Are All the Requested Air Traffic Flow Management Regulations Actually Indispensable?

Identifying effective regulations with adaptive tabu search and strategic oscillations

Ramon Dalmau, Leila Zerrouki, Camille Anouraud & Darren Smith
Network (NET) Research Unit
EUROCONTROL Innovation Hub (EIH)
Brétigny-Sur-Orge, France

Benjamin Cramet
Software Engineering Unit
EUROCONTROL
Brussels, Belgium

***Abstract*—The most common air traffic flow management measure used by the European Network Manager to resolve overloads (i.e., imbalances between demand and capacity) consists of imposing air traffic flow management regulations, which delay flights on ground using the *first-come, first-served* principle. During busy days, the number of regulations coordinated between the Network Manager operations centre and the flow management positions could be high. In this situation, flights may be subject to several regulations simultaneously. The interactions between regulations, which depend on the flights that have in common, are complex and extremely difficult to predict. This paper is founded on the hypothesis that, during busy days, some of the requested air traffic flow management regulations could be avoided without generating overloads elsewhere in the network. The problem of identifying whether a regulation is essential or not is addressed by using the adaptive tabu search algorithm and the strategic oscillations principle. The performance of the proposed algorithm is assessed by replaying one of the busiest days of 2019.**

***Keywords*—Air traffic flow management; tabu search**

NOMENCLATURE

$\alpha, \rho \in \Re_{\geq 0}$	Penalty weight multipliers
$\delta, \Delta \in \mathbb{Z}_{\geq 0}$	Time window slice and width
$\mathcal{C}$	Set of constraints to be satisfied
$\mathcal{N}(\boldsymbol{x})$	Neighbourhood of solution $\boldsymbol{x}$
$\mathcal{R}$	Set of regulations
$\mathcal{T}$	Tabu list (or set of prohibited solutions)
$\mathcal{W}$	Set of monitored time windows
$\mathcal{W}_i \subseteq \mathcal{W}$	Time windows within the regulation i
$\mathcal{X}$	Search space
$\tau_i^{[}, \tau_i^{)} \in \mathbb{Z}_{\geq 0}$	Start and end times of the regulation i
$\boldsymbol{w}, \boldsymbol{v} \in \Re_{\geq 0}^{\lVert\mathcal{C}\rVert}$	Original and adaptive penalty weights
$\boldsymbol{x} \in \mathcal{X}$	Solution vector
$d_{ij} : \mathcal{X} \rightarrow \mathbb{Z}_{\geq 0}$	Traffic load in s_i during the window j
$f_i : \mathcal{X} \rightarrow \mathbb{Z}_{\geq 0}$	ATFM delay created the regulation i
$p : \mathcal{X} \rightarrow \Re_{\geq 0}$	Penalty function
$p_l : \mathcal{X} \rightarrow \mathbb{Z}_{\geq 0}$	Violation (infeasibility) of constraint l
$q : \mathcal{X} \times \Re_{\geq 0}^{\lVert\mathcal{C}\rVert} \rightarrow \Re_{\geq 0}$	Evaluation function
$r_i \in \mathbb{Z}_{\geq 0}$	Entry rate of the regulation i
s_i	Traffic volume of regulation i
$t \in \mathbb{Z}_{\geq 0}$	Tabu tenure

I. INTRODUCTION

Before the COVID-19 pandemic paralysed the planet, the air traffic demand forecast for the coming years was optimistic. According to the most likely scenario of the EUROCONTROL statistics and forecast, there would be around 16.2M of flights in Europe in 2040 (53% more traffic than in 2017) [1]. Despite the fact that these demand figures are more uncertain than ever due to the unclear impact of the pandemic on the travel habits [2], the upward trend is expected to resume once the vaccination process advances and travel restrictions lighten [3]. With air traffic likely to increase, ensuring that demand does not exceed capacity will become extremely important.

Adapting capacity to demand is the first step in aligning demand and capacity. This can be accomplished by modifying the airspace sectorisation, for example. If demand still cannot be met, air traffic flow management (ATFM) measures are implemented to match demand with available capacity. In Europe, the most common ATFM measure consists of limiting the rate at which aircraft enter the congested traffic volume[1] during a given period of time, i.e., to activate a regulation.

The flights subject to a regulation are issued with a ground delay (i.e., the ATFM delay) that is assigned on a *first-come, first-served* basis (widely accepted as equitable), so that the maximum entry rate in the regulated traffic volume is not exceeded during the regulated period of time. The basic mechanisms that are used to assign and manage ground delays using ATFM regulations are summarised in Section II-A.

At present, the Network Manager (NM) is in charge of activating regulations in those traffic volumes where it is necessary, after coordination with the local flow management positions (FMPs) that requested them when detecting overloads. The number of requested regulations that NM has to process may be in the order of tens or hundreds (depending on the day), and activating or cancelling a single regulation could have a massive impact in the total delay, as well as in the demand figures of traffic volumes elsewhere in the network.

[1]A traffic volume is related to a single geographical *entity* (either an aerodrome, a set of aerodromes, an airspace sector or a point), and may consider all traffic passing through that entity or only specific flows.

Because of the complex dynamics emerging in the network in the presence of multiple regulations, predicting the magnitude and scope of this chain reaction is extremely difficult [4]. This work is founded on the hypothesis that, especially during busy days, only some of the requested regulations are actually indispensable, i.e., a subset of the regulations requested by FMPs could still solve all traffic overloads with less delay.

Finding the subset of regulations that solves all overloads could be approached with constraint programming algorithms. There may exist, however, several feasible solutions to this problem. The objective of this work is to develop a decision-support tool to propose the *best* of such feasible solutions (if any), i.e., the optimal set of regulations to be activated by NM. Instead of producing this optimal set from scratch (i.e., defining new regulations), however, the algorithm proposed herein takes the regulations requested by the FMPs as starting point, assuming that otherwise all of them would be activated, and tries to cancel those regulations that are not strictly necessary. On this premise, determining the optimal set of regulations to be activated by NM becomes a binary optimisation problem, where each regulation can be either active (1) or cancelled (0).

