Hyper-heuristic approach based on reinforcement learning for air traffic complexity mitigation

Paveen Juntama * and Daniel Delahaye†
ENAC, French Civil Aviation University, Toulouse 31055 Cedex 4, France

Supatcha Chaimatanan‡
Kasetsart University, Bangkok 10900, Thailand

Sameer Alam§
Nanyang Technological University, Singapore 639798, Republic of Singapore

Airspace capacity has become a critical resource for air transportation. Complexity in traffic patterns is a structural problem, whereby airspace capacity is sometimes saturated before the number of aircraft has reached the capacity threshold. This paper addresses a strategic planning problem with an efficient optimization approach that minimizes traffic complexity based on linear dynamical systems in order to improve the traffic structure. Traffic structuring techniques comprise departure time adjustment, en-route trajectory deviation, and flight level allocation. The resolution approach relies on the hyper-heuristic framework based on reinforcement learning to improve the searching strategy during the optimization process. The proposed methodology is implemented and tested with a full day of traffic in the French airspace. Numerical results show that the proposed approach can reduce air traffic complexity by 92.8%. The performance of the proposed algorithm is then compared with two different algorithms, including the random search and the standard simulated annealing. The proposed algorithm provides better results in terms of air traffic complexity and the number of modified trajectories. Further analysis of the proposed model was conducted by considering time uncertainties. This approach can be an innovative solution for capacity management in the future Air Traffic Management (ATM) system.

*Ph.D. Candidate, Optimization and Machine Learning group, paveen.juntama@enac.fr
†Professor, Optimization and Machine Learning group, daniel.delahaye@recherche.enac.fr
‡Ph.D., Department of Aerospace Engineering, supatcha.chai@ku.th
§Associate Professor, School of Mechanical & Aerospace Engineering, sameeralam@ntu.edu.sg

Nomenclature

d_i	=	decision to modify the trajectory of the ith aircraft
f_i	=	flight level of the ith aircraft in the initial flight plan
$\hat{f}_i$	=	flight level of the ith aircraft in the modified flight plan
h_i	=	ith heuristic operator
i, j, k	=	general index
l_i	=	flight level shifts of the ith aircraft
l_{max}	=	maximum number of flight level shifts
r_t	=	reward at time step t
t_i	=	initial departure time of the ith aircraft in the initial flight plan, s
$\hat{t}_i$	=	departure time of the ith aircraft in the modified flight plan, s
t_ϵ	=	time uncertainty duration, s
w_i	=	set of waypoints of the ith aircraft
M	=	maximum number of waypoints of each flight
N	=	number of aircraft
N_i	=	number of trajectory sample points of the ith aircraft
N_e	=	number of eigenvalues
N_h	=	number of heuristic operators
α	=	learning rate
γ	=	discount factor
ϵ	=	ϵ-greedy control parameter
ϵ_{min}	=	initial value of ϵ
ϵ_{max}	=	final value of ϵ
η_i	=	maximum allowed route length extension coefficient of the ith aircraft
σ	=	value of the reward
τ_i	=	number of departure time shifts, s
τ_a	=	maximum allowed departure time shifts to advance flight departure time
τ_d	=	maximum allowed departure time shifts to delay flight departure time
$\Psi_{i,k}$	=	local traffic complexity associated to the ith aircraft of the kth trajectory sample
Ψ_i	=	summation of local traffic complexities associated to the ith aircraft
Ψ	=	aggregated traffic complexity in the airspace

I. Introduction

The current air traffic management (ATM) system is reaching its structural constraints when considering the continuously growing demand. Although there has been an unprecedented decline in traffic volume due to the COVID-19 crisis, global air traffic is expected to grow long-term. According to the recent forecast in [1], the International Air Transport Association expects overall traveler numbers to reach 4.0 billion in 2024, exceeding pre-2019 levels (103% of 2019 total). This predicted growth will produce a level of demand that requires the efficient use of available capacity. However, as capacity limits are reached, airspace congestion will expand quite rapidly, which will result in extra pressure on the network and further delays. While improvements in air traffic management and systems worldwide have led to gradual capacity enhancements, it has been widely stated that the current system design is near saturation levels.

Air Traffic Flow Management (ATFM) [2] is a service designed to contribute to a safe, efficient, and expeditious flow of air traffic, ensuring that Air Traffic Control (ATC) capacity is set to the highest maximum capabilities. The ATFM is organized into three levels over different time horizons: 1) the strategic level takes place between 18 months and seven days prior to the day of operation and includes planning and coordination activities. The output at this level is the capacity plan for the following year; 2) the pre-tactical level is applied during the six days prior to the day of operation. The ATFM daily plan is the output at this level; and 3) the tactical level is applied on the day of operation. This level updates the daily plan according to the actual traffic and capacity.

To enhance the airspace capacity and optimize the flow of air traffic, demand and capacity balancing (DCB) is a concept that is integrated into the ATFM operation. The DCB is a strategic protection framework that manages a traffic situation compatible with the ATC workload. Several solutions delivered by Single European Sky ATM Research (SESAR) are connected to DCB issues. For example, SESAR has introduced automated support tools for traffic complexity detection and resolution (CAR) [3] to local ATFM units. These support tools assess traffic demand and complexity and adjust capacity in collaboration with the Network Manager (NM) and airspace users. Capacity Optimization in Trajectory-based operations (COTTON) [4] enhances the use of trajectory-based complexity and workload assessment to support Capacity Management (CM) enabled by Trajectory-Based Operations (TBO). Different complexity metrics are applied to traffic assessment over different time horizons. However, this project does not cover any DCB resolution processes.

The level of ATC workload is a significant factor leading to capacity limits of the ATM system [5]. To determine the challenges encountered by air traffic controllers in terms of the difficulty and effort required to manage their workload, *air traffic complexity* metrics are useful, for which reviews have been established [6]. The traffic complexity is first evaluated by comparing the number of aircraft with the sector capacity. However, in some situations, controllers accept the number of aircraft above the capacity threshold with the same control workload. At other times, they refuse traffic even if the number of aircraft is far below the maximum capacity. This evidence shows that the number of aircraft

is insufficient to account for the controller's workload effectively. However, the relationship between the structure of the ATM system and cognitive complexity has been investigated in [7]. As a result, structural considerations related to traffic structure and airspace geometry have become a new factor in building air traffic complexity metrics [8]. Therefore, one way to increase airspace capacity is to structure the traffic into more organized flows in order to reduce the associated workload. However, few studies have investigated such a solution.

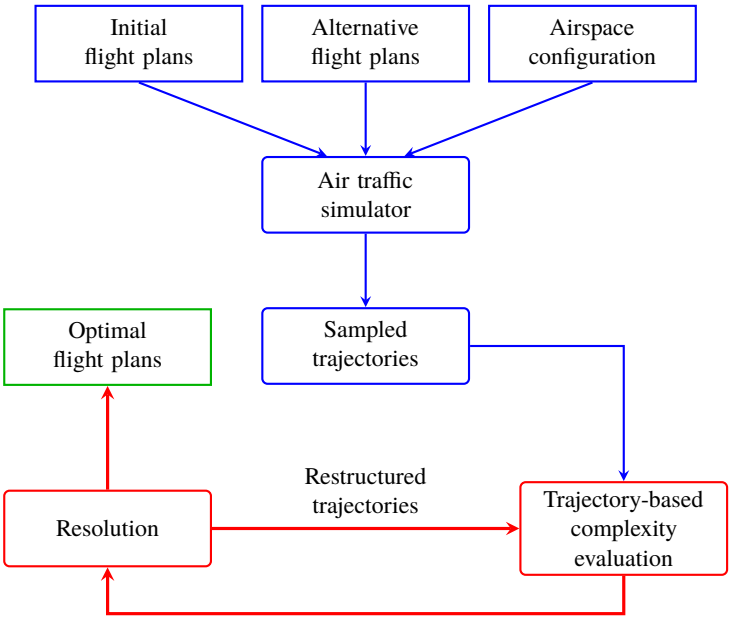


Fig. 1 Strategic 4D aircraft trajectory planning framework with contributions (colored in red) in this paper.

To unlock hidden capacity in a TBO environment, this paper proposes an optimization framework to address a strategic planning problem in such an environment. As illustrated in Fig. 1, the air traffic simulator exports data needed for optimization. It receives flight plans and constructs 4D aircraft trajectories with a fixed time interval. Upon receiving a set of 4D trajectories, the computation of air traffic complexity for all trajectories is performed. The objective of the optimization algorithm is to provide optimal trajectories in order to minimize the complexity of air traffic. Significant contributions in this paper are summarized as follows:

1) The first contribution is a trajectory-based complexity evaluation to compute air traffic complexity between trajectories over a full-time horizon, including uncertainty. Such an evaluation is developed from the air traffic complexity metric based on linear dynamical systems.
2) A method for strategic planning of 4D trajectories is proposed using departure time adjustment, route assignment and flight level allocation.
3) A novel hyper-heuristic approach based on reinforcement learning is proposed to find an optimal trajectory plan for a full day of traffic in the French airspace. The Q-learning mechanism is integrated into the learning process, providing a memory of each low-level heuristic's long-term performance at a given state for selection. The

low-level heuristics are designed and implemented based on intensification and diversification techniques. The proposed approach uses the ϵ-greedy selection method for heuristic selection and a metropolis-based criterion for the move acceptance method.

4) In comparison with different algorithms, simulation results demonstrate the performance of the proposed algorithm, which can effectively reduce the air traffic complexity by efficiently re-planning aircraft trajectories based on initial flight plans.
5) Passenger delays at the departure airport, human intervention, adverse weather conditions, and instability in the network can generate uncertainties in a TBO environment. Therefore, the proposed strategic planning procedure is also extended with time uncertainty considerations. The proposed methodologies are implemented and tested with a full day of traffic in the French airspace, which involves more than 8000 flights.

The following sections of this paper are organized: Section II reviews related works on aircraft trajectory optimization and air traffic complexity. The development of the mathematical formulation and the objective function is presented in Section III. The trajectory processing method is illustrated in Section IV. Section V gives details of proposed resolution algorithms, and Section VI illustrates experimental studies and preliminary results. Finally, our conclusions are presented in Section VII.

II. Previous related works

This section first presents a review of the air traffic complexity. Next, it highlights related approaches involving aircraft trajectory optimization. Finally, the historical development of the hyper-heuristic approaches is also given at the end of this section.

A. Air traffic complexity

The Dynamic Density (DD) suggested by Laudeman et al. from NASA [9] is one of the first air complexity assessments considering the number of aircraft and traffic structures. Combining traffic features produces a single positive real number reflecting the level of complexity. In the studies of Sridhar et al. [10], the DD is also used to determine a predictive model up to a given time horizon. However, the features used in the DD are not enough to explain airspace complexity. This motivation has driven the development of new approaches to complexity measures, such as the fractal dimension [11], the input-output approach [12], and the intrinsic complexity [13].

The fractal dimension approach [11] is a scalar measure based on the geometrical complexity of the traffic pattern observed over an infinite time horizon. This measure evaluates the number of degrees of freedom used in the airspace. A higher fractal dimension indicates more degrees of freedom. The input-output approach in [12] assesses complexity based on the control effort required to bring a new aircraft into the airspace safely. In order to calculate air traffic complexity, the authors highlight aircraft heading change in response to an intrusive aircraft within a sector.

Intrinsic complexity metrics were first introduced in [13]. These metrics quantify airspace congestion instead of simply counting the number of aircraft. Aircraft positions and speed vectors are used to compute such metrics. The research has investigated two classes of indicators: the former uses geometrical properties to build a metric; and the latter formalizes a representation of air traffic as a dynamical system. However, the geometrical metrics would fail when a traffic situation has a more complex organization. To deal with these problems, metrics based on dynamical systems can quantify the disorder in a complex traffic situation. The metrics can distinguish different air traffic situations: translation, convergence, divergence, and rotation. Few studies have been performed to investigate the use of this kind of complexity metric for automated ATM systems.

B. Aircraft trajectory optimization

For several decades, the problem of improving automation and decision support for ATFM has been a rich subject of research. Comprehensive literature surveys on this topic have been presented in [14, 15]. Mathematical models developed in relation to this problem are typically based on considering either individual 4D (space and time) aircraft trajectories [16] or flow aggregation modeling [17, 18]. Several approaches aim to satisfy the capacity constraints to reduce congestion in the network, whereas others focus on minimizing the potential conflicts between aircraft. One or more of the following actions are mainly used in such approaches to reduce the congestion: ground delays, airborne speed regulation, traffic rerouting, flight level allocation, and flight cancellation [19].

