An ASP-based Approach to UAM Strategic Deconfliction: preliminary results

Gioacchino Sterlicchio[1,2,*], Angelo Oddi[3], Riccardo Rasconi[3] and Francesca A. Lisi[2,4]

[1]Dept. of Mechanics, Mathematics and Management, Polytechnic University of Bari, Via G. Amendola 126/b - 70126 Bari, Italy
[2]Dept. of Computer Science, University of Bari "Aldo Moro", Via E. Orabona 4, Bari, 70125, Italy
[3]CNR ISTC - Institute of Cognitive Sciences and Technologies, National Research Council, Italy
[4]Centro Interdipartimentale di Logica e Applicazioni (CILA), University of Bari "Aldo Moro", Via E. Orabona 4, Bari, 70125, Italy

Abstract

Urban Air Mobility (UAM) promises to revolutionize transportation in metropolitan areas by introducing "air taxis" for passenger and cargo transport. However, the envisioned dense operations of UAM vehicles in low-altitude airspace pose unprecedented challenges for air traffic management (ATM). The Strategic Deconfliction (SD) problem in UAM is about designing the pre-flight "air traffic plan" for potentially hundreds or thousands of simultaneous urban flights, allocating routes, times, and resources in a way that guarantees separation and operational feasibility before any aircraft even leaves the ground. This short paper presents an approach based on Answer Set Programming (ASP) to solve the SD problem in UAM and reports preliminary results on a use case. In particular, the modelling choices will be described with regard to the air network topology, the fleet of drones to be scheduled and the SD problem.

Keywords

Urban Air Mobility, Strategic Deconfliction, Answer Set Programming

1. Introduction

Urban Air Mobility (UAM) envisions a future where electric Vertical Takeoff and Landing (eVTOL) aircraft, often referred to as "air taxis," seamlessly integrate into urban transportation networks, alleviating congestion and reducing emissions [1]. Projections suggest thousands of daily UAM operations in major cities by 2030-2040. Unlike traditional commercial aviation: 1) UAM operates in low-altitude airspace densely populated with obstacles (buildings, towers) and dynamic hazards (drones, birds); 2) manned and unmanned aircraft (e.g., cargo drones, medical eVTOLs) share the same airspace, each with distinct performance profiles and mission priorities; 3) takeoff/landing sites (vertiports) are scattered across cities, acting as "air traffic hubs" with constrained capacity, akin to urban airports, and 4) UAM services need on-demand scheduling, contrasting with the pre-planned nature of commercial flights. These factors amplify the need for Strategic Deconfliction (SD) – proactively resolving conflicts before eVTOLs depart – to ensure safe, efficient, and scalable UAM ecosystems.

In this short paper, we want to introduce to the reader the preliminary results on the development of an approach based on Answer Set Programming (ASP) to solve the problem of *strategic deconfliction* in UAM. In particular, the modelling choices will be described with regard to the air network topology, the fleet of drones to be scheduled and the SD problem.

The paper is organized as follows. In Section 2 we briefly recall the basics of ASP. Then, in Section 3, we define the strategic deconfliction problem showing a practical example. In Section 4 we explain the rationale behind the modelling choices of the various parts of the problem with a small example to

CILC 2025: 40th Italian Conference on Computational Logic, June 25–27, 2025, Alghero, Italy
*Corresponding author.
g.sterlicchio@phd.poliba.it (G. Sterlicchio); angelo.oddi@istc.cnr.it (A. Oddi); riccardo.rasconi@istc.cnr.it (R. Rasconi); FrancescaAlessandra.Lisi@uniba.it (F. A. Lisi)
https://www.istc.cnr.it/it/people/angelo-oddi (A. Oddi); https://www.istc.cnr.it/it/people/riccardo-rasconi (R. Rasconi); https://www.uniba.it/it/docenti/lisi-francesca-alessandra (F. A. Lisi)
0000-0002-2936-0777 (G. Sterlicchio); 0000-0003-4370-7156 (A. Oddi); 0000-0003-2420-4713 (R. Rasconi); 0000-0001-5414-5844 (F. A. Lisi)

demonstrate how the proposed solution works. Section 5 overviews the recent literature regarding the SD problem in UAM. Finally, Section 6 concludes the paper with final remarks and future works.