One could be tempted to use supervised learning techniques because they have shown outstanding results in many ATFM applications [5]–[7]. For the problem addressed in this paper, however, the algorithm should not reproduce the historical data, but to propose better solutions that those applied in past. Considering this issue, reinforcement learning techniques could be used to effectively train a policy from thousands of experiences [8], [9]. Collecting such amount of experiences, however, would require a very fast and realistic ATFM simulator, capable of accurately determining the effect of cancelling a regulation under a wide variety of network conditions.

This paper follows a more conservative approach and proposes a variant of the well-known tabu search algorithm [10], [11] that does not rely on historical data (which may be *polluted* with sub-optimal actions) nor on a finite set of experiences generated with a simulator (which, nonetheless, is a simplified representation of the real world). The tabu search implementation presented herein solves a weighted constraint satisfaction problem (WCSP), which weights are adaptively updated during the search for transitioning between feasible and infeasible space in a strategic oscillation fashion [12].

The performance of the algorithm is assessed by replaying one of the busiest days of 2019 with a real-time simulator fed with operational traffic data. According to preliminary results, the algorithm is capable of reducing total ATFM delay of the network with minimal impact on traffic load and overloads.

II. Background

In this paper, the optimisation of applied ATFM regulations is formulated as a WCSP, which is then solved by using a tabu search algorithm based on the general problem solver for combinatorial optimisation problems proposed in [13], which working principle has demonstrated to be effective in a wide variety of combinatorial optimisation problems [14]–[16].

Section II-A describes the concept of ATFM regulation, Section II-B formalises the WCSP, and Section II-C summarises the working principle of the solver proposed in [13].

A. Fundamentals of air traffic flow management regulations

As introduced in Section I, nowadays regulations are the most common ATFM measures in Europe. The activation of a regulation is the result of a coordination between the local FMP and the NM when the demand exceeds the capacity.

The essential parameters of a regulation are: the traffic volume that is regulated, the cause of the regulation (e.g., weather, airport capacity, etc.), the start and end times of the regulation, and the maximum entry rate (in aircraft per hour)[2].

For each active regulation, the computer-assisted slot allocation system (CASA) creates and manages a list of time *slots*. For example, the list of slots for a 2-hours duration regulation, with a maximum entry rate of 20 aircraft per hour, would be composed of 40 slots evenly spaced by 3 minutes. These slots are assigned to the regulated flights by using a *first-come, first-served* policy. In other words, CASA sequences the regulated flights in the order they would have overflown the restricted entity (e.g., an airspace) in the absence of ATFM measure [17].

For each regulated flight, the difference between the time of the assigned slot and its estimated time over (ETO) the restricted entity (e.g., airspace) is translated into a ground delay. Note that a flight crossing many restricted entities may be subject to more than one regulation simultaneously. In this case, the delay of the regulation that allocates the greatest delay (the most penalising regulation - MPR) takes precedence. That is, the slot in each one of the other regulations will be *forced* according to the delay assigned by the MPR.

Whenever the flight data is updated or the state of the network changes (e.g., a new regulation is applied, or an existing regulation is cancelled or modified), CASA tries to assign the available slot that is closest to the ETO, i.e, the slot that produces less ATFM delay. If the slot of the ETO is free, then the flight will not receive delay. It the slot is already pre-allocated to another flight, then the slot will be given to whichever flight planned to overfly the restricted entity first. This mechanism inevitably leads to a chain reaction, since the flight whose slot has been taken attempts to obtain another slot, likely by taking the slot of another flight, and so on [17].

Note that regulations interact due to the fact that they have flights in common. Even a minor change in the flights sequence of a regulation may create a domino effect and significantly impact the delay and throughput in other regulations. These interactions, however, are very difficult to model and predict, especially when the number of regulations is large.

It should be noted that some flight, such as flights departing from airports outside the ECAC area or flights that were already airborne when the regulation was created, are exempted from the CASA rules. For these unconcerned flights, the closest slot to their ETO the restricted entity is reserved.

[2] A regulation may be also divided in several periods, with a different maximum entry rate defined for each period.

B. Weighted constraint satisfaction problem

Let us define $\boldsymbol{x} \in \mathcal{X}$ as the solution vector, where $\mathcal{X}$ is the solution space. For instance, for a problem with N binary variables (as the problem addressed in this paper), $\mathcal{X} = \{0,1\}^N$. Let us also define $\mathcal{C}$ as a the set of constraints to be satisfied, and $p_l(\boldsymbol{x}) : \mathcal{X} \rightarrow \mathbb{Z}_{\geq 0}$ as the violation of the constraint $l \in \mathcal{C}$ for a specific solution vector $\boldsymbol{x}$. For instance, if a constraint l were defined as $\sum_{i=1}^{N} a_{li} x_i \leq b_l$, where $\boldsymbol{a}_l$ and b_l are fixed parameters of the problem, then $p_l(\boldsymbol{x}) = \max\left(0, \sum_{i=1}^{N} a_{li} x_i - b_l\right)$. Note that this is just an example and constraints do not necessarily have to be linear.

The goal of a constraint satisfaction problem (CSP) is to find a solution vector $\boldsymbol{x}_* \in \mathcal{X}$ such that $p_l(\boldsymbol{x}_*) = 0 \, \forall l \in \mathcal{C}$, i.e., a solution vector that satisfies all constraints.

If the constraints are very restrictive, however, a solution vector that satisfies all of them may not exist. The goal in this case is to satisfy as many constraints as possible, or to minimise the sum of constraint violations to the greatest extent.

In many practical applications, some of the constraints must be satisfied, while others may be violated if it is really difficult to meet them. In the optimisation slang, the former are known as *hard* constraints, and the latter are known as *soft* constraints.

A widely-used technique to solve highly constrained problems involving hard and soft constraints consists of assigning each constraint $l \in \mathcal{C}$ a weight $w_l \in \Re_{\geq 0}$, and then minimise the weighted sum of constraint violations $p(\boldsymbol{x}) : \mathcal{X} \rightarrow \mathbb{R}_{\geq 0}$:

$$\min_{\boldsymbol{x} \in \mathcal{X}} \qquad p(\boldsymbol{x}) = \sum_{l \in \mathcal{C}} w_l p_l(\boldsymbol{x}) \,. \tag{1}$$

$p(\boldsymbol{x})$ is also known in the literature as the penalty function. In the weighted constraint satisfaction problem (WCSP) formalised in Eq. (1), hard constraints are assigned large weights, while soft constraints are assigned relatively small weights.

