



HAL
open science

Selective Simulated Annealing for Large Scale Airspace Congestion Mitigation

Daniel Delahaye, Julien Lavandier, Arianit Islami, Supatcha Chaimatanan,
Amir Abecassis

► **To cite this version:**

Daniel Delahaye, Julien Lavandier, Arianit Islami, Supatcha Chaimatanan, Amir Abecassis. Selective Simulated Annealing for Large Scale Airspace Congestion Mitigation. Aerospace, 2021. hal-03359666

HAL Id: hal-03359666

<https://enac.hal.science/hal-03359666>

Submitted on 30 Sep 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Selective Simulated Annealing for Large Scale Airspace Congestion Mitigation

Julien Lavandier^{1,†,‡,*} , Arianit Islami^{1,‡}, Daniel Delahaye^{1,‡}, Supatcha Chaimatanan^{1,‡} and Amir Abecassis²

¹ ENAC; firstname.lastname@enac.fr

² GISDAS;supatcha@gistda.or.th

* Correspondence: delahaye@recherche.enac.fr

† Current address: Affiliation 3

‡ These authors contributed equally to this work.

Abstract: This paper presents a methodology to minimize the airspace congestion of aircraft trajectories based on slot allocation techniques. The traffic assignment problem is modeled as a combinatorial optimization problem for which a selective simulated annealing has been developed. Based on the congestion encountered by each aircraft in the airspace, this metaheuristic selects and changes the time of departure of the most critical flights in order to target the most relevant aircraft. The main objective of this approach is to minimize the aircraft speed vector disorder. The proposed algorithm was implemented and tested on simulated trajectories generated with real flight plans on a day of traffic over France airspace with 8800 flights.

Keywords: Optimization; Trajectory; Large Scale; Metaheuristic; Airspace congestion

1. Introduction

Air traffic management (ATM) is a system that supports and guides aircraft from a departure airport to a destination airport in order to ensure its safety while minimizing delays and airspace congestion. It manages the air traffic through the management of the three following complementary systems: airspace management (ASM), air traffic flow management (ATFM), and air traffic control (ATC). The ATC then controls the air traffic in real-time. It uses the flight plan information to predict the traffic situation, then issues necessary changes to the flight plan in order to ensure aircraft separation, and to maintain the order of air traffic flow, while satisfying as much as possible the pilot's request. For this purpose, the airspace is partitioned into different sectors, each sector is assigned to a group of controllers monitoring the air traffic. In order to prevent overloaded controller the controller from being overloaded, the number of aircraft allowed to enter a given sector at any given time is limited. When the number of aircraft reaches this limit, the corresponding sector is said to be congested. Generally, congestion in air traffic management air transportation can be categorized into two groups according to the part of airspace it involves. Terminal congestion is the congestion that occurs around the terminal control area¹ (TCA, or TMA outside the U.S. and Canada). En-route congestion is the congestion involved in the en-route section of the flight between TMAs. In the U.S., the congestion occurs more often in the terminal areas, whereas the en-route congestion is more critical in Europe due to the fragmented nature of its airspace where there are extra difficulties for coordinating the air traffic over across the boundaries, in particular between two different countries. Air traffic regulations impose that aircraft must always be separated by some prescribed distance, noted N_v for the vertical separation and N_h for the horizontal separation. Current ATC

Citation: Title. *Journal Not Specified* 2021, 1, 0. <https://doi.org/>

Received:

Accepted:

Published:

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Submitted to *Journal Not Specified* for possible open access publication under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

¹ A terminal control area (also known as a terminal maneuvering area) is controlled airspace surrounding major airports, generally designed as a cylindrical or upside-down wedding cake shape airspace of 30 to 50 miles radius and high of 10,000 feet.

34 regulations require aircraft operating in the terminal maneuvering area (TMA) to be
35 vertically separated by at least $N_v = 1,000$ feet and horizontally separated by a minimum
36 of $N_h = 3$ nautical miles. In the en-route environment, for aircraft operating up to (and
37 including) FL410², the horizontal minimum separation is increased to 5 nautical miles.
38 Then, And for aircraft operating above FL410, the vertical separation is increased to 2,000
39 feet.

40 As the air traffic demand keeps on increasing, the airspace becomes more and more
41 congested. Over past decades, several methods have been proposed to address the
42 air traffic management problem aiming at balancing air traffic demand and airspace
43 capacity and preventing airspace congestion. There are two frequently-used air traffic
44 decongestion strategies frequently used. The first one adapts the airspace capacity to
45 the increased demand. The second air traffic decongestion strategy is to regulate the
46 air traffic demand to the current capacity. This strategy focuses on decongesting the
47 ATM system through several approaches, such as: allocating delays to each aircraft in
48 order to reduce congestion in sectors or at destination airports, re-routing flights, or
49 changing flight levels in order to avoid congestion in sectors or airport TMAs, etc. More
50 precisely, the strategic trajectory planning problem under consideration can be presented
51 as follows:

- 52 • We are given a set of flight plans for a given day associated with a nationwide scale
53 or continent-scale air traffic.
- 54 • For each flight, f , we suppose that a set of possible departure times is given.

55 Based on those “alternate trajectories”, we propose to develop an optimization
56 strategy in order to minimize the associated airspace congestion with a minimum
57 deviation from the user preferences. To reach this goal AI decision support tool based
58 on a metaheuristics algorithm has been proposed.

59 Currently, congestion (complexity) of the traffic is measured only as an operational
60 capacity: the maximum number of aircraft that ATC controllers are able to manage
61 are willing to accept is defined fixed on a per sector basis and complexity is assessed
62 by comparing the real number of aircraft with the sector capacity. It must be noted
63 that under some circumstances controllers will accept aircraft beyond the capacity
64 threshold while rejecting traffic at other times although the number of aircraft is well
65 below the maximum capacity. This simple fact clearly shows that capacity as a raw
66 complexity metric is not enough to represent by itself to fully account for the controller’s
67 workload. In order to better quantify the complexity, geometric features of the traffic
68 have to be included. As previously stated, depending on the traffic structure, ATC
69 controllers will perceive situations differently, even if the number of aircraft present in
70 the sector is the same. Furthermore, exogenous parameters like the workload history
71 can be influential on the perceived complexity at a given time (a long period of heavy
72 load will tend to reduce the efficiency of a controller). Some reviews of complexity in
73 ATC have been completed, mainly from the controller’s workload point of view [1,2],
74 and have recognized that complexity is related to both the structure of the traffic and the
75 geometry of the airspace. This tends to prove that controller’s workload has two facets:

- 76 • An intrinsic complexity related to traffic structure.
- 77 • A human factor aspect related to the controller itself.

