

D3.3 EVALUATION AND VALIDATION OF COLLABORATIVE TRAJECTORY PREDICTION ALGORITHM

DART

Grant:

699299

Call:

ER-2-2015

Topic:

Data Science in ATM

Consortium coordinator:

University of Piraeus Research Center

Edition date:

16 May 2018

Edition:

[02.00.00]

Founding Members



Authoring & Approval

Authors of the document

Name/Beneficiary	Position/Title	Date
George Vouros	Project Coordinator	16.05.2018
Konstantinos Blekas	Project Member	16.05.2018
Theocharis Kravaris	Project Member	16.05.2018
Christos Spatharis	Project Member	16.05.2018
Natalia Andrienko	Project Member	16.05.2018
Gennady Andrienko	Project Member	16.05.2018
Georg Fuchs	Fraunhofer Group Leader	16.05.2018

Reviewers internal to the project

Name/Beneficiary	Position/Title	Date
Georg Fuchs/Fraunhofer	Fraunhofer Group Leader	18.05.2018
Enrique Casado/BR&T-E	BR&T-E representative, Dissemination and Exploitation Manager	18.05.2018
Jose Manuel Cordero/CRIDA	CRIDA Representative, Project Member	18.05.2018

Approved for submission to the SJU By — Representatives of beneficiaries involved in the project

Name/Beneficiary	Position/Title	Date
George Vouros/UPRC	Project Coordinator	18.05.2018
Enrique Casado/BR&T-E	Project Member	18.05.2018
George Fuchs/FRHF	Project Member	18.05.2018
Jose Manuel Cordero/CRIDA	Project Member	18.05.2018

Rejected By - Representatives of beneficiaries involved in the project

Name/Beneficiary	Position/Title	Date
------------------	----------------	------

Document History

Edition	Date	Status	Author	Justification
00.05.00	29/03/2018	First Draft	George Vouros	
00.06.00	12/04/2018	First Draft with results	Christos Spatharis, Theocharis Kravaris, Konstantinos Blekas	



00.08.00	16/04/2018	First Draft with results and visualizations	Gennady Andrienko, Natalia Andrienko, Georg Fuchs	
00.09.00	17/04/2018	Draft with results, visualizations and description/comments on results	George Vouros, Konstaninos Blekas	
01.00.00	20/04/2018	Final with results, visualizations and description/comments on results	George Vouros, Konstaninos Blekas	
02.00.00	16/05/2018	Additional clarifications and explanations	George Vouros	Addressing SJU comments

DART

DATA DRIVEN AIRCRAFT TRAJECTORY PREDICTION RESEARCH

This document is part of a project that has received funding from the SESAR Joint Undertaking under Grant Agreement No 699299 under European Union's Horizon 2020 research and innovation programme.



Abstract

This deliverable presents evaluation results from the collaborative reinforcement learning algorithms designed and implemented towards resolving DCB problems at the pre-tactical stage of operations.

Towards that, the document – to be self-contained- presents succinctly the operational context of DART, the specific problem considered towards assessing the impact of traffic on multiple flights' trajectories with respect to the Demand-Capacity Balance (DCB) problem, as well as the proposed multi-agent Collaborative Reinforcement Learning algorithms proposed. Then, it presents the specific methodology for constructing evaluation cases and the cases themselves, by exploiting the data provided in DART consortium. Then, it proceeds to present and discuss thoroughly the experimental results for each of the methods also in comparison to the CFMU data, discussing the benefits and limitations of individual methods.

Finally, the document presents visualizations of a specific solution in space and time for one of the evaluation cases considered (the one requiring the higher average delay for the regulated flights), providing further evidence for the quality of solutions, which is representative of the other cases.



Table of Contents

<i>Executive Summary</i>	<i>6</i>
<i>1 Introduction</i>	<i>7</i>
<i>2 Operational Context, Problem Specification and Collaborative Reinforcement Learning Algorithms</i>	<i>11</i>
<i>3 Evaluation Results</i>	<i>18</i>
<i>4 Visualizations of solutions overview in space and time</i>	<i>49</i>
<i>5 Conclusions</i>	<i>72</i>
<i>6 References.....</i>	<i>74</i>

Executive Summary

This deliverable reports on evaluation results of collaborative Reinforcement Learning (RL) algorithms for assessing delays on regulated flights to resolve Demand-Capacity imbalances. These algorithms implement agent-based modelling approaches towards accounting for complex phenomena in ATM due to network effects. The developed methods allow agents – representing individual flights- to learn offline and in batch-mode, and in a totally distributed way, *own* policies (i.e. regulations) to *jointly* resolve Demand-Capacity imbalances. Jointly here means that agents' policies happen concurrently and in ways that the policy of one affects the other agents' policies, given also the overall dynamic contextual information regarding operational constraints.

The objective is to understand if agent-based modelling approaches are capable of assessing delays to flights to effectively resolve DCB problems at the pre-tactical stage, considering all trajectories and exogenous factors.

Therefore, results aim to provide evidence on the feasibility of the proposed methods, based on quantitative measurements, as well as on qualitative aspects regarding the quality of the solutions produced. For this purpose, several real-world cases have been identified as representative of a variety of operational cases to evaluate the proposed methods.

Exploiting DART datasets, the specific evaluation procedure focuses on real-world evaluation cases comprising intended trajectories (flight plans). In every case, the evaluation process involves replaying these cases individually, benchmarking the results of algorithms with respect to known regulations from CFMU data provided in DART.¹

¹ The opinions expressed herein reflect the author's view only. Under no circumstances shall the SESAR Joint Undertaking be responsible for any use that may be made of the information contained herein.

⁶ Copyright 2018 DART

1 Introduction

1.1 Purpose and Scope

The goal is to deliver an understanding on the suitability and efficacy of agent-based models for resolving the DCB problem in Air Traffic Management at the pre-tactical stage and contribute to the envisioned collaborative decision-making method among ATM actors towards reaching agreements on trajectories to be flown.

Towards this purpose, this deliverable to be self-contained presents very succinctly the operational context of DART, the specific problem considered towards assessing the impact of traffic on multiple flights' trajectories with respect to the Demand-Capacity Balance (DCB) problem, as well as the proposed multi-agent Collaborative Reinforcement Learning algorithms proposed.

Then, the document presents the specific methodology for constructing evaluation cases and the cases themselves by exploiting the data provided in DART consortium. Then, it proceeds to present and discuss the experimental results for each of the methods also in comparison to the CFMU data.

Finally, the document presents visualizations of a specific solution in space and time for one of the evaluation cases considered (the one requiring the higher average delay for the regulated flights), providing further evidence for the quality of solutions, which is representative of the other cases.

1.2 Intended readership

This document is intended to be used by DART members and SJU.

1.3 Acronyms and Terminology

Term	Definition
ANSP	Air Navigation Service Provider
ATM	Air Traffic Management
ATC	Air Traffic Control
ATS	Air Traffic Services
AO	Aircraft Operators
AU	Airspace User
CFMU	Central Flow Management Unit
DCB	Demand and Capacity Balancing
FIR	Flight Information Region
HEC	Hourly Entry Count
HFIR	Estimated Entry Date and Time to FIR
Horizon 2020	EU Research and Innovation programme implementing the Innovation Union, a Europe 2020 flagship initiative aimed at securing Europe's global competitiveness.
HRL	Hierarchical Reinforcement Learning
IOBT	Initial Off-Block Time
NM	Network Manager
MDP	Markov Decision Process
RBT	Reference Business Trajectory
RL	Reinforcement Learning
SESAR	Single European Sky ATM Research Programme
SJU	SESAR Joint Undertaking (Agency of the European Commission)
WP	Work Package

Table 1: Acronyms and Terminology

1.4 Relation to other Work Packages and Deliverables

This deliverable is related to WP1, since it exploits information of flight plans and airspace sectorizations (configurations) to re-construct real-world evaluation cases, which are instances of DCB problems. It also exploits CFMU data - providing regulations imposed to flights to resolve DCB problems- in order to identify flights to be regulated and compare the quality of solutions achieved by the proposed methods to those produced by the NM. These data sources are being reported in D1.3 “DART Data Pool”. In addition to these sources of data, methods exploit cost indicators related to strategic delay costs for European airlines, according also to the aircraft model used as part of the multi-agent methods applied.

In addition to that, WP3 methods reported are applicable to the trajectories predicted by WP2 methods, although developments in both work packages are quite independent, as planned.

Finally, this deliverable is closely connected to work carried out in task 3.1 “Scenarios setup and specification of requirements” and the operational context of research described in deliverable D3.1 “Collaborative Trajectory Prediction Scenarios and Requirements Specification”: A succinct reference to the operational context is provided below.

1.5 Research Approach

Task 3.2 “Collaborative reinforcement learning for trajectory predictions” aims to formulate the problem of assessing the impact of traffic to individual trajectories as a Markov Decision Process (MDP). Based on this formulation collaborative RL algorithms for trajectory prediction have been designed and implemented, allowing agents to learn offline and in batch-mode policies to resolve DCB problem cases jointly, taking into account other agents’ trajectories, contextual information, cost indicators and own preferences. The MDP formulation of the problem and the algorithms proposed are described in D3.2 “Collaborative Trajectory Prediction Algorithm”.

Therefore, following our research methodology, this deliverable, to be self-contained, succinctly reports on the specification of the particular problem considered in WP3 scenario, the multi-agent MDP framework that formulates the problem considered, as well as on collaborative reinforcement learning algorithms implemented and tested in real-world datasets towards resolving the problem cases considered in D3.1.

Finally, Task 3.4 “Model Test, Validation & Visualization” aims to test and validate the algorithms developed using actual and synthetic data gathered and/or generated in WP1, according to the scenarios and the requirements specified in Task 3.1. The evaluation cases constructed, the construction methodology, and the results of the proposed methods are reported in this deliverable. Multiple criteria for algorithms evaluation/ validation are considered: resulting number of hotspots, the average delay for the regulated and for all flights, distribution of delays to flights and distribution/evolution of demand in sectors and time periods after flights being regulated. Visualizations of solutions’ overview in space and time provide further insights into the quality of solutions computed by the proposed methods.

1.6 Expected Results

Our goal in DART is to develop collaborative RL algorithms that will be trained in batch mode (i.e. offline) and will be applied to assessing the effect of multiple flights co-occurring in specific contexts to the Demand-Capacity Balance (DCB) phenomena, taking into account data from multiple sources, including single trajectory predictions and/or flight plans, while learning efficiently in few exploration episodes.

The objective is to understand if an agent-based model is capable of resolving effectively DCB problems at the pre-tactical stage, considering all trajectories and exogenous factors.

Results are quantifiable assessments (metrics) concerning the quality of solutions produced, as well as qualitative aspects regarding the feasibility of the methods. For this purpose, several datasets – representing real-world cases – have been identified as representative of a variety of operational cases to evaluate the proposed methods.

Exploiting the datasets provided in DART, the specific evaluation procedure applied focuses on real-world cases comprising flight plans, given the availability of this data in DART as well as the availability of data to assess the quality of solutions. In every case, the evaluation process involves replaying these cases individually, benchmarking the results with respect to known regulations (this can be done as the datasets contain every snapshot of flight plan status, from planning phase to flight cancellation after landing, while CFMU data provided in DART contain regulated flights).

It must be noted that while DART methods operate at the pre-tactical stage to assess the delay that should be imposed to flights towards resolving all hotspots, CFMU regulations, to which DART solutions are compared, concern the delays imposed to flights to resolve some of the hotspots: However, this does not hinder the comparability of solutions, given that (a) CFMU provides a kind of “ground truth” for the measures to be applied in “real life” and (b) DART contributes to the strategic and pre-tactical demand-capacity balancing evaluation, simulation and display tools, aiming to reduce ATC workload. Thus, this comparison provides evidence for reaching this target.

2 Operational Context, Problem Specification and Collaborative Reinforcement Learning Algorithms

The WP3 scenario objective, as this has been specified in deliverable D3.1 “Collaborative Trajectory Prediction Scenarios and Requirements Specification”, is to demonstrate how DART agent-based modelling capability can help to accounting for the complexity of the ATM due to network effects regarding the influence of the traffic to individual trajectories.

Specifically, the scenario concerns regulating flights towards resolving DCB problems, assuming that the whole process happens at the planning phase (i.e., days before operation), as opposed to the tactical phase (i.e. in real-time during operation). The scenarios are considered to be developed in a specific geographical area (Spain), without affecting the generality of the solutions proposed, while interests of different stakeholders, such as Air Navigation Service Providers (ANSPs) and airspace users (AUs), are taken into account: Air Navigation Service Providers require resolving the demand-capacity imbalances efficiently, while airspace users (e.g. airlines) aim to operate safely and efficiently without large delays.

The ANSP role is represented by CRIDA (local level) and airspace users’ role is represented by BR&T-E. The separation between aircraft is guaranteed; therefore, resolutions adopted by ATCO won’t be part of the scope in the operational scenario WP3.

In this case, regulations of type C (i.e. delays) will be applied to trajectories due to the imbalance between demand and capacity of airspace sectors, so DART will have to apply such regulations and obtain the final trajectories taken into account surrounding traffic.

2.1 Data and Steps

The data involved in this scenario is:

- Flight Plans: Plans associated with the trajectories.
- Airspace Structure & Capacity: Sectorization information available at operation day (sector configurations and airblocks)
- Strategic Delay Costs, as estimated in [Cook et al, 2015].
- CFMU datasets providing data for regulated flights.

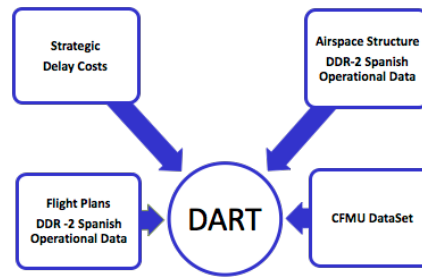


Figure 1 Data required for WP3 Scenario

Steps

WP3 Scenario considers demand and capacity balance per sector (DCB): This should result to regulating flights by imposing delays to individual flights, taking into account strategic delay cost and operational constraints. In doing so we apply algorithms to fulfil two main objectives: firstly, *detect DCB imbalances* per sector, and secondly, *resolve the imbalances*.

