

Graph-based Airspace Control and Path Optimization for Last-Mile Delivery Drones

Jorge Luiz Franco, Filipe Alves Neto Verri (advisor)

jorge.luiz@usp.br, filipe.verri@institutocurvelo.org.br

Abstract. *Drones in last-mile delivery introduce challenges in airspace management, collision avoidance, and route optimization. This MSc dissertation models the shared airspace as a time-expanded graph and formulates the Last-Mile Delivery Drone (LMDD) problem as a Multi-Agent Path Finding (MAPF) instance, solved exactly by a multi-commodity-flow Mixed-Integer Linear Program and, at scale, by a prioritized breadth-first heuristic running in $\mathcal{O}(P^{1.5}K)$ for P cells and K drones. Virtual per-drone source and sink vertices make variable take-off times, zero-cost loitering, and synchronized landings first-class in the flow model, and priority insertion supports emergency re-tasking at $\mathcal{O}(1)$ cost. The dissertation produced a Computers & Operations Research paper; its edge-centric methodology subsequently led to graph neural networks for flight-delay prediction (Physica A) and to a spectral edge-centrality theory for directed networks (under review).*

Candidacy Information

Candidate: Jorge Luiz Franco

Institution where the degree was obtained: Instituto Tecnológico de Aeronáutica (ITA), Graduate Program in Electronic and Computer Engineering (Informatics area), São José dos Campos, São Paulo, Brazil

Advisor: Prof. Dr. Filipe Alves Neto Verri, Instituto Tecnológico de Aeronáutica (ITA), São Paulo, Brazil

Track: MSc

Defense date: July 7, 2025

Members of the examination committee:

1. Prof. Dr. Elton Felipe Sbruzzi, Instituto Tecnológico de Aeronáutica (ITA), São Paulo, Brazil
2. Prof. Dr. Filipe Alves Neto Verri (advisor), Instituto Tecnológico de Aeronáutica (ITA), São Paulo, Brazil
3. Dr. Renato Tinós, University of São Paulo (USP), São Paulo, Brazil
4. Dr. Fabricio Aparecido Breve, São Paulo State University (UNESP), São Paulo, Brazil

Link to the full dissertation: <https://github.com/JorgeLuizFranco/MastersDissertationITA/blob/master/Disserta%C3%A7%C3%A3o%20Mestrado%20ITA/tese.pdf>

Links to the publications derived from the dissertation (journals and preprints): [Franco et al. 2025, Franco et al. 2026a, Franco et al. 2026b], detailed in Section 4.4 and in the Appendix

Link to the Google Scholar profile: <https://scholar.google.com/citations?user=qr3p7e4AAAAJ>

Link to the Lattes CV: <http://lattes.cnpq.br/4380015118739774>

Link to the DBLP profile: <https://dblp.org/pid/381/2895>

1. Introduction

The proliferation of UAVs in urban logistics raises a critical operational question: how can a centralized authority route hundreds of drones through shared low-altitude airspace without collisions, in polynomial time, while keeping total flight distance near-optimal? The dissertation answers it with a *centralized MAPF-based framework* that embeds collision avoidance and dynamic scheduling directly into the planning stage, and that pairs an exact method with a scalable one so that the same graph model serves both as a correctness benchmark and as a deployable planner. Its contributions are:

1. **LMDD-MAPF formulation.** A time-expanded graph model in which *variable take-off times*, *in-air loitering*, and *synchronized landings* are expressed directly by the graph topology, through virtual per-drone source and sink vertices, rather than by auxiliary constraints.
2. **Exact MILP method.** A multi-commodity integer-flow formulation over the time-expanded graph that returns collision-free, displacement-optimal plans for a given horizon, solved with Gurobi/JuMP and used as an exact benchmark (Section 3.3).
3. **Scalable prioritized-BFS heuristic.** A constraint-aware breadth-first planner with hash-based $\mathcal{O}(1)$ conflict checks, running in $\mathcal{O}(P^{1.5}K)$ time, which enables replanning for fleets of hundreds of drones (Section 3.4).
4. **Hybrid time-horizon refinement.** A binary-search procedure that exploits monotonicity of feasibility in the horizon to locate the minimal feasible makespan with $\mathcal{O}(\log T_h)$ MILP solves, warm-started from heuristic plans (Section 3.5).
5. **Dynamic reprioritization.** Emergency insertion at the head of the planning queue in $\mathcal{O}(1)$, demonstrated on a wildfire case study, with no replanning cost for already scheduled aircraft (Section 3.6).
6. **Open-source release.** Source code (C++ and Julia) and datasets are publicly available at <https://github.com/JorgeLuizFranco/CentralizedLMDD>.

Section 4.3 states explicitly what is and is not established by the evidence reported here.

2. Related Work and Positioning

MAPF. Multi-Agent Path Finding is a core problem in Artificial Intelligence, at the intersection of automated planning, multiagent systems, and heuristic search; Stern et al. [Stern et al. 2019] standardize its variants, benchmarks, and—importantly for this work—its taxonomy of conflict types. The problem is computationally hard: it is NP-hard on directed graphs [Nebel 2020], and fine-grained analyses map the tractability frontier even on 2D grids [Geft 2023]. Three solution families are relevant here. *Reduction-based* methods encode MAPF as an integer program or network flow, an approach introduced by Yu and LaValle [Yu and LaValle 2014] and surveyed in [Surynek 2022]; the MILP used in this dissertation belongs to this family. *Optimal search* methods, notably Conflict-Based

Search [Sharon et al. 2015] and branch-and-cut-and-price [Lam et al. 2019], define the current state of the art for optimal solving. *Prioritized planning* orders the agents and plans them sequentially against previously fixed plans [Čáp et al. 2015, Ma et al. 2019]; it is fast and simple but incomplete in general.