Many studies have been conducted to design conflict resolution methods in trajectory-based planning. For example, Cafieri et al. [20] present a mixed-integer nonlinear model for the conflict resolution problem. To avoid conflicts between two aircraft, such aircraft are regulated with speed acceleration or deceleration in a time window. Dougui et al. [21] modify en-route trajectories using a light propagation model and solve conflicts between aircraft in a 4D environment by the Branch and Bound algorithm. This algorithm is also applied in [22] to avoid conflicts between trajectories using a slot allocation approach. Durand et al. [23] present genetic algorithms to overcome aircraft density on a particular flight route. Chaimatanan et al. [24] developed a hybrid metaheuristic approach using the simulated annealing algorithm improved with a hill-climbing local search method to minimize aircraft trajectory interactions during strategic flow management. To separate aircraft trajectories, ground delay and traffic rerouting techniques are applied. This approach can generate conflict-free trajectories for a full day of air traffic in the European airspace. Later, the same authors extend their work in [25] by taking uncertainties into account to improve the robustness of their approach.

Several studies have paid more attention to ATC workload to minimize airspace congestion. Previous studies in [26] attempt to reduce the airspace congestion in terms of workload induced in a control sector. Such workload is a function of the conflict workload, the coordination workload, and the monitoring workload. These studies present a flow modeling of the traffic network and solve the route-time allocation problem using the genetic algorithm. Briel

et al. [27] provide a method to reduce congestion at the tactical level by building temporary routes in crowded areas. A complexity metric is built based on a convergence indicator, which can quantify congestion levels in areas where traffic situations are difficult to organize by air traffic controllers. At the strategic level, Juntama et al. [28] attempt to minimize air traffic complexity in airspace by re-allocating either departure times or flight levels. In this work, the König theorem is used to measure traffic complexity. A distributed metaheuristic algorithm based on the metropolis criterion is implemented to solve the associated optimization problem.

C. Hyper-heuristic

Over the past decade, more attention has focused on the provision of hybrid metaheuristic approaches that do not purely follow the paradigm of a single traditional metaheuristic [29]. With these approaches, researchers can combine additional algorithms in order to achieve better performance in solving their hard optimization problems. However, a certain approach is not sufficiently generic to adapt to changing search spaces, even for the different instances of the same problem.

Hyper-heuristic is one of the hybrid metaheuristic approaches that has recently received considerable attention for addressing some of the above issues. It can select and apply a suitable low-level heuristic to produce a good solution at each decision point [30]. Unlike traditional metaheuristics and other hybrid metaheuristics, which directly operate on the solution space, the hyper-heuristic offers flexible integration and adaptive manipulation of low-level heuristics. Moreover, it can evolve its heuristic selection and acceptance mechanism in searching for a good-quality solution.

Early work on hyper-heuristics focused on using metaheuristics at a high level to manage or generate low-level heuristics. Several population-based metaheuristics have been applied to solve combinatorial optimization problems, such as genetic programming [31], ant colony optimization [32, 33], tabu search [34], and particle swarm optimization [35, 36]. However, one of the motivations behind the hyper-heuristic is to facilitate applicability to different problem instances having different characteristics as well as different problem domains. To deal with various instance characteristic, the previous work in [37] gives a Monte Carlo tree search (MCTS) algorithm to search the tree, involving a sequence of low-level heuristics to be applied to the given instance solution. Machine learning techniques also become crucial for hyper-heuristic to interact with the search environment to adapt the selection of heuristics during the search process. Learning in the hyper-heuristic can be classified as offline or online learning. In [38], a case-based reasoning system has been proposed for offline learning to solve exam timetabling problems. The set of problems and good solutions is stored and used to train such a system before solving unseen problem instances. As presented in [39–42], most of the existing online learning hyper-heuristics are reinforcement learning (RL) mechanisms that use reward and punishment schemes [43] in the hyper-heuristic frameworks. However, most previous studies perform a straightforward learning process without considering the long-term reward value of taking specific action in a specific state.

Several studies have confirmed the effectiveness of using Q-learning as a learning algorithm to historically select an appropriate low-level heuristic at a given decision point. Q-learning proceeds by estimating the best state-action pair by manipulating memory based on a Q-table. Each value in such a table expresses the long-term reward value of selecting a particular action at a particular state. In the context of the hyper-heuristic, Choong et al. [44] automatically designed an Iterated Local Search-based hyper-heuristic by using Q-learning to select a proper pair of selection-move acceptance during each episode of the optimization process. An action represents a low-level heuristic with a selection-move acceptance pair. The learning procedure operates by choosing actions based on their Q-values. Mosadegh et al. [45] presented an approach based on simulated annealing to solve the mixed-model sequencing problem (MMSP) with stochastic processing times. The approach incorporates a Q-learning algorithm to select appropriate heuristics through its search process. The selection mechanism is based on ϵ-greedy method. Duflo et al. [46] proposed a Q-learning-based hyper-heuristic to automate the design of UAV swarming behaviors in order to optimize the coverage of a connected swarm, the so-called Coverage of a Connected-UAV Swarm (CCUS). A CCSU heuristic corresponds to a move of each UAV to its next destination. The UAVs move to their destinations to maximize their fitness functions while the Q-learning aims to find the best possible definition for the fitness function.

As presented in [47], an adaptive metaheuristic approach with roulette wheel-based heuristic selection and additive reinforcement learning mechanism has addressed a problem in structuring air traffic. Traffic restructuring techniques consisting of rescheduling departure time and reshaping routes have been formulated as an optimization problem. The objective of this work aims to minimize complexity in air traffic. The complexity metric based on linear dynamical systems has been used to identify the level of disorder of 4D trajectories in a TBO environment. Preliminary results presented in this work show the potential of this approach to minimize air traffic complexity for short-term and long-term trajectory planning. However, these studies have not treated the strategic planning problem in detail, and little attention has been paid to selecting a low-level heuristic and the learning process.

This paper extends the work presented in [47] by addressing the associated strategic planning problem. We reformulate an alternative mathematical formulation to improve the structure of air traffic flow by minimizing air traffic complexity. Traffic structuring strategies include departure time adjustment, traffic rerouting, and flight level allocation. The air traffic complexity metric based on linear dynamical systems has been demonstrated by Delahaye et al. [13] to quantify the disorder of air traffic situations at a given time. Regarding the associated strategic planning, the trajectory-based complexity evaluation is developed from such a metric to compute air traffic complexity for any airspace over a full-time horizon, including uncertainty. However, applying a flight-level allocation strategy causes the solution space to become larger than in previous work [47]. We propose a new hyper-heuristic framework representing a new high-level strategy and a set of low-level heuristics to overcome this impact. The high-level strategy represents e-greedy selection as the heuristic selection and a metropolis-based criterion as the move acceptance. The low-level heuristics are developed based on intensification and diversification techniques. The state representation

of the environment relies on a diversification-intensification cycle [41]. The learning process employs a Q-learning agent such that the proposed hyper-heuristic approach can learn how to improve the decision at a given time step. The learning behavior is controlled by a learning rate and a discount factor. The learning rate controls how fast the Q-values are changed, and the discount factor regulates how the agent cares about long-term cumulative rewards. This research is motivated to use the Q-learning algorithm as it has shown significant success in keeping track of good-performing low-level heuristics in particular states [44–46]. We also propose a methodology to consider the time uncertainty of aircraft trajectories at the strategic level. The proposed methodology has been implemented and validated with air traffic data in the French airspace.

III. Mathematical formulation

The problem in this work is to determine a set of optimized 4D trajectories in order to maximize the traffic structure which is linked to the associated complexity. In this section, we start by reformulating the optimization problem. Finally, air traffic complexity metric based on linear dynamical system is introduced for the objective function.

A. Input data

The problem instance is given by:

1) $\mathcal{F}$: Set of flight,
2) N: Number of aircraft,
3) t_s: Trajectory sampling time,
4) t_i: Departure time of the initial flight plan,
5) f_i: Flight level of the initial flight plan,
6) M: Maximum number of waypoints of each flight,
7) τ_a: Maximum allowed advance of the departure time,
8) τ_d: Maximum allowed delay of the departure time,
9) l_i: Maximum allowed flight level shifts,
10) d_i: Maximum allowed route length extension coefficient of the ith flight,
11) L_i: Length of the initial en-route segment for each flight.

B. Decision variables

The three following decision variables are used to structure the aircraft trajectories: departure time adjustment, alternative route, and flight level allocation.

1. Departure time adjustment

The first decision variable is the departure time shift τ_i of the ith flight. As given in initial flight plans, the ith flight has an initial departure time t_i. If the flight is selected to perform this option, the new departure time will be expressed as follows:

$$\hat{t}_i = t_i + \tau_i \tag{1}$$

2. Flight level allocation

We define another decision variable associated with the ith flight, a flight level shift l_i, to structure aircraft trajectories in the vertical plane. Thus, the following equation is used to calculate the new allocated flight level of the ith flight:

$$\hat{f}_i = f_i + l_i \tag{2}$$

where z_i is the initial flight level of the ith flight. Figure 2 illustrates an initial trajectory with two alternative flight levels.

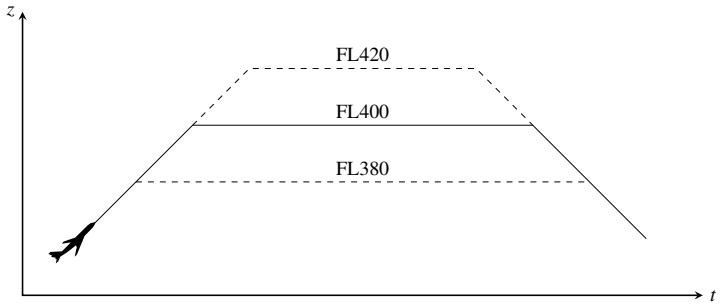


Fig. 2 An initial trajectory (solid line) with two optional vertical profiles (dashed lines).

Regarding to the instrument flight rules (IFR), the aircraft cruises eastbound with the odd numbered flight levels, whereas the westbound aircraft cruises at the even numbered flight levels. Therefore, the aircraft can change its flight level in the same direction with a minimum shift of 2000 ft.

3. Alternative route

The third option is to reconstruct an alternative route by adding a set of waypoints defined by:

$$w_i = \{w_i^m | w_i^m = (w_{ix'}^m, w_{iy'}^m)\}_{m=1}^M \tag{3}$$

The set of waypoints allows us to give a new shape with straight-line segments deviated from the initial en-route segment, as illustrated in Fig. 3. More details related to this method to modify the trajectory are presented in [24].

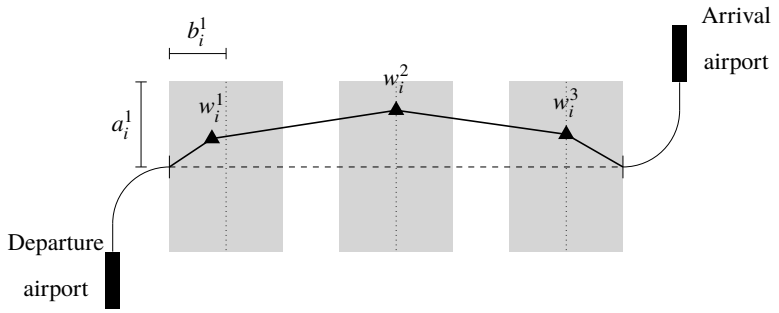


Fig. 3 Initial (dashed line) and alternative (solid line) trajectory with three additional waypoints within possible rectangular areas.

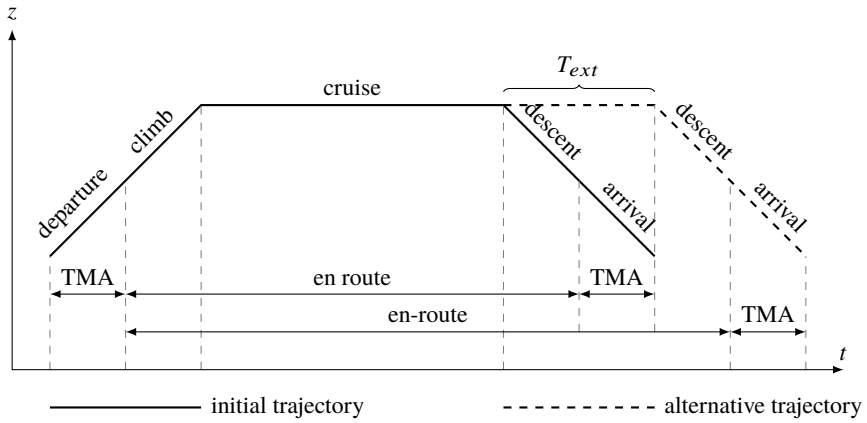


Fig. 4 Updated altitude profile due to the impact of the horizontal route deviation.