2. Answer Set Programming

Answer Set Programming (ASP) [2] is a declarative programming paradigm that allows for the representation and solving of complex combinatorial problems. It is based on the logical formalism of logic programs, specifically, disjunctive logic programs with answer set semantics. ASP provides a powerful tool for solving problems in various domains such as planning, scheduling, reasoning about actions, and knowledge representation. In an ASP program, rules are defined using predicates and logical connectors such as conjunctions, disjunctions, and negations. The program consists of a set of rules which define relationships between different elements in the problem domain. These rules are then used to generate answer sets, i.e., sets of consistent interpretations that satisfy all constraints specified in the program. One advantage of ASP over other declarative programming paradigms is its expressiveness and flexibility in concisely representing complex problems through logical constraints. Additionally, ASP programs can be easily modified or extended without changing their overall structure due to their modular nature. ASP solvers use sophisticated algorithms based on efficient search techniques to compute answer sets; two of the most important instances are Clingo [3] and DLV [4]. An example of general rule is: $a_1 \vee \ldots \vee a_n \leftarrow b_1, \ldots, b_k, not\ b_{k+1}, \ldots, not\ b_m$. The rule says that if $b_1, \ldots, b_k$ are true and there is no reason to believe that $b_{k+1}, \ldots, b_m$ are true then at least one of the $a_1, \ldots, a_n$ is believed to be true. The left hand side and the right hand side of the $\leftarrow$ are called *head* and *body* respectively. Rules without body are called *facts*. The head is unconditionally true and the arrow is usually omitted. Conversely, rules without head are called *constraints* and are used to discard stable models, thus reducing the number of answers returned by the ASP solver.

3. The Strategic Deconfliction Problem

The *Strategic Deconfliction* (SD) problem consists in producing a set of scheduled flight paths such that no two aircraft ever get closer than a specified safety distance (or *headway*) either in time or space. SD is the first of the three layers into which Air Traffic Management (ATM) is typically divided [5]. The second layer is called *tactical deconfliction*. The goal is to maintain separation provision of ongoing traffic using real-time information such as current location, heading, and speed. The last layer is about *collision avoidance* (or detect-and-avoid) and uses onboard technologies to avoid imminent collisions. The SD problem follows the principle of the *lane-based* airspace structure highlighted in [6] relatively to the creation and the layout of the track network. Indeed, lanes allow for efficient and effective real-time deconfliction to mitigate contingencies [7].

The lane-based approach defines a set of one-way lanes where each lane is defined by an entry point, an exit point, and a one-dimensional curve between the two. The lane-based structure is modelled as a directed graph $G = (V, E)$, where V is the set of vertices or lane entry-exit points and E the set of lanes. Vertices could be vertiports, vertistops or waypoints. Two-way traffic between vertices can be achieved by having air lanes next to each other at the same altitude or at different altitude. Figure 1 shows two examples of two-lane layout. They are represented as a graph $G = (V, E)$, where $V = \{1, 2, 3\}$ and $E = \{(1, 2), (2, 3)\}$ for the one-way structure (a) and $E = \{(1, 2), (2, 1), (2, 3), (3, 2)\}$ for the two-way alternative (b). In particular, vertices 1 and 3 are ground nodes, and 2 is a waypoint of the airspace structure, located at some altitude.

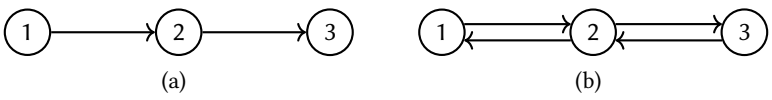
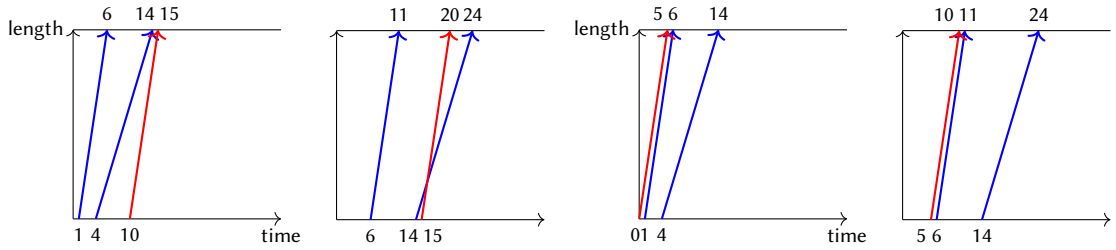


