Data-Driven Merge-Point Selection in Dynamic-Aircraft-Arrival-Routes Optimization*

Roghayeh Hajizadeh[1][0000−0002−7402−6292], Maksym Moroz[2][0009−0007−5392−7466], Tatiana Polishchuk[2][0000−0002−9869−5992], Elina Rönnberg[1][0000−0002−2081−2888], and Christiane Schmidt[2][0000−0003−2548−5756]

[1] Department of Mathematics (MAI), Linköping University, Sweden
[2] Department of Science and Technology (ITN), Linköping University, Sweden
firstname.lastname@liu.se

Abstract. With dynamic arrival routes (DARs) we can route and schedule aircraft arriving at an airport. DARs entail an arrival tree that merges all arriving traffic during a given time period towards the runway, such that we can guarantee temporal separation and optimal descent profiles for all the arriving aircraft. In this paper, we aim to reduce the complexity of the resulting arrival trees. With DARs, we aim to answer the need for reduced environmental impact while alleviating air-traffic-controller workload through the automated separation. While DARs provide a high degree of flexibility—they allow to accommodate even high-traffic scenarios without reducing capacity—the points in which the entry-point–runway paths merge along the tree, so-called merge points, can be located differently in the terminal maneuvering area (TMA) during different time periods. However, switching these merge points often yields high complexity for air traffic controllers. In this paper, we propose a data-driven approach to select a candidate set of merge points, integrate the restriction to this candidate set into the mathematical framework for DARs, and demonstrate our approach on an example of a realistic high-traffic scenario at Stockholm Arlanda airport. We are able to obtain feasible solutions even for very few merge-point candidates, with a tradeoff of solution quality/performance in the TMA and the complexity induced by the number of merge-point candidates.

Keywords: Dynamic aircraft arrival routes · Integer programming · Data-driven · Candidate merge points.

1 Introduction

A central problem in aviation is to reduce the environmental impact [1, 8], especially with the projected growth in air travel demand (with an average annual growth rate of 4.3% [7]). While this goal is dominant in air traffic in general, a

* This research was supported by grant 2022-03178 (OPT@ATM: Optimization Methods for Large Air Traffic Management Problems) from the Swedish Research Council and by the Swedish Transport Administration via the Friendly TMA project.

specific focus is on terminal maneuvering areas (TMAs), where all traffic converges: environmentally friendly, fuel- and noise-optimized descent profiles can curtail emissions and noise, however, they are not supported by today's strategic standard arrival routes. Hence, air traffic controllers (ATCOs) may increase separation buffers, which decreases throughput, or issue instructions to deviate from optimal trajectories, which maintains throughput but does not leverage the environmental benefits. Any solution must also take the work of the ATCOs into account: the increasing traffic raises ATCOs' workload and the complexity of their task, possibly impacting safety.

A promising way to reduce both complexity and workload is to design arrival-route structures that support automated separation while keeping high throughput. Fixed topologies such as point merge (PM) [3] and trombone procedures [15] have been studied as means to support path stretching and shortcutting. PM is expected to improve both controller workload and environmental performance [2, 3]. For PM, Hardell et al. [6] showed that all arrivals can follow the most fuel-efficient continuous descent operations (CDOs) while improving time and distance in TMA. Sáez et al. [12] provided a similar analysis for trombones and showed that, for Frankfurt am Main airport, all aircraft under low traffic but not under high traffic can be scheduled with CDOs. Liu et al. [9] showed PM outperforms trombones in capacity and procedure. These concepts, however, rely on fixed structures and therefore offer limited flexibility.

For this reason, we have studied dynamic arrival routes (DARs), a route structure in which aircraft from different entry points are guided along paths. The entry-point–runway paths are required to form an arrival tree: each entry point has exactly one path to the runway, and whenever two paths merge, they remain merged all the way to the runway. Intuitively, the structure looks like a tree that starts at the runway (root) and branches out toward the entry points (leaves). Figure 1 shows an example of an arrival tree for Stockholm Arlanda airport, where traffic from the airport's four entry points is merged toward the runway. The tree structure allows the route geometry to adapt to changing traffic patterns while still maintaining automated separation. The dynamic aspect stems from the fact that the paths constituting the tree may change from one time period to another, resulting in different arrival trees for different traffic situations. An operational concept for DARs was proposed by Sáez et al. [16], and our most recent mathematical model based on a Dantzig–Wolfe reformulation was presented in [4, 5]. In contrast to PM and trombone procedures, DARs can occupy any part of the TMA and therefore better adapt to varying traffic situations. However, we pay for those dynamics with reduced predictability and increased complexity: traffic from different entry points can merge at very different points at different times.

In this paper, we reduce that complexity by restricting merge points to a candidate set—a suggestion from operational ATCOs. This induces three questions: How to identify candidate merge points, how to integrate them into the mathematical model, and what performance impact they have? We answer the first question with a proof-of-concept data-driven approach, the second by a small

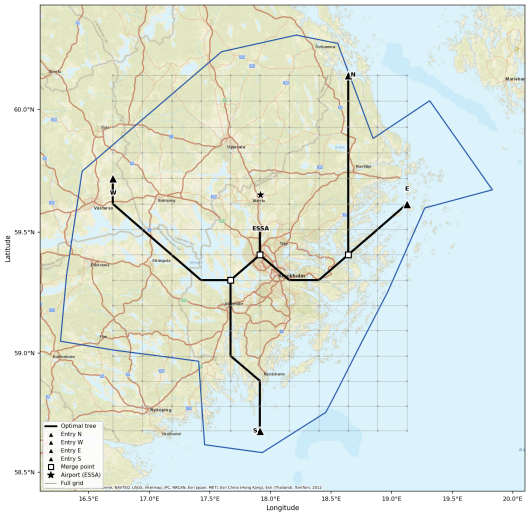


Fig. 1: An arrival tree for Stockholm Arlanda airport

modification of the model, and the third by computing DARs for Stockholm Arlanda airport and studying how the number of candidates affects the results.