Drone routing. The LMDD literature addresses routing cost and truck–drone coordination [Dukkanci et al. 2023] but largely treats collision avoidance as a downstream concern rather than a modeling constraint. Market-based decentralized allocation of airspace permits [Verri et al. 2020] achieves emergent stability but relies on simplified agent behavior and faces regulatory compliance difficulties. MAPF has been applied to drone–truck coordination over road networks [Choudhury et al. 2022], confirming the fit between the two problems.

Positioning of this work. We state the relationship to the above explicitly. Time-expanded networks, flow/ILP encodings, and prioritized planning are all established MAPF machinery, and the heuristic proposed here belongs squarely to the prioritized planning family. The dissertation’s contribution is therefore not a new search paradigm, but the domain formulation and the exact/scalable pairing built on it: (i) virtual per-drone source and sink vertices that turn asynchronous take-off and synchronized landing into topology rather than side constraints; (ii) zero-cost loitering self-loops that make waiting a first-class action; (iii) $\mathcal{O}(1)$ priority insertion for emergency re-tasking; and (iv) the hybrid scheme in which the heuristic’s makespan bounds and warm-starts the exact solver. What is offered to practitioners is a single graph model in which an operator can choose optimality at exponential cost or near-optimality at polynomial cost.

3. Methodology

3.1. Notation and Problem Formalization

The airspace is discretized into an $N \times M$ grid of cells; let $P = NM$ be the number of cells. A fleet $\mathcal{R} = \{1, \dots, K\}$ of K drones must each travel from a start cell to a goal cell. The planning graph is a digraph $G = (\mathcal{V}, \mathcal{A})$ whose vertices $\mathcal{V} = \mathcal{S} \cup \mathcal{B} \cup \mathcal{E}$ comprise the physical cells $\mathcal{S}$, the virtual sources $\mathcal{B} = \{b_k\}_{k \in \mathcal{R}}$ (launch pads) and the virtual sinks $\mathcal{E} = \{e_k\}_{k \in \mathcal{R}}$ (landing pads); arcs $\mathcal{A}$ encode admissible one-step transitions, including a self-loop ($i \rightarrow i$) at every cell that models *loitering* in place. Only drone k may leave b_k or enter e_k , so boundary conditions need no per-drone side constraints (Figure 2a).

Time is discretized into T steps. The *time-expanded graph* $G_T = (V_T, E_T)$ replicates each vertex at each $t \in \{0, \dots, T\}$ and connects (i, t) to $(j, t+1)$ whenever $(i, j) \in \mathcal{A}$ (Figure 2b). The binary decision variable $x_{i,j,t}^k = 1$ iff drone k traverses arc (i, j) at step t . A plan is a multi-commodity integer flow on G_T , one commodity per drone (Figure 1a), minimizing total displacement:

$$\min \sum_{k \in \mathcal{R}} \sum_{t=1}^T \sum_{\substack{(i,j) \in \mathcal{A} \\ j \notin \mathcal{B} \cup \mathcal{E}}} x_{i,j,t}^k, \quad (1)$$

subject to the constraints of Section 3.2.

Two points deserve emphasis, since the objective is stated as a count of arcs. First, every grid arc represents a move between adjacent cells of equal length, so under this

uniform-cost metric minimizing the number of traversed movement arcs *is* minimizing total flown distance; non-uniform arc costs c_{ij} would be accommodated by weighting $x_{i,j,t}^k$ in Eq. (1), which was not explored. Second, loitering self-loops and the virtual arcs incident to $\mathcal{B}$ and $\mathcal{E}$ are excluded from the sum and thus carry zero cost: waiting in place is free and only actual displacement is penalized. This is deliberate—holding an aircraft costs no distance—but it means the objective does not by itself discourage long missions, which is the role of the horizon T .

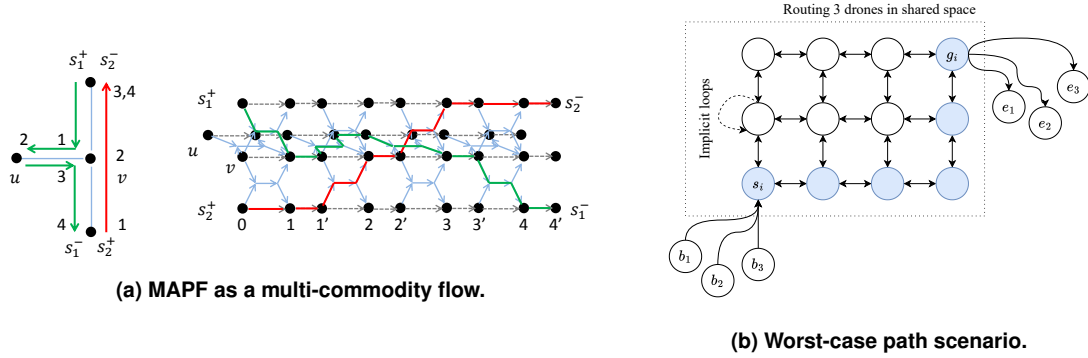


Figure 1. (a) Equivalence between a MAPF instance (left) and a multi-commodity flow on the time-expanded graph (right): u, v are cells, s_k^+ and s_k^- the source and sink of commodity k , and each coloured path is the flow of one drone across time layers $0, 1, 1', \dots$ (b) Worst case used in the complexity analysis: K drones share start cells s_i and goal cells g_i on an $N \times M$ grid, so plans serialize and the horizon grows as $K(N + M)$.

