Transportation Research Part C 119 (2020) 102734

Contents lists available at ScienceDirect

Transportation Research Part C

journal homepage: www.elsevier.com/locate/trc

Multi-objective routing and scheduling for airport ground movement

Michal Weiszer [a], Edmund K. Burke [b], Jun Chen [a,*]

[a] School of Engineering and Materials Science, Queen Mary University of London, Mile End Road, London, United Kingdom
[b] University of Leicester, University Road, Leicester, United Kingdom

ARTICLE INFO

Keywords:
OR in airlines
Multiple objective optimisation
Shortest path
Airport ground movement
Multi-graph

ABSTRACT

Recent research on airport ground movement introduced an Active Routing framework to support multi-objective trajectory-based operations. This results in edges in the airport taxiway graph having multiple costs such as taxi time, fuel consumption and emissions. In such a graph, multiple edges exist between two nodes reflecting different trade-offs among the multiple costs. Aircraft will have to choose the most efficient edge from multiple edges in order to traverse from one node to another respecting various operational constraints. In this paper, we introduce a multi-objective routing and scheduling algorithm based on the enumerative approach that can be used to solve such a multi-objective multi-graph problem. Results using the proposed algorithm for a range of international airports are presented. Compared with other routing and scheduling algorithms, the proposed algorithm can find a representative set of optimal or near optimal solutions in a single run when the sequence of aircraft is fixed. In order to accelerate the search, heuristic functions and a preference-based approach are introduced. We analyse the performance of different approaches and discuss how the structure of the multi-graph affects computational complexity and quality of solutions.

1. Introduction

Airports around the world are increasingly employing complex automated systems such as Advanced Surface Movement, Guidance and Control Systems (A-SMGCS) (ICAO, 2004). This is due to increasing air traffic and hence pressure to achieve higher throughput using existing airport infrastructure. Within this context, a better coordination of aircraft moving on the airport surface plays a key role, as it is considered a critical link between other airport operations (runway scheduling, gate assignment) (Atkin et al., 2010). The adoption of A-SMGCS enables more precise guidance and control for all aircraft on the airport surface via specialised functions such as routing, guidance, control and surveillance. Among the four functions, the automated routing function is the key element for increasing the efficiency of ground movement as it provides optimised aircraft routes and schedules.

Apart from time efficiency, which can improve the utilisation of existing airport infrastructure, environmental and other cost aspects of ground movements are also of a concern in achieving sustainable airport operations. Due to financial pressures and stricter regulations, objectives such as minimising fuel consumption and related emissions are of increasing importance and benefit all stakeholders of airport operations such as airports and airlines. Recent research has seen a trend of moving away from single-objective optimisation of ground movement towards multi-objective models, taking into account objectives such as taxi time, fuel consumption and emissions (Ravizza et al., 2013b; Chen et al., 2016b,a; Evertse and Visser, 2017). The experimental results in (Chen

* Corresponding author.
E-mail addresses: m.weiszer@qmul.ac.uk (M. Weiszer), edmund.burke@leicester.ac.uk (E.K. Burke), jun.chen@qmul.ac.uk (J. Chen).

https://doi.org/10.1016/j.trc.2020.102734
Received 28 October 2019; Received in revised form 29 April 2020; Accepted 23 July 2020
Available online 10 August 2020

et al., 2016b,a) have confirmed the presence of trade-off among different objectives, indicating that usually there is not a single solution that can minimise all the objectives.

Different fuel consumption and emissions are a result of different speed profiles determined by the acceleration/deceleration rate and their durations, which can be used by aircraft to taxi between two nodes, along a taxiway segment. This implies, that the costs for each segment (i.e. taxi time, fuel consumption, emissions) are no longer a vector, but a matrix. Therefore, the routing algorithm has to make two decisions: (1) which segment to use when more segments are possible at an intersection; (2) which speed profile to apply within that segment. The combination of those two decisions over a series of segments defines a potential route. The introduction of this cost matrix is equivalent to a multi-graph where nodes are connected with more than one parallel edges corresponding to different speed profiles. Formulating the ground movement problem as a multi-graph increased complexity and search space for the routing algorithm. Recent research in other fields such as the time-constrained vehicle routing problem (Garaix et al., 2010; Lai et al., 2016) followed the formulation of the airport ground movement problem. Both of them employ a multi-graph structure for finding better routes.

The ground movement problem is a combined routing and scheduling problem (Atkin et al., 2010). The algorithm has to find an efficient route from the start node to the end node for each aircraft, avoiding conflicts with previously routed aircraft. Due to this fact, conventional routing algorithms need to be modified to take into account scheduling aspects of the problem. Several single-objective routing and scheduling algorithms have been proposed for the ground movement problem (Ravizza et al., 2013a; Lesire, 2010; Clare and Richards, 2011). However, the single-objective routing and scheduling algorithms can only find a single route, rather than a set of optimal routes representing the trade-off between different objectives. To the best of our knowledge, apart from (Ravizza et al., 2013b; Chen et al., 2016a; Evertse and Visser, 2017) no other multi-objective multi-graph routing and scheduling algorithms have been proposed for the ground movement problem. However, approaches in (Ravizza et al., 2013b; Chen et al., 2016a; Evertse and Visser, 2017) have their limitations in that not all optimal solutions can be discovered or several runs of the algorithm are needed.

In this paper, we extend a conventional multi-objective shortest path algorithm (Mandow et al., 2005; Mandow and De La Cruz, 2010) to support trajectory-based ground operations (Chen et al., 2016b,a) formulated as a multi-graph. In order to accelerate the search, while not compromising the optimality of solutions, two heuristic functions are described. Two cases are considered (1) multi-objective search, (2) search incorporating user preferences. The preference-based approach using economic value of objectives is further proposed to reduce the search space of the problem. The contributions of this paper are summarised as follows:

1. We introduce a multi-objective routing and scheduling algorithm for airport ground movement capable of finding a representative set of optimal or near optimal solutions in a single run when the sequence of aircraft is fixed.
2. A useful insight into how the structure of the multi-graph affects computational complexity and quality of solutions. This in turn can shed light upon how to construct the multi-graph itself. We also investigated conditions when a multi-objective multi-graph search can be simplified into a single-objective simple graph search when preference information is available.

The paper is organised as follows. The ground movement problem and related work are defined and reviewed in Section 2. Section 3 describes the proposed algorithm. Section 4 presents experimental results of the proposed algorithm for the cases without and with preferences on data instances from Hong Kong International, Beijing Capital International and Doha International Airports. Lastly, Section 5 draws conclusions and identifies challenges for the future work.

2. Ground movement problem

In this section, we review previous research on ground movement and multi-objective shortest path problem. Then, the ground movement problem is defined in detail.

2.1. Previous research

The previous works related to the ground movement problem can be categorised into two groups:

1. Sequential approach where aircraft are routed and scheduled according to a predetermined sequence (usually First-come-first-served order). The algorithm searches for a route and schedule of a single aircraft, respecting the routes of previously routed aircraft.
2. Global approach where routes and schedules of all aircraft are considered as a decision variable at the same time.

The sequential approaches use derivatives of the shortest path algorithms such as Dijkstra's (Dijkstra, 1959) and A* (Hart et al., 1968) algorithms to route aircraft. The Quickest Path Problem with Time Windows (QPPTW) algorithm (Ravizza et al., 2013a) is a modified Dijkstra's algorithm taking into account reservations of edges imposed by previously routed aircraft. The time to traverse an edge was based on a constant speed determined by an estimated taxi time. The QPPTW algorithm was also used in Benlic et al. (2016), Brownlee et al. (2018). Similarly, the A* algorithm was adapted in studies by Lesire (2010) and Zhou and Jiang (2015), again based on constant speeds for calculating traversal times.

In the global approaches, (1) a set of predefined routes is available for each aircraft and the aim is to select the best route for all aircraft at the same time. Genetic algorithms was employed for this purpose in Gotteland et al. (2003), Deau et al. (2009) and Mori (2013), whereas Guépet et al. (2017) and Roling and Visser (2008) used mixed integer linear programming (MILP) to find the best route. Adacher et al. (2018) formulated the ground movement problem as a job-shop scheduling problem and adopted a heuristic

approach to select a route. (2) The sequence of nodes traversed is the decision variable. In Marín and Codina (2008), Clare and Richards (2011) and Samà et al. (2017), aircraft can take any route on the graph and MILP searches for the nodes and when they should be traversed. Genetic algorithms were used to evolve routes in Jiang et al. (2013).

In the above mentioned approaches, the best route was defined as the one with the minimum taxi time or other time associated objectives, e.g. deviation from the assigned runway slot. However, as demonstrated in Chen et al. (2016b,a), reducing taxi time requires longer and more frequent acceleration phases, resulting in a higher fuel burn. Furthermore, the assumption of constant speeds can lead to unrealistic instructions causing potential conflicts with other aircraft (Chen et al., 2016a). Recently, research on ground movement has started to consider more detailed aircraft movement, capturing acceleration and deceleration phases and the related fuel consumption. This leads to the multi-objective shortest path problem (MSPP) algorithms being adapted for the ground movement problem. We will review approaches for the general MSPP and then describe algorithms specifically designed for the ground movement problem based on the MSPP.

The enumerative algorithms proposed for the MSPP, including label setting (Martins, 1984) and label correcting (Skriver and Andersen, 2000) algorithms, work in a similar way to the single-objective Dijkstra's algorithm (Dijkstra, 1959). In order to address the multi-objective nature of the problem, the Pareto dominance is used for comparing the costs of routes. The enumerative search can be accelerated if additional heuristic information is available. Stewart and White (1991) extended the A* algorithm (Hart et al., 1968) to a multi-objective A* (MOA*). Tung and Chew (1992) proposed an efficient way to calculate heuristic information as the minimum of each objective. The NAMOA* (New approach to multi-objective A*) algorithm (Mandow et al., 2005; Mandow and De La Cruz, 2010) improved the performance of the search compared to the MOA* by selecting and expanding routes instead of nodes.

Ranking approaches such as Climaco and Martins (1982) generate several k shortest routes in terms of the first objective, in order to find optimal routes for the other objectives. In practical applications, the value of k is usually set to a low value to maintain reasonable computational times. Two phase methods (Raith and Ehrgott, 2009) for bi-objective problems aggregate the objective function in the first phase. As not all solutions can be discovered this way, the enumerative approach is used for the remaining solutions in the second phase. Other approaches include metaheuristics (Chitra and Subbaraj, 2010; Pangilinan and Janssens, 2007) to approximate the Pareto front.

The approaches for the MSPP can be applied to the multi-objective multi-graph ground movement problem by iterating over all parallel edges and considering the routes of previously routed aircraft. A sequential approach based on the ranking method (Climaco and Martins, 1982) was introduced in Ravizza et al. (2013b) denoted as k-QPPTW for this problem. The algorithm decomposed the combined routing (search for a route) and scheduling problem (search for speed profiles) into two sequential steps. (1) In the first stage, the algorithm generated $k = 3$ shortest routes in terms of taxi time assuming a constant taxiing speed. The assumption of constant speed enables the separation of the search for a route from the search for a speed profile and avoiding the construction and search of the entire multi-graph. (2) Subsequently, the algorithm assigned feasible speed profiles to each route. As showed in Chen et al. (2016a), the timings generated by constant speed are difficult to comply with realistic speed profiles that take accelerations/decelerations into account. Therefore, these timings were not considered when selecting feasible speed profiles. Finally, for each aircraft one nondominated solution with corresponding route and speed profile was selected. The disadvantages of this approach, as pointed out in Chen et al. (2016a), are: (1) only a limited number of routes are explored during the search. The algorithm is guaranteed to discover all optimal routes only if $k \to \infty$. (2) A constant taxiing speed is assumed during the search for a route. As a result, the shortest route in terms of taxi time assuming a constant speed is not necessarily the same as the shortest one considering realistic speed profiles. It should be noted, that even if costs from realistic speed profiles would be used to search for routes, the algorithm would still only explore a limited number of routes thus potentially missing good solutions. As demonstrated later in this paper, the above mentioned shortcomings significantly compromise the quality of solutions found by the k-QPPTW algorithm. A global approach based on MILP was employed in Evertse and Visser (2017) considering taxi time, deviation from departure slots and emissions in a weighted aggregation objective function. For each edge of the graph, the algorithm selects a speed for traversal. Acceleration/deceleration after/before the constant speed phases related to breakaway, hold or turn were determined for calculation of taxi time and fuel flow. However, the algorithm assumes acceleration/deceleration always to/from the maximum allowed speed and speed changes between two edges to be instant. Furthermore, due to the weighted aggregation objective function, several runs of the algorithm with different weights are needed in order to approximate a trade-off among objectives.

The main advantage of sequential algorithms is their lower computational complexity compared to the global approaches. This is caused by focusing the search for a single (or limited number of) aircraft at a time. On the other hand, global approaches consider all the decision variables at the same time thus obtaining potentially better solutions at the cost of higher complexity. A quantified comparison of sequential and global approaches in terms of the quality of solutions and complexity is often missing in the literature. Computational experiments showed that the taxi time found by the sequential QPPTW algorithm (Ravizza et al., 2013a) can be improved by 0.88% when the interdependence of aircraft is considered as well.

It is evident from the review of previous literature that a multi-objective routing algorithm for the ground movement that can find high quality solutions or a representative subset of them is lacking. In light of this, the NAMOA* algorithm (Mandow et al., 2005; Mandow and De La Cruz, 2010) is adapted in this paper to the ground movement problem, as it can find a set of Pareto optimal or near optimal solutions while utilising a heuristic function for speeding up the search.