We note that such an alternative trajectory is likely to yield an increase in flight duration compared with the initial trajectory. To compensate for this increased flight duration, the altitude profile will be updated to avoid a premature descent. Let T_{ext} be the increased flight duration of the ith flight. In the case of a regional flight, where flight phases are all performed in the same (current) area, the altitude profile is updated by extending the cruise phase at the top of descent for a duration of T_{ext} as illustrated in Fig. 4.

However, for a flight whose origin or destination airports are outside of the current airspace, the top of descent of such flight may not be in the considered airspace. Therefore, we update the altitude profile by extending the flight at maximum altitude (in the current airspace) for a duration T_{ext}. Here, the vertical profile is updated according to six possible cases according to whether the origin/destination airports are in the current airspace or not and whether the initial trajectory has a cruise (constant-level) phase or not, as illustrated in Fig. 5.

C. Problem constraints

The constraints are defined as follows:

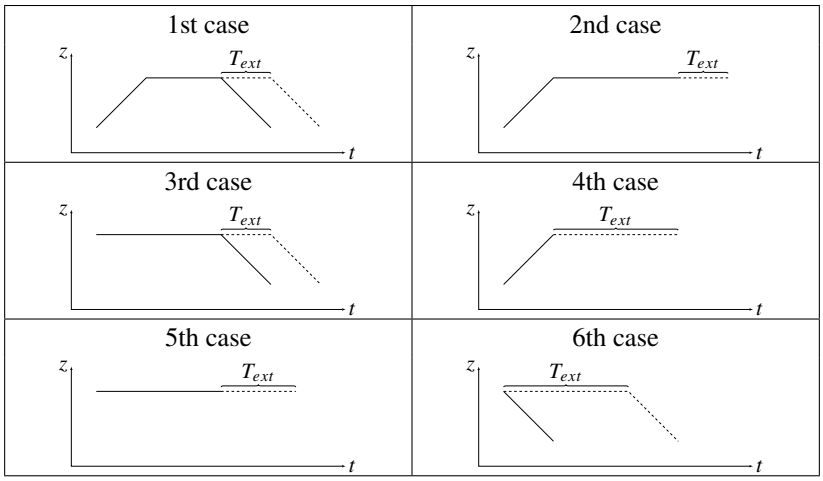


Fig. 5 Six possible changes in vertical profiles caused by the horizontal route deviation.

1. Maximum allowed departure time shifts

The departure time shift is restricted to a maximum advance ($-\tau_a$) and maximum delay ($+\tau_d$). The range of departure time shifts can be expressed as follows:

$$\tau_i \in \{-\tau_a, -(\tau_a - 1), \cdots, 0, \cdots, \tau_d - 1, \tau_d\} \tag{4}$$

2. Maximum changes of flight level

We define the restricted changes of flight level in the symmetrical range as follows:

$$l_i \in \{-l_{max}, -l_{max} + 1, ..., 0, ..., l_{max} - 1, l_{max}\} \tag{5}$$

3. Maximum route length extension

To limit the route length extension, the alternative en-route profile of the ith flight shall satisfy:

$$L_i(w_i) \leq (1 + \eta_i) \cdot L_i \tag{6}$$

where $L_i(w_i)$ is the new length of the alternative en-route profile determined by the set of waypoints w_i and η_i denotes the maximum allowed route length extension coefficient of the ith flight.

4. Allowed waypoint location

As illustrated in Fig. 3, alternative trajectories are constructed with $M = 2$ waypoints, where the location of each waypoint is constrained to be in a rectangular shape. For each trajectory, the normalized longitudinal component $w^m_{ix'}$

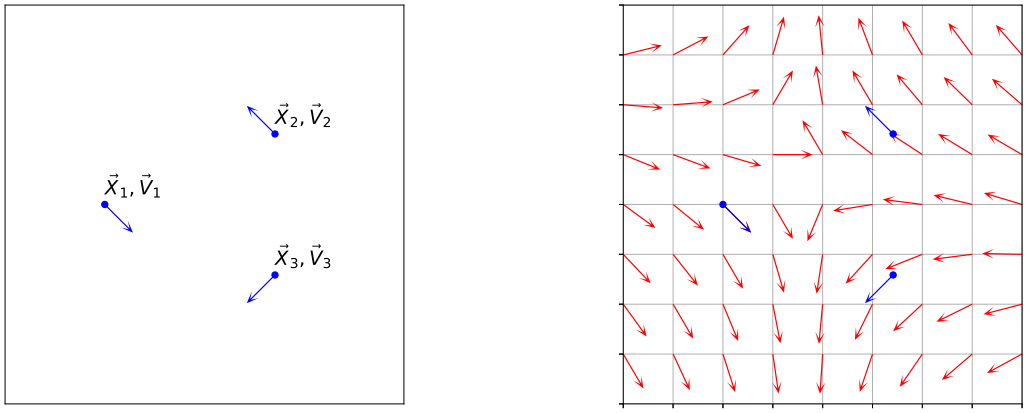
is set to lie in the interval $\left[\left(\frac{m}{1+M} - b_i\right), \left(\frac{m+1}{1+M} + b_i\right)\right]$ where b_i is a parameter that defines the range of possible normalized longitudinal component of the mth virtual waypoint on the ith trajectory. This constraint encourages the algorithm to select b_i so that $b_i < \frac{1}{2(M+1)}$

Similarly, the normalized lateral component $w_{iy'}^m$ is restricted to lie in the interval $[-a_i, a_i]$ where $a_i \in [0, 1]$ is a parameter that defines the range of possible normalize lateral location of the mth virtual waypoint on the ith trajectory, chosen a priori so as to satisfy Eq. (6). More detail about the method to modify the trajectory is presented in [24].

D. Objective function

This subsection introduces the model and analysis of the air traffic complexity metric and then clarifies the approach to develop the objective function with such a metric.

The complexity metric based on linear dynamical system is adapted to evaluate the congestion in the traffic over a full-time horizon. Instead of simply counting the number of aircraft, the traffic complexity associated with a traffic situation can be measured using aircraft position and speed vectors.



(a) **(b)**

Fig. 6 Traffic situation at a given time: (a) measurement or observation vectors, (b) vector fields derived by the linear dynamical model.

Assume that there are some aircraft present in a given area at a given time. For each aircraft, we consider the two observation vectors (see Fig. 6a): a position measurement $\vec{X}_i = \begin{bmatrix} x_i & y_i & z_i \end{bmatrix}^T$ and a speed measurement $\vec{V}_i = \begin{bmatrix} vx_i & vy_i & vz_i \end{bmatrix}^T$. To compute the local complexity associated with a given traffic situation, it is necessary to formulate such a situation into a linear dynamical system, as presented in Fig. 6b. Such a system is controlled by the following equation of motion:

$$\dot{\vec{X}}_i = \mathbf{A} \cdot \vec{X}_i + \vec{B} \tag{7}$$

where $\dot{\vec{X}}_i$ is the estimated speed vector associated with each point in the state space, $\vec{X}_i$ is the position vector, the coefficient matrix $\mathbf{A}$ is the linear mapped between $\vec{X}$ and $\dot{\vec{X}}$, and $\vec{B}$ represents the static behavior of the system.

To determine an accurate dynamical system model best fitted to observations in the state space, it is necessary to find the matrix $\mathbf{A}$ and vector $\vec{B}$, which minimize the error between speed observations and estimated speed vectors. The minimization problem can be expressed as follows:

$$\mathbf{A}^*, \vec{B}^* = \underset{\mathbf{A},\vec{B}}{\operatorname{argmin}} \quad \sum_{n \in N} \left\| \vec{V}_n - \left(\mathbf{A}.\vec{X}_n + \vec{B} \right) \right\|^2 \tag{8}$$

The calculation of the matrix $\mathbf{A}^*$ and vector $\vec{B}^*$ is detailed in [13].

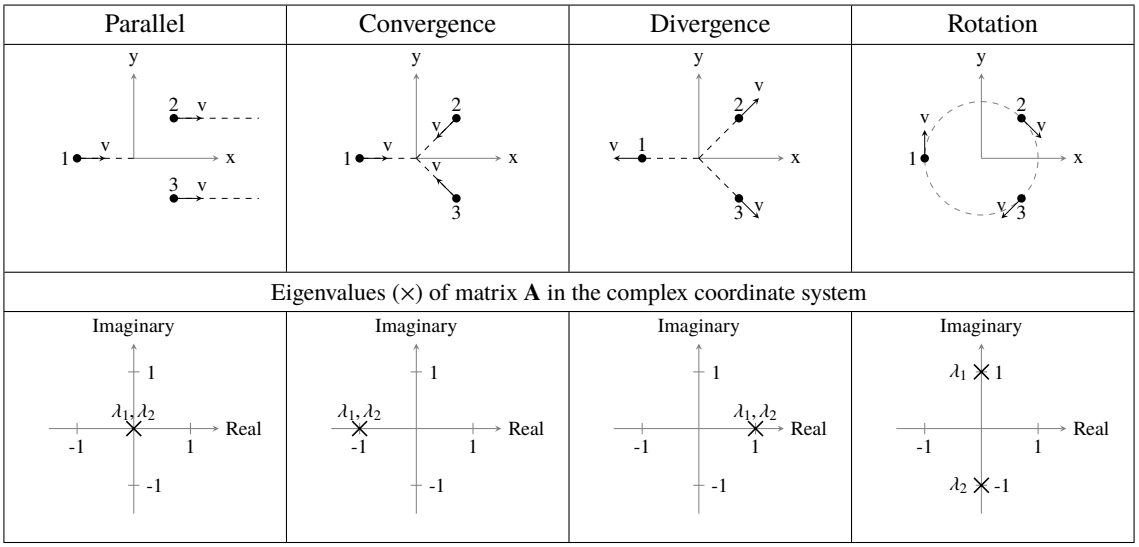


Fig. 7 Eigenvalue locations for four different traffic situations.

As an illustration, Fig. 7 shows the locations of the eigenvalues of matrix $\mathbf{A}$ for four artificial traffic organizations: parallel, convergence, divergence, and rotation. With the first situation, the eigenvalues are null because the aircraft are flying in a parallel flow: distances between aircraft remain unchanged with time. In contrast with the second situation, the eigenvalues are real negative; the system evolves in a contraction mode, and the four aircraft are converging: the norms of the relative distances between aircraft decrease with time. The third situation represents an expansion evolution where eigenvalues are real positive, and the aircraft are diverging: the relative distances increase with time. Finally, the rotation situation is associated with full imaginary and null real parts of eigenvalues because the aircraft stay at the same distance from each other in a curl movement.

Having obtained matrix $\mathbf{A}$ and its eigenvalues, the local complexity metric is defined as follows:

1) The metric will be the addition, in absolute values, of the negative real parts of eigenvalues.

2) If none of the eigenvalues possess a negative real part, the metric will be 0.

With this definition, we measure the intensity of the convergence tendency in a traffic situation at a given time. If the metric is 0, the traffic will be organized, and complexity will be 0. It must be noticed that when the resulting

behavior is a divergence motion, it is considered an organized pattern (the metric is then 0). The divergence motion will not affect the workload of the air traffic controller since divergent aircraft will not give rise to air traffic conflicts.

According to traffic situations in Fig. 7, the associated eigenvalues are used to produce the local complexity metrics based on the preceding definition. The numerical results are summarized in Table 1. No negative real parts of eigenvalues are present for the following situations: parallel, divergence, and rotation. Therefore, the associated metrics are equal to 0. Such metrics show that the associated situations are well organized without any risk of potential conflicts. On the contrary, all eigenvalues for the convergence situation possess negative real parts. The associated metric can be determined by adding eigenvalues using their absolute values (the metric equals 2). Therefore, such a non-null metric represents a risk of potential conflicts, where the air traffic controller would pay more attention to the situation. Thus, a higher metric value indicates a greater level of risk.

Table 1 Eigenvalues at any instance of time from four traffic situations.