78 While most complexity metrics tend to capture those effects within a single aggregate
79 indicator, the purpose of this work is to design a measure of intrinsic complexity only
80 since it is the most relevant metric for a highly automated ATC system (no human
81 factors).

82 Section 2 of this paper will present the previous related works associated with
83 this large-scale trajectory planning problem. Section 3 will develop the associated
84 mathematical model in order to identify the decision variable, the objective function,

² Flight level (FL) is a pressure altitude, expressed in hundreds of feet, e.g. an altitude of 32,000 feet is referred to as FL320.

85 and the associated constraint. Section 4 will present the resolution algorithm which has
86 been developed to tackle such a problem. Section 5 will introduce the test cases ~~which~~
87 ~~has been~~ used for validating our approach. Finally, section 6 will conclude the paper.

88 2. State of the Art

89 This section describes, at first, the previous work related to large-scale airspace
90 congestion mitigation. Secondly, the complexity metric is used to evaluate the controllers'
91 workload to manage the aircraft in a given airspace.

92 2.1. Previous related work

93 In this section, the existing strategies in the literature are presented to address the
94 large-scale airspace congestion mitigation. The first category is the trajectory deconflic-
95 tion methods and the second one is the air traffic decongestion methods.

96 2.1.1. Trajectory deconfliction strategies

97 Instead of ~~only considering~~~~considering only~~ the capacity constraints, several re-
98 searches looked at deconfliction of aircraft trajectories. Different conflict detection and
99 resolution are described in the literature. Conflict detection methods are categorized
100 into three categories: nominal, worst-case, and probabilistic conflict detections. Nominal
101 conflict means no error is considered in aircraft's trajectories. Worst-case is the largest
102 envelope in which the aircraft might be. Probabilistic conflict detection is an improve-
103 ment of worst-case, by introducing a probability density function for aircraft's position
104 inside the worst-case envelope.

105 Conflict resolution strategies are: 1-against-N, pair-wise, global. The first strategy
106 ~~adds add~~ aircraft to the airspace following a priority order, and solving conflicts with all
107 aircraft in the airspace. The pair-wise strategy ~~considers consider~~ each pair of aircraft and
108 ~~solves solve~~ the conflict with each other. Finally, the global deconfliction strategy solves
109 all air traffic situations, whereas the previous strategies does not, but is computationally
110 demanding.

111 The main researches in the literature addressing trajectory deconfliction are pre-
112 sented in the following paragraph. Genetic algorithms, that deconflict aircraft trajectories,
113 are considered in [3]. However, for large-scale air traffic, the memory required is too high.
114 In [4,5], air traffic is deconflicted with ground holding and flight level alternates. The
115 conflicts are solved by allocating alternative flight levels, and then by ground holding
116 aircraft. However, for large-scale air traffic, some conflicts remains. Trajectory deconflic-
117 tion, with Light-Propagation Algorithm is described in [6,7]. The principle is to use the
118 light-propagation model, with conflicts areas equivalent to high refractive-index areas.
119 However, for large-scale air traffic, some conflicts are unsolved.

120 In the free-flight concept of operation, the strategies are based on Trajectory Based
121 Operations (TBOs). TBO is adapting the air traffic demand to the current air traffic
122 capacity, with Trajectory Actions (TAs). Those TAs are changing the departure time,
123 the flight level, or the route. To ensure the capacity is not exceeded, negotiated 4D
124 trajectories are provided to each aircraft by influencing its TAs. In [8], time uncertainties
125 has been also included in order to ~~build built~~ robust large scale trajectory ~~planning~~
126 ~~planing~~. When trajectory planning is done at pre-tactical level, conflicts between aircraft
127 are quite difficult to predict and congestion reduction objective is used instead of conflict
128 mitigation.

129 2.1.2. Air traffic decongestion strategies

130 In this section, the existing strategies in the literature are presented, to address air
131 traffic decongestion problem. Congestion is a situation where the number of aircraft in a
132 given airspace exceeds the maximum number of aircraft allowed to enter the airspace.
133 Several researches have been done to minimize the air traffic congestion. ~~Their~~~~ts~~ main
134 goal is to manage the air traffic demand in function of airspace's capacity. In this case,

135 action on aircraft are quite similar (flight level setting, delays, route assignment) but for
136 airspace congestion mitigation purpose.

137 Ground holding approaches is the simplest way to regulate air traffic demand
138 in order to meet the airspace's capacity. The method allocates a delay to the initially-
139 planned flight departure time. This strategy transfer air delays to ground delays at the
140 departure airport, because it is safer and less expensive. Ground holding strategy was
141 first studied in [9].

142 Many other extensions of this problem have been proposed in the literature ([10–
143 17]).

144 Air traffic flow management approaches consider departure and arrival time to
145 regulate the air traffic demand. These approaches rely on ~~branch-and-bound algo-~~
146 ~~rithms, mixed-integer programming solvers, genetic algorithms or other algorithms.~~
147 ~~on branch-and-bound algorithms, or mixed-integer programming solver, or genetic~~
148 ~~algorithm, or other algorithms.~~ Some other efforts have investigated airspace congestion
149 reduction by using distributed approaches ([18,19]).

150 All the previous methods use some artificial trick in order to circumvent the un-
151 derlying complexity (objective linearization, objective time-space objective separability,
152 distributed algorithm approximation).

153 The current approach addresses the full complexity of the airspace congestion
154 mitigation by using a dedicated metaheuristic which is able to strongly reduce the
155 overall congestion in the airspace.

156 In this paper, in a first approach the proposed method is only changing the aircraft
157 times of departure to reduce air traffic congestion. The congestion of the air traffic is
158 measured with the speed covariance metric, described in the next section.

159 3. Mathematical model

160 As for any real optimization problem ~~to be solved~~, the modeling step is critical
161 and has to be done carefully. It exhibits the state space (the definition of the decision
162 variables), the objective function, and the associated constraints. ~~The decision variables~~
163 ~~and the given data define the objective and the constraints. Both of which must be~~
164 ~~defined in terms of the decision variables and the given data.~~

165 3.1. Input data

- 166 • F : set of flights, noted f ,
- 167 • Γ : set of trajectories,
- 168 • $\gamma_f \in \Gamma$: trajectory corresponding to a flight $f \in F$,
- 169 • dt_f^+ : upper bound of departure time shift, $\forall f \in F$,
- 170 • dt_f^- : lower bound of departure time shift, $\forall f \in F$,

171 3.2. Decision variables

172 During the scheduling process, each flight may be scheduled at a different time of
173 departure. The decision variable dt_f indicates the difference between the scheduled and
174 requested departure times. All those decision variables are grouped into the state space
175 X .