The remainder of the paper is organized as follows. Section 2 defines the problem, Section 3 presents the optimization framework, Section 4 proposes the data-driven candidate selection approach, Section 5 presents the experiments, and Section 6 concludes the paper.

2 Problem Formulation

We are given the locations of TMA entry points and the runway, a set of aircraft planned for arrival within a given time interval, the planned arrival time at the entry point for each aircraft, path-dependent descent profiles for these aircraft, and separation requirements. We aim to compute:

A. An optimal arrival tree, such that all aircraft follow the path in the tree from their entry point to the final approach fix, from where they will proceed to the runway. We usually refer to these as entry-point–runway paths.
B. A set of points, $\mathcal{M}$, in the TMA that serves as a candidate set for merge points, where the traffic from the different entry points merges to the runway.

The tree must fulfill various conditions:

1. No more than two routes merge at a point from $\mathcal{M}$ and no routes merge at any point not in $\mathcal{M}$ (to limit complexity for ATCOs).
2. Merge points are separated by a minimum distance D, [11] (to prevent many merge points in close vicinity).
3. Routes do not make sharp turns in order to respect aircraft dynamics.

4. Obstacles, like no-fly zones, are avoided.
5. Temporal separation depending on the wake-turbulence categories of leading and trailing arriving aircraft is ensured.
6. All aircraft arrive to their entry points within a given time interval around the planned time.
7. All aircraft follow a set of descent profiles; in this paper, all aircraft follow optimal descent profiles.

Our tree should be evaluated w.r.t. two criteria: the total distance flown in the TMA (short flight routes for the aircraft), and the weight of the tree (low complexity for the ATCOs and a smaller noise-affected area).

3 Dynamic-Arrival-Routes Framework

In this section, we describe the components of our optimization framework. We review our path-based model from [5] in Subsection 3.1 to make this paper self-contained and in Subsection 3.2, we present our approach to restricting the merge points of the arrival tree for the DARs to a candidate set $\mathcal{M}$.

3.1 The Path-Based Model

In this subsection, we present the path-based model for the case in which the merge points are not restricted to a candidate set—instead, we only require that no more than two routes merge at any point. However, all the other "ingredients" remain the same even after we introduce the candidate set.

Our model is a path-based model using paths in a graph obtained by discretizing the TMA into a square grid of size $q \times n$ with node set V, where the side length of each grid pixel equals the minimum separation distance D. This Selection also ensures that merge points are separated by at least D (Condition 2, Section 2). We snap all entry points and the runway to the grid and consider the bi-directed grid graph on V, in which each node is connected to its eight neighbors with edges in E. We delete all edges that interact with obstacles (Condition 4), all incoming edges of entry points, and all outgoing edges of the runway. Let l_{ij} denote the length of edge $(i,j) \in E$, $\mathcal{P} \subseteq V$ the set of entry points, and $r \in V$ the runway. Finally, let $V' = V \setminus \{\text{last grid row}\} \setminus \{\text{last grid column}\}$.

We further need some notation on the aircraft we aim to schedule. We denote the set of aircraft arriving to entry point $b \in \mathcal{P}$ by $\mathcal{A}_b$, the set of all aircraft by $\mathcal{A} = \bigcup_{b \in \mathcal{P}} \mathcal{A}_b$, and the planned arrival time of aircraft a at its entry point by $\bar{t}_a$. Since these fixed times may cause infeasibility, we allow an aircraft to be scheduled to arrive at any point in an interval $[\bar{t}_a - \mu, \bar{t}_a + \mu]$ for $\mu \geq 0$. Additionally, to limit the time horizon for which the schedule needs to be computed, we denote an upper bound on the time in which all considered aircraft land as $\overline{T}$. We discretize the time and denote the set of time stamps by $T = \{0, \ldots, \overline{T}\}$.

In order to provide automatic separation, let $\mathcal{C} = \{1, 2, \ldots, n_c\}$ be the set of indices of all different disjoint wake-turbulence categories. We denote by C_κ for

$\kappa \in \mathcal{C}$ a set of aircraft from the same category. Hence, we have $\mathcal{A} = \bigcup_{\kappa \in \mathcal{C}} C_\kappa$. We denote the required temporal separation for a leading aircraft from C_{κ_1} and a trailing aircraft from C_{κ_2} by $\sigma_{\kappa_1,\kappa_2}$. Let $\Omega = \max_{\kappa_1,\kappa_2 \in \mathcal{C}} \quad \sigma_{\kappa_1,\kappa_2}$.