Determining the specific weight of each constraint is not straightforward, yet the quality of the solution is highly sensitive to $\boldsymbol{w}$: constraints with large weights are likely to dominate the solution, while constraints with relatively small weights may be neglected. To overcome this problem, Ref. [13] proposed to guide the search by an evaluation function $q(\boldsymbol{x}, \boldsymbol{v})$ different from the original penalty function $p(\boldsymbol{x})$, and to dynamically adjust the weights $\boldsymbol{v}$ in such evaluation function during the search based on the sub-gradient method [18].

Although this technique to guide the search and adapt the weights could be used with many optimisation algorithms [15], [16], previous works have demonstrated that it provides excellent results when combined with classical tabu search [14].

C. Adaptive tabu search with strategic oscillations

Tabu search is a local search meta-heuristic that has proven to be effective in solving a large variety of hard combinatorial problems [10]. Like any other local search strategy, tabu search starts from a candidate solution $\boldsymbol{x}$, and then iteratively moves to a neighbour solution based on certain quality measure. Tabu search is frequently referred to as the integration of memory structures into traditional local search algorithms.

Let us define $\mathcal{N}(\boldsymbol{x})$ as the neighbourhood of $\boldsymbol{x}$, where each neighbour $\boldsymbol{x}' \in \mathcal{N}(\boldsymbol{x})$ can be reached from $\boldsymbol{x}$ by performing a single *move*. For solutions composed of N binary variables (i.e., $\mathcal{X} = \{0,1\}^N$), for instance, the n-flip neighbourhood (with $n \in \mathbb{N}_{\leq N}$) includes all solutions that can be reached from $\boldsymbol{x}$ by flipping at most n variables. In this paper, the simplest 1-flip neighbourhood has been implemented:

$$\mathcal{N}(\boldsymbol{x}) = \{\boldsymbol{x}(\neg x_i) \mid i = 1, 2, \ldots, N\}, \tag{2}$$

where $\boldsymbol{x}(\neg x_i)$ is the result of negating the binary variable x_i in $\boldsymbol{x}$ (i.e., of applying a flip move to variable x_i).

The basic form of local search (also known as hill climbing) only permits moves to those solutions in the neighbourhood that improve the quality of the current solution, and ends when no improving solution can be found in the current neighbourhood. This process yields a local optimum solution $\boldsymbol{x}_{\text{local}}$, which is as good or better than all the solutions in its neighbourhood $\mathcal{N}(\boldsymbol{x}_{\text{local}})$. The limitation of this method is that the resulting local optimum is not necessarily the global optimum. It should be noted that the quality of the local optimum typically improves when exploring large neighbourhoods. The time required to explore a large neighbourhood, however, increases dramatically with the number of variables.

The fundamental principle of tabu search is to penalise moves towards previously visited search spaces, which attributes are stored and managed in the memory structures. Furthermore, moving to a non-improving solution may be accepted in order to prevent getting stuck in local optimum.

The most popular memory structure used in tabu search implementations is the recency-based memory. This memory structure, as the name indicates, keeps track of the variables that changed recently (less than t iterations ago, where t is also called the tabu tenure). The variables that changed recently are marked as tabu (prohibited), and solutions containing tabu-active variables cannot be selected. Accordingly, the tabu list $\mathcal{T}$ includes all solution that can be reached from $\boldsymbol{x}$ by flipping any of the variables that changed in the last t iterations:

$$\mathcal{T} = \{\boldsymbol{x}(\neg x_i) \mid x_i \text{ changed in the last } t \text{ iterations}\}. \tag{3}$$

Different from the basic hill climbing strategy, the tabu search changes the current solution $\boldsymbol{x}$ by the *best* solution in the modified neighbourhood $\boldsymbol{x}' \in \mathcal{N}'(\boldsymbol{x})$, which excludes the solutions in $\mathcal{T}$. Furthermore, moving to a non-improving solution is accepted, provided that it is the best in $\mathcal{N}'(\boldsymbol{x})$.

A straightforward approach to quantify the quality of a solution in the modified neighbourhood for guiding the tabu search consists of using the original penalty function $p(\boldsymbol{x})$. In other words, at each iteration, move to the non-tabu neighbour that yields to the lowest $p(\boldsymbol{x})$. This simple approach may not be a good idea when solving difficult WCSPs, because the search may be confined to a small space where only the constraints with very large weights are satisfied. Some of the constraints with relatively small weights, however, may carry priceless information to effectively reach the global optimum.

Strategic oscillations is a technique that has proven successful on solving optimisation problems with very tight constraints. Roughly speaking, the strategic oscillations technique consists of crossing back and forth through feasible and infeasible space, aiming to reach high-quality solutions that otherwise could be missed if only allowing the local search algorithm to explore the (rahter limited) feasible space [12].

In this paper, strategic oscillations are achieved by dynamically adjusting the weight of each constraint individually, right after executing a complete tabu search round. Each tabu search round attempts to solve the following optimisation problem:

$$\min_{\boldsymbol{x}\in\mathcal{X}} \quad q\left(\boldsymbol{x},\boldsymbol{v}\right) = \sum_{l\in\mathcal{C}} v_l p_l\left(\boldsymbol{x}\right). \tag{4}$$

where the evaluation function $q\left(\boldsymbol{x},\boldsymbol{v}\right) : \mathcal{X} \times \Re_{\geq 0}^{\|\mathcal{C}\|} \rightarrow \mathbb{R}_{\geq 0}$ is identical to the original penalty function $p\left(\boldsymbol{x}\right)$, but replacing the original weights $\boldsymbol{w}$ by the adaptive weights $\boldsymbol{0} \leq \boldsymbol{v} \leq \boldsymbol{w}$.

The tabu search round ends when a maximum number of iterations K are executed and local optimality (in terms of the evaluation function) has been achieved at least once (i.e., at some iteration, $\boldsymbol{x}_{\text{local}}$ was not improved). Algorithm 1 lists the basic steps of a single tabu search round for solving Eq. (4).