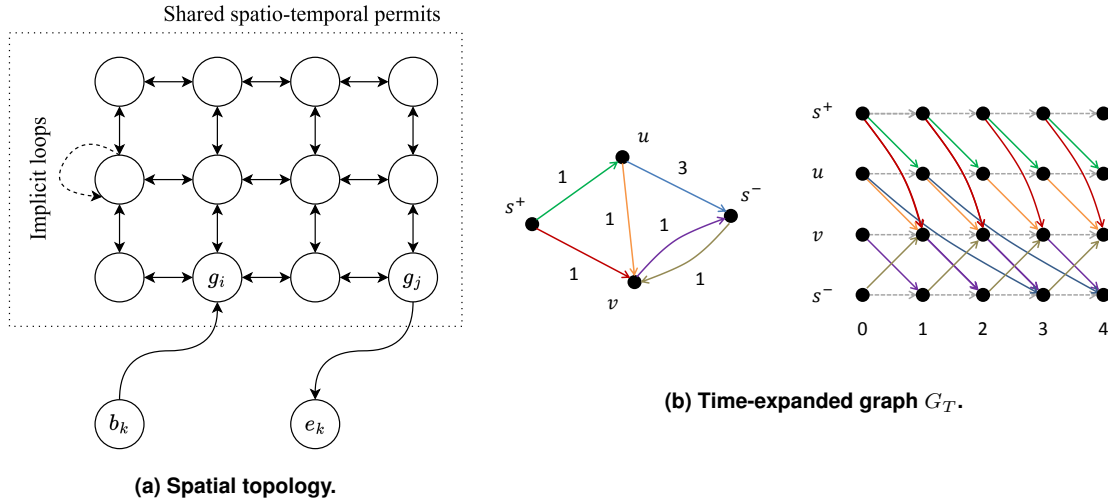


Figure 2. (a) Planning graph: the dotted box encloses the physical cells $\mathcal{S}$, while b_k and e_k are the virtual source and sink of drone k placed outside it; every cell carries an implicit self-loop for loitering. (b) Time expansion: each vertex is replicated at $t = 0, 1, \dots, T$ and arcs connect consecutive layers, so a drone's plan is a path through the layers and mutual exclusion is a per-layer constraint.

3.2. Constraints

Each drone leaves its own source exactly once, which is what turns the take-off instant into a decision rather than a fixed input:

$$\sum_{t=1}^T \sum_{j \in \mathcal{S}} x_{b_k, j, t}^k = 1, \quad \forall k \in \mathcal{R}. \quad (2)$$

Flow is conserved at every vertex and step, so each commodity traces a connected path across the time layers:

$$\sum_{i \in \mathcal{V}} x_{i, j, t-1}^k = \sum_{l \in \mathcal{V}} x_{j, l, t}^k, \quad \forall k \in \mathcal{R}, \forall j \in \mathcal{V}, t = 1, \dots, T. \quad (3)$$

The initial condition places each drone on its own launch pad, $x_{b_k, b_k, 0}^k = 1$ and $x_{i, j, 0}^k = 0$ for every $(i, j) \neq (b_k, b_k)$. Mutual exclusion admits at most one drone per cell per step:

$$\sum_{k \in \mathcal{R}} \sum_{i \in \mathcal{V}} x_{i, j, t}^k \leq 1, \quad \forall j \in \mathcal{V}, t = 1, \dots, T. \quad (4)$$

Finally, every drone must land at its own sink:

$$\sum_{t=1}^T \sum_{i \in \mathcal{S}} x_{i, e_k, t}^k \geq 1, \quad \forall k \in \mathcal{R}. \quad (5)$$

Temporal feasibility follows from conservation over the finite horizon, and self-loops at b_k and e_k absorb pre-launch holding and post-landing idling at zero cost.

Eq. (4) is what enforces safety, and its reach deserves precision: it forbids *vertex* conflicts, but not *swap* conflicts, in which two drones exchange cells along the same arc in the same step. Excluding those requires the family $x_{i, j, t}^k + x_{j, i, t}^{k'} \leq 1$ (Section 4.3).

3.3. Exact MILP Method and its Cost

The formulation above is solved exactly for a fixed horizon T . Its size grows as $\mathcal{O}(KPT)$ binary variables. Estimating the solve cost as interior-point work on the linear relaxation, $\mathcal{O}((KPT)^{3.5})$ [Huhn and Borgwardt 2002], compounded with branch-and-bound over the binaries, $\mathcal{O}(2^{KPT})$ [Nemhauser and Wolsey 1988], and substituting the worst-case horizon $T = \mathcal{O}(K(N + M)) = \mathcal{O}(K\sqrt{P})$ implied by Figure 1b, the worst-case effort of the procedure is

$$\mathcal{O}\left(K^7 P^{5.25} 2^{K^2 P \sqrt{P}}\right). \quad (6)$$

Eq. (6) is an *upper* bound on the effort of the solution procedure adopted here— $\mathcal{O}(\cdot)$ is used in its standard sense—and it characterizes solver cost rather than the intrinsic complexity of LMDD, which inherits NP-hardness from MAPF [Nebel 2020]. The operationally relevant content of Eq. (6) is the exponential factor: it is what makes the exact method a benchmark rather than a deployable planner, and what motivates the heuristic.