In our path-based model, we select a unique path from each entry point to the runway. By restricting how these paths may interact, we enforce Conditions 2–7 and, in the original model, an adapted Condition 1, which only requires that no more than two routes merge at any point. We use one variable per path and therefore generate all feasible paths a priori. Feasibility of a path is determined by the path length, with an upper bound λ on the number of grid-graph edges, and by the angle between consecutive path edges, to ensure Condition 3. We generate the paths with an angle-limited depth-first search from each entry point to the runway, and improve runtime by pruning clearly infeasible branches: we precompute shortest paths, in number of edges, from each grid point to the runway, and cut off a branch at node v whenever the number of edges from the entry point to v plus the precomputed shortest path from v to r exceeds λ.

Since the length of each path is known, we can specify the optimal speed profile of each aircraft on each path. In this paper, we compute descent profiles under international standard atmosphere conditions and no wind for each entry-point–runway path length, assuming neutral CDOs. We employ the method from [16] to determine the optimal vertical profile for each distance to go.

We define some sets based on the paths. We denote the set of paths from entry point b to the runway by Π_b, the set of all paths crossing node $i \in V$ by $\overline{\Pi}_i$, the set of all paths by $\mathbf{\Pi} = \bigcup_{b \in \mathcal{P}} \Pi_b$, and the set of edges that path π passes through for any $\pi \in \mathbf{\Pi}$ by θ_π. Moreover $\xi_{a,\pi,i}$ denotes the time when aircraft a with entry time $\bar{t}_a$ occupies node i on path π based on our pre-computed descent profiles, $\Xi_{a,i,t} = \{\pi \in \overline{\Pi}_i \cap \Pi_b : \quad \xi_{a,\pi,i} = t\}$ denotes the set of all paths on which aircraft a with entry time $\bar{t}_a$ occupies node i at time t, and $\Upsilon_{a,i,t,\sigma} = \{\pi \in \overline{\Pi}_i \cap \Pi_b : \quad t \leq \xi_{a,\pi,i} \leq t + \sigma - 1\}$ denotes the set of all paths on which aircraft a with entry time $\bar{t}_a$ occupies node i sometime between time t and $t + \sigma - 1$. We employ $\Xi_{a,i,t}$ and $\Upsilon_{a,i,t,\sigma}$ to enforce the separation minima (Condition 6). Finally, we introduce binary variables:

- ρ_π indicates whether path π is used in the arrival tree.
- $x_{i,j}$ indicates whether edge $(i,j) \in E$ is used in the arrival tree.
- $\tau_{a,\pi,t}$ indicates whether aircraft a uses path π and arrives at its entry point at time t (to handle flexible entry times).

Using the defined sets and parameters, and introduced variables, we present the path-based formulation, model $\mathrm{M}_{\mathrm{all}}$ given in [5], as:

$$\min \quad \beta \sum_{(i,j) \in E} l_{i,j} x_{i,j} + (1-\beta) \sum_{b \in \mathcal{P}} \sum_{\pi \in \Pi_b} \sum_{(i,j) \in \theta_\pi} |\mathcal{A}_b| l_{i,j} \rho_\pi \tag{1}$$

$$\text{s.t.} \quad \sum_{j:(j,i) \in E} x_{j,i} \leq 2, \qquad \forall i \in V \setminus \{\mathcal{P} \cup r\}, \tag{2}$$

$$\sum_{j:(i,j) \in E} x_{i,j} \leq 1, \qquad \forall i \in V \setminus \{\mathcal{P} \cup r\}, \tag{3}$$

$$x_{i,i+1+n} + x_{i+1+n,i} + x_{i+n,i+1} + x_{i+1,i+n} \leq 1,$$

$$\forall i \in V' \setminus \{\mathcal{P} \cup r\} : i+1+n, i+n, i+1 \notin \{\mathcal{P} \cup r\}, \quad (4)$$

$$x_{i,i+1+n} + x_{i+n,i+1} + x_{i+1,i+n} \leq 1, \qquad \forall i \in \mathcal{P} \cap V', \quad (5)$$

$$x_{i,i+1+n} + x_{i+1+n,i} + x_{i+1,i+n} \leq 1, \qquad \forall i : i+1 \in \mathcal{P}, \quad (6)$$

$$x_{i,i+1+n,} + x_{i+n+1,i} + x_{i+n,i+1} \leq 1, \qquad \forall i : i+n \in \mathcal{P}, \quad (7)$$

$$x_{i+1+n,i} + x_{i+n,i+1} + x_{i+1,i+n} \leq 1, \qquad \forall i : i+n+1 \in \mathcal{P}, \quad (8)$$

$$\sum_{\pi \in \Pi_b} \rho_\pi = 1, \qquad \forall b \in \mathcal{P}, \quad (9)$$

$$\rho_\pi \leq x_{i,j}, \qquad \forall \pi \in \mathbf{\Pi}, \forall (i,j) \in \theta_\pi, \quad (10)$$

$$\sum_{\pi \in \Pi_b} \sum_{t=\bar{t}_a-\mu}^{\bar{t}_a+\mu} \tau_{a,\pi,t} = 1, \qquad \forall b \in \mathcal{P}, \forall a \in \mathcal{A}_b, \quad (11)$$

$$\sum_{t=\bar{t}_a-\mu}^{\bar{t}_a+\mu} \tau_{a,\pi,t} = \rho_\pi, \qquad \forall b \in \mathcal{P}, \forall a \in \mathcal{A}_b, \forall \pi \in \Pi_b, \quad (12)$$