2.2. Problem description

The airport layout is represented as a directed graph $G = (V, E)$. Nodes $n \in V$ represent gates, stands, taxiway intersections, intermediate points and runway exits. Edges $e \in E$ represent taxiways between two nodes. Each edge e can be occupied by only

Table 1
Definitions.

Variable	Description
$G = (V, E)$	The directed graph of airport taxiways with nodes $n \in V$ and edges $e \in E$.
$\mathcal{F}_e$	The set of time windows associated with edge e.
$conf(e)$	The set of conflicting edges with edge e.
$s_{n,m} = (e_1, e_2, \dots, e_h)$	A segment which is a sequence of edges connecting two nodes n, m.
$V^s \subseteq V$	The subset of nodes which can be a start node of a segment.
p	The number of speed profiles.
q	The number of objectives.
$C_{n,m}$	The cost matrix of a segment between two nodes n, m with size $p \times q$.
$c_{n,m,l,*}$	Cost vector, i.e. the lth row of $C_{n,m}$.
$r \in R_i$	A route from the set of routes R_i with nondominated cost vectors for aircraft a_i.
$cost(r)$	The cost vector of route r.
$a_i \in \{ARR \cup DEP\}$	The aircraft a_i from the set of arrivals ARR and departures DEP.
obj_1, obj_2	The objectives, i.e. taxi time and fuel consumption.
$R_i = route(a_i)$	The set of routes R_i with nondominated cost vectors found by the procedure $route$ for aircraft a_i.
$start$, end	The start and end node of aircraft a_i.
t_i	The start time of aircraft a_i.
w_i	The weight category of aircraft a_i.
SG	The search graph for recording partial routes.
G_n^{op}, G_n^{cl}	The open and closed sets of nondominated cost vectors of routes reaching node n.
$OPEN$	The list of alternatives (n, g_n, f_n), which represent partial routes reaching node n.
g_n	The cost vector of a partial route reaching n.
$f_n = g_n + H(n, end)$	The vector of heuristic estimates for node n.
$H(n, end)$	The heuristic function returning a vector of estimates for a route from n to end.
$COSTS$	The set for recording cost vectors of routes which have reached the end node.
u	The number of rows in the submatrix $C'_{n,m}$ of $C_{n,m}$.
$w^P = (w_1^P, w_2^P)$	The vector of preferences representing unit costs.

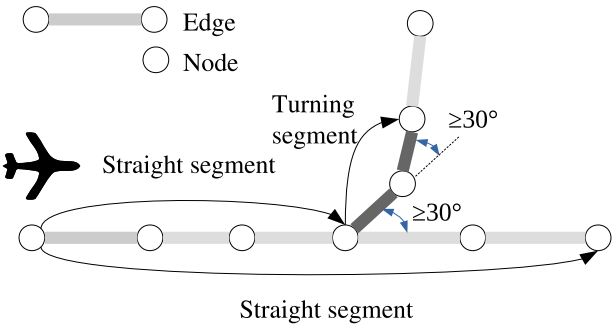

Fig. 1. Creation of segments.

one aircraft at a time. For this purpose, nodes are placed on taxiways to divide them into relatively short edges ensuring minimum safe separation distance of aircraft. In this paper, this distance is set to 60 m. It should be noted that due to short edges several aircraft can taxi simultaneously on a long taxiway. Each e has a list of time windows $\mathcal{F}_e$ associated with it. A time window is a time interval during which the edge is free to be traversed by an aircraft. Additionally, for each e a set of conflicting edges $conf(e)$ is defined. An edge conflicting with e is located within a threshold distance of 60 m. To ensure a safe separation of aircraft, when an aircraft is using e, time windows of $conf(e)$ are updated to prevent other aircraft occupying $conf(e)$ at the same time. A side effect of this approach is a potential over restriction of locations. For example if an aircraft is located at the beginning of a 60 m long edge, the neighbouring edge (60 m long as well) will be marked as conflicting even though the far end of the edge has a distance of 120 m to the aircraft. However, this can be mitigated by dividing the airport taxiways into shorter edges. The adoption of time windows and the sequential routing of aircraft effectively prevent any potential conflict between aircraft, e.g. head-on conflicts. Table 1 summarises the definitions used in this paper.

In addition to edges, we define a segment $s_{n,m} = (e_1, e_2, \dots, e_h)$ as a sequence of edges connecting two nodes n, m. The segments are created as showed in Fig. 1. If an edge and its predecessor edge (in the direction of a taxiing aircraft) have an angle ≥ 30 degrees (Khadilkar and Balakrishnan, 2012; Ravizza et al., 2013b), then it will belong to a turning segment. Otherwise, it is part of a straight segment. Consecutive edges of a similar type (straight, turning) are grouped together. For all nodes n, if they are a start node of a segment, they belong to $V^s \subseteq V$.

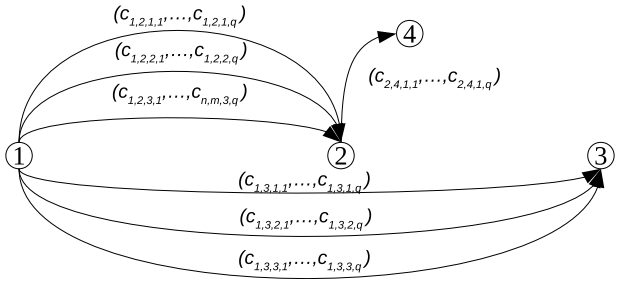

Fig. 2. Example of a directed multi graph.

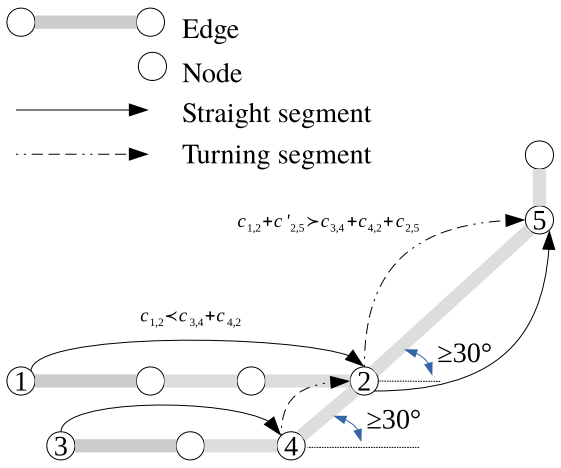

Fig. 3. Illustration of the nonadditivity property.

Each segment between nodes n, m has a cost matrix $C_{n,m} \in \mathbb{R}^{p\times q}$ with size $p \times q$ associated with it (defined in (1)), where p and q are the number of speed profiles and objectives, respectively.

$$C_{n,m} = \begin{pmatrix} c_{n,m,1,1} & c_{n,m,1,2} & \cdots & c_{n,m,1,q} \\ c_{n,m,2,1} & c_{n,m,2,2} & \cdots & c_{n,m,2,q} \\ \vdots & \vdots & \ddots & \vdots \\ c_{n,m,l,1} & c_{n,m,l,2} & \cdots & c_{n,m,l,q} \\ \vdots & \vdots & \ddots & \vdots \\ c_{n,m,p,1} & c_{n,m,p,2} & \cdots & c_{n,m,p,q} \end{pmatrix} \tag{1}$$

For a segment between two nodes n, m, each lth row $(c_{n,m,l,1}, c_{n,m,l,2}, \ldots, c_{n,m,l,q}) \in C_{n,m}$ is a cost vector which corresponds to a speed profile for that segment. Segments and corresponding cost matrices can be viewed as a directed multi-graph, for which an example is shown in Fig. 2.

A route r in G is defined as a sequence of segments $s_{n,m}, n, m \in V^s$ and corresponding cost vector $(c_{n,m,l,1}, c_{n,m,l,2}, \ldots, c_{n,m,l,q})$ for each segment. A cost vector of r $cost(r)$ is the sum of cost vectors of its component segments.

The set of all aircraft consists of a set of arriving aircraft ARR and departing aircraft DEP. Each aircraft a_i has a corresponding start time t_i, start node $start$, end node end and weight category $w_i \in \{\text{medium, heavy}\}$ associated with it. Note, that light weight category aircraft are not considered here, as aircraft within this category (small aircraft) are not present in the data instances used in this paper. The aim of the ground movement problem is to find a set of routes with nondominated cost vectors for each $a_i \in \{ARR \cup DEP\}$ in G from $start$ to end, starting at t_i and respecting time windows.

Cost vector c is said to be dominating c', denoted as $c \prec c'$,

$$\forall c, c' \in \mathbb{R}^q \quad c \prec c' \Leftrightarrow \forall j \quad c_j \leq c'_j \wedge c \neq c', \tag{2}$$

where c_j denotes the jth element of c. $nondom(X)$ is defined as a set of nondominated vectors in set X:

$$nondom(X) = \{c \in X : \nexists c' \in X \quad c' \prec c\}. \tag{3}$$

It is worth pointing out that as the designation of an edge as part of the turning segment is depending on the predecessor edge, the objectives are nonadditive (Carraway et al., 1990). As a result, for an optimal route, its sub-route is not necessarily optimal. Such

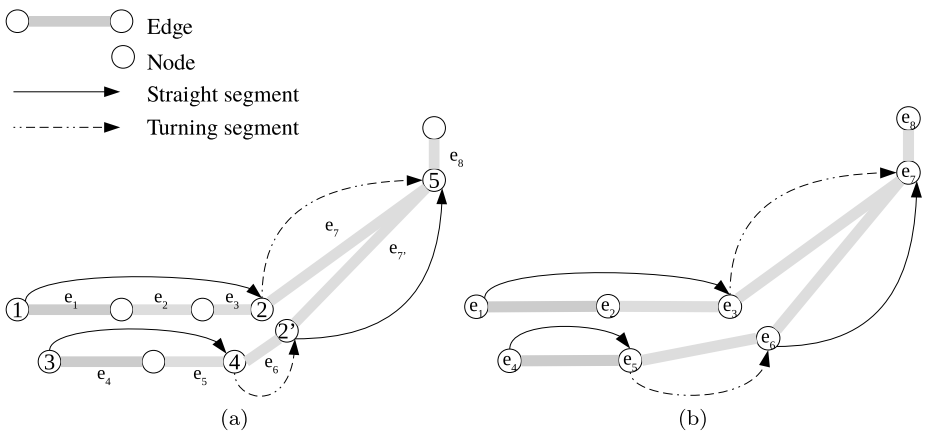

Fig. 4. Modelling approaches for preventing nonadditivity: (a) edges having a fixed designation as a turn or straight, (b) dual model.

sub-routes are discarded during the search. Therefore, this optimal route may be not found. We illustrate such a case in Fig. 3. There are two routes reaching node 2, the first one consisting of a straight segment with cost $c_{1,2}$. The second route consists of a straight and turning segment with costs $c_{3,4}$ and $c_{4,2}$. For costs of the sub-routes to node 2, $c_{1,2} \prec c_{3,4} + c_{4,2}$ holds. The subsequent segment 2–5 is a turning segment with cost $c'_{2,5}$ if the predecessor segment is 1–2 and a straight segment with cost $c_{2,5}$ if the predecessor is 4–2. A turning segment has usually higher costs (i.e. longer taxi time and fuel consumption) than a straight segment with the same length due to aircraft moving more slowly during turning. As 2–5 is a longer turning segment than 4–2, 2–5 has higher costs, i.e. $c_{1,2} + c'_{2,5} \succ c_{3,4} + c_{4,2} + c_{2,5}$. During the search, the second route via node 4 is eliminated upon reaching node 2 and the optimal route to node 5 via node 4 is therefore not found.

Fig. 4 shows alternative modelling approaches for preventing the above mentioned nonadditivity. A more detailed modelling of turns such as duplicating node 2 would enable each edge having a fixed designation as a turn or straight. Another approach is a dual model (Gotteland et al., 2003), where the nodes represent the taxiway edges and edges represent the taxiway intersections. Nodes have weights corresponding to taxi time or fuel consumption and edges have weights for transitions between the taxiways. A transition between two edges representing a turn imposes weights corresponding to turn taxi time and fuel consumption. For two edges within a straight segment the transition imposes zero weights. These approaches could be adopted in future work. In this paper we found nonadditivity cases were rare in practice as it is possible only if (1) a node is an end of both straight and turning segment at the same time, and (2) all objectives are dominated in order to be eliminated from the search. For the airport instance from Hong Kong International Airport investigated in this paper, the number of such cases has been counted maximum 1.72% of all pairs of parallel segments similar as in Fig. 3. Furthermore, adopting the same modelling approach as QPPTW and k-QPPTW algorithms facilitates fair comparison of these algorithms with the proposed algorithm and enables to employ the same database of speed profiles (Weiszer et al., 2015b) regardless of the routing and scheduling algorithm.

2.3. Objectives

As explained above, each row in $C_{n,m}$ is a cost vector corresponding to a speed profile for segment $s_{n,m}$. Each objective for a route r of a_i is a sum of the jth components of the cost vectors corresponding to each segment $s_{n,m}$ in that route:

$$obj_j = \sum_{s_{n,m} \in r} c_{n,m,l,j}. \tag{4}$$

Where, n, m and l are the decision variables. In this paper, we consider two objectives: obj_1 is the taxi time and obj_2 is the fuel consumption. Speed profiles, with the corresponding nondominated costs for each objective are stored in a database. The database of speed profiles is created in the preprocessing step using G, as described in Weiszer et al. (2015b), such that every possible segment for a specific airport is stored in the database. Each straight segment is further categorised as breakaway, intermediate and holding determining their start/end speed. The start/end speed is fixed to 5.14 m/s (10 knots) except for breakaway and holding segments where the start and end speeds are 0, respectively. As mentioned above, consecutive straight and turning edges are grouped together and therefore straight and turning segments always alternate in a route (i.e. there are no two consecutive straight/turn segments as these would be grouped together). Due to the fixed start/end speed of each segment, straight and turning segments from the database can be seamlessly combined such that any route between gates/stands and runway (or vice versa) can be recreated. Speed profiles for a single segment are a continuous function of time as detailed in Chen et al. (2016b). Due to infinite degrees of freedom, the speed profile in this paper is reduced by adopting a piece-wise linear function with four phases (Chen et al., 2016b) including acceleration, constant speed, deceleration and rapid deceleration as shown in Fig. 5. However, more complex speed profiles (e.g. non-linear) could be adopted and stored in the database in future.