3.4. Prioritized-BFS Heuristic

Algorithm 1 plans drones sequentially in ascending order of the priority key π_d , the Euclidean distance between start and goal, so that short missions are committed first. For each drone a breadth-first search expands the time-expanded graph subject to the reservations $\mathcal{S}$ already committed by higher-priority drones, which are stored in a hash map so that the validity test costs $\mathcal{O}(1)$ per expansion. Because a BFS over the cells visits $\mathcal{O}(P)$ states with path length $\mathcal{O}(\sqrt{P})$ and each of the K drones is planned once, the total cost is $\mathcal{O}(P^{1.5}K)$: linear in fleet size. Figure 3 traces the procedure, including a delayed take-off caused by an occupied launch cell.

The predicate `VALID` in line 8 is what enforces safety, and we make its content explicit. It holds when w is inside the grid and not a static obstacle, and when the reservation $(w, t+1)$ is free—that is, it forbids *vertex* conflicts in the taxonomy of Stern et al. [Stern et al. 2019]. It does *not* forbid *swap* (head-on) conflicts, in which two drones exchange cells across the same arc in the same step; Section 4.3 states the consequence and the remedy.

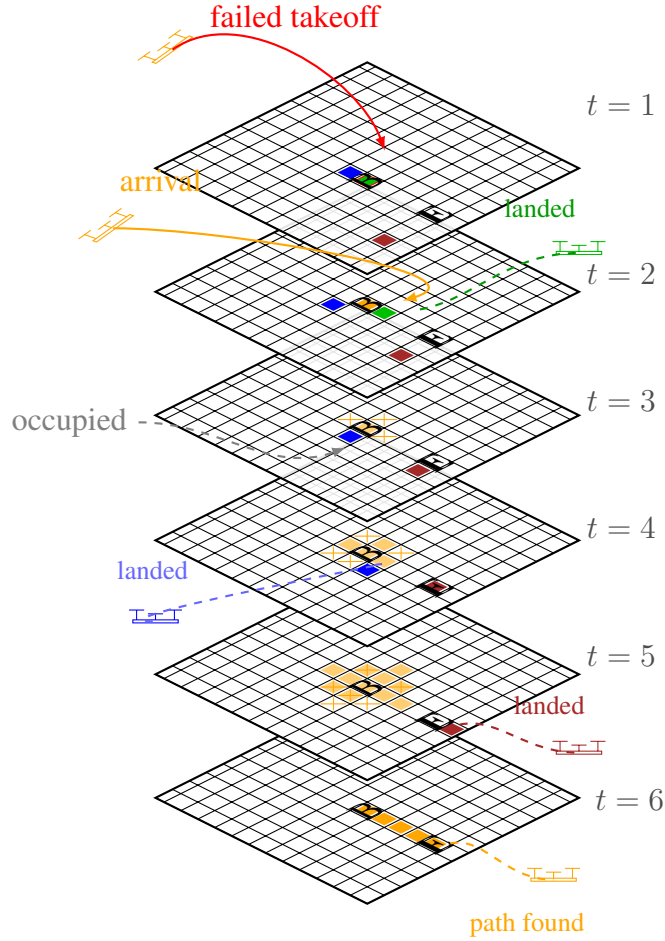


Figure 3. Algorithm visualization: routing a drone (orange) around scheduled routes (green, blue, brown). Steps $t = 1$ to $t = 6$ show conflict resolution and takeoff delay.

Algorithm 1 Priority-Driven BFS Path Planner for LMDD

```
1: Input: graph  $G$ , fleet  $\mathcal{D}$ ; reservations  $\mathcal{S} \leftarrow \emptyset$ 
2: Sort  $\mathcal{D}$  by  $\pi_d = \sqrt{(x_d^s - x_d^g)^2 + (y_d^s - y_d^g)^2}$ 
3: for each drone  $d$  in sorted order do
4:   Initialize BFS queue with  $(s_d, t_{\text{begin}})$ 
5:   while queue not empty do
6:     Dequeue  $(v, t)$ 
7:     if  $v = g_d$  then reconstruct  $P_d$ ;  $\mathcal{S} \leftarrow \mathcal{S} \cup P_d$ ; break
8:     end if
9:     for each neighbor  $w \in \mathcal{N}(v)$  (including  $w = v$ , i.e. loiter) do
10:      if  $\text{VALID}(w, t+1)$  then  $\triangleright w$  in grid, not an obstacle, and  $(w, t+1) \notin \mathcal{S}$ 
11:        Enqueue  $(w, t+1)$ 
12:      end if
13:    end for
14:  end while
15: end for
16: return  $\{P_d\}_{d \in \mathcal{D}}$ 
```

3.5. Hybrid Time-Horizon Refinement

Eq. (1) minimizes displacement for a *given* horizon T ; it does not minimize the makespan, and the two objectives are distinct. The minimal feasible horizon is recovered by a separate outer procedure. Feasibility in T is monotone—any plan valid for T remains valid for $T+1$, since each drone may loiter for one extra step at zero cost—so the predicate “an instance is feasible within T ” switches once over $[1, T_h]$. Taking the heuristic makespan T_h as an upper bound and warm-starting each solve from the heuristic plan, binary search locates the minimal feasible T^* in $\mathcal{O}(\log T_h)$ MILP solves; the displacement-optimal plan at T^* is then the output. This refinement is a design contribution of the dissertation; as recorded in its own limitations chapter, it was not evaluated empirically, and measuring how much it saves over solving at T_h directly remains open.

