planning and Development Office, Sep. 2010. Concept of Operations for the Next Generation Air Transportation System. Version 3.2.
URL `http://jpe.jpdo.gov/ee/docs/conops/NextGen_ConOps;_v3_2.pdf`

Khalvati, F., Kianpour, M., Tizhoosh, H., 2011. Cascaded Window Memoization for Medical Imaging. In: Iliadis, L., Maglogiannis, I., Papadopoulos, H. (Eds.), Artificial Intelligence Applications and Innovations. Vol. 364 of IFIP Advances in Information and Communication Technology. Springer Berlin Heidelberg, pp. 275–284.
URL `http://dx.doi.org/10.1007/978-3-642-23960-1_33`

Lesire, C., 2010. An Iterative A* Algorithm for Planning of Airport Ground Movements. 19th European Conference on Artificial Intelligence (ECAI)/6th Conference on Prestigious Applications of Intelligent Systems (PAIS), Lisbon, Portugal, August 16-20, 2010.

Lewis, A., Randall, M., Galehdar, A., Thiel, D., Weis, G., 2009. Using Ant Colony Optimisation to Construct Meander-Line RFID Antennas. In: Lewis, A., Mostaghim, S., Randall, M. (Eds.), Biologically-Inspired Optimisation Methods. Vol. 210 of Studies in Computational Intelligence. Springer Berlin Heidelberg, pp. 189–217.
URL `http://dx.doi.org/10.1007/978-3-642-01262-4_8`

Li, X., Lo, H. K., 2014. An energy-efficient scheduling and speed control approach for metro rail operations. Transportation Research Part B:

Methodological 64 (0), 73–89.
URL http://www.sciencedirect.com/science/article/pii/S0191261514000484

Mandow, L., De La Cruz, J. L. P., 2010. Multiobjective A* search with consistent heuristics. Journal of the ACM (JACM) 57 (5), 27.

Marín, A., 2006. Airport management: taxi planning. Annals of Operations Research 143 (1), 191–202.

Marín, A., Codina, E., 2008. Network design: taxi planning. Annals of Operations Research 157 (1), 135–151.

Mensing, F., Bideaux, E., Trigui, R., Ribet, J., Jeanneret, B., 2014. Eco-driving: An economic or ecologic driving style? Transportation Research Part C: Emerging Technologies 38 (0), 110–121.
URL http://www.sciencedirect.com/science/article/pii/S0968090X13002374

Mensing, F., Trigui, R., Bideaux, E., 2011. Vehicle trajectory optimization for application in ECO-driving. In: Vehicle Power and Propulsion Conference (VPPC), 2011 IEEE. IEEE, pp. 1–6.

Michie, D., 1968. Memo functions and machine learning. Nature 218 (5136), 19–22.

Pesic, B., Durand, N., Alliot, J., 2001. Aircraft ground traffic optimisation using a genetic algorithm. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO), San Francisco, California, USA. pp. 1397–1404.

Ravizza, S., Atkin, J. A., Burke, E. K., 2013a. A more realistic approach for airport ground movement optimisation with stand holding. Journal of Scheduling, 1–14.

Ravizza, S., Chen, J., Atkin, J. A., Burke, E. K., Stewart, P., 2013b. The trade-off between taxi time and fuel consumption in airport ground movement. Public Transport 5 (1-2), 25–40.

Roling, P. C., Visser, H. G., 2008. Optimal Airport Surface Traffic Planning Using Mixed-Integer Linear Programming. International Journal of Aerospace Engineering vol. 2008, 11.

Ruiz, S., Piera, M. A., Pozo, I. D., 2013. A Medium Term Conflict Detection and Resolution system for Terminal Maneuvering Area based on Spatial Data Structures and 4D Trajectories. Transportation Research Part C: Emerging Technologies 26 (0), 396–417.
URL http://www.sciencedirect.com/science/article/pii/S0968090X12001271

Sanders, P., Schultes, D., 2005. Highway hierarchies hasten exact shortest path queries. In: Algorithms–Esa 2005. Springer, pp. 568–579.

Sanders, P., Schultes, D., 2007. Engineering fast route planning algorithms. In: Experimental Algorithms. Springer, pp. 23–36.

Schaper, M., Gerdes, I., Oct 2013. Trajectory based ground movements and their coordination with departure management. In: Digital Avionics Systems Conference (DASC), 2013 IEEE/AIAA 32nd. pp. 1–23.

SESAR, Oct. 2012. European ATM Master Plan.
URL https://www.atmmasterplan.eu/download/29

Smeltink, J., Soomer, M., de Waal, P., van der Mei, R., 2004. An optimisation model for airport taxi scheduling. In: Proceedings of the INFORMS Annual Meeting, Denver, Colorado, USA.

Sturtevant, N. R., Geisberger, R., 2010. A Comparison of High-Level Approaches for Speeding Up Pathfinding. In: Proceedings of the Sixth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, AIIDE 2010, October 11-13, 2010, Stanford, California, USA.
URL http://aaai.org/ocs/index.php/AIIDE/AIIDE10/paper/view/2139;http://dblp.uni-trier.de/rec/bib/conf/aiide/SturtevantG10

Wagner, M., Day, J., Neumann, F., 2013. A fast and effective local search algorithm for optimizing the placement of wind turbines. Renewable Energy 51 (0), 64–70.
URL http://www.sciencedirect.com/science/article/pii/S0960148112005745

Weiszer, M., Chen, J., Ravizza, S., Atkin, J., Stewart, P., July 2014. A heuristic approach to greener airport ground movement. pp. 3280–3286.

Xue-Jun, Z., Xiang-Min, G., Deng-Feng, S., Shao-Ting, T., 2013. The Effect of Queueing Strategy on Network Traffic. Communications in Theoretical Physics 60 (4), 496.

Yousefi, A., Zadeh, A. N., 2013. Dynamic allocation and benefit assessment of NextGen flow corridors. Transportation Research Part C: Emerging Technologies 33 (0), 297–310.
URL http://www.sciencedirect.com/science/article/pii/S0968090X12000666

Zúñiga, C., Piera, M., Ruiz, S., Pozo, I. D., 2013. A CD&CR causal model based on path shortening/path stretching techniques. Transportation Research Part C: Emerging Technologies 33 (0), 238–256.
URL http://www.sciencedirect.com/science/article/pii/S0968090X11001628