176 3.3. Objective

177 In order to evaluate a solution, the following complexity metric will be used. This
178 metric is based on the aircraft speed vector ~~disorderdesorder~~. The main objective is to
179 reduce air traffic complexity.

180 In control airspace, the higher the number of aircraft, the more the control workload
181 increases. Hence, the controllers' level of mental effort needed to manage those aircraft
182 increases. A limit exists ~~in terms of maximum number of aircraft that can be managed~~
183 ~~by the controllers.when the controllers can manage only so many aircraft.~~ ThisThe
184 threshold is very difficult to estimate as it depends on the geometry of the air traffic, the

185 distribution of aircraft inside the airspace, etc. A simple measurement of the number
 186 of aircraft is easy to evaluate but not representative enough ~~to consider, because of~~
 187 disordered traffic. Disordered traffic is more demanding than ordered ~~traffic forte~~ the
 188 controller.

189 Thus, a simple count of aircraft in a neighborhood is not enough. Therefore, traffic
 190 complexity metrics are developed. Traffic complexity is an intrinsic measurement of the
 191 complexity associated with a traffic situation. This measurement is only dependent on
 192 the geometry of the trajectories.

193 This approach consists of evaluating the covariance of the speed vectors for each
 194 vector in a neighborhood ~~and. As well as,~~ evaluating the relative distance of each pair of
 195 points.

196 Each curvilinear trajectory is sampled in time into a 4D trajectory. Considering a
 197 4D point of a 4D trajectory, a spatial neighborhood is considered, as shown in Figure 1.

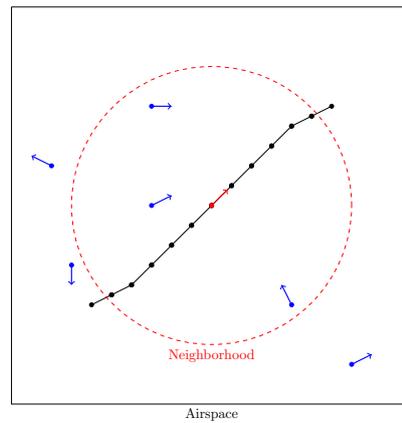


Figure 1. Spatial neighborhood of a 4D point in a curvilinear trajectory sampled in time.

198 Assuming there are N observations at a given time in a given neighborhood. Each
 199 observation is represented by a position measurement:

$$\vec{X}_i = \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} \quad (1)$$

and a speed measurement:

$$\vec{V}_i = \begin{bmatrix} vx_i \\ vy_i \\ vz_i \end{bmatrix} \quad (2)$$

200 The observations are shown on Figure 1 as the blue points and the speed vector
 201 associated, plus the reference point (red point) and its speed vector.

Therefore, the speed covariance is described in Equation (3):

$$Cov = \sum_{i=1}^N (|vx_i - \bar{vx}| + |vy_i - \bar{vy}| + |vz_i - \bar{vz}|) \quad (3)$$

with the mean values computed as follows:

$$\bar{vx} = \sum_{i=1}^N \frac{vx_i}{N} \quad \bar{vy} = \sum_{i=1}^N \frac{vy_i}{N} \quad \bar{vz} = \sum_{i=1}^N \frac{vz_i}{N} \quad (4)$$

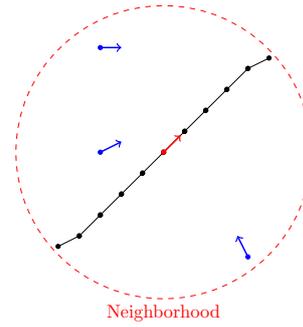


Figure 2. Spatial neighborhood of a 4D point.

The speed covariance does not differentiate the proximity of aircraft. Hence, the evaluation of pair-wise distance enables it. It is computed as follows:

$$Prox = \sum_{i=1}^N \sum_{j=i+1}^N \left| \mathbf{1}_{[-N_v, N_v]}(z_i - z_j) \cdot \mathbf{1}_{[0, 2N_h]}(d_{i,j}) \cdot (2N_h - d_{i,j}) \right| \quad (5)$$

202 with, $d_{i,j} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$, the distance in the horizontal plane between two
 203 points; $\mathbf{1}_X(x)$, the indicative function, that equals 1 if x is in the ensemble X and 0 else.

204 The further the points are from themselves, the lower the evaluation has to be.
 205 Moreover, if the points are separated enough ($d_{i,j} \geq 2N_h$), the evaluation has to be null,
 206 since it does not create any congestion. Hence, the relative distance to the horizontal
 207 norm separation is evaluated, as $\mathbf{1}_{[-N_v, N_v]}(z_i - z_j) \cdot \mathbf{1}_{[0, 2N_h]}(d_{i,j}) \cdot (2N_h - d_{i,j})$. The prox-
 208 imity evaluation is the sum of these relative distances for each pair of points in the
 209 neighborhood.

210 Finally, the metric evaluation is the sum of the speed covariance and the proximity.
 211 It is computed for a reference point p , and its neighborhood. Thus, it is noted c_p . Its
 212 complete formula is described in Equation (6):

$$c_p = \sum_{i=1}^N (|vx_i - \bar{v}\bar{x}| + |vy_i - \bar{v}\bar{y}| + |vz_i - \bar{v}\bar{z}|) + \sum_{i=1}^N \sum_{j=i+1}^N \left| \mathbf{1}_{[-N_v, N_v]}(z_i - z_j) \cdot \mathbf{1}_{[0, 2N_h]}(d_{i,j}) \cdot (2N_h - d_{i,j}) \right| \quad (6)$$

213 The complexity evaluation y of the air traffic solution is the sum of all flights'
 214 complexity, see Equation (7):

$$y = \sum_{f \in F} y_f \quad (7)$$

215 The flight, f , is represented by its curvilinear trajectory. The trajectory, γ_f , is
 216 sampled in time, with 4D points, p , represented in Figure 1. The congestion of the
 217 trajectory is the sum of complexity for each point in the curvilinear trajectory, see
 218 Equation (8):

$$y_f = \sum_{p \in \gamma_f} c_p \quad (8)$$

219 The point's congestion, noted c_p , is computed using the speed covariance metric,
 220 and the point's neighborhood. The formula of c_p computing is detailed in Equation (6).

Besides the congestion, [the algorithm must minimize the introduced delays](#)
[introduced delays must be minimized](#) to best suit the airlines' requests. The evaluation

of the total delays is the sum of all absolute gap between the requested and allocated time of departure, see Equation (9):

$$\sum_{f \in F} |dt_f| \quad (9)$$

The objective function $f(X)$, described hereafter:

$$f(X) = y + w_1 \sum_{f \in F} |dt_f| = \sum_{f \in F} \sum_{p \in \gamma_f} c_p + w_1 \sum_{f \in F} |dt_f| \quad (10)$$

with w_1 , the weight to balance the evaluations.

