Routing and Scheduling in Multigraphs with Time Constraints - A Memetic Approach for Airport Ground Movement

Lilla Beke, Lourdes Uribe, Adriana Lara, Carlos A. Coello Coello, *Fellow, IEEE*, Michal Weiszer, Edmund K. Burke, Jun Chen

***Abstract*—Routing and scheduling problems with increasingly realistic modelling approaches often entail the consideration of multiple objectives, time constraints, and modelling the system as a multigraph. This detailed modelling approach has increased computational complexity and may also lead to violation of the additivity property of the costs. In the worst scenario, increased complexity makes the problem intractable for exact algorithms. Even when the problem is solvable, exact algorithms may not provide solutions within the given time budget, and the found solutions are not guaranteed to be optimal due to the additivity property violation. Approximate solution methods become more suitable in this case. This paper focuses on one particular real-world application, the Airport Ground Movement Problem, where both time constraints and parallel arcs are involved. We introduce a novel Memetic Algorithm for Routing in Multigraphs with Time constraints (MARMT) and present a comprehensive study of its different variants based on diverse genetic representation methods. We propose a local search operator that enhances search efficiency and effectiveness. MARMT is tested on real data based on two airports of different sizes. Our results show that MARMT does not suffer from the non-additivity property problem as it outperforms the state-of-the-art exact algorithm when allowed to converge. When a time budget of 10 seconds is imposed on MARMT, it is able to provide solutions with quality comparable (within 1-5% degradation) to the ones given by the exact algorithm with respect to the aggregated objective values. MARMT can be adapted for other applications, such as train operations.**

***Index Terms*—Multiobjective routing and scheduling, Multigraphs, Airport ground movement, Memetic algorithm, Time windows.**

I. Introduction

THE efficiency of transportation systems is essential for satisfying the rising demand from industry and commerce while balancing economic cost and environmental impact. Many transportation problems can be formulated as variations of the Shortest Path Problem [1]. In these cases, there are often conflicting objectives, such as travel time and energy consumption (including both fossil and sustainable sources), which means that it is not always possible to find a single optimal solution. Instead, the aim is to identify a set of solution paths with non-dominated costs that also meet certain time constraints.

To solve routing problems, the infrastructure in a transportation system is often described through a graph [2]. Nodes in the graph correspond to significant locations in the system, such as junctions, stations, and starting and ending points. In a simple graph model, a directed arc between two nodes implies a direct link between the corresponding places in the system following the indicated direction. A series of connected arcs (a path) in the graph corresponds to a route.

Optimising airport ground operations exemplifies the multiobjective routing and scheduling problems with time constraints and can be viewed as a special case of the energy-efficient driving problem. A taxiing aircraft is more fuel efficient at certain speeds and on routes with fewer turns. For this reason, there is a trade-off between taxi time and fuel consumption [3]. The multigraph modelling approach was found to produce better solutions than the simple graph approach in [4]. A similar trade-off is often observed when routing different vehicles [5], suggesting a wider applicability for algorithms developed for airport ground movement.

Vehicle speed is a decision variable in many real-world applications. In the presence of time constraints, the choice of speed can affect the feasibility of the solutions. Therefore, it is important to manage routing and scheduling in an integrated way. The multigraph representation makes this possible by including the choice of speed profiles as discrete decision variables. A series of connected arcs in a multigraph can represent a trajectory, describing the movement of the vehicle in terms of time (hence scheduling) and space (hence routing), whereas a route only describes the movement in space. The need for an integrated routing and scheduling approach applies to the airport ground movement problem [6], maritime transportation [7], train operations [8] and the transportation of hazardous materials [9]. The multigraph modelling approach has also been employed for the vehicle routing problem [10] and multimodal transportation [11].

The Multiobjective Shortest Path Problem (MSPP) is NP-hard even on simple graphs without time constraints [12]. The multigraph approach further increases the size of the search space and its associated computational complexity.

L. Beke, J. Chen and M. Weiszer are with Queen Mary University of London, London, UK. e-mail: l.beke@qmul.ac.uk, jun.chen@qmul.ac.uk, m.weiszer@qmul.ac.uk

L. Uribe and A. Lara are with ESFM, Instituto Politécnico Nacional, Mexico City, Mexico. e-mail: luriber@ipn.mx, alaral@ipn.mx

Carlos A. Coello Coello is with the Department of Computer Science, CINVESTAV-IPN (Evolutionary Computation Group), México, D.F. 07300, México. He is also with the Faculty of Excellence, School of Engineering and Sciences, Tecnologico de Monterrey, Monterrey, N.L., Mexico

E. K. Burke was with the University of Leicester, UK. He is now with Bangor University, Wales, UK. e-mail: ekb@bangor.ac.uk

Corresponding Author: J. Chen

Manuscript received ..., ...; revised August ..., XXXX.

Empirical results support the high computational complexity of the airport ground movement problem. The best performing algorithm for this problem, AMOA* [4], could not provide solutions for some aircraft in a reasonable time frame, while earlier algorithms [3], [13], [14] showed lower solution quality because of their use of less detailed modelling approaches. In practical settings, finding a good representation of the Pareto front in a given time budget is often important. Metaheuristics are popular for this reason compared to exact approaches. In addition, as the costs of the same arc can be different depending on its predecessors, the costs no longer satisfy the additivity property (detailed in Section III-D). Most exact approaches rely on such a property to prune dominated solutions while maintaining optimality. Metaheuristics are able to find better solutions that exact algorithms are not able to find when the additivity property does not hold.

Genetic algorithms (GA) are metaheuristics that have been widely applied to the MSPP [15], [16], [17], [18], and to multimodal transport problems [19], [11]. Previous work explored different representation schemes for the multigraph MSPP in artificial problem instances without time constraints [20]. Time constraints call for the incorporation of additional constraint management techniques. When using constraints, it is widely accepted that GAs require a considerable amount of resources to calculate a suitable approximation of the solution. A natural way to improve the convergence properties of GAs is to include a local search procedure [21], [22], [23]. This procedure explores the search space around a specific candidate solution. By employing this mechanism, the local information of the selected solution is exploited, giving a new and improved solution. Then, this improved solution is incorporated into the population.

In light of these observations, we propose a Memetic Algorithm for Routing in Multigraphs with Time constraints (MARMT) for the Airport Ground Movement problem, and the family of problems it represents, with variants based on different solution encoding schemes. All variants of MARMT are based on non-dominated sorting [24]. Our focus is on the design of a local search operator and a constraint handling scheme, in addition to a comparison of the different encoding schemes used for MSPPs. Our results are also compared to a state-of-art exact (enumerative) solution approach [4].

The main contributions include the following: (i) MARMT is developed for the multigraph MSPP with time windows. Three different genetic representation methods are adapted to the problem and compared. MARMT is shown to handle a higher number of parallel arcs and the non-additivity property of the costs more effectively than the enumerative approach. (ii) A local search operator is proposed based on single objective search with a varied weight vector for aggregating different objectives. Integrating the local search operator into the metaheuristic significantly improves solution quality as measured by multiple quality indicators. (iii) MARMT is tailored to a representative real-world application (the airport ground movement problem), and is tested on real-world data. (iv) Constraints related to aircraft movements and time windows are incorporated into the algorithm. A mixed approach is proposed for constraint-handling based on fitness penalties and preserving feasibility.

The remainder of this paper is structured as follows. The background is presented in Section II. The airport ground movement problem is described in Section III. The proposed representations, operators and constraint-handling schemes are described in Section IV. Implementation details are given in Section V and our results are discussed in Section VI. Finally, concluding comments are presented in Section VII.

II. Background

A. Solution approaches for MSPP

1) Enumerative algorithms: The three main categories of MSPP algorithms are: ranking methods, two-phase methods and labeling methods. Ranking methods [25] for the bi-objective case generate a specified number of shortest paths in non-decreasing order regarding one of the objectives, and eliminate dominated solutions. Two-phase methods [26], [27] first list solutions that can be found by aggregating objectives, and then explore a restricted search space in the second phase to find the remaining Pareto optimal solutions. Labeling methods, including label setting [28] and label correcting [29] represent a multiobjective generalisation of the single objective approach inspired by Dijkstra [30]. The efficiency of the above approaches has been compared empirically in [27], where labeling methods and two-phase methods were found to be the best in most cases.

Extensions of labeling algorithms based on the A* algorithm [31] such as The New Approach to Multiobjective A* (NAMOA*) [32] are able to make use of heuristic information and accelerate the optimisation process for the MSPP, while still finding the whole Pareto front, assuming additive costs.

The above approaches are not guaranteed to find optimal solutions if the costs are non-additive, or if time constraints are present. In both of these cases, a partial solution that is dominated by some other partial solutions is discarded, even though it might have turned out to be part of a global Pareto optimal solution. In the case of time constraints it might be impossible to complete the dominating partial path without violating time constraints, or satisfying the time constraints might entail additional costs. The case of non-additivity is explained in Section III-D.

There is a lack of studies of the general case of MSPP with time constraints. In [33], the four reviewed studies about the shortest path problem with time windows [34], [35], [36], [37] are all from the last century. Moreover, these studies are single objective, and a single finite time window was assigned for each node in the network, which is a great simplification of the general problem. Examples of ranking and labeling methods proposed for the MSPP with time constraints for the airport ground movement problem are reviewed in Section II-B.

2) Metaheuristic algorithms: Several studies applied GAs to shortest path problems with various representation methods, including direct variable length [38], direct fixed length [39], random keys [40] and integer-valued priority [41] representations. In general, it is hard to determine which one of these representations is more favourable than the others, as echoed in [20] that suitability depends on the instance at

hand. The representation scheme determines the search space to be explored and the available evolutionary operators for exploration. Therefore, the choice of the representation method can influence the effectiveness of the search [42].

The direct variable length representation [38] has been employed for the MSPP by multiple authors [16], [43], [17]. A chromosome based on this representation lists nodes of a solution path directly. Its greatest advantage is the one-to-one mapping from solution paths to chromosomes, which avoids creating unnecessary plateaus in the search space. Its disadvantage is the possibility of loop formation in crossover and that for some pairs of parents, the crossover operator might not be able to produce any novel candidates.

The two priority based representations are the integer-valued priority representation [44], [41] and random keys [45]. The random keys representation employs floating-point numbers as priorities. In both representations, a path is encoded by assigning a priority value for each of the nodes in the graph. The path can be decoded from the priority values by starting at the origin node and each time moving to the neighbouring node with the highest priority that has not yet been visited. The main advantage of priority based representations is that any priority values can be decoded to a path, and that crossover can be applied to any pair of parents. A disadvantage is that these representations offer one-to-n mapping, thereby forming plateaus in the search space. The random keys representation has a higher ambiguity than the Integer valued priority representation, suggesting larger plateaus.

The direct fixed length representation specifies the next node to visit at every node. The path is decoded by following the pointers to neighbouring nodes from the origin node. The length of the chromosomes is equal to the number of nodes in the graph. Consequently, this is also a one-to-n mapping, with generally less ambiguity than the priority based representations.

The above representations have been adapted for the multigraph MSPP in our previous work [20], with further adaptation required for the airport ground movement problem. Without time constraints, we found that different representations are best for different artificial problem instances, depending on the network type. Therefore, it is worthwhile to further investigate multiple representations for constrained problems.