Algorithm 1 Tabu Search

Require: $\boldsymbol{x}, \boldsymbol{v}$
$(\boldsymbol{x}_{\text{local}}, \boldsymbol{x}_{\text{global}}) \leftarrow (\boldsymbol{x}, \boldsymbol{x})$
$\mathcal{T} \leftarrow \emptyset$
$k \leftarrow 1$
Local optimum $\leftarrow$ False
repeat
  $\boldsymbol{x} \leftarrow \arg\min_{\boldsymbol{x}' \in \mathcal{N}'(\boldsymbol{x})} q(\boldsymbol{x}', \boldsymbol{v})$
  if $q\left(\boldsymbol{x},\boldsymbol{v}\right) < q\left(\boldsymbol{x}_{\text{local}},\boldsymbol{v}\right)$ **then**
    $\boldsymbol{x}_{\text{local}} \leftarrow \boldsymbol{x}$
  else
    Local optimum $\leftarrow$ True
  end if
  if $p\left(\boldsymbol{x}\right) < p\left(\boldsymbol{x}_{\text{global}}\right)$ **then**
    $\boldsymbol{x}_{\text{global}} \leftarrow \boldsymbol{x}$
  end if
  Update the tabu list $\mathcal{T}$
  $k \leftarrow k+1$
until $k \geq K \wedge$ Local optimum
return $(\boldsymbol{x}_{\text{local}}, \boldsymbol{x}_{\text{global}})$

The optimisation problem Eq. (4) addressed by the tabu search round is just an approximation of the original problem Eq. (1). For this reason, the best solution $\boldsymbol{x}_{\text{local}}$ may not be a valid solution in the sense of the original penalty weights $\boldsymbol{w}$.

The tabu search round is repeated by using the best solution found in the previous round as starting point. After each tabu search round, the weight vector $\boldsymbol{v}$ is updated according tho the violations in the current solution, if compared to those of the best solution found so far $\boldsymbol{x}_*$ (the incumbent solution), which is the best solution among all the $\boldsymbol{x}_{\text{global}}$ obtained in the past tabu search rounds (in terms of the original penalty function).

Note that the optimisation problem Eq. (4) could be considered as a Lagrangian relaxation of the original optimisation problem Eq. (1), where $\boldsymbol{v}$ are the Lagrangian multipliers. Accordingly, the following inequality is always satisfied:

$$\min_{\boldsymbol{x}\in\mathcal{X}} q\left(\boldsymbol{x},\boldsymbol{v}\right) \leq \min_{\boldsymbol{x}\in\mathcal{X}} p\left(\boldsymbol{x}\right). \tag{5}$$

On the one hand, if $q\left(\boldsymbol{x},\boldsymbol{v}\right) < p\left(\boldsymbol{x}_*\right)$ holds, it indicates that the weights v_l, $l \in \mathcal{L} \subseteq \mathcal{C}$, are relatively small in comparison with the original weights, where $\mathcal{L}$ is the subset of constraints for which $v_l < w_l$ and $p_l(\boldsymbol{x}) > 0$. Accordingly, these weights are increased by an amount proportional to (1) the corresponding violation $p_l\left(\boldsymbol{x}\right), l \in \mathcal{L}$, and (2) how far the evaluation value of the current solution $q\left(\boldsymbol{x},\boldsymbol{v}\right)$ is from the penalty value of the incumbent solution $p\left(\boldsymbol{x}_*\right)$. This rule to increase the weights is shown in Operation 6 of Algorithm 2.

On the other hand, if $q\left(\boldsymbol{x},\boldsymbol{v}\right) \geq p\left(\boldsymbol{x}_*\right)$ holds, the weights $\boldsymbol{v}$ are reduced before applying Operation 6. The reduction factor is the product of two scalars: α and ρ. The former is a fixed parameter selected by the user, while the latter is introduced in so that no weight $\boldsymbol{v}$ becomes larger than the penalty value of the incumbent solution $p\left(\boldsymbol{x}_*\right)$. The rule to decrease the weights is shown in Operations 2-3 of Algorithm 2.

Algorithm 2 Penalty Weight Update

Require: $\boldsymbol{v}, \boldsymbol{x}, \boldsymbol{x}_*$
1: **if** $q\left(\boldsymbol{x},\boldsymbol{v}\right) \geq p\left(\boldsymbol{x}_*\right)$ **then** ▷ Weights are decreased
2:   $\rho \leftarrow \min\{1, p\left(\boldsymbol{x}_*\right) / \max_l v_l\}$
3:   $\boldsymbol{v} \leftarrow \alpha\rho\boldsymbol{v}$
4: **end if**
5: $\mathcal{L} \leftarrow \{l \in \mathcal{C} \,|\, v_l < w_l \wedge p_l\left(\boldsymbol{x}\right) > 0\}$
6: $v_l \leftarrow \min\{w_l, v_l + \frac{p(\boldsymbol{x}_*) - q(\boldsymbol{x},\boldsymbol{v})}{\sum_{l\in\mathcal{L}} p_l(\boldsymbol{x})^2} p_l\left(\boldsymbol{x}\right)\}, \forall l \in \mathcal{C}$
7: **return** $\boldsymbol{x}$

The standard tabu search in Algorithm 1 is repeatedly executed, updating the penalty weights after each round according to Algorithm 2. The complete adaptive tabu search with strategic oscillations algorithm is described in Algorithm 3.

Algorithm 3 Adaptive Tabu Search with Strategic Oscillations

Require: $\boldsymbol{x}, \boldsymbol{w}$
$\boldsymbol{x}_* \leftarrow \boldsymbol{x}$
$\boldsymbol{v} \leftarrow \boldsymbol{w}$
repeat
  $(\boldsymbol{x}_{\text{local}}, \boldsymbol{x}_{\text{global}}) \leftarrow$ TABUSEARCH $(\boldsymbol{x}, \boldsymbol{v})$
  $\boldsymbol{x} \leftarrow \boldsymbol{x}_{\text{local}}$
  if $p\left(\boldsymbol{x}_{\text{global}}\right) < p\left(\boldsymbol{x}_*\right)$ **then**
    $\boldsymbol{x}_* \leftarrow \boldsymbol{x}_{\text{global}}$
  end if
  $\boldsymbol{v} \leftarrow$ PENALTYWEIGHTUPDATE $(\boldsymbol{v}, \boldsymbol{x}, \boldsymbol{x}_*)$
until TERMINATION
return $\boldsymbol{x}_*$

The Algorithm 3 continues until some TERMINATION criterion is met, e.g., the maximum execution time and/or the maximum number of tabu search rounds are exceeded.