Figure 1: A simple one-way (a) and two-way (b) two-lane layout.

A flight must schedule its entry-exit times through a sequence of lanes, where the exit time from the previous lane equals the entry time of the following lane. In order to determine whether flights have a conflict, we use the Space-Time Lane Diagram (STLD) proposed in [6] to graphically represent the situation, as shown in Figure 2. The horizontal axis represents time, while the vertical axis represents the distance travelled along the lane (suppose 10 for each lane). An STDL is created for each lane. Before proceeding with the example, we assume for the remainder of the article that, although two different flights may have different speeds, each speed remains constant throughout the flight. The two blue lines represent two scheduled flights named f_1 and f_2 with start times of 1 and 4 in lane 1–2 with speeds 2 and 1, respectively. The STDL show their progress through the two lanes; it can be seen that there is always a time headway of at least 1 unit between the flights. Suppose a new flight f_3 (red line) must be scheduled, with speed 2, and the requested launch interval is $[0, 15]$. This means that the earliest launch time is 0 while the latest one is 15. The goal is to establish departure times that do not conflict with those already present and to find the trajectory for all lanes that does not conflict with all flights traveling in their respective lanes. For example, consider f_3 starts at time 10, then it exits Lane 1-2 and enters Lane 2-3 at time 15; as the figure shows, this flight would cross the path of f_2 and therefore it is disallowed. On the other hand, if the proposed flight starts at time 0, then it is one time unit from f_1, and since they go the same speed, they never get any closer. Moreover, for Lane 2–3 the proposed flight is one unit to the left of f_1, so it is allowed.



(a) Lane 1–2 (b) Lane 2–3 (c) Lane 1–2 (d) Lane 2–3

Figure 2: STDL representation of the flights. Conflict case (a–b), no conflict case (c–d).

4. Modeling the SD Problem with ASP

In this section, we present our ASP encoding of the SD problem. For illustrative purposes, we consider a possible use case that could be encountered in future UAM corridors, with grid layout and one-way lanes (see Figure 3). It simulates a regular city with four vertiports (nodes 1, 2, 7 and 8). The remaining nodes (3 – 6) refer to waypoints up in the air.

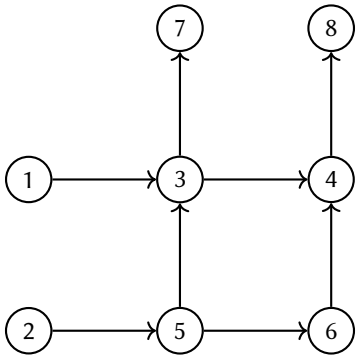


Figure 3: UAM use case with grid layout and one-way lanes.

Listing 1: ASP encoding of UAM use case reported in Figure 3.

```
node(1..8). startn(1;2). endn(7;8).
edge(1,3). edge(3,4). edge(2,5). edge(5,6). edge(3,7). edge(4,8). edge(5,3). edge(6,4).
length((1,3),7). length((3,4),7). length((2,5),7). length((5,6),7). length((5,3),9).
length((6,4),9). length((3,7),9). length((4,8),9).
```

This grid-shaped use case is represented in ASP by the facts reported in Listing 1. The predicate `node/1` is the node id, `startn/1` and `endn/1` describe the initial node and the end node of the network. `edge/2` is the lane that connects two nodes and `length/2` is the lane length. For the two-way layout the new `edge/2` and `length/2` atoms must be introduced together with `startn(3)` and `endn(1)`.