Constraint handling for multiobjective evolutionary algorithms is an active area of research, with most studies focused on balancing the search between the feasible and infeasible regions, and having as the main source of difficulty the high numbers of objectives and/or constraints [46]. Penalty functions are the simplest and perhaps the most widely applied methods. They can be sufficient for multiobjective problems with fewer constraints [47]. However, they are thought to be less suitable for handling a high number of constraints, because tuning the penalty function is difficult. For combinatorial problems, preserving feasibility and repair mechanisms are also popular choices to limit the search space. Therefore, a mixed approach is proposed in this paper.

Memetic algorithms (MAs) complement the evolutionary process with a local search process [21], [22]. MAs have been a popular extension of GAs, which aims to avoid premature convergence and guide the population towards promising areas of the search space. An MA approach was proposed for the dynamic shortest path problem in simple graphs in [19]. In local search, all possible alternative partial routes were enumerated that might replace a single arc in a route, and the one that dominated the highest number of other alternative routes was chosen. The disadvantage of this approach is that it only replaces a single arc, and the number of alternative routes might be very high, especially in a multigraph.

B. Airport ground movement problem

The airport ground movement problem is concerned with routing and scheduling aircraft between gates and runways in an efficient and safe way. Airports are often overloaded. Multiple departing and arriving aircraft are on the taxiways at the same time, resulting in a complex and interconnected transportation system. Airport ground movement efficiency can be evaluated according to multiple objectives. The two most important are taxi time and fuel consumption, although other objectives such as emissions can also be considered [6].

Studies concerning the ground movement problem can be separated into two main categories: the sequential approach and the global approach. In the sequential approach, aircraft are routed in the order of their starting times, where the trajectory of the already routed aircraft needs to be respected by later aircraft. The global approach, on the other hand, considers the order of the aircraft as a decision variable, and usually assigns routes to aircraft from a predetermined set of routes in an attempt to ensure that the complexity of the problem is manageable. In this paper, the sequential approach is considered.

Earlier studies [13], [14] suffered from multiple limitations, such as considering only a single objective and assuming a constant speed for calculating traversal times. Single objective approaches cannot provide the available trade-offs in a single run. Realistic calculation of the traversal times is essential to provide the decision maker with accurate information and to allow good conformance during the execution stage.

A multiobjective approach (k-QPPTW) was studied in [3]. However, a decomposition method was applied to separate the routing and scheduling aspects of the problem. Realistic speed profiles are only considered for the scheduling component, while constant speeds are assumed for the routing component. Thus, only a limited number of routes are being explored for the scheduling component, which compromises solution quality compared to an integrated approach. Significant improvements were achieved regarding both taxi time and fuel consumption by Chen et. al. with the Active Routing trajectory-based ground movement operations framework [48], [6], by managing routing and scheduling in an integrated way, with realistic speed profiles.

Weiszer et. al. adapted the NAMOA* algorithm [32] for solving the ground movement problem. The introduced algorithm, AMOA* provided 5-16% improvements for the objective values on the considered test data compared to other baseline algorithms. This improvement can be attributed to the integrated routing and scheduling and using AMOA* instead

of the k-shortest path algorithm. However, in some cases, especially for larger airports and for a higher number of parallel arcs, the running times of AMOA* can be unacceptable. Multigraph reduction was hence proposed [4] to decrease the search-space, with some compromise on solution quality. AMOA* also suffers from the problem of non-additivity.

As pointed out in [4], a metaheuristic solution approach can scale better to a higher number of parallel arcs. Furthermore, there is no requirement for the costs to satisfy the non-additivity property [49] (see Section III-D).

C. Other real-world applications

The multigraph MSPP is a relevant problem related to many real-world transportation systems. Some of these have a heavier routing component, such as multimodal transportation problems; others have a heavier scheduling component, such as urban rail transit. The common features of such problems are the presence of multiple objectives, the availability of alternative trajectories between the same two points and the interactions between different vehicles, such that the optimal solutions for individual vehicles do not necessarily result in system level optimality. Interactions can be modelled through time constraints.

One of the problems where the multigraph model has shown to be valuable is time-constrained vehicle routing. Using a multigraph model for an on-demand transportation problem reduced the associated costs compared to a simple graph model [10]. A similar approach was followed by multiple authors in vehicle routing problems [50], [51].

Optimising speed profiles and timetables in an integrated way in urban rail transit can also be conceptualised through a multigraph, providing significant energy savings [8]. In urban rail transit, vehicles interact not only through inflicting time constraints on each other, but through regenerative braking, which entails synchronization of the accelerating/braking actions.

Optimal speed control of individual electric vehicles taking into account queues at intersections is studied in [52]. It is pointed out in [52] that optimising for individual vehicles might compromise system level efficiency. However, this was not investigated. For the system level study, a multigraph approach can be used, where alternative speed profiles are included for each vehicle for each leg of its route.

In marine transportation, the speed of a ship is optimised concerning fuel price and travel time [53]. It has also been shown that the optimal route depends on the optimised objective [54], suggesting that maritime transportation problems can also be modelled through a multigraph. Routing and speed decision problems for fleets of ships are a recent area of research [7], where a similar routing and scheduling framework such as the one proposed in this paper might be of great use.

Multimodal transportation problems [11], [55] and ride-sharing problems [56], [57] concern routing passengers or goods in a network where multiple modes of transport are available for the same leg of a route. The multigraph representation is natural to such problems. Travel time and economic cost are usually relevant objectives. Time constraints stem from timetables, which can be adjusted to target system level optimality (e.g., balancing congestion and customer demands).

Efficiently solving the multigraph MSPP with time windows is relevant for all the above applications, as there is a benefit in formulating them as multigraphs. Time constraints are also often included due to other vehicles moving in the same transportation system, traffic signals or contracts. Therefore, MARMT can be adapted to these problems, by specifying the application specific parts of the problem: the graph that describes the transportation system, the conditions of feasible solutions and the objective functions.

III. Airport ground movement as a combinatorial optimisation problem

The ground movement problem is decomposed into a series of MSPPs on multigraphs by the framework introduced in [6]. Realistic speed profiles are precomputed for certain sections of taxiways based on their geometry, called segments (defined in Section III-C). The speed profiles and the corresponding costs are stored in a database, saving computational time [58]. Trajectories for each aircraft can then be defined by consecutive segments with a specified speed profile between the origin (v_O) and destination (v_D) nodes, or equivalently, by specifying a path in the multigraph. The physical constraints of aircraft manoeuvring such as the maximum speed and acceleration rate are handled by the speed profile generation algorithm [58].

A. Sequential routing of aircraft

Aircraft are routed on a first come first served basis sequentially, as described in Algorithm 1 [4]. The corresponding notations are explained in Table I. In Line 3, a set of non-dominated solutions Θ_i is found by procedure $Route$, which can be based on any MSPP solver algorithm. Here, we consider AMOA* and MARMT (see Section IV). In Line 5, Aircraft are held at the gate for 1 min before $Route$ is reattempted if no solutions are found. We do not consider holding during taxiing.

Algorithm 1 Sequential routing of aircraft

```
1: Sort AircraftSequence according to t_i
2: for all aircraft_i ∈ AircraftSequence do
3:     Θ_i ← Route(aircraft_i)
4:     if Θ_i is empty then
5:         t_i ← t_i + 60s {1 min postponement}
6:         Go to line 3
7:     end if
8:     θ ← Preferred solution from Θ_i
9:     Reserve route θ and adjust corresponding time constraints
10: end for
```

Even though only a single trajectory is realised by the current aircraft, it is important to find the whole Pareto front or a good approximation of it. This ensures that the Decision Maker (DM) gets accurate information about the available trade-offs. In general, the most suitable solution is identified by the DM through an economic lens, taking into account conflicting interests among different stakeholders and the current operational situation [6]. Our primary interest is

TABLE I
NOTATIONS.

Notation	Description	Notation	Description
$aircraft_i$	The i^{th} aircraft	$G = (V, A)$	Speed profile graph (multigraph), $V \subset V_0$
t_i	Start time of $aircraft_i$	v_O	Origin node
θ	A trajectory	v_D	Destination node
Θ_i	Set of feasible trajectories with non-dominated cost vectors found for $aircraft_i$	u	Number of considered speed profiles in multigraph reduction
$\theta(nodes, indices)$	The trajectory defined by $nodes$ and $indices$	$(v, w)^k \in A$	The k^{th} arc between nodes v and w in G
w_1, w_2	Weights of the first and second objective when choosing a trajectory for $aircraft_i$ from Θ_i	$I(e, (v, w))$	The set of indices k, such that speed profile $(v, w)^k$ is a valid continuation of the route r with last edge e
$pred((v, v_{i+1}), \theta)$	Predecessor edge of segment (v, v_{i+1}) in θ	$c_{(v,w)^k}$=(c_1, c_2)	Cost vector associated with speed profile $(v, w)^k$
$Route$	The routing procedure that finds trajectories	c_1	Cost component associated with taxi time
$G_0 = (V_0, E)$	Layout graph (simple graph)	c_2	Cost component associated with fuel consumption
$e_i \in E$	An edge in the layout graph	$C(\theta)$	Sum of cost vectors associated with trajectory θ
$\mathcal{F}_e = \{(t_{e,i,start}, t_{e,i,end}) \mid 0 < i <\mid \mathcal{F}_e \mid\}$	Set of time windows assigned to edge e	M	Priority based chromosome. For node v, $M_{1,v}$ encodes the priority value and $M_{2,v}$ encodes parallel arc index

to solve the routing problem for each aircraft efficiently. For this reason, we use a simple strategy to simulate the role of a DM. Out of Θ_i, the realised trajectory is chosen according to a weighted sum of the costs, such that $w_2 = 1 - w_1$.

The airport ground movement problem for a given aircraft can be described by two graphs, one depicting the layout of the airport and the other one depicting all possible speed profiles for a given aircraft. These two graphs are described below.

B. The layout graph

The *layout graph* contains the geographical information about all available taxiways in the airport. The layout graph is a directed graph $G_0 = (V_0, E)$, where the set of nodes V_0 represent gates, stands, runway exits, taxiway intersections and intermediate points. Intermediate points are distributed in such a way that taxiways between two nodes in the layout graph are at most as long as the minimum safe separation of the aircraft. This minimum safe separation is set to 60m [4].

In line with the established terminology for ground movement operations, the sections of taxiways between two nodes in G_0 are called *edges* $E = \{e_1, e_2, ..., e_{|E|}\}$. To avoid confusion, in this paper we refer to arcs of graphs when we use the term in general and reserve the term "edge" only for the airport layout graph.

Edges are used to govern the scheduling component of the problem as multiple aircraft move on the airport ground at the same time. Avoiding conflict between aircraft is ensured by (1) allowing at most one aircraft at a time on each edge and (2) allowing no aircraft on edges that are conflicting with an occupied edge at any time. The set of conflicting edges with edge e consists of e', such that the distance of e and e' is smaller than the minimum safe separation (as measured along taxiways). To keep track of the occupation of the edges, a set of time windows ($\mathcal{F}_e$) are assigned to each edge. Time windows correspond to time intervals when the edge is not occupied and not conflicting with occupied edges.