III. Optimisation Algorithm

Given a set of regulations $\mathcal{R}$ proposed by the local FMPs, the objective of this work is to find the optimal subset of regulations $\mathcal{R}_* \subseteq \mathcal{R}$ that minimises the total ATFM delay without creating overloads elsewhere in the network.

Overloads are measured per traffic volume and time window as the traffic load above the declared capacity. The declared capacity, however, does not always reflect the actual capacity. To avoid the algorithm missing a high-quality solution simply to avoid overloading traffic volumes where the declared capacity is well below the actual capacity (and thus these overloads would not cause safety issues), the optimisation problem has been simplified to: minimise the total ATFM delay without creating overloads in the regulated traffic volumes during the applicability period of the corresponding regulations.

The algorithm proposed herein attempts to find the combination of regulations interacting in such a way that the maximum entry rate of each regulation (which does reflects the actual capacity) is not exceeded in the cancelled regulations. In mathematical terms, this problem can be expressed as:

$$\min_{\boldsymbol{x} \in \{0,1\}^{\|\mathcal{R}\|}} \quad \sum_{i \in \mathcal{R}} f_i(\boldsymbol{x}) \tag{6a}$$

$$\text{s.t.} \sum_{j \in \mathcal{W}_i} \max\left(0, d_{ij}(\boldsymbol{x}) - r_i\right) \leq 0, \ \forall i \in \mathcal{R}, \tag{6b}$$

where each element x_i of the solution vector $\boldsymbol{x} \in \{0,1\}^{\|\mathcal{R}\|}$ is a binary variable, which indicates whether the associated regulation i is active ($x_i = 1$) or cancelled ($x_i = 0$); and $f_i : \{0,1\}^{\|\mathcal{R}\|} \to \mathbb{Z}_{\geq 0}$ represents the total ATFM delay of the flights which most penalising regulation is i. Remember from Section II-A that a flight may be subject to several regulations simultaneously, but its assigned ATFM delay is determined by the MPR. Note as well that $\mathcal{R}_* = \{i \in \mathcal{R} \,|\, x_i = 1\}$.

In Eq. (6), $d_{ij} : \{0,1\}^{\|\mathcal{R}\|} \to \mathbb{Z}_{\geq 0}$ represents the traffic load (in terms of entry counts) in the regulation i during the time window j, and r_i is the entry rate of that regulation[3]. As discussed in Section II-A, the interactions between regulations are very difficult to model and predict due to the mechanisms used by CASA to allocate slots. Accordingly, both f_i and d_{ij} are non-linear functions of $\boldsymbol{x}$, i.e., the ATFM delay and traffic load in each regulation are determined by which regulations $i \in \mathcal{R}$ are active ($x_i = 1$) and which are not ($x_i = 0$) .

The time windows $j \in \mathcal{W}$ in which traffic load is aggregated slice every δ minutes, and are Δ minutes width. If $\delta = 20$ and $\Delta = 60$, for instance, the set $\mathcal{W}$ would be composed of {[00:00, 01:00), [00:20, 01:20), [00:40, 01:40), …}.

In this paper, only the time windows within the applicability period of the proposed regulation are monitored. That is:

$$\mathcal{W}_i = \left\{ [n\delta, n\delta + \Delta) \forall n \in \mathbb{Z}_{\geq 0} \,|\, \tau_i^{[} \leq n\delta < \tau_i^{)} \right\}, \ \forall i \in \mathcal{R}, \tag{7}$$

[3]Extending the formulation to regulations composed of multiple periods is straightforward and has not been included for the sake of clarity.

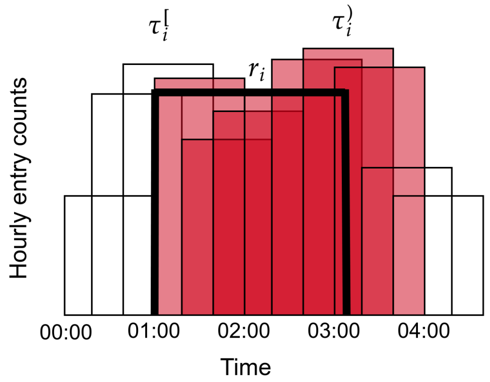


Figure 1. With $\tau_i^{[}$=01:00 and $\tau_i^{)}$=03:10, δ=20 and Δ=60, $\mathcal{W}_i$={[01:00, 02:00), [01:20, 02:20), …, [03:00, 04:00)}

$\tau_i^{[}$ (inclusive) and $\tau_i^{)}$ (exclusive) are the start and end times times of the regulation i, respectively. Figure 1 shows the time windows (in red) monitored for an hypothetical regulation.

The optimisation problem Eq. (6) can be transformed into a WCSP of the form Eq. (1) by simply defining the following set of constraints $\mathcal{C} = \mathcal{C}^c \cup \mathcal{C}^f$, where:

$$\mathcal{C}^f = \left\{ f_i(\boldsymbol{x}) \leq 0 \,|\, i \in \mathcal{R} \right\}, \tag{8a}$$

$$\mathcal{C}^c = \left\{ \sum_{j \in \mathcal{W}_i} \max\left(0, d_{ij}(\boldsymbol{x}) - r_i\right) \leq 0 \,|\, i \in \mathcal{R} \right\}. \tag{8b}$$

On the one hand, Eq. (8a) represents delay constraints, which objective is to decrease the ATFM delay of each regulation as much as possible. In the best of the scenarios, the algorithm would be able to cancel all regulations (i.e., $\mathcal{R}_* = \emptyset$) without creating overloads and therefore $f_i(\boldsymbol{x}) = 0 \, \forall \, i \in \mathcal{R}$.