3.4. Constraints

The problem is subjected to some constraints. In fact, the time shift of the departure time needs to be between the minimal and maximal bound of the time displacement for each flight.

$$dt_f^- \leq dt_f \leq dt_f^+ \quad \forall f \in F \quad (11)$$

The evaluation of the objective function involves a high computation time. Moreover, the objective function may have multiple local optima. Therefore, the choice of a stochastic algorithm to optimize the air traffic congestion is more valued. Hence, the algorithm chosen is Simulated Annealing algorithm and is the choice of a Simulated Annealing algorithm presented hereafter in Section 4.

4. Simulated annealing

4.1. Standard Simulated Annealing

Simulated Annealing (SA) is one of the simplest and best-known metaheuristic methods for addressing the difficult *black box* global optimization problems (those whose objective function is not explicitly given and can only be evaluated via some costly computer simulation). Real-life applications massively use Simulated Annealing. It is massively used in real-life applications. The expression "simulated annealing" yields over one million hits when searching through the Google Scholar web search engine dedicated to the scholarly literature. In the early 1980s, three IBM researchers, Kirkpatrick, Gelatt, and Vecchi [20], introduced the concepts of annealing in combinatorial optimization. These concepts are based on a strong analogy with the physical annealing of materials. This process involves bringing a solid to a low energy state after raising its temperature. It can be summarized by the following two steps:

- Bring the solid to a very high temperature until "melting" of the structure;
- Cooling the solid according to a very particular temperature decreasing scheme in order to reach a solid-state of minimum energy.

In the liquid phase, the particles are distributed randomly. It is shown that the minimum energy state is reached provided that the initial temperature is sufficiently high and the cooling time is sufficiently long. If this is not the case, the solid will be found in a metastable state with non-minimal energy. This state; this is referred to as *hardening*, which consists of the sudden cooling of a solid.

In 1953, three American researchers (Metropolis, Rosenbluth, and Teller [21]) developed an algorithm to simulate physical annealing. They aimed to reproduce faithfully the evolution of the physical structure of a material undergoing annealing. This algorithm is based on Monte Carlo techniques, which generate which consist of generating a sequence of states of the solid in the following way.

Starting from an initial state i of energy E_i , a new state j of energy E_j is generated by modifying the position of one particle.

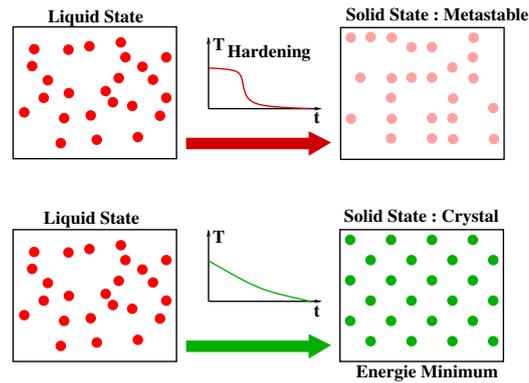


Figure 3. When the temperature is high, the material is in a liquid state (left). For a hardening process, the material reaches a solid-state with non-minimal energy (metastable state; top right). In this case, the structure of the atoms has no symmetry. During a slow annealing process, the material ~~also reachesreaches also~~ reaches a solid-state but for which atoms are organized with symmetry (crystal; bottom right).

256 If the energy difference, $E_i - E_j$, is positive (the new state features lower energy),
 257 the state j becomes the new current state. If the energy difference is less than or equal to
 258 zero, then the probability that the state j becomes the current state is given by:

$$Pr\{\text{Current state} = j\} = \exp\left(\frac{E_i - E_j}{k_b \cdot T}\right)$$

259 where T represents the temperature of the solid and k_B is the Boltzmann constant
 260 ($k_B = 1.38 \times 10^{-23}$ joule/Kelvin).

The acceptance criterion of the new state is called the *Metropolis criterion*. If the cooling is carried out sufficiently slowly, the solid reaches a state of equilibrium at each given temperature T . In the Metropolis algorithm, this equilibrium is achieved by generating a large number of transitions at each temperature. The thermal equilibrium is characterized by the *Boltzmann statistical distribution*. This distribution gives the probability that the solid is in the state i of energy E_i at the temperature T :

$$Pr\{X = i\} = \frac{1}{Z(T)} e^{-\left(\frac{E_i}{k_b T}\right)}$$

261 where X is a random variable associated with the current state of the solid, $Z(T)$ is the
 262 distribution function of X at temperature T . This allows the normalization:

$$Z(T) = \sum_{j \in S} e^{-\left(\frac{E_j}{k_b T}\right)}.$$

263 ~~TheIn the SA algorithm, the~~ Metropolis algorithm is applied to generate a sequence
 264 of solutions in the state space S ~~in the SA algorithm~~. To do this, an analogy is made
 265 between a multi-particle system and our optimization problem by using the following
 266 equivalences:

- 267 • The state-space points represent the possible states of the solid;
- 268 • The function to be minimized represents the energy of the solid.

269 A control parameter c , acting as a temperature, is then introduced. This parameter is
 270 homogeneous to the criterion that is optimized.

271 It is also assumed that the user provides for each point of the state space, a neigh-
 272 borhood, and a mechanism for generating a solution in this neighborhood. We then
 273 define the acceptance principle:

Definition 1. Let (S, f) be an instantiation of a combinatorial minimization problem, and i, j two points of the state space. The acceptance criterion for accepting solution j from the current solution i is given by the following probability:

$$\Pr\{\text{accept } j\} = \begin{cases} 1 & \text{if } f(j) < f(i) \\ \exp\left(\frac{f(i)-f(j)}{c}\right) & \text{else.} \end{cases}$$

274 By analogy, the principle of generation of a neighbor corresponds to the perturbation
275 mechanism of the Metropolis algorithm, and the principle of acceptance represents the
276 Metropolis criterion.

277 The principle of SA can be summarized as follows:

Simulated annealing

1. **Initialization** $i := i_{start}, k := 0, c_k = c_0, L_k := L_0$;
2. **Repeat**
3. **For** $l = 0$ **to** L_k **do**
 - **Generate a solution j from the neighborhood S_i of the current solution i ;**
 - **If $f(j) < f(i)$ then $i := j$ (j becomes the current solution);**
 - **Else, j becomes the current solution with probability $e^{\left(\frac{f(i)-f(j)}{c_k}\right)}$;**
4. $k := k + 1$;
5. **Compute** (L_k, c_k) ;
6. **Until** $c_k \simeq 0$

278 One of the main features of simulated annealing is its ability to accept transitions
279 that degrade the objective function.