C. The speed profile graph

Before turning, aircraft generally slow down, and then accelerate after turning. Thus, it makes sense to group sequences of edges depending on their geometry. For this reason, speed profiles are modelled as *straight* and *turning segments* as defined in [59], [3]. An edge belongs to a turning segment if its angle with the previous edge in the trajectory (*predecessor edge*) is above 30 degrees. Otherwise, it belongs to a straight segment. Sequential edges of the same type are grouped together. The segments are generated in such a way as to cover all possible edge sequences in the layout graph [58].

The speed profile graph $G = (V, A)$ stores information about the pre-computed efficient speed profiles for all segments. For the same segment, multiple alternative speed profiles are possible. Therefore, G is a multigraph . The nodes of G are the endpoints of segments, $V \subset V_0$, $V = \{1, 2, ...|V|\}$. Arcs in G are associated with a sequence of edges in G_0. The arcs $(v, w)^k \in A$ are defined by their endpoints $v, w \in V$ and a parallel arc index $1 \leq k \leq |A_{(v,w)}|$. Arcs imply speed profiles, and thus the predecessor edge to the segment (v, w) in a given trajectory θ affects which speed profiles out of $\{(v, w)^1, ...(v, w)^{|A_{(v,w)}|}\}$ are available in θ. The set of indices of speed profiles between nodes v and w that can follow a given predecessor edge, e, are denoted by $I(e, (v, w))$.

There is a cost-vector associated with each speed profile $c_{(v,w)^k} = (c_1, c_2)$, which describes the taxi time (c_1) and fuel consumption (c_2). Speed profiles of the same type (straight or turning) for the same segment can be thought of as a cost matrix, which includes non-dominated cost-vectors as its rows.

The number of alternative speed profiles considered for each segment greatly influences the size of the search space. For this reason, multigraph reduction techniques are introduced in [4] to reduce the number of speed profiles from the database. In this paper, we employ the multigraph reduction technique that keeps the first u speed profiles. It is important to note that the number of parallel arcs in G can sometimes be different than u for some pairs of nodes. This is because, in rare cases, two segments might connect the same two nodes, but use different edges. An example of this is shown in Figure 1. In this case, there will be u speed profiles for each straight (angles below 30 degrees) segment between the same nodes. All of those speed profiles show up in G, which leads to $2u$ parallel arcs between some pairs of nodes.

The speed profiles and costs also depend on the weight

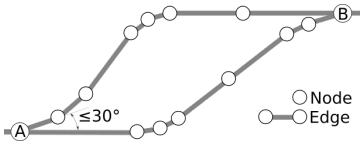


Fig. 1. Source of inhomogeneous numbers of parallel arcs in G.

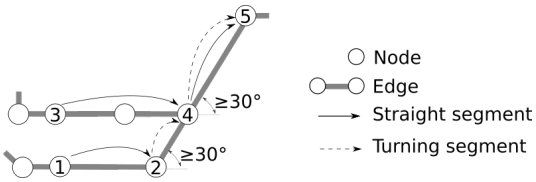


Fig. 2. An illustration of the non-additivity. The segment 4-5 is a turning segment, when approached via segment 3-4, and is a straight segment when approached via segment 2-4. Depending on the direction, the cost-vector of segment 4-5 is different; a turning segment is more costly. It is possible that up to node 4, the trajectory via node 3 dominates the trajectory via node 2, while up to node 5, the trajectory via node 2 dominates.

category of aircraft. If a segment is the first or last one in a trajectory, it implies greater acceleration or deceleration than if it is in the middle. Therefore, speed profile graphs differ based on the weight category of the aircraft. However, within the same weight category, the difference is small and can be quickly modified before routing each aircraft.

D. Non-additivity of costs

Straight or turning speed profiles can be associated with the same segment. Which speed profiles are appropriate is determined by the predecessor edges of partial trajectories. This leads to costs being non-additive. Labelling approaches for the MSPP only find all possible solutions when the costs satisfy the additivity property because they eliminate dominated partial solutions. Metaheuristic approaches can easily overcome this challenge. Figure 2 shows a detailed example. It was found that 1.72% of parallel arc pairs exhibit the non-additivity property [4].

E. Problem description for a single aircraft

The focus of this paper is solving the multigraph MSPP with time windows for a single aircraft. The description of this problem is provided next.

The inputs to the multigraph MSPP with time windows are: (1) the airport instance (G, G_0, including $pred$, I, v_O, v_D, c), (2) the current state of the time windows ($\mathcal{F}$), (3) the aircraft being routed ($aircraft_i$), and (4) the number of speed profiles (u) considered in multigraph reduction.

A solution, trajectory $\theta \in \Theta_i$ for $aircraft_i$ can be specified as a path in G (multigraph). In general, θ has the form:

$$(v_1, v_2)^{k_1}, (v_2, v_3)^{k_2}, \ldots, (v_{|\theta|-1}, v_{|\theta|})^{k_{|\theta|-1}}, \quad (1)$$

$$\text{s.t.} \quad (v_j, v_{j+1})^{k_j} \in A, \quad \forall j \in (1, 2, \ldots, |\theta| - 1) \quad (2)$$

The cost-vector of a feasible trajectory θ can be calculated according to Equation (3).

$$C(\theta) = \sum_{(v_j, v_{j+1})^k \in \theta} c_{(v_j, v_{j+1})^{k_j}}. \quad (3)$$

We are looking for a set of feasible trajectories Θ_i with Pareto optimal costs. A solution θ_1 is said to be Pareto-optimal if another solution θ_2 does not exist, such that θ_2 is at least as good as θ_1 according to both objectives and better according to at least one objective.

However, not all paths in the multigraph correspond to a feasible trajectory. The following constraints need to be satisfied:

1) Satisfy predecessor edges. $k_j \in I(e, (v_j, v_{j+1})), \ \forall j \in (1, 2, \ldots, |\theta| - 1)$, where $e = pred((v_j, v_{j+1}), \theta)$
2) Satisfy time windows. For each edge e in trajectory θ, there exists a time window $tw \in \mathcal{F}_e$, such that the traversal period of edge e according to θ falls into tw.
3) Not containing any loops in the layout graph. $v_i \neq v_j$ if $v_i, v_j \in \theta$ for all $v_i, v_j \in G_0$.
4) Comply with the origin node and the destination node. $v_O = v_1$ and $v_D = v_{|\theta|}$.

Constraint 1 ensures that the trajectory describes a realistic speed profile in terms of acceleration and deceleration. Constraint 2 ensures the trajectory complies with the time windows of each edge. Constraint 3 prohibits routes with loops in G_0, as a practical consideration. Although loops could be a way of achieving compliance with time windows, holding before taxiing or during taxiing is generally a better choice. Note that this is a stronger statement than $v_1, v_2, v_3 \ldots, v_{|\theta|-1}, v_{|\theta|}$ being all distinct, which only concerns the end nodes of segments. Constraint 4 ensures that the end points of the trajectory are as required. Constraints 1 and 2 are highly specific to the ground movement problem. In other applications, time constraints might be defined for nodes or for arcs of the multigraph.

IV. Our proposed memetic algorithm: MARMT

MARMT is presented in this section with three variants based on one direct and two indirect representation methods. MARMT is based on non-dominated sorting and binary tournament selection with crowded-comparison [24]. However, it can be easily modified to use other multiobjective evolutionary strategies [60], [61]. The operators are performed in the following order: mutation, crossover and local search. Thus, the diversity of the population is increased before crossover and the results of local search always reach evaluation without further modification. We do not investigate the direct fixed length representation. The specified next node, in general, cannot be guaranteed to be a valid continuation of the trajectory. Therefore, evolutionary operators are expected to often lead to invalid offspring. In comparison, in priority based representations, the priorities specify an order between the neighbours of any given node. If the neighbour with the highest priority is not a valid continuation, the neighbour with the second highest priority can be used, and so on.

A. Search based on direct variable length representation

The direct variable length representation specifies a trajectory by listing node IDs (v) to form a path in the speed profile graph (G) and the corresponding parallel arc indices (k_i) in the following form: $[v_1, k_1, v_2, k_2, ...v_{|\theta|}]$.

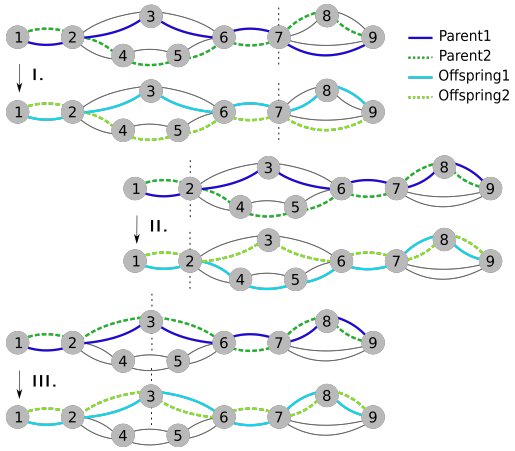


Fig. 3. Direct crossover in multigraphs. Example I. shows an ideal crossover akin to the simple graph case, with novel node sequences in the offspring. Example II. shows the lack of an ideal crossing site with distinct parent node sequences. Example III. shows parents with identical node sequences.

Algorithm 2 CrossoverOutline(parent1, parent2)

Input: P_1 := parent1, P_2 := parent2
Output: $ch1, ch2$:= Offspring
1: **if** node sequences of P_1, P_2 are identical **then**
2: $site \leftarrow$ randomly chosen node from P_1
3: $ch1, ch2 \leftarrow$ Recombine($P_1, P_2, site$)
4: **else**
5: $sites \leftarrow$ crossing sites
6: $idealSites \leftarrow$ ideal crossing sites out of $sites$
7: **for all** $site$ in $idealSites$ **do**
8: $ch1, ch2 \leftarrow$ Recombine($P_1, P_2, site$)
9: **if** at least one child is feasible **then**
10: **break**
11: **end if**
12: **end for**
13: **if** there is a crossing site adjacent to v_D **then**
14: $site \leftarrow$ crossing site adjacent to v_D
15: $ch1, ch2 \leftarrow$ Recombine($P_1, P_2, site$)
16: **else if** there is a crossing site adjacent to v_O **then**
17: $site \leftarrow$ crossing site adjacent to v_O
18: $ch1, ch2 \leftarrow$ Recombine($P_1, P_2, site$)
19: **else**
20: $ch1, ch2 \leftarrow P_1, P_2$
21: **end if**
22: **end if**
23: **return** $ch1, ch2$

1) Decoding: Decoding a candidate to a trajectory in G is straightforward, with the exception of handling Constraint 3. Once the decoding reaches a speed profile that includes an edge with an end node already in the decoded part of the trajectory, the decoding is stopped to avoid a loop. The already decoded part is returned, which will be penalised for not reaching the destination node in fitness evaluation (see Section IV-D). Repair in general would be difficult, because the search process operates at the level of segments (G), while repeated nodes appear at the level of edges (G_0).

2) Mutation: A node in the candidate path is chosen at random. Then, part of the chromosome is regenerated by a random walk starting from the chosen node, taking predecessor edges into account.