On the other hand, Eq. (8b) represents the capacity constraints of the original optimisation problem Eq. (6b). Note that, using this formulation, the number of constraints is exactly twice the number of regulations, i.e., $\|\mathcal{C}\| = 2\,\|\mathcal{R}\|$, because each regulation is associated a capacity constraint and a delay constraint. The violation of each constraint $l \in \mathcal{C}$ in the Eq. (1), taking into account the definitions of Eq. (8), is:

$$p_l(\boldsymbol{x}) = \begin{cases} f_i(\boldsymbol{x}) & \text{if } l \in \mathcal{C}^f \\ \sum_{j \in \mathcal{W}_i} \max\left(0, d_{ij}(\boldsymbol{x}) - r_i\right) & \text{if } l \in \mathcal{C}^c, \end{cases} \tag{9}$$

The generic adaptive tabu search with strategic oscillations presented in Section II-C could be used to solve this problem. Within each round of tabu search (Algorithm 1), a naïve evaluation of the neighbourhood of $\boldsymbol{x}$ in Operation 6 would require to run $\|\mathcal{R}\|$ times the CASA algorithm and perform a network impact assessment (NIA) to compute the new ATFM delay created by each regulation as well as the traffic load in each traffic volume and time window. In the high-fidelity, real-time simulator used in this study, evaluating a single solution $\boldsymbol{x}$ takes between 3 and 10 seconds (depending on the number of flights and traffic volumes to be monitored in the NIA).

The volatility of the network is very high, requiring the algorithm to find a solution (ideally) in less than 5 minutes. For this reason, the algorithm should converge after evaluating, at most, 100 different solutions $\boldsymbol{x}$. For a typical problem with 20 regulations, the number of possible combinations assuming that each regulation can take two states (active or cancelled) is $2^{20} \simeq 10^6$, which implies that the algorithm is only allowed to explore around 0.01% of the total search space.

In order to reduce the number of solutions evaluated in each iteration of the tabu search round (Algorithm 1), the *aspiration plus* strategy has been implemented [19]. This strategy establishes an aspiration threshold for the quality of a move to be explored, and examines moves until finding one that satisfies this threshold. In this paper, the threshold is set to the evaluation value in the current solution, i.e., $q(\boldsymbol{x}, \boldsymbol{v})$.

After finding a candidate $\boldsymbol{x}'$ for which $q(\boldsymbol{x}', \boldsymbol{v}) < q(\boldsymbol{x}, \boldsymbol{v})$ holds, `plus` additional moves are examined, and the best move (in terms of the evaluation value) is selected. In order to ensure that neither too few nor too many moves are examined, at least `min_moves` moves and at most `max_moves` are examined. Both `plus`, `min_moves` and `max_moves` are fixed parameters selected by the user. Note that by setting $\texttt{plus} = \texttt{max\_moves} = \infty$, one explores the whole neighbourhood, just like in the standard tabu search.

The order in which moves are evaluated is really important when using the aspiration plus strategy. During the evaluation of the neighbourhood of $\boldsymbol{x}$, the moves that have not yet been explored are randomly selected to encourage diversification. The probability of exploring a move (i.e., activating a cancelled regulation or cancelling one that is active) is proportional to the amount of evaluation value that can be attributed to the corresponding regulation in the current solution.

Therefore, each unexplored move is evaluated with a probability proportional to the penalised amount of violations caused by the corresponding regulation in the current solution.

Note that Algorithm 3 has three parameters: α, K and t. The criteria proposed in Ref. [13] has been adopted to determine the values of α and K. Regarding t, the effective tabu tenure t' is randomly selected from the set $\{t-1, t, t+1\}$ in each iteration of the tabu search, aiming to promote diversification, i.e., when constructing the tabu list, t' is used instead of t .

IV. Illustrative Example

The generic adaptive tabu search with strategic oscillations presented in Section II-C was tailored to the problem of finding the most effective subset of ATFM regulations (from those proposed by the local FMPs) according to Section III.

In order to obtain representative results, the validation exercise was performed by using the Network Manager validation platform (NMVP), which fully replicates the real Network Management operational systems (including CASA).

The NVMP can be used with live traffic for *shadow* mode trials, and also with historical traffic to replay past days. The validation exercise was performed by replaying a busy day in real-time, iteratively calling CASA and performing a NIA to evaluate each solution $\boldsymbol{x}$ during the optimisation process.

The metrics used to assess the performance of the algorithm include: the difference of ATFM delay as a result of cancelling the regulations suggested by the algorithm (if any), as well as the variation in the traffic load, in terms of entry counts.

Remember that the algorithm operates by monitoring the ATFM delay and overloads only in the regulations that are being optimised. Cancelling a specific group of regulations, however, may have substantial consequences throughout the network. Accordingly, the optimisation process must be always followed by a comprehensive NIA to verify the solution.

Section IV-A describes the setup of the validation exercise. The main results of the optimisation process are presented in Section IV-B. Finally, a comprehensive network impact assessment after cancelling the regulations proposed by the algorithm is thoroughly discussed in Section IV-C.

A. Setup of the validation exercise

The real-time simulation in the NVMP started at 07:40 AM of the 27th of June 2019, when 120 regulations were active in the network, generating around 43K minutes of ATFM delay. For this particular validation exercise, the objective of the algorithm was stated as to find effective regulations caused by air traffic control (ATC) capacity issues in a sub-region composed of all traffic volume sets in LE*, LF* and ED*, which embrace Spain, France, Belgium, Germany, Luxembourg and the Netherlands. The total number of regulations due to ATC capacity issues applied in the operational system when starting the simulation (at 07:40 AM) over the sub-region of interest was 21, generating 5343 minutes of ATFM delay.

The algorithm was executed without pausing the clock, aiming to replicate as closely as possible a real-world situation where the execution time cannot be neglected.

Table I lists the parameters of the optimisation algorithm that were selected for this validation exercise.