Suppose we need to schedule three flights for the grid-shaped layout. Listing 2 shows the modelling choices for the drone fleet. We first define the `flight/1` predicate, that is the drone id. Drone speed is modelled by the `speed(f,v)` predicate, where `f` is the flight and `s` is its speed. At this time, we assume that the speed is constant. The drone route is defined through the `lane(f,x,y)` predicate. `lane/3` allows to model the route through the sequence of nodes that the drone crosses. The direction of the drone route is defined by `start(f,n)` and `end(f,n)` that model the beginning and end node of the established route for the drone `f`. The launch time window is defined through the predicate `requested(f,e,l)`, stating that the flight `f` should depart between time `e` (earliest start time) and `l` (latest start time).

Listing 2: ASP encoding of the drone fleet to schedule.

```
flight(f1). speed(f1,2).
start(f1,1). end(f1,8).
lane(f1,1,3). lane(f1,3,4). lane(f1,4,8).
requested(f1,2,6).

flight(f2). speed(f2,3).
start(f2,2). end(f2,8).
lane(f2,2,5). lane(f2,5,3). lane(f2,3,4). lane(f2,4,8).
requested(f2,3,5).

flight(f3). speed(f3,2).
start(f3,1). end(f3,7).
lane(f3,1,3). lane(f3,3,7).
requested(f2,4,5).
```

In the following, we describe the ASP encoding of the SD problem as shown in Listing 3. Each answer set is a feasible scheduling plan. Line 1 defines the headway (safety distance) with the predicate `headway(h)` where `h` is a constant taken as input. At the moment, we assume that this is defined in terms of time. Rule at line 3 assigns a start time point `stpoint(F,X,T)` to each flight `flight(F)` from its requested launch time interval `requested(F,E,L)` where the start node `X` is defined by `start(F,X)`. The constraint at Line 5 eliminates all the answer sets where: 1) at least two flights share the same starting time point for the same start node, and 2) there is no safe distance between flights at the same start point. Rules at Lines 7–8 compute the estimated time of arrival `eta/3`. The predicate `eta(F,Y,T)` (Line 7) is true if there is a flight `F` that starts its trip at time `Ti` at node `X` and `F` travels through the lane `(X,Y)` with speed `S`. The arrival time is calculated taking into account the lane length `length((X,Y),D)` applying the formula $T = Ti + (D/S)$. Line 8 is applied to all other nodes knowing the arrival time at the previous one. Line 10 guarantees a safe distance between flights at the same node. Lines 12–15 resolve the conflict between two flights that share the same lane.

Listing 3: ASP encoding of the SD problem.

```
1  headway(h).
2
3  1{stpoint(F,X,T) : T=E..L}1 :- flight(F), requested(F,E,L), start(F,X).
4
5  :- headway(H), stpoint(F1,X,T1), stpoint(F2,X,T2), F1!=F2, |T1-T2|<H.
6
7  eta(F,Y,T) :- stpoint(F,X,Ti), speed(F,S), lane(F,X,Y), length((X,Y),D), T=(Ti+(D/S)).
8  eta(F,Y,T) :- eta(F,X,Ti), speed(F,S), lane(F,X,Y), length((X,Y),D), T=(Ti+(D/S)).
9
10 :- headway(H), eta(F1,X,T1), eta(F2,X,T2), F1!=F2, |T1-T2|<H.
```

```
11
12  :- eta(F1,X,Tx1), eta(F1,Y,Ty1), eta(F2,X,Tx2), eta(F2,Y,Ty2),
13     lane(F1,X,Y), lane(F2,X,Y), F1!=F2, Tx1<Tx2, Ty2<Ty1.
14  :- stpoint(F1,X,Tx1), stpoint(F2,X,Tx2), eta(F1,Y,Ty1), eta(F2,Y,Ty2), lane(F1,X,Y),
15     lane(F2,X,Y), F1!=F2, Tx1<Tx2, Ty2<Ty1.
```

The solution of the SD problem instance with the drone fleet of Listing 2 as input, with `headway(2)`, is given in the following Listing 4. Information about start time `stpoint/2` and estimated time of arrival `eta/3` are shown for each answer set and for each flight. The graphical representation of the solution is given in Figure 4.