Table 2
Definitions of aicraft parameters.

Variable	Description
acc	acceleration rate
$\tau_i^{phase}(s_{n,m})$	taxi time spent in phase $phase$ of segment $(s_{n,m})$
η	thrust level
F_{oo}	maximum power output of jet engine
$weight$	aircraft weight
μ	rolling resistance coefficient
$\phi_i^{phase}(s_{n,m})$	fuel flow in phase $phase$ of segment $(s_{n,m})$
t^{taxi}	total taxi time
$\tau_i(s_{n,m})$	taxi time for segment $(s_{n,m})$
$\boldsymbol{\Phi}_i(s_{n,m})$	fuel consumption for segment $(s_{n,m})$

The duration $\tau_i^{phase}(s_{n,m}) \geq 0$ of each $phase$ for segment $s_{n,m}$ determines the required taxi time $\tau_i(s_{n,m})$ for aircraft i to cover the distance between nodes n, m:

$$\tau_i(s_{n,m}) = \sum_{phase=1}^{4} \tau_i^{phase}(s_{n,m}). \tag{5}$$

Note, that the taxi time of a segment does not depend on aircraft start time t_i. Following the approach described in Chen et al. (2016b), fuel consumption needed to follow a speed profile depends on thrust levels η which are determined for each taxiing $phase$. During deceleration and rapid deceleration, $\eta = 5\%$ of the full rated power, and during turning $\eta = 7\%$ (Nikoleris et al., 2011). For acceleration and constant speed phases, η is calculated in (6) where $weight$ is the weight of the aircraft, acc is the acceleration rate, $\mu \cdot weight \cdot g^{acc}$ is the rolling resistance force and F_{oo} is the maximum power output of the jet engine. μ is rolling resistance coefficient, set to 0.015 (Chen et al., 2016b) in this paper. $g^{acc} = 9.81\ \mathrm{m \cdot s^{-2}}$ is the gravitational acceleration. Note, that the air resistance is not assumed here due to low speeds involved.

$$\eta = \frac{weight \cdot acc + \mu \cdot weight \cdot g^{acc}}{F_{oo}} \tag{6}$$

The thrust level η corresponds to a fuel flow $\phi_i^{phase}(s_{n,m})$ which is calculated by linearly interpolating or extrapolating fuel flow values for $\eta = 7\%$ and $\eta = 30\%$ reported in ICAO database, following the approach in Nikoleris et al. (2011). The fuel $\boldsymbol{\Phi}_i(s_{n,m})$ consumed to traverse segment $s_{n,m}$ is then calculated as:

$$\boldsymbol{\Phi}_i(s_{n,m}) = \sum_{phase=1}^{4} \phi_i^{phase}(s_{n,m}) \cdot \tau_i^{phase}(s_{n,m}). \tag{7}$$

In order to make the calculation tractable, for aircraft belonging to the same weight category, a representative aicraft is chosen in this study and its values are used in calculation. The aircraft parameters used to calculate obj_1 and obj_2 are summarised in Table 2.

In order to generate the Pareto set of speed profiles for each segment, a Population Adaptive Based Immune Algorithm (PAIA) (Chen et al., 2016b; Chen and Mahfouf, 2006) is employed in this paper. PAIA is an population-based evolutionary algorithm which iteratively evaluates, sorts, selects and improves the solutions representing the values of $\tau_i^{phase}(s_{n,m})$. The detailed description of PAIA for the speed profile generation is given in Chen et al. (2016b). In this paper, the parameters related to PAIA are set according to Chen et al. (2016b). The initial population size is 7; the number of generations is 40; the maximum clonal size is 95; clonal selection threshold is 0.4 and network suppression threshold is set to 0.008. It should be noted, that any search algorithm can be adopted for the speed profile generation.

The obtained Pareto set is then filtered by a filtering procedure (Weiszer et al., 2018) into p evenly distributed solutions. For each straight segment $s_{n,m}$, p speed profiles with nondominated costs are stored. Following our previous work (Weiszer et al., 2018), $p = 10$ and $p = 20$ for medium and heavy category aircraft, are respectively chosen as a representative subset of the Pareto set. For turning segments, a constant speed profile of 5.14 m/s (10 knots) is adopted as in Ravizza et al. (2013b).

If the aircraft under investigation cannot traverse segment $s_{n,m}$ as $s_{n,m}$ is occupied by other aircraft, it can be held before entering $s_{n,m}$ until $s_{n,m}$ becomes available. The corresponding cost of $s_{n,m}$ is then increased by holding time and fuel. The fuel consumed during the holding is based on idle fuel flow with $\eta = 4\%$. After holding, the aircraft needs also to accelerate to the speed matching the initial speed of $(s_{n,m})$ which incurs time to accelerate and corresponding fuel. The acceleration rate is fixed to $acc = 0.98\ \mathrm{m \cdot s^{-2}}$ in this case as in (Chen et al., 2016b).

3. Routing and scheduling framework

The routing and scheduling framework is outlined in Algorithm 1. The set of aircraft $\{ARR \cup DEP\}$ is sorted according to their start time t_i. This first-come-first-served sequencing has an advantage to consider aircraft sequentially as they become ready to start taxiing. For each aircraft a_i, a set of routes R_i with nondominated costs are found by procedure $route$. In Line 4, the algorithm considers two ways to proceed when no time windows are available for a_i. Firstly, if $holding$ is enabled the aircraft can wait at an intermediate node until a time window becomes available as will be detailed later. Otherwise, the start time is postponed by 60 s.

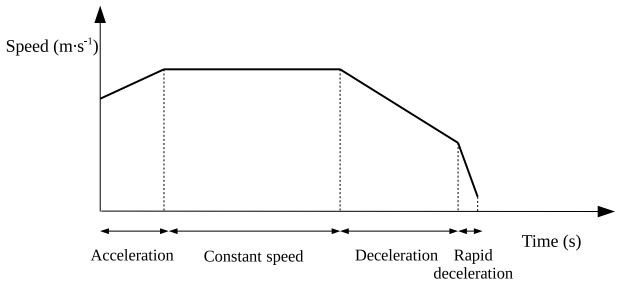

Fig. 5. An example of a speed profile with four phases.

This value is set approximately as airports usually operate (e.g. estimated time of departure) with a precision in minutes. Note, that postponing the aircraft will not change their order in the FCFS sequence. The start time is iteratively extended until time windows become eventually available. Once R_i is obtained, one route r is selected for a_i and time windows are updated for edges belonging to the segments of the route and their associated conflicting edges.

The selection of a route r in Line 8 enables to consider constraints such as target take-off time. Routes which violate such constraints can be eliminated in this step as well as when expanding routes as described later. Also, postponing aircraft for too long can cause gate allocation time violation. In such case, the algorithm will enable *holding* at an intermediate nodes and vacate the gate. The target take-off time and gate allocation constraints are not considered in the computational experiments in this paper as the data do not contain such information. However, as described above these constraints can be handled by the algorithm.

Algorithm 1: Routing and scheduling algorithm.

```
1  Sort aircraft according to t_i;
2  forall the a_i ∈ {ARR ∪ DEP} do
3  |  Route aircraft R_i = route(a_i);
4  |  if R_i = ∅ and holding not enabled then
5  |  |  t_i = t_i + 60 s;
6  |  |  Go to Line 3;
7  |  end
8  |  Select preferred r ∈ R_i;
9  |  Reserve route r and update time windows;
10 end
```

3.1. AMOA*

Procedure *route* is called every time an aircraft is to be routed. *route* is performed by airport multi-objective A* (AMOA*) algorithm which is based on multi-objective shortest path algorithm NAMOA* (Mandow et al., 2005; Mandow and De La Cruz, 2010). Algorithm 2 outlines AMOA*. Firstly, an acyclic search graph SG is created in Line 1. SG is used to record partial routes. A single node *start* is initially in SG. For each node n in SG, G_n^{op} and G_n^{cl} are the sets of nondominated cost vectors of routes reaching n. If a successor of n has been explored, the corresponding cost vector is in G_n^{cl} (closed), otherwise it is in G_n^{op} (open). A list $OPEN$ contains alternatives (n, g_n, f_n), which represent partial routes reaching n, which can be further expanded. An alternative is a list with node n, cost vector g_n and a vector of heuristic estimates $f_n = g_n + H(n, end)$. $H(n, end) = (c_1^{est}, c_2^{est}, \ldots, c_q^{est})$ is a function that for each node $n \in V$ returns a vector of heuristic estimates for the route from n to end. c_j^{est} denotes an estimate of the minimum cost of the jth objective (i.e. the lower bound). The heuristic function H is discussed in more detail in Section 3.2. Initially, $OPEN$ contains $(start, g_{start}, f_{start})$, where $g_{start} = (0, 0)$. In Line 4, $COSTS$ is used to record cost vectors of routes which have reached the end node.

In Line 5, the algorithm checks if $OPEN$ is empty. In such a case, no alternatives are left to expand, the search terminates and SG contains all nondominated routes from $start$ to end. To obtain an individual route r with cost $cost(r) \in COSTS$, SG is traversed backwards from end to $start$.

In line 8, the algorithm selects a nondominated alternative (n, g_n, f_n) from $OPEN$ based on cost estimate f_n. This alternative is deleted from $OPEN$ and g_n is moved from G_n^{op} to G_n^{cl}. If n is the end node end, then a route has been found. The cost g_n of this route is included in $COSTS$ and used to prune alternatives in $OPEN$ which are dominated by g_n. It should be noted, that additional constraints such as the target take-off time can be used to prune selected solutions which violate such constraints in Line 8.

If n is not the end node, then it is further expanded in Line 15. Algorithm 3 details the expansion procedure. In Line 1, successor segments starting in n are generated. A successor segment $s_{n,m}$ from n to m is created by iteratively expanding the edge n, n', where n' is the successor of n in G, if the angle between the edge and its preceding edge is not greater than or equal to 30 degrees (as shown in Fig. 1). Straight segments are further divided into straight breakaway (if n is gate/stand) and straight holding segments

Algorithm 2: AMOA*.

Create:
An acyclic search graph SG ;
Two sets $G^{op}_{start} = \{(0,0)\}, G^{cl}_{start} = \{\emptyset\}$;
A list of alternatives $OPEN = \{(start, g_{start}, f_{start})\}$;
$COSTS = \{\emptyset\}$;
/* Check Termination: */
if $OPEN$ *is empty* **then**
  search backward in SG from *end* node and return set of routes R_i.
end
/* Path selection: */
Select an alternative (n, g_n, f_n) from $nondom(OPEN)$.;
Delete (n, g_n, f_n) from $OPEN$ and move g_n from G^{op}_n to G^{cl}_n.;
if *n is end node* **then**
  /* Solution recording: */
  Include g_n in $COSTS$;
  Eliminate from $OPEN$ all alternatives (x, g_x, f_x) if $g_n \prec f_x$;
  Go to Line 5;
else
  /* Path expansion: */
  Expand (n, g_n, f_n). ;
  Go to Line 5;
end

(if m is gate/stand or runway exit). Then, for each successor segment that does not produce a cycle in SG speed profiles and the corresponding cost matrix $C_{n,m}$ are retrieved from the database. The loop in Line 4 iterates over p rows in $C_{n,m}$, and the current lth row is considered in Line 5. For each edge belonging to segment $s_{n,m}$, entry and exit times are calculated, according to the speed profile. For each edge e within the segment and conflicting edges $conf(e)$, time windows $\mathcal{F}_e$ are checked. If some edge e does not have any time window available, then the following options are carried out. If *holding* is enabled at intermediate nodes, the holding time is calculated as maximum time window violation among all edges of $s_{n,m}$. Then, the cost of holding is added to $c_{n,m,l,*}$ as described in Section 2.3 and time windows are checked again. If *holding* is not enabled, then the current row in $C_{n,m}$ is not considered. The new cost of the route found to m is calculated as $g_m = g_n + (c_{n,m,l,1}, c_{n,m,l,2}, \dots, c_{n,m,l,q})$. The new cost is used to update $OPEN$, G^{op}_m and SG in Lines 18–18. Note that for node n and its successor m, an edge (m, n) is inserted in SG as SG is searched backwards from *end* to *start* in Algorithm 2. Any vectors dominated by g_m are discarded from G^{op}_m and G^{cl}_m, together with the corresponding alternatives from $OPEN$ in Line 26.

One of the challenges of real-world airport operations is the inherent nature of uncertainty. Due to the influence of various factors such as human factors of pilots, weather or delays, the aircraft movements may deviate from the pre-calculated speed profiles. In general, there are two ways how to consider these uncertainties in routing and scheduling: (1) a proactive approach where the movements include buffer times (Ravizza et al., 2013a) or are characterised by distribution of taxi times, fuel consumption, etc., rather than a deterministic value. The routing and scheduling algorithm then incorporates times with buffers or distributions and selects robust routes and schedules. One example considering uncertain distributions is the Fuzzy QPPTW algorithm (Brownlee et al., 2018); (2) A reactive approach where the routing and scheduling algorithm reacts to disturbances after they occur by replanning (e.g. Evertse and Visser, 2017). Quantifying buffer times for continuous speed profiles or the integration of speed profiles with uncertainty distributions in the context of a multi-objective routing and scheduling algorithm are complex tasks and beyond the scope of this paper. AMOA* can be used in the reactive mode when the computational time is short enough. Indeed, the sequential nature of AMOA* facilitates replanning of a single aircraft without affecting other aircraft. For cases with long computational times, an approximate algorithm can be adopted in future work.