280 4.2. Evaluation-based simulation

281 ~~The~~In many optimization applications, the objective function is evaluated in many
282 optimization applications thanks to a computer simulation process that requires a sim-
283 ulation environment. In such a case, the optimization algorithm controls the vector of
284 decision variables, X , which are used by the simulation process in order to compute the
285 performance (quality), y , of such decisions, as shown in Figure 4.

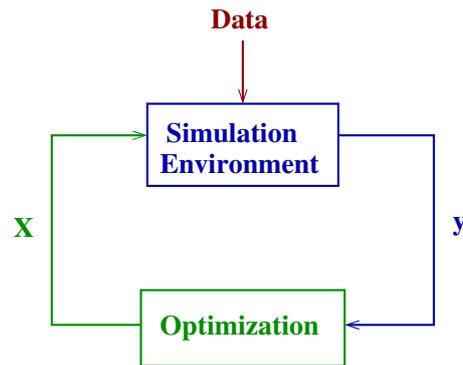


Figure 4. Objective-function evaluation based on a simulation process

286 In this situation, population-based algorithms may not be adapted to address such
287 problems, mainly when the simulation environment requires a **considerablehuge** amount
288 of memory space, **as it is as-is** often the case in nowadays real-life complex systems. **InAs**
289 **a-matter-of** fact, in the case of a population-based approach, the simulation environment
290 has to be duplicated for each individual of the population of solutions, which may
291 require an excessive amount of memory. In order to avoid this drawback, one may
292 think about having only one simulation environment **thatwhich** could be used each
293 time a point in the population has to be evaluated as follows. In order to evaluate one
294 population, one first considers the first individual. Then, the simulation environment is

295 initiated, and the simulation associated with the first individual is run. The associated
 296 performance is then transferred to the optimization algorithm. After that, the second
 297 individual is evaluated, but ~~the algorithm must first clear the simulation environment~~
 298 ~~simulation environment must be first cleared~~ from the events of the first simulation. The
 299 simulation is then run for the second individual, and ~~up to so on until~~ the last individual
 300 of the population is evaluated. In this case, the memory space is not an issue anymore
 301 .Still,,but the evaluation time may be excessive and the overall process too slow because
 302 ,due to the fact that the simulation environment is reset at each evaluation.

~~In the standard simulated annealing algorithm,~~ a copy of a state-space point is
 requested ~~in the standard simulated annealing algorithm~~ for each proposed transition.
~~In fact,~~ a point \vec{X}_j is generated from the current point \vec{X}_i through a copy in the memory
 of the computer. In the case of state spaces of large dimensions, the simple process
 of implementing such a copy may be inefficient and ~~drastically reduce the simulated~~
~~annealing performance may reduce drastically the performance of simulated annealing.~~
 In such a case, it is much more efficient to consider a *come back* operator, which cancels
 the effect of a generation. Let G be the generation operator which transforms a point
 from \vec{X}_i to \vec{X}_j :

$$G \\ \vec{X}_i \rightarrow \vec{X}_j$$

303 the comeback operator is the inverse, G^{-1} , of the generation operator.

304 Usually, such a generation modifies only one component of the current solution.
 305 In this case, the vector \vec{X}_i can be modified without being duplicated. According to the
 306 value obtained when evaluating this new point, two options may be considered:

- 307 1. the new solution is accepted and, in this case, only the current objective-function
 308 value is updated.
- 309 2. else, the comeback operator G^{-1} is applied to the current position in the state space
 310 in order to come back to the previous solution before the generation, again without
 311 any duplication in the memory.

312 This process is summarized in Figure 5.

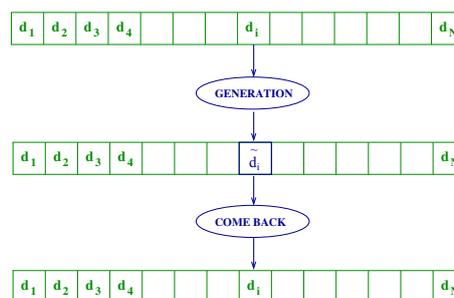


Figure 5. Optimization of the generation process. In this figure, the state space is built with a vector of decision for which the generation process consists ~~in~~ changing only one decision (d_i) in the current solution. If this generation, is not accepted, this component of the solution recovers its former value. The only information to be stored is the integer i and the real number d_i .

313 The *come back* operator has to be used carefully because it can easily generate
 314 undesired distortions in the way the algorithm searches the state-space. For example,
 315 ~~supposeif~~ some secondary evaluation variables are used and modified for computing
 316 the overall evaluation. ~~In that case,,~~ such variables must also recover their initial value,
 317 and the *come back* operator must therefore ensure the coherence of the state space.

318 4.3. Selective Simulated Annealing (SSA)

319 When a decision is put or removed from the simulation environment, one must
 320 compute the effect on the objective function y . Several situations may happen depending
 321 on the structure of the objective function. The easiest case is when it is possible to
 322 ~~efficiently compute~~ ~~compute easily~~ the impact of a single decision change on the objective
 323 function. The notion is closely related to the separability property of the objective
 324 function. If we consider that the current objective function is noted y_{old} associated to the
 325 current decision vector \vec{X}_{old} and suppose that a decision change is proposed for decision
 326 i (d_{new}^i) inducing a new state vector \vec{X}_{new} . One must determine if the impact on the
 327 objective function may be computed without evaluating all the ~~decisions in~~ ~~decision on~~
 328 the simulation environment. For some problems, such re-evaluation is limited to some
 329 decision variables. ~~It and it~~ is quite easy to compute the impact on the objective function
 330 by using a limited differential objective gap (Δ_y^i) without re-evaluating all the decisions.
 331 So, when a solution is removed from the simulation environment, one can compute
 332 ~~the impact of the objective function easily~~ ~~the impact of the objective function~~ by
 333 using the following equation :

$$y_{new} = y_{old} - \Delta_y^{i_{remove}} + \Delta_y^{i_{put}}$$

334 where y_{new} is the new objective function after inserting the selected decision in the
 335 simulation environment, $\Delta_y^{i_{remove}}$ is the impact on the objective function when the former
 336 decision d_{old}^i is removed from the simulation environment and $\Delta_y^{i_{put}}$ is the impact on the
 337 objective function when the new decision d_{new}^i is inserted in the simulation environment.