3.6. Dynamic Reprioritization: Wildfire Case Study

A fleet is actively routing when a fire alarm is raised at time t^* . The scheduler inserts a firefighting drone at the head of the priority queue in $\mathcal{O}(1)$ and plans its path against the existing reservations, which are left untouched: incumbent deliveries incur *zero* re-planning cost. The trade-off deserves to be named precisely. This is priority *insertion* for planning order, not a full reprioritization of the fleet: because committed trajectories are held fixed, the emergency aircraft is routed *around* them and may take a longer path than it would if lower-priority drones were replanned. Full reprioritization—preempting and replanning incumbents to clear a corridor—would yield shorter emergency paths at the cost of recomputation proportional to the number of affected drones, and is the natural next step for time-critical missions.

4. Relevant Results

4.1. Experimental Setup

Experiments ran on an Intel Core i7-7700 @ 3.60 GHz / 16 GB RAM machine. The heuristic was implemented in C++ and the MILP in Julia with JuMP and

Gurobi [Lubin et al. 2023]. For each $K \in \{1, \dots, 64\}$, 20 randomized instances were generated on a 6×6 grid; for the high-density study, populations up to $K = 99$ were tested with 30 samples each, all on the same grid.

4.2. Solution Quality and Computational Efficiency

For $K = 1$ both methods produce identical paths. As K grows the heuristic exhibits mild suboptimality, but median path lengths remain close to the exact optimum across all densities tested (Figure 4). At $K = 64$ the heuristic runs more than **300 $\times$ faster** than the MILP (milliseconds versus minutes). Figure 5 extends the heuristic to extreme saturation—99 drones in 36 cells—where routing time grows linearly in K and the normalized distance shows the congestion transition.

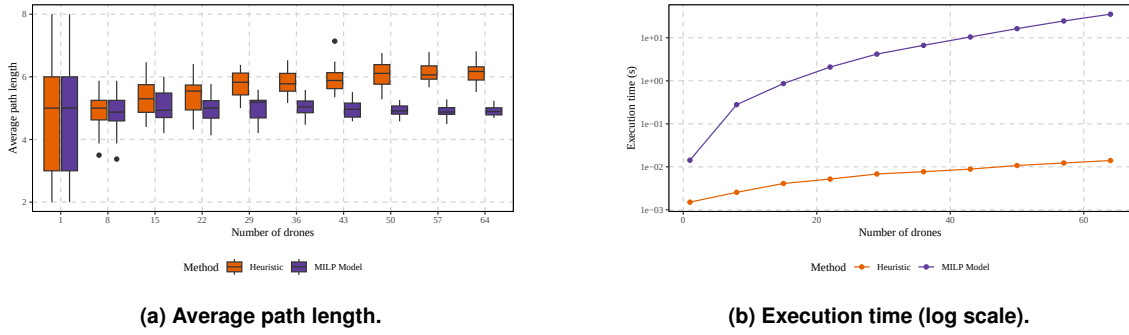


Figure 4. MILP (exact) versus prioritized-BFS (heuristic) on the 6×6 grid, as a function of the number of drones K .

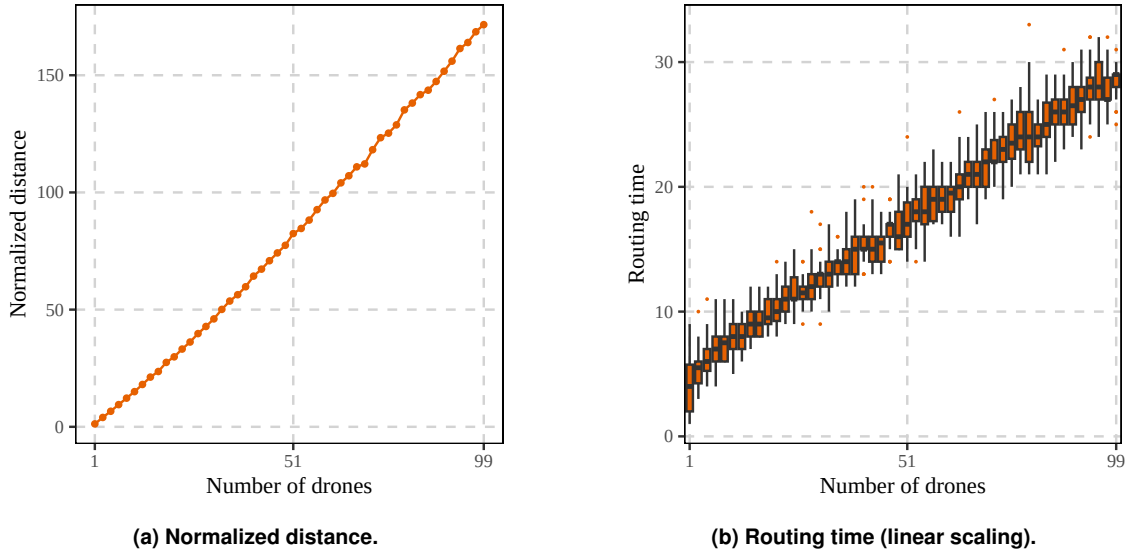


Figure 5. Heuristic behaviour up to extreme congestion (up to $K = 99$ drones in $P = 36$ cells), 30 random instances per fleet size.