3.2. Heuristic functions

The heuristic function H plays an important role in AMOA* as it determines if an alternative should be included in $OPEN$ and which alternative from $OPEN$ should be selected for expansion. It can speed up the search by limiting the number of alternatives to be expanded (Lines 18 and 28 in Algorithm 3 and Line 12 in Algorithm 2). Function $H(n, end)$ provides optimistic estimates of costs for the route from n to end, i.e. estimates are always smaller than the real costs. Let $H^*(n, end)$ be the real cost vector of the route from n to end. The heuristic function H is admissible if $H(n, end) \preceq H^*(n, end) \quad \forall n, end \in V$, where $\preceq$ denotes "dominates or equals". It has been proven in Mandow and De La Cruz (2010) that if H is admissible, the search with H will return no fewer optimal routes than it would without the heuristic function.

The estimates provided by H represent an approximate lower bound of costs. The more accurate H is, the less alternatives are considered for expansion, thus speeding up the search. Let $H0(n, end) = \{0, \dots, 0\}$ be a heuristic function that returns a value of 0

Algorithm 3: Route expansion.

```
   /* Path expansion:                                                                    */
1  Generate all segments starting at node n;
2  for All successor segments of n to node m without cycles in SG do
3      Read C_{n,m} from the database;
4      for l = 1, ..., p do
5          Cost vector is c_{n,m,l,*} = (c_{n,m,l,1}, c_{n,m,l,2}, ..., c_{n,m,l,q});
6          Calculate entry and exit times for all edges in segment s_{n,m};
7          Check time windows for all edges within segment;
8          if Time windows are not available for all edges in s_{n,m} then
9              if holding enabled then
10                 add cost of holding to c_{n,m,l,*} and Go to Line 6;
11             else
12                 l = l + 1 and Go to Line 5;
13             end
14         end
15         Calculate the cost of the new route found to m: g_m = g_n + c_{n,m,l,*};
16         if m is not in SG then
17             Calculate f_m = g_m + H(m, end);
18             if f_m is not dominated by any vector in COSTS, then
19                 put (m, g_m, F_m) in OPEN;
20                 put g_m in G_m^{op};
21                 create edge (m, n) in SG with labels g_m and c_{n,m,l,*};
22             end
23         else if g_m equals some cost vector G_m^{op} ∪ G_m^{cl} then
24             create edge (m, n) in SG with labels g_m and c_{n,m,l,*};
25         else if g_m is not dominated by any vector in G_m^{op} ∪ G_m^{cl} then
               /* a path to m with new interesting cost has been found                   */
26             Eliminate from G_m^{op} and G_m^{cl} vectors dominated by g_m. For each vector g'_m eliminated from G_m^{op}, eliminate the
               corresponding (m, g'_m, f'_m) from OPEN;
27             Calculate f_m = g_m + H(m, end);
28             if f_m is not dominated by any vector in COSTS, then
29                 put (m, g_m, F_m) in OPEN;
30                 put g_m in G_m^{op};
31                 create edge (m, n) in SG with labels g_m and c_{n,m,l,*};
32             end
33         end
34     end
35 end
```

for all objectives. In this case, the estimate information is not utilised at all. This can be considered as a baseline heuristic. In this paper, in order to speed up the search, we propose two heuristic functions:

(1) $H1$ is defined as follows. (I) For taxi time, it is calculated as the length of the shortest path to *end* in terms of distance divided by the maximum allowed speed (30 kts). (II) For the fuel consumption, taxi time estimated using (I) is multiplied by the idle fuel flow for medium category aircraft. The shortest path and idle fuel flow provide a lower bound, as aircraft cannot travel a shorter distance or consume less fuel. Note that the idle fuel flow of heavy category aircraft is assumed to be greater than or equal to the idle fuel flow for medium category aircraft, satisfying the admissibility condition.

(2) $H2$ is based on the heuristic function proposed by Tung and Chew (1992) and defined as follows. For each $n \in V^s$ which is a possible start node of a segment all nondominated routes to *end* are found by AMOA*, assuming no other aircraft are present at the airport. $H2(n, end)$ then returns the minimum value of taxi time and fuel consumption from the set of costs of all nondominated routes. Therefore, a tuple (taxi time, fuel consumption) in this case is effectively an ideal point of Pareto front of nondominated and unimpeded routes, representing the minimum time/fuel consumption that can be achieved from n to *end*.

As heuristic function $H2$ requires finding optimal unimpeded routes for each n to every possible *end*, they are time consuming due to the large number of nodes. Therefore, only $n \in V^s$ and $end \in V^{end}$ are considered, where $V^{end} \subseteq V$ denotes a set of gates and runway exits. Furthermore, all $C_{n,m}$ are assumed to be for the medium category aircraft and no breakaway segment is considered in order to use the same heuristic function for arrivals and departures. Note that these assumptions do not violate the admissibility property of the heuristic function.

3.3. Multi-graph reduction

The size of $C_{n,m}$ significantly affects the size of the search space. However, the size of the search space can be reduced if only selected rows of $C_{n,m}$ are considered. Let $C'_{n,m}$ with size $u \times q$ be a submatrix of $C_{n,m}$ for $\forall n, m \in V$ and $u \leq p$. Considering only $C'_{n,m}$ instead of $C_{n,m}$ reduces the size of the multi-graph. Two approaches for submatrix selection are considered in this paper: (1) From the Pareto front for a segment, u evenly distributed solutions are selected. Rows in $C_{n,m}$ are ordered according to the first objective and u solutions are selected with even distance from each other according to the first objective. Note, that no preference information for selecting the submatrix is needed. (2) If preferences for the search are known beforehand, the decision maker can express a preference according to the ranking of the rows in $C_{n,m}$, from which a subset of $C_{n,m}$ of size u is selected. For example, the decision maker can be interested only in the lth row in $C_{n,m}$ for each n, m, corresponding to the fastest solution or the most fuel efficient one. In this case, $u = 1$. Similarly, the decision maker can be interested in several rows in C, i.e. $u \geq 2$. This approach is denoted as AMOA* with preferences.

The reduction affects the number of found solutions. Therefore, not all solutions may be discovered. Furthermore, with a reduction $u < p$ the combination of all p speed profiles, e.g. the fastest speed profile on one segment and the most fuel efficient one on another segment, is not searched. This may affect not only the number of found solutions but also available time windows as a result leading to worse solutions.

Note, that if $u = 1$, the following reduction is possible: (1) the multi-graph is *de facto* reduced to a simple graph with multiple objectives on its segments. (2) the multi-objective search can be transformed into a single-objective one. The transformation is achieved by aggregating objective values pertaining to segments into a single value if preferences used to select the route for a_i from R_i are the same ones as those used to create $C'_{n,m}$. Note that such transformation into a single objective search can compromise optimality of solutions in multiple cases. Firstly, if the weighted sum aggregation is used, it is not able to find solutions on a non-convex Pareto front. Secondly, the single objective search is more likely to encounter non-nonadditivity as described in Section 2.2 due to the absent Pareto dominance check. This will be demonstrated later in Section 4.6. Finally, as mentioned above the combination of all p speed profiles is not explored, potentially missing good solutions as the *preferred* speed profile may not comply with time windows. It is worth noting that the multi-objective search is equivalent to the single-objective search in a special case when the cardinality of the Pareto front is equal to one. This is often the case when objective values pertaining to segments are correlated as indicated in Mote et al. (1991).

As an example of preferences, economic costs pertaining to each objective were specified in Chen et al. (2016a). Each second of taxi time and kg of fuel burned incur economic costs. The values of obj_1, obj_2 of each row in $C_{n,m}$ are multiplied by corresponding unit costs $w^P = (w_1^P, w_2^P)$, where w_1^P, w_2^P are the unit costs of taxi time and fuel consumption, respectively. The aggregated costs then give the ranking of the rows in $C_{n,m}$ and lower costs are preferred. If any of the unit costs equals M, where M is a large positive number, the decision maker prefers the solution with the minimum of the corresponding objective. The same approach is applied for selecting the cost optimal route from a set of nondominated routes (Chen et al., 2016a).

4. Computational results

This section provides details of experiments with AMOA*. The AMOA* was programmed in Python as a single-threaded application and executed on a laptop computer (Intel Core i7, 2.6 GHz, 16GB RAM). For experiments with runtimes of days a university high-performance computer [1] (HPC) was used. In order to speed up the online computation, heuristic function values, set of conflicting edges and set of segments were pre-computed. As will be described later, the computational time of $H2$ is highly dependent on the size of the airport. Therefore, it was only used in experiments to investigate the impact of different heuristics on the computational time. In all other experiments, H1 was used. Note that the choice of the heuristic function only affects the computational time, not the optimality of solutions. In practical applications, it is assumed that enough computational time and resources are available to calculate $H2$.

4.1. Data

In this paper, we use a set of instances of real arrival and departure flights from 3 airports: Doha International Airport (DOH), Hong Kong International Airport (HKG) and Beijing Capital International (PEK). The complexity of the taxiway layout ranges from simple (DOH), medium (HKG) to complex (PEK), as shown in Fig. 6. The graphs and flights are detailed in Table 3 and the instances contain flights from 16.3.2014 (17:00–23:00) for DOH, 17.1.2017 (0:00–24:00) for HKG and 9.7.2014 (9:00–14:00) for PEK. The taxiway layout was processed from OpenStreetMap[2] by tools from (Brownlee et al., 2014). All edges were set as bi-directional. The flights for DOH and HKG instances were captured with specialised tools described in (Brownlee et al., 2014) from freely-available data on FlightRadar24.com. The data specifies landing/pushback times, gates/runway exits and weight category for each flight.

As mentioned in Section 2.3 all aircraft have been categorised into medium and heavy weight categories and representative aircraft is designated for each category: Airbus A320 and A333 for medium and heavy category, respectively. The specifications of the representative aircraft are used for calculation of fuel consumption (see Table 4).

[1] https://doi.org/10.5281/zenodo.438045.

[2] www.openstreetmap.org.

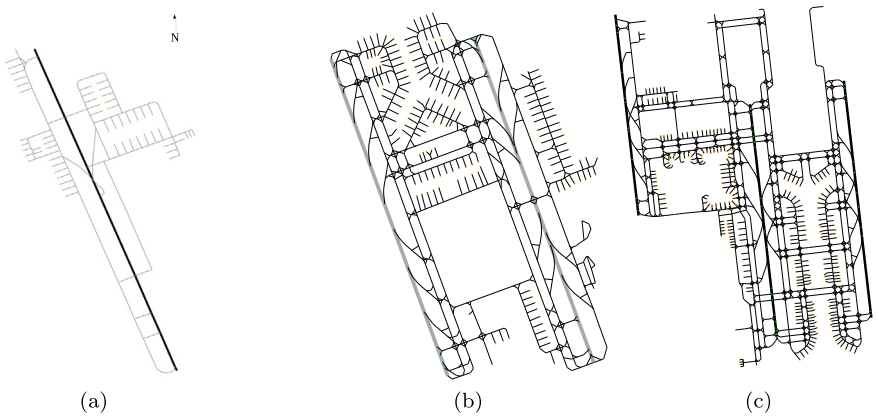

Fig. 6. A directed graph representation of the airport surface for (a) Doha International Airport, (b) Hong Kong International Airport, (c) Beijing Capital International Airport.

Table 3
Data instances.

	nodes	edges	gates	rwy exits	arrivals	departures	aircraft
DOH	434	436	55	14	103	77	180
HKG	1309	1491	160	38	290	216	506
PEK	3194	3928	286	53	185	164	349

Table 4
Specifications of the representative aircraft.

	Airbus A320	Airbus A333
Take-off weight	78000 kg	230000 kg
Engines	CMF56-5-A1	CF6-80E1A2
Number of engines	2	2
Rated output F_{oo}	2 × 111.2 kN	2 × 287 kN
Rolling resistance	11.48 kN	33.84 kN
Fuel flow at 7% F_{oo}	0.101 kg·s^{-1}	0.228 kg·s^{-1}
Fuel flow at 30% F_{oo}	0.291 kg·s^{-1}	0.724 kg·s^{-1}

4.2. Description of baseline approaches

To illustrate the difference between AMOA* and previous approaches, k-QPPTW (Ravizza et al., 2013b), QPPTW (Ravizza et al., 2013a) and Shortest path algorithm are used for comparison. k-QPPTW and QPPTW algorithms used in this paper follow Ravizza et al. (2013b). These two algorithms have been adapted to use the database of speed profiles (Weiszer et al., 2015b).

k-QPPTW is a representative algorithm of a class of multi-objective ranking algorithms. As mentioned in Section 2.1, the main shortcoming of k-QPPTW algorithm is the limited number of explored routes and assumption of constant speed in the route search.

The k-QPPTW algorithm processes aircraft sequentially and the procedure for routing a single aircraft is shown in Algorithm 4. k fastest routes are generated, based on assumed constant speed 10 m/s for straight segments and 5.14 m/s (10 knots) for turns, respecting time windows. Each route is divided into segments. For each segment $s_{n,m}$, the lth row of $C_{n,m}$ is considered and the route with the summed cost is added to the Pareto front. If no solutions are found in Line 10 as a result of no available time windows, then the start time is postponed by 60 s. The option with $holding$ enabled was not implemented in k-QPPTW and therefore is not used here. After the Pareto front for each route has been iteratively generated, the Pareto fronts are combined in Line 13, keeping only nondominated solutions. The final Pareto front is then discretised into $disc$ roughly equally spaced solutions. $disc$ is set to 10 and 20 for medium and heavy category aircraft, respectively. Finally, the preferred route is selected and reserved for aircraft a_i in Line 14.