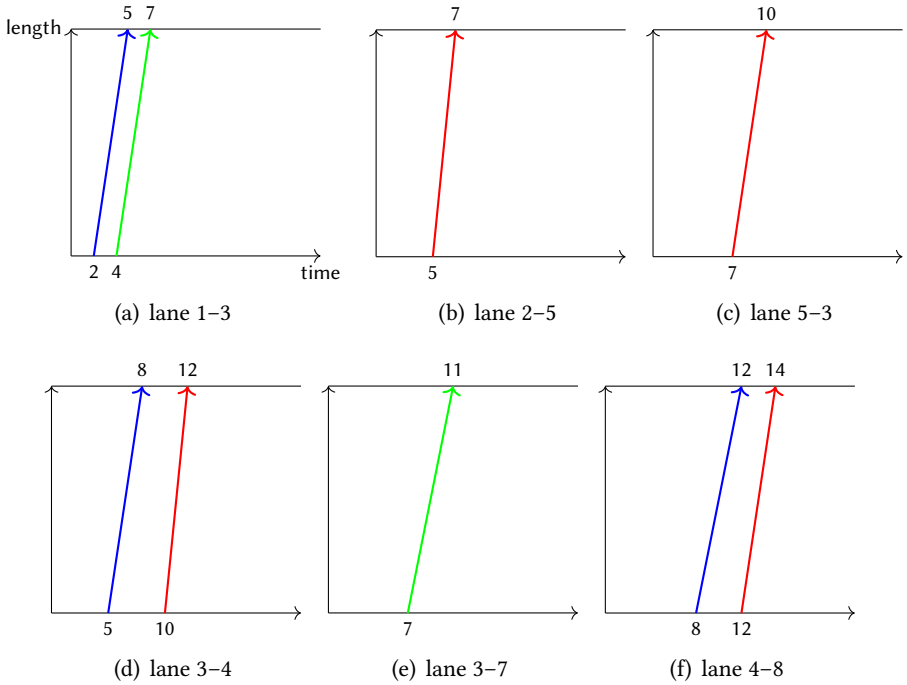
Listing 4: Possible solutions of the SD problem instance with the drone fleet of Listing 2 as input.

```
Answer: 1
stpoint(f1,1,2) eta(f1,3,5) eta(f1,4,8) eta(f1,8,12)
stpoint(f2,2,5) eta(f2,5,7) eta(f2,3,10) eta(f2,4,12) eta(f2,8,15)
stpoint(f3,1,4) eta(f3,3,7) eta(f3,7,11)

Answer: 2
stpoint(f1,1,2) eta(f1,3,5) eta(f1,4,8) eta(f1,8,12)
stpoint(f2,2,4) eta(f2,5,6) eta(f2,3,9) eta(f2,4,11) eta(f2,8,14)
stpoint(f3,1,4) eta(f3,3,7) eta(f3,7,11)

Answer: 3
stpoint(f1,1,2) eta(f1,3,5) eta(f1,4,8) eta(f1,8,12)
eta(f2,5,7) stpoint(f2,2,5) eta(f2,3,10) eta(f2,4,12) eta(f2,8,15)
stpoint(f3,1,5) eta(f3,3,8) eta(f3,7,12)

Answer: 4
stpoint(f1,1,3) eta(f1,3,6) eta(f1,4,9) eta(f1,8,13)
stpoint(f2,2,5) eta(f2,5,7) eta(f2,3,10) eta(f2,4,12) eta(f2,8,15)
stpoint(f3,1,5) eta(f3,3,8) eta(f3,7,12)
```



(a) lane 1–3 (b) lane 2–5 (c) lane 5–3

(d) lane 3–4 (e) lane 3–7 (f) lane 4–8

Figure 4: Graphical representation of answer 1. `f1` is highlighted in blue, `f2` in red and `f3` in green.

5. Related Works

This section overviews the literature relevant for the task addressed in the UAM context. To the best of our knowledge, an approach based on ASP or in general with declarative approach does not exist.