- Detecting DCB imbalances. The first step is to focus on detecting demand and capacity imbalances due to a lack of airspace capacity. This is possible by using data of airspace configurations and de-conflicted trajectories.
- Application of agent-based methods to DCB problems resolution. In doing so, WP3 takes into account interactions among trajectories (i.e. traffic conditions), airspace configurations, and cost indicators, considering all trajectories in a joint manner.

The final output will be the most appropriate trajectory that aircraft must finally follow (RBT), jointly with others. The output will be a single trajectory per flight, specifying the delay imposed due to DCB problems.

2.2 Problem Specification

The DCB problem (or process) considers two important types of objects in the ATM system: *trajectories* and *airspace sectors*.

Aircraft trajectories are series of spatio-temporal points of the generic form $(long_i, lat_i, alt_i, t_i)$, denoting the longitude, latitude and altitude, respectively, of the aircraft at a specific time point t_i . A specific type of trajectory is a *flight plan*, which is an intended trajectory consisting of events of flights crossing air blocks and sectors, and flying over specific waypoints. Specifically, each event specifies the element that is crossed (air block or sector), the entry and exit locations (coordinates + flight levels), and the entry and exit times, or the time that the flight will fly over a specific waypoint. Other information such as estimated take-off time are specified, and, in case of delay, the calculated take-off time, or the Estimated time of entry to FIR (HFIR).

Sectors are air volumes segregating the airspace, each defined as a group of air volumes. The airspace sectorization may be done in different ways, depending on the number of active (open) sectors. Only one sector configuration is active at a time for a specific ATC.

Now, let there be \mathbf{N} trajectories in a set \mathbf{T} that must be executed over the airspace in a total time period of duration \mathbf{H} (typically in 24 hours). The set of sectors in all possible airspace configurations is denoted by \mathbf{S} . Time can be divided in intervals of duration Δ_t , equal to that of the period duration of the respected measure for measuring demand evolution.

Thus, a trajectory T in the set of trajectories considered, \mathbf{T} , is a time series of elements of the form:

$$T = [(sector_1, entry_{t1}, exit_{t1}) \dots (sector_m, entry_{tm}, exit_{tm})],$$

where $sector_i$ is in \mathbf{S} , $i=1, \dots, m$.

It must be noticed that (a) given different delays imposed to a trajectory, sectors crossed may differ, due to the changing sector configurations; (b) this may result to a number of alternative representations of a single trajectory (each representation crossing a different set of sectors), one for each possible delay.

This information per trajectory suffices to measure the demand $D_{s,p}$ for each of the sectors s in the airspace in any period p of duration Δ_t . Specifically, $D_{s,p} = |\mathbf{T}_{s,p}|$, i.e. the number of trajectories in $\mathbf{T}_{s,p}$, where

$$\mathbf{T}_{s,p} = \{T \in \mathbf{T} \mid T = (\dots, (s, entry_t, exit_t), \dots), \text{ and the temporal interval } [entry_t, exit_t] \text{ overlaps with } p\}.$$

The trajectories in $\mathbf{T}_{s,p}$ are defined to be *interacting trajectories* for the period p and the sector s .

In other words, interacting trajectories are considered to be those that co-occur in space (within sectors) and time (with time periods). These are candidates to be delayed, i.e. any subset of these may finally result with delays.

In order to calculate the total demand at any given state we use the Hourly Entry Count measure (HEC), creating a series of vectors (one for each period) for each sector. Given the demand per sector, the DCB problem consists of these cases where the demand exceeds capacity:

Each sector s in \mathbf{S} has a specific capacity C_s that determines the maximum number of flights flying within the sector during a specific time interval. Imbalances of sectors' demand and capacity occur when $D_{s,p} > C_s$, for any period p of duration Δ_t in \mathbf{H} . These cases result to *hotspots* in the airspace.

In case of capacity violation for a period p and sector s , the interacting trajectories in $\mathbf{T}_{s,p}$ are defined as *hotspot-constituting trajectories*: one or more of these trajectories must be regulated in order to resolve the imbalance in s and p .

Towards the agent-based formulation of the problem, we consider the following:

Each agent A_i is specified to be the aircraft performing a specific trajectory in a specific date and time. Thus, we consider that agents and trajectories coincide in our case and we may interchangeably speak of agents A_i , trajectories T_i , or agents A_i executing trajectories T_i . Agents, as it will be specified, have own interests and preferences, although they are assumed to be collaborative, and take autonomous decisions on their delays.

Therefore, agents have to learn *joint* delays to be imposed to their trajectories w.r.t. the operational constraints concerning the capacity of sectors crossed by these trajectories.

It must be noted that agents –although considered collaborative- have conflicting preferences, since they prefer to impose the smallest delay possible (preferably none) to their own trajectory, minimizing costs, while also executing their planned trajectories safely and efficiently.

Given an agent A_i , the traffic for that agent is determined to be the trajectories of all other agents with whom it interacts. More formally:

Traffic(A_i) = $\{ T_j \mid T_j \text{ is a trajectory that interacts with the trajectory } T_i \text{ executed by } A_i \text{ for any specific sector crossed by } T_i \text{ and any time period within } H \}$,

or

$$\mathbf{Traffic}(A_i) = \bigcup_{s_i, p} \mathbf{T}_{s_i, p},$$

where s_i is any sector crossed by trajectory T_i , and p is any time period in H .

A society of agents (\mathbf{A}, \mathbf{E}) is modelled as a **coordination graph** with one vertex per agent A_i in \mathbf{A} and any edge (A_i, A_j) in \mathbf{E} connecting agents with interacting trajectories in \mathbf{T} . This set of edges are dynamically updated by adding new edges when new interacting pairs of trajectories appear.

$\mathbf{N}(A_i)$ denotes the *neighbourhood* of agent A_i in the coordinating graph, i.e. the set of agents interacting with agent A_i including also itself: i.e. agents executing trajectories in **Traffic**(A_i). These are the peers/neighbors of A_i in the agent society.

The options available in the inventory of any agent A_i for contributing to the resolution of hotspots may differ between agents: These, for agent A_i are in $\mathbf{D}_i = \{0, 1, 2, \dots, \text{MaxDelay}_i\}$. We consider that these may be ordered by the preference of agent A_i to any such option, according to the function $\gamma(i): \mathbf{D}_i \rightarrow \mathbb{R}$. We do not assume that agents in $\mathbf{A} - \{A_i\}$ have any information about $\gamma(i)$. This represents the situation where airlines set own options and preferences for delays even in different individual own flights, depending on operational circumstances, goals and constraints. However, we expect that the order of preferences should be decreasing from 0 to MaxDelay_i , although, with a different pace for different agents.

Problem statement: Considering any two peers A_i , and A_j in the society (\mathbf{A}, \mathbf{E}), with $\mathbf{N}(A_i) - \{A_i\}$, these agents must select among the sets of available options \mathbf{D}_i and \mathbf{D}_j respectively, so as to increase their expected payoff w.r.t. their preferences on options $\gamma(i)$ and $\gamma(j)$, and resolve the DCB problem.

2.3 Collaborative Reinforcement Learning Algorithms

We now recall the proposed RL methods to deal with the multiagent joint DCB policy search problem. The key concept includes interactions between flights.

2.3.1 Independent Reinforcement Learners (IndLearners)

In an IndLearners framework, each agent learns its own policy independently and treats other agents as part of the environment.

Each local action-state function, Q_i , for agent A_i is calculated according to the local state, s_i , and the local strategy, str_i (i.e. the amount of delay for its own regulation),

Q_i is updated according to the temporal-difference error, as follows:

$$Q_i(s, str) = Q_i(s, str) + \alpha [Rwd_i(s, str) + \delta \max_{str'} Q_i(s', str) - Q_i(s, str)]$$

The reward received by the agent A_i takes into account only its local state and local strategy.

2.3.2 Edge-Based Collaborative Reinforcement Learners (EdgeBased)

Given two peer agents A_i and A_j connected by an edge in the coordination graph, the Q-function for these agents is denoted as $Q_{ij}(\mathbf{s}_{ij}, \mathbf{str}_{ij})$, where \mathbf{s}_{ij} denotes the joint state related to the set of the two agents A_i and A_j , and \mathbf{str}_{ij} denotes the joint strategy for these two agents.

The Q-learning update rule in this case is given by the following equation:

$$Q_{ij}(\mathbf{s}_{ij}, \mathbf{str}_{ij}) = Q_{ij}(\mathbf{s}_{ij}, \mathbf{str}_{ij}) + \alpha \left[\frac{Rwd_i(s_i, str_i)}{N(A_i)} + \frac{Rwd_j(s_j, str_j)}{N(A_j)} + \delta Q_{ij}(\mathbf{s}'_{ij}, \mathbf{str}^*_{ij}) - Q_{ij}(\mathbf{s}_{ij}, \mathbf{str}_{ij}) \right],$$

where, \mathbf{str}^* is the best known strategy for agents, and it is depicted directly from the agent's value function, $Q_i(s, str)$, which is calculated as the summation of local Q_{ij} values in its neighbourhood:

$\mathbf{str}^*_i = \operatorname{argmax}_{str_i} Q_i(s_i, str_i)$, and

$$Q_i(s_i, str_i) = \frac{1}{2} \sum_{j \in N(A_i)} Q_{ij}(\mathbf{s}_{ij}, \mathbf{str}_{ij}).$$

2.3.3 Agent-Based Collaborative Reinforcement Learners (AgentBased)

As in EdgeBased method, given two peer agents performing their trajectories, A_i and A_j , their joint Q-function is denoted succinctly $Q_{ij}(\mathbf{s}_{ij}, \mathbf{str}_{ij})$, where \mathbf{s}_{ij} and \mathbf{str}_{ij} denote the joint state and strategy, respectively, related to the two agents, as defined in the previous section. The update rule is then:

$$Q_{ij}(\mathbf{s}_{ij}, \mathbf{str}_{ij}) = Q_{ij}(\mathbf{s}_{ij}, \mathbf{str}_{ij}) + \alpha \left[\sum_{k \in \{i, j\}} \frac{Rwd_{ij}(\mathbf{s}_{ij}, \mathbf{str}_{ij}) + \delta Q_k(\mathbf{s}'_k, \mathbf{str}^*_k) - Q_k(\mathbf{s}_k, \mathbf{str}_k)}{N(k)} \right]$$

where, \mathbf{str}^*_k is the best known strategy for agent A_k in state \mathbf{s}'_k , k in $\{i, j\}$. Agents, compute their local Q-functions and their best local strategy as in the EdgeBased method.

2.3.4 Hierarchical Reinforcement Learning Approach (Hierarchical)

In order to make RL computationally efficient and maybe more effective to produce solutions in complex problems, we apply abstraction or generalization operators. The idea behind state abstraction is that, instead of working in the *ground (original) state space*, the decision maker usually finds solutions in the *abstract state space* much faster by treating groups of states as a unit by ignoring irrelevant state information.

The hierarchical collaborative RL framework comprises two levels: The ground level and an abstracted level at level L . The proposed hierarchical RL method consists of the following stages:

1. **Start with the original state space.** This is the ground representation at state space *State*. At this “ground” level the distance between consecutive time points is one time instant.

2. **Map *State* to an abstract-feature space $State_L$** , where $|State_L| \ll |State|$. This includes the abstraction of the state space so as to reduce the original space *State*.

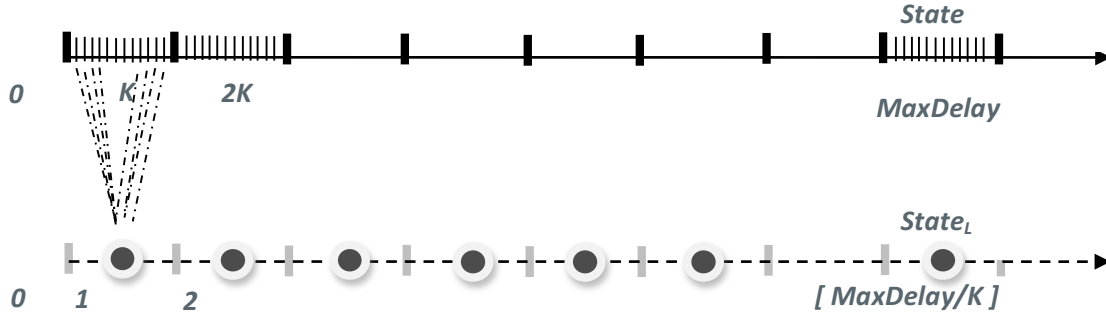


Figure 2. Obtaining the abstract space: Delay is partitioning into a number of K equidistant intervals.

As shown in Figure 2, all ground states corresponding to delays between consecutive time points in $[t, t+ts_L]$ are mapped to the same state in the abstract space.

3. **Solve MDP in $State_L$ space.**
4. **Map solution from abstract space $State_L$ to ground *State* space.** In this step we consider that states s_i in the ground set of states *State* that have been mapped to the same abstract state s_i^L in $State_L$ have the same Q^* values per agent and strategy, equal to the Q^* value computed by solving the MDP in the abstract space s_i^L .
5. **Solve MDP in the original *State* space.** One of the multi-agent RL methods used is applied to refine the solution in the abstract state.

Details on the above algorithms are provided in D3.2.

2.3.5 Reward Function

For all the above-mentioned methods, the reward function of agents, as specified in D3.2 is as follows:

The *local reward* of an agent A_i , denoted Rwd_{A_i} , is the reward that the agent gets by executing its own trajectory in a specific joint state with any agent executing a trajectory in $Traffic(A_i)$, according to the sectors' capacities, and the joint strategy of agents. The *joint reward*, denoted by Rwd_{Ag} , for a set of peers A_g specifies the reward received by agents in A_g by executing their trajectories in their joint state, according to their joint strategy.

The reward Rwd_{Ag} for an subset Ag of \mathbf{A} depends on agents participation in (contribution to) hotspots occurring while executing their trajectories according to their joint strategy \mathbf{str}_{Ag}^t in their joint state \mathbf{s}_{Ag}^t , i.e. according to their decided delays. Formally:

$$Rwd_{Ag}(\mathbf{s}_{Ag}^t, \mathbf{str}_{Ag}^t) = \lambda_1 * C(\mathbf{s}_{Ag}^t, \mathbf{str}_{Ag}^t) + \lambda_2 * DC(\mathbf{s}_{Ag}^t, \mathbf{str}_{Ag}^t)$$

where, $C(\mathbf{s}_{Ag}^t, \mathbf{str}_{Ag}^t)$ is a function that depends on the participation of agents in hotspots while executing their joint strategy in their joint state, and $DC(\mathbf{s}_{Ag}^t, \mathbf{str}_{Ag}^t)$ is a function aggregating agents' strategic delay costs.