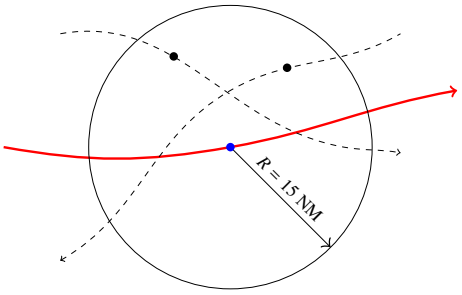
Situations	Eigenvalues		Complexity metric
	λ_1	λ_2	
Parallel	0	0	0
Convergence	-1	-1	2
Divergence	1	1	0
Rotation	$-j$	j	0

This paper uses the preceding definition to obtain the local complexity metric $\Psi_{i,k}$ for the ith aircraft at the kth trajectory sample in a TBO environment, the process begins by identifying the traffic situation around the ith aircraft at the kth trajectory sample by considering the neighboring aircraft in horizontal and vertical dimensions as represented in Fig. 8. The horizontal search space is dedicated as a circular region with a radius of 15 NM, centered at each *reference aircraft* position and always with the same dimension over the whole evaluation phase. In RVSM (Reduced Vertical Separation Minima) airspace, en-route aircraft vertical separation is 1000 ft. Therefore, the vertical search space is created to find the neighboring aircraft whose altitudes are at risk of loss separation in relation to the reference aircraft. The speeds and positions of neighboring aircraft are then considered for the computation of the local complexity metric.

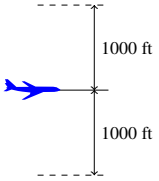
Assume that the positions and speeds of aircraft are present in the search space under the preceding conditions, such observations are used to determine the matrix $\mathbf{A}$ from Eq. (8). the matrix $\mathbf{A}$ represents a set of complex eigenvalues of the ith aircraft at the kth trajectory sample, denoted by $\Lambda_{i,k} = \left\{\lambda_{i,k}^{(1)}, \lambda_{i,k}^{(2)}, \ldots, \lambda_{i,k}^{(N_e)}\right\}$. Then, we can produce the local complexity metric as follows:

$$\Psi_{i,k} = \sum_{n \in \mathcal{N}} |\mathrm{Re}\left\{\lambda_{i,k}^{(n)}\right\}|, \mathcal{N} = \left\{n : \mathrm{Re}\left\{\lambda_{i,k}^{(n)}\right\} < 0\right\} \tag{9}$$

To compute the summation of the local complexity along the trajectory of the ith aircraft, we can determine Ψ_i as



(a) Horizontal search space



(b) Vertical search space

Fig. 8 Search space for the local complexity computation.

follows:

$$\Psi_i = \sum_{k=1}^{N_i} \Psi_{i,k} \tag{10}$$

where N_i is the number of trajectory samples of the ith aircraft.

Therefore, the aggregated complexity Ψ determined from all aircraft in airspace can be determined as follows:

$$\Psi = \sum_{i=1}^{N} \Psi_i = \sum_{i=1}^{N} \sum_{k=1}^{N_i} \Psi_{i,k} \tag{11}$$

where N is the number of aircraft.

E. Time uncertainties

The extension of the time uncertainty is taken into account when an aircraft may arrive early or late at a given location. This situation might be caused by passenger delay, airport, and network problems. We apply the maximum time error t_ϵ to all aircraft within the search space. This error is the predicted arrival time of aircraft at a given position under uncertainty lying in the interval t_ϵ where $t - t_\epsilon/2 \leq t \leq t + t_\epsilon/2$ and t is the actual arrival time of the aircraft. Fig. 9 shows the time uncertainty example. All observation vectors presented in this time interval will be considered for the metric computation.

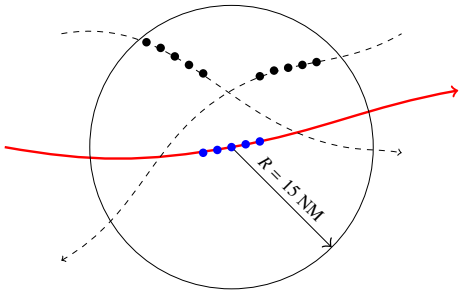


Fig. 9 Searching for neighboring aircraft with consideration of time uncertainty.

IV. Trajectory processing

To evaluate the performance of an aircraft in the airspace, we first build its 4D trajectory based on the associated decision. Then, such 4D trajectory is inserted into the airspace for evaluation. When it must be changed, it is first removed from the airspace, then it is modified based on a new decision, and inserted again into the airspace for evaluation. To use the decision, the following two methods are introduced in this section: decision encoding and decision evaluation.

A. Decision encoding

Let γ_i be the trajectory of the ith flight. When we initiate the decision d_i, the system shall register the discretized trajectory γ_i into the airspace. Let $\mathcal{D} = \{d_i : \forall i \in \mathcal{F}\}$ be a set of decisions where d_i represents a set of decision variables associated with the ith aircraft, $d_i := (\tau_i,\ l_i,\ w_i)$. To modify the trajectory γ_i in the airspace, the system removes the current trajectory γ_i and then inserts a new trajectory associated with the current decision d_i into the 4D airspace.

B. Decision evaluation

In order to compute the local complexity associated with each trajectory sample, it is necessary to locate neighboring aircraft within a search space. The naive approach, which performs a pairwise comparison, leads to a large amount of computation. To overcome this exhaustive computation, the concept of hash table is used to store and retrieve trajectories which are identified by their 4D coordinates (3D space and time coordinates). With an example depicted in Fig. 10a, the reference trajectory (green line) and three other trajectories (black line) are stored in the 4D grid system. Square markers are observations present at a current time step whilst circle markers are observations that have occurred at different time steps. The algorithm is checking for neighboring aircraft (black square markers) to take them into account when computing local complexity associated with the reference trajectory sample at a current time (red square marker).

Fig. 10b shows the search results obtained from the search space in Fig. 10a at the kth trajectory sample. Then, we can compute the local complexity associated with the trajectory γ_1 at the kth trajectory sample ($\Psi_{1,k}$) from position and speed observations ($\vec{X}, \vec{V}$) present in this search space by using Eq. (9). Thus, we can determine the full complexity (Ψ_1) associated with the trajectory γ_1 by repeating such a local process until the last trajectory sample and then perform the computation by using Eq. (10).

V. Resolution algorithms

In this section, the hyper-heuristic based on the Q-learning (HQL) framework is proposed to solve our strategic planning problem. Before outlining the details of HQL, the following methods are introduced in this section:

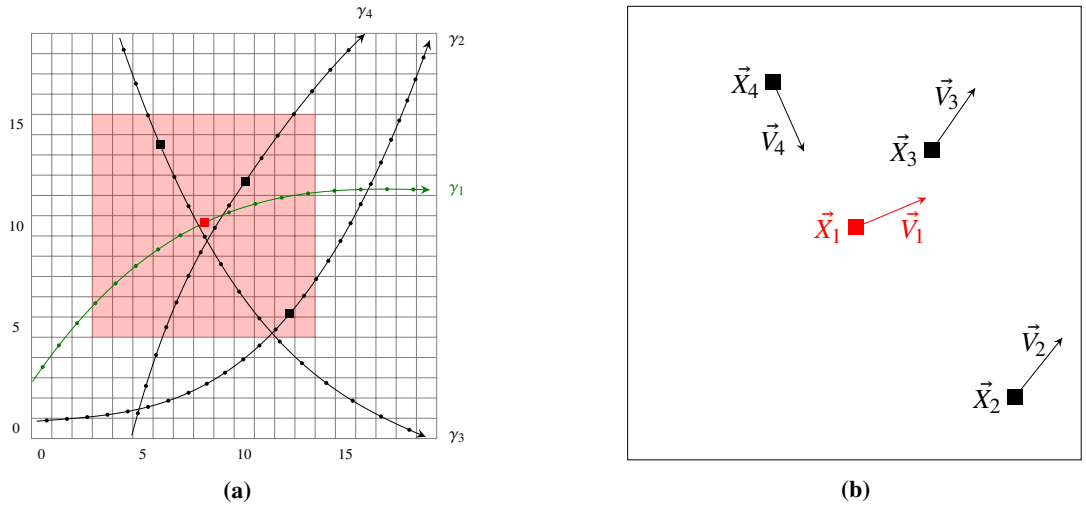


Fig. 10 Traffic situation processing of the reference trajectory (green line) at a given time: (a) 4D grid with the search space (red area) surrounding the current position (red square marker) of the reference trajectory, (b) captured observations in the search space for computing the local traffic complexity.

simulation-based evaluation, reinforcement learning, Q-learning, and hyper-heuristics.

A. Simulation-based evaluation

In the context of our large-scale optimization, the objective functions must be evaluated in a simulation environment. Fig. 11 illustrates the general implementation of the optimization architecture with the evaluation-based simulation. The optimization algorithm generates a set of state variables X. Such variables are then passed into the simulation environment for evaluation through the simulation process. The performance y calculated by the simulation is used as a criterion to decide the subsequent operation of the optimization process. A more detailed explanation of this approach can be found in [48].

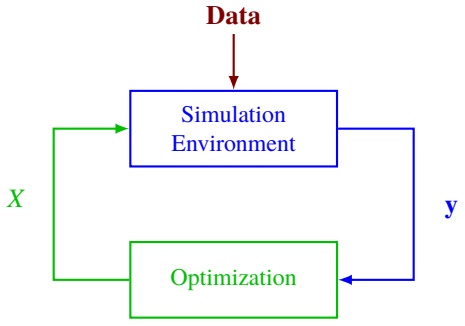


Fig. 11 Objective-function evaluation based on a simulation process.

B. Reinforcement learning

The typical RL problem is formalized as a discrete-time stochastic control process in which an agent interacts with its environment described by a set of states. A basic reinforcement learning agent interacts with its environment in discrete time steps. At each time t, the agent receives the current state s_t and reward r_t. It then chooses an action a_t from the set of available actions, which is subsequently sent to the environment. The environment moves to a new state s_{t+1} and the reward r_{t+1} associated with (s_t, a_t, s_{t+1}) is determined. The goal of a reinforcement learning agent is to learn a policy that maximizes the expected cumulative reward. An agent can use one or more of the following components to learn a policy: 1) a value function that provides a prediction of how good a particular action will be in each state; 2) an exact policy; and 3) a model of the environment that contains complete information of the process (such as transition functions, reward functions etc.).

C. Q-Learning

Q-learning [49] is a value-based RL method where the agent aims to obtain the optimal state-action values or Q-values by interacting with the environment. The agent uses the Q-table to store and update Q-values. The Q-table is a two-dimensional lookup table whose columns and rows represent actions and states, respectively. At time step t, when the action a_t is taken in the state s_t, the agent receives a reward r, and moves to the next state s_{t+1} and the state-action value $Q(s_t, a_t)$ based on Q-learning is updated as follows:

$$Q\left(s_t, a_t\right) := \left(1 - \alpha\right) Q\left(s_t, a_t\right) + \alpha \left(r_t + \gamma \max_{a \in \mathcal{A}} Q\left(s_{t+1}, a\right)\right) \tag{12}$$

where $\alpha \in [0, 1]$ denotes the learning rate, $\gamma \in [0, 1]$ denotes the discount factor and r denotes the immediate reward signal. The Q-table converges using iterative updates for each state-action pair. In our application context, the ϵ-greedy approach is used in order to efficiently establish the convergence of the Q-table.

In ϵ-greedy selection, the agent uses both exploitation and exploration to take advantage of prior knowledge and look for new options, respectively. At first, Q-values of the Q-table are initialized with the same values. To establish the convergence of the Q-table using iterative updates, exploration and exploitation trade-off ϵ is used. At the current state s_t, the exploration strategy randomly selects the action, whereas the exploitation strategy selects the action a_t pointing to the largest Q-value.

D. Hyper-heuristics

Hyper-heuristics are divided into two main categories: first, generation hyper-heuristics, which generate new heuristics; and second, selection hyper-heuristics, which select a heuristic for application from a set of low-level heuristics. A traditional selection hyper-heuristic framework consists of two levels: first, the low-level, which contains

a representation of the problem, evaluation function(s), and a set of low-level heuristics; and second, the high-level, which performs two separate tasks: heuristic selection and move acceptance. The selection method is accountable for selecting a low-level heuristic from the set of low-level heuristics and applying it to the solution. The second task is to decide the acceptance or rejection of the new solution. Both heuristic selection and move acceptance methods have a crucial impact on the performance of the hyper-heuristic model. Choosing a suitable heuristic selection and a move acceptance method for a particular problem is a non-trivial task when designing a robust hyper-heuristic model.