In fact, as will be described below, most of the works focus on numerical approaches. Since our work concerns a problem that is close to problems in scheduling, we include in our overview also works that us ASP for this purpose in several domains. Moreover, UAM is a new domain of application for ASP and appears to be challenging and worthy to be considered.

Ricca et al. [8] present a system based on ASP for the automatic generation of the teams of employees in the seaport of Gioia Tauro. Dodaro and Maratea [9] face the Nurse Scheduling problem that consists of assigning nurses to shifts according to given practical constraints. ASP is used for this purpose by also developing an encoding to find the optimal schedule. Alviano et al. [10] use ASP for the Nurse Rescheduling problem computing new optimal schedules that minimizing changes of nurse shift. Dodaro et al. [11] present a solution based on ASP to the problem of scheduling chemotherapy in oncology clinics. In [12, 13], authors present an ASP-based solution to real-world train scheduling problems, involving routing, scheduling, and optimization. In particular, they use an hybrid approach that extends ASP with difference constraints using the ASP system Clingo[DL].

Interest in UAM strategic deconfliction management is growing thanks to large agencies that are defining the concepts of operation [14, 15]. Sacharny and Henderson [16] present an airspace structure inspired by roadway roundabouts, and a computationally tractable trajectory scheduling algorithm for UAS Service Suppliers. Sacharny et al. [17] propose a lane-based airway navigation framework in which each lane is one-way, and intersections are handled using polygonal roundabouts of the lane. They also show a method to determine all allowable launch times i.e., strategically deconflicted given a requested launch time interval and a set of scheduled flights. Tang et al. [18] focus on designing a flight planning system that can plan conflict free trajectories for UAM flights operating in high density traffic. In [19], the strategic conflict issue in low-altitude UAM operations with multi-agent reinforcement learning (MARL) is addressed. Thompson et al. [20] propose a framework for generating routes and accompanying contracts of operational volumes (OVs). To generate the routes for OV generation they propose a rapidly-exploring random tree (RRT) based algorithm focusing on the cruise phase of flight. In [21] is described a framework that combines demand capacity balancing (DCB) for strategic conflict management and reinforcement learning for tactical separation. Pradeep et al. [22] face the problem of SD for an urban package delivery formulating a mixed-integer non-linear programming (MINLP) model.

6. Conclusions and Future Work

In this article, we have shown a possible solution to the SD problem through ASP. The article analyzed the problem by describing a practical model using the lane-based approach. From the ASP point of view, a possible lane-based model of the air network of the drone fleet to be scheduled and a solution to the problem have been discussed. We have showed the application to a real use case with an example of drones to be scheduled for the grid layout. The results are promising and encourage us to pursue the work. In particular, we would like to stress that ASP finds an excellent application in UAM due to its ability to efficiently handle complex optimization and reasoning tasks. Indeed, UAM involves managing a dynamic and intricate network of aerial vehicles, which requires robust scheduling, traffic management, and security protocols. ASP declarative nature allows for concise representation of these problems, enabling effective solutions for optimal route planning, congestion mitigation, and attack pattern detection.

The presented approach is still work in progress and much remains to be done. The limitations of the proposed version relate primarily to speed management. In fact, we have assumed that the speed is constant throughout the flight, but there could be situations where speed adjustments are necessary to maintain safe distances from other aircraft and/or obstacles. In addition, speed may be modified to minimize the effects of severe turbulence or other weather disturbances. Consequently, it is also important to model scenarios involving variable speeds. Another point to work on concerns the choice of plan. At the moment, possible plans are shown, but among them it is necessary to find the best one taking into account the metrics used in the literature. One of these concerns the minimisation of the

delay with respect to the required launch interval and the actual interval given. Moreover, although a potential use case has been presented in this article to illustrate the output, the example involves only a limited number of flights. A further point of improvement will involve evaluating the efficiency of the proposed approach, in terms of both execution time and memory usage. It is expected that memory consumption will grow as the number of flights increases, particularly due to the grounding phase. Last, it will be necessary to understand the real benefits of an ASP-based approach compared to work presented in the literature such as numerical approaches based on mathematical programming.

