IEEE Computational Intelligence Society
Transactions on Evolutionary Computation

Memetic Algorithms for Multi-objective Routing and Scheduling of Airport Ground Movement

Journal:	*Transactions on Evolutionary Computation*
Manuscript ID	TEVC-00048-2022
Manuscript Type:	Regular Papers
Date Submitted by the Author:	27-Jan-2022
Complete List of Authors:	Beke, Lilla; Queen Mary University of London, School of Engineering and Materials Science Uribe, Lourdes; IPN ESFM, Mathematics Department Lara, Adriana; IPN ESFM, Mathematics Department Coello Coello, Carlos; CINVESTAV IPN, Department of Computer Science; Basque Center for Applied Mathematics, Heuristic Optimization Weiszer, Michal; Queen Mary University of London, School of Engineering and Materials science Burke, Edmund; University of Leicester, University Road Chen, Jun; Queen Mary University of London, School of Engineering and Materials Science
Keywords:	Multigraph, Multi-objective routing and scheduling, Airport ground movement, Memetic algorithm, Time windows

Memetic Algorithms for Multi-objective Routing and Scheduling of Airport Ground Movement

Lilla Beke, Lourdes Uribe, Adriana Lara, *Member, IEEE*, C.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. The latter is required in multiple applications to represent alternative routes with different costs linking the same nodes. The detailed modelling approach increases computational complexity and may also lead to violation of the additivity property of costs. Therefore approximate solution methods become more suitable. 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 on its different variants; these variants are based on diverse genetic representation methods. We propose a local search operator that provides significant improvements. Our results also show that the best variant of MARMT is consistently producing high quality results in shorter times compared to a state of the art enumerative algorithm. The algorithms are tested on real data. MARMT can be adapted for other applications with minor modifications, such as train operations or electric vehicle routing.**

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

I. Introduction

EFFICIENCY of transportation systems is key to satisfying the increasingly high levels of industrial and commercial demands of today, while balancing the economic cost and environmental impact. Transportation related problems are often formulated as variations of the Shortest Path Problem [1]. Generally, there are conflicting objectives to be considered in such problems, often including travel time and energy consumption (including fossil and sustainable energy). The presence of multiple objectives implies that generally there is not a single solution optimising all objectives. Therefore, the goal is to find a set of solution paths with non-dominated costs in compliance with certain time constraints.

Often, the infrastructure in a transportation system is described through a graph for the purposes of the routing problem [2]. Nodes correspond to important places in the system, such as junctions, stations, and starting and destination points. In a simple graph model, a directed arc between two nodes implies a direct link between the corresponding places in the system in the marked direction. A series of connected arcs (a path) in the graph corresponds to a route.

Optimising airport ground operations exemplifies the multi-objective 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 most 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 shown to provide better solutions than the simple graph approach in [4]. A similar trade-off is often present when routing different vehicles [5], suggesting a wider applicability for algorithms devised 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 feasibility of the solution, 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], routing in maritime transportation [7], train operations [8] and transport of hazardous materials [9]. In other contexts the multigraph modelling approach has also been employed for the vehicle routing problem [10] and multi-modal transportation [11].

The Multiobjective Shortest Path Problem (MSPP) is NP-hard even on simple graphs without time constraints [12] and it is NP-complete when time constraints are also considered [13]. The multigraph approach further increases search space and computational complexity. 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, unlike most exact approaches, they can handle costs that do not satisfy the additivity property (detailed in Section III-D).

Genetic algorithms (GA) are metaheuristics that are widely applied to the MSPP [14], [15], [16], [17], and to multimodal transport problems [18], [11]. Our previous work explored the choice of representation schemes for the multigraph MSPP

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

C.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 Basque Center for Applied Mathematics (BCAM) & Ikerbasque, Spain.

E. K. Burke is with University of Leicester, UK. e-mail: edmund.burke@leicester.ac.uk

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

in artificial problem instances without time constraints [19]. The presence of time constraints calls for 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 [20], [21], [22]. The local search procedure explores the search space around a specific candidate solution. This way the local information of the selected solution is exploited, giving a new, improved solution; then, this improved solution is incorporated into the population.

In light of above, we introduce the 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 [23]. Our focus is on the design of local search operator and constraint handling scheme, and the comparison of the different encoding schemes used for MSPPs. Our results are also compared to a state-of-art enumerative solution approach [4].

The main contributions include: (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 the higher number of parallel arcs and non-additivity of costs better compared to the enumerative approach. (ii) A local search operator is proposed based on single objective search. Integrating the local search operator to 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 are 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 results are discussed in Section VI. Finally, conclusion is drawn 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 labelling methods. Ranking methods [24] 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 [25], [26] 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 [27] and label correcting [28] methods generalise the labeling solution techniques used for the single-objective shortest path problem, such as Dijkstra's algorithm [29] for more objectives. The efficiency of the above approaches have been compared empirically in [26], where labelling methods and two phase methods were found to be the best in most cases.

Extensions of labeling algorithms based on the A* algorithm [30] such as The New Approach to Multiobjective A* (NAMOA*) 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 all possible 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 Pareto optimal solution globally. 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 are few studies of MSPP with time constraints. Most studies on constrained shortest path problems are considering resource constraints, and they are overwhelmingly about single objective problems [31]. Time constraints pose a unique challenge. Examples of ranking and labeling methods proposed for the MSPP with time constraints are reviewed in II-B.

2) Metaheuristic algorithms: Several studies applied GAs to shortest path problems with various representation methods, including direct variable length [32], direct fixed length [33], random keys [34] and integer-valued priority [35] representations. 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 [36].

The direct variable length representation [32] has been employed for the MSPP by multiple authors [15], [37], [16]. 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 crossover might not be able to produce any novel candidates.

The two priority based representations are the integer-valued priority representation [38], [35] and random keys [39]. The random keys representation employs floating-point numbers as priorities. In both representations a path is encoded through 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 key representation

has 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 equal 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 [19], 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 complications due to high numbers of objectives or constraints [40]. Penalty functions are the simplest and perhaps the most widely applied methods. They can be sufficient for multiobjective problems with fewer constraints [41]. However, they are thought to be less suited for handling a larger 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 (MA) supplement the evolutionary process with a local search process [20], [21]. This is a popular extension of GAs, in order to avoid premature convergence and guide the population towards promising areas of the search space. MA approach has been proposed for the dynamic shortest path problem in simple graphs in [18]. 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 a very high, especially in a multigraph.

B. Airport ground movement problem

The airport ground movement problem is concerned with routing and scheduling of 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. Efficiency of airport ground movement 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 order to keep the complexity of the problem manageable. In this paper the sequential approach is considered.

Earlier studies [42], [43] suffered from multiple limitations, such as considering only a single objective and assumption of a constant speed for calculating traversal times. Single objective approaches can not provide the available trade-offs in a single run. Realism of calculating traversal times is of key importance 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. Great improvements were achieved regarding both taxi time and fuel consumption by Chen et. al. with the trajectory-based ground movement operations framework [44], [6], by managing routing and scheduling in an integrated way, with realistic speed profiles.

Weiszer et. al. adapted the NAMOA* algorithm [45] 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 [46] (see Section III-D) for the metaheuristic approach.

C. Other real-world applications

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

One of the problems where the multigraph model was shown to be valuable is time-constrained vehicle routing problems. Using a multigraph model for an on demand transportation problem reduced associated costs compared to a simple graph model [10]. A similar approach was followed by multiple authors in vehicle routing problems [47], [48].

Optimising energy efficiency of urban rail transit can also be conceptualised through a multigraph, where optimising speed

profiles and time tables in an integrated way provides significant energy savings [8]. In urban rail transit, vehicles interact not only through inflicting time constraints on each other, but through regenerative breaking, which entails synchronization of the accelerating/braking actions.

Optimal speed control of individual electric vehicles taking into account queues at intersections is studied in [49]. It is pointed out that optimising for individual vehicles might compromise system level efficiency, however, this is 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 with respect to fuel price and travel time [50]. It has also been shown that the optimal route depends on the optimised objective [51], 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 as proposed in this paper might be of great use.

Multimodal transportation problems [11], [52] and ride-sharing problems [53], [54] 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).

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 computation time [55]. 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 [55].

A. Sequential routing of aircraft

Aircraft are routed on a first come first serve 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 $Route$. $Route$ 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 there aren't any solutions found. We do not consider holding during taxiing.

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. This ensures that the Decision Maker (DM) gets accurate information about the available trade-offs. Our primary interest is 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 one realised trajectory is chosen according to a weighted sum of the costs. The weights of the objectives are set such that $w_2 = 1 - w_1$.

Algorithm 1 Sequential routing of aircraft

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

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 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 moving 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 occupation of the edges, a set of time windows ($\mathcal{F}_e$) are assigned to each edge. Time windows are corresponding 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 after turning accelerate. Thus, it makes sense to group together sequences of edges depending on their geometry. For this

TABLE I
NOTATIONS.

Notation	Description	Notation	Description
$aircraft_i$	The ith 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 kth 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

reason, speed profiles are modelled as *straight* and *turning segments* as defined in [56], [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 a way to cover all possible edge-sequences in the layout graph [55].

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 $I(e, (v, w))$.

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 including the first u speed profiles.

It is important to note that the number of parallel arcs in G sometimes can 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 on 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 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 for aircraft in different weight category. However, within the same weight category, the difference is small, and can be quickly modified before routing each aircraft.

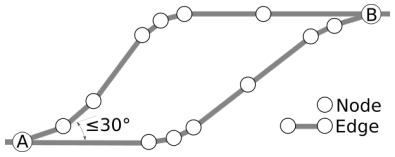

Fig. 1. Source of inhomogeneous numbers of parallel arcs in G.

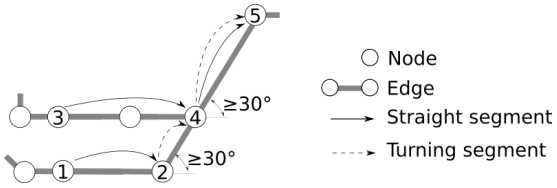

Fig. 2. Illustration of non-additivity property. The segment 4-5 is a turning segment, when approached via segment 3-4, and is a straight segment when approached via segment 2-4. Depending on the direction, the cost-vector of segment 4-5 is different, a turning segment is more costly. It is possible 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.

D. Non-additivity of costs

Straight or turning speed profiles can be associated with the same segment. Which speed profiles are appropriate is governed through 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.

E. Routing problem considering a single aircraft

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

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

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

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

1) Satisfy predecessor edges. $k_j \in \boldsymbol{I}(e,(v_j,v_{j+1})),\ \forall j \in (1,2,\ldots,|\theta|-1)$, where $e = pred((v_j,v_{j+1}),\theta)$
2) Satisfy time windows. For each edge e in trajectory θ, 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. Listing the end nodes of all edges in θ should not contain duplicates.
4) The trajectory should start with the origin node and end with 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 end nodes of segments. Constraint 4 ensures that the end points of the trajectory are as required. Constraint 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.

We are looking for the set of feasible trajectories Θ_i with Pareto optimal costs. The cost-vector of a feasible trajectory θ can be calculated according to Equation (3).

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

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.

IV. The 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 [23]. However, it can be easily modified to use other multi-objective evolutionary strategies [57], [58]. The operators are performed in the following order: mutation, crossover and local search. This way 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 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) that form a path in the speed profile graph (G) and the corresponding parallel arc indices (k_i) in the following form: $[v_1, k_1, v_2, k_2, ...v_{|\theta|}]$.

1) Decoding: Decoding a candidate to a trajectory in G is straightforward, with an 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 [32], that 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*.

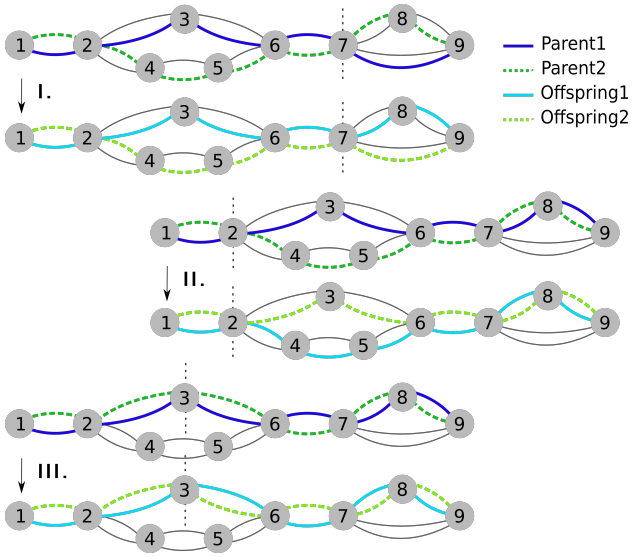

Fig. 3. Direct crossover in multigraphs. Example I. shows an ideal crossover akin to the simple graph case, with novel node sequences in the offspring. Example II. shows the lack of an ideal crossing site with distinct parent node sequences. Example III. shows parents with identical node sequences.

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, while there isn't any 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

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** $ch1, ch2$

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 (Line 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 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 the 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 [29]. Single objective shortest path problems can be solved efficiently by exact algorithms. 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 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, akin to a death penalty, 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, where n is the number of nodes. In the case of 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 [19]. 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.

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: $x \leftarrow \mid I(predEdge, prevNode, nextNode) \mid$
14: $currentIndex \leftarrow \lfloor M_{2,nextNode} * x \rfloor + 1$
15: $indices = indices \cup currentIndex$
16: $predEdge = pred((prevNode, nextNode), \theta(nodes, indices))$
17: **end if**
18: **end while**
19: **return** $nodes, indices$

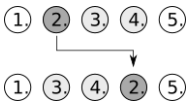

Fig. 4. Illustration of insertion mutation.

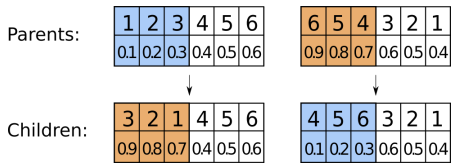

Fig. 5. Illustration of WMX for the matrix chromosome.

3) Crossover: For the Integer priority representation, Weight Mapping Crossover (WMX) [35] is adopted, that 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 the most efficient in [59]. 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 input, together defining 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 unvisited nodes in the graph have lower priorities. In Lines 9-13, the rest of the genes are filled up with 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: $M_{1,nodes[i]} \leftarrow priority$
5: $predEdge \leftarrow$ predecessor edge for segment $nodes[i], nodes[i+1]$
6: $M_{2,nodes[i]} \leftarrow \frac{indices[i]}{\mid I(predEdge,(nodes[i],nodes[i+1])) \mid}$
7: $priority \leftarrow priority - 1$
8: **end for**
9: **for** $j \leftarrow 1$ to n **do**
10: **if** $M_{1,j}$ is not yet specified **then**
11: $M_{1,j} \leftarrow priority$
12: $M_{2,j} \leftarrow$ Random floating-point number $\in [0, 1]$
13: $priority \leftarrow priority - 1$
14: **end if**
15: **end for**
16: **return** M

C. Initialisation

Heuristic initialisation is used from our previous work [59]. 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 $h(v, v_D, G)$, and τ_{max} denotes the maximum value of the randomisation coefficient τ. The likelihood of detours appearing in the decoded paths can be controlled by the parameter τ_{max}. The higher τ_{max} is, the more random the priorities are, and the less prominent is the effect of the heuristic initialisation compared to a purely random one. 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 [60]. 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). For not reaching v_D, the level of violation is measured as the minimum distance of θ and v_D (Line 3). For 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 [61], 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: $\theta :=$ cost vector of trajectory
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 [62]. The methods are implemented in Python 3, and the inspyred package [63] 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 tuning the parameters, the irace package [61] 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 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 approximately equal number of local search 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 a day of operations at the Hong Kong International Airport (7.1.2017, 0:00–24:00). This taxiway layout can be categorised as medium complexity, with 1309 nodes, 1491 edges, 160 gates and 38 runway exits. In this study, 506 aircraft will be routed

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

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 AMOA* in a concise way, *relative weighted aggregate* (RWA) is also introduced to characterise how the MARMT performs compared to AMOA* regarding the reserved trajectories. For the ith 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 ε quality indicator is used for assessing proximity to the real Pareto front [64]. It signals higher quality by lower values. When the approximate front is the same as the reference front, ε equals 1. The unimpeded Pareto fronts - obtained with ignoring time windows - are used as reference fronts. This way, the reference is always the same for the same aircraft, regardless of the current strategy for choosing the reserved trajectory from Θ_i. Another relevant metric is the size of the Pareto front. It is preferred to have more and uniformly distributed solutions [15], 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 complying 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]	sum [h]
1	80.1	41.4	0.9	602.1	106.3	11.26
0.5	45.3	22.9	0.4	338.8	61.6	6.37
0	40.9	20.1	0.4	322.8	56.1	5.74

TABLE IV
NUMBER OF SOLUTIONS FOUND BY AMOA* (U=3)

w_1	mean	median	min	max	std
1	13.78	9	1	91	14.04
0.5	13.89	9	1	91	14.15
0	13.86	9	1	91	14.13

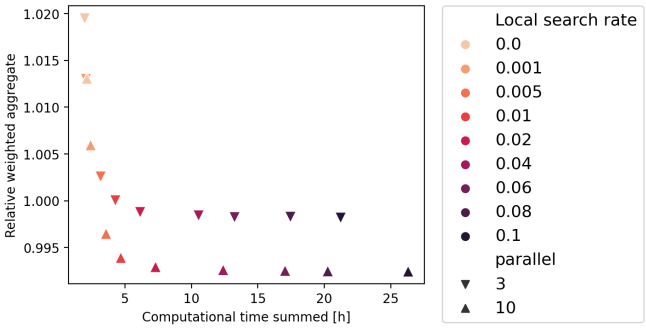

Fig. 6. Decreasing marginal improvement in solution quality as measured by the mean RWA and increase in running time as local search rate is increased with MARMT-D. Experiments with different w_1 values are grouped together.

VI. RESULTS

First, the results obtained by the state-of-the-art enumerative solution approach are described as a baseline. Then, results of MARMT are presented. 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, one-sided 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 all 506 aircraft, because that is the highest number of speed profiles per segment that can be solved in a reasonable time. Table III describes the distribution of the running times for all aircraft. We can see that running times of AMOA* range from 0.4 seconds to 602 seconds. Higher running times are observed when the fastest trajectory ($w_1 = 1$) is reserved for each aircraft than in the other two cases. The mean of the running times is approximately twice of the median, showing a skewed distribution with most aircraft being routed in shorter times, while a smaller number of them taking significantly longer. The average running time is much higher than 10 seconds, which is the limit acceptable for on-line decision support [65].

Table IV describes the number of optimal solutions found. A skewed distribution where the average number of solutions found is 13 and the maximum is 91 can be observed for all three values of $w1$.

B. Results based on convergence based termination

First, we consider the case when the algorithm is allowed to run until convergence. Convergence is assumed when there is no improvement in the Pareto front found so far for 10 consecutive generations. For the purpose of comparing to AMOA*, $u = 3$ is used and $u = 10$ is also included to show how MARMT scales to higher numbers of parallel arcs.

1) Quality of reserved trajectories: In Table V we show quality of solutions found through the mean RWA for the whole day of operation. Statistical significance between the best result (bold) 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 MARMT-D outperforms the priority based ones, 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. We can therefore conclude that MARMT-D performs the best in terms of RWA, when the computational time is not limited. Table V also shows that with the local search rate value of 0.02, MARMT-D is able to reach the same or slightly better results as AMOA*, when it is used with the same strategy for reserving routes for individual aircraft. This is possible, because of the non-additivity property and the presence of time windows.

Increasing the local search rate brings decreasing marginal improvement in solution quality, while increasing running time, as can be seen in Figure 6. We see a sharp improvement in RWA until the local search rate reaches 0.02. The sum of running times with the local search rate of 0.02 is 6.13 hours for the whole day of operations with $u = 3$ and 7.29 hours with $u = 10$. This is close to the computational times observed with AMOA*, that is between 5.6 hours and 11.2 hours for $u = 3$. With the local search rate of 0.1, the computational time is above 21 hours, but there is only a modest further improvement in RWA. 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 for $u = 3$ and until the local search rate 0.04 for $u = 10$. Regarding the number of parallel arcs, the RWA is lower for all local search rates with $u = 10$, at least with MARMT-D, as can be seen in Table V. The same trend is shown in Figure 6, where we also can see that the additional computational time required is low, especially compared to AMOA*. For $u = 10$, AMOA* would take longer than 10 days [4].

The results shown so far only consider the RWA. Figure 7 shows the objectives separately for AMOA* and for different variants of MARMT. With local search rate of 0.02, MARMT-D dominates the results achieved by AMOA*.

2) Quality of Pareto fronts found: In Table VI we can see results regarding the ε indicator, which quantifies the quality of the Pareto fronts found for individual aircraft by MARMT. MARMT-D is the best again among the three regardless of how often local search is performed. The statistical signifi-

TABLE V
MEAN RWA OF THE 506 AIRCRAFT WITH VARIED LOCAL SEARCH RATES.

		$w_1 = 0.0$			$w_1 = 0.5$			$w_1 = 1.0$		
	local search r.	RK	D	IP	RK	D	IP	RK	D	IP
$u = 3$	0.000	1.0267 **	**1.0231**	1.0265 **	1.0207 *	**1.0180**	1.0198 *	1.0194 **	**1.0174**	1.0188 *
	0.001	1.0226 **	**1.0160**	1.0193 **	1.0145 **	**1.0117**	1.0127	1.0128 *	1.0112	**1.0111**
	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 *
	0.080	1.0019 **	**0.9978**	0.9980	0.9992 **	**0.9978**	0.9982 **	1.0004 **	**0.9992**	0.9993
	0.100	1.0017 **	**0.9977**	0.9978	0.9990 **	**0.9977**	0.9980 **	1.0003 **	**0.9991**	0.9994 *
$u = 10$	0.000	1.0163 **	**1.0083**	1.0163 **	1.0207 **	**1.0148**	1.0204 **	1.0210 **	**1.0161**	1.0196 **
	0.001	1.0133 **	**0.9998**	1.0098 **	1.0140 **	**1.0075**	1.0116 **	1.0149 **	**1.0105**	1.0132 *
	0.005	1.0042 **	**0.9883**	0.9960 **	1.0059 **	**0.9988**	1.0021 **	1.0073 **	**1.0024**	1.0046 **
	0.010	0.9985 **	**0.9850**	0.9900 **	1.0027 **	**0.9967**	0.9988 **	1.0045 **	**1.0000**	1.0019 **
	0.020	0.9957 **	**0.9834**	0.9867 **	1.0007 **	**0.9961**	0.9969 **	1.0031 **	**0.9993**	1.0001 **
	0.040	0.9932 **	**0.9831**	0.9846 **	0.9992 **	**0.9959**	0.9962	1.0022 **	**0.9988**	0.9990
	0.060	0.9915 **	**0.9829**	0.9842 **	0.9988 **	**0.9960**	0.9961	1.0015 **	**0.9988**	0.9988
	0.080	0.9916 **	**0.9828**	0.9836 **	0.9983 **	**0.9959**	**0.9959**	1.0014 **	**0.9987**	0.9988
	0.100	0.9906 **	**0.9829**	0.9836 **	0.9984 **	**0.9958**	0.9959	1.0013 **	**0.9987**	0.9988

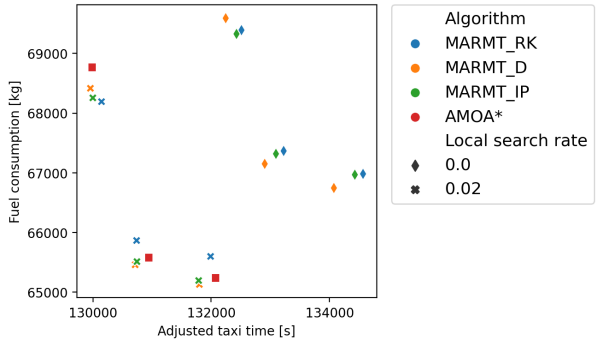

Fig. 7. Difference between the proposed algorithm with and without local search and AMOA*. Three different weights are used for reserving trajectories for individual aircraft. Each marker represents the average of 10 data points.

cance of this is stronger with $u = 10$. In the case of $u = 3$, MARMT-IP can also be competitive. This is especially the case when $w_1 = 1$, that is when the fastest trajectories are reserved for each aircraft. Table VII shows the average number of solutions in Θ_i, which also supports the superiority of MARMT-D. With the increasing local search rate, the number of Pareto optimal solutions increases, with the exception that the variants with the lowest local search rates produce less solutions than the ones without local search. In the case of $u = 3$, MARMT-D finds 9-12 solutions, while AMOA* finds 13-14. In the case of $u = 10$, the number of solutions obtained by MARMT-D is nearly twice as high as with $u = 3$.

We have seen that when computational time of MARMT is not limited, MARMT-D performs the best in terms of overall objective values, ε and the number of solutions in Θ_i. Higher local search rates generally lead to better solution quality. We have also seen that better solution quality can be reached with more parallel arcs (alternative speed profiles) in the multigraph. These results are important as they establish that MARMT is able to find Pareto fronts close to or better than those of AMOA*, when given enough time.

C. Potential for real time decision support

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 [65]. This time budget is used as the termination criteria in the following experiments. For routing the 506 aircraft, this amounts to 1.4 hours, which is lower than any computational times observed in Figure 6, therefore, we can expect compromised solution quality. Only results of MARMT-D are shown, without loss of generality. In Table VIII statistical significance of marginal improvement in mean RWA with increasing local search rates can be seen until the value of 0.02, for both values of u. Note, that even with the 10 seconds time budget, the mean RWA is within 1% of the mean RWA obtained by AMOA*. In Table IX we can see the number of solutions in Θ_i decrease as the local search rate increases, which is expected as local search is computationally intensive and thus less evaluations are performed in the same time budget. However, the lower local search rates below 0.02 lead to poorer objective values, which are the primary consideration. With the local search rate of 0.02, the average number of optimal solutions found is approximately half of the the number of optimal solutions found without limiting the computation time. It is not obvious from our experiments if including more speed profiles is worthwhile with small time budgets. The RWA is similar in the cases of $u = 3$ and $u = 10$. However, there are more solutions found in case of $u = 10$.

VII. CONCLUSION AND FUTURE DIRECTIONS

In this paper a metaheuristic solution approach has been proposed for the airport ground movement problem, as a representative of transportation problems with realistic modelling based on multigraphs. The adaptation includes modifying existing operators for the specific problem, incorporating time window constraints and constraint handling and proposing a local search operator for the problem. The proposed algorithms were evaluated using real data about one day of operation at

TABLE VI
MEAN ε INDICATOR FOR SEQUENTIAL ROUTING OF THE 506 AIRCRAFT WITH VARIED LOCAL SEARCH RATES.

		$w_1 = 0.0$			$w_1 = 0.5$			$w_1 = 1.0$		
		RK	D	IP	RK	D	IP	RK	D	IP
	local search r.									
$u = 3$	0.000	1.0348 *	**1.0325**	1.0351 *	1.0341 *	**1.0327**	1.0343	1.0345 **	**1.0320**	1.0346 **
	0.001	1.0324 **	**1.0269**	1.0291 *	1.0315 **	**1.0262**	1.0286 *	1.0311 **	**1.0252**	1.0280 **
	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
	0.080	1.0134 **	**1.0097**	**1.0097**	1.0119 **	**1.0080**	1.0085 **	1.0124 **	**1.0082**	1.0084 *
	0.100	1.0130 **	**1.0094**	**1.0094**	1.0117 **	**1.0079**	1.0082 *	1.0121 **	**1.0082**	1.0084
$u = 10$	0.000	1.0344 **	**1.0291**	1.0344 **	1.0334 **	**1.0279**	1.0343 **	1.0331 **	**1.0281**	1.0325 **
	0.001	1.0325 **	**1.0222**	1.0294 **	1.0293 **	**1.0206**	1.0264 **	1.0287 **	**1.0223**	1.0271 *
	0.005	1.0249 **	**1.0137**	1.0185 **	1.0213 **	**1.0111**	1.0159 **	1.0208 **	**1.0120**	1.0156 **
	0.010	1.0197 **	**1.0103**	1.0140 **	1.0175 **	**1.0079**	1.0110 **	1.0172 **	**1.0087**	1.0112
	0.020	1.0180 **	**1.0087**	1.0117 **	1.0151 **	**1.0066**	1.0088 **	1.0148 **	**1.0073**	1.0085 **
	0.040	1.0163 **	**1.0082**	1.0101 **	1.0128 **	**1.0062**	1.0071 **	1.0125 **	**1.0065**	1.0070 **
	0.060	1.0146 **	**1.0081**	1.0095 **	1.0122 **	**1.0062**	1.0068 **	1.0116 **	**1.0063**	1.0067 *
	0.080	1.0150 **	**1.0079**	1.0090 **	1.0115 **	**1.0061**	1.0065 **	1.0114 **	**1.0062**	1.0066 *
	0.100	1.0139 **	**1.0079**	1.0089 **	1.0117 **	**1.0060**	1.0064 **	1.0112 **	**1.0063**	1.0066

TABLE VII
MEAN NUMBER OF PARETO OPTIMAL SOLUTIONS FOUND FOR THE 506 AIRCRAFT WITH VARIED LOCAL SEARCH RATES.

		$w_1 = 0.0$			$w_1 = 0.5$			$w_1 = 1.0$		
		RK	D	IP	RK	D	IP	RK	D	IP
	local search r.									
$u = 3$	0.000	5.4	**9.9**	5.0	5.4	**9.8**	5.0	5.4	**9.9**	5.0
	0.001	5.2	**9.3**	4.9	5.2	**9.3**	4.9	5.2	**9.3**	4.9
	0.005	5.1	**9.5**	5.3	5.1	**9.4**	5.3	5.1	**9.5**	5.4
	0.010	5.3	**10.2**	5.9	5.3	**10.3**	6.0	5.3	**10.4**	6.0
	0.020	5.6	**11.1**	6.9	5.7	**11.3**	7.0	5.7	**11.2**	7.0
	0.040	6.2	**11.9**	7.9	6.2	**11.9**	7.9	6.3	**11.9**	7.9
	0.060	6.5	**12.1**	8.4	6.6	**12.2**	8.3	6.6	**12.3**	8.4
	0.080	6.7	**12.4**	8.6	6.6	**12.4**	8.6	6.7	**12.5**	8.7
	0.100	6.7	**12.5**	8.8	6.7	**12.5**	8.8	6.8	**12.6**	8.8
$u = 10$	0.000	6.7	**19.4**	6.1	6.8	**19.5**	6.1	6.8	**19.7**	6.2
	0.001	6.5	**17.9**	5.9	6.5	**18.1**	6.0	6.6	**18.0**	6.0
	0.005	6.4	**17.9**	6.5	6.4	**18.0**	6.5	6.5	**18.1**	6.6
	0.010	6.7	**18.9**	7.4	6.7	**19.2**	7.5	6.7	**19.4**	7.4
	0.020	7.2	**20.4**	8.7	7.2	**20.8**	8.8	7.3	**20.9**	8.9
	0.040	8.1	**21.8**	10.2	8.1	**22.3**	10.3	8.2	**22.1**	10.3
	0.060	8.4	**22.5**	11.1	8.5	**22.6**	11.3	8.5	**22.7**	11.3
	0.080	8.6	**22.7**	11.7	8.7	**23.1**	11.8	8.7	**23.0**	11.8
	0.100	8.6	**22.9**	12.2	8.6	**23.2**	12.2	8.7	**23.3**	12.2

Hong Kong International airport. Three genetic representations including the direct, the integer priority based and the random keys representations were compared. Based on the converged solution quality, MARMT-D proved to be the most effective at exploring the search space. MARMT-D found around twice as many Pareto optimal solutions as the other variants. This is likely due to (1) the direct representation being the only one that provides 1-to-1 encoding and (2) the adaptation of the crossover operator to the multigraph case providing more effective exploration of parallel arc sequences. The performance of MARMT-D is very close to a state-of-the-art enumerative solution approach even with a 10 seconds time budget, and it outperformed said enumerative approach when allowed to converge. 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 great improvements in terms of all considered measures of solution quality with the use of the local search operator for the case of convergence based termination. With the 10 seconds time budget as the stopping criteria, local search improved objective values. However, with increasing local search rate the number of Pareto optimal solutions decreased. Including more speed profiles slightly improved solution quality with convergence termination in terms of all considered measures. The same conclusion could not be drawn for the 10 seconds time budget, where only the number of Pareto optimal solutions increases. Possibly, the flexibility allowed by more speed profiles cannot be fully capitalised, because the order of aircraft is fixed beforehand, as explained in [66]. This question remains to be explored in future research.

This study based on a medium sized airport considering two objectives shows great potential for reaching real-time decision support with MARMT. A natural progression of this work is to investigate larger airports, scenarios with denser traffic and including emissions as a third objective. There is a high interest in many-objective shortest path problems, in

TABLE VIII
MEAN RWA FOR THE 506 AIRCRAFT ROUTED BY MARMT-D WITH 10 SECONDS TIME BUDGET WITH VARIED LOCAL SEARCH RATES.

	local search r.	0.0	0.001	0.005	0.01	0.02	0.04	0.06	0.08	0.1
$u = 3$	$w_1 = 0.0$	1.0239	1.0194	1.0126	1.0094	1.0096	1.0103	1.0096	1.0095	**1.0091**
	$w_1 = 0.5$	1.0182	1.0139	1.0060	1.0039	**1.0031**	1.0034	1.0033	**1.0031**	1.0040
	$w_1 = 1.0$	1.0180	1.0131	1.0065	1.0049	**1.0034**	1.0038	1.0040	1.0044	1.0040
$u = 10$	$w_1 = 0.0$	1.0096	1.0051	0.9974	0.9965	**0.9948**	0.9966	0.9975	0.9992	0.9978
	$w_1 = 0.5$	1.0155	1.0110	1.0043	1.0026	**1.0025**	**1.0025**	1.0035	1.0030	1.0034
	$w_1 = 1.0$	1.0163	1.0124	1.0062	1.0054	**1.0045**	**1.0045**	1.0054	1.0058	1.0047

TABLE IX
MEAN NUMBER OF PARETO OPTIMAL SOLUTIONS FOR THE 506 AIRCRAFT ROUTED BY MARMT-D WITH 10 SECONDS TIME BUDGET.

	local search r.	0.0	0.001	0.005	0.01	0.02	0.04	0.06	0.08	0.1
$u = 3$	$w_1 = 0.0$	9.0	8.7	7.6	7.2	6.5	5.9	5.5	5.3	5.3
	$w_1 = 0.5$	8.9	8.6	7.7	7.3	6.7	6.0	5.6	5.4	5.2
	$w_1 = 1.0$	9.0	8.7	7.7	7.3	6.6	6.1	5.6	5.3	5.2
$u = 10$	$w_1 = 0.0$	16.2	15.4	12.7	11.3	9.7	8.3	7.3	6.7	6.6
	$w_1 = 0.5$	16.1	15.1	12.6	11.3	9.7	8.2	7.2	6.9	6.6
	$w_1 = 1.0$	16.4	15.2	12.8	11.2	9.7	8.5	7.2	6.9	6.8

line with the more realistic and detailed modelling of routing problems. A recent benchmark suite is provided in [17] 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 solve 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 leading to slow convergence. Strategies aimed at ensuring the modification of the encoded solution might mitigate some of the disadvantage of the ambiguity associated with the priority based representations.

ACKNOWLEDGEMENT

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. SIP20211847. Carlos A. Coello Coello acknowledges support from the CONACyT project no. 1920, a 2018 SEP-Cinvestav grant (application no. 4), and the Basque Government through the BERC 2018-2021 program by the Spanish Ministry of Science.

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] J. Hartmanis, "Computers and intractability: a guide to the theory of np-completeness (michael r. garey and david s. johnson)," *Siam Review*, vol. 24, no. 1, p. 90, 1982.

[14] 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.

[15] 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.

[16] 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.

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

[18] 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.

[19] 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.
[20] 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.
[21] F. Neri and C. Cotta, "Memetic algorithms and memetic computing optimization: A literature review," *Swarm and Evolutionary Computation*, vol. 2, pp. 1–14, 2012.
[22] C.-K. Goh, Y.-S. Ong, and K. C. Tan, *Multi-objective memetic algorithms*. Springer, 2008, vol. 171.
[23] 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.
[24] 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.
[25] 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.
[26] 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.
[27] E. Q. V. Martins, "On a multicriteria shortest path problem," *European Journal of Operational Research*, vol. 16, no. 2, pp. 236–245, 1984.
[28] 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.
[29] E. W. Dijkstra *et al.*, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.
[30] 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.
[31] N. Shi, S. Zhou, F. Wang, Y. Tao, and L. Liu, "The multi-criteria constrained shortest path problem," *Transportation Research Part E: Logistics and Transportation Review*, vol. 101, pp. 13–29, 2017.
[32] 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.
[33] 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.
[34] 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.
[35] 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.
[36] 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.
[37] 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.
[38] 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.
[39] 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.
[40] 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.
[41] 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.
[42] 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.
[43] C. Lesire, "An iterative a* algorithm for planning of airport ground movements." in *ECAI*, vol. 2010, 2010, pp. 413–418.
[44] 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.
[45] L. Mandow, J. P. De la Cruz *et al.*, "A new approach to multiobjective a* search." in *IJCAI*, vol. 8. Citeseer, 2005.
[46] 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.
[47] 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.
[48] 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.
[49] 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.
[50] 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.
[51] ——, "Ship speed optimization: Concepts, models and combined speed-routing scenarios," *Transportation Research Part C: Emerging Technologies*, vol. 44, pp. 52–69, 2014.
[52] 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.
[53] M. Enzi, S. N. Parragh, and J. Puchinger, "The bi-objective multimodal car-sharing problem," *arXiv preprint arXiv:2010.10344*, 2020.
[54] 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.
[55] 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.
[56] 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.
[57] 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.
[58] 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.
[59] 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.
[60] A. Homaifar, C. X. Qi, and S. H. Lai, "Constrained optimization via genetic algorithms," *Simulation*, vol. 62, no. 4, pp. 242–253, 1994.
[61] 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.
[62] "This research utilised queen mary's apocrita hpc facility, supported by qmul research-it." http://doi.org/10.5281/zenodo.438045.
[63] "Inspyred: Bio-inspired algorithms in python," https://pythonhosted.org/inspyred/, accessed: 2019-10-30.
[64] 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.
[65] "Icao, 2004. advanced surface movement guidance and control systems (a-smgcs) manual. international civil aviation organization," http://www.icao.int/Meetings/anconf12/Document.
[66] M. Weiszer, E. K. Burke, and J. Chen, "A note on multi-graph structure for constrained routing and scheduling problems," *Unpublished manuscript*, vol. -, pp. –, 2021.