In the interest of the learning process between heuristic selection and the move acceptance module, hyper-heuristics are capable of learning which sequences or heuristics are more effective during the search process. The learning process is classified as online or offline. The online algorithm learns during the search while solving the main problem. In contrast, the offline algorithm learns by solving a set of training instances that prepare the algorithm to encounter new instances effectively. Hyper-heuristics operate without the need of any information regarding the functionality of the low-level heuristics. In return, they provide useful feedback, such as the utilization rate of each heuristic and the change in the objective function. This information is crucial for the learning process.

E. Hyper-heuristic based on Q-learning

The HQL is a selection hyper-heuristic framework with online learning process which is extended from a traditional hyper-heuristic framework. The HQL framework applies the Q-learning method in the learning process and the metropolis-based criterion in the move acceptance method. As depicted in Fig. 12, the algorithm uses the Q-learning agent to select a heuristic operator or low-level heuristic, which would then be applied to generate a candidate decision and to recalculate its performance under the simulation environment. The move acceptance method makes a decision regarding the acceptance or rejection of a candidate decision. If such a candidate decision can improve its performance, the current decision will be updated. Otherwise, all changes in the solution space involved with such a candidate decision will be discarded, thanks to the comeback operation. The Q-learning agent determines the rewards by checking the current state, the selected operator and the evolution of decisions. Q-values in the Q-table will be updated with rewards to favor the selection of heuristic operators when the algorithm comes to visit this state again.

F. Low-level heuristics

The specialization of HQL for a particular optimization problem requires the definition of intensification and diversification operators. In this paper, generation operators perform the diversification task. Some improvement heuristics are used for intensification. In this application context, six heuristic operators are proposed.

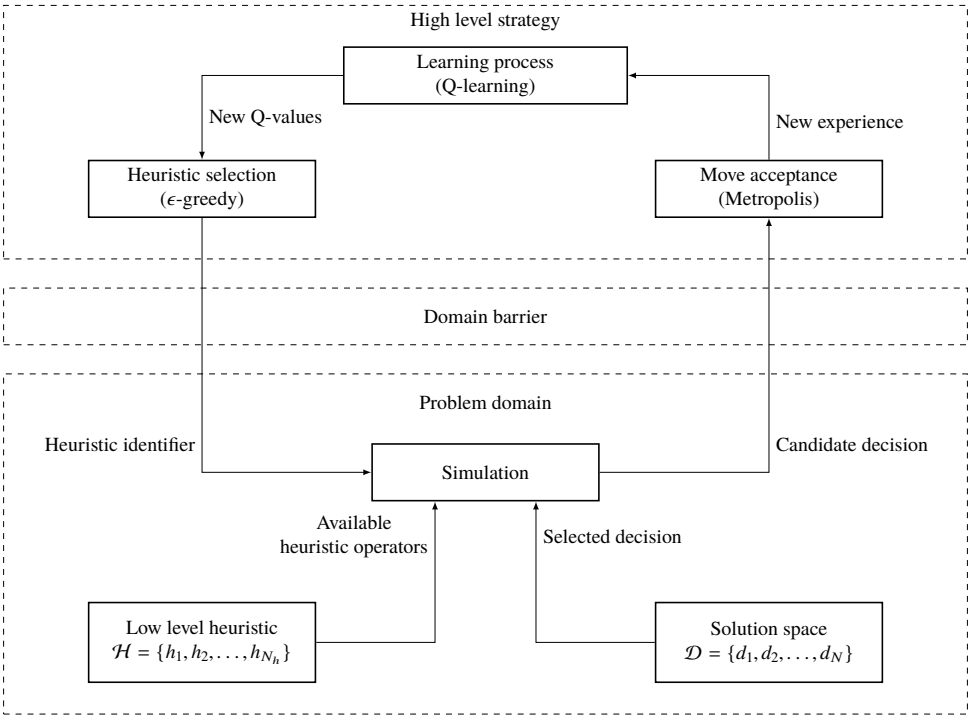
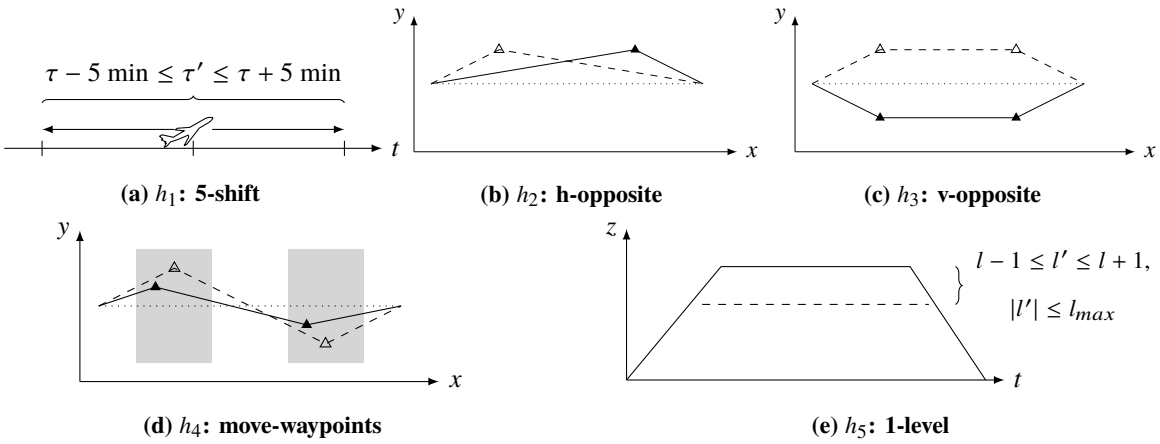


Fig. 12 Framework of Hyper-heuristic based on Q-learning with the high-level strategy and the problem domain.

1. Intensification operators

The five intensification operators are based on different neighborhood structures: *5-shift*, *h-opposite*, *v-opposite*, *move-waypoints*, and *1-level*. These neighborhood structures are used in a local improvement procedure that performs a sequence of moves or a random change based on the previous decision. Fig. 13 represents example actions of such heuristic operators.

1) *5-shift* (h_1): the local search which consists of randomly advancing/delaying departure time not more than 5 minutes
2) *h-opposite* (h_2): the local search, which horizontally flip current waypoints in the x-y plane
3) *v-opposite* (h_3): the local search, which vertically flip current waypoints in the x-y plane
4) *move-waypoints* (h_4): the local search, which randomly performs the route deviation by changing the position of each existing waypoint
5) *1-level* (h_5): the operator, which aims to randomly select an adjacent flight level under the constraint of maximum allowed flight level shifts



(a) h_1**: 5-shift** **(b)** h_2**: h-opposite** **(c)** h_3**: v-opposite**

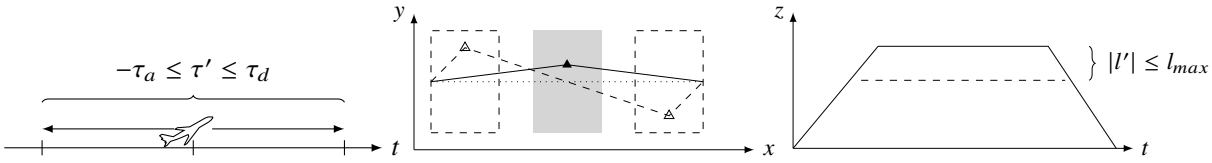
(d) h_4**: move-waypoints** **(e)** h_5**: 1-level**

Fig. 13 Representation of intensification heuristic operators which allow the algorithm to refine the search in the vicinity of the current decision.

2. Diversification operators

The set of diversification operators is composed of three generation procedures. These operators are used for the diversification process during optimization. Example actions of such heuristic operators are shown in Fig. 14.

1) *new-shift* (h_6): the generation operator which randomly advances or delays the departure time within maximum allowed departure time shift
2) *new-route* (h_7): the generation operator which randomly adds/removes one or more waypoints under problem constraints
3) *new-level* (h_8): the generation operator which aims to randomly shift flight level within maximum flight level shift (l_{max})



(a) h_6**: new-shift** **(b)** h_7**: new-route** **(c)** h_8**: new-level**

Fig. 14 Representation of diversification heuristic operators which allow the algorithm to perform a global search in order to escape from local minima.

G. Heuristic selection

The Q-learning agent performs the ϵ-greedy approach to select a heuristic operator based on a lookup table (Q-table) of values $Q(s, h)$. The Q-table follows the shape of $\langle$state, action$\rangle$, where the actions correspond to the intensification and diversification operators ($\mathcal{H}$).

The implementation of the ϵ-greedy approach for each time step is illustrated in Algorithm 1. At the initial level of

iterative updates, exploration is needed and at a later time of the iterative updates, more exploitation is required. The random action is selected using a probability ϵ, and the action is chosen from the Q-table using probability $1 - \epsilon$. At the beginning, the value of ϵ is set to a maximum value ϵ_{max} (user-defined) and reduces with time, and once the table converges, it reaches a minimum value ϵ_{min} (user-defined).

Algorithm 1 The ϵ-greedy heuristic selection at time step t

Input: $Q(s_t, h), \epsilon, \epsilon_{min}, \epsilon_{decay}$
p := random(0,1);
if $p < 1 - \epsilon$ **then**
    $h := \underset{h \in \mathcal{H}}{\text{argmax}}\ Q(s_t, h)$
else
    h := random_action($\mathcal{H}$)
end if
if $\epsilon > \epsilon_{min}$ **then**
    $\epsilon := \epsilon \cdot \epsilon_{decay}$
end if
return h, ϵ

The set of states allows us to perform the Diversification-Intensification cycle (D-I cycle). The cycle starts by selecting one of the diversification operators. Each state is determined based on the type of operator previously applied. Based on the application of five intensification operators in our problem, seven states are represented as follows:

1) s_0: a diversification operator has been previously applied.
2) s_1: intensification operator h_1 has been previously applied.
3) s_2: intensification operator h_2 has been previously applied.
4) s_3: intensification operator h_3 has been previously applied.
5) s_4: intensification operator h_4 has been previously applied.
6) s_5: intensification operator h_5 has been previously applied.
7) s_6: two successive intensification operators have been previously applied without modifying the current decision.

To support the above definition of the D-I cycle, Table 2 represents the Q-table at initialization. The Q-table has rows representing a set of states ($\mathcal{S}$) and columns representing a set of heuristic operators ($\mathcal{H}$), including intensification and diversification operators. Some Q-values are initially set to 1 to ensure that a specific heuristic operator can be selected at a specific state. Others are set to 0 to ensure the transition from one state to another.

H. Learning process

The learning process in HQL is performed during the optimization to improve the Q-learning agent's search strategy. This section describes how the Q-learning agent receives a reward to determine the Q-values in the Q-table.

At each iteration in HQL, the system stores an experience containing the current state s with the selected operator h and the gain g which is the cost difference between the new decision and the previous one. Until the end of the D-I

Table 2 Initialization of Q-values in the Q-table

State	Intensification operator					Diversification operator		
	h_1	h_2	h_3	h_4	h_5	h_6	h_7	h_8
s_0	1	1	1	1	1	— —	— —	— —
s_1	0	1	1	1	1	— —	— —	— —
s_2	1	0	1	1	1	— —	— —	— —
s_3	1	1	0	1	1	— —	— —	— —
s_4	1	1	1	0	1	— —	— —	— —
s_5	1	1	1	1	0	— —	— —	— —
s_6	— —	— —	— —	— —	— —	1	1	1

cycle, the Q-learning agent learns a set of experiences and then determines the rewards to update the Q-values in the Q-table. The Q-values at step t whose state-action pairs are found in such experiences, will be updated with the following formula:

$$Q(s_t, h_t) := (1 - \alpha)Q(s_t, h_t) + \alpha(r_t + \gamma \max_{h \in \mathcal{H}} Q(s_{t+1}, h)) \tag{13}$$

where $\alpha \in [0, 1]$ denotes the learning rate, $\gamma \in [0, 1]$ denotes the discount factor and r_t denotes the immediate reward signal whose value is defined under the following conditions:

$$r_t = \begin{cases} \sigma, & \text{if } g \neq 0, \\ 0, & \text{otherwise.} \end{cases} \tag{14}$$

I. Move acceptance

Move acceptance decides whether to accept or reject a new decision at each step during the search process. The iterations continue until a termination criterion is met.