The parameters λ_1 and λ_2 are used for balancing between the number of hotspots and delays experienced by agents towards reaching a solution: Zero hotspots and the minimum possible delay per agent.

In the DCB problem, both functions $C(s_{Ag}^t, \text{str}_{Ag}^t)$ and $DC(s_{Ag}^t, \text{str}_{Ag}^t)$ represent costs: We have chosen $C(s_{Ag}^t, \text{str}_{Ag}^t)$ to depend on the total duration of the time interval in which agents fly over a congested sector. This is multiplied by 81 which is the average strategic delay cost per minute (in Euros) in Europe when 92% of the flights do not have delays [Cook et al, 2015]. If there is not any congestion, then this is a large positive constant that represents the reward agents get by not participating in any hotspot.

The actual form of $C(s_{Ag}^t, \text{str}_{Ag}^t)$ is as follows:

$$C(s_{Ag}^t, \text{str}_{Ag}^t) = \begin{cases} -TDC * 81 & \text{if } TDC > 0 \\ \text{PositiveReward} & \text{if } TDC = 0 \end{cases}$$

TDC is the total duration in hotspots for agents in A_g . The first case holds when there are hotspots in which agents participate (thus, the total duration in hotspots, TDC, is above 0), while the second case holds when agents do not participate in hotspots.

The $DC(s_{Ag}^t, \text{str}_{Ag}^t)$ component of the reward function corresponds to the *strategic delay cost* when flights delay at gate. In our implementation, this depends solely on the minutes of delay and the aircraft type, as specified in [Cook et al, 2015].

As such, the actual form of this function is as follows:

$$DC(s_{Ag}^t, \text{str}_{Ag}^t) = - \sum_{A \in A_g} \text{Delay}_A * \text{StrategicDelayCost}(\text{Delay}_A, \text{AircraftType}(A))$$

Where Delay_A is the delay imposed to the agent A and $\text{StrategicDelayCost}$ is a function that returns the strategic delay cost given the aircraft type of agent A and its delay.

Notice however that in the general case the function $DC(s_{Ag}^t, \text{str}_{Ag}^t)$ could be taking into account broader airline-specific strategic policies and considerations regarding flight delays.

2.3.6 Exploration - Exploitation Scheme

All algorithms utilize the ϵ -greedy policy as combination of exploration and exploitation. At every time step, each agent makes a decision based on ϵ , i.e. the probability to choose randomly or, based on experience, act in a greedy way. The parameter epsilon is initialized to 0.9 and is diminished by 0.01 every 80 episodes. When the threshold of 0.001 is reached, epsilon is considered zero. This results at a policy of pure exploitation after 7200 episodes, where only greedy actions are chosen.

3 Evaluation Results

3.1 Experimental settings

3.1.1 Evaluation cases

To evaluate the proposed methods, we have constructed evaluation cases of varying complexity/difficulty, by inspecting problem parameters in conjunction to the average delay considering CFMU reported regulations.

Each case corresponds to a specific day of 2016 above Spain and its complexity/difficulty has been determined by means of the number of flights involved, the average number of interacting flights per flight (which is translated to the average degree for each agent in the coordination graph, connecting that agent with its peers), the maximum delay imposed to flights for that day to resolve DCB problems according to CFMU data, the average delay for all regulated flights according to CFMU data, and the number of hotspots in relation to the number of flights participating in these hotspots.

The specific method used for constructing these evaluation cases is detailed in Section 3.2.

In Table 1 we can see the different cases named by the day in which they occurred.

Specifically, Table 1 specifies per case:

Number of flights: The number of flights for that particular day above Spain.

Average Degree in Coordination Graph (min/max): This indicates in average the traffic (i.e. the number of interacting flights) for each of the agents (flights) in each evaluation case. It is expected that as the coordination graph becomes more “dense”, i.e. as the average degree increases, the problem becomes more computationally demanding.

Min/Max indicates the minimum and the maximum degree reported in the coordination graph per evaluation case, while ignoring zeros.

MaxDelay (according to CFMU data): This is the MaxDelay that is allowed to all flights. It is equal to the maximum delay reported by CFMU data for that particular day.

Average Delay (according to CFMU data): This is the average delay for regulated flights reported by CFMU data for that particular day.

Number of regulated flights (type C – according to CFMU data): These are the number of flights with regulations of type C (i.e. delays due to DCB problems) reported by CFMU data for that particular day.

It must be noted that regulated flights by CFMU leave a large number of hotspots unresolved in any of the cases considered (only 1 or 2 hotspots are resolved per case.).

Number of hotspots (number of flights): It indicates the number of hotspots to which a flight participates in this evaluation case, together (in parenthesis) with the number of flights that participate to that number of hotspots (each flight may participate in different combinations of hotspots). This is also an indication of problems' difficulty: Of course, this difficulty may also depend on other factors such as the duration of flights to hotspots, the excess on capacity for these hotspots etc. It is not the purpose of this deliverable to delve into these issues, but we do need to indicate major differences among evaluation cases.

Evaluation case	Number of Flights (Agents)	Average Degree in Coordination Graph (non-zero min/max)	MaxDelay (according to CFMU data)	Average Delay (according to CFMU data)	Number of Regulated Flights (type C) (according to CFMU data)	Number of hotspots (number of flights)
Aug4	5544	6.41 (17-120)	66	12.41	179	33 (853)
Aug7	5868	8.03 (23-121)	112	17.54	475	42 (1104)
Aug10	5500	5.92 (19-125)	59	15.37	402	27 (759)
Aug13	6000	10.89 (22-105)	147	16.02	434	53 (1460)
Jul2	5572	6.39 (29-107)	80	17.89	521	29 (778)
Jul10	5824	9.98 (21-175)	175	17.35	305	51 (1320)
Jul12	5408	5.84 (21-95)	95	18.55	281	28 (820)
Jun5	5348	6.77 (20-111)	84	14.99	162	32 (803)
Sep2	5498	5.41 (21-112)	88	13.95	165	27 (754)
Sep3	5788	5.24 (18-77)	61	14.41	297	26 (783)

Table 1: Evaluation cases used for evaluating the proposed methods.

3.1.2 Tuning the reward function

To apply the proposed methods in experimental settings, we need to balance between the two reward constituents, by tuning the values of λ_1 and λ_2 . To do that, we have set $\lambda_1=1$ and experimentally configured the value for λ_2 . The table below shows experiments from one of our evaluation cases, that of July 2, which – according to the average of CFMU delays imposed- seems to be one of the “difficult” cases, using one of our methods: This evaluation case has one of the largest average delay imposed to regulated flights. The results are representative of the other methods and show that setting λ_2 to a value which is greater than 30, the methods need more time (more exploration rounds) to converge, but with no remarkably better results (i.e. the difference in the average delay achieved for $\lambda_2=50$ is +0.03), while for larger values (more than 50) methods become unstable (i.e. do not converge to a specific joint policy for all agents). Nevertheless, for values less than 10, the situation as far as the average delay is the same (the difference in the average delay achieved compared to the case where

$\lambda_2=50$ is -0.1), while the number of regulated flights show to increase. Therefore, we decided that the value 20 is the one that should be used for λ_2 , i.e. for balancing between the “cost” of participating in hotspots and the strategic delay cost when there are not hotspots.

Evaluation case	Number of Resulting Hotspots	Number of Regulated Flights	Average Delay for regulated flights (according to CFMU data) (min/max)	Average Delay for regulated flights (IndLearners)	Comments
$\lambda_2=1$	0	727	17.89 (1-80)	15.06	
$\lambda_2=10$	0	731.2	17.89 (1-80)	14.48	
$\lambda_2=20$ (final)	0	724	17.89 (1-80)	14.58	
$\lambda_2=50$	0	731.6	17.89 (1-80)	14.61	50% more exploration was needed to achieve convergence
$\lambda_2=100$	0-1	716	17.89 (1-80)	15.44	50% more exploration was utilized, convergence not always achieved

Table 2: Experiments with different λ_2 values for evaluation case “July2” with the IndLearners method: Results show that a proportion of 1:20 for λ_1 : λ_2 is the most suitable one, as it balances effectively between low average delay and low number of regulated flights, while methods converge with less exploration rounds.

3.2 Construction of the evaluation cases

The first step towards constructing an evaluation case for a chosen day is to collect all the Flight Plans for that day as provided by the Spanish Operational Data source. Each Flight Plan may be associated to multiple Flight Plan Messages. According to the domain experts, we construct evaluation cases using the Flight Plan specified in the last message arriving before takeoff. In order to identify it, the timestamp of the Message arrival is compared to the Estimated Entry Date and Time to FIR (HFIR). Some Flight Plans span in two consecutive days, for example a flight could take off before and land after midnight. These Flight Plans are considered for both days. In addition, the model of the aircraft is stored for the calculation of strategic delay costs. Finally, flights are distinguished between commercial and non-commercial, using their ID and the Flight Rules (FLRL) column. Delays cannot be imposed to non-commercial flights (e.g. military), although all flights participate to the evaluation case and distinguished by a flag value.

After collecting the Flight Plans described above, we cross-check them with the CFMU Dataset. In order to calculate hotspots, it is necessary to consider all those flights the NM has information about, thus the Flight Plans that do not correspond to a CFMU entry are dropped. The cross identification is achieved by utilizing the ID of the flight, the departure and destination airport and the Initial Off-Block

Time (IOBT). Moreover, delays imposed from the NM to resolve hotspots occurring inside the Spanish Airspace, are identified.

At this point, the Flight Plans contain a trajectory crossing air volumes. This sequence is exploited to compute the series of active sectors that each flight crosses- depending on the open airspace configurations, together with the entry and exit time for each of these sectors. For the first (last) sector of the flight, where the departure (resp. arrival) airport resides, the entry (resp. exit) time is the departure (resp. arrival) time. However, there may exist flights that cross the airspace but do not depart and/or arrive in any of the sectors of our airspace: In that case we only consider the entry and exit time of sectors within the airspace of our interest.

Therefore, air volumes have to be converted to sectors, in order to be attributed with capacity and determine the occurring hotspots. Airspace sectorization changes frequently during the day, given different operational conditions and needs. To take into account the different sectorizations, we apply the following procedure:

1. for each Flight Plan
2. for each Possible Delay
3. for each Air Volume
4. for each Sector corresponding to the Air Volume
5. for each Configuration corresponding to the Sector
6. check if Configuration is active while the Air Volume is crossed
7. end of loops

This procedure, exploiting the Airspace Structure – i.e. the sector configurations and sectorization- as well as the mappings from air volumes to sectors, all provided by the corresponding DART datasets, “translates” air volumes crossed by trajectories to sectors.

It must be noticed that (a) given the delay imposed to a trajectory, sectors crossed may vary, due to the changing configurations; (b) this may result to a number of alternative representations of a single trajectory (each representation crossing a different set of sectors) one for each possible delay.

As an example, consider a trajectory T that crosses the volume R, staying inside the volume for 31 minutes:

$$T = [(R, 10:59, 11:31)]$$

The translation to sector (or sectors) may vary when delays are imposed. For delay equal to zero the result could be:

$$T = [(S_1, 10:59, 11:00), (S_2, 11:00, 11:30), (S_3, 11:30, 11:31)]$$

Meaning that the active configuration changes at 11:00 and then changes again at 11:30. Imposing one minute of delay will result to the following trajectory, eliminating the first sector completely:

$$T = [(S_2, 11:00, 11:30), (S_3, 11:30, 11:32)]$$

Finally, imposing 31 minutes of delay would eliminate the second sector as well:

$$T = [(S_3, 11:30, 12:02)]$$

This process results in Flight Plans consisting of Sectors and a list of all the necessary Sectors with their capacity. This information is consumed to create an evaluation case as an input to the algorithms.

The resulting files contain all the necessary information for the experiments. The first line of the evaluation case contains all the vital hyper parameters:

1. The dimensions of the Sector Grid;
2. The number of flights (i.e. participating agents);
3. The size of the counting period for computing demand evolution;
4. The counting step for computing demand evolution;
5. The maximum possible delay (derived from the corresponding maximum delay from the CFMU dataset);
6. The total duration of the experiment (here 24 hours);
7. The learning rate α (set to 0.01 for all methods);
8. The discount factor δ (set to 0.99 for all methods);
9. The hyper parameters $\lambda_1:\lambda_2$ (set to 1:20 for all methods).

The second line contains the capacities of all needed sectors that are activated and crossed during the 24 hours considered. The last two sectors are virtual and have a capacity set to 500.

Each of the rest of the lines contains the information of one flight:

1. Flight ID;
2. The trajectory for each possible delay (takeoff time, sectors crossed and time spent in each one);
3. The aircraft model;
4. A binary value representing if this flight is a commercial one or not;
5. Maximum delay for that flight.

3.3 Evaluation criteria

To measure the efficiency of the methods and the quality of solutions achieved, we have specified qualitative criteria (metrics) as follows:

- **Learning curves** of all methods showing the computational efficiency of the methods: These curves show per round of methods' application the average delay for all flights in the evaluation case, while agents chose their policies (i.e. as they learn the regulation-policy to be applied). As algorithms converge to solutions, the number of hotspots should be reduced and eventually reach to zero, while the average delay should be reduced, signifying the computation of a solution. Therefore, the speed of reaching that point (zero hotspots) and the round at which methods stabilize agents' joint policy (remaining to zero hotspots and to a specific value for flights' average delay - without oscillating between non-solutions and/or solutions, and/or different average delay values) signify the computational efficiency of the method to reaching solutions. Of course, in case that a method cannot reach a solution to a

specific case it may converge to a joint policy whose application results to more than zero hotspots.

- **Number of regulated flights.**
- **Average delays of flights:** We report on (a) the average delay considering only the regulated flights (in tables with results), as well as (b) the average delay for all flights (in learning curves). In any case, a clarification on the computation of average delays is provided.
- **Distribution of delays to flights:** To show how delays are distributed to flights, we provide histograms showing the number of flights with (a) 0-9 minutes of delay, (b) 10-29 minutes of delay, (c) 30-59 and (d) 60-MaxDelay minutes of delay. Of course, if it happens that MaxDelay is less than 60, 30 etc., the histogram does not provide data for the corresponding slot of delays.