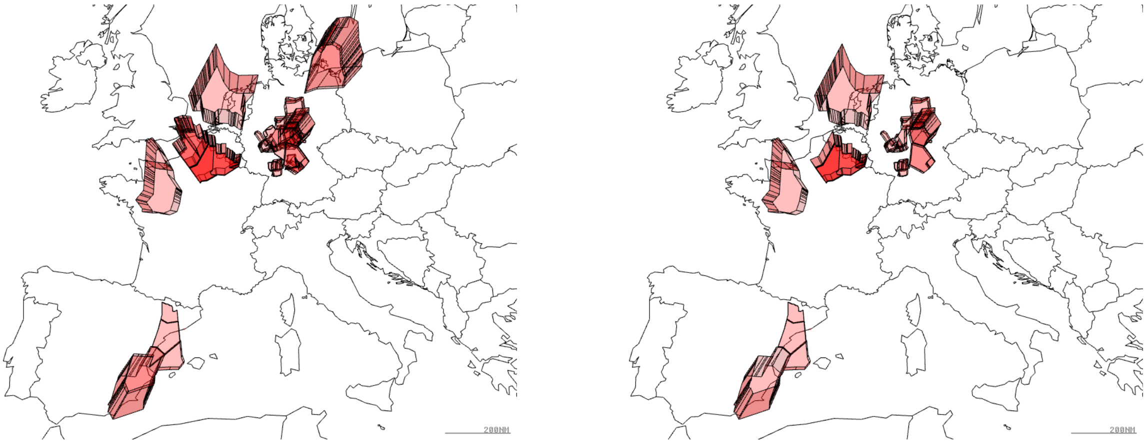
TABLE I. Parameters of the optimisation algorithm

Parameter	Value
α	0.5
t	3
$w_l,\ l \in \mathcal{C}^f$	0.1.
$w_l,\ l \in \mathcal{C}^c$	18.
`plus`	5
`min_moves`	7
`max_moves`	15
Maximum number of solutions	100

According to Table I, 1 single overload was penalised as 180 minutes of ATFM delay. The algorithm was allowed to explore 100 solutions out of the $\sim 2 \cdot 10^6$ possible combinations in the solution space $\{0,1\}^{21}$. After execution, the suggestion of the algorithm was simulated, followed by a NIA.

B. Results of the optimisation

Figure 2 shows the set of applied regulations before (a) and after (b) the optimisation process. Table II shows the number of flights within each regulation as well as the generated ATFM delay, before and after the optimisation process.



(a) Before optimisation $\mathcal{R}$ (21 regulations) (b) After optimisation $\mathcal{R}_*$ (13 regulations)

Figure 2. Applied regulations - Figure generated with the research network strategic monitoring tool (R-NEST)

Cancelled regulations (in red) show a significant reduction of the generated ATFM delay, as expected. Some of these regulations, however, still create some delay after being cancelled. The reason of this counter-intuitive result is that some of the regulated flights may be already airborne or very close to the calculated off-block time (COBT), and in these specific circumstances cancelling the regulation has no effect. Regulations that were deemed effective (in black) either maintain or increase the ATFM delay before optimisation.

The local effect of cancelling the regulations not in $\mathcal{R}_*$ was a reduction of 1380 (26%) minutes of delay. As mentioned previously, however, the algorithm just solved a local problem, and one needs to perform a NIA in order to ensure that cancelling the regulations also has a positive effect network-wide (and not only in the sub-region being optimised).

TABLE II. Impact for regulations in $\mathcal{R}$

Regulation i	Flights		Delay		
	Before	After	Before	After	Difference
KWUR327E	183	181	634	262	-372
KFFM2427	132	127	345	0	-345
LELVL27	93	58	280	0	-280
E4N27	246	175	278	0	-278
EDG7NO27	57	35	236	3	-233
EDG4DK27	51	22	103	0	-103
EDG3GH27	33	26	57	20	-37
KOSE1O27	52	46	87	54	-33
LEBLA27	44	44	107	107	0
E5R27	278	278	0	0	0
EDG3GI27	47	47	222	222	0
KWUR1C27	67	67	132	132	0
K1UFX27	67	67	50	50	0
EDG2NT27	44	44	165	165	0
LEGE1V27	72	72	132	145	+13
KWUR2427	127	127	537	552	+15
RQXU27E	51	51	145	170	+25
EDG3TA27	34	34	365	399	+34
EDWHRZ27	70	70	271	312	+41
LELVU27	98	98	518	561	+43
YD6WH27M	145	145	679	809	+130
Total	1991	1814	5343	3963	-1380

C. Network impact assessment

Tables III and IV summarise the main results of the NIA.

Table III shows the change in flights and generated ATFM delays for the 120 regulations that were applied at 07:40 AM in the network, before and after cancelling the 8 (out of 21) regulations proposed by the algorithm. Regulations are ordered by delay difference, i.e., the 10 first regulations are those that decreased more the delay, while the 10 last regulations suffered an increase of delay after putting in place the suggestion of the algorithm. The network-wide effect of cancelling the 8 regulations was a reduction of 2209 (5%) minutes of delay.

Table IV shows the change in overloads (in terms of hourly entry counts above the declared capacity) for the traffic volumes that were impacted by the 8 cancellations. As in Table III, results are sorted by difference (after-before), from most negative to most positive. According to Table IV, the impact was rather neutral, with some traffic volumes increasing the amount of overloads and some others decreasing it. The network-wide effect was a residual reduction of 3 overloads (0.16%), and the narrow distribution ranged from -2 to +2.

V. Conclusions

This paper proposed a meta-heuristic algorithm based on adaptive tabu search with strategic oscillations to detect effective regulations from all those proposed by the local flow management positions in a sub-region of the network.

Initial results suggest that the algorithm is able to identify effective regulations by exploring a limited number of solutions in the search space, thus being suitable during the tactical phase of operations. Furthermore, the outcome of the network impact assessment indicates that the local solution has minor impact (in terms of delays and entry counts) network-wide.

In future work, the algorithm must be tested on more scenarios in order to obtain statistically meaningful results from which to draw solid conclusions. Furthermore, a comprehensive comparison with other algorithms, like harmony search, adaptive large neighbourhood search, or branch and bound is also foreseen in future publications.