In our framework, we apply the *metropolis-based criterion* to perform the move acceptance mechanism. In case of minimization, the newly produced decision would be accepted if it is better and it would also be accepted with a probability $\exp\left(\frac{f(d_i) - f(\tilde{d}_i)}{T}\right)$ which is worse, where $f(d_i)$ and $f(\tilde{d}_i)$ denote the cost value of the current decision and the newly generated decision, respectively, and T denotes the temperature. When T is high/low, the chance of accepting a deteriorated move is high/low.

Based on the metropolis mechanism, the HQL algorithm requires an initial temperature T_0 to launch the optimization process. The algorithm begins by initializing parameters, decision variables and the Q-table containing initialized Q-values. The system predetermines the initial temperature by performing the heating process with the initial acceptance

rate $\chi = 0.8$. Based on the initial temperature, we can calculate the final temperature and perform the optimization process, as detailed in Fig. 15. The algorithm determines the current state by performing ϵ-greedy selection. It then selects a heuristic operator to be applied with the selected decision d_i. The algorithm selects a decision whose cost value is greater than the threshold value, Ψ_T. The process then allows the move acceptance to accept or reject the candidate decision based on the metropolis criterion. If the iteration has reached the end of a D-I cycle, then the Q-learning agent determines rewards from the experiences that have occurred in such a cycle. If some experiences can improve their decisions, the Q-learning agent will update Q-values associated with state-operator pairs in the Q-table. The optimization process repeats until the final decreased temperature and then returns the final solution.

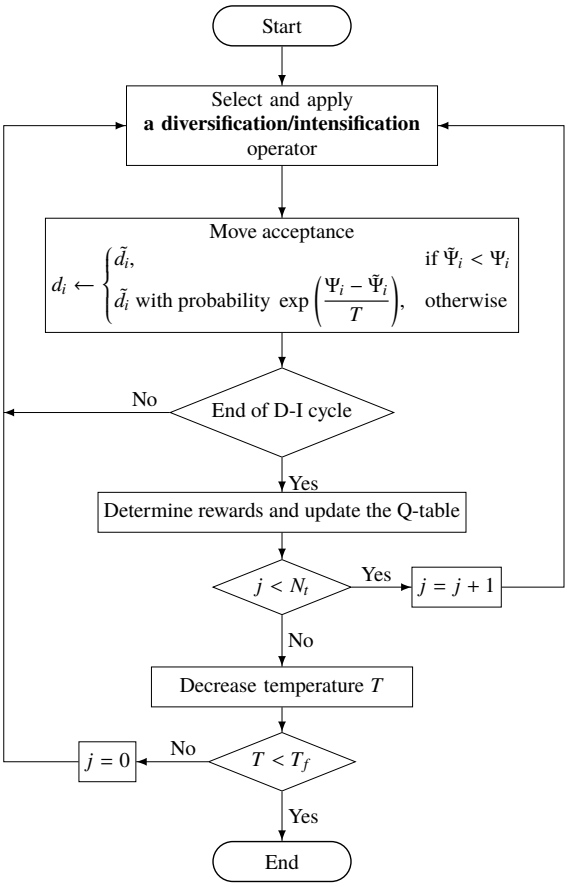


Fig. 15 The optimization process using the Q-learning based hyper-heuristic approach.

To summarize the overall process based on our context, the simulation process relies on the proposed HQL algorithm. The process selects a low-level heuristic, thanks to the Q-learning agent. An aircraft trajectory is modified according to such a low-level heuristic and its performance is recalculated (local complexity) under the simulation environment. The move acceptance allows acceptation or rejection of such a modification with metropolis-based criterion. Every experience obtained from the move acceptance module is determined as a reward of selection. The algorithm executes the Q-function with such a reward to compute a new Q-value in the Q-table. The Q-table guides the Q-learning agent to

select a better low-level heuristic in the next state. At the end of the process, it finalizes the optimal aircraft trajectories.

VI. Empirical results

In this section, we present an empirical evaluation of the proposed methodology. Our methodology is assessed on real air traffic data in the French airspace. We investigate the robustness of our algorithm with the same air traffic data by taking time uncertainty into account. All experiments have been conducted with Java-based software development on an Ubuntu system with Intel Xeon at 2.4 Ghz with 16 GB of memory. Each experiment was executed for 10 runs with different random seed values.

A. Real traffic in the French airspace

Air traffic data in this experiment represent a full day of en-route traffic in the French airspace. This consists of $N = 8836$ trajectories. Fig. 16 illustrates the initial given trajectories.

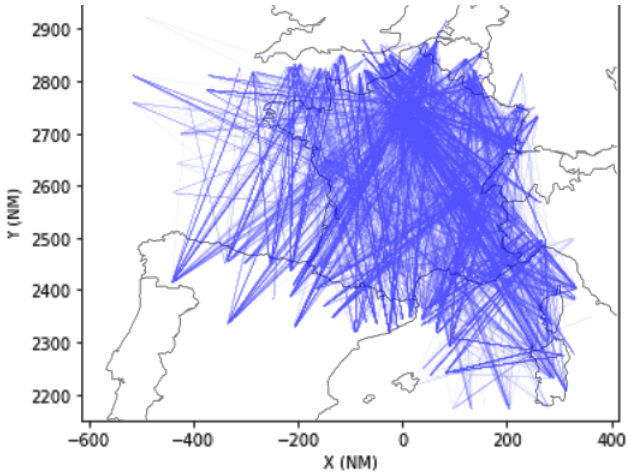


Fig. 16 Initial trajectories of a full day of traffic in the French airspace.

To give an idea of the complexity of the computation of the objective function of this problem instance, when using the sampling time-step value $t_s = 15$ seconds, the N trajectories are discretized into between 1,843,750 and 2,545,824 sample 4D points, according to the location of waypoints used to modify the shape of trajectories. With regard to the dimension of the search space, we note that our optimization problem involves the following features:

1) $2MN = 53{,}016$ (continuous) waypoint variables (w);
2) $N\left(\frac{|\tau_a| + |\tau_d|}{t_s}\right) + 1 = 2{,}120{,}641$ (discrete) departure-time shifts variables (τ);
3) $N(2 \cdot l_{\max} + 1) = 44{,}180$ (discrete) flight-level shifts variables (l);

Table 3a provides the parameter values that define the problem. The parameters used for our resolution algorithm have been conducted by several empirical experiments and are separately defined in Table 3b.

Based on real traffic data, aircraft trajectories can be categorized into possible six cases, as previously described

Table 3 User-defined parameters corresponding to (a) the problem formulation and (b) the HQL framework

Parameters	Value
Sampling time step, t_s	15 s
Minimum time step delay/advance departure time	15 s
Maximum changes of departure time, τ_a, τ_d	30 min
Maximum number of waypoints, M	3
Maximum flight level changes, l_{max}	2
Maximum route length extension coefficient, η_{max}	0.15 (15%)

(a)

Parameters	Value
Complexity threshold value, Ψ_T	$0.95 \cdot \Psi_{max}$ [a]
Geometric cooling rate, β	0.99
Final temperature, T_f	$10^{-9} \cdot T_0$
Start epsilon, ϵ_{max}	0.9
Epsilon decay, ϵ_{decay}	0.01
Stop epsilon, ϵ_{min}	0.01
Learning rate, α	0.9
Discount factor, γ	0.9
Reward value, σ	1.0

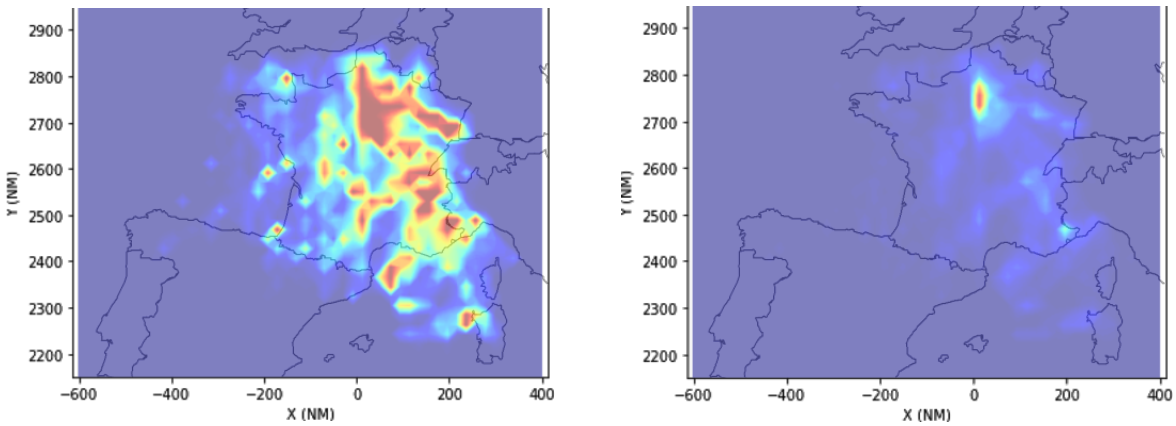
[a] The maximum complexity value

(b)

Table 4 Activation of heuristic operators against different trajectory modes.

Case no.	Intensification operator			Diversification operator		
	h_1	h_2	h_3	h_4	h_5	h_6
1	×	×	×	×	×	×
2	×	×	×	×	×	×
3	×	×	×	×	×	×
4	×	— —	— —	×	— —	— —
5	×	×	×	×	×	×
6	×	— —	— —	×	— —	— —

in Fig. 5. To produce optimal 4D trajectories, it is necessary to apply suitable heuristic operators to each case. Some trajectories are only modified with time-based heuristic operators whereas others can be modified with both time-based and space-based heuristic operators. Table 4 illustrates the activation of heuristic operators against different cases.



(a) (b)

Fig. 17 Complexity map of (a) initial trajectories and (b) final trajectories of a full day of traffic in the French airspace.

Initial trajectories represent an overall complexity of $\Psi = 230946.92$, and the complexity map for such trajectories is represented in Fig. 17a. The average number of neighboring aircraft obtained at a given time for each position is 1.37. After the optimization process, simulation results are reported in Table 5. The complexity map related to final trajectories is shown in Fig. 17b. Restructuring the traffic can improve the overall complexity by 92.8%, compared to initial trajectories. The comparison of aggregated traffic complexity over time between the initial and final trajectories is presented in Fig. 18. Throughout the day, the complexity of the optimized plan is lower than the initial plan during the same period of time.

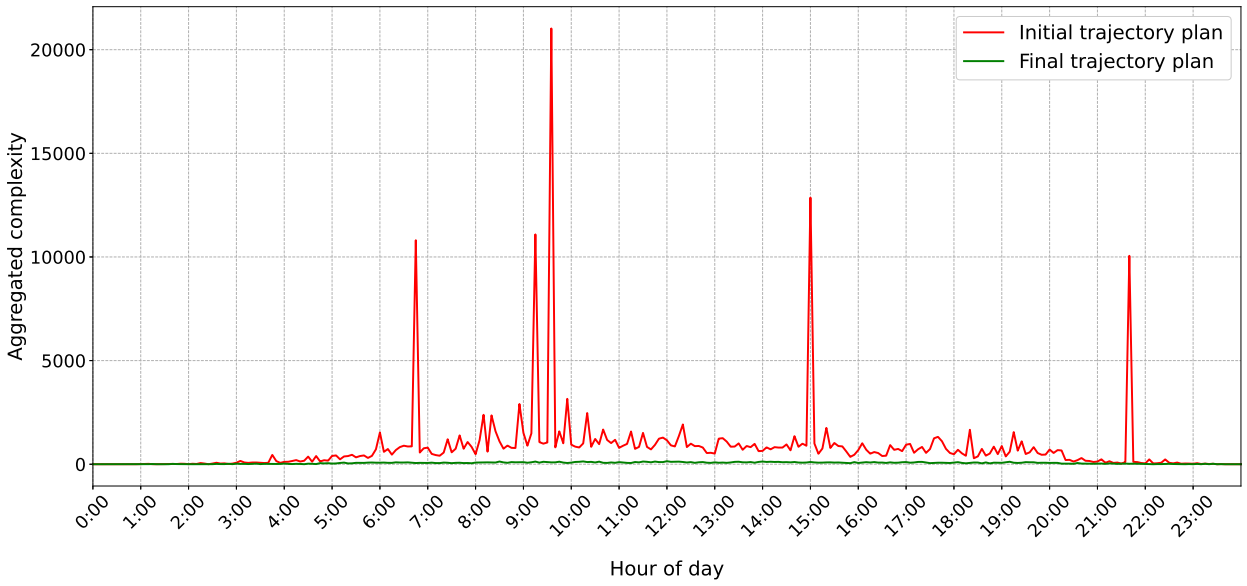


Fig. 18 Comparison of initial complexity and final complexity over time for a full day of traffic in the French airspace.