4.3. Limitations and Scope of the Claims

We state the boundaries of the evidence deliberately, since several of them point directly at the next stage of this research programme.

1. **Completeness.** Prioritized planning is incomplete: a particular priority order may leave some drone without a path even when a joint solution exists [Čáp et al. 2015].

The heuristic inherits this, and no completeness guarantee is claimed. Orderings with completeness guarantees under structural assumptions [Ma et al. 2019] are a concrete remedy.

2. **Optimality.** The heuristic is near-optimal *empirically*, on the instances reported; no approximation ratio is proven, and the term should be read as an experimental observation rather than a guarantee.
3. **Benchmarking scope.** The heuristic is compared against our own exact MILP, which establishes the optimality gap but not competitiveness against the MAPF state of the art. A comparison with CBS [Sharon et al. 2015] and branch-and-cut-and-price [Lam et al. 2019] on standard benchmarks [Stern et al. 2019] is the most valuable experiment still missing.
4. **Instance diversity.** All experiments use a single 6×6 grid and vary only K , so the scalability evidence is about fleet density, not about map size, topology, or obstacle density. The extension to 3D is structural—each vertex simply gains two neighbours and the scheduling logic is unchanged—and was not evaluated empirically.
5. **Conflict types.** As noted in Section 3.4, the mutual exclusion constraint forbids vertex conflicts but not swap conflicts. Ruling them out requires adding $x_{i,j,t}^k + x_{j,i,t}^{k'} \leq 1$ for all $k \neq k'$, $(i, j) \in \mathcal{A}$ and $t = \mathcal{O}(K^2|\mathcal{A}|T)$ extra constraints—and, in the heuristic, extending VALID to reject a move whose reverse is reserved in the same step, at $\mathcal{O}(1)$ using the same hash map. Neither was active in the version evaluated here.
6. **Untested refinement.** The binary-search horizon refinement of Section 3.5 is argued from monotonicity but not measured.

4.4. Publications and Software

Table 1 summarizes the outputs, distinguishing the publication that reports the dissertation’s own results from later works that build on its methodology. All software is open source.

Table 1. Publications and software. Only the first paper reports results contained in the dissertation; the other two are subsequent developments.

Venue	Relation	Status	Qualis/JCR	Cit. ^a
Computers & Oper. Res.	Dissertation core	Published 2025	A1 / Q1	18
Physica A	Subsequent work	Published 2026	A1 / Q1	10
arXiv:2602.15736	Subsequent work	Under review ^b	—	—
Software	Language	Year	Description	
CentralizedLMDD	C++/Julia	2025	MILP + BFS heuristic for LMDD-MAPF	
airnet-holding-ml	Python	2026	Graph ML for holding prediction	
svd-centrality	Python	2026	SVD/Hodge spectral centrality	

^a Google Scholar, September 2026. ^b Submitted to *Physical Review E*; not yet accepted, and therefore reported without a quality stratum.

Core publication [Franco et al. 2025]. *An exact method and a heuristic for last-mile delivery drones routing with centralized graph-based airspace control*, *Computers & Operations Research* 178:107006, 2025 (<https://doi.org/10.1016/j.cor.2025.107006>). This paper reports the dissertation’s own contribution: the LMDD-MAPF formulation, the MILP, the $\mathcal{O}(P^{1.5}K)$ heuristic, the complexity analysis, and the comparative study of Section 4.

Subsequent work [Franco et al. 2026a]. *Graph machine learning for flight delay prediction due to holding maneuver*, *Physica A* 685:131318, 2026 (<https://doi.org/10.1016/j.physa.2026.131318>). This work follows the research agenda set out in the dissertation’s future-work chapter, which called for graph neural networks and for robustness to stochastic delays. It carries the edge-centric modeling decision into a different airspace problem: where the dissertation treats in-air waiting as loitering arcs in a temporal graph, this paper models the Brazilian air-transportation network as a directed multigraph—airports as nodes, flights as edges—and predicts holding as an *edge feature*, over 42,336 operations from an ICEA/ITA dataset. CatBoost with graph-theoretic features outperforms Graph Attention Networks on the 1.7%-positive class (accuracy 0.90, recall 0.58). The GAT itself is generalized to carry edge features and to separate *multi-edges*, so that each individual flight between two airports earns its own attention coefficient (Figure 6): two flights sharing endpoints but differing in weather or runway conditions contribute differently—the same edge-level granularity the time-expanded model gives individual drone moves.

Subsequent work [Franco et al. 2026b]. *SVD incidence centrality: a unified spectral framework for node and edge analysis in directed networks and hypergraphs*, arXiv:2602.15736, under review at *Physical Review E*. It generalizes the same edge-centric view into a spectral theory, and its motivation cites the holding study as evidence that edge-centric measures matter in prediction. Vertex and edge centralities are read from the diagonals of the pseudoinverses of the Hodge Laplacians $L_0 = BB^\top$ and $L_1 = B^\top B$ of the oriented incidence matrix B , which share the spectrum of B ’s singular values. On undirected graphs this reproduces current-flow closeness; on directed graphs the equivalence breaks precisely where symmetrization would destroy information, and direction is recovered by aggregating edge scores into hub and authority rankings. The Hodge decomposition splits edge space into gradient and cyclic flows—the very space in which the dissertation’s MILP routes drones as integer edge flows (Figure 7).