338 When such a differential evaluation of the objective function is not possible at the
 339 microscopic decision level, one must recompute all the decision variable evaluations in
 340 order to determine y_{new} . For some ~~cases,~~ ~~problems~~ such as re-evaluation may request
 341 quite a lot of computation. In order to avoid this issue, we propose an alternative
 342 approximation of the standard simulated annealing call "Selective Simulated Annealing".
 343 This approximation starts to evaluate all the decisions d_i and associates a cost to each of
 344 them y_i . For our problem, such evaluation will be given by ~~summing the summation~~
 345 ~~of~~ the congestion along the arc length of the associated trajectory $\gamma_i(t)$. We ~~then have~~
 346 ~~have then~~ a vector of decisions with their associated "costs" as shown in 6.

| | | | | | | | | | | |
|-------|-------|-------|-------|--|--|-------|--|--|--|-------|
| d_1 | d_2 | d_3 | d_4 | | | d_i | | | | d_N |
| y_1 | y_2 | y_3 | y_4 | | | y_i | | | | y_N |

Figure 6. Vector of evaluated decision.

347 The summation of individual costs gives the overall evaluation ~~The overall evaluation~~
 348 ~~is given by the summation of individual costs~~ :

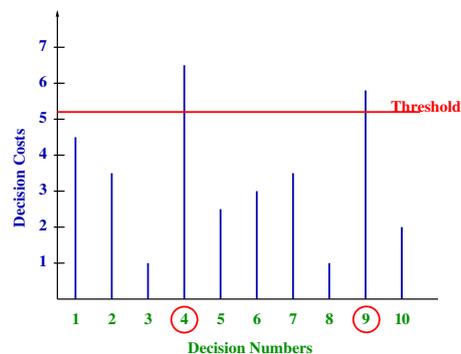
$$y = \sum_{i=1}^{i=N} y_i$$

349 The heating process consists of applying individual decision changes and individual
 350 cost evaluations in order to compute y_i^{old} and $y_i^{new} \forall i = 1..N$. If y_i^{new} is lower than y_i^{old}
 351 the microscopic transition is considered as accepted and if not, it could be accepted
 352 based on the Metropolis criterion. ~~The following equation can summarize this~~ ~~This can~~
 353 ~~be summarized by the following equation~~ :

$$Pr\{\text{accept } j\} = \begin{cases} 1 & \text{if } y_i^{new} < y_i^{old} \\ \exp\left(\frac{y_i^{new} - y_i^{old}}{c}\right) & \text{else.} \end{cases}$$

354 where c is the overall temperature. Such temperature is then increased until the accep-
 355 tance rate reaches $\simeq 80\%$.

356 For the cooling process, the algorithm first identifies the worst decision in terms
 357 of cost. Based on this “max” cost, a threshold is established in order to determine the
 358 decision that will undergo a neighborhood operator (see Figure 7).



359 **Figure 7.** In this example ten decisions are considered and their costs are illustrated
 360 by the vertical bars for which the highest cost is 6.5. The threshold is then given by $6.5 \times 0.8 = 5.2$.
 361 The decision with a cost higher than 5.2 are then selected to undergo the neighboring operator.

362 This process focuses mainly on decisions with worse costs. But as previously
 363 mentioned, decision changes may impact others’ decisions, which which in our case
 364 are not easy to identify (no explicit clear decision dependencies in the objective function). It
 365 means that a reduction of cost on a decision may increase the cost of induce an increase
 366 of cost on another decision. Still, but in our case, it is difficult to identify which decision
 367 will be impacted by the change of the former decision. In order to ensure coherence
 of the overall objective function, a complete full evaluation of the decision vector is
 regularly computed. As we will see in the result, this approximation really improves the
 computation performance without sacrificing the quality of the final solution.

368 4.4. Implementation SSA to our problem

369 4.4.1. Coding of the solution

370 The state-space coding used for our problem is quite simple and easy to manipulate.
 371 As illustrated in Figure 6, our state space is coded by the mean of a decision vector. Each
 372 dimension of such a vector represents a decision that can be applied to an aircraft, in our
 373 case, a time shift. Such a time shift is coded by an integer (positive or negative) which
 374 corresponds to the amount of time (in time slots) the aircraft is shifted when it enters the
 375 airspace. This time shift can be absorbed before take-off or onboard in some previous
 376 neighboring airspace. Each decision also contains contains also a field representing the
 377 aircraft trajectory’s associated performance that represents the associated performance
 378 of the aircraft trajectory in the airspace (y).

379 4.4.2. Neighboring Operator

380 The For a given transition, the decision that undergoes a neighboring operator for
 381 a given transition, is selected thanks to the cost threshold comparison. Suppose if
 382 the current decision has an individual cost higher than the computed cost threshold (80 %
 383 of the max cost). In that case, then it is changed by randomly modifying the time shift
 384 associated with such a decision considering the aircraft’s feasible time shift range (see
 385 Figure 8).

386 4.4.3. Objective Function Computation

387 In order to evaluate the objective function, we rely on a grid-based airspace def-
 388 inition which is implemented in a so-called hash table as presented in [22,23]. First,
 389 the airspace is discretized using a 4D grid (3D space + time), as illustrated in Figure
 390 9. The size of each cell in the x, y, z , and t direction is defined by the neighborhood
 391 area, which has to be checked (in space and time dimension around a given aircraft

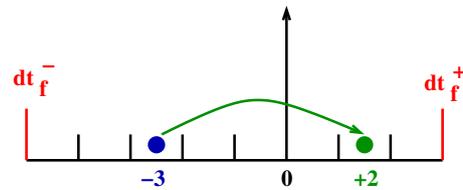


Figure 8. Time shift operator. The new time shift for the flight f is randomly selected in a time domain defined by two bound. A negative bound : dt_f^- and a positive bound : dt_f^+ . In this example, the former time shift was -3 (blue dot) and the new generated time shift is +2 (green dot).

392 i at a given time t). All trajectories are first inserted in such a 4D cube with, for each
 393 trajectory sample, its associated grid cell coordinates : (I_x, I_y, I_z, I_t) . To compute the
 394 complexity associated with a given trajectory sample for which we know the associated
 395 grid coordinate, the neighboring cells in all dimensions are checked in order to establish
 396 the list of neighboring aircraft around the considered aircraft. Based on their associated
 397 positions and speed vectors, one can compute the speed vector disorder metric associ-
 398 ated with the considered trajectory sample called c_k ~~representing which represents~~
 399 the complexity metric associated with the trajectory sample number k of the considered
 400 aircraft. The process is repeated for all the trajectory samples constituting the ~~considered~~
 401 ~~aircraft trajectory~~ ~~trajectory of the considered aircraft in order~~ to compute the complexity
 402 cost (y_i) of aircraft i . This computation is then iterated for all aircraft involved in the
 403 simulation.