Table 5 Numerical results for restructuring a full day of traffic in the French airspace (10 runs for average computation).

Numerical result	Value
avg. number of iterations	215646
avg. computation time (minutes)	78.89
avg. final complexity	16719.33
avg. number of modified flight plans	50.67%
avg. ratio of modified departure time	43.58%
avg. ratio of modified routes	26.42%
avg. ratio of modified flight levels	17.94%
avg. departure time changes (minutes)	11.53
avg. route length extensions	0.909%
avg. flight level shifts	1.297

B. Comparison with other resolution algorithms

We evaluate the performance of the proposed algorithm with the following two baselines: Random Search and Simulated Annealing (SA), since the common thread behind these baselines and the purposed algorithm is the use of randomness. In particular, Random Search is one of the most straightforward search algorithms to implement, describe, understand, and help us ground the empirical results in this context. SA is a state-of-the-art algorithm commonly used to solve large-scale strategic 4D trajectory planning problems, as presented in [24, 28, 48, 50].

To establish such a comparison, the experiments were implemented using different resolution algorithms and tested with a full day of traffic in the French airspace. Each algorithm performed 10 independent runs and each run was terminated after 200,000 iterations. According to Table 6, the best, average, and standard deviation of the final complexity and the number of modified flight plans obtained by the proposed algorithm are better than those of other algorithms. Although the computation time consumed by the proposed method is about 23-29 min longer than those in baseline methods, it is acceptable since the algorithm works in the strategic planning context. The evolution of average complexity and its standard deviation against the number of iterations is given in Fig. 19. Impact of using different resolution algorithms on the number of modified departure times, on the number of modified routes, and on the number of modified flight levels is illustrated in Fig. 20.

Table 6 Performance comparison of the proposed algorithm with two baselines for restructuring a full day of traffic in the French airspace. The best values are highlighted in bold.

Algorithm	Traffic complexity (200,000 iterations)			Number of modified flight plans (%)			Computation time (min)
	Best	Average	Std.	Best	Average	Std.	
HQL (proposed)	**16091.10**	**17691.21**	**1310.55**	**46.88**	**49.73**	**1.130**	68.72
Simulated annealing [42]	20428.08	26833.68	5030.71	83.56	89.06	3.391	**40.58**
Random search	20764.33	27121.68	4941.30	76.22	80.24	2.397	45.54

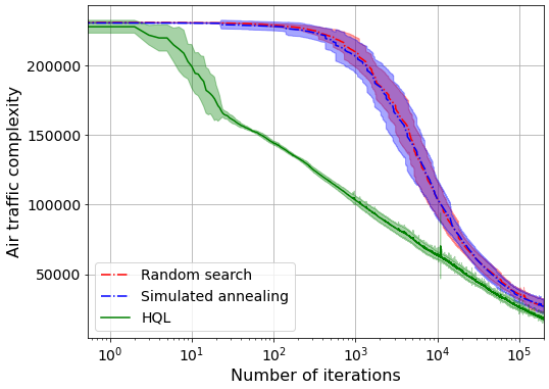


Fig. 19 The mean value of the air traffic complexity and its corresponding standard deviation of each iteration during the optimization process for the proposed algorithm and two baselines.

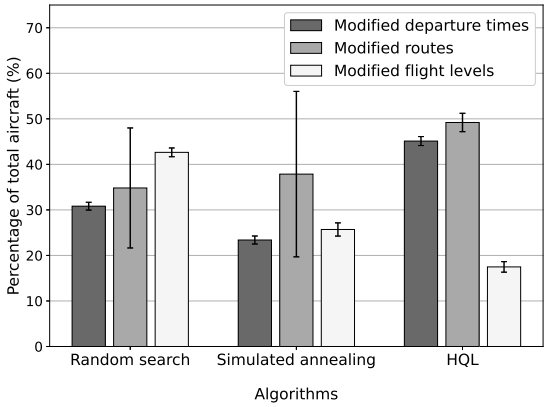


Fig. 20 The average number of modified departure times, routes, and flight levels, and their corresponding standard deviations after running the optimization process for the proposed algorithm and two baselines.

C. Time uncertainty consideration

In this subsection, we take into account time uncertainty to evaluate our approach. The user-defined parameters of our mathematical model and resolution algorithms are set as in the previous subsection. The simulation was performed 10 times for each time uncertainty of 1, 2, and 3 min with the same traffic as in the previous subsection. During the complexity computation, the average number of neighboring aircraft obtained at a given time for each position is 8.91, 15.78 and 22.36 for given time uncertainties of 1, 2, and 3 min, respectively. As we can see in Table 7, we find a significant reduction of complexity in aircraft trajectories for all three cases. The proposed approach can minimize air traffic complexity by 81.3%, 82.5%, and 80.5% for given time uncertainties of 1, 2, and 3 min, respectively. The required computation time is significantly longer when higher uncertainty is considered. The number of neighboring aircraft required for the local complexity computation affects the computation performance of the algorithm.

The average changes made to the initial flight plan are shown in Table 8. The proposed algorithm modifies 43.8%, 37.2%, and 38.0% of the initial flight plans for given time uncertainties of 1, 2, and 3 min, respectively. The number of modified flight plans for the three cases is detailed in Fig. 21.

Table 7 Numerical results for restructuring a full day of traffic considering time uncertainties of 1, 2, and 3 min.

t_ϵ (minutes)	initial Ψ	final Ψ	computation time (minutes)	no. of iterations
1	66690.18	12439.70	609.72	214587
2	109726.93	19148.48	1581.10	216772
3	111503.05	21693.83	2638.55	217141

Table 8 Average adjustments applied to initial flight plans with time uncertainties of 1, 2, and 3 min.

t_ϵ (minutes)	Average departure time changes (min)	Average route length extension (%)	Average flight level changes
1	10.890	0.840	1.276
2	11.140	0.791	1.252
3	11.402	0.782	1.254

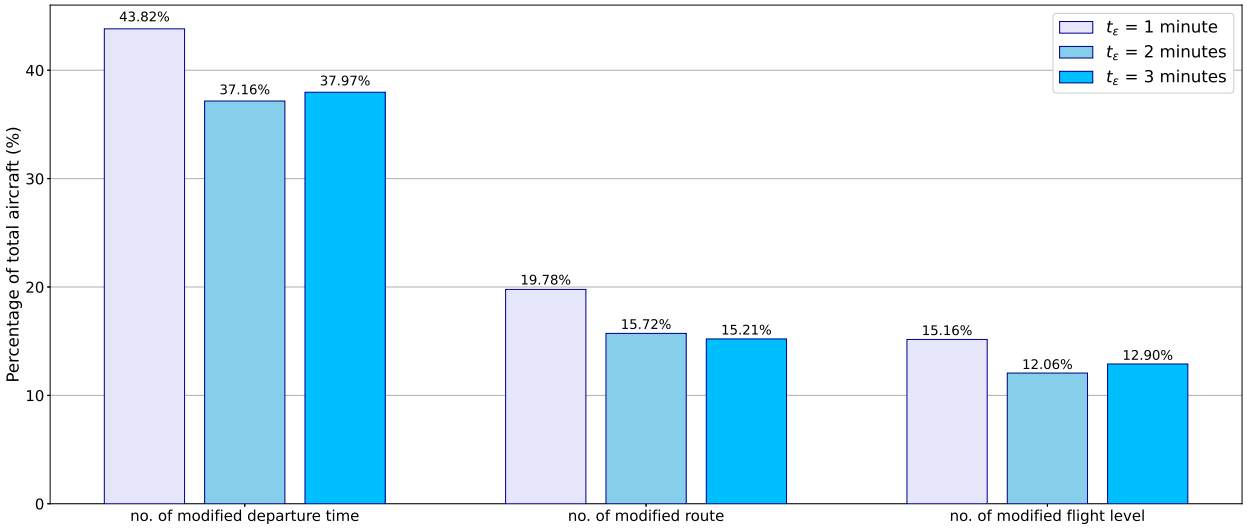


Fig. 21 Number of modified flight plans considering time uncertainty of 1, 2, and 3 min.

VII. Conclusions

In this paper, a traffic structuring approach to address a strategic planning problem in a TBO environment is introduced. Based on real flight plans, the proposed method is tested with 8836 aircraft trajectories in the French airspace. Our methodology aims to restructure air traffic patterns using departure time adjustment, trajectory rerouting, and flight level allocation. Mathematically, the problem is formulated as an optimization problem to reduce air traffic complexity.

We have developed a complexity metric based on a linear dynamical system to assess air traffic congestion. As illustrated from four identical traffic situations, this metric is able to quantify different levels of disorder. In particular, the metric is adapted to identify convergence traffic situations, which are a major issue of real-world traffic congestion today.

A hyper-heuristic based on reinforcement learning has been proposed to solve this problem. The algorithm uses several operators to perform the heuristic search, which is controlled by the adaptive selection process. This selection process is based on an ϵ-greedy selection strategy aimed at balancing exploitation and exploration. The Q-learning agent updates selection rules by updating the Q-table with the Q-function during the optimization process in order to drive the system to select the best low-level heuristic in the next state. After generating a candidate decision with a selected low-level heuristic, the system will accept this plan with a metropolis-based criterion.

Experimental studies on a full day of traffic in the French airspace suggest that our methodology can minimize complexity in aircraft trajectories within a computation time compatible with strategic planning. The significant reduction in the final average complexity ensures that the situation is easier to manage by air traffic controllers. Furthermore, with the same number of aircraft, the capacity can be enhanced by improving the traffic structure where one or more future aircraft are able to enter the airspace without congestion. The proposed algorithm are much more efficient in comparison with the random search and standard simulated annealing regarding the lower traffic complexity and the number of modified flight plans.

Finally, time uncertainties have been taken into account. Instead of considering a single observation as a fixed position and speed vector at a given time, the observation was modeled as its multiple predicted position and speed vectors over a time range. A significant reduction of traffic complexity shows that our complexity evaluation favors assessing the traffic structure under time uncertainty. The computation time represented in this experiment is also reasonable for trajectory planning at the strategic level.

Further research might explore the development of an extended model with other traffic structuring techniques, such as speed regulation. In addition, meteorological information would also help us to establish a more realistic uncertainty model. However, the issue of computation time is an intriguing one that could be usefully explored in further study. We intend to examine the existing learning process and heuristic selection more closely to reduce the extra computation time caused by the HQL framework. In addition, The exhaustive computation of air traffic complexity based on dynamical systems may be ineffective in terms of computation time for tactical applications. Using a graphics processing unit (GPU) may be an alternative approach for complexity computation in this framework. Last but not least, another future prospective study would compare the current method to others that work well with multi-level heuristics to investigate the high-level selection strategy.

Acknowledgments

This work has been supported by French Government Scholarship (BGF). The author gratefully acknowledges the support of Mr. Laurent Lapasset for providing computational resources for this research.

References

[1] 10, I. A. T. A. P. R. N., "Air Passenger Numbers to Recover in 2024," , Mar. 2022. https://www.iata.org/en/pressroom/2022-releases/2022-03-01-01 (retrieved 04 April 2022).

[2] *Manual on Collaborative Air Traffic Flow Management*, International Civil Aviation Organization (ICAO), 3rd ed., 2018.

[3] ENAIR, EUROCONTROL, and DFS, "STEP1 V3 Final Complexity Management OSED," Tech. Rep. P04.07.01-D68, Single European Sky ATM Research Joint Undertaking (SESARJU), Brussels, Belgium, Sep. 2016.

[4] Puntero, E., "D5.3 Final Project Results Report," Tech. Rep. D5.3, Single European Sky ATM Research Joint Undertaking (SESARJU), Brussels, Belgium, Dec. 2019.

[5] Prandini, M., Putta, V., and Hu, J., "Air traffic complexity in future Air Traffic Management systems," *Journal of Aerospace Operations*, , No. 3, 2012, pp. 281–299.
https://doi.org/10.3233/AOP-2012-0015.

[6] Hilburn, B., "Cognitive Complexity in Air Traffic Control - A Literature Review," Tech. Rep. 2004-007EEC Note 2004/04, EUROCONTROL, Jan. 2014.

[7] Histon, J. M., Hansman, R. J., Aigoin, G., Delahaye, D., and Puechmorel, S., "Introducing Structural Considerations into Complexity Metrics," *Air Traffic Control Quarterly*, Vol. 10, No. 2, 2002, pp. 115–130.
https://doi.org/10.2514/atcq.10.2.115.