Acknowledgments

This work was partially supported by the project FAIR- Future AI Research (PE00000013), under the NRRP MUR program funded by the NextGenerationEU.

Declaration on Generative AI

The authors have not employed any Generative AI tools.

References

[1] H. Pak, L. Asmer, P. Kokus, B. I. Schuchardt, A. End, F. Meller, K. Schweiger, C. Torens, C. Barzantny, D. Becker, J. M. Ernst, F. Jäger, T. Laudien, N. Naeem, A. Papenfuß, J. Pertz, P. S. Prakasha, P. Ratei, F. Reimer, P. Sieb, C. Zhu, R. Abdellaoui, R.-G. Becker, O. Bertram, A. Devta, T. Gerz, R. Jaksche, A. König, H. Lenz, I. C. Metz, F. Naser, L. Schalk, S. Schier-Morgenthal, M. Stolz, M. Swaid, A. Volkert, K. Wendt, Can urban air mobility become reality? opportunities and challenges of uam as innovative mode of transport and dlr contribution to ongoing research, CEAS Aeronautical Journal (2024). URL: https://doi.org/10.1007/s13272-024-00733-x. doi:10.1007/s13272-024-00733-x.

[2] V. Lifschitz, Answer sets and the language of answer set programming, AI Magazine 37 (2016) 7–12.

[3] M. Gebser, R. Kaminski, B. Kaufmann, T. Schaub, Clingo= ASP+ control: Preliminary report, arXiv preprint arXiv:1405.3694 (2014).

[4] N. Leone, C. Allocca, M. Alviano, F. Calimeri, C. Civili, R. Costabile, A. Fiorentino, D. Fuscà, S. Germano, G. Laboccetta, B. Cuteri, M. Manna, S. Perri, K. Reale, F. Ricca, P. Veltri, J. Zangari, Enhancing DLV for large-scale reasoning, in: M. Balduccini, Y. Lierler, S. Woltran (Eds.), Logic Programming and Nonmonotonic Reasoning - 15th International Conference, LPNMR 2019, Philadelphia, PA, USA, June 3-7, 2019, Proceedings, volume 11481 of *Lecture Notes in Computer Science*, Springer, 2019, pp. 312–325. URL: https://doi.org/10.1007/978-3-030-20528-7_23. doi:10.1007/978-3-030-20528-7_23.

[5] Y. Liu, Z. Zhou, W. Naqvi, J. Chen, Strategic deconfliction of unmanned aircraft based on hexagonal tessellation and integer programming, Journal of Guidance, Control, and Dynamics 46 (2023) 2362–2372.

[6] D. Sacharny, T. C. Henderson, V. Marston, Lane-based large-scale UAS traffic management, IEEE Trans. Intell. Transp. Syst. 23 (2022) 18835–18844. URL: https://doi.org/10.1109/TITS.2022.3160378. doi:10.1109/TITS.2022.3160378.

[7] D. Sacharny, T. C. Henderson, E. Guo, A DDDAS protocol for real-time large-scale uas flight coordination, in: Dynamic Data Driven Applications Systems: Third International Conference, DDDAS 2020, Boston, MA, USA, October 2-4, 2020, Proceedings 3, Springer, 2020, pp. 49–56.

[8] F. Ricca, G. Grasso, M. Alviano, M. Manna, V. Lio, S. Iiritano, N. Leone, Team-building with answer set programming in the gioia-tauro seaport, Theory Pract. Log. Program. 12 (2012) 361–381. URL: https://doi.org/10.1017/S147106841100007X. doi:10.1017/S147106841100007X.

[9] C. Dodaro, M. Maratea, Nurse scheduling via answer set programming, in: M. Balduccini, T. Janhunen (Eds.), Logic Programming and Nonmonotonic Reasoning - 14th International Conference, LPNMR 2017, Espoo, Finland, July 3-6, 2017, Proceedings, volume 10377 of *Lecture Notes in Computer Science*, Springer, 2017, pp. 301–307. URL: https://doi.org/10.1007/978-3-319-61660-5_27. doi:10.1007/978-3-319-61660-5_27.