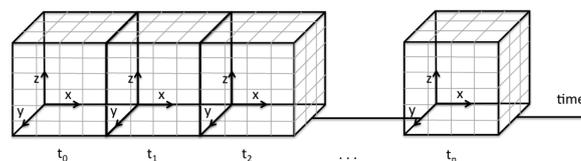


Figure 9. 4D-Grid coding of the airspace. 4D-Grid coding of the airspace. This structure strongly speeds up the neighborhood search for a given aircraft.

404 5. Results

405 5.1. ~~Benchmark data~~ ~~Data benchmark~~

406 The data set corresponds to air traffic over French airspace during a full day July,
 407 16, 2019). It consists of 8,800 flights ~~that which~~ have been simulated ³ based on ~~actual~~ ~~real~~
 408 flight plans over ~~French~~ ~~french~~ airspace. Figure 10 illustrates the initial given trajectories.
 409 The trajectories are represented by a curvilinear curve, sampled in time every 15 s.
 410 Therefore, a trajectory is a list of 4D points ~~positioned~~ ~~positionned~~ in space (latitude,
 411 longitude, altitude) and time step. ~~The~~ ~~For each point, the~~ velocity and heading are
 412 known ~~for each point~~ because it is needed to find the air traffic congestion. With the
 413 sampling time of 15 s, the total number of 4D points in the airspace is over 7,500,000.
 414 The congestion has to be computed for each point of the airspace. Thus, the objective
 415 function has a high computation time.

416 On Figure 10 the trajectories are colored ~~according to their~~ ~~in function of its~~ initial
 417 complexity (speed covariance metric described in the mathematical model section).
 418 Trajectories with the lowest complexity are shown in blue, whereas the highest are
 419 drawn in red, based on a logarithmic scale.

³ ENAC BADA ~~arithmetical~~ ~~metrie~~ simulator.

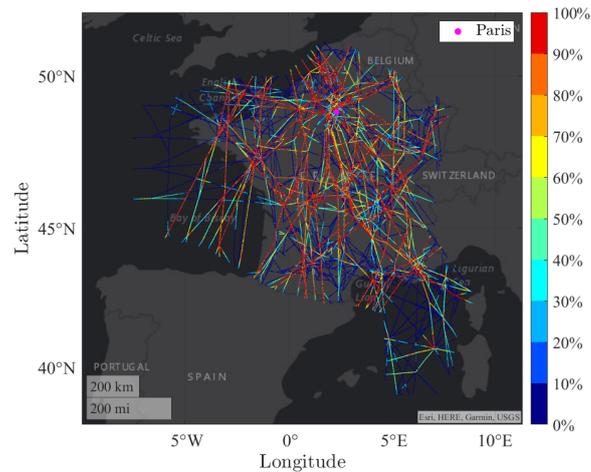


Figure 10. Full All-day air traffic over the French airspace, colored according to their initial complexity. The trajectories with the lowest complexity are shown in blue, whereas the highest are drawn in red.

420 5.2. Benchmark results

421 The proposed strategic 4D trajectory planning methodology is implemented in the
422 programming language Java on a computer with the following configuration:

- 423 • CPU: Intel Xeon Gold 6230 at 2.10 Ghz
- 424 • RAM: 1 TB

425 The algorithm is tested on the data explained in Section 5.1. As shown in Figure 10, the
426 complexity is high and has to be reduced with the proposed algorithm. The initial **worst**
427 **worse** congestion of the data set is 1,500,000.

428 After running the algorithm, for about two hours, the **worst flight of the data**
429 **set has a congestion value of worse congestion of the data set is** 120,000, see detailed
430 results in Table 1. Moreover, on Figure 11, there are fewer trajectories that are red and
431 more trajectories that are blue the trajectories are less blue and purple and more yellow
432 and green. This means the trajectories are less complex. Hence, the air traffic is less
433 congested.

| | Number of flights | Initial worst worse congestion | Final worst worse congestion | Computation time |
|---------------|-------------------|---------------------------------------|-------------------------------------|------------------|
| Time shifting | 8800 | 1500000 | 120000 | 7700 (2h) |

Table 1: Results of the algorithm.

434 On Figure 12 the complexity of each trajectory is represented in a bar chart. A
435 logarithmic scale groups the complexity of each trajectory to compare the benefits of
436 optimization easily. The complexity is computed after optimization using only time shift
437 of the departure time. **The number of trajectories with high complexity is reduced.**

438 The two hours computation which **hashave** been used for such complexity reduction
439 may be reduced for further experiments. **AfterAs a matter of fact, after** 45 minutes, the
440 objective function **doesdo** not evolve anymore, and we could consider that the algorithm
441 has reached the "optimum". We will address this point in **further researchsome further**
442 **researches in order** to adjust the right amount of computation for a given problem size.

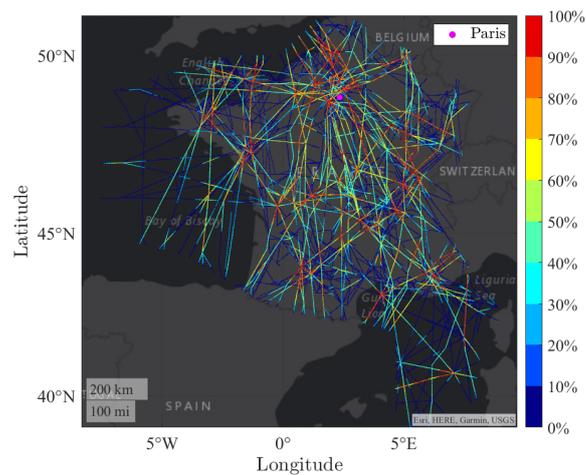


Figure 11. Full All-day air traffic over the French airspace, colored according to their function of its complexity, after optimization of the trajectories to minimize the congestion using time shift of departure time. The trajectories with the lowest complexity are shown in blue, whereas the highest are drawn in red.

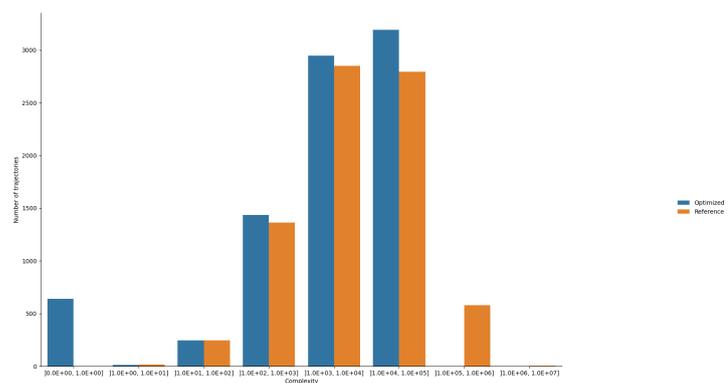


Figure 12. Comparison of the complexity of each trajectory before and after using only departure time shifting.