Evolution of demand: To further delve into the quality of solutions provided by the proposed methods, we provide for highly-demanded sectors the evolution of demand per time period at the initial state (i.e. at the problem state) and the evolution of demand per time period at the solution state (i.e. at the state where each method has converged to a joint policy for agents – and thus, flights have been regulated).

All measurements provided result by averaging the measurements recorded by 5 independent experiments per case and method.

3.4 Experimental results.

The following tables provide results for all methods and evaluation cases.

Evaluation case	Number of Regulated Flights (type C) (according to CFMU data)	Number of Resulting Hotspots (All methods)	Number of Regulated Flights (IndLearners)	Number of Regulated Flights (EdgeBased)	Number of Regulated Flights (AgentBased)	Number of Regulated Flights (Hierarchical)	Comments
Aug4	179	0	672.2	<u>609.8</u>	665.7	339.2	
Aug7	475	0	986	<u>933</u>	994	493.8	
Aug10	402	0	620.2	<u>614.4</u>	635	199.5	
Aug13	434	0	1231	<u>1185.2</u>	1249.6	876	
Jul2	521	0	724	<u>719.4</u>	727.8	424.5	2 out of 5 experiments solved with hierarchical approach
Jul10	305	0	1066	<u>1017</u>	1053.2	569.2	

Jul12	281	0	618	577.4	<u>573.2</u>	187.8	
Jun5	162	0	649.6	<u>581.8</u>	637.5	234.4	
Sep2	165	0	559.6	<u>458.8</u>	473.8	186.4	
Sep3	297	0	728.8	<u>647.8</u>	680.5	393	

Table 3: The number of regulated flights per method and evaluation case: All methods resolve DCB problems, resulting to 0 hotspots per evaluation case. Bold indications show the best results, while the underlined ones show the second best.

Regarding the number of regulated flights, it must be pointed out that according to CFMU data, CFMU regulated flights do not resolve all DCB problems: I.e. even if we impose regulations to the CFMU regulated flights we still have a large number of hotspots. This is also indicated in Section 4 in a specific evaluation case, as representative of all cases. Therefore, while the proposed methods do increase the number of regulated flights in all evaluation cases, the imposed regulations result to 0 hotspots. Among the methods, the Hierarchical one, reduces considerably (more than 30% in all cases) the number of regulated flights. Indeed, the number of regulated flights resulting from the Hierarchical methods are comparable to the CFMU regulated flights (with the later leaving a large number of hotspots unresolved). Beyond this remarkable result from the Hierarchical method, the EdgeBased method manages to have the less number of regulated flights among the rest of the methods (except in the Jul2 evaluation case).

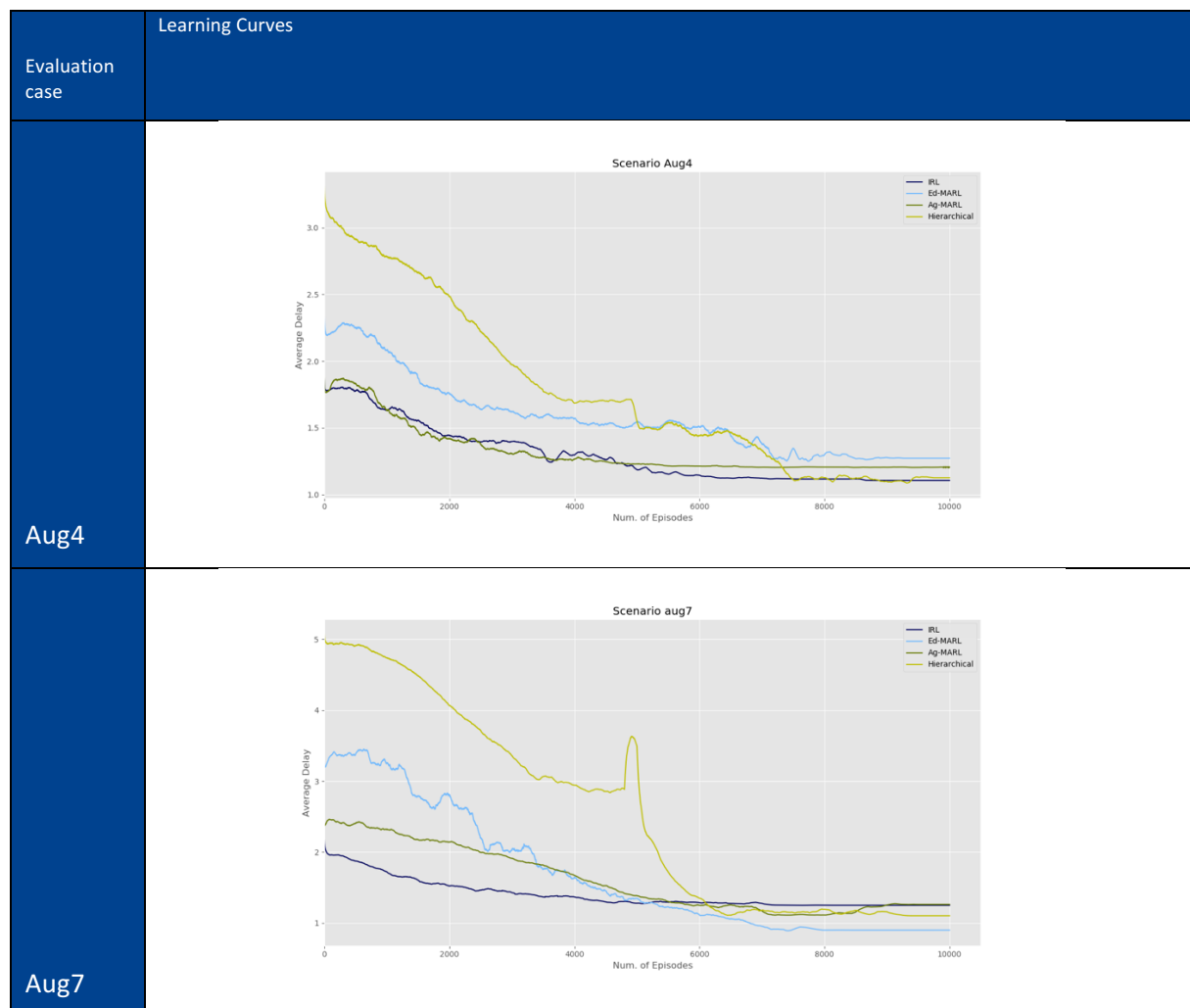
Evaluation case	Average Delay for regulated flights (according to CFMU data) (min/max)	Average Delay for regulated flights (IndLearners)	Average Delay for regulated flights (EdgeBased)	Average Delay for regulated flights (AgentBased)	Average Delay for regulated flights (Hierarchical)
Aug4	12.41	9.1	11.55	9.9	18.38
Aug7	17.54	7.43	5.65	7.44	13.03
Aug10	15.37	3.9	4.3	4.69	9.18
Aug13	16.02	8.08	6.78	7.7	12.27
Jul2	17.89	14.58	14.93	14.68	25.46
Jul10	17.35	7.11	4.7	5.91	8.8
Jul12	18.55	3.73	2.89	3.17	5.68
Jun5	14.99	6.33	4.24	5.11	9.41
Sep2	13.95	6.15	3.28	4.54	7.3
Sep3	14.41	7.64	11.5	10.88	19.98

Table 4: The average delays considering all regulated flights, per method and evaluation case, compared to CFMU average delays per evaluation case: All methods manage to considerably reduce the average delays for the regulated flights, compared to CFMU values. Bold in the Hierarchical column indicate the cases where the

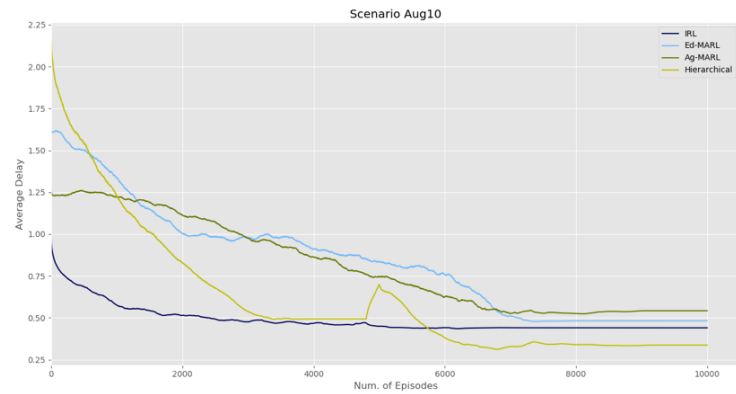
average delay reported is greater than that of CFMU, while bold indications in other columns indicate the best results.

Regarding the average delay to all regulated flights, as results reported in Table 4 indicate, all methods – except the Hierarchical one- manage to reduce considerably the CFMU imposed average delay in all cases. This is a remarkable result for all methods. However, the Hierarchical method increases considerably the average delay for regulated flights in three cases (indicated in bold in Table 4.

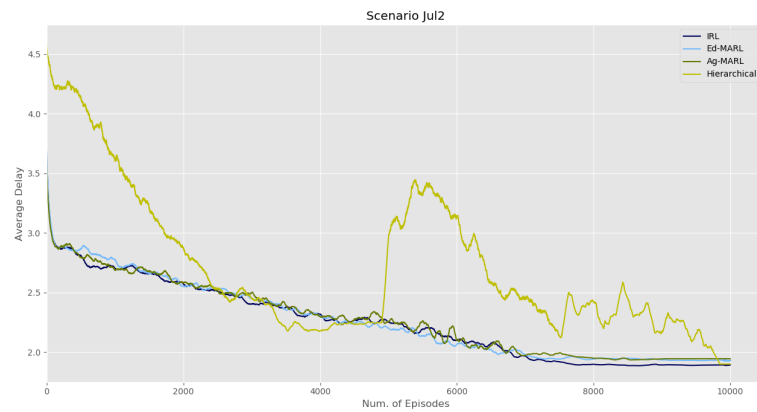
The EdgeBased approach seems to perform better than the other methods in all cases, except in three of them where IndLearners are more effective in reducing the delay. It should be noticed that in all cases the difference between the best and the second best average delays is quite large.