The thread. Across the three, the decisive information lives on the *edges*: routes are edge flows, holding events are edge features, and centrality is computed for edges directly. Only the first is a result of the dissertation; the other two show where its methodology led.

5. Broader Impact

The framework addresses a bottleneck for operators such as iFood and Amazon Prime Air, and aligns with the FAA’s UTM concept, which presumes organized airspace management [Prevot et al. 2016]. The same planner serves hub location-routing, green vehicle routing and relief logistics: only the topology changes.

6. Conclusions

The dissertation presents a centralized framework for LMDD routing that avoids vertex conflicts by construction, pairing an exact MILP benchmark with a prioritized-BFS planner more than two orders of magnitude faster at $K = 64$: optimality at exponential cost, or empirically near-optimal plans at polynomial cost, within a single graph model that also supports asynchronous take-off, free loitering and $\mathcal{O}(1)$ emergency insertion.

References

- [Choudhury et al. 2022] Choudhury, S., Solovey, K., Kochenderfer, M., and Pavone, M. (2022). Coordinated multi-agent pathfinding for drones and trucks over road networks. In *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems, AAMAS '22*, page 272–280, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.
- [Dukkanci et al. 2023] Dukkanci, O., Campbell, J. F., and Kara, B. Y. (2023). Facility location decisions for drone delivery: A literature review. *European Journal of Operational Research*.
- [Franco et al. 2025] Franco, J. L., Curtis, V. V., Senne, E. L. F., and Verri, F. A. N. (2025). An exact method and a heuristic for last-mile delivery drones routing with centralized graph-based airspace control. *Computers & Operations Research*, 178:107006.
- [Franco et al. 2026a] Franco, J. L., Machado Neto, M. V., Verri, F. A. N., and Amancio, D. R. (2026a). Graph machine learning for flight delay prediction due to holding maneuver. *Physica A: Statistical Mechanics and Its Applications*, 685:131318.
- [Franco et al. 2026b] Franco, J. L., Peron, T., Dal Col, A., Petronetto, F., Verri, F. A. N., Tokuda, E. K., and Nonato, L. G. (2026b). SVD incidence centrality: A unified spectral framework for node and edge analysis in directed networks and hypergraphs. arXiv:2602.15736. Under review at Physical Review E.
- [Geft 2023] Geft, T. (2023). Fine-grained complexity analysis of multi-agent path finding on 2d grids. *Proceedings of the Sixteenth International Symposium on Combinatorial Search*, 16(1).
- [Huhn and Borgwardt 2002] Huhn, P. and Borgwardt, K. H. (2002). Interior-point methods: Worst case and average case analysis of a phase-i algorithm and a termination procedure. *Journal of Complexity*, 18(3):833–910.
- [Lam et al. 2019] Lam, E., Le Bodic, P., Harabor, D. D., and Stuckey, P. J. (2019). Branch-and-cut-and-price for multi-agent pathfinding. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 1289–1296. International Joint Conferences on Artificial Intelligence Organization.
- [Lubin et al. 2023] Lubin, M., Dowson, O., Dias Garcia, J., Huchette, J., Legat, B., and Vielma, J. P. (2023). JuMP 1.0: Recent improvements to a modeling language for mathematical optimization. *Mathematical Programming Computation*.
- [Ma et al. 2019] Ma, H., Harabor, D., Stuckey, P. J., Li, J., and Koenig, S. (2019). Searching with consistent prioritization for multi-agent path finding. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 7643–7650.
- [Nebel 2020] Nebel, B. (2020). On the computational complexity of multi-agent pathfinding on directed graphs. *Proceedings of the International Conference on Automated Planning and Scheduling*, 30(1):212–216.
- [Nemhauser and Wolsey 1988] Nemhauser, G. L. and Wolsey, L. A. (1988). *Integer and Combinatorial Optimization*. Wiley-Interscience.

- [Prevot et al. 2016] Prevot, T., Rios, J., Kopardekar, P., III, J. E. R., Johnson, M., and Jung, J. (2016). *UAS Traffic Management (UTM) Concept of Operations to Safely Enable Low Altitude Flight Operations*.
- [Sharon et al. 2015] Sharon, G., Stern, R., Felner, A., and Sturtevant, N. R. (2015). Conflict-based search for optimal multi-agent pathfinding. *Artificial Intelligence*, 219:40–66.
- [Stern et al. 2019] Stern, R., Sturtevant, N., Felner, A., Koenig, S., Ma, H., Walker, T., Li, J., Atzmon, D., Cohen, L., Kumar, T., et al. (2019). Multi-agent pathfinding: Definitions, variants, and benchmarks. In *Proceedings of the International Symposium on Combinatorial Search*, volume 10, pages 151–158.
- [Surynek 2022] Surynek, P. (2022). Problem compilation for multi-agent path finding: a survey. In Raedt, L. D., editor, *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 5615–5622. International Joint Conferences on Artificial Intelligence Organization. Survey Track.
- [Verri et al. 2020] Verri, F. A. N., Marcondes, C. A. C., Loubach, D. S., Sbruzzi, E. F., Marques, J. C., Júnior, L. A. P., Maximo, M. R. O. D. A., and Curtis, V. V. (2020). An analysis on tradable permit models for last-mile delivery drones. *IEEE Access*, 8:186279–186290.
- [Yu and LaValle 2014] Yu, J. and LaValle, S. M. (2014). Multi-agent path planning and network flow. In *Algorithmic Foundations of Robotics X (WAFR) 2012*, Springer Tracts in Advanced Robotics, Vol. 86, pages 157–173. Springer.
- [Čáp et al. 2015] Čáp, M., Novák, P., Kleiner, A., and Selecký, M. (2015). Prioritized planning algorithms for trajectory coordination of multiple mobile robots. *IEEE Transactions on Automation Science and Engineering*, 12(3):835–849.