[8] Delahaye, D., and Puechmorel, S., "New trends in air traffic complexity," *2009 ENRI International Workshop on ATM/CNS (EIWAC)*, Tokyo, Japan, 2009, pp. 55–60.

[9] Laudeman, I. V. S., "Dynamic Density: An Air Traffic Management Metric," Tech. Rep. NASA-TM-1998-112226, Apr. 1998.

[10] Sridhar, B., "Airspace Complexity and its Application in Air Traffic Management," *2nd USA/Europe Air Traffic Management R&D Seminar*, Orlando, FL, United States, 1998, pp. 1–6.

[11] Mondoloni, S., and Liang, D., "Airspace Fractal Dimensions and Applications," *4th USA/Europe Air Traffic Management R&D Seminar*, Federal Aviation Administration and EUROCONTROL, Santa Fe, NM, 2001, pp. 191–197.

[12] Lee, K., Feron, E., and Pritchett, A., "Air traffic complexity: An input-output approach," *2007 American Control Conference*, Inst. of Electrical and Electronics Engineers, New York, NY, 2007, pp. 474–479.
https://doi.org/10.1109/ACC.2007.4282989.

[13] Delahaye, D., Paimblanc, P., Puechmorel, S., Histon, J., and Hansman, R., "A new air traffic complexity metric based on dynamical system modelization," *IEEE/AIAA 21st Digital Avionics Systems Conference*, Vol. 1, Inst. of Electrical and Electronics Engineers, Irvine, CA, 2002, pp. 4A2–1–4A2–12.
https://doi.org/10.1109/dasc.2002.1067960.

[14] Bertsimas, D., and Odoni, A. R., "A Critical Survey of Optimization Models for Tactical and Strategic Aspects of Air Traffic Flow Management," , No. NASA/CR-97-206409, 1997.

[15] Vossen, T. W. M., Hoffman, R., and Mukherjee, A., "Air Traffic Flow Management," *Quantitative Problem Solving Methods in the Airline Industry*, International Series in Operations Research & Management Science, Springer, 2012, pp. 385–453.
https://doi.org/10.1007/978-1-4614-1608-1_7.

[16] Delahaye, D., Puechmorel, S., Tsiotras, P., and Féron, E., "Mathematical models for aircraft trajectory design: A survey," *Air Traffic Management and Systems*, Springer, 2014, pp. 205–247.
https://doi.org/10.1007/978-4-431-54475-3_12.

[17] Sridhar, B., Soni, T., Sheth, K., and Chatterji, G., "Aggregate Flow Model for Air-Traffic Management," *Journal of Guidance, Control, and Dynamics*, Vol. 29, No. 4, 2006, pp. 992–997.
https://doi.org/10.2514/1.10989.

[18] Lulli, G., and Odoni, A., "The European Air Traffic Flow Management Problem," *Transportation Science*, Vol. 41, No. 4, 2007, pp. 431–443.
https://doi.org/10.1287/trsc.1070.0214.

[19] Balakrishnan, H., "Control and optimization algorithms for air transportation systems," *Annual Reviews in Control*, Vol. 41, 2016, pp. 39–46.
https://doi.org/10.1016/j.arcontrol.2016.04.019.

[20] Cafieri, S., Brisset, P., and Durand, N., "A mixed-integer optimization model for Air Traffic Deconfliction," *Proceedings of the Toulouse Global Optimization workshop*, ENSEEIHT, Toulouse, France, 2010, pp. 27–30.

[21] Dougui, N., Delahaye, D., Puechmorel, S., and Mongeau, M., "A light-propagation model for aircraft trajectory planning," *Journal of Global Optimization*, Vol. 56, No. 3, 2012, pp. 873–895.
https://doi.org/10.1007/s10898-012-9896-1.

[22] Barnier, N., and Allignol, C., "4D-trajectory deconfliction through departure time adjustment," *8th USA/Europe Air Traffic Management Research and Development Seminar*, Federal Aviation Administration and EUROCONTROL, Napa, United States, 2009, pp. 542–551.

[23] Durand, N., Alliot, J.-M., and Noailles, J., "Automatic aircraft conflict resolution using genetic algorithms," *Proceedings of the 1996 ACM symposium on Applied Computing*, ACM Press, Philadelphia, PA, 1996, pp. 289–298.
https://doi.org/10.1145/331119.331195.

[24] Chaimatanan, S., Delahaye, D., and Mongeau, M., "A Hybrid Metaheuristic Optimization Algorithm for Strategic Planning of 4D Aircraft Trajectories at the Continental Scale," *IEEE Computational Intelligence Magazine*, Vol. 9, No. 4, 2014, pp. 46–61. https://doi.org/10.1109/MCI.2014.2350951.

[25] Chaimatanan, S., Delahaye, D., and Mongeau, M., "Hybrid metaheuristic for air traffic management with uncertainty," *Recent developments of metaheuristics*, Operations Research/Computer Science Interfaces Series, Vol. 62, 2018, pp. 219–251. https://doi.org/10.1007/978-3-319-58253-5.

[26] Oussedik, S., and Delahaye, D., "Reduction of air traffic congestion by genetic algorithms," *Lecture Notes in Computer Science*, Vol. 1498, Springer Berlin Heidelberg, 1998, pp. 855–864. https://doi.org/10.1007/BFb0056927.

[27] Breil, R., Delahaye, D., Lapasset, L., and Feron, E., "Multi-agent systems to help managing air traffic structure," *Journal of Aerospace Operations*, Vol. 5, No. 1-2, 2018, pp. 119–148. https://doi.org/10.3233/aop-170066.

[28] Juntama, P., Chaimatanan, S., Alam, S., and Delahaye, D., "A Distributed Metaheuristic Approach for Complexity Reduction in Air Traffic for Strategic 4D Trajectory Optimization," *2020 International Conference on Artificial Intelligence and Data Analytics for Air Transportation (AIDA-AT)*, Singapore, 2020, pp. 1–9. https://doi.org/10.1109/AIDA-AT48540.2020.9049200.

[29] Blum, C., Puchinger, J., Raidl, G. R., and Roli, A., "Hybrid metaheuristics in combinatorial optimization: A survey," *Application of Software Computing*, Vol. 11, No. 6, 2011, pp. 4135–4151. https://doi.org/10.1016/j.asoc.2011.02.032.

[30] Burke, E. K., Hyde, M. R., Kendall, G., Ochoa, G., Özcan, E., and Woodward, J. R., "A Classification of Hyper-Heuristic Approaches: Revisited," *Handbook of Metaheuristics*, Springer, 2019, pp. 453–477. https://doi.org/10.1007/978-3-319-91086-4_14.

[31] Burke, E. K., Hyde, M. R., Kendall, G., Ochoa, G., Ozcan, E., and Woodward, J. R., "Exploring hyper-heuristic methodologies with genetic programming," *Intelligent Systems Reference Library*, Intelligent systems reference library, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009, pp. 177–201. https://doi.org/10.1007/978-3-642-01799-5_6.

[32] Burke, E., Kendall, G., Silva, D. L., O'Brien, R., and Soubeiga, E., "An ant algorithm hyperheuristic for the project presentation scheduling problem," *2005 IEEE Congress on Evolutionary Computation*, Vol. 3, Inst. of Electrical and Electronics Engineers, Edinburgh, Scotland, UK, 2005, pp. 2263–2270. https://doi.org/10.1109/CEC.2005.1554976.

[33] Ferreira, A. S., Pozo, A., and Gonçalves, R. A., "An Ant Colony based Hyper-Heuristic Approach for the Set Covering Problem," *Advances in Distributed Computing and Artificial Intelligence Journal*, Vol. 4, No. 1, 2015, pp. 1–21. https://doi.org/10.14201/adcaij201541121.

[34] Burke, E., De Causmaecker, P., and Vanden Berghe, G., "A hybrid Tabu search algorithm for the nurse rostering problem," *Simulated Evolution and Learning*, Springer Berlin Heidelberg, Berlin, Heidelberg, 1999, pp. 187–194. https://doi.org/10.1007/3-540-48873-1_25.

[35] Koulinas, G., Kotsikas, L., and Anagnostopoulos, K., "A particle swarm optimization based hyper-heuristic algorithm for the classic resource constrained project scheduling problem," *Journal of Information Sciences*, Vol. 277, 2014, pp. 680–693. https://doi.org/10.1016/j.ins.2014.02.155.

[36] Castro, O. R., Jr, Fritsche, G. M., and Pozo, A., "Evaluating selection methods on hyper-heuristic multi-objective particle swarm optimization," *Journal of Heuristics*, Vol. 24, No. 4, 2018, pp. 581–616. https://doi.org/10.1007/s10732-018-9369-x.

[37] Sabar, N. R., and Kendall, G., "Population based Monte Carlo tree search hyper-heuristic for combinatorial optimization problems," *Journal of Information Sciences*, Vol. 314, 2015, pp. 225–239. https://doi.org/10.1016/j.ins.2014.10.045.

[38] Burke, R., "A Case-Based Reasoning Approach to Collaborative Filtering," *Advances in Case-Based Reasoning*, Springer-Verlag, Berlin, Heidelberg, 2000, pp. 370–379. https://doi.org/10.1007/3-540-44527-7_32.

[39] Nareyek, A., *Choosing Search Heuristics by Non-Stationary Reinforcement Learning*, Applied Optimization, Springer, Boston, MA, 2003, pp. 523–544. https://doi.org/10.1007/978-1-4757-4137-7_25.

[40] Burke, E. K., Kendall, G., and Soubeiga, E., "A Tabu-Search Hyperheuristic for Timetabling and Rostering," *Journal of Heuristics*, Vol. 9, 2003, pp. 451–470. https://doi.org/10.1023/B:HEUR.0000012446.94732.b6.

[41] Meignan, D., Koukam, A., and Créput, J.-C., "Coalition-based metaheuristic: a self-adaptive metaheuristic using reinforcement learning and mimetism," *Journal of Heuristics*, 2009. https://doi.org/10.1007/s10732-009-9121-7.

[42] Pylyavskyy, Y., Kheiri, A., and Ahmed, L., "A reinforcement learning hyper-heuristic for the optimisation of flight connections," *2020 IEEE Congress on Evolutionary Computation*, Inst. of Electrical and Electronics Engineers, 2020. https://doi.org/10.1109/CEC48606.2020.9185803.

[43] Sutton, R. S., and Barto, A. G., *Reinforcement learning: an introduction*, 2nd ed., Adaptive computation and machine learning series, The MIT Press, Cambridge, MA, 2018.

[44] Choong, S. S., Wong, L.-P., and Lim, C. P., "Automatic design of hyper-heuristic based on reinforcement learning," *Journal of Information Sciences*, Vol. 436-437, 2018, pp. 89–107.
https://doi.org/10.1016/j.ins.2018.01.005.

[45] Mosadegh, H., Fatemi Ghomi, S. M. T., and Süer, G. A., "Stochastic mixed-model assembly line sequencing problem: Mathematical modeling and Q-learning based simulated annealing hyper-heuristics," *European Journal of Operational Research Ranking*, 2019.
https://doi.org/10.1016/j.ejor.2019.09.021.

[46] Duflo, G., Danoy, G., Talbi, E.-G., and Bouvry, P., "A Q-learning based hyper-heuristic for generating efficient UAV swarming behaviours," *Intelligent Information and Database Systems*, Springer International Publishing, Cham, Switzerland, 2021, pp. 768–781.
https://doi.org/10.1007/978-3-030-73280-6_61.

[47] Juntama, P., Chaimatanan, S., Alam, S., and Delahaye, D., "Air Traffic Structuration based on Linear Dynamical Systems," *10th SESAR Innovation Days*, 2020.

[48] Delahaye, D., Chaimatanan, S., and Mongeau, M., "Simulated Annealing: From Basics to Applications," *Handbook of Metaheuristics*, Springer International Publishing, 2018, pp. 1–35.
https://doi.org/10.1007/978-3-319-91086-4_1.

[49] Watkins, C. J. C. H., and Dayan, P., "Q-learning," *Machine Learning*, Vol. 8, No. 3-4, 1992, pp. 279–292.
https://doi.org/10.1007/BF00992698.

[50] Hernández-Romero, E., Valenzuela, A., Rivas, D., and Delahaye, D., "Metaheuristic Approach to Probabilistic Aircraft Conflict Detection and Resolution Considering Ensemble Prediction Systems," *9th SESAR Innovation Days*, Athenes, Greece, 2019.