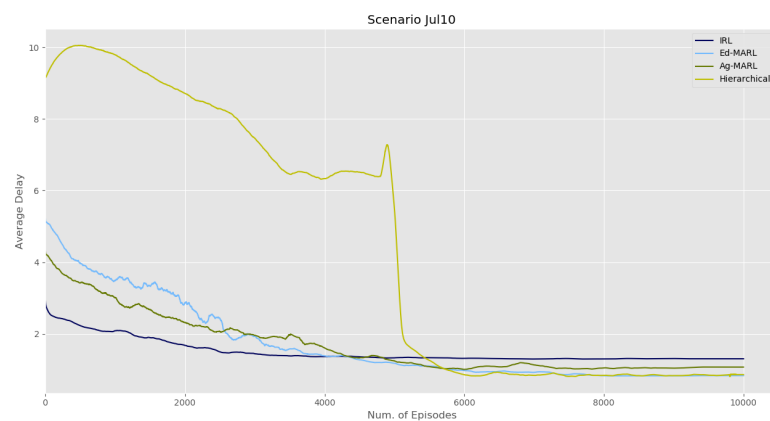
Aug13



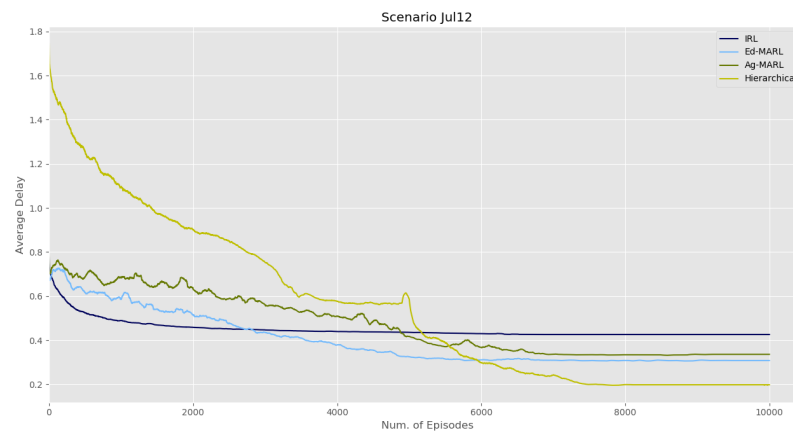
Jul2



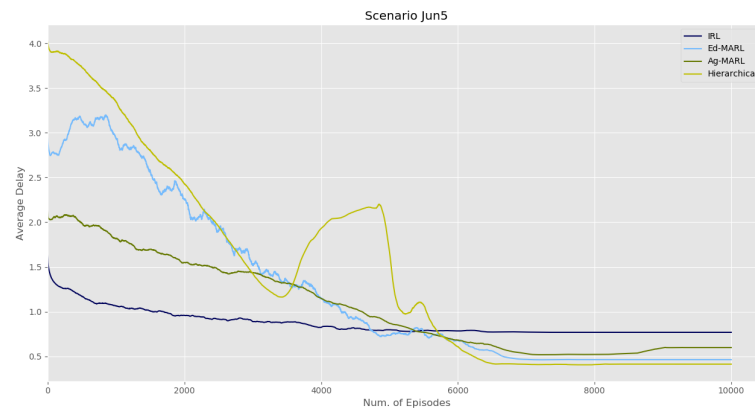
Jul10



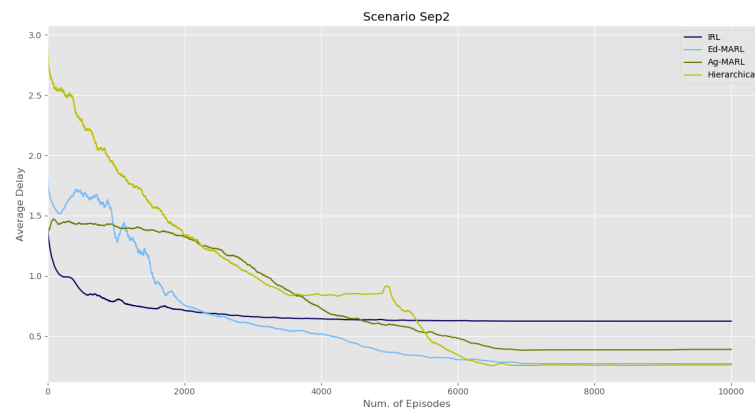
Jul12



Jun5



Sep2



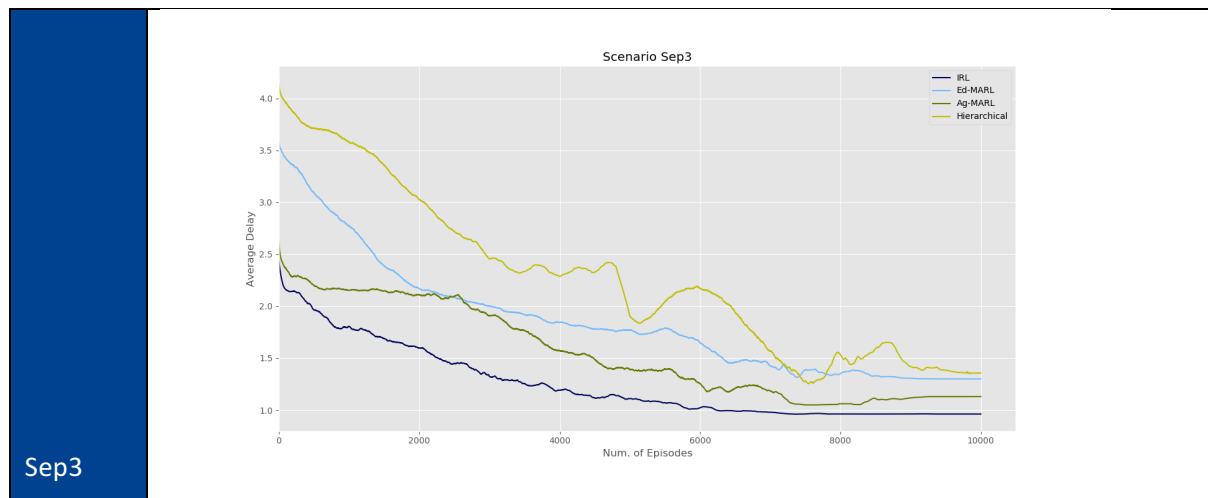
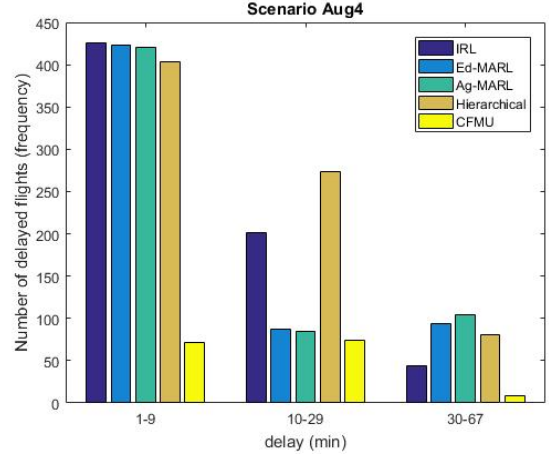
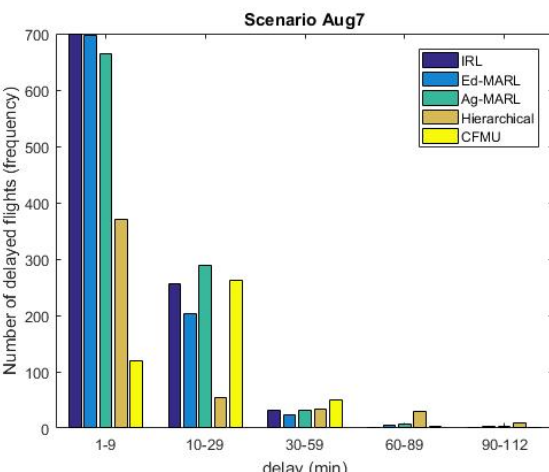
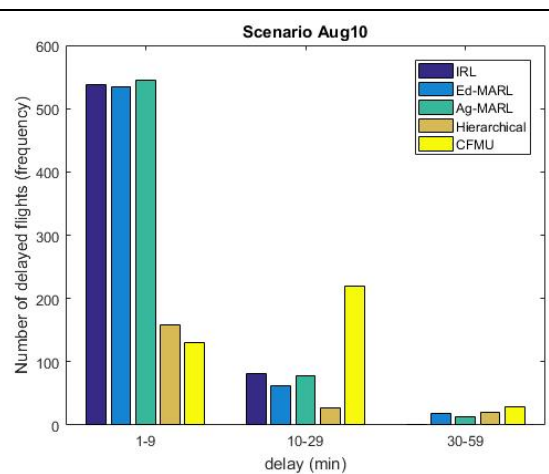


Table 5: The learning curves of all methods per evaluation case, showing how methods manage to learn agents (flights) joint policies to resolve DCB problems, resulting to 0 hotspots, while reducing the average delays for all (regulated or not) flights. The x axis corresponds to the episodes of methods, while the y axis to the average delay reported for all flights.

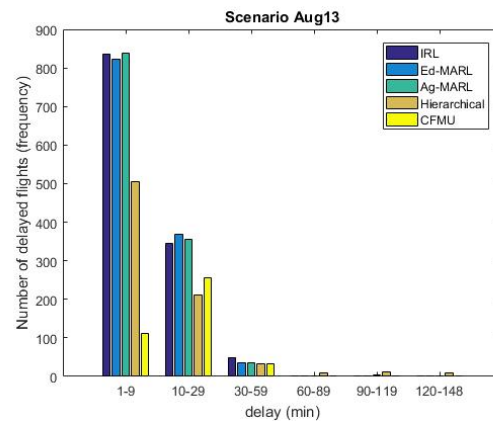
Results in Table 5 show that methods manage to converge effectively, given that until episode 7200 they do intervene exploitation with exploration. However, it seems that there are cases where methods can converge even earlier. Specifically, IndLearners manage to converge or at least approximate effectively the convergence point even earlier than episode 6000, except in one case – Jul2. All methods converge effectively after exploration round 7200, approaching the converge point. It should be noticed that “convergence” does not imply solving the problem: A method may converge to a joint policy, imposing regulations to flights that when applied may still imply DCB problems. Fortunately, this happens in one case and only for the Hierarchical method.

Also, for Jul2, which it seems to be the hardest case for all methods, all methods converge quite late (i.e. after a large number of episodes). For that evaluation case, the Hierarchical method fails in 3 out of the 5 methods, which is reflected to the learning curve that results by averaging the results per episode for these 5 experiments.

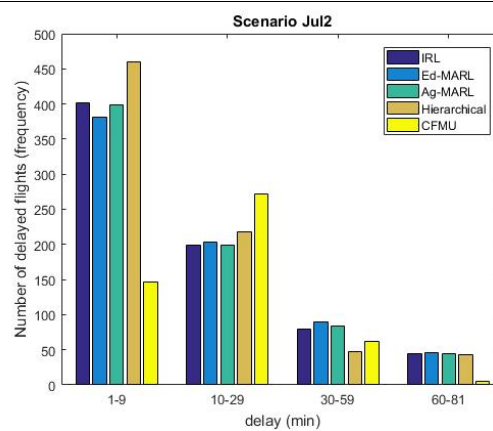
Finally, it must be noticed that the Hierarchical method manages to achieve the lowest average delay among all flights, which is explained by the low number of flights that it manages to regulate (although with higher – in average – delay for regulated flights).

Delays Distributions																																					
Evaluation case																																					
Aug4	<div><p>Scenario Aug4</p><table><thead><tr><th>delay (min)</th><th>IRL</th><th>Ed-MARL</th><th>Ag-MARL</th><th>Hierarchical</th><th>CFMU</th></tr></thead><tbody><tr><td>1-9</td><td>430</td><td>425</td><td>425</td><td>405</td><td>70</td></tr><tr><td>10-29</td><td>205</td><td>85</td><td>85</td><td>275</td><td>75</td></tr><tr><td>30-67</td><td>45</td><td>95</td><td>105</td><td>80</td><td>10</td></tr></tbody></table></div>	delay (min)	IRL	Ed-MARL	Ag-MARL	Hierarchical	CFMU	1-9	430	425	425	405	70	10-29	205	85	85	275	75	30-67	45	95	105	80	10												
delay (min)	IRL	Ed-MARL	Ag-MARL	Hierarchical	CFMU																																
1-9	430	425	425	405	70																																
10-29	205	85	85	275	75																																
30-67	45	95	105	80	10																																
Aug7	<div><p>Scenario Aug7</p><table><thead><tr><th>delay (min)</th><th>IRL</th><th>Ed-MARL</th><th>Ag-MARL</th><th>Hierarchical</th><th>CFMU</th></tr></thead><tbody><tr><td>1-9</td><td>700</td><td>700</td><td>670</td><td>370</td><td>120</td></tr><tr><td>10-29</td><td>260</td><td>205</td><td>290</td><td>55</td><td>265</td></tr><tr><td>30-59</td><td>30</td><td>25</td><td>35</td><td>40</td><td>50</td></tr><tr><td>60-89</td><td>10</td><td>10</td><td>10</td><td>30</td><td>10</td></tr><tr><td>90-112</td><td>5</td><td>5</td><td>5</td><td>10</td><td>10</td></tr></tbody></table></div>	delay (min)	IRL	Ed-MARL	Ag-MARL	Hierarchical	CFMU	1-9	700	700	670	370	120	10-29	260	205	290	55	265	30-59	30	25	35	40	50	60-89	10	10	10	30	10	90-112	5	5	5	10	10
delay (min)	IRL	Ed-MARL	Ag-MARL	Hierarchical	CFMU																																
1-9	700	700	670	370	120																																
10-29	260	205	290	55	265																																
30-59	30	25	35	40	50																																
60-89	10	10	10	30	10																																
90-112	5	5	5	10	10																																
Aug10	<div><p>Scenario Aug10</p><table><thead><tr><th>delay (min)</th><th>IRL</th><th>Ed-MARL</th><th>Ag-MARL</th><th>Hierarchical</th><th>CFMU</th></tr></thead><tbody><tr><td>1-9</td><td>540</td><td>535</td><td>545</td><td>160</td><td>130</td></tr><tr><td>10-29</td><td>85</td><td>65</td><td>80</td><td>30</td><td>220</td></tr><tr><td>30-59</td><td>20</td><td>20</td><td>20</td><td>25</td><td>30</td></tr></tbody></table></div>	delay (min)	IRL	Ed-MARL	Ag-MARL	Hierarchical	CFMU	1-9	540	535	545	160	130	10-29	85	65	80	30	220	30-59	20	20	20	25	30												
delay (min)	IRL	Ed-MARL	Ag-MARL	Hierarchical	CFMU																																
1-9	540	535	545	160	130																																
10-29	85	65	80	30	220																																
30-59	20	20	20	25	30																																

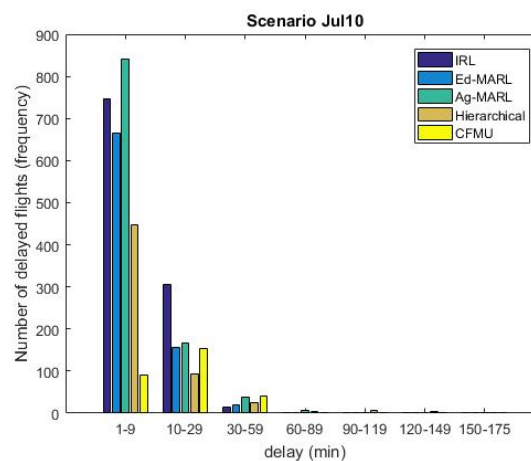
Aug13



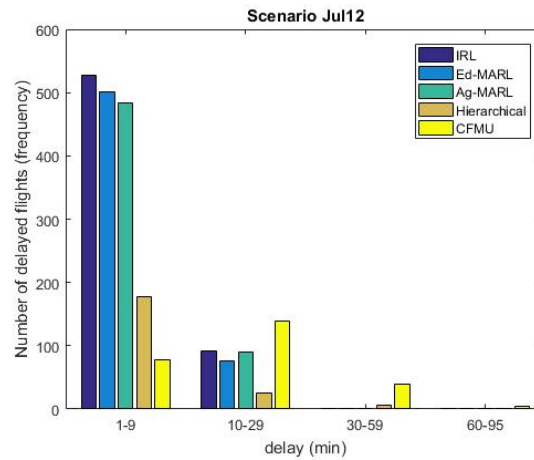
Jul2



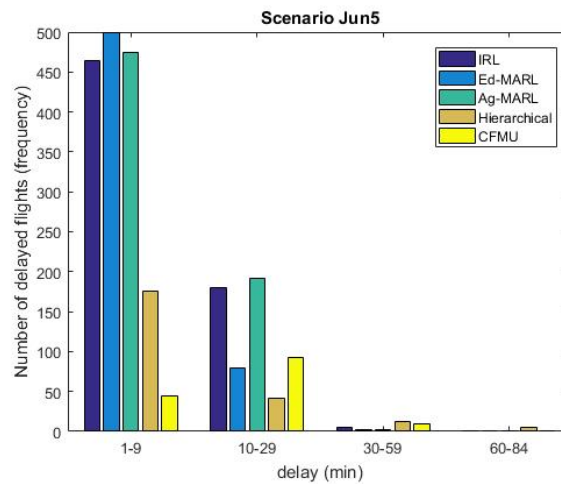
Jul10



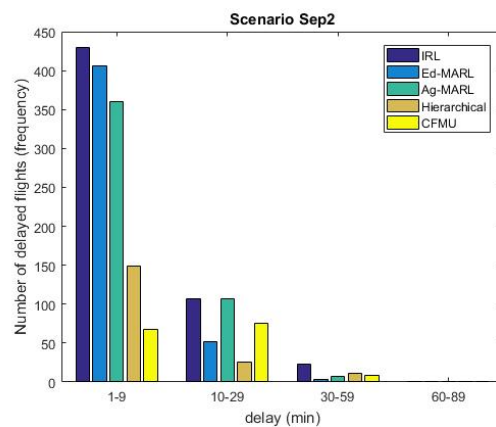
Jul12



Jun5



Sep2



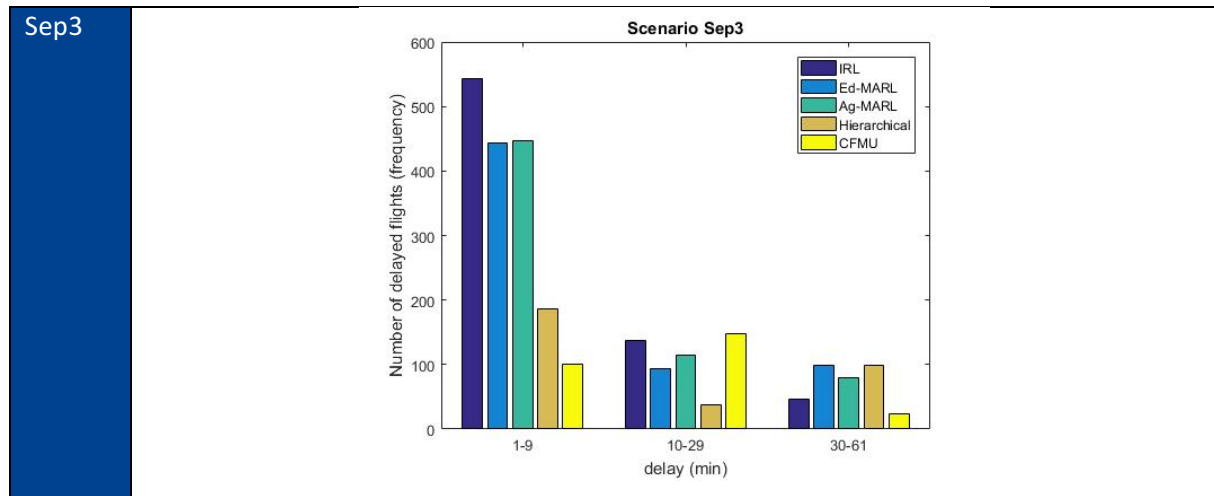
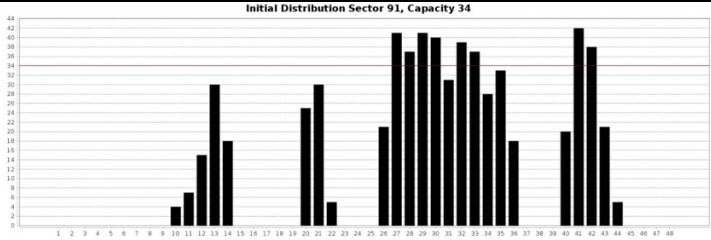
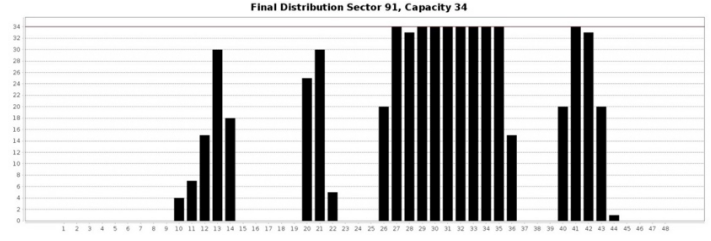
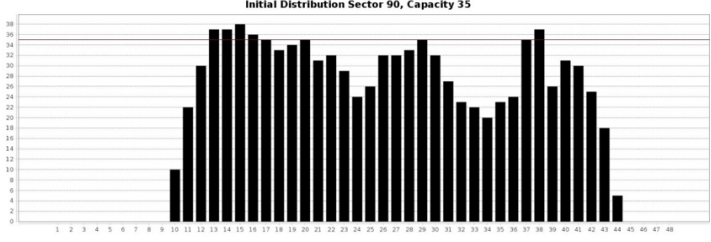
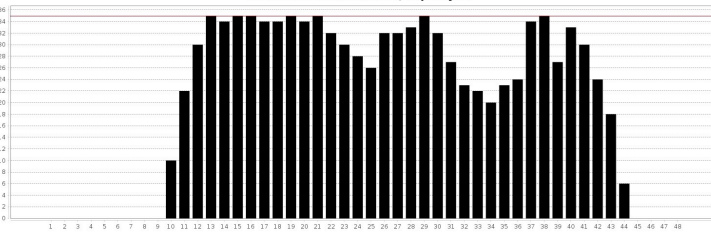
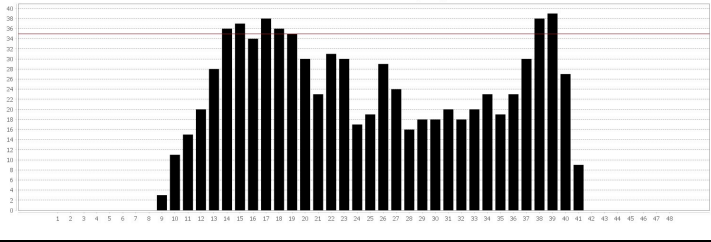


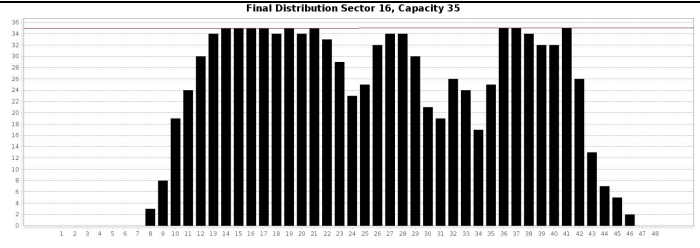
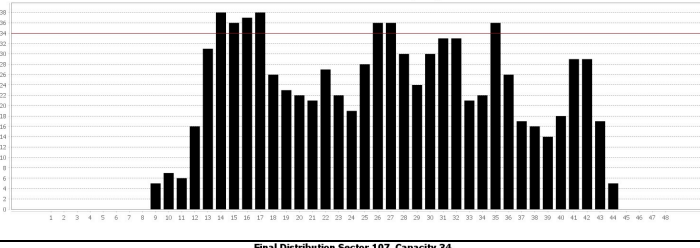
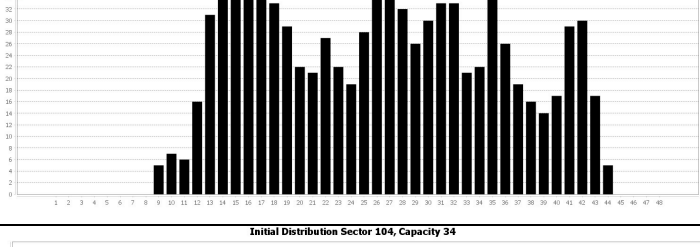
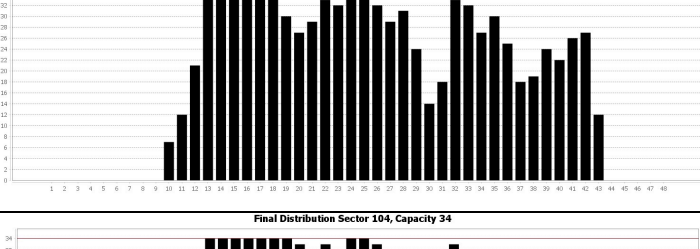
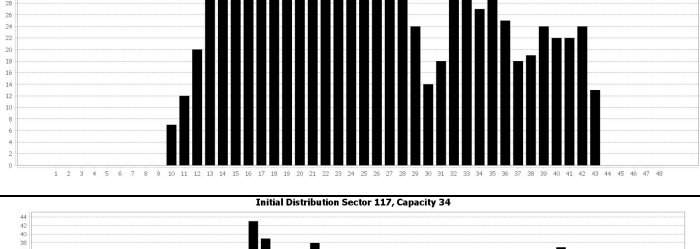
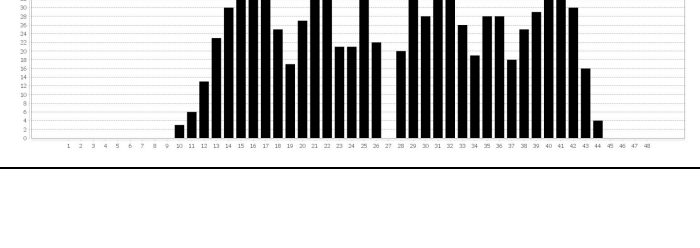
Table 6: The distribution of delays to flights by all methods and CFMU, and per experimental case. The x axis shows the delay imposed while the y axis corresponds to the number of flights. Notice that the maximum delay differs between evaluation cases.

Table 6 provides evidence on the fairness of all methods: Indeed, this is an inherent feature of all methods, given that each of the agents – in collaboration with its neighbours in the coordination graph (i.e. those that correspond to interacting flights due to traffic)- decides on own regulations towards resolving the DCB problems in which it participates. According to the reward function that each agent evaluates independently from others, it aims to reduce hotspots and the delay imposed to it. This is shown in Table 6, given that the number of flights with delays are reduced drastically, while moving from small to large delays. This happens in all cases. Notably, this happens in a more effective way for IndLearners, and EdgeBased methods, rather than for the Hierarchical and AgentBased methods.

Table 6 provides further comparison of distribution of delays from agent-based methods and CFMU: In all cases – as noticed above- agent-based methods assign delays to more flights than CFMU. It must further be noticed that delays imposed by CFMU to the majority of the flights are within the range of 10 to 30 minutes. In very few cases, CFMU imposes delays until 60 minutes to a considerable number of flights (e.g Aug10, Jul2, Jul12). However, given the fact that CFMU regulations do not resolve but 1 or 2 hotspots per evaluation case, it seems that delays from 1-9 minutes are not preferable or are within the margins of ATM system's tolerance. In any case, imposing delays from 1-10 minutes to more flights and 10-29 minutes delays to less flights than those of CFMU, may result to resolving all hotspots.

Evaluation case	Initial and Final Evolution of Demand Per Period (for the most demanded sectors)
Aug4 Initial	
Aug4 IndLearners	
Aug7 Initial	
Aug7 IndLearners	
Aug10 Initial	

Aug10 IndLearners	
Aug13 Initial	
Aug13 IndLearners	
Jul2 Initial	
Jul2 IndLearners	
Jul10 Initial	

Jul10 IndLearners	
Jul12 Initial	
Jul12 IndLearners	
Jun5 Initial	
Jun5 IndLearners	
Sep2 Initial	

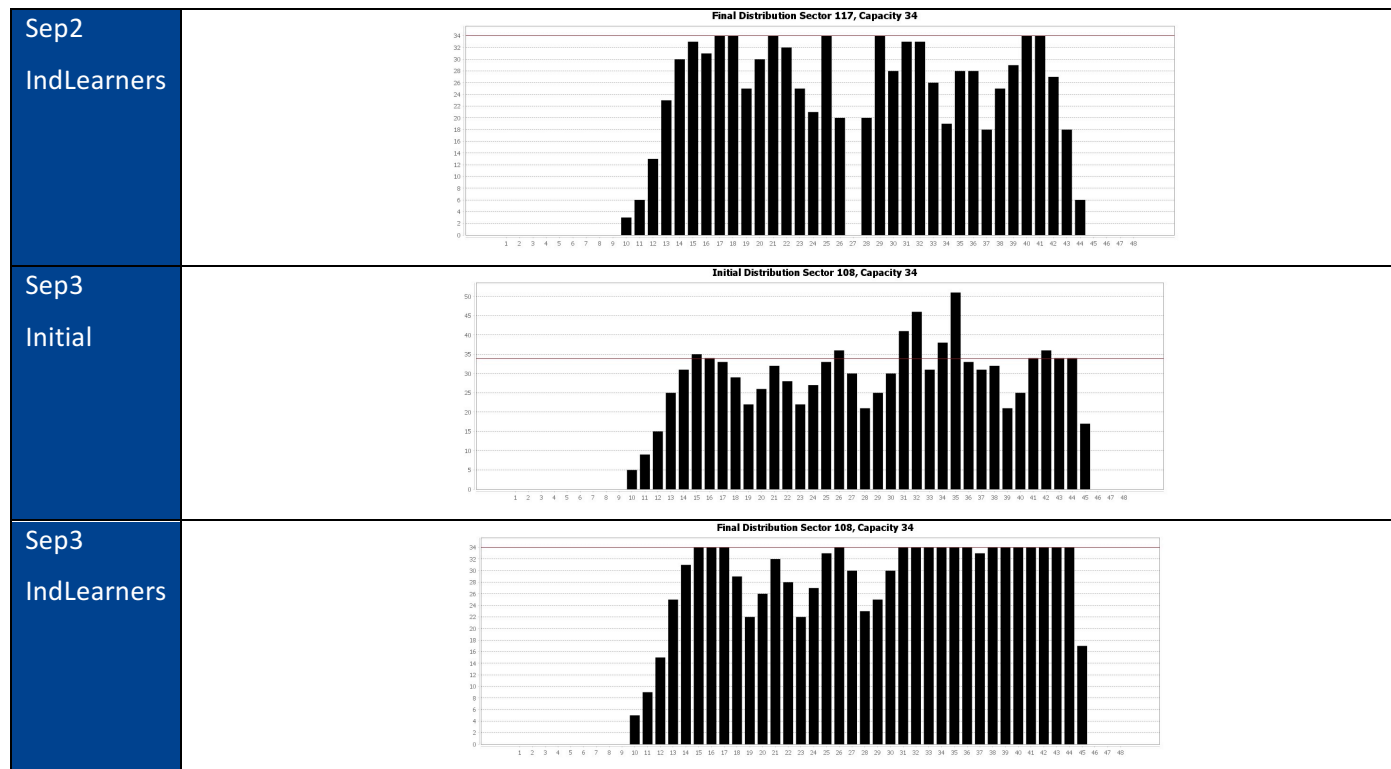


Table 7: The evolution of demand for the most demanded sector per evaluation case (a) in the initial problem state, and (b) in the solution proposed by the IndLearners method. The x axis shows the periods for measuring demand according to the hourly counting period metric (60' window with a step of 30'), while the y axis shows the demand. The red line indicates the capacity for each sector. Thus, any bar above that line indicates excess in capacity (hotspot).

Finally, Table 7 shows the evolution of demand in different periods for all cases in (a) the initial problem, and (b) after imposing the regulations decided by the IndLearners method. Results from the other methods are similar (actually very close to those presented by the IndLearners) so we did not include them here.

As results show, methods do “push” excess of capacity in subsequent periods within the same sector, or in other sectors (not shown here). This happens in small scale, i.e. solutions affect the demand for only 2 or 3 subsequent periods within the sector: This shows that delays imposed do not increase the workload per sector considerably, leaving much space for increasing further the demand, if this is also the case in the initial problem.

3.5 Discussion of results w.r.t. to methods efficiency, efficacy and quality of solutions.

The results reported for the four methods show the following qualities:

- They manage to find solutions – i.e. they do manage to regulate flights crossing an operational space in a day so as to resolve all hotspots.

- They manage to find solutions effectively: They do converge to solutions quite fast, few rounds after exploration, in most of the cases.
- They manage to reduce the average delay for the regulated flights considerably, compared to the average delay for the regulated flights reported by CFMU. The same holds for the average delay considering all flights.
- The Hierarchical method, has the potential to reduce significantly the regulated flights, and thus the average delay for all flights, compared to the other methods and of course CFMU. However, in its current implementation is not that efficient, and the average delay to the regulated flights is much higher than that reported by the other methods.

Below we provide the reported results in a consolidated form in order to reach final conclusions.

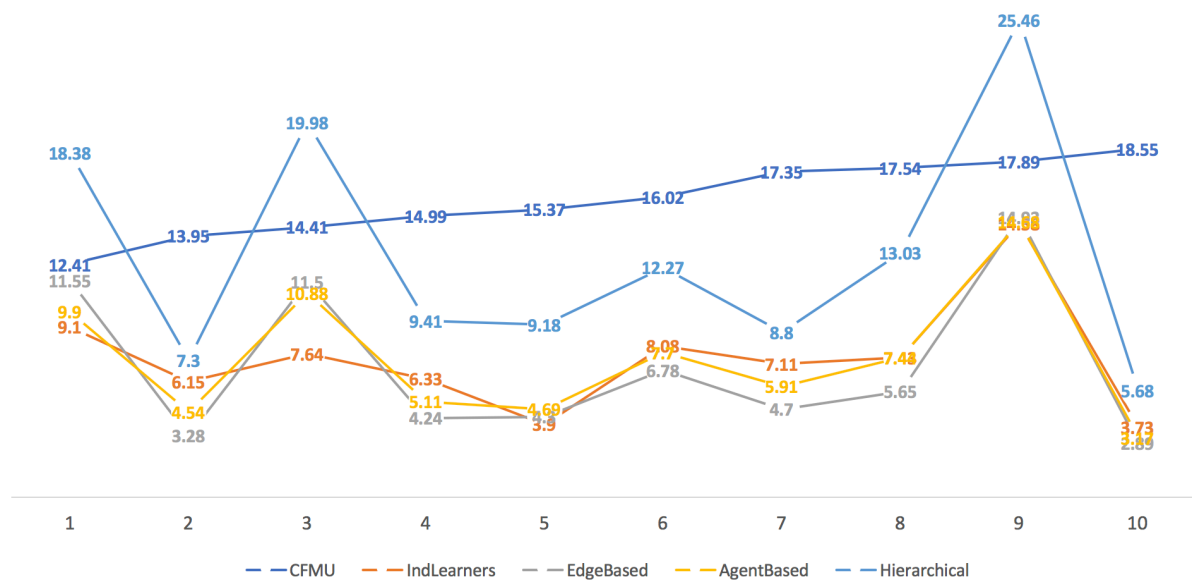


Figure 3. Average delays for regulated flights from CFMU (dark blue line), Hierarchical (light blue line), IndLearners (red line), EdgeBased (gray line), AgentBased (yellow line). The x axis shows evaluation cases and the y axis the average delays. Evaluation cases have been sorted according to CFMU average delays.

Figure 3 provides the average delays for the regulated flights per method (coloured lines), and evaluation case (x – axis). Evaluation cases are ordered according to the average delay reported by CFMU in increasing order.

It must be noticed that the trend of the average delays from all methods do not follow that of CFMU: The average delay does not seem to increase consistently with that ordering of cases, while there are some peaks in delays reported, which are consistent among methods. All methods follow the same pattern, and with average delays that are lower than those reported by CFMU, while the Hierarchical has three peaks which result to average delays much higher than those of CFMU.

These differences in patterns among the explored methods and CFMU reflect the shift of paradigm agent-based methods provide: While CFMU regulate flights in a “first enters – first regulated” basis,

agent-based methods devised regulate all flights jointly, so as to reach a solution that is of best interest to all flights – as much as it is possible.

Among the methods, the IndLearners (red line) seems more “stable” across cases, while the EdgeBased (gray line) seems to provide more qualitative results.

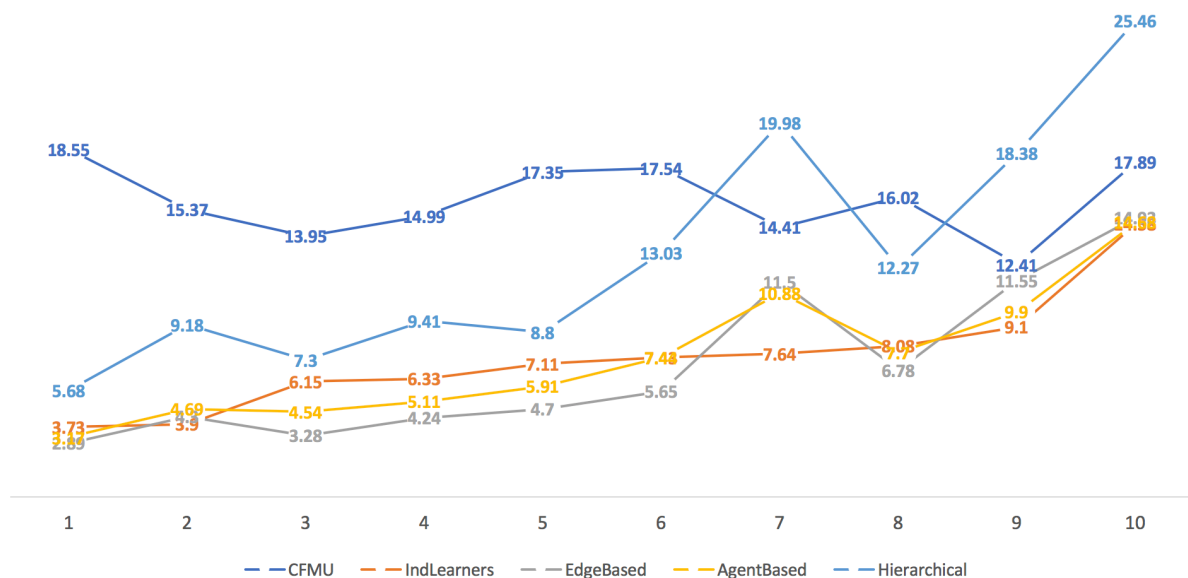


Figure 4. Average delays for regulated flights from CFMU (dark blue line), Hierarchical (light blue line), IndLearners (red line), EdgeBased (grey line), AgentBased (yellow line). The x axis shows evaluation cases and the y axis the average delays. Evaluation cases have been sorted according to IndLearners average delays.

These differences among methods are shown in in a better way in Figure 4, where evaluation cases are ordered according to IndLearners average delay for regulated flights, in increasing order: The pattern of average delay reported by CFMU is different from that the proposed methods, which is consistent among methods. Again, among them the EdgeBased method (grey line) provides more qualitative solutions in nearly all cases. The Hierarchical method follows the same pattern as the other methods, however with consistently increased average delay in all cases where it reached a solution.

This is also the case for the number of regulated flights per evaluation case, as shown in Figure 5: All methods follow the same pattern, but now, the Hierarchical method manages to provide solutions with consistently less regulated flights than the other methods. Among the other methods, the EdgeBased method seems to report the lower number of regulated flights.

CFMU regulated flights are less than those provided by all methods; however a large number of hotspots per evaluation case is unresolved with those regulated flights. It should be noted that the Hierarchical method manages to resolve all hotspots with a comparable number of (in 3 of the cases with less) regulated flights.

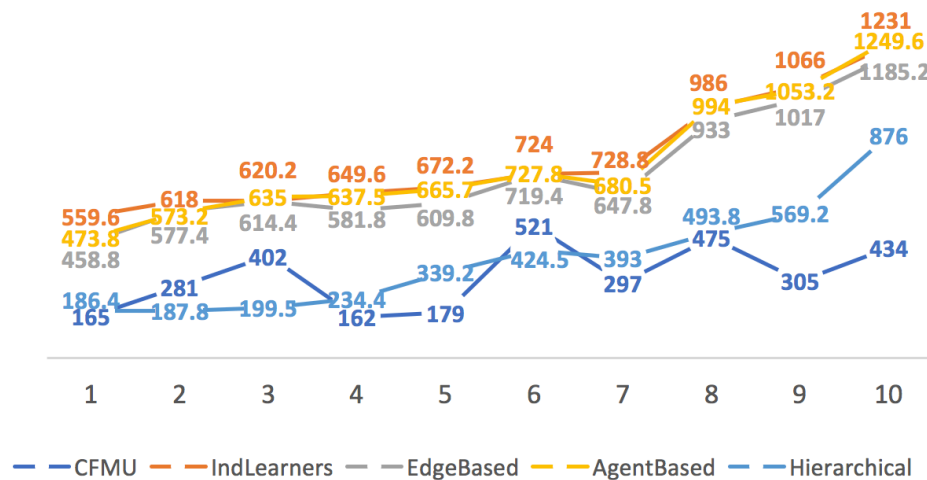


Figure 5. Number of regulated flights from Hierarchical, IndLearners, EdgeBased, AgentBased compared to CFMU regulated flights per case. The x axis shows evaluation cases and the y axis the number of flights. Evaluation cases have been sorted according to IndLearners average delays (i.e. corresponds to Figure 4).

As a conclusion of the above, the EdgeBased methods provides in all cases the more qualitative solutions, in terms of the average delay for the regulated flights, and the number of the regulated flights.

However, given that the Hierarchical method manages to reduce considerably the number of regulated flights, below we delve into the differences among Hierarchical, EdgeBased and IndLearners (AgentBased do not seem to qualify for any of the reported measures and scenario).

In doing so, Figure 6 shows the difference in average delay in regulated flights between Hierarchical and the EdgeBased methods ($\text{AverageDelay.Hierarchical} - \text{AverageDelay.EdgeBased}$), while Figure 7 shows the corresponding differences between Hierarchical and IndLearners. In both cases the average difference is approx. 6 minutes with a standard deviation of 2.5 and 4 minutes, respectively. Although more experiments are necessary to delve into the differences of methods, Hierarchical has significant differences to EdgeBased and IndLearners, as far as the average delay reported is concerned (up to 11 min for EdgeBased and up to 12 min for IndLearners), which seems to slightly increase as methods impose larger delays, but the trend is the same: i.e. all methods follow a polynomial trend as the difficulty of the case increases (forcing them to impose larger delays).

As Figures 8 and 9 show, the Hierarchical method manages to regulate nearly half of the flights regulated by the other methods. The trend in increasing the number of flights is again the same for the methods, but it must be noted that evaluation cases are ordered in a different way according to that criterion, compared to the order according to the average delay. This signifies that the difficulty of a case depends on different dimensions that are orthogonal.

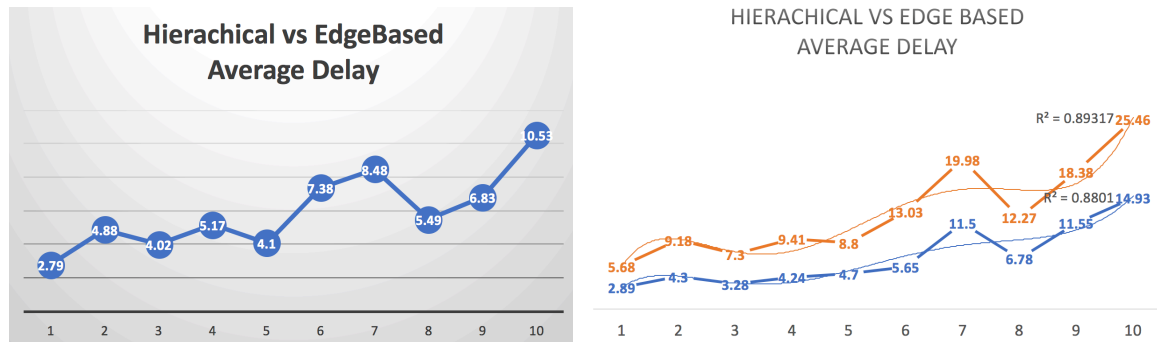


Figure 6. (Left) The difference in average delay in regulated flights between Hierarchical and the EdgeBased methods (AverageDelay.Hierarchical-AverageDelay.EdgeBased). The x axis shows evaluation cases which have been sorted according to IndLearners average delays (i.e. corresponds to Figure 4). (Right) The trends to impose delays for Hierarchical and EdgeBased methods in cases – cases are ordered according to the average delay imposed by the Edge Based method in increasing order and without including Aug13.

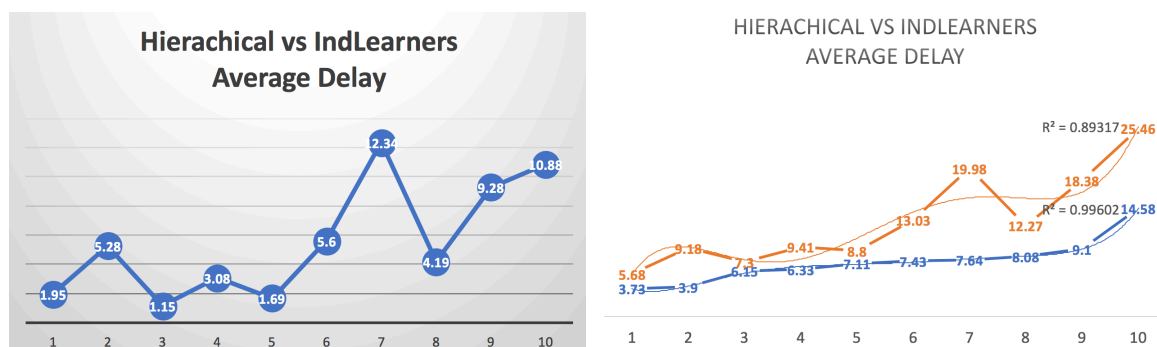


Figure 7. (Left) The difference in average delay in regulated flights between Hierarchical and the IndLearners methods (AverageDelay.Hierarchical-AverageDelay.IndLearners). The x axis shows evaluation cases which have been sorted according to IndLearners average delays (i.e. corresponds to Figure 4). (Right) The trends to impose delays for Hierarchical (red line) and EdgeBased (blue line) methods in all cases – cases are ordered according to the average delay imposed by the IndLearners method in increasing order.

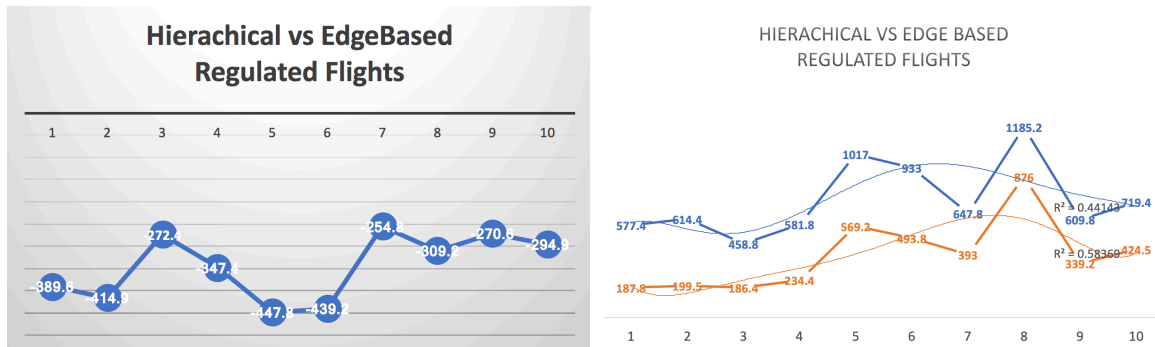


Figure 8. (Left) The difference in in regulated flights between Hierarchical and the EdgeBased methods ($\text{RegulatedFlights.Hierarchical} - \text{RegulatedFlights.EdgeBased}$). The x axis shows evaluation cases which have been sorted according to IndLearners average delays (i.e. corresponds to Figure 4). (Right) The trends to the number of regulated flights for Hierarchical (red line) and EdgeBased (blue line) methods in all cases – cases are ordered according to the number of regulated flights imposed by the IndLearners method in increasing order.

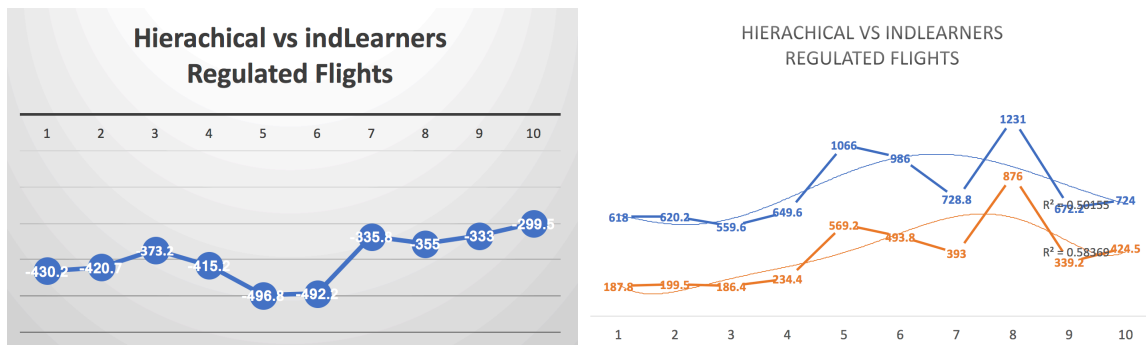


Figure 9. (Left) The difference in in regulated flights between Hierarchical and the IndLearners methods ($\text{RegulatedFlights.Hierarchical} - \text{RegulatedFlights.IndLearners}$). The x axis shows evaluation cases which have been sorted according to IndLearners average delays (i.e. corresponds to Figure 4). (Right) The trends to the number of regulated flights for Hierarchical (red line) and IndLearners (blue line) methods in cases – cases are ordered according to the number of regulated flights imposed by the IndBased method in increasing order.

3.6 Incorporating airlines preferences /constraints

3.6.1 Constructing evaluation cases with airlines preferences / constraints

In this section we explore two issues: (a) The tolerance of the methods to strict max delay (MaxDelay) restriction for all flights (denoted as *Global Max delay*), and (b) to the ability of methods to solve problems by incorporating strict conditions and preferences to the MaxDelay of some of the flights (denoted by *Local Max Delay*: The MaxDelay to a subset of flights).

The difference in the evaluation cases where airlines preferences are incorporated, is that the individual maximum delay of each flight may vary to that of the others. Here, a subset of flights were chosen (by utilizing the departure airport), to be assigned a smaller amount of maximum delay. These airports were the five biggest in Spain, thus representing the need of less delay in airports with high traffic.

We present results from the cases Aug07. This is a typical case, among the cases considered.

The result of selecting flights for strictest allowed delay, is that roughly 30% of the flights in the evaluation case have a lower delay threshold than the rest. These flights are also responsible to roughly the 30% of the occurring hotspots.

We considered subcases with Global Max Delay varying in {30,40,50} and Local Max Delay varying in {5,10, 15,25,35, 45, 55}.

3.6.2 Experimental results

Evaluation case (Aug7 - GlobalMaxDelay)	Number of Resulting Hotspots (IndLearners)	Number of Resulting Hotspots (EdgeBased)	Number of Resulting Hotspots (AgentBased)	Number of Resulting Hotspots (Hierarchical)	Number of Regulated Flights (IndLearners)	Number of Regulated Flights (EdgeBased)	Number of Regulated Flights (AgentBased)	Number of Regulated Flights (Hierarchical)
30	1	1-2	4	1	974.6	<u>888</u>	998	585.4
40	0	0	0	0	986.2	<u>903</u>	985	673.5
50	0	0	0	0	975.8	<u>884</u>	943	810.5

Table 8: The number of regulated flights per method and evaluation case when there are strict Global Max Delays (i.e. for all flights): Methods cannot always resolve DCB problems with strict MaxDelays (indicated in red for the Aug7 evaluation case). Bold indications show the best results, while the underlined ones show the second best.

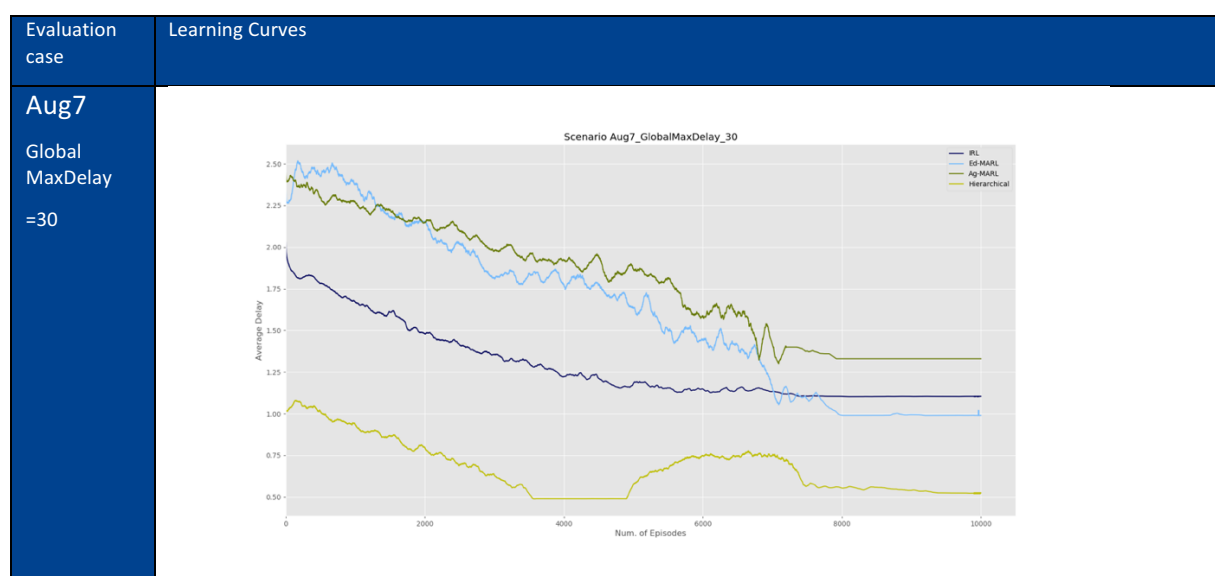
Evaluation case (Aug7 - GlobalMaxDelay)	Average Delay for regulated flights (according to CFMU data) (min/max)	Average Delay for regulated flights (IndLearners)	Average Delay for regulated flights (EdgeBased)	Average Delay for regulated flights (AgentBased)	Average Delay for regulated flights (Hierarchical)
30	17.54	6.64	6.55	7.84	10.64
40	17.54	7.14	6.21	8.85	9.98
50	17.54	7.39	6.28	9.1	7.72

Table 9: The average delays achieved for all regulated flights per method and evaluation case when there are strict Global Max Delays (i.e. for all flights) compared to CFMU average delays per evaluation case: All methods manage to reduce considerably the average delays for the regulated flights, compared to CFMU values. Bold indications show the best results.

Tables 8 and 9 show that all methods manage to solve DCB problems, even if strict max delays conditions are set to all flights. However, for MaxDelay=30' all methods could not provide a solution. It must be noted that this depends on each evaluation case: For instance for Jun5, even with MaxDelay=30 we did have solutions from the proposed methods (i.e. delays to flights resulting to zero hotspots).

In all cases the EdgeBased approach provides the best average delay for the regulated flights, and the second best (after the Hierarchical) total number of regulated flights.

It must be noticed, that contrary to the other methods, the Hierarchical method reduces the number of regulated flights as conditions become stricter, with the cost of increasing the average delay for those flights. The other methods increase slightly the number of regulated flights (notice that for 30' methods do not provide solutions), but reduce the average delay on regulated flights.



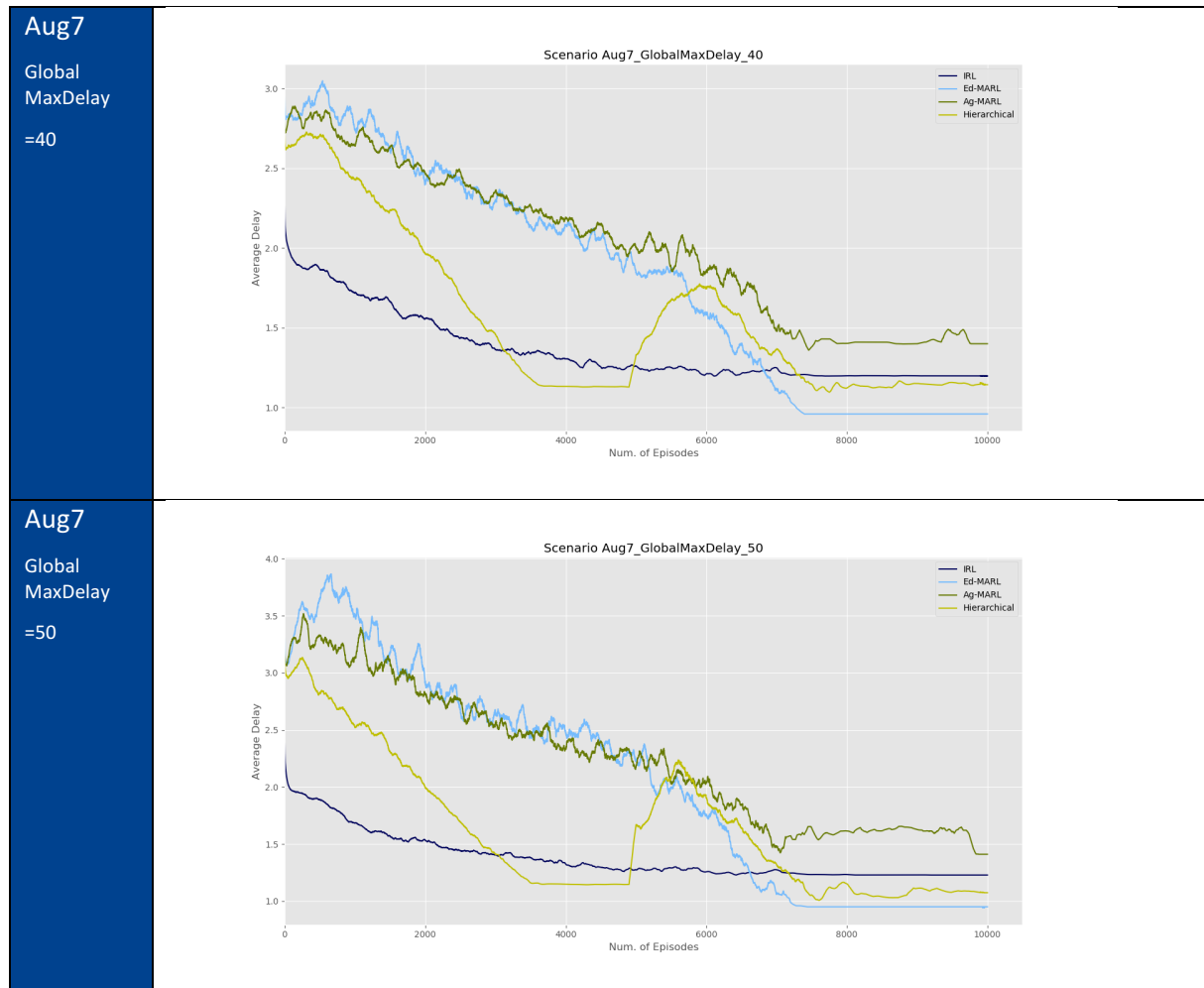


Table 10: The learning curves of all methods for the Aug7 evaluation case, showing how methods manage to learn agents' (flights) joint policies to resolve DCB, when requirements for the global delay (i.e. the MaxDelay for all flights) are strict. The x axis corresponds to the episodes, while the y axis to the average delay reported for all flights.

Learning curves in all sub-cases show that methods do converge rather slowly, compared to the original case, which is explained by the more strict conditions they have to satisfy. It should be noticed that EdgeBased and IndLearners converge more effectively, while this is not the case for AgentBased and Hierarchical, even when the Global Max Delay is 50'.