The QPPTW and Shortest path algorithms are representatives of the single objective routing algorithms. It should be noted, that the single objective search applies only to finding a route. Once the route is found, multi-objective speed profiles are applied to this route, resulting in multi-objective final solutions. In order to fairly compare the algorithms with AMOA*, the QPPTW and Shortest path algorithms use a similar framework as is outlined in Algorithm 4. The QPPTW and the Shortest path algorithm are single-objective algorithms, and therefore return a single route in Line 1. QPPTW is a specialised routing algorithm taking into account time windows during the search for the route. The k-QPPTW algorithm is equivalent to QPPTW for $k = 1$. Therefore, k is set to 1 in Line 1. The found route is divided into segments, for which p rows of $C_{n,m}$ are iteratively applied in Lines 4–7. The rest of Lines 8–12, 14 are unchanged. Line 13 is skipped, as the number of solutions p is equal to $disc$.

Algorithm 4: k-QPPTW algorithm.

```
1  Generate the fastest k routes for a_i w.r.t. to time windows;
2  for each route k do
3  |  Divide the route into segments;
4  |  for l = 1, 2, ..., p do
5  |  |  Use the l-th row of C_{n,m} for all segments s_{n,m};
6  |  |  Add a solution to the Pareto front;
7  |  end
8  end
9  if no solutions comply with time windows then
10 |  t_i = t_i + 60 s;
11 |  Go to Line 1;
12 end
13 Combine and discretise the Pareto front into disc solutions;
14 Select the preferred solution and reserve the relevant route;
```

In contrast, the Shortest path algorithm is a simple approach based on Dijkstra's algorithm. A shortest route in terms of physical distance is found in Line 1 without considering time windows. Time windows are only considered after the route has been found, discarding solutions not complying with them. The rest of Lines 2–12, 14 are the same as for the QPPTW algorithm.

In order to illustrate the similarity and difference between the multi-objective and single-objective search under multi-graph reduction when $u = 1$ and available preferences as described in Section 3.3, AMOA* is also compared with a single objective version of itself. To this end, the Pareto dominance check in AMOA* is replaced with a comparison based on economic costs. For each solution, the value of taxi time and fuel consumption is multiplied by the corresponding unit costs and solutions with smaller economic costs are selected. This version of the algorithm is denoted as airport single-objective A* (ASOA*). Note that ASOA* is a modified version of the multi-objective AMOA* and therefore not the most computationally efficient one compared to the pure single objective A* algorithm.

4.3. Pareto front for a single aircraft

In this section, we will give examples of results for a single aircraft in order to illustrate the performance of AMOA* in two specific cases. As all algorithms compared in this study are sequential, using a single aircraft can demonstrate the performance more clearly without the loss of generality.

Firstly, to compare different approaches, a medium weight category aircraft (id 61) from HKG instance with $p = 10$ speed profiles for each straight segment is examined without considering other aircraft in the airport. In sequential routing and scheduling algorithms routes of preceding aircraft affect the routes of subsequent aircraft. Excluding all other aircraft ensures that the time windows are all set to $(0, \infty)$ (i.e. unimpeded) for the different algorithms presented here.

AMOA* was run using $H1$ and $u = 10$ without preferences. For AMOA* with preferences, $u = 1$. Without the loss of generality, four arbitrary preferences are considered, (1) the solution with minimum taxi time for which $w^P = (M, 0)$, where M is a large positive number, (2) solution with minimum economic cost for $w^P = (0.469, 0.71)$, where the unit cost for taxi time is 0.469 €/s and 0.71 €/kg for fuel consumed as in Weiszer et al. (2015a), (3) solution with minimum economic cost for $w^P = (0.469, 1.42)$ with double unit cost for fuel and (4) solution with minimum fuel consumption for which $w^P = (0, M)$.

The Pareto front of routes obtained by different algorithms is shown in Fig. 7a. The Pareto front for AMOA* contains 89 nondominated solutions whereas the Pareto front for k-QPPTW/QPPTW/Shortest path algorithms has 10 nondominated solutions. AMOA* with preferences and ASOA* returned a single solution for each preference. As can be seen, AMOA*, AMOA* with preferences and ASOA* resulted in better solutions in terms of both objectives. k-QPPTW, QPPTW and Shortest path generated the same solutions for this aircraft. Fig. 7b shows that ASOA* reached solutions in different regions of the Pareto front with different preferences. When changing $u = 1, 2, \ldots, 10$ using the two multi-graph reduction methods (described in Section 3.3), the number of solutions found by AMOA* without and with preferences gradually increased with the larger value of u as shown in Fig. 7c and Fig. 7d. The results also illustrate the difference between the multi-graph reduction methods. The multi-graph reduction based on evenly distributed solutions gradually covers the whole Pareto front, the reduction based on preferences and ranking concentrates solutions close to one region of the Pareto front. The fastest routes for each algorithm from Fig. 7a are shown in Fig. 8. The fastest route found by AMOA* (409 s, 95 kg) is better than the fastest route found by other algorithms (448 s, 109 kg) due to lower number of turns. Specifically, the route found by AMOA* has only 5 turning segments compared to 6 in the other route. This results in shorter taxi time and less fuel consumption due to less acceleration after turning segments. Other algorithms cannot discover the most efficient route as: (1) The other algorithms explore only limited number of routes (3 routes for k-QPPTW and 1 route for QPPTW and Shortest path algorithm). (2) The search for the route is based on other objectives rather than real costs corresponding to the speed profiles. The k-QPPTW and QPPTW algorithms search for the fastest route based on the constant speed, the Shortest path algorithm is based on the shortest distance.

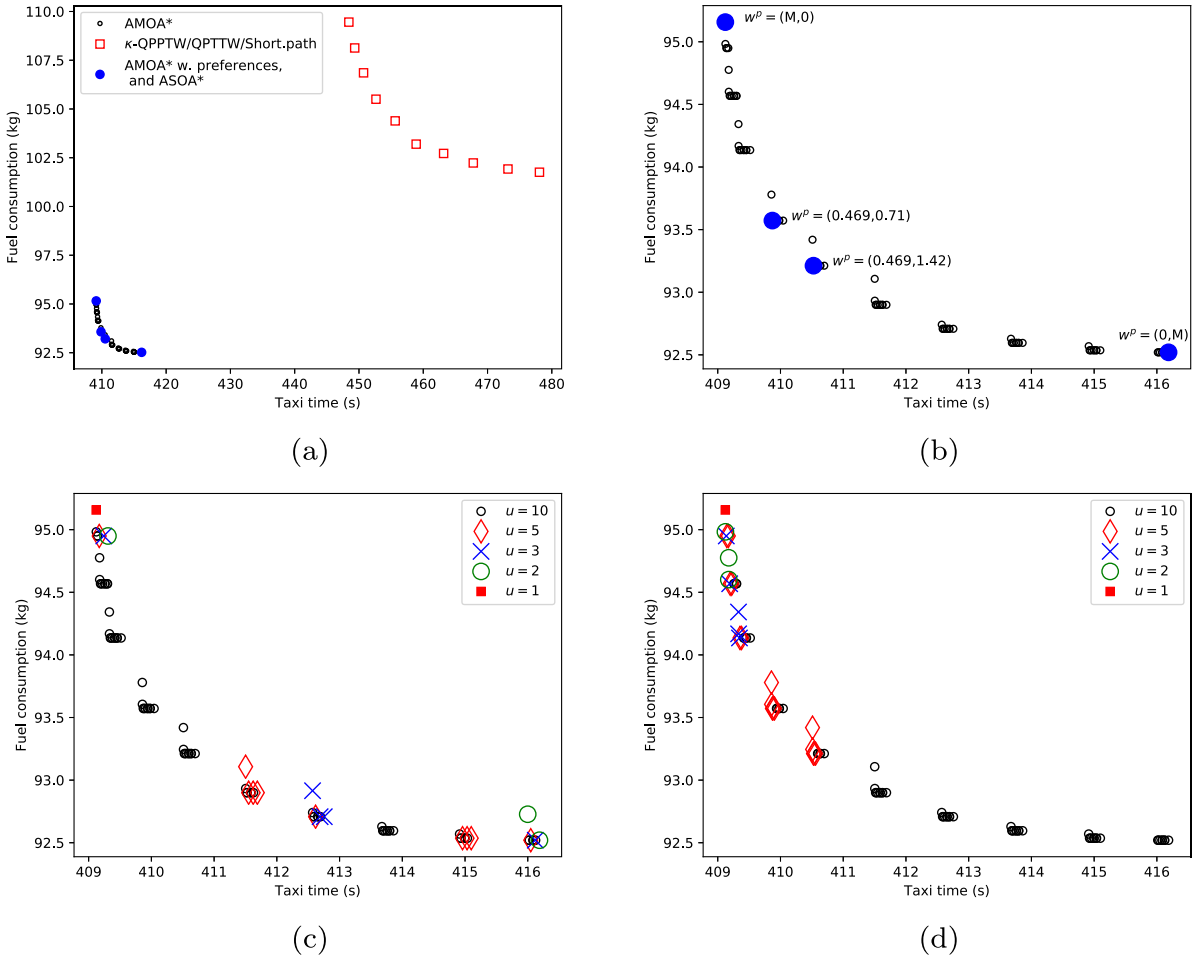

Fig. 7. Pareto front for aircraft id 61: (a) AMOA* with $u = 10$ compared with other approaches, (b) solutions obtained with $u = 1$ and different preferences, (c) solutions obtained by AMOA* with multi-graph reduction based on evenly distributed solutions and increasing u, (d) solutions obtained by AMOA* with preferences with $w^p = (M, 0)$ and increasing u.

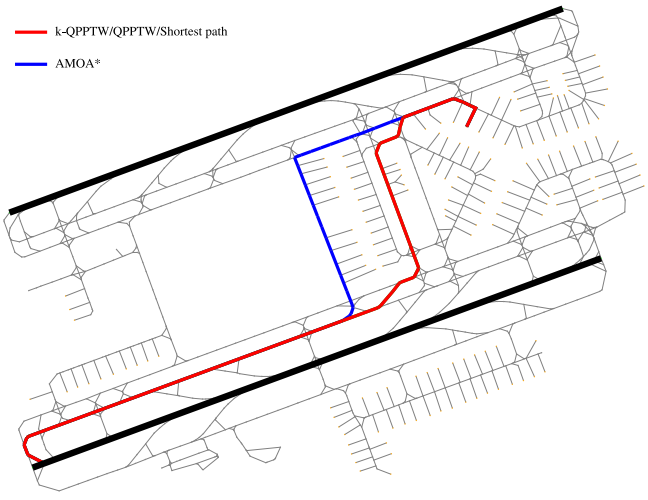

Fig. 8. Route of aircraft id 61.

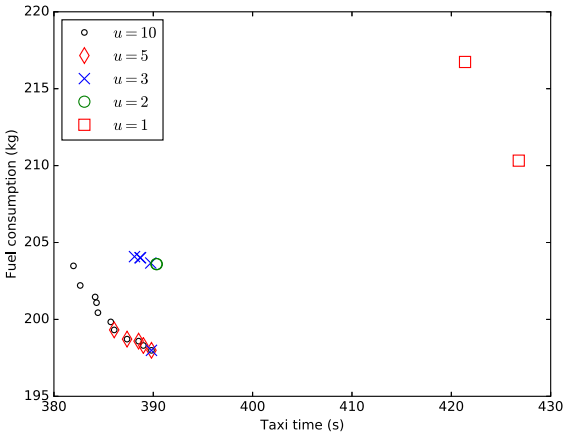

Fig. 9. Pareto front of solutions obtained by AMOA* with increasing u for aircraft id 321.

In the second case, we investigate the impact of increasing u on a single aircraft (id 321) from HKG instance. As mentioned in Section 3.3, considering a limited number of speed profiles affects the quality of solutions. In the following experiment, the multi-graph was reduced using the preferences. However, similar results are expected from using another multi-graph reduction method described in Section 3.3. Firstly, AMOA* with preferences was run using $H1$, $u = 1$ and $w^p = (0, M)$ for all aircraft preceding (id 321) in order to initiate the time windows. Then, AMOA* with preferences was run with varied $u = 1, 2, \ldots, 10$ for aircraft (id 321). The Pareto front of the obtained routes is shown in Fig. 9. As can be seen, with the increasing value of u, the algorithm found better solutions. When the value of u is small, the algorithm could not find efficient routes due to the tight time windows imposed by other aircraft. Larger values of u enable the algorithm to consider more speed profiles which may comply with the tight time windows, leading to better and more routes.

In the next section, a detailed analysis of factors affecting the performance of AMOA* is carried out on the complete instances of DOH, HKG and PEK.

4.4. Parameter analysis

Two factors are examined: the size of the search space and the heuristic function. Note that for experiments using PEK instance and heuristic function $H2$, only the first 50 aircraft were considered (denoted as PEK50). This is due to the size of the graph and the resulting long time to calculate $H2$ for all combinations of nodes and gates and runway exits. As only combinations of nodes and gates and runway exits used by the aircraft in the instance need to be calculated, by considering a limited number of aircraft, the number of gates/runway exits is capped at maximum 50. AMOA* with $holding$ not enabled is used in the following experiments without the loss of generality. The effect of the parameter $holding$ on the computational time was very small during the initial experiments and therefore not investigated further.