3) Crossover: A modified one point crossover is adopted [38], which is illustrated in Figure 3. The crossover operator is based on finding crossing sites between two parents. A crossing site is one node or a list of sequential nodes that appear in both parents other than v_O or v_D. If there are differences in the node sequences of the parents both before and after a given crossing site, the node sequences of the offspring can be different from both parents. We call these crossing sites *ideal crossing sites*.

In simple graph problems, crossover can only be conducted if there are ideal crossing sites. Figure 3 (I) shows an example of an ideal crossing site, which does not appear in (II) and (III). In multigraph problems, the offspring may be different from the parents, as long as the parents have differences in k_i (see Figure 3 (III)). Algorithm 2 describes the different cases for the crossover process. The cases are based on the comparison of the two parents, which determines how the crossing site is chosen. When the parents are identical, a crossover is not possible. If only the node sequences are identical, a crossover can be performed at a randomly chosen site (Line 3), as shown in Figure 3 (III). If the node sequence of the parents differs, ideal crossing sites are tried first (Line 7). If none of the ideal crossing sites produced offspring satisfying Constraint 1, other crossing sites are considered. There can be at most two of these, one adjacent to v_O and one to v_D (Lines 18 and 15. respectively). In this application, a repair mechanism aimed at eliminating loops is not enough to ensure feasibility of the offspring, as Constraint (1) can still be violated. Therefore, when a potential crossing site is found in Algorithm 2, the next step is to execute the modified one-point crossover according to Algorithm 3 to remove any loops from the offspring and check for violation of Constraint 1. If a violation occurs, the part up to the violation site is returned as a new candidate (Line 6 of Algorithm 3), which will be penalised in its fitness assignment accordingly. Note that in applications without Constraint 1, removing loops is sufficient.

Algorithm 3 Recombine(*parent1, parent2, site*)

Input: P_1 := parent1, P_2 := parent2, $site$
Output: $ch1, ch2$:= Offspring
1: $ch1 \leftarrow P_1$ up to $site$, and P_2 from $site$
2: $ch2 \leftarrow P_2$ up to $site$, and P_1 from $site$
3: Remove loops from $ch1, ch2$ {regarding speed profile graph}
4: **for all** speed profile in ch_i for $i = 1, 2$ **do**
5: **if** speed profile violates predecessor edges **then**
6: Remove nodes from the ch_i starting from the end node of the speed profile
7: **end if**
8: **end for**
9: **return** $ch1, ch2$

4) Local search procedure: Local search operators in memetic algorithms improve some candidate solutions in the population with some probability in each iteration. The improved candidates can give a jump start to the evolutionary process through crossover with other individuals. Local search is costly in terms of computational resources and running time, and might lead to premature convergence. Therefore, it should be employed infrequently. The local search operator employed in this work is based on Dijkstra's algorithm [30], as single objective shortest path problems can be solved efficiently by exact algorithms. The complexity of Dijkstra's algorithm is $O(E * log(n))$, where n is the number of nodes and E is the number of arcs. In the local search, the objective values are aggregated to a single objective with a random weight. A single objective shortest path respecting time windows is

found between two randomly chosen nodes in a candidate, and the newly found partial solution replaces the part between the two nodes of the original candidate. The part of the trajectory being overwritten cannot be longer than a certain percentage l_{rel} of the whole length measured in hopcounts. We also avoid local search in the case of trajectories shorter than a certain minimum length l_{min}. If no solution is found by the local search, the initial part of the candidate is returned, up to the node from where the local search is started. The solution will be highly penalised for not reaching the destination in the fitness evaluation, as explained in Section IV-D. The above local search process can be easily incorporated into the direct representation but not into other representations, as explained in Section IV-B.

B. Search based on priority based representations

Priority based representations encode paths indirectly as a priority value assigned to each node. For integer-valued priority representation, chromosomes are permutations of the first n integers. For random keys representation, priority values are floating-point numbers. The priorities only encode paths, to encode trajectories, parallel arc indices are also needed. As seen in Section III-C, $|I(e,(v,w)|$ depends on nodes v and w and the predecessor edge. Unlike the direct representation, the offspring might include segments that are not in any of the parents. The parallel arc index inherited from a parent may be higher than the number of available parallel arcs for a given segment, and thus the solution would be infeasible. For this reason, we use an indirect way of encoding parallel arcs so that the decoded parallel arc indices will always be feasible [20]. A chromosome for the priority based representations for multigraph problems can be conceptualised as a 2 by n matrix M. $M_{1,v}$ is the priority value for node v. $M_{2,v}$ is a real number between 0 and 1 that determines the parallel arc to be used when leaving node v. The index of the parallel arc to be used when leaving node v towards node w with predecessor edge e, can be calculated as $\lfloor M_{2,v} * |I(e,(v,w)|\rfloor + 1$.

1) Decoding: The decoding process iteratively finds the neighbour with the highest priority among the ones satisfying Constraints 1 and 3, and adds them to the decoded trajectory. The process is detailed in Algorithm 4. The loop in Lines 4-15 first identifies the *allowed* neighbour list (Line 6) that consists of the nodes that are (1) directly reachable from the last node of the already decoded part of the trajectory, (2) do not introduce loops in G_0 and (3) satisfy predecessor edges. If there are no such nodes, the already decoded part is returned (Line 8). Otherwise, the node with the highest priority is identified, and the lists *nodes* and *indices* defining the trajectory are updated (Lines 12 and 15).

2) Mutation: Insertion mutation is employed for both priority based representations. A randomly picked gene (a random column in M) is removed from the chromosome and inserted back at a new random locus. The loci of genes between the place of removal and insertion change accordingly. The process is illustrated in Figure 4.

3) Crossover: For the Integer priority representation, Weight Mapping Crossover (WMX) [41] is adopted, which has been proposed specifically for the MSPP. In the integer priority representation, chromosomes are always a permutation of the first n integers. Therefore, the original one point crossover cannot be used. WMX reorders part of the priorities in a chromosome according to the order of the corresponding priority values in another chromosome. For the random keys representation, 2-point crossover is used, as it was found to be the most efficient in [62]. WMX and 2-point crossover operates on priority values, the first row of M. We perform 2-point crossover on the columns of M, so that the priority value and the parallel arc for a given node is derived from the same parent. In WMX, if the priority of a node changes, the parallel arc indicator also changes as illustrated in Figure 5.

4) Integrating local search to priority based representation: Dijkstra's algorithm operates on a direct representation of the graph. It cannot be used directly with priority based representations. Therefore, priority-based chromosomes are decoded before local search, and converted back afterwards. Algorithm 5 takes the node sequence (*nodes*) and the index sequence (*indices*) as its input, both of which define the trajectory. It returns a priority-based chromosome, a 2 by n matrix, M, that encodes the trajectory specified in the input. In Lines 3-8, the priorities of the nodes that appear in the trajectory are set. These priorities are increasing from the destination node towards the origin node. This ensures that in the decoding process, the neighbour with the highest priority is the next node in the trajectory for each node, as all other

Algorithm 4 DecodingPriorityBased(M)

Input: $M :=$ priority based chromosome
Output: $nodes, indices$
1: $nodes \leftarrow$ list with a single element: v_O
2: $indices \leftarrow$ empty list
3: $predEdge \leftarrow$ None
4: **while** Last element of $nodes \neq v_D$ **do**
5: $neighbours \leftarrow$ Set of nodes reachable from last element in $nodes$ in G
6: $allowed \leftarrow$ Set of nodes in $neighbours \notin nodes$, {fulfill the $predEdge$, and do not introduce loops in G_0}
7: **if** $allowed = \emptyset$ **then**
8: **return** $nodes, indices$
9: **else**
10: $prevNode \leftarrow$ Last element of $nodes$
11: $nextNode \leftarrow$ Node with maximum priority in $allowed$ according to M
12: $nodes = nodes \cup nextNode$
13: $x \leftarrow| I(predEdge, prevNode, nextNode) |$
14: $currentIndex \leftarrow \lfloor M_{2,prevNode} * x \rfloor + 1$
15: $indices = indices \cup currentIndex$
16: $predEdge = pred((prevNode, nextNode), \theta(nodes, indices))$
17: **end if**
18: **end while**
19: **return** $nodes, indices$

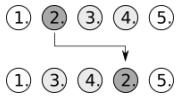


Fig. 4. Illustration of insertion mutation.

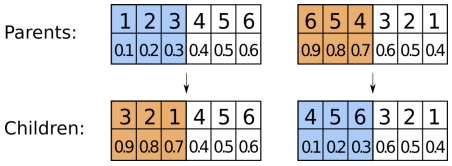


Fig. 5. Illustration of WMX for the matrix chromosome.

unvisited nodes in the graph have lower priorities. In Lines 9-13, the remainder of the genes use the lower priority values, and random parallel indices. Converting to random keys is similar, apart from the priority values being floating point numbers.

Algorithm 5 directToPriority($nodes, indices$)

Input: $nodes, indices$
Output: $M :=$ priority based chromosome
1: $n \leftarrow$ number of nodes in G
2: $priority \leftarrow n$
3: **for** $i \leftarrow 1$ to $\mid nodes \mid$ **do**
4: $\quad M_{1,nodes[i]} \leftarrow priority$
5: $\quad predEdge \leftarrow$ predecessor edge for segment $nodes[i], nodes[i+1]$
6: $\quad M_{2,nodes[i]} \leftarrow \frac{indices[i]}{|I(predEdge,(nodes[i],nodes[i+1]))|}$
7: $\quad priority \leftarrow priority - 1$
8: **end for**
9: **for** $j \leftarrow 1$ to n **do**
10: $\quad$ **if** $M_{1,j}$ is not yet specified **then**
11: $\quad\quad M_{1,j} \leftarrow priority$
12: $\quad\quad M_{2,j} \leftarrow$ Random floating-point number $\in [0, 1]$
13: $\quad\quad priority \leftarrow priority - 1$
14: $\quad$ **end if**
15: **end for**
16: **return** M

C. Initialisation

Heuristic initialisation is used from our previous work [62]. Initial solutions are generated semi-randomly through priority values that specify a random walk with a bias to get closer to v_D. The process starts from the random keys representation, as a chromosome from this representation is readily convertible to the other two. $M_{2,v}$ are initialised randomly between 0 and 1. Each node $v \in G$ is assigned a priority value according to:

$$M_{1,v} = -h(v, v_D, G) + \tau, \qquad \tau \in (0, \tau_{max}). \tag{4}$$

$M_{1,v}$ depends on the hopcount (the minimum number of edges in a path) from the destination node and a parameter τ_{max}. The hopcount between nodes v and v_D in G is denoted by $h(v, v_D, G)$, and τ_{max} represents the maximum value of the randomisation coefficient τ. The likelihood of detours appearing in the decoded paths is controlled by the parameter τ_{max}. The greater the value of τ_{max}, the more random the priorities are, and the less prominent the effect of the heuristic initialisation is compared to random initialisation. The hopcount information can be calculated beforehand, as it uses a simple graph and does not rely on the cost vectors and time constraints. Therefore, it does not increase computational time.