$$\sum_{b \in \mathcal{P}} \sum_{a \in \mathcal{A}_b \cap C_{\kappa_2}} \sum_{t'=-\mu}^{\mu} \sum_{\pi \in \Upsilon_{a,i,t-t',\sigma_{\kappa_1,\kappa_2}}} \tau_{a,\pi,\bar{t}_a+t'} \leq$$

$$\Omega(1 - \sum_{b \in \mathcal{P}} \sum_{a' \in \mathcal{A}_b \cap C_{\kappa_1}} \sum_{t'=-\mu}^{\mu} \sum_{\pi' \in \Xi_{a',i,t-t'}} \tau_{a',\pi',\bar{t}_{a'}+t'}),$$

$$\forall \kappa_1, \kappa_2 \in \mathcal{C} : \kappa_1 \neq \kappa_2, \ \forall i \in V, \ \forall t \in \{0, \ldots, \overline{T} - \sigma_{\kappa_1,\kappa_2} + 1\}, \quad (13)$$

$$\sum_{b \in \mathcal{P}} \sum_{a \in \mathcal{A}_b \cap C_\kappa : a \neq a'} \sum_{t'=-\mu}^{\mu} \sum_{\pi \in \Upsilon_{a,i,t-t',\sigma_{\kappa,\kappa}}} \tau_{a,\pi,\bar{t}_a+t'} \leq \Omega(1 - \sum_{t'=-\mu}^{\mu} \sum_{\pi' \in \Xi_{a',i,t-t'}} \tau_{a',\pi',\bar{t}_{a'}+t'}),$$

$$\forall \kappa \in \mathcal{C}, \forall a' \in C_\kappa, \forall i \in V, \ \forall t \in \{0, \ldots, \overline{T} - \sigma_{\kappa,\kappa} + 1\}, \quad (14)$$

$$x_{i,j} \quad \text{binary}, \qquad \forall (i,j) \in E, \quad (15)$$

$$\rho_\pi \quad \text{binary}, \qquad \forall \pi \in \mathbf{\Pi}, \quad (16)$$

$$\tau_{a,\pi,t} \quad \text{binary}, \qquad \forall b \in \mathcal{P}, \forall a \in \mathcal{A}_b, \forall t \in [\bar{t}_a - \mu, \bar{t}_a + \mu]. \quad (17)$$

The objective function (1) is a convex combination of the paths length and the tree weight. Constraints (2) and (3) ensure that no more than two routes merge at any point—the version without restricting the merge points to the set $\mathcal{M}$. Although degree constraints prevent crossings at nodes, additional constraints such as temporal separations may cause crossings within grid squares, which are prevented by Constraints (4)–(8). On the path level, Constraint (9) ensures that exactly one path is used from each entry point, and Constraint (10) couples the path variables with the selected grid-graph edges. On the aircraft level, Constraints (11) and (12) ensure that exactly one trajectory is chosen for each aircraft. (Constraints (12) and (9) yield Constraint (11), but explicitly including it improved computational performance). Constraints (13) and (14) enforce temporal separations according to the wake-turbulence categories of the leading and trailing aircraft, handling aircraft from different and coinciding categories, respectively. Finally, Constraints (15)–(17) define the domains of variables.

3.2 Integrating the Set of Candidate Merge Points in the Model

As given by Condition 1 for the arrival tree in Section 2, we want to limit the merge points of the tree to nodes from a candidate set $\mathcal{M}$. In model $\mathrm{M}_{\mathrm{all}}$, we allowed all grid nodes (except for the entry points and the runway) to be merge

points, by limiting the indegree of any grid node by 2 with Constraint (2). Hence, we can easily integrate the constraint on the set $\mathcal{M}$: We allow at most 2 incoming edges for any node in $\mathcal{M}$ and at most 1 incoming edge for all other nodes. To do so, we substitute Constraint (2) by Constraints (18) and (19).

$$\sum_{j:(j,i)\in E} x_{j,i} \leq 2, \qquad \forall i \in \mathcal{M} \tag{18}$$

$$\sum_{j:(j,i)\in E} x_{j,i} \leq 1, \qquad \forall i \in V \setminus \mathcal{M} \tag{19}$$

The objective function (1), and Constraints (3)–(16), (18)–(19) yield our model for the case that a set of candidate merge points is given—$\mathrm{M}_{\mathrm{cand}}$.

4 Data-Driven Approach to Select the Set of Candidate Merge Points

We propose a data-driven approach to select the set of candidate merge points $\mathcal{M}$, as a proof-of-concept reflecting the data available in practice, to compute (B) in Section 2. We run model $\mathrm{M}_{\mathrm{all}}$ (1)–(17) on a large set of instances $\mathcal{I}$ and track how many times each node appeared as a merge point in the resulting arrival trees. Let m_v denote this number for all $v \in V \setminus \{\mathcal{P} \cup r\}$:

$$m_v = |\{I \in \mathcal{I} |\ v \text{ is merge point in optimal tree for } I\}| \tag{20}$$

We then define $\mathcal{M} := \{v \in V | m_v \geq \bar{m}\}$ for a threshold $\bar{m}$ and use our model $\mathrm{M}_{\mathrm{cand}}$ for this set for deciding on the actual DARs and the underlying tree.

Note that using only a subset of all grid nodes does not guarantee a global optimum for any input instance. Our design selects candidate merge points that (at all or often, for larger values of $\bar{m}$) contributed to global optimal trees for instances from $\mathcal{I}$. If $\mathcal{I}$ with our choice of $\bar{m}$ gives a good representation of the possible traffic situations in the TMA, we often get optimal trees when using these candidates to compute DARs and the underlying arrival tree for a period.