TABLE III. NETWORK IMPACT ASSESSMENT: DELAYS

Regulation i	Flights		Delay		
	Before	After	Before	After	Difference
KWUR327E	183	181	634	262	-372
KFFM2427	132	127	345	0	-345
LHLYBA27	290	290	1410	1074	-336
LELVL27	93	58	280	0	-280
E4N27	246	175	278	0	-278
EDG7NO27	57	35	236	3	-233
KNTM3C27	156	156	1583	1356	-227
LOWB3527	65	65	577	395	-182
LOWB1227	155	155	2021	1906	-115
EDG4DK27	51	22	103	0	-103
...					
EDG3TA27	34	34	365	399	+34
LHWSLM27	177	177	571	610	+39
EDWHRZ27	70	70	271	312	+41
LPPTA27	120	120	3679	3721	+42
LELVU27	98	98	518	561	+43
EHAMA27M	224	224	2306	2349	+43
LHENU27	116	116	241	296	+55
EGLLA27M	112	112	2476	2556	+80
KNTM1C27	90	90	1386	1479	+93
YD6WH27M	145	145	679	809	+130
Total	7304	7127	43051	40842	-2209

TABLE IV. NETWORK IMPACT ASSESSMENT: OVERLOADS

Traffic volume	Period	Before	After	Difference
LFEUBN	08:20 - 11:40	8	6	-2
EG12GCLE	06:20 - 08:40	7	5	-2
LEMDARR	08:20 - 10:40	2	0	-2
LOVVWB12	06:00 - 11:20	15	13	-2
LFEKR	11:00 - 11:40	2	1	-1
MASB5WL	05:40 - 11:00	10	9	-1
LOVVE15	07:20 - 12:20	13	12	-1
LHLYBALL	09:20 - 10:40	2	1	-1
LCWU	10:00 - 12:00	5	4	-1
LDULWX	07:40 - 08:20	2	1	-1
...				
LPCEU1	11:20 - 11:20	0	1	+1
LFMY34	09:20 - 10:20	2	3	+1
LFEKHR	08:40 - 15:00	17	18	+1
EDG1LA05	07:20 - 07:20	0	1	+1
LFMMOML	08:00 - 09:40	2	3	+1
EGGWARR	10:40 - 15:40	4	5	+1
EGDTS	05:40 - 07:20	4	5	+1
EDG4DKB	05:40 - 08:40	0	1	+1
LOVVWB35	07:00 - 11:20	14	16	+2
LYBATWES	06:40 - 11:20	14	16	+2
Total	-	1831	1828	-3

ACKNOWLEDGEMENT

The authors would like to thank the Network Manager experts Giovanni Lenti, Sylvie Besnard and Kristof Voet for their support and priceless operational feedback.

REFERENCES

[1] *European Aviation in 2040: Challenges of growth*, EUROCONTROL, October 2018.

[2] F. Serrano and A. Kazda, "The future of airports post covid-19," *Journal of Air Transport Management*, vol. 89, p. 101900, 2020.

[3] *Forecast Update 2021-2024. European Flight Movements and Service Units. Three Scenarios for Recovery from COVID-19*, EUROCONTROL, May 2021.

[4] S. Ruiz, H. Kadour, and P. Choroba, "An innovative safety-neutral slot overloading technique to improve airspace capacity utilisation," in *SESAR Innovation Days (SID) 2019*, Athens, Greece, December 2019.

[5] R. Sanaei, B. A. Pinto, and V. Gollnick, "Toward atm resiliency: A deep cnn to predict number of delayed flights and atfm delay," *Aerospace*, vol. 8, no. 2, 2021.

[6] A. Jardines, M. Soler, and J. García-Heras, "Estimating entry counts and atfm regulations during adverse weather conditions using machine learning," *Journal of Air Transport Management*, vol. 95, p. 102109, 2021.

[7] L. Delgado, G. Gurtner, T. Bolić, and L. Castelli, "Estimating economic severity of air traffic flow management regulations," *Transportation Research Part C: Emerging Technologies*, vol. 125, p. 103054, 2021.

[8] A. K. Agogino and K. Tumer, "A multiagent approach to managing air traffic flow," *Autonomous Agents and Multi-Agent Systems*, vol. 24, no. 1, pp. 1–25, 2012.

[9] A. Marcio, F. Crespo, L. Weigang, and A. Gomes De Barros, "Reinforcement learning agents to tactical air traffic flow management," *International Journal of Aviation Management*, no. 3, pp. 145–161, 2012.

[10] A. Tuson, F. Glover, and M. Laguna, "Tabu search," *The Journal of the Operational Research Society*, vol. 50, p. 106, 01 1999.

[11] K. Nonobe and T. Ibaraki, "A tabu search approach to the constraint satisfaction problem as a general problem solver," *European Journal of Operational Research*, vol. 106, no. 2, pp. 599–623, 1998.

[12] F. Glover and J.-K. Hao, "The case for strategic oscillation," *Annals of Operations Research*, vol. 183, pp. 163–173, 03 2011.

[13] K. Nonobe and T. Ibaraki, "An improved tabu search method for the weighted constraint satisfaction problem," *INFOR: Information Systems and Operational Research*, vol. 39, no. 1, pp. 131–151, 2001.

[14] D. Santos, P. Fernandes, H. L. Cardoso, and E. Oliveira, "A weighted constraint optimization approach to the nurse scheduling problem," in *2015 IEEE 18th International Conference on Computational Science and Engineering*, 2015, pp. 233–239.

[15] S. Umetani, "Exploiting variable associations to configure efficient local search algorithms in large-scale binary integer programs," *European Journal of Operational Research*, vol. 263, no. 1, pp. 72–81, 2017.

[16] S. Umetani, M. Arakawa, and M. Yagiura, "Relaxation heuristics for the set multicover problem with generalized upper bound constraints," *Computers & Operations Research*, vol. 93, pp. 90–100, 2018.

[17] *ATFCM Users Manual*, EUROCONTROL, November 2018, rev. 22.1.

[18] M. Held, P. Wolfe, and H. P. Crowder, "Validation of subgradient optimization," *Mathematical Programming*, vol. 6, pp. 62–88, 1974.

[19] F. Glover, M. Laguna, P. Pardalos, D.-Z. Du, and R. Graham, "Tabu search: effective strategies for hard problems in analytics and computational science," *Handbook of Combinatorial Optimization*, vol. 21, pp. 3261–3362, 01 2013.