D. Fitness function and constraint handling

For any valid solution, fitness is defined over the objective functions to minimise. For invalid solutions, trajectories that do not reach v_D, or violate time windows, we apply static penalties [63]. The severity of the penalty, and how much the violation of each constraint contributes to it, is controlled through weights. The fitness assignment including penalties is described in Algorithm 6. The cost vector of trajectory θ is calculated according to Equation (3) (Line 1). To get the fitness value of θ, the penalties need to be added for violation of Constraints 2 and 4. The maximum value of any cost component in any speed profiles in G, $maxCost$ is used to establish the magnitude of the penalties (Line 2). When not reaching v_D, the level of violation is measured as the minimum distance of θ and v_D (Line 3). When violating time windows, the level of violation is measured as the number of time-windows violated (Line 4). The level of constraint violation and $maxCost$ are multiplied to give the base penalty, p_0.

We set up four weights respectively for the two objectives and two constraints, $\alpha_1, \alpha_2, \alpha_3, \alpha_4$. These weights for the penalty function were tuned by irace [64], and their values are set to be $1, 7, 5, 3$ respectively. The weights are applied in Lines 7 and 11. One possible advantage of this weight set-up is that the first objective value is penalised more for violating time windows and the second for not reaching the destination. Therefore, the population can be expected to not be biased towards any of the two constraints.

Algorithm 6 FitnessAssignment(θ)

Input: $\theta :=$ trajectory to be evaluated
Output: $fitness :=$ fitness value
1: $fitness \leftarrow C(\theta)$
2: $maxCost \leftarrow$ Maximum value of any cost component in G.
3: $minHop \leftarrow$ Hopcount between θ and v_D
4: $conflicts \leftarrow$ The number of time window violations
5: **if** $v_D \neq v_{|\theta|}$ **then**
6: $\quad p_0 \leftarrow maxCost * minHop$
7: $\quad fitness \leftarrow fitness + (p_0 * \alpha_1, p_0 * \alpha_2)$
8: **end if**
9: **if** $conflicts > 0$ **then**
10: $\quad p_0 \leftarrow maxCost * conflicts$
11: $\quad fitness \leftarrow fitness + (p_0 * \alpha_3, p_0 * \alpha_4)$
12: **end if**
13: **return** $fitness$

V. Implementation details

All numerical tests are performed on Queen Mary's Apocrita HPC facility [65]. The methods are implemented in Python 3, and the inspyred package [66] was used for the evolutionary computation. Parallelisation has not been utilised. The variants of MARMT are the following: Direct (D), Integer Priority (IP) and Random Keys (RK), as discussed in Section IV. For parameter tuning, the irace package [64] was used. The minimum and maximum length of candidate trajectories for local search was set to $l_{min} = 3$ and $l_{rel} = 80\%$ in all experiments, as tuned by irace. The value of τ_{max} controlling the randomisation of the initial population (see Section IV-C) is also constant in all experiments, so that all variants start with approximately the same quality of their initial population. Tuning was carried out respectively for the different representations for values of crossover and mutation rates. Population size is kept the same across all variants, in order to ensure that the same local search rate will lead to an approximately an equal number of local search operations to be performed per generation. The tuned parameters are shown in Table II. The value of the local search rate is examined in Section VI.

All algorithms are tested using real data of operations at the Hong Kong International Airport (HKG) (7.1.2017, 0:00–24:00) and the Beijing Capital International Airport (PEK) (9.7.2014, 9:00–12:00). The taxiway layout of HKG and PEK can be categorised as medium and high complexity

TABLE II
PARAMETER VALUES

Variants	Pop. s.	Cross. r.	Mut. r.	τ_{max}	l_{min}	l_{rel}
D	120	0.90	0.19	4.5	3	0.8
IP	120	0.95	0.29	4.5	3	0.8
RK	120	0.83	0.13	4.5	3	0.8

respectively, with 1309 nodes, 1491 edges, 160 gates and 38 runway exits for HKG and 3194 nodes, 3928 edges, 286 gates and 53 runway exits for PEK. There are 506 aircraft routed for HKG and 200 for PEK. Aircraft will be routed sequentially using Algorithm 1, with time windows inflicted on later aircraft due to already routed aircraft. The most straightforward way of comparison is the overall travel time and fuel consumption realised for the whole day of operation. This is an important practical measure of efficiency for longer intervals of airport operations. Apart from the overall taxi time, it is also important to report how often there were no solutions found and a one minute postponement was applied until a solution became available. For this reason, we use *adjusted taxi time*, to account for the total postponements. For trajectory θ, the adjusted taxi time $C_{1,\theta,adj}$ in seconds can be calculated from the taxi time of the trajectory ($C_{1,\theta}$), and the number of postponements P for the given aircraft according to

$$C_{1,\theta,adj} = C_{1,\theta} + 60 * P. \tag{5}$$

The weights (w_1, w_2) for choosing the reserved trajectory from Θ_i for each aircraft are used as the surrogates of the operational cost coefficients to aggregate the two objectives for showing insights in a more concise form. This aggregate represents the real operational cost of the airport after a decision is made by air traffic controllers. Note that using any other weights could skew the results. The *weighted aggregate* ($C_{aggr,\theta}$) of a trajectory θ is calculated according to

$$C_{aggr,\theta} = C_{1,\theta,adj} * w_1 + C_{2,\theta,adj} * w_2. \tag{6}$$

To compare with AMOA* in a concise way, *relative weighted aggregate* (RWA) is introduced to characterise how MARMT performs compared to AMOA* regarding the reserved trajectories. For the i^{th} aircraft, RWA is calculated as

$$C_{aggr,i,rel} = \frac{C_{aggr,\theta_{i,MARMT}}}{C_{aggr,\theta_{i,AMOA*}}}. \tag{7}$$

from the weighted aggregate of the trajectory reserved by MARMT ($C_{aggr,\theta_{i,MARMT}}$) and by AMOA* ($C_{aggr,\theta_{i,AMOA*}}$).

We are not only interested in the reserved trajectories, but also in finding a close approximation of the real Pareto front for each aircraft. The real Pareto front is not known, because the existing enumerative solution approaches cannot guarantee to find all solutions. The unimpeded Pareto fronts - obtained by AMOA* by ignoring any time windows - are used as reference fronts. Thus, detours caused by the time windows are avoided and the resulting reference fronts are the same for the same aircraft, regardless of the previously routed aircraft. Note that these reference fronts contain solutions that my not be possible to achieve in reality. The ε quality indicator is used for assessing proximity to a reference Pareto front [67]. It signals higher quality by lower values. When the approximate front is the same as the reference front, ε equals 1. Another relevant metric is the size of the Pareto front. It is preferred to have more and uniformly distributed solutions [16], so that the trade-offs between the objectives can be assessed by air traffic controllers. Also, with more solutions, the chance for at least one of them to comply with time windows is better. However, it is easier to find many low-quality solutions than many high-quality ones. Therefore, both metrics are important.

TABLE III
RUNNING TIMES OF AMOA* (U=3) ROUTING A SINGLE AIRCRAFT.

	w_1	mean [s]	median [s]	min [s]	max [s]	std [s]
HKG	1	80.1	41.4	0.9	602.1	106.3
	0.5	45.3	22.9	0.4	338.8	61.6
	0	40.9	20.1	0.4	322.8	56.1
PEK	1	374.6	96.9	3.4	4747.1	715.2
	0.5	482.6	135.7	5.0	5787.4	863.2
	0	419.6	107.2	3.4	5426.2	789.5

VI. RESULTS

First, the results obtained by the state-of-the-art enumerative solution approach, AMOA* are described as a baseline. Then, results using MARMT are presented. AMOA* was chosen as the basis for comparison, as it provides the best solution quality among the previously proposed algorithms. As mentioned in Section II-B, AMOA* found 5-16% better solutions than the other approaches including k-QPPTW [4], when tested on the same airports as in the current study. MARMT is introduced with the aim of finding a solution faster than AMOA* with only a small compromise in solution quality. Two different termination criteria are explored for MARMT: (1) 10 generations without change in the Pareto optimal solutions found so far, to evaluate convergence properties and (2) 10 seconds time budget, to evaluate the potential use for real-time decision support. In the following, 10 independent runs of MARMT were performed for each parameter setting, where all aircraft are routed sequentially in each run. The Wilcoxon signed rank test was used to decide statistical significance.

A. Results based on the enumerative solution approach

AMOA* with $u = 3$ is used to route the aircraft, because 3 is the highest number of speed profiles per segment that can be solved in a reasonable time [4]. Table III describes the distribution of the running times for the aircraft. We can see that the running times of AMOA* range from 0.4 seconds to 602 seconds for HKG and from 3.4 seconds to 5787 seconds for PEK. The mean of the running times is much higher than the median, with most aircraft being routed in shorter times. However, a smaller number of them take significantly longer. Even for the smaller airport instance, the average running time is much higher than 10 seconds, which is the limit that is acceptable for on-line decision support [68].

B. Results based on convergence based termination

The case when the algorithm is allowed to run until convergence is considered first. Convergence is assumed when

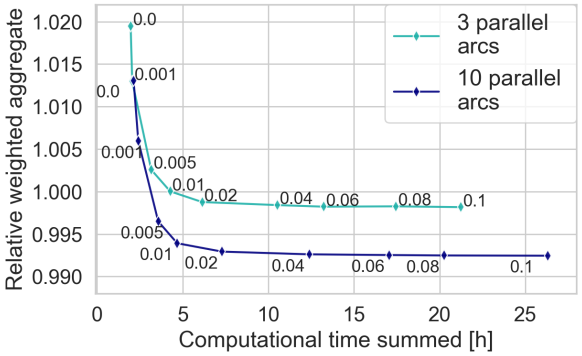


Fig. 6. Decreasing marginal improvement in mean RWA as local search rate is increased (annotated values) with MARMT-D. Experiments with different w_1 values are grouped together. Airport instance: HKG.

there is no improvement in the Pareto front for 10 consecutive generations. For the purpose of comparing to AMOA*, $u = 3$ is used. The case of $u = 10$ is also considered to investigate how MARMT scales to higher numbers of parallel arcs.

1) Quality of reserved trajectories: In Table IV, we show the quality of the solutions found through the mean RWA for the two instances. Statistical significances between the best result (in boldface) and the others in each sub-row are indicated as (*) : $p < 0.05$, (**): $p < 0.005$, (***): $p < 0.0005$. We can see that, in almost all cases for HKG, MARMT-D outperforms the priority based approaches, and random keys representation is the worst of the three. There are only a few cases where the statistical significance of the difference between MARMT-IP and MARMT-D cannot be established. For the larger PEK instance, the priority based representations perform better, with the priority based representation reaching the best values of RWA. It can also be seen from Table IV that with the local search rate value of 0.02, some variants of MARMT is able to reach the same or slightly better results as AMOA*, as can be seen from the values of RWA that are below 1. This is possible because of the non-additivity property and the presence of time windows. In fact, the difference between MARMT-D with a local search rate of 0.02 and AMOA* is statistically significant at $p = 0.05$ for all three values of $w1$ for HKG in terms of RWA. For PEK, all three variants (MARMT-D, MARMT-IP, MARMT-RK) with a local search rate of 0.02 outperform AMOA* at a statistical significance level $p = 0.005$.

Increasing the local search rate brings decreasing marginal improvement in solution quality, while increasing running time. In Figure 6, we see a sharp improvement in RWA until the local search rate reaches 0.02. Note, that the improvement upon the highest previous local search rate is statistically significant with $p = 0.005$ until the local search rate 0.06.

2) Quality of Pareto fronts found: Table V, shows results regarding the ε indicator, which quantifies the quality of the Pareto fronts found for individual aircraft by MARMT compared to the reference front. Similarly to Table IV, MARMT-D is the best for the HKG instance and MARMT-RK and MARMT-IP are the best for PEK. The ε values are close to 1, suggesting a good representation of the reference front.

We have seen that when the computational time of MARMT

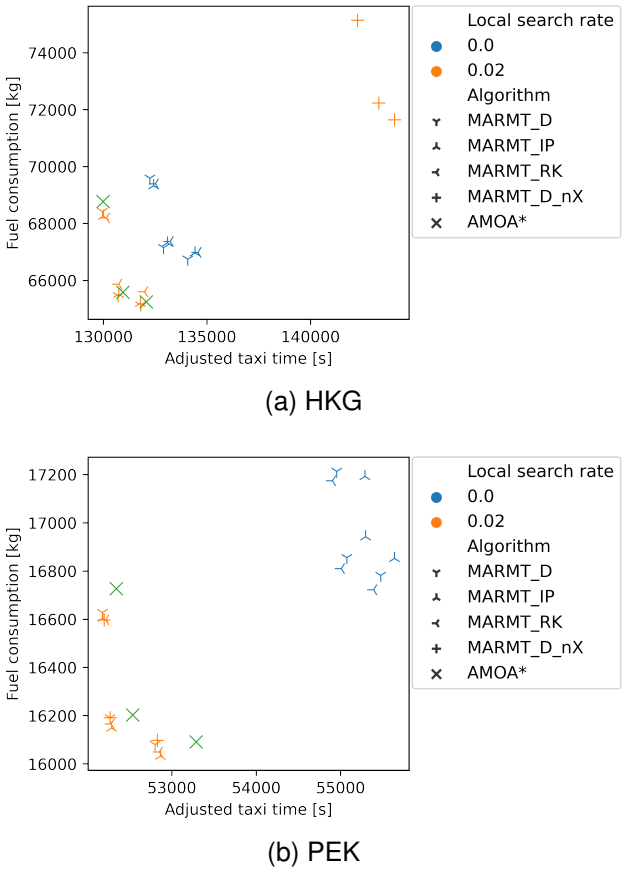


Fig. 7. Difference between MARMT (with and without local search) and AMOA*. Three different weights are used for reserving trajectories for individual aircraft. Each marker represents the average of 10 data points.

is not limited, MARMT is able to find Pareto fronts close to or better than those of AMOA*, when given enough time.

3) The contribution of operators: To demonstrate the importance of the local search for all variants and the modified crossover for MARMT-D, we compare the results of the algorithms to variants without local search and a version of MARMT-D that uses a naive version of crossover (not adjusted to the multigraph case), "MARMT-D-nX." MARMT-D-nX is the representation widely applied in simple graph problems [38], [16]. The results are shown in Figure 7. Some variants of MARMT dominate the results achieved by AMOA*, as also suggested by Table IV. The variants of MARMT without local search perform worse according to both objectives, which highlights the importance of local search. The role of the adjusted crossover for MARMT-D is less clear. MARMT-D-nX performs similar to MARMT-D for PEK, while for HKG, MARMT-D-nX performs much worse than any other variants. Therefore, it seems that the importance of the crossover being adjusted for the multigraph case depends on the characteristics of the airport layout, but our adjusted crossover produced better or no worse results across all instances of different types.

4) Higher numbers of parallel arcs: AMOA* is unable to handle higher numbers of parallel arcs. As it was reported in [69], AMOA* could not solve the HKG instance with $u = 10$, and the PEK instance with $u = 5$ within 10 days. MARMT on the other hand is able to handle a higher number of parallel

TABLE IV
MEAN RWA OF THE AIRCRAFT IN EACH OF THE TWO INSTANCES WITH VARIED LOCAL SEARCH RATES AND $u = 3$.

		$w_1 = 0.0$			$w_1 = 0.5$			$w_1 = 1.0$		
	local search r.	RK	D	IP	RK	D	IP	RK	D	IP
HKG	0.005	1.0136 **	**1.0029**	1.0083 **	1.0054 **	**1.0013**	1.0032 **	1.0056 **	1.0034	**1.0031**
	0.010	1.0092 **	**1.0002**	1.0026 **	1.0026 **	**0.9991**	1.0000 **	1.0025 **	**1.0007**	1.0016
	0.020	1.0055 **	**0.9983**	0.9993 **	1.0004 **	**0.9982**	0.9986 **	1.0012 **	**0.9998**	1.0001 *
	0.040	1.0033 **	**0.9979**	0.9983 *	0.9995 **	**0.9979**	0.9982 *	1.0005 **	**0.9994**	0.9996
	0.060	1.0023 **	**0.9977**	0.9981 **	0.9994 **	**0.9978**	0.9981 *	1.0002 **	**0.9992**	0.9994 *
PEK	0.005	**1.0034**	1.0060 *	1.0063 **	**1.0009**	1.0030 *	1.0031 **	**1.0013**	1.0035 *	1.0059 **
	0.010	1.0004	1.0017 *	**1.0002**	0.9977	**0.9975**	0.9977	0.9990	**0.9981**	0.9998 **
	0.020	0.9974 **	0.9993 **	**0.9963**	0.9957	0.9959 *	**0.9956**	0.9976	**0.9969**	**0.9969**
	0.040	0.9962 **	0.9983 **	**0.9948**	0.9953 **	0.9954 **	**0.9950**	0.9971 **	**0.9964**	0.9968 *
	0.060	0.9957 *	0.9979 **	**0.9942**	0.9952 **	0.9954 **	**0.9950**	0.9970 *	**0.9965**	0.9968 **

TABLE V
MEAN ε INDICATOR FOR SEQUENTIAL ROUTING OF AIRCRAFT IN EACH OF THE TWO INSTANCES WITH VARIED LOCAL SEARCH RATES AND $u = 3$.

		$w_1 = 0.0$			$w_1 = 0.5$			$w_1 = 1.0$		
	local search r.	RK	D	IP	RK	D	IP	RK	D	IP
HKG	0.005	1.0236 **	**1.0155**	1.0188 **	1.0213 **	**1.0147**	1.0165 *	1.0213 **	**1.0153**	1.0158
	0.010	1.0198 **	**1.0127**	1.0138 *	1.0174 **	**1.0108**	1.0120 *	1.0176 **	**1.0109**	1.0123 *
	0.020	1.0162 **	**1.0105**	1.0109	1.0144 **	**1.0090**	1.0094 *	1.0150 **	**1.0093**	1.0100 **
	0.040	1.0145 **	**1.0098**	1.0100	1.0129 **	**1.0082**	1.0087 **	1.0131 **	**1.0086**	1.0089 *
	0.060	1.0135 **	**1.0095**	1.0098 *	1.0122 **	**1.0081**	1.0084 *	1.0122 **	**1.0084**	1.0085
PEK	0.005	**1.0177**	1.0212 **	1.0218 **	**1.0162**	1.0198 **	1.0196 **	**1.0157**	1.0193 *	1.0197 **
	0.010	1.0142	1.0148	**1.0138**	1.0117	1.0124	**1.0113**	1.0122	1.0130	**1.0125**
	0.020	1.0104 **	1.0119 **	**1.0096**	1.0083 **	1.0095 **	**1.0077**	1.0096 *	1.0107 **	**1.0082**
	0.040	1.0093 **	1.0107 **	**1.0076**	1.0068 **	1.0085 **	**1.0061**	1.0081 **	1.0097 **	**1.0069**
	0.060	1.0086 **	1.0104 **	**1.0072**	1.0065 **	1.0082 **	**1.0057**	1.0077 **	1.0094 **	**1.0065**

TABLE VI
MEAN RWA OF THE AIRCRAFT IN EACH OF THE TWO INSTANCES WITH A LOCAL SEARCH RATE OF 0.02 AND $u = 10$.

		$w_1 = 0.0$			$w_1 = 0.5$			$w_1 = 1.0$		
	u	RK	D	IP	RK	D	IP	RK	D	IP
HKG	u=10	0.9957 **	**0.9834**	0.9867 **	1.0007 **	**0.9961**	0.9969 **	1.0031 **	**0.9993**	1.0001 **
	u=3	1.0055 **	**0.9983**	0.9993 **	1.0004 **	**0.9982**	0.9986 **	1.0012 **	**0.9998**	1.0001 *
PEK	u=10	0.9637 *	0.9708 **	**0.9619**	0.9959 **	0.9944 **	**0.9938**	1.0002 **	**0.9965**	0.9975 *
	u=3	0.9974 **	0.9993 **	**0.9963**	0.9957	0.9959 *	**0.9956**	0.9976	**0.9969**	**0.9969**

arcs. The results regarding RWA with $u = 10$ are shown in Table VI. The results have been obtained in less than 25 hours even for the PEK instance with $u = 10$, and less than 8 hours for the HKG instance with $u = 10$, which demonstrates the scalability of MARMT for higher numbers of parallel arcs. It can be seen from Table VI, that the best results in the $u = 10$ case are better than in the $u = 3$ case, thereby the solution quality is not worse even with the increased search space.

C. Potential for real-time decision support

The promptness of routing decisions can be crucial in real-world problems. In the airport ground movement problem, a trajectory needs to be found for each aircraft under 10 seconds for on-line decision support [68]. This time budget is used as the termination criteria in the following experiments. We can expect compromised solution quality with this low time budget, especially for the PEK instance. where AMOA* took 374-482 seconds on average to route a single aircraft for PEK, and 40-80 seconds for HKG. The results regarding RWA are shown in Table VII for the 10 second time budget. In the case of HKG, the RWA of MARMT-D is within 1 % of AMOA*, for both $u = 3$ and $u = 10$, which is remarkable considering the high running time of AMOA*. In the more challenging case of PEK, the RWA of MARMT is within 1-4 % of AMOA*. Considering that AMOA* was found to be 5-16 % better than the previous state-of-the-art [69], the performance can be judged as good. Without local search, the RWA values are consistently higher: within 1-5 % of AMOA* for HKG and 2-6 % for PEK, supporting the importance of local search in the case of having tight time budgets.

VII. CONCLUSION AND FUTURE DIRECTIONS

In this paper, a metaheuristic approach was proposed for the airport ground movement problem, as a representative of transportation problems that are best modelled as multigraphs. The adaptation to the specific problem includes modifying existing operators, incorporating time window constraints, constraint handling and proposing a local search operator for the problem. The proposed algorithms were evaluated using real data from two international airports of different sizes: HKG (medium) and PEK (complex). Three genetic representations were compared, including the direct, integer

TABLE VII
MEAN RWA OF THE AIRCRAFT IN EACH OF THE TWO INSTANCES WITH A TIME BUDGET OF 10 SECONDS.

			$w_1 = 0.0$			$w_1 = 0.5$			$w_1 = 1.0$		
			RK	D	IP	RK	D	IP	RK	D	IP
local search r.	instance	u									
0.02	HKG	3	1.0206 **	**1.0096**	1.0151 *	1.0091 **	**1.0031**	1.0068 **	1.0071 **	**1.0034**	1.0071 **
0.02	HKG	10	1.0116 **	**0.9948**	1.0049 **	1.0103 **	**1.0025**	1.0068 **	1.0118 **	**1.0045**	1.0079 **
0.02	PEK	3	1.0309 *	**1.0254**	1.0294	1.0255	**1.0243**	1.0268	1.0247	**1.0241**	1.0290 *
0.02	PEK	10	1.0031	1.0061 *	**1.0022**	1.0363 **	**1.0282**	1.0323	1.0441 **	**1.0355**	1.0388 *
0.0	HKG	3	1.0299 **	**1.0239**	1.0290 **	1.0215 **	**1.0182**	1.0219 **	1.0201 *	**1.0180**	1.0190
0.0	HKG	10	1.0190 **	**1.0096**	1.0213 **	1.0220 **	**1.0155**	1.0208 **	1.0224 **	**1.0163**	1.0222 **
0.0	PEK	3	1.0539	**1.0518**	1.0542	1.0590 **	**1.0531**	1.0565	1.0622 *	**1.0574**	1.0598
0.0	PEK	10	**1.0184**	1.0241 **	1.0238	1.0603 **	**1.0528**	1.0611 **	1.0692 **	**1.0587**	1.0723 **

priority based and random keys representations. Although, we cannot expect one representation to perform the best in all cases, the results were highly consistent for the same airport instance with the same termination criteria. With convergence based termination, MARMT-D performed the best on the HKG instance and MARMT-IP on the PEK instance. The performance of MARMT is very close to a state-of-the-art enumerative solution approach even with a 10 seconds time budget for HKG and is close enough for PEK. When allowed to converge, MARMT outperformed the said enumerative approach for both airports. The local search operator enhances the search capability of MARMT by introducing new high quality candidates to the population based on single objective search. We observed significant improvements in solution quality with the use of the local search operator for both the convergence based termination and the termination based on 10 second time budget.

It is also worth noticing that similar results were obtained using different weight values to choose a single trajectory, suggesting a good generalization to different preferences of the DM, that might stem from different operational situations. Including more speed profiles slightly improved solution quality in most cases, one exception is PEK with the 10 second time budget, where the increase in search space might be too large for a small time budget. It is possible in general that the flexibility allowed by more speed profiles cannot be fully capitalised, because the order of aircraft is fixed beforehand, as explained in [69]. The question of optimising trajectories of individual aircraft together with the sequence of aircraft - the global formulation of the problem - remains to be explored in future research. This might be tackled by a metaheuristic algorithm, or a reinforcement learning approach. An interesting method based on multiagent reinforcement learning is proposed in [70] for routing in packet networks, where the problem is modelled as a partially observable Markov decision process. The global formulation of the problem is of much higher complexity. Therefore, including one agent for each aircraft can be the right approach for managing a more dynamic environment. The meta-learning strategy is expected to be advantageous in adapting to changing conditions, such as varied levels of traffic.

This study based on both medium and large airports considering two objectives shows great potential for reaching real-time decision support with MARMT. Being able to achieve high-quality approximate solutions for the multigraph MSPP in a short time opens a way to investigating different airports, scenarios with denser traffic, including emissions as a third objective and considering intermediate holding of aircraft [71]. There is a high interest in many-objective shortest path problems, in line with the more realistic and detailed modelling of routing problems. A recent benchmark suite is provided in [18] for simple graph problems. The multigraph modelling approach investigated in this paper can be readily extended to many-objectives and the proposed solution approaches pave the first step to solving such problems effectively. The improvement of operators for priority based representations might become a fruitful area of further research. Often, the changes introduced in the chromosome are not sufficient to modify the encoded solution, limiting the exploration capabilities of these algorithms and leading to slow convergence. Strategies aimed at ensuring a modification of the encoded solution might mitigate some of the disadvantages of the ambiguity associated with the priority based representations.

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). Adriana Lara acknowledges support from project no. IPN-SIP 20221938. Lourdes Uribe acknowledges support from project no. IPN-SIP 20232208. Carlos A. Coello Coello gratefully acknowledges support from CONACyT grant no. 2016-01-1920 (Investigación en Fronteras de la Ciencia 2016).

REFERENCES

[1] S. Zajac and S. Huber, "Objectives and methods in multi-objective routing problems: a survey and classification scheme," *European Journal of Operational Research*, 2020.

[2] M. Kurant and P. Thiran, "Extraction and analysis of traffic and topologies of transportation networks," *Physical Review E*, vol. 74, no. 3, p. 036114, 2006.

[3] S. Ravizza, J. Chen, J. A. Atkin, E. K. Burke, and P. Stewart, "The trade-off between taxi time and fuel consumption in airport ground movement," *Public Transport*, vol. 5, no. 1-2, pp. 25–40, 2013.

[4] M. Weiszer, E. K. Burke, and J. Chen, "Multi-objective routing and scheduling for airport ground movement," *Transportation Research Part C: Emerging Technologies*, vol. 119, p. 102734, 2020.

[5] M. Gallet, T. Massier, and T. Hamacher, "Estimation of the energy demand of electric buses based on real-world data for large-scale public transport networks," *Applied energy*, vol. 230, pp. 344–356, 2018.

[6] J. Chen, M. Weiszer, G. Locatelli, S. Ravizza, J. A. Atkin, P. Stewart, and E. K. Burke, "Toward a more realistic, cost-effective, and greener ground movement through active routing: A multiobjective shortest path approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 12, pp. 3524–3540, 2016.

[7] M. Wen, D. Pacino, C. Kontovas, and H. Psaraftis, "A multiple ship routing and speed optimization problem under time, cost and environmental objectives," *Transportation Research Part D: Transport and Environment*, vol. 52, pp. 303–321, 2017.

[8] X. Yang, X. Li, B. Ning, and T. Tang, "A survey on energy-efficient train operation for urban rail transit," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 1, pp. 2–13, 2016.

[9] Q. Meng, D.-H. Lee, and R. L. Cheu, "Multiobjective vehicle routing and scheduling problem with time window constraints in hazardous material transportation," *Journal of transportation engineering*, vol. 131, no. 9, pp. 699–707, 2005.

[10] T. Garaix, C. Artigues, D. Feillet, and D. Josselin, "Vehicle routing problems with alternative paths: An application to on-demand transportation," *European Journal of Operational Research*, vol. 204, no. 1, pp. 62–75, 2010.

[11] G. Xiong and Y. Wang, "Best routes selection in multimodal networks using multi-objective genetic algorithm," *Journal of Combinatorial Optimization*, vol. 28, no. 3, pp. 655–673, 2014.

[12] P. Serafini, "Some considerations about computational complexity for multi objective combinatorial problems," in *Recent advances and historical development of vector optimization*. Springer, 1987, pp. 222–232.

[13] S. Ravizza, J. A. Atkin, and E. K. Burke, "A more realistic approach for airport ground movement optimisation with stand holding," *Journal of Scheduling*, vol. 17, no. 5, pp. 507–520, 2014.

[14] C. Lesire, "An iterative a* algorithm for planning of airport ground movements." in *ECAI*, vol. 2010, 2010, pp. 413–418.

[15] J. M. A. Pangilinan and G. K. Janssens, "Evolutionary algorithms for the multiobjective shortest path problem," *International Journal of Mathematical and Computational Sciences*, vol. 1, no. 1, pp. 7–12, 2007.

[16] C. Chitra and P. Subbaraj, "A nondominated sorting genetic algorithm solution for shortest path routing problem in computer networks," *Expert Systems with Applications*, vol. 39, no. 1, pp. 1518–1525, 2012.

[17] R. Li, Y. Leung, B. Huang, and H. Lin, "A genetic algorithm for multiobjective dangerous goods route planning," *International Journal of Geographical Information Science*, vol. 27, no. 6, pp. 1073–1089, 2013.

[18] J. Weise and S. Mostaghim, "A scalable many-objective pathfinding benchmark suite," *IEEE Transactions on Evolutionary Computation*, 2021.

[19] O. Dib, M. Dib, and A. Caminada, "Computing multicriteria shortest paths in stochastic multimodal networks using a memetic algorithm," *International Journal on Artificial Intelligence Tools*, vol. 27, no. 07, p. 1860012, 2018.

[20] L. Beke, M. Weiszer, and J. Chen, "A comparison of genetic representations and initialisation methods for the multi-objective shortest path problem on multigraphs," *SN Computer Science*, vol. 2, no. 3, pp. 1–22, 2021.

[21] P. Moscato and M. G. Norman, "A memetic approach for the traveling salesman problem implementation of a computational ecology for combinatorial optimization on message-passing systems," *Parallel computing and transputer applications*, vol. 1, pp. 177–186, 1992.

[22] F. Neri and C. Cotta, "Memetic algorithms and memetic computing optimization: A literature review," *Swarm and Evolutionary Computation*, vol. 2, pp. 1–14, 2012.

[23] C.-K. Goh, Y.-S. Ong, and K. C. Tan, *Multi-objective memetic algorithms*. Springer, 2008, vol. 171.

[24] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE transactions on evolutionary computation*, vol. 6, no. 2, pp. 182–197, 2002.

[25] J. C. N. Climaco and E. Q. V. Martins, "A bicriterion shortest path algorithm," *European Journal of Operational Research*, vol. 11, no. 4, pp. 399–404, 1982.

[26] J. Mote, I. Murthy, and D. L. Olson, "A parametric approach to solving bicriterion shortest path problems," *European Journal of Operational Research*, vol. 53, no. 1, pp. 81–92, 1991.

[27] A. Raith and M. Ehrgott, "A comparison of solution strategies for biobjective shortest path problems," *Computers & Operations Research*, vol. 36, no. 4, pp. 1299–1331, 2009.

[28] E. Q. V. Martins, "On a multicriteria shortest path problem," *European Journal of Operational Research*, vol. 16, no. 2, pp. 236–245, 1984.

[29] A. J. Skriver and K. A. Andersen, "A label correcting approach for solving bicriterion shortest-path problems," *Computers & Operations Research*, vol. 27, no. 6, pp. 507–524, 2000.

[30] E. W. Dijkstra *et al.*, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[31] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[32] L. Mandow, J. P. De la Cruz *et al.*, "A new approach to multiobjective a* search." in *IJCAI*, vol. 8. Citeseer, 2005.

[33] L. D. P. Pugliese and F. Guerriero, "A survey of resource constrained shortest path problems: Exact solution approaches," *Networks*, vol. 62, no. 3, pp. 183–200, 2013.

[34] J. Desrosiers, P. Pelletier, and F. Soumis, "Plus court chemin avec contraintes d'horaires," *RAIRO-Operations Research*, vol. 17, no. 4, pp. 357–377, 1983.

[35] M. Desrochers and F. Soumis, "A generalized permanent labelling algorithm for the shortest path problem with time windows," *INFOR: Information Systems and Operational Research*, vol. 26, no. 3, pp. 191–212, 1988.

[36] I. Ioachim, S. Gelinas, F. Soumis, and J. Desrosiers, "A dynamic programming algorithm for the shortest path problem with time windows and linear node costs," *Networks: An International Journal*, vol. 31, no. 3, pp. 193–204, 1998.

[37] W. B. Powell and Z.-L. Chen, "A generalized threshold algorithm for the shortest path problem with time windows." in *Network Design: Connectivity and Facilities Location*, 1997, pp. 303–318.

[38] C. W. Ahn and R. S. Ramakrishna, "A genetic algorithm for shortest path routing problem and the sizing of populations," *IEEE transactions on evolutionary computation*, vol. 6, no. 6, pp. 566–579, 2002.

[39] J. Inagaki, M. Haseyama, and H. Kitajima, "A genetic algorithm for determining multiple routes and its applications," in *ISCAS'99. Proceedings of the 1999 IEEE International Symposium on Circuits and Systems VLSI (Cat. No. 99CH36349)*, vol. 6. IEEE, 1999, pp. 137–140.

[40] M. Gen, F. Altiparmak, and L. Lin, "A genetic algorithm for two-stage transportation problem using priority-based encoding," *OR spectrum*, vol. 28, no. 3, pp. 337–354, 2006.

[41] L. Lin and M. Gen, "An effective evolutionary approach for bicriteria shortest path routing problems," *IEEJ Transactions on Electronics, Information and Systems*, vol. 128, no. 3, pp. 416–423, 2008.

[42] C. Blum and A. Roli, "Metaheuristics in combinatorial optimization: Overview and conceptual comparison," *ACM computing surveys (CSUR)*, vol. 35, no. 3, pp. 268–308, 2003.

[43] Z. Ji, Y. S. Kim, and A. Chen, "Multi-objective α-reliable path finding in stochastic networks with correlated link costs: A simulation-based multi-objective genetic algorithm approach (smoga)," *Expert Systems with Applications*, vol. 38, no. 3, pp. 1515–1528, 2011.

[44] M. Gen, R. Cheng, and D. Wang, "Genetic algorithms for solving shortest path problems," in *Proceedings of 1997 IEEE International Conference on Evolutionary Computation (ICEC'97)*. IEEE, 1997, pp. 401–406.

[45] M. Gen and L. Lin, "A new approach for shortest path routing problem by random key-based ga," in *Proceedings of the 8th annual conference on genetic and evolutionary computation*. ACM, 2006, pp. 1411–1412.

[46] C. A. C. Coello, "Constraint-handling techniques used with evolutionary algorithms," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, 2021, pp. 692–714.

[47] D. M. Miranda, J. Branke, and S. V. Conceição, "Algorithms for the multi-objective vehicle routing problem with hard time windows and stochastic travel time and service time," *Applied Soft Computing*, vol. 70, pp. 66–79, 2018.

[48] J. Chen, M. Weiszer, P. Stewart, and M. Shabani, "Toward a more realistic, cost-effective, and greener ground movement through active routing—part i: Optimal speed profile generation," *IEEE Transactions on Intelligent Transportation Systems*, 2016.

[49] R. L. Carraway, T. L. Morin, and H. Moskowitz, "Generalized dynamic programming for multicriteria optimization," *European journal of operational research*, vol. 44, no. 1, pp. 95–104, 1990.

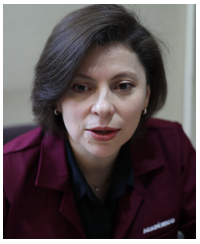
[50] D. S. Lai, O. C. Demirag, and J. M. Leung, "A tabu search heuristic for the heterogeneous vehicle routing problem on a multigraph," *Transportation Research Part E: Logistics and Transportation Review*, vol. 86, pp. 32–52, 2016.

[51] H. B. Ticha, N. Absi, D. Feillet, and A. Quilliot, "Empirical analysis for the vrptw with a multigraph representation for the road network," *Computers & Operations Research*, vol. 88, pp. 103–116, 2017.

[52] X. Wu, X. He, G. Yu, A. Harmandayan, and Y. Wang, "Energy-optimal speed control for electric vehicles on signalized arterials," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 5, pp. 2786–2796, 2015.
[53] H. N. Psaraftis and C. A. Kontovas, "Speed models for energy-efficient maritime transportation: A taxonomy and survey," *Transportation Research Part C: Emerging Technologies*, vol. 26, pp. 331–351, 2013.
[54] ——, "Ship speed optimization: Concepts, models and combined speed-routing scenarios," *Transportation Research Part C: Emerging Technologies*, vol. 44, pp. 52–69, 2014.
[55] D. Li, M. Yang, C.-J. Jin, G. Ren, X. Liu, and H. Liu, "Multi-modal combined route choice modeling in the maas age considering generalized path overlapping problem," *IEEE Transactions on Intelligent Transportation Systems*, 2020.
[56] M. Enzi, S. N. Parragh, and J. Puchinger, "The bi-objective multimodal car-sharing problem," *arXiv preprint arXiv:2010.10344*, 2020.
[57] J. Hrnčíř, M. Rovatsos, and M. Jakob, "Ridesharing on timetabled transport services: A multiagent planning approach," *Journal of Intelligent Transportation Systems*, vol. 19, no. 1, pp. 89–105, 2015.
[58] M. Weiszer, J. Chen, and P. Stewart, "A real-time active routing approach via a database for airport surface movement," *Transportation Research Part C: Emerging Technologies*, vol. 58, pp. 127–145, 2015.
[59] H. Khadilkar and H. Balakrishnan, "Estimation of aircraft taxi fuel burn using flight data recorder archives," *Transportation Research Part D: Transport and Environment*, vol. 17, no. 7, pp. 532–537, 2012.
[60] Q. Zhang and H. Li, "Moea/d: A multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on evolutionary computation*, vol. 11, no. 6, pp. 712–731, 2007.
[61] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: solving problems with box constraints," *IEEE transactions on evolutionary computation*, vol. 18, no. 4, pp. 577–601, 2013.
[62] L. Beke, M. Weiszer, and J. Chen, "A comparison of genetic representations for multi-objective shortest path problems on multigraphs," in *European Conference on Evolutionary Computation in Combinatorial Optimization (Part of EvoStar)*. Springer, 2020, pp. 35–50.
[63] A. Homaifar, C. X. Qi, and S. H. Lai, "Constrained optimization via genetic algorithms," *Simulation*, vol. 62, no. 4, pp. 242–253, 1994.
[64] M. López-Ibáñez, J. Dubois-Lacoste, L. Pérez Cáceres, T. Stützle, and M. Birattari, "The irace package: Iterated racing for automatic algorithm configuration," *Operations Research Perspectives*, vol. 3, pp. 43–58, 2016.
[65] "This research utilised queen mary's apocrita hpc facility, supported by qmul research-it." http://doi.org/10.5281/zenodo.438045.
[66] "Inspyred: Bio-inspired algorithms in python," https://pythonhosted.org/inspyred/, accessed: 2019-10-30.
[67] A. Liefooghe and B. Derbel, "A correlation analysis of set quality indicator values in multiobjective optimization," in *Proceedings of the Genetic and Evolutionary Computation Conference 2016*, 2016, pp. 581–588.
[68] "Icao, 2004. advanced surface movement guidance and control systems (a-smgcs) manual. international civil aviation organization," http://www.icao.int/Meetings/anconf12/Document.
[69] M. Weiszer, E. K. Burke, and J. Chen, "Search graph structure and its implications for multi-graph constrained routing and scheduling problems," *Scientific Reports*, vol. 12, no. 1, pp. 1–13, 2022.
[70] L. Chen, B. Hu, Z.-H. Guan, L. Zhao, and X. Shen, "Multiagent meta-reinforcement learning for adaptive multipath routing optimization," *IEEE Transactions on Neural Networks and Learning Systems*, 2021.
[71] T. Zhang, L. Beke, S. Liu, M. Weiszer, and J. Chen, "An extended memetic algorithm for multiobjective routing and scheduling of airport ground movements with intermediate holding," in *2022 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2022, pp. 1–8.

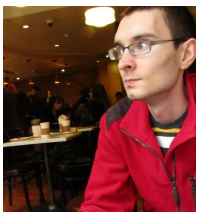
Lilla Beke is currently a PhD candidate at the Queen Mary University of London. She received the master's degree in Applicable Mathematics in 2017 from the London School of Economics and Political Science and the bachelor's degree in Energy Engineering from Budapest University of Technology and Economics in 2016. Her interests include multi-objective combinatorial optimisation and machine learning.

Lourdes Uribe received a PhD (Cum Laude) in physical-mathematical sciences from ESFM-Instituto Politécnico Nacional, México, in 2020. She is currently Professor at the Mathematical Department of ESFM-IPN, in Mexico City, Mexico. Her major research interests are: hybrid algorithms, multi-objective optimization and constraint-handling techniques.



Adriana Lara is a Full-Time Professor with the Physics and Mathematics School (ESFM) at the IPN in México City. Her research interests include Multi-objective Optimization, Bio-inspired Algorithms, Memetic Techniques, and Data Science. Her research has received the IEEE Transactions on Evolutionary Computation Outstanding Paper Award for 2010 and 2012. She also received the 2010 Engineering Award granted by Mexico City's Science and Technology Institute (ICyTDF) and the Ph.D. dissertation award by SMIA and ANIEI in 2013. She received a B.Sc. in Physics and Mathematics from the National Polytechnic Institute of Mexico (IPN) and an M.Sc. and Ph.D. in Computer Sciences from Centro de Investigación y Estudios Avanzados (CINVESTAV-IPN), in Mexico City, Mexico.

Carlos Artemio Coello Coello (M'98-SM'04-F'11) received a PhD in computer science from Tulane University, USA, in 1996. He is currently Professor (CINVESTAV-3F Researcher) at the Computer Science Department of CINVESTAV-IPN, in Mexico City, Mexico. He has authored and co-authored over 550 technical papers and book chapters. His publications report over 66,060 citations in Google Scholar. His major research interests are: evolutionary multi-objective optimization and constraint-handling techniques for evolutionary algorithms.



Dr. Michal Weiszer is a Research Assistant and Teaching Fellow at School of Engineering and Materials Science, Queen Mary University of London. His research interests span computer science and operational research with systems engineering approach. He has applied simulation models of systems in diverse areas from supply chain, air traffic management and utility networks. In order to balance different objectives within a system, Dr Weiszer has used multi-objective optimisation techniques with application in transportation networks (freight and passenger). His research often relies on Data-Centric approach to successfully model systems.

Edmund K. Burke is Vice-Chancellor at Bangor University. His research interests are in intelligent decision support methodologies in complex environments. His inter-disciplinary work in Operational Research lies at the interface of Computer Science and Mathematics. He is a Fellow of the Royal Academy of Engineering and Past- President of the Operational Research Society.

Jun Chen received the Ph.D. degree in control and systems engineering from the University of Sheffield, Sheffield, U.K., in 2010. He is now a Reader in Intelligent Systems Engineering at Queen Mary University of London, London, U.K. He has published widely in areas of multi-objective optimisation, interpretable fuzzy systems, data-driven modelling, and intelligent transportation systems. From 2020, he serves as a full member of the EPSRC Peer Review College. From 2018, he have been Turing Fellow with the national artificial intelligence research institute – the Alan Turing Institute.