[10] M. Alviano, C. Dodaro, M. Maratea, Nurse (re)scheduling via answer set programming, Intelligenza Artificiale 12 (2018) 109–124. URL: https://doi.org/10.3233/IA-170030. doi:10.3233/IA-170030.

[11] C. Dodaro, G. Galatà, A. Grioni, M. Maratea, M. Mochi, I. Porro, An asp-based solution to the chemotherapy treatment scheduling problem, Theory Pract. Log. Program. 21 (2021) 835–851. URL: https://doi.org/10.1017/S1471068421000363. doi:10.1017/S1471068421000363.

[12] D. Abels, J. Jordi, M. Ostrowski, T. Schaub, A. Toletti, P. Wanko, Train scheduling with hybrid ASP, in: M. Balduccini, Y. Lierler, S. Woltran (Eds.), Logic Programming and Nonmonotonic Reasoning - 15th International Conference, LPNMR 2019, Philadelphia, PA, USA, June 3-7, 2019, Proceedings, volume 11481 of *Lecture Notes in Computer Science*, Springer, 2019, pp. 3–17. URL: https://doi.org/10.1007/978-3-030-20528-7_1. doi:10.1007/978-3-030-20528-7_1.

[13] D. Abels, J. Jordi, M. Ostrowski, T. Schaub, A. Toletti, P. Wanko, Train scheduling with hybrid answer set programming, Theory Pract. Log. Program. 21 (2021) 317–347. URL: https://doi.org/10.1017/S1471068420000046. doi:10.1017/S1471068420000046.

[14] P. Kopardekar, J. Rios, T. Prevot, M. Johnson, J. Jung, J. E. Robinson, Unmanned aircraft system traffic management (utm) concept of operations, in: AIAA AVIATION Forum and Exposition, ARC-E-DAA-TN32838, 2016.

[15] F. A. Administration, Urban Air Mobility Concepts of Operations (v2.0), Technical Report, April 2023. URL: https://www.faa.gov/sites/faa.gov/files/Urban%20Air%20Mobility%20%28UAM%29%20Concept%20of%20Operations%202.0_1.pdf.

[16] D. Sacharny, T. C. Henderson, A lane-based approach for large-scale strategic conflict management for uas service suppliers, in: 2019 International Conference on Unmanned Aircraft Systems (ICUAS), IEEE, 2019, pp. 937–945.

[17] D. Sacharny, T. C. Henderson, M. Cline, An efficient strategic deconfliction algorithm for large-scale uas traffic management, School Comput., Univ. Utah, Salt Lake City, UT, USA, Rep. UUCS-20-010 (2020).

[18] H. Tang, Y. Zhang, V. Mohmoodian, H. Charkhgard, Automated flight planning of high-density urban air mobility, Transportation Research Part C: Emerging Technologies 131 (2021) 103324.

[19] C. Huang, I. Petrunin, A. Tsourdos, Strategic conflict management for performance-based urban air mobility operations with multi-agent reinforcement learning, in: 2022 International Conference on Unmanned Aircraft Systems (ICUAS), IEEE, 2022, pp. 442–451.

[20] E. L. Thompson, Y. Xu, P. Wei, A framework for operational volume generation for urban air mobility strategic deconfliction, in: 2023 International Conference on Unmanned Aircraft Systems (ICUAS), IEEE, 2023, pp. 71–78.

[21] S. Chen, A. Evans, M. Brittain, P. Wei, Integrated conflict management for UAM with strategic demand capacity balancing and learning-based tactical deconfliction, IEEE Trans. Intell. Transp. Syst. 25 (2024) 10049–10061. URL: https://doi.org/10.1109/TITS.2024.3351049. doi:10.1109/TITS.2024.3351049.

[22] P. Pradeep, G. S. Yarramreddy, N. Amirsoleimani, A. Munishkin, R. A. Morris, M. Xue, K. Chour, K. Kalyanam, Rolling horizon with k-position search method for strategic deconfliction of package delivery uas, in: AIAA AVIATION FORUM AND ASCEND 2024, 2024, p. 4456.