Appendix: Subproducts of the Dissertation

Software. All code and data are open source: <https://github.com/JorgeLuizFranco/CentralizedLMDD> (C++ heuristic, Julia/JuMP MILP and datasets for the LMDD-MAPF framework); <https://github.com/graph-learning-ita/airnet-holding-ml> (graph ML pipeline and web tool for holding prediction); <https://github.com/JorgeLuizFranco/svd-centrality> (SVD incidence centrality).

Publication reporting the dissertation’s results — Qualis A1, JCR Q1, 18 citations (Google Scholar, September 2026). Franco, J. L., Curtis, V. V., Senne, E. L. F., and Verri, F. A. N. (2025). An exact method and a heuristic for last-mile delivery drones routing with centralized graph-based airspace control. *Computers & Operations Research*, 178:107006. <https://doi.org/10.1016/j.cor.2025.107006>

Subsequent work — Qualis A1, JCR Q1, 10 citations. Franco, J. L., Machado Neto, M. V., Verri, F. A. N., and Amancio, D. R. (2026). Graph machine learning for flight delay prediction due to holding maneuver. *Physica A*, 685:131318. <https://doi.org/10.1016/j.physa.2026.131318>

Subsequent work, under review — submitted to *Physical Review E*; no decision has been issued, so no quality stratum is claimed. Franco, J. L., Peron, T., Dal Col, A., Petronetto, F., Verri, F. A. N., Tokuda, E. K., and Nonato, L. G. (2026). SVD incidence

centrality: a unified spectral framework for node and edge analysis in directed networks and hypergraphs. arXiv:2602.15736. <https://arxiv.org/abs/2602.15736>

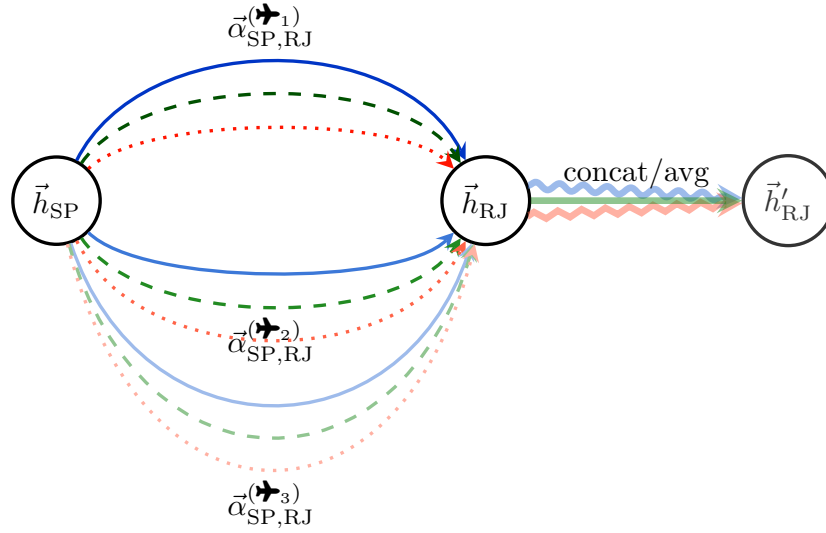


Figure 6. GAT layer with multi-head attention on the airport multigraph, with separate coefficients for three flights between São Paulo (SP) and Rio de Janeiro (RJ) [Franco et al. 2026a].

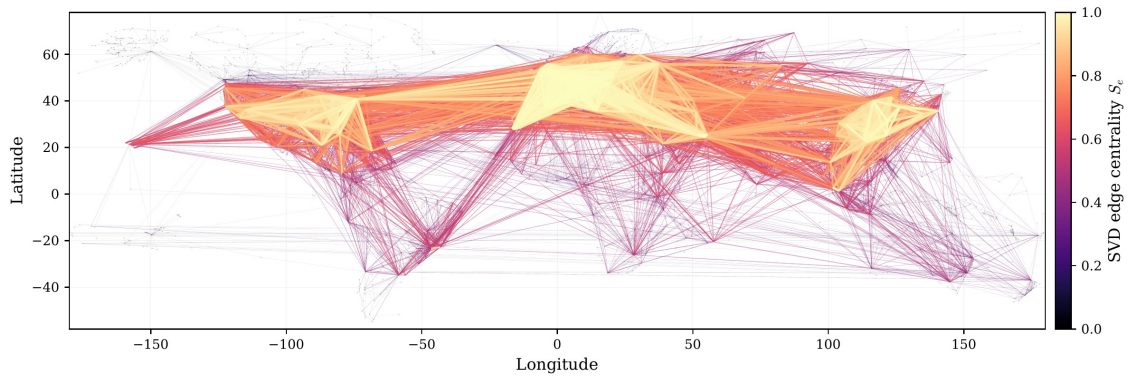


Figure 7. SVD edge incidence centrality of the OpenFlights network (3,188 airports, 36,860 directed routes) at real geographic coordinates; the brightest segments are the spectrally dominant corridors of global aviation [Franco et al. 2026b].