443 6. Conclusion

444 This paper introduced the work done on the large-scale trajectory planning. In
 445 the context of free-flight, the trajectory deconfliction algorithms have to be updated to
 446 enable large-scale air traffic. Controllers are increasing their workload with free-flight
 447 since aircraft do not always follow patterns. Thus, the airspace has a limited capacity
 448 that directly impacts the flight by changing its departure time. On the
 449 other hand, airlines wish to have efficient flights with few departure time changes due to
 450 the congested airspace. We have developed a decision support
 451 tool which can help the strategic planning of free-flights in given airspace to solve
 452 those issues.

453 After reviewing the concepts and previous works related to our subject, we based
 454 our study on a mathematical modeling of the problem followed by an optimization
 455 algorithm in order to reduce air traffic congestion. The selective simulated
 456 annealing algorithm for optimizing flight decisions appeared to be a good choice
 457 given its efficiency and adaptability properties.

458 A first trial of our solution on real traffic data over French airspace displayed a
 459 good congestion reduction and an acceptable time shift of flights' departure time.

References

1. Hilburn, B.; Gelatt, C.; Vecchi, M. Cognitive complexity and air traffic control: A literature review. Technical report, Eurocontrol, 2004.
2. Mogford, H.; Guttman, J.; Morrowand, P.; Kopardekar, P. The complexity construct in air traffic control : A review and synthesis of the literature. Technical report, FAA, 1995.
3. Durand, N.; Gotteland, J.B. Genetic Algorithms Applied to Air Traffic Management. In *Metaheuristics for Hard Optimization: Simulated Annealing, Tabu Search, Evolutionary and Genetic Algorithms, Ant Colonies, ... Methods and Case Studies*; Dréo, J.; Siarry, P.; Pétrowski, A.; Taillard, E., Eds.; Springer: Berlin, Heidelberg, 2006; pp. 277–306.
4. Barnier, N.; Allignol, C. 4D-trajectory deconfliction through departure time adjustment. p. 11.
5. Barnier, N.; Allignol, C. Combining flight level allocation with ground holding to optimize 4D-deconfliction. 2011, p. pp xxxx.
6. Dougui, N.; Delahaye, D.; Puechmorel, S.; Mongeau, M. A new method for generating optimal conflict free 4D trajectory. 2010, pp. pp 185–191.
7. Dougui, N.; Delahaye, D.; Puechmorel, S.; Mongeau, M. A light-propagation model for aircraft trajectory planning. *Journal of Global Optimization* **2013**, *56*, 873–895.
8. Islami, A.; Chaimatanan, S.; Delahaye, D. Large Scale 4D Trajectory Planning. In *Air Traffic Management and Systems – II*; Springer, 2016; Vol. 420, *Lecture Notes in Electrical Engineering*, pp. pp 27–47.
9. Odoni, A. The Flow Management Problem in Air Traffic Control. *Flow Control of Cogested Networks*; Odoni, A.e.a., Ed. NATO, 1987, Vol. F38, *ASI Series*, pp. 269–288.
10. Terrab, M.; Odoni, A. Strategic Flow Management for Air Traffic Control. *Operations Research* **1993**, *41*, 138–152.
11. Richetta, O.; Odoni, A. Dynamic Solution to the Ground-Holding Problem in Air Traffic Control. *Transportation Research* **1994**, *28A*, 167–185.
12. Andreatta, G.; Romanin-Jacur, G. Aircraft Flow Management under Congestion. *Transportation Science* **1987**, *21*.
13. Richetta, O.; Odoni, A. Solving Optimally the Static Ground Holding Policy Problem in Air Traffic Control. *Transportation Science* **1993**, *27*, 228–238.
14. Wang, H. A Dynamic Programming Framework for the Global Flow Control Problem in Air Traffic Management. *Transportation Science* **1991**, *25*, 308–313.
15. Andreatta, G.; Odoni, A.; Richetta, O. Models for the Ground Holding Problem. *Large Scale Computation and Information Processing in Air Traffic Control*; Bianco, L.; Odoni, A., Eds. Springer-Verlag, 1993, *Transportation Analysis*, pp. 125–168.
16. Bertsimas, D.; Stock, S. The Air Traffic Flow Management Problem with En-Route Capacities. Technical report, A.P Sloan School of Management. M.I.T, 1994.
17. Tasic, V.; Babic, O.; Cangalovic, M.; Hohlacov, D. Some Model Algorithms for En-Route Air Traffic Flow Management Faculty Transport. *Proceedings of the US Europe ATM Seminar. Eurocontrol-FAA, 1997.*
18. Alam, S.; Delahaye, D.; Chaimatanan, S.; Féron, E. A Distributed Air Traffic Flow Management Model for European Functional Airspace Blocks. *12th ATM R&D Seminar*; , 2017.
19. Juntama, P.; Chaimatanan, S.; Alam, S.; Delahaye, D. A Distributed Metaheuristic Approach for Complexity Reduction in Air Traffic for Strategic 4D Trajectory Optimization. *AIDA-AT 2020, 1st conference on Artificial Intelligence and Data Analytics in Air Transportation*; IEEE: Singapore, Singapore, 2020; *AIDA-AT 2020 International Conference on Artificial Intelligence and Data Analytics for Air Transportation*, pp. 1–9 / ISBN : 978–1–7281–5381–0.
20. Kirkpatrick, S.; Gelatt, C.; Vecchi, M. Optimization By Simulated Annealing. IBM Research Report RC 9355, Acts of PTRC Summer Annual Meeting, 1982.
21. Metropolis, N.; Rosenbluth, A.; Rosenbluth, M.; Teller, A.; Teller, E. Equation of state calculation by fast computing machines. *Journal of Chemical Physics* **1953**, *21*, 1087–1092.
22. Chaimatanan, S.; Delahaye, D.; Mongeau, M. Strategic deconfliction of aircraft trajectories. In *ISIATM 2013, the 2nd International Conference on Interdisciplinary Science for Innovative Air Traffic Management*, Toulouse (France), 2013.
23. Chaimatanan, S.; Delahaye, D.; Mongeau, M. A hybrid metaheuristic optimization algorithm for strategic planning of 4d aircraft trajectories at the continental scale. *Computational Intelligence Magazine* **2014**, *9*, 46–61.