Firstly, experiments with the varying size of the search space are carried out. The size of the search space with a given graph G is determined by the size $p \times q$ of $C_{n,m}$ for each segment $s_{n,m}$. As described in Section 3.3, the size of the search space can be reduced by setting $u < p$ according to some reduction methods. Without the loss of generality, for the analysis of computational time the first u rows of $C_{n,m}$ are considered based on $w^P = (0, M)$ as the preference for ranking. From the set of nondominated routes, the route with minimum taxi time is selected for each aircraft. The heuristic function $H2$ was used during the experiments, as $H2$ provided the best results as will be described later. Table 5 details minimum, maximum and average computational time for routing a single aircraft with the increasing value of u. A-SMGCS specifies that the time to generate a route for a single aircraft should not exceed 10 s for on-line decision support (ICAO, 2004). The computational times quickly increase with larger values of u. The experiments marked with * were run on the HPC, which has a maximum limit of 10 days for a single run. The missing experiments could not be completed within that limit. The results show that only for $u = 1$ the average time is within 10 s limit for all instances, although the maximum time for PEK is more than doubled than this limit. The increase in computational time is caused by more alternatives in $OPEN$ to be expanded in Algorithm 2. In comparison, the average computational times for ASOA* with $H2$ did not exceed 1 s for DOH, HKG and PEK50 instances, demonstrating the value of multi-graph reduction when preference information is available and computational time requirement is stringent.

Fig. 10 shows a distribution of computational times for HKG instance with $u = 5$. As can be seen, the majority of runtimes is relatively fast. Only a smaller portion of aircraft had very long algorithm runtimes. Similar distributions were observed in other cases with different u.

Table 6 further illustrates the effect of u on the number of nondominated routes for each aircraft. With larger u, more nondominated solutions are found. For $u = 1$, the number of nondominated solutions is close to 1 and seems to increase with higher complexity of the airport taxiway network. For many aircraft only 1 optimal route is found, and if preference information

Table 5
Computational times for a single aircraft in seconds for $H2$ and varying u.

		DOH	HKG	PEK50
$u = 1$	min	<0.01	0.01	0.01
	max	0.14	5.71	21.76
	avg	0.03	1.07	2.47
$u = 2$	min	0.01	0.02	0.05
	max	2.16	71.93	280.37
	avg	0.19	6.87	24.48
$u = 3$	min	0.01	0.05	0.11*
	max	13.73	324.98	2439.91*
	avg	0.61	25.62	149.41*
$u = 5$	min	0.01	0.09*	0.29*
	max	29.12	8370.49*	9269.09*
	avg	1.73	252.90*	456.73*
$u = 10$	min	0.02	0.05*	–
	max	1283.48	28984.62*	–
	avg	24.81	1416.65*	–

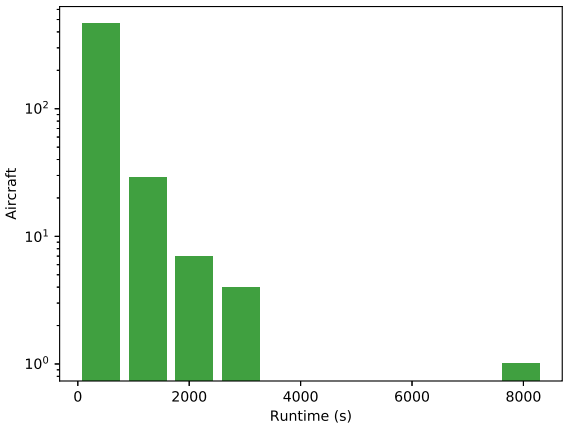

Fig. 10. Histogram of runtime for HKG and $u = 5$.

Table 6
Number of nondominated solutions.

		DOH	HKG	PEK50
$u = 1$	max	3	6	5
	avg	1.12	1.56	1.68
$u = 2$	max	28	32	58
	avg	6.51	6.82	8.22
$u = 3$	max	67	85	57
	avg	12.63	13.48	13.34
$u = 5$	max	198	214	62
	avg	24.77	25.34	21.04
$u = 10$	max	567	529	–
	avg	67.18	48.38	–

is available, the multi-objective search is equivalent to the single-objective search with preferences in those cases. This supports the hypothesis of low cardinality of the Pareto front with correlated objectives (Mote et al., 1991). Although taxi time and fuel consumption are conflicting for a single segment as demonstrated in Chen et al. (2016b), in general longer segments require longer taxi time, resulting in higher fuel consumption and therefore reduce the cardinality of the Pareto front.

The effect of u on the performances of the obtained results is further investigated. To enable comparison, the submatrix of costs for each edge and the selected route for each aircraft is selected according to preferences w^P and ranking as outlined in Section 3.3. The preferences are set to the same values as in Section 4.3. Table 7 shows savings of costs obtained with varying u, relative to

Table 7
Comparison of cost savings compared with $u = 1$.

	w^P	DOH	HKG	PEK	PEK50
$u = 2$	$(0, M)$	0.00%	0.00%	−0.04%	0.00%
	$(0.469, 0.71)$	0.34%	0.24%	−0.01%	0.00%
	$(0.469, 1.42)$	0.01%	0.36%	0.05%	0.02%
	$(M, 0)$	0.00%	0.17%	0.00%	0.00%
$u = 3$	$(0, M)$	0.00%	0.02%	0.00%	0.14%
	$(0.469, 0.71)$	0.34%	0.29%	0.33%	0.00%
	$(0.469, 1.42)$	0.05%	0.47%	0.07%	0.02%
	$(M, 0)$	0.00%	0.18%	0.01%	0.00%
$u = 5$	$(0, M)$	0.00%	−0.11%	0.37%	0.14%
	$(0.469, 0.71)$	0.49%	0.29%	–	0.00%
	$(0.469, 1.42)$	0.03%	0.55%	–	0.02%
	$(M, 0)$	0.01%	0.31%	–	0.00%
$u = 10$	$(0, M)$	0.17%	0.05%	–	–
	$(0.469, 0.71)$	0.51%	–	–	–
	$(0.469, 1.42)$	0.20%	–	–	–
	$(M, 0)$	−0.08%	–	–	

Table 8
Computational time analysis of H for u=1 in seconds.

		DOH	HKG	PEK	PEK50
$H0$	min	0.03	0.11	0.48	–
	max	0.17	7.88	62.71	–
	avg	0.08	2.27	18.25	–
$H1$	min	0.01	0.04	0.15	–
	max	0.15	5.94	49.72	–
	avg	0.05	1.47	7.55	–
$H2$	min	<0.01	0.01	–	0.01
	max	0.14	5.71	–	21.76
	avg	0.03	1.07	–	2.47

the base case with $u = 1$. Only experiments which could be completed within 10 days on the HPC are reported. Positive values refer to lower costs compared to the case with $u = 1$. As can be seen, larger values of u resulted in increasing but relatively small savings in most cases. In some cases as demonstrated in Section 4.3, experiments with larger u resulted in higher costs than the base case with $u = 1$. This seems in contradiction with the results reported in Section 4.3 for a single aircraft. However, further investigation reveals that this is caused by sequential routing of aircraft in AMOA*. In individual cases, AMOA* with larger u can find a route with better costs, however this may have adverse impact on the available route for the subsequent aircraft, resulting in larger total costs. Similar results were observed with the multi-graph reduction method based on evenly distributed solutions. It should be noted, that the multi-graph reduction method affected the computational time and in some cases the experiments with evenly distributed solutions could finish within the time limit where with the other reduction method could not. With increasing u, more solutions are found as documented in Table 6 but not necessarily with higher quality.

We can observe that higher values of u can be beneficial in individual cases as documented in Section 4.3, when a tight time window is on an edge which usually occurs during high traffic conditions. In such case, AMOA* with a small value of u makes a detour or holds the aircraft if no feasible speed profile exists for that edge. With larger u, more speed profiles can be considered to find feasible routes with lower costs compared to a detour. An example of such case will be described later. However, such benefit is limited for the examined instances and the sequential approach that AMOA* is currently following. A more detailed analysis of the effect of higher u on the quality of solutions is needed for high traffic instances as well as when a global approach (i.e. searching for sequence of aircraft and their route) is adopted in future work.

Next, different heuristic functions are examined for AMOA* with preferences with $u = 1$. Without the loss of generality, w^p was set to $(M, 0)$. Note that the heuristic function does not affect the number or cost of nondominated routes (under the admissibility condition as explained in Section 3.2). Only the computational time will be different. Table 8 details the computational time of AMOA* for each heuristic function. Note that the pre-calculation time is excluded. As expected, the computational times decrease with more accurate heuristic function.

4.5. Comparison with baseline approaches

In this section, AMOA* is compared with k-QPPTW, QPPTW and the Shortest path algorithms in terms of the costs of the resulting routes. AMOA* is intended to be used in a general case, when no preference information is specified beforehand. AMOA*, k-QPPTW,

Table 9
Taxi time and fuel consumption of routes for different algorithms.

Algorithm	w^P	DOH		HKG		PEK	
		obj_1	obj_2	obj_1	obj_2	obj_1	obj_2
AMOA* $u = 3$ hold	$(M, 0)$	32641	16081	130024	68848	89010	30027
	$(0.469, 0.71)$	33783	14584	132803	64256	90151	28255
	$(0.469, 1.42)$	34964	13970	135900	63002	91732	27528
	$(0, M)$	36125	13838	139381	62468	95650	27171
AMOA* $u = 3$	$(M, 0)$	33576	16251	131019	68803	89522	29917
	$(0.469, 0.71)$	34581	14562	134394	64389	91056	28225
	$(0.469, 1.42)$	35539	13970	137202	62807	92633	27490
	$(0, M)$	36885	13866	141202	62525	96080	27098
AMOA* $u = 5$ hold	$(M, 0)$	32636	16077	130023	68890	88994	29966
	$(0.469, 0.71)$	33710	14453	133163	63951	90299	27950
	$(0.469, 1.42)$	34377	14091	134861	63000	91093	27539
	$(0, M)$	36127	13833	139285	62414	95604	27170
AMOA* $u = 5$	$(M, 0)$	33571	16253	131062	68863	89503	29887
	$(0.469, 0.71)$	34612	14465	134633	63968	91208	27919
	$(0.469, 1.42)$	34995	14085	136152	62867	91980	27526
	$(0, M)$	36886	13860	141215	62524	96081	27107
k-QPPTW	$(M, 0)$	32472	15915	143856	72945	98742	32747
	$(0.469, 0.71)$	33306	14515	146117	68495	100871	30925
	$(0.469, 1.42)$	34251	14072	149008	67084	103126	30154
	$(0, M)$	35928	13839	156547	66497	109189	29444
QPPTW	$(M, 0)$	32522	15941	146568	74400	101493	33854
	$(0.469, 0.71)$	33306	14515	151021	69872	102814	32136
	$(0.469, 1.42)$	34251	14072	156621	68388	106104	31181
	$(0, M)$	36067	13864	161258	67434	113852	30289
shortest path	$(M, 0)$	33962	15941	141089	74011	97776	33964
	$(0.469, 0.71)$	34257	14335	143216	68300	98303	31539
	$(0.469, 1.42)$	34868	14033	145833	67260	99601	30884
	$(0, M)$	36993	13881	152979	67154	106996	30202

QPPTW and the Shortest path algorithms return a Pareto set of solutions for each aircraft. However, due to the sequential nature of the algorithms, one solution for each aircraft need to be chosen before the next aircraft can be processed. For the this purpose, the same preferences w^P were used as in Section 4.3. It should be noted, that the preferences are not used here for multi-graph reduction. Multi-graph reduction was applied with $u = 3$ and $u = 5$ and evenly distributed speed profiles. With $u = 3$, for each segment, two extreme and a middle speed profile in terms of the first objective were considered. With $u = 5$, the speed profiles were evenly distributed. Lower values of $u < 3$ seemed not to represent the Pareto front for each segment as seen in results in Section 4.3, while values $u > 5$ are too computationally demanding and therefore not reported here.

The heuristic function $H1$ was used. Table 9 shows the obtained results for obj_1 (s) and obj_2 (kg). For AMOA*, its variant with $holding$ enabled (indicated as AMOA* hold) is also investigated. It should be noted that obj_1 includes holding time and time if the aircraft was postponed. For AMOA* with $holding$ enabled obj_2 includes the fuel corresponding to holding. For AMOA* with $holding$ not enabled no extra fuel is considered, as the postponing happens at the start/end of the route. For departures this is achieved at the gate with engines turned off. For arrivals, it is assumed that postponing can be achieved before landing via ATC procedures and therefore not affecting taxiing costs. Tables 10 and 11 show the corresponding increase/decrease in the values of obj_1 and obj_2 for the solutions of AMOA* with $holding$ enabled with respect to AMOA* without holding, k-QPPTW, QPPTW and the Shortest path algorithm.