5 Experiments: Stockholm Arlanda Airport

In our experiments, we use data for Stockholm Arlanda obtained from the open-source database of the Opensky Network [10]. The dataset contains 33 aircraft arriving to runway 19L (one of the three runways of Arlanda airport), and the four main entry points HMR (north), XILAN (east), NILUG (south), and ELTOK (west) during the busiest hour of operation in 2018. We use an 11×15 grid, which ensures a merge-point separation of 6 NM. Moreover, we generate all feasible entry-point–runway paths with at most $\lambda = 14$ edges, enforcing a minimum turning angle of 135° between consecutive edges. This results in 3821 feasible paths, computed in less than one second.

Aircraft are classified according to the ICAO wake-turbulence categories [13]: LIGHT (L), MEDIUM (M) and HEAVY (H). In the experiments, we aggregate these categories into two groups, $C_1 = \{H, M\}$ and $C_2 = \{L\}$. Following ICAO separation minima [14], we set the temporal separation parameters as $\sigma_{1,2} = 3$ minutes and $\sigma_{1,1} = \sigma_{2,2} = \sigma_{2,1} = 2$ minutes. Finally, we set $\beta = 0.1$.

In Subsection 5.1, we describe how we obtain a set of instances $\mathcal{I}$ and the resulting set of candidate merge points, and in Subsection 5.2, we present the results for the DARs for different sets of $\mathcal{M}$ based on different values of $\bar{m}$.

5.1 Selection of Merge-Point Candidates

We can generate the set of instances $\mathcal{I}$ either from historic instances or by creating artificial instances from historic data. In this paper, we use the latter to obtain a large variety of arrival trees, based on 33 aircraft from real-world arrivals at Stockholm Arlanda TMA during the busiest hour in 2018: May 16, 5:00–5:59AM.

Algorithm 1: Computation of m_v, $\forall v \in V$.

$m_v = 0$, $\forall v \in V \setminus \{\mathcal{P} \cup r\}$;
$N = |\mathcal{I}|/3$;
for $i = 1, \ldots, N$ **do**
  $k_i \leftarrow$ uniformly at random from $\{7, 8, \ldots, 17\}$;
  Uniformly at random draw a subset $A_i \subset \overline{A}$ with $|A_i| = k_i$;
  for $a \in A_i$ **do**
    $\bar{t}_a \leftarrow$ uniformly at random from $[5{:}30, 5{:}59]$;
  for $\mu = 0, 2, 4$ **do**
    % Input instance I_i^μ is defined;
    Run $\mathrm{M}_{\mathrm{all}}$ on I_i^μ (with parameter μ) to obtain tree T_i^μ;
    if I_i^μ *is feasible* **then**
      for $v \in V \setminus \{\mathcal{P} \cup r\}$ **do**
        if v *is merge point in* T_i^μ **then**
          m_v ++;

From the 33 aircraft, we uniformly at random pick 3 aircraft and artificially model them as light aircraft (light aircraft are hardly present in the traffic mix at Stockholm Arlanda). Let the set of all 33 aircraft (including 3 light aircraft) be $\overline{A}$. We randomly generate the input instances $\mathcal{I}$ for the time window 5:30-5:59AM, during which 17 aircraft arrived in the real-world data, by Algorithm 1. For each selected subset of aircraft A_i and each value of μ, Algorithm 1 creates instances I_i^μ with corresponding arrival times, which we then solve using model $\mathrm{M}_{\mathrm{all}}$.

Algorithm 1 is implemented in Python 3.9.6. We solve the MIP models with Gurobi 13.0.1. All runs were executed on a MacBook Pro M4 (2025).

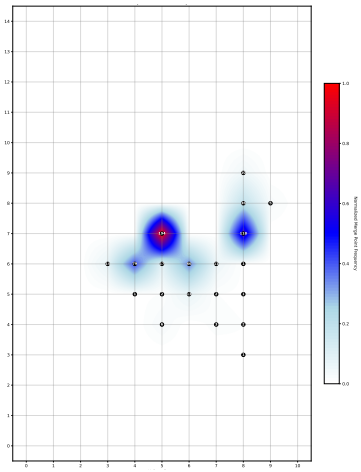


Fig. 2: Heatmap of merge points of all arrival trees computed for the instances $\mathcal{I}$

We ran the algorithm with $|\mathcal{I}| = 300$. The resulting heatmap for used merge points is presented in Figure 2, where each grid node is labeled with the number of times it appears as a merge point in the generated trees. This set of instances resulted in 20 candidate merge points, i.e, for all feasible instances I_i^{μ}, the merge points of the DARs computed with $\mathrm{M}_{\mathrm{all}}$ were a subset of these 20 grid points.

5.2 Dynamic Arrival Routes with Merge-Point Candidates

The implementation of the optimization model is done in Python 3.13.7. We solve the MIP models with Gurobi 12.0.3 using the `gurobipy` interface. All experiments were executed on a MacBook Pro M1 (2020).