For HKG and PEK instances, the routes found by AMOA* hold are 6.8%–16.0% better in taxi time and 5.6%–13.0% better in fuel consumption for both $u = 3$ and $u = 5$ compared to the k-QPPTW, QPPTW and the Shortest path algorithms. These results confirm that k-QPPTW, QPPTW and the Shortest path algorithms could not efficiently search the search space due to limited number of explored routes. Furthermore, not considering real costs pertaining to speed profiles during the search further limits the chance of finding good solutions. For the DOH instance, AMOA* hold performed slightly worse in some cases (maximum 2.1%) than k-QPPTW, QPPTW and the Shortest path algorithm. Possible reasons are the following: (1) A limited number of possible routes exists due to the simple taxiway layout of DOH. Therefore, the optimal route is easy to be found by k-QPPTW, QPPTW and the Shortest path algorithm. (2) As u is low, only few speed profiles are considered during the search by AMOA* and the Pareto front is not covered adequately. Furthermore, if no feasible time windows exist for an edge, AMOA* will try to hold the aircraft or take a detour to find edges with feasible time windows. The holding and detour cause longer taxi time and higher fuel consumption. In contrast, k-QPPTW, QPPTW and the Shortest path algorithm have only a limited number of alternative routes available, hence they have to utilise the complete set of p speed profiles and postponing (if necessary) to comply with time windows. An example of a route with a detour found by AMOA* hold and a better route found by k-QPPTW for the same aircraft is shown in Fig. 11. However, the overall effect of u is relatively small as reported in Table 7. (3) Another reason for worse results is the sequential processing of aircraft,

Table 10
Increase/decrease in the values of obj_1 and obj_2 for the solutions of AMOA* hold with $u = 3$ with respect to the different algorithms.

Algorithm	w^P	DOH		HKG		PEK	
		obj_1	obj_2	obj_1	obj_2	obj_1	obj_2
AMOA* $u = 3$	$(M, 0)$	2.8%	1.0%	0.8%	−0.1%	0.6%	−0.4%
	$(0.469, 0.71)$	2.3%	−0.2%	1.2%	0.2%	1.0%	−0.1%
	$(0.469, 1.42)$	1.6%	0.0%	0.9%	−0.3%	1.0%	−0.1%
	$(0, M)$	2.1%	0.2%	1.3%	0.1%	0.4%	−0.3%
k-QPPTW	$(M, 0)$	−0.5%	−1.0%	9.6%	5.6%	9.9%	8.3%
	$(0.469, 0.71)$	−1.4%	−0.5%	9.1%	6.2%	10.6%	8.6%
	$(0.469, 1.42)$	−2.1%	0.7%	8.8%	6.1%	11.0%	8.7%
	$(0, M)$	−0.6%	0.0%	11.0%	6.1%	12.4%	7.7%
QPPTW	$(M, 0)$	−0.4%	−0.9%	11.3%	7.5%	12.3%	11.3%
	$(0.469, 0.71)$	−1.4%	−0.5%	12.1%	8.0%	12.3%	12.1%
	$(0.469, 1.42)$	−2.1%	0.7%	13.2%	7.9%	13.5%	11.7%
	$(0, M)$	−0.2%	0.2%	13.6%	7.4%	16.0%	10.3%
Shortest path	$(M, 0)$	3.9%	−0.9%	7.8%	7.0%	9.0%	11.6%
	$(0.469, 0.71)$	1.4%	−1.7%	7.3%	5.9%	8.3%	10.4%
	$(0.469, 1.42)$	−0.3%	0.4%	6.8%	6.3%	7.9%	10.9%
	$(0, M)$	2.3%	0.3%	8.9%	7.0%	10.6%	10.0%

Table 11
Increase/decrease in the values of obj_1 and obj_2 for the solutions of AMOA* hold with $u = 5$ with respect to the different algorithms.

Algorithm	w^P	DOH		HKG		PEK	
		obj_1	obj_2	obj_1	obj_2	obj_1	obj_2
AMOA* $u = 5$	$(M, 0)$	2.8%	1.1%	0.8%	−0.0%	0.6%	−0.3%
	$(0.469, 0.71)$	2.6%	0.1%	1.1%	0.0%	1.0%	−0.1%
	$(0.469, 1.42)$	1.8%	−0.0%	0.9%	−0.2%	1.0%	−0.1%
	$(0, M)$	2.1%	0.2%	1.4%	0.2%	0.5%	−0.2%
k-QPPTW	$(M, 0)$	−0.5%	−1.0%	9.6%	5.6%	9.9%	8.5%
	$(0.469, 0.71)$	−1.2%	0.4%	8.9%	6.6%	10.5%	9.6%
	$(0.469, 1.42)$	−0.4%	−0.1%	9.5%	6.1%	11.7%	8.7%
	$(0, M)$	−0.6%	0.0%	11.0%	6.1%	12.4%	7.7%
QPPTW	$(M, 0)$	−0.4%	−0.9%	11.3%	7.4%	12.3%	11.5%
	$(0.469, 0.71)$	−1.2%	0.4%	11.8%	8.5%	12.2%	13.0%
	$(0.469, 1.42)$	−0.4%	−0.1%	13.9%	7.9%	14.1%	11.7%
	$(0, M)$	−0.2%	0.2%	13.6%	7.4%	16.0%	10.3%
Shortest path	$(M, 0)$	3.9%	−0.9%	7.8%	6.9%	9.0%	11.8%
	$(0.469, 0.71)$	1.6%	−0.8%	7.0%	6.4%	8.1%	11.4%
	$(0.469, 1.42)$	1.4%	−0.4%	7.5%	6.3%	8.5%	10.8%
	$(0, M)$	2.3%	0.3%	9.0%	7.1%	10.6%	10.0%

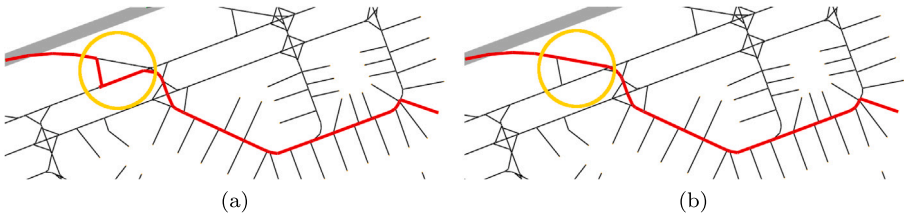

Fig. 11. Route of aircraft id 270 (a) with a detour (AMOA* hold with preferences), (b) (k-QPPTW) with differences encircled.

where one aircraft with better route can randomly cause a longer route for subsequent aircraft. For the two variants of AMOA* with and without *holding* enabled, both algorithms achieved similar results. In some cases, AMOA* with *holding* performed better, in others worse. These results show that it is difficult to determine the best holding strategy. If *holding* is enabled, potentially better routes can be found without a detour. On the other hand, holding at an intermediate node incur additional costs while if *holding* is not enabled the aircraft will be postponed without a fuel penalty. The future work could focus on the search for the best allocation of the holding times.

Table 12
Comparison of holding time (s).

Algorithm	w^P	DOH	HKG	PEK
AMOA* $u = 3$ hold	$(M, 0)$	86	365	184
	(0.469, 0.71)	98	327	288
	(0.469, 1.42)	94	436	320
	$(0, M)$	119	458	406
AMOA* $u = 3$	$(M, 0)$	600	1140	780
	(0.469, 0.71)	660	1320	900
	(0.469, 1.42)	480	1260	840
	$(0, M)$	420	1140	660
AMOA* $u = 5$ hold	$(M, 0)$	83	366	174
	(0.469, 0.71)	86	398	266
	(0.469, 1.42)	81	414	246
	$(0, M)$	122	430	399
AMOA* $u = 5$	$(M, 0)$	600	1140	780
	(0.469, 0.71)	660	1320	900
	(0.469, 1.42)	480	1260	840
	$(0, M)$	420	1140	660
k-QPPTW	$(M, 0)$	0	7500	5160
	(0.469, 0.71)	0	7020	6120
	(0.469, 1.42)	0	6900	6540
	$(0, M)$	0	8580	6840
QPPTW	$(M, 0)$	0	8520	6540
	(0.469, 0.71)	0	9960	6540
	(0.469, 1.42)	0	11700	7680
	$(0, M)$	0	11160	9060
shortest path	$(M, 0)$	1440	4260	3600
	(0.469, 0.71)	960	3720	2960
	(0.469, 1.42)	960	4140	2880
	$(0, M)$	840	3180	2400

The results obtained for AMOA* with $u = 3$ and $u = 5$ are very similar and only slightly better for $u = 5$. This suggests that the coverage of the Pareto front is sufficient for $u = 3$ for most of the given data instances. Also, the effect of higher u is small as presented in Table 7.

Since k-QPPTW explored $k = 3$ routes, better solutions were found than those of QPPTW and Shortest path algorithms, which explored only one route. Interestingly, QPPTW found routes with higher costs than the Shortest path algorithm. In contrast to the Shortest path algorithm, QPPTW takes into account time windows during the search. Therefore, QPPTW may take a detour if no time windows are available, whereas the Shortest path algorithm postpones the aircraft (extends t_i) instead.

Table 12 shows the total holding time, summed for all aircraft. For AMOA* with *holding* enabled aircraft can be held at an intermediate node. For AMOA* without *holding* enabled and the baseline algorithms, if no time windows are available for the aircraft to traverse an edge, the start time t_i of aircraft a_i is postponed by 60 s (Line 5 in Algorithm 1 and Line 10 in Algorithm 4).

k-QPPTW and QPPTW have a higher total holding time than the other algorithms. As described in Section 4.2, k-QPPTW and QPPTW search for the route assuming constant speeds, and the time for edge traversal is calculated according to this speed. However, this is often not realistic and therefore the real speed profiles can violate the time windows. As a result, postponing is applied to resolve the violation. As expected, k-QPPTW has a lower total holding time than QPPTW, due to a higher number of routes explored during the search and thus a higher chance of finding a feasible route.

Surprisingly, the Shortest path algorithm achieved a shorter total holding time than k-QPPTW and QPPTW. This is despite the fact that the Shortest path algorithm does not take into account time windows during the search for the route. The results suggest that assuming constant speeds during the search in k-QPPTW and QPPTW mislead the algorithms into routes with infeasible time windows. By not considering time windows and using constant speeds, the Shortest path algorithm could find more feasible routes on average than k-QPPTW and QPPTW. The shortest holding time is achieved by AMOA* for both *holding* variants with better values for *holding* enabled. The better results compared to other algorithms are achieved by efficient holding at intermediate nodes, considering speed profiles and the ability to take detours. An exception is the DOH instance with a simple taxiway layout. As explained above, $u < p$ results in either a detour or holding time for AMOA*, in contrast to k-QPPTW and QPPTW which can consider p speed profiles to resolve time window violations.

In terms of computational time, AMOA* with $u = 3$ and $u = 5$ could not find routes within 10 s on average as documented in Table 5. In the next section, we explore how the search can be carried out faster when preferences are formulated.

Table 13
Taxi time and fuel consumption of routes for different algorithms.

Algorithm	w^P	DOH		HKG		PEK	
		obj_1	obj_2	obj_1	obj_2	obj_1	obj_2
AMOA* /w pref. hold	$(M, 0)$	32644	16116	130063	69139	89051	30184
	$(0.469, 0.71)$	33480	14431	132385	64002	90003	28153
	$(0.469, 1.42)$	34102	14133	134080	63099	90894	27681
	$(0, M)$	36234	13868	139999	62563	95800	27205
AMOA* /w pref.	$(M, 0)$	33678	16288	131499	69214	90192	30199
	$(0.469, 0.71)$	34403	14431	134001	64120	91066	28159
	$(0.469, 1.42)$	34899	14099	135956	63371	91947	27677
	$(0, M)$	37157	13848	141529	62503	96404	27143
ASOA*	$(M, 0)$	33678	16288	131656	69297	90014	30170
	$(0.469, 0.71)$	34403	14431	134234	64218	91204	28255
	$(0.469, 1.42)$	35003	14128	135928	63383	92080	27696
	$(0, M)$	37221	13848	141678	62567	96523	27158
ASOA* hold	$(M, 0)$	32644	16116	130081	69235	89183	30208
	$(0.469, 0.71)$	33480	14431	132299	63966	90021	28159
	$(0.469, 1.42)$	34102	14138	134074	63157	91045	27711
	$(0, M)$	36256	13874	139999	62563	95891	27217

Table 14
Comparison of holding time (s).

Algorithm	w^P	DOH	HKG	PEK
AMOA* /w pref. hold	$(M, 0)$	109	587	313
	$(0.469, 0.71)$	114	606	514
	$(0.469, 1.42)$	115	617	504
	$(0, M)$	116	771	472
AMOA* /w pref.	$(M, 0)$	660	1320	900
	$(0.469, 0.71)$	660	1260	1020
	$(0.469, 1.42)$	660	1200	900
	$(0, M)$	720	1320	660
ASOA* hold	$(M, 0)$	109	616	446
	$(0.469, 0.71)$	114	676	556
	$(0.469, 1.42)$	124	710	663
	$(0, M)$	128	771	521
ASOA*	$(M, 0)$	660	1320	960
	$(0.469, 0.71)$	660	1200	960
	$(0.469, 1.42)$	660	1200	960
	$(0, M)$	780	1320	720

4.6. Search with preferences

As explained in Section 3.3, when preference information is available, a multi-graph can be reduced into a simple graph and a multi-objective search to a single-objective one. The same preferences were used as in Section 4.3. For the multi-objective search, AMOA* with preferences and $u = 1$ was used, whereas ASOA* was run for the single-objective search. It should be noted, when the multi-graph is transformed into the simple graph with single objectives on its edges, k-QPPTW and QPPTW algorithm are expected to give the same performance as ASOA*. If k-QPPTW and QPPTW search on a simple graph with aggregated costs according to preferences, only one shortest route exists which is found with $k = 1$.

Tables 13 and 14 show the obtained results in terms of objectives and holding time, respectively. As described in Section 3.3, transformation into a single-objective search with ASOA* can compromise the quality of solutions. However, AMOA* with preferences performed very similarly to ASOA* in terms of the quality of the obtained routes. Small differences are caused by the nonadditivity effect explained in Section 2.2 to which ASOA* is more prone due to the absent Pareto dominance check. Together with the results for a single aircraft in Section 4.3, it can be observed that: (1) AMOA* with preferences can simultaneously find set of nondominated solutions, which is particularly evident when $u > 1$. Also, as indicated in the results for a single aircraft, AMOA* with preferences with $u > 1$ can find better solutions than ASOA* in some cases. (2) When $u = 1$, AMOA* with preferences performs similarly to ASOA*. Furthermore, ASOA* is much faster than AMOA* with preferences. To conclude, if preferences are available and the decision maker is interested only in one solution, ASOA* can be considered as a better option than AMOA* with preferences.

5. Conclusions

In this paper, a multi-objective routing and scheduling algorithm AMOA* for airport ground movement was introduced to find optimal or near optimal routes for a fixed sequence of aircraft in terms of taxi time and fuel consumption taking into account realistic speed profiles. Two heuristic functions were proposed in order to accelerate the search. The ground movement problem was modelled as a multi-graph, which proved to be computationally challenging on larger instances of HKG and PEK and higher number of parallel segments. As a solution, two multi-graph reduction methods were proposed, one for a general case and another one when preferences for the search in terms of economic costs for each objective are available. AMOA* with reduced multi-graph found routes on medium (HKG) and large (PEK) instances which are 6.8%–16.0% better in taxi time and 5.6%–13.0% better in fuel consumption compared to the previously used baseline algorithms. In some cases, the proposed algorithm generated a maximum 2.1% worse route in both objectives than the baseline algorithms due to no available time windows for all the speed profiles. The results showed only a limited effect on optimality of found solutions with more parallel edges in the multi-graph while computational times increased quickly. Although the computational time still remained high for the reduced multi-graph, the results suggest that if preferences are available and the decision maker is interested only in one solution, a single-objective search can find a high quality solution faster than the multi-objective AMOA* with preferences.

For the future work, several areas deserve more attention:

1. The routing algorithm could incorporate more accurate speed profiles incorporating nonlinear jet engine behaviour and uncertainty due to pilots following the speed profiles.
2. As shown in this paper, there are different solutions for cases when no time windows are feasible for the speed profiles retrieved from the database. The algorithm can postpone the start time of the aircraft, hold it at an intermediate node, take a detour or generate another speed profile in real time such as in (Zhang et al., 2018). In fact, there is trade-off between different holding approaches, a longer route and an alternative speed profile meeting the time window which should be investigated.
3. Results suggest that a lower value of u is sufficient in most cases to provide faster computational time without compromising the quality of solutions. However, the best value of u differs depending on the conditions. When edges have tight time windows such as during high traffic conditions, higher values of u could improve the quality of solutions. This could be utilised in construction of the multi-graph where value of u (i.e. number of parallel edges) could be varied for different aircraft or scenarios.
4. Computational times showed a wide distribution, where a few aircraft needed extremely long times. Further research could investigate the attributes of these cases in order to predict them before the search. Suitable search space reduction methods could be employed for those aircraft.
5. In this paper, a sequential approach was implemented. However, the sequencing of aircraft as demonstrated in Ravizza et al. (2013a) has an impact on the best taxi time/fuel consumption that can be obtained.
6. An alternative approach, such as the one based on metaheuristics is needed for the larger number of parallel segments in the multi-graph to be computationally tractable.
7. Faster computational time could be achieved by preprocessing techniques, such as multi-objective shortcuts (Delling and Wagner, 2009).
8. It is believed that knowledge about the multi-graph search for this problem could be utilised for other similar problems such as the time-constrained vehicle routing problem (Garaix et al., 2010; Lai et al., 2016).

CRediT authorship contribution statement

Michal Weiszer: Conceptualization, Methodology, Software, Validation, Writing - original draft. **Edmund K. Burke:** Writing - review & editing, Supervision, Funding acquisition. **Jun Chen:** Conceptualization, Methodology, Writing - review & editing, Supervision, Funding acquisition.

Acknowledgements

This work is supported in part by the Engineering and Physical Sciences Research Council (EP/N029496/1, EP/N029496/2, EP/N029356/1, EP/N029577/1, EP/N029577/2). The airport layout data sets are available here: https://doi.org/10.5281/zenodo.3967562. The aircraft movement data arising from FR24 cannot be shared due to licencing restrictions. This research utilised Queen Mary's Apocrita HPC facility, supported by QMUL Research-IT, United Kingdom. https://doi.org/10.5281/zenodo.438045

References

Adacher, L., Flamini, M., Romano, E., 2018. Airport ground movement problem: Minimization of delay and pollution emission. IEEE Trans. Intell. Transp. Syst. PP (99), 1–10. http://dx.doi.org/10.1109/TITS.2017.2788798.

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

Benlic, U., Brownlee, A.E., Burke, E.K., 2016. Heuristic search for the coupled runway sequencing and taxiway routing problem. Transp. Res. C 71, 333–355. http://dx.doi.org/10.1016/j.trc.2016.08.004.

Brownlee, A.E.I., Atkin, J.A.D., Woodward, J.A.W., Benlic, U., Burke, E.K., 2014. Airport ground movement: Real world data sets & approaches to handling uncertainty. In: Proc. PATAT, York, UK.

Brownlee, A.E., Weiszer, M., Chen, J., Ravizza, S., Woodward, J.R., Burke, E.K., 2018. A fuzzy approach to addressing uncertainty in airport ground movement optimisation. Transp. Res. C 92, 150–175. http://dx.doi.org/10.1016/j.trc.2018.04.020, http://www.sciencedirect.com/science/article/pii/S0968090X18305461.

Carraway, R.L., Morin, T.L., Moskowitz, H., 1990. European J. Oper. Res. 44 (1), 95–104.

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. In: Lecture Notes in Computer Science, 4163, Springer Berlin Heidelberg, pp. 280–293. http://dx.doi.org/10.1007/11823940_22.

Chen, J., Weiszer, M., Locatelli, G., Ravizza, S., Atkin, J., Stewart, P., Burke, E.K., 2016a. Towards a more realistic, cost effective and greener ground movement through active routing: A multi-objective shortest path approach. IEEE Trans. Intell. Transp. Syst..

Chen, J., Weiszer, M., Stewart, P., Shabani, M., 2016b. Toward a more realistic, cost-effective, and greener ground movement through active routing - part I: Optimal speed profile generation. IEEE Trans. Intell. Transp. Syst. 17 (5), 1196–1209. http://dx.doi.org/10.1109/TITS.2015.2477350.

Chitra, C., Subbaraj, P., 2010. A nondominated sorting genetic algorithm for shortest path routing problem. Int. J. Comput. Sci. 5 (1), 55–63.

Clare, G., Richards, A., 2011. Optimization of taxiway routing and runway scheduling. Int. Transp. Syst. IEEE Trans. 12 (4), 1000–1013. http://dx.doi.org/10.1109/TITS.2011.2131650.

Climaco, J.C.N., Martins, E.Q.V., 1982. A bicriterion shortest path algorithm. European J. Oper. Res. 11 (4), 399–404.

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: International Symposium on Experimental Algorithms. Springer, pp. 125–136.

Dijkstra, E.W., 1959. A note on two problems in connexion with graphs. Numer. Math. 1 (1), 269–271.

Evertse, C., Visser, H., 2017. Real-time airport surface movement planning: Minimizing aircraft emissions. Transp. Res. C 79, 224–241. http://dx.doi.org/10.1016/j.trc.2017.03.018.

Garaix, T., Artigues, C., Feillet, D., Josselin, D., 2010. Vehicle routing problems with alternative paths: An application to on-demand transportation. European J. Oper. Res. 204 (1), 62–75. http://dx.doi.org/10.1016/j.ejor.2009.10.002, http://www.sciencedirect.com/science/article/pii/S0377221709007048.

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.

Guépet, J., Briant, O., Gayon, J.-P., Acuna-Agost, R., 2017. Integration of aircraft ground movements and runway operations. Transp. Res. E: Logist. Transp. Rev. 104, 131–149. http://dx.doi.org/10.1016/j.tre.2017.05.002.

Hart, P.E., Nilsson, N.J., Raphael, B., 1968. A formal basis for the heuristic determination of minimum cost paths. IEEE Trans. Syst. Sci. Cybern. 4 (2), 100–107.

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

Jiang, Y., Liao, Z., Zhang, H., 2013. A collaborative optimization model for ground taxi based on aircraft priority. Math. Probl. Eng. 2013, 1–9. http://dx.doi.org/10.1155/2013/854364.

Khadilkar, H., Balakrishnan, H., 2012. Estimation of aircraft taxi fuel burn using flight data recorder archives. Transp. Res. D 17 (7), 532–537. http://dx.doi.org/10.1016/j.trd.2012.06.005.

Lai, D.S., Demirag, O.C., Leung, J.M., 2016. A tabu search heuristic for the heterogeneous vehicle routing problem on a multigraph. Transp. Res. E: Logist. Transp. Rev. 86, 32–52.

Lesire, C., 2010. An iterative A* algorithm for planning of airport ground movements. In: 19th European Conference on Artificial Intelligence (ECAI)/6th Conference on Prestigious Applications of Intelligent Systems (PAIS), Lisbon, Portugal, August 16-20, 2010. http://dx.doi.org/10.3233/978-1-60750-606-5-413.

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

Mandow, L., Mandow, L., De la Cruz, J., De la Cruz, J., et al., 2005. A new approach to multiobjective a* search. In: Proceedings of the 19th International Joint Conference on Artificial Intelligence. Morgan Kaufmann Publishers Inc., pp. 218–223.

Marín, A., Codina, E., 2008. Network design: Taxi planning. Ann. Oper. Res. 157 (1), 135–151. http://dx.doi.org/10.1007/s10479-007-0194-0.

Martins, E.Q.V., 1984. On a multicriteria shortest path problem. European J. Oper. Res. 16 (2), 236–245.

Mori, R., 2013. Aircraft ground-taxiing model for congested airport using cellular automata.. IEEE Trans. Intell. Transp. Syst. 14 (1), 180–188, http://dblp.uni-trier.de/db/journals/tits/tits14.html#Mori13. http://dx.doi.org/10.1109/TITS.2012.2208188. http://www.bibsonomy.org/bibtex/291e77730fff0b16a5591b514ef964827/dblp.

Mote, J., Murthy, I., Olson, D.L., 1991. A parametric approach to solving bicriterion shortest path problems. European J. Oper. Res. 53 (1), 81–92.

Nikoleris, T., Gupta, G., Kistler, M., 2011. Detailed estimation of fuel consumption and emissions during aircraft taxi operations at dallas/fort worth international airport. Transp. Res. D 16 (4), 302–308. http://dx.doi.org/10.1016/j.trd.2011.01.007.

Pangilinan, J.M.A., Janssens, G., 2007. Evolutionary algorithms for the multi-objective shortest path problem. Int. J Math. Comput. Phys. Electr. Comput. Eng. 1 (1), 7–12.

Raith, A., Ehrgott, M., 2009. A comparison of solution strategies for biobjective shortest path problems. Comput. Oper. Res. 36 (4), 1299–1331.

Ravizza, S., Atkin, J.A., Burke, E.K., 2013a. A more realistic approach for airport ground movement optimisation with stand holding. J. Sched. 1–14. http://dx.doi.org/10.1007/s10951-013-0323-3.

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 Transp. 5 (1–2), 25–40. http://dx.doi.org/10.1007/s12469-013-0060-1.

Roling, P.C., Visser, H.G., 2008. Optimal airport surface traffic planning using mixed-integer linear programming. Int. J. Aerosp. Eng. vol. 2008, 11.

Samà, M., D'Ariano, A., Corman, F., Pacciarelli, D., 2017. Coordination of scheduling decisions in the management of airport airspace and taxiway operations. In: Papers Selected for the 22nd International Symposium on Transportation and Traffic Theory Chicago, Illinois, USA, 24-26 July, 2017.. Transp. Res. Procedia 23, 246–262. http://dx.doi.org/10.1016/j.trpro.2017.05.015.

Skriver, A., Andersen, K., 2000. A label correcting approach for solving bicriterion shortest-path problems. Comput. Oper. Res. 27 (6), 507–524. http://dx.doi.org/10.1016/S0305-0548(99)00037-4, http://www.sciencedirect.com/science/article/pii/S0305054899000374.

Stewart, B.S., White, C.C., 1991. Multiobjective a*. J. ACM 38 (4), 775–814.

Tung, C.T., Chew, K.L., 1992. A multicriteria pareto-optimal path algorithm. European J. Oper. Res. 62 (2), 203–209.

Weiszer, M., Chen, J., Locatelli, G., 2015a. An integrated optimisation approach to airport ground operations to foster sustainability in the aviation sector. Appl. Energy 157, 567–582. http://dx.doi.org/10.1016/j.apenergy.2015.04.039, http://www.sciencedirect.com/science/article/pii/S0306261915004948.

Weiszer, M., Chen, J., Stewart, P., 2015b. A real-time active routing approach via a database for airport surface movement. Transp. Res. C 58, Part A, 127–145. http://dx.doi.org/10.1016/j.trc.2015.07.011, http://www.sciencedirect.com/science/article/pii/S0968090X1500251X.

Weiszer, M., Chen, J., Stewart, P., Zhang, X., 2018. Preference-based evolutionary algorithm for airport surface operations. Transp. Res. C 91, 296–316. http://dx.doi.org/10.1016/j.trc.2018.04.008, http://www.sciencedirect.com/science/article/pii/S0968090X18304650.

Zhang, T., Ding, M., Zuo, H., Chen, J., Weiszer, M., Qian, X., Burke, E.K., 2018. An online speed profile generation approach for efficient airport ground movement. Transp. Res. C 93, 256–272.

Zhou, H., Jiang, X., 2015. Research on taxiway path optimization based on conflict detection. PLoS One 10 (7), e0134522.