In this paper, we present a proof of concept for restricting arrival-tree merge points to a grid subset of candidates, with the focus on the feasibility of computing this set and integrating it into the optimization framework. In real-world applications, the set of aircraft used in the instances in $\mathcal{I}$ should reflect both the traffic mix of the airport and which aircraft types typically appear together in certain time intervals. Hence, we use the same real-world traffic data as in Subsection 5.1, i.e, 33 arrival aircraft in the Stockholm Arlanda TMA during the busiest hour of operation in 2018 (May 16, 05:00–05:59), now with the original arrival times to the entry points. In order to show that different aircraft type with different separation minima (which complicate scheduling) can be handled, we artificially model a random subset of nine aircraft as light aircraft.

We divide the data into two consecutive half-hour periods: 05:00–05:29 and 05:30–05:59. The first period contains 16 arrivals (6 modelled as light) and the second period contains 17 arrivals (3 modelled as light). And we compute the DARs for each of these periods separately (we refer to [5] for approaches of handling consistency between arrival trees for consecutive time periods).

Table 1: Results of the experiments for arrivals between 5:00 AM and 5:29 AM

$\bar{m}$	$\lvert\mathcal{M}\rvert$	μ	objective	tree weight	paths length	average time in TMA (m)	run time (s)
0	165	1	167.376	33.728	182.225	20.6	67
		2	149.026	31.142	162.125	13.5	70
		3	149.026	31.142	162.125	13.5	126
		4	149.026	31.142	162.125	13.5	245
		5	139.567	28.728	151.882	8.8	130
1	20	1	167.576	35.728	182.225	20.6	100
		2	149.026	31.142	162.125	13.5	55
		3	149.026	31.142	162.125	13.5	97
		4	149.026	31.142	162.125	13.5	183
		5	139.567	28.728	151.882	8.8	174
2	16	1	167.676	36.728	182.225	20.6	149
		2	149.026	31.142	162.125	13.5	62
		3	149.026	31.142	162.125	13.5	104
		4	149.026	31.142	162.125	13.5	181
		5	139.567	28.728	151.882	8.8	177
3	13	1	167.676	36.728	182.225	20.6	133
		2	149.026	31.142	162.125	13.5	56
		3	149.026	31.142	162.125	13.5	103
		4	149.026	31.142	162.125	13.5	183
		5	139.567	28.728	151.882	8.8	180
5	12	1	167.676	36.728	182.225	20.6	127
		2	149.026	31.142	162.125	13.5	54
		3	149.026	31.142	162.125	13.5	102
		4	149.026	31.142	162.125	13.5	205
		5	139.567	28.728	151.882	8.8	171
6	11	1	167.676	36.728	182.225	20.6	143
		2	149.026	31.142	162.125	13.5	51
		3	149.026	31.142	162.125	13.5	100
		4	149.026	31.142	162.125	13.5	192
		5	139.567	28.728	151.882	8.8	168
10	10	1	167.676	36.728	182.225	20.6	148
		2	149.026	31.142	162.125	13.5	49
		3	149.026	31.142	162.125	13.5	98
		4	149.026	31.142	162.125	13.5	187
		5	139.567	28.728	151.882	8.8	169
11	9	1	167.676	36.728	182.225	20.6	136
		2	149.026	31.142	162.125	13.5	70
		3	149.026	31.142	162.125	13.5	99
		4	149.026	31.142	162.125	13.5	157
		5	139.567	28.728	151.882	8.8	168
13	8	1	167.676	36.728	182.225	20.6	80
		2	149.026	31.142	162.125	13.5	62
		3	149.026	31.142	162.125	13.5	94
		4	149.026	31.142	162.125	13.5	170
		5	139.567	28.728	151.882	8.8	170
14	7	1	167.676	36.728	182.225	20.6	110
		2	149.026	31.142	162.125	13.5	64
		3	149.026	31.142	162.125	13.5	97
		4	149.026	31.142	162.125	13.5	201
		5	139.567	28.728	151.882	8.8	173
17	6	1	167.676	36.728	182.225	20.6	123
		2	151.446	32.970	164.61	13.5	55
		3	151.446	32.970	164.61	13.5	125
		4	151.446	32.970	164.61	13.5	263
		5	139.567	28.728	151.882	8.8	169

Table 1: Results of the experiments for arrivals between 5:00 AM and 5:29 AM

$\bar{m}$	$\lvert\mathcal{M}\rvert$	μ	objective	tree weight	paths length	average time in TMA (m)	run time (s)
36	5	1	167.676	36.728	182.225	20.6	108
		2	151.446	32.970	164.61	13.5	56
		3	151.446	32.970	164.61	13.5	126
		4	151.446	32.970	164.61	13.5	256
		5	139.567	28.728	151.882	8.8	168
66	4	1	167.676	36.728	182.225	20.6	136
		2	151.446	32.970	164.61	13.5	58
		3	151.446	32.970	164.61	13.5	119
		4	151.446	32.970	164.61	13.5	360
		5	139.567	28.728	151.882	8.8	179
79	3	1	171.486	37.556	186.367	20.6	300
		2	151.446	32.970	164.61	13.5	103
		3	151.446	32.970	164.61	13.5	120
		4	151.446	32.970	164.61	13.5	214
		5	139.567	28.728	151.882	8.8	182

Table 2: Results of the experiments for arrivals between 5:30 AM and 5:59 AM

$\bar{m}$	$\lvert\mathcal{M}\rvert$	μ	objective	tree weight	paths length	average time in TMA (m)	run time (s)
0	165	≤ 3	—				
		4	173.886	34.556	189.367	11.9	372
		5	154.867	28.728	168.882	9.3	232
1	20	4	173.886	34.556	189.367	11.9	465
		5	154.867	28.728	168.882	9.3	230
2	16	4	173.886	34.556	189.367	11.9	389
		5	154.867	28.728	168.882	9.3	264
3	13	4	173.886	34.556	189.367	11.9	359
		5	154.867	28.728	168.882	9.3	236
5	12	4	173.886	34.556	189.367	11.9	412
		5	154.867	28.728	168.882	9.3	232
6	11	4	173.886	34.556	189.367	11.9	477
		5	154.867	28.728	168.882	9.3	243
10	10	4	173.886	34.556	189.367	11.9	414
		5	154.867	28.728	168.882	9.3	232
11	9	4	173.886	34.556	189.367	11.9	452
		5	154.867	28.728	168.882	9.3	231
13	8	4	173.886	34.556	189.367	11.9	412
		5	154.867	28.728	168.882	9.3	230
14	7	4	173.886	34.556	189.367	11.9	367
		5	154.867	28.728	168.882	9.3	229
17	6	4	173.986	35.556	189.367	11.9	404
		5	154.867	28.728	168.882	9.3	228
36	5	4	173.986	35.556	189.367	11.9	425
		5	154.867	28.728	168.882	9.3	231
66	4	4	173.986	35.556	189.367	11.9	389
		5	154.867	28.728	168.882	9.3	250
79	3	4	173.986	35.556	189.367	11.9	371
		5	154.867	28.728	168.882	9.3	225

We construct the candidate set $\mathcal{M}$ for each run using the heatmap from Subsection 5.1. Since only 20 grid nodes appear as merge points for instances in $\mathcal{I}$, the candidate set does not change for every threshold value $\bar{m}$. We therefore consider only values for which $\mathcal{M}$ changes. For each such $\bar{m}$, we solve model $\mathrm{M}_{\mathrm{cand}}$ with different time-window offsets μ. We also include the case $\bar{m} = 0$, corresponding to model $\mathrm{M}_{\mathrm{all}}$, i.e., the unrestricted case, which yields a globally optimal solution for the considered data and serves as a reference for evaluating the impact of restricting the merge-point set.

Tables 1 and 2 report the computational results for the two time periods. Each table shows the threshold $\bar{m}$, the number of candidate points $|\mathcal{M}|$, the time-window offset μ, the total objective value, the tree weight, the total paths lengths, the average time (in minutes) each aircraft spends in the TMA, and the run time for solving $\mathrm{M}_{\mathrm{cand}}$ (in seconds). Since no feasible solutions exist for the second half-hour with $\mu \leq 3$, we do not include these cases in Table 2. We also provide the resulting optimal arrival trees for some of our results in Figures 3–4.

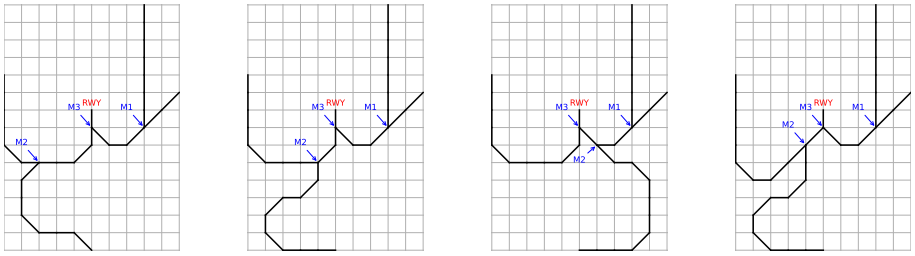
The parameter $\bar{m}$ controls the maximum size of the candidate set used when constructing merge points and, therefore, directly influences the feasible merge structures. When $\bar{m} = 0$, no candidate set is imposed and every node in the grid graph may serve as a merge point. As $\bar{m}$ increases, the set of admissible merge points becomes progressively restricted, which alters both the topology of the resulting merge tree and the flexibility in selecting aircraft paths.

Objective values may coincide for different values of $\bar{m}$ but this does not imply that the resulting trees are identical. For a given μ, a tree that is optimal for $\bar{m} = m^*$ is feasible also for all $\bar{m} < m^*$, and if the objective value coincides, it is also an optimal tree for that $\bar{m} < m^*$. However, the opposite is not true, and we may not always inherit trees from smaller values of $\bar{m}$ because some used merge-point candidates may no longer be present. Moreover, for relatively small $\bar{m}$, we may achieve the global optimal objective value. However, when $\bar{m}$ becomes large, deviations appear for some values of μ, leading to slightly larger objective values. This indicates that the restriction on merge points eventually affects the tree structure and overall solution quality. Hence, there is a tradeoff between the complexity for ATCOs (in terms of the number of grid points selected as merge-point candidates) and the quality of aircraft trajectories in the TMA.

The parameter μ introduces a second trade-off: increasing μ improves the objective value and reduces tree weight and average time in the TMA by allowing more flexible entry times, but at a cost for aircraft operators, as larger offsets may require speed adjustments or path stretching.

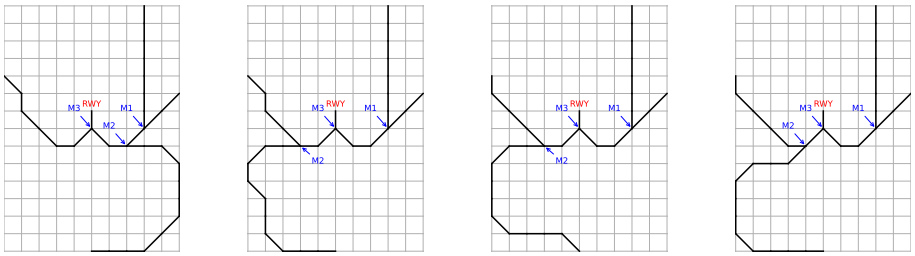
6 Conclusions

We presented an approach for reducing complexity of DARs for ATCOs by restricting merge points of the entry-point–runway paths to a set of candidates, following recommendations from operational ATCOs. We give a proof of concept of a data-driven approach to determine candidate merge points and of integrating this candidate set into our framework to compute DARs.



(a) (b) (c) (d)

Fig. 3: Optimal arrival trees for 5:00–5:29 with $\mu = 1$, (a) all points as candidates, (b)-(d) with merge point candidates obtained by $\bar{m} = 1, 11$, and 79, respectively



(a) (b) (c) (d)

Fig. 4: Optimal arrival trees for 5:30–5:59 with $\mu = 4$, (a) all points as candidates, (b)-(d) with merge point candidates obtained by $\bar{m} = 1, 11$, and 79, respectively

With our approach, we aim to balance the flexibility of DARs, which allows us to schedule and automatically separate all aircraft following environmentally friendly, fuel-optimized descent profiles even in high-traffic scenarios, and the complexity of the resulting arrival-route structure for ATCOs. Since merge points require the most attention of ATCOs, restricting them to a few spots has the potential to reduce complexity. Moreover, DARs with merge-point candidates still allow the arrival routes to occupy varying parts of the TMA during different time periods, which can accommodate a wide variety of traffic scenarios.

With experiments for Stockholm Arlanda airport, we verify that we can obtain feasible solutions for different cardinalities (with $|\mathcal{M}| \geq 3$) of the merge-point candidate set. We observe the expected tradeoff between complexity and the quality of aircraft trajectories: the more well-chosen merge-point candidates we have, the better the objective function and performance measures like the average time in TMA, but also the higher the complexity. Consequently, identifying an appropriate balance between the size of the candidate set, DARs performance and operational complexity, as well as between sequencing flexibility and schedule stability, remains a key direction for future research.

References

1. EASA: European aviaton environmental report 2025. https://www.eurocontrol.int/sites/default/files/2025-01/eurocontrol-easa-eaer-2025.pdf
2. Eurocontrol: Point merge implementation: A quick guide, simplifying and enhancing arrival operations with closed loop sequencing
3. Eurocontrol: Point merge, improving and harmonising arrival operations. https://www.eurocontrol.int/concept/point-merge
4. Hajizadeh, R., Hardell, H., Polishchuk, T., Rönnberg, E., Schmidt, C.: Integrating atmospheric conditions into the dynamic-arrival-routes framework. In: SIDs (2025)
5. Hajizadeh, R., Polishchuk, T., Rönnberg, E., Schmidt, C.: A Dantzig-Wolfe reformulation for automated aircraft arrival routing and scheduling. unpublished (2025), https://arxiv.org/abs/2509.13065
6. Hardell, H., Otero, E., Polishchuk, T., Smetanova, L.: Optimizing air traffic management through point merge procedures: Minimizing delays and environmental impact in arrival operations. Journal of Air Transport Management **123** (2025)
7. IATA: Global outlook for air transport: Highly resilient, less robust (2023)
8. ICAO: Resolution a41-21: Consolidated statement of continuing icao policies and practices related to environmental protection — climate change. https://www.icao.int/sites/default/files/sp-files/environmental-protection/Documents/Assembly/Resolution_A41-21_Climate_change.pdf
9. Liu, W., Delahaye, D., Cetek, F.A., Zhao, Q., Notry, P.: Comparison of performance between PMS and trombone arrival route topologies in terminal maneuvering area. Journal of Air Transport Management **115**, 102532 (2024)
10. OpenSky Network. https://opensky-network.org (Jul 2019), acc. December 2, 2018
11. Prete, J., Krozel, J., Mitchell, J., Kim, J., Zou, J.: Flexible, Performance-Based Route Planning for Super-Dense Operations. In: AIAA Guidance, Navigation and Control Conference and Exhibit. American Institute of Aeronautics and Astronautics (Aug 2008)
12. Sáez, R., Prats, X., Polishchuk, T., Polishchuk, V.: Traffic Synchronization in Terminal Airspace to Enable Continuous Descent Operations in Trombone Sequencing and Merging Procedures: An Implementation Study for Frankfurt Airport. Transportation Research Part C: Emerging Technologies **121** (2020)
13. SKYbrary: Icao wake turbulence categories. https://skybrary.aero/articles/icao-wake-turbulence-category
14. SKYbrary: Mitigation of wake turbulence hazard, wake turbulence separation minima. https://skybrary.aero/articles/mitigation-wake-turbulence-hazard, acc. Sept 22, 2025
15. Sprong, K., Haltli, B., DeArmon, J., Bradley, S.: Improving flight efficiency through terminal area rnav. In: 6th USA/Europe ATM Seminar (2005)
16. Sáez, R., Polishchuk, T., Schmidt, C., Hardell, H., Smetanová, L., Polishchuk, V., Prats, X.: Automated sequencing and merging with dynamic aircraft arrival routes and speed management for continuous descent operations. Transportation Research Part C: Emerging Technologies **132**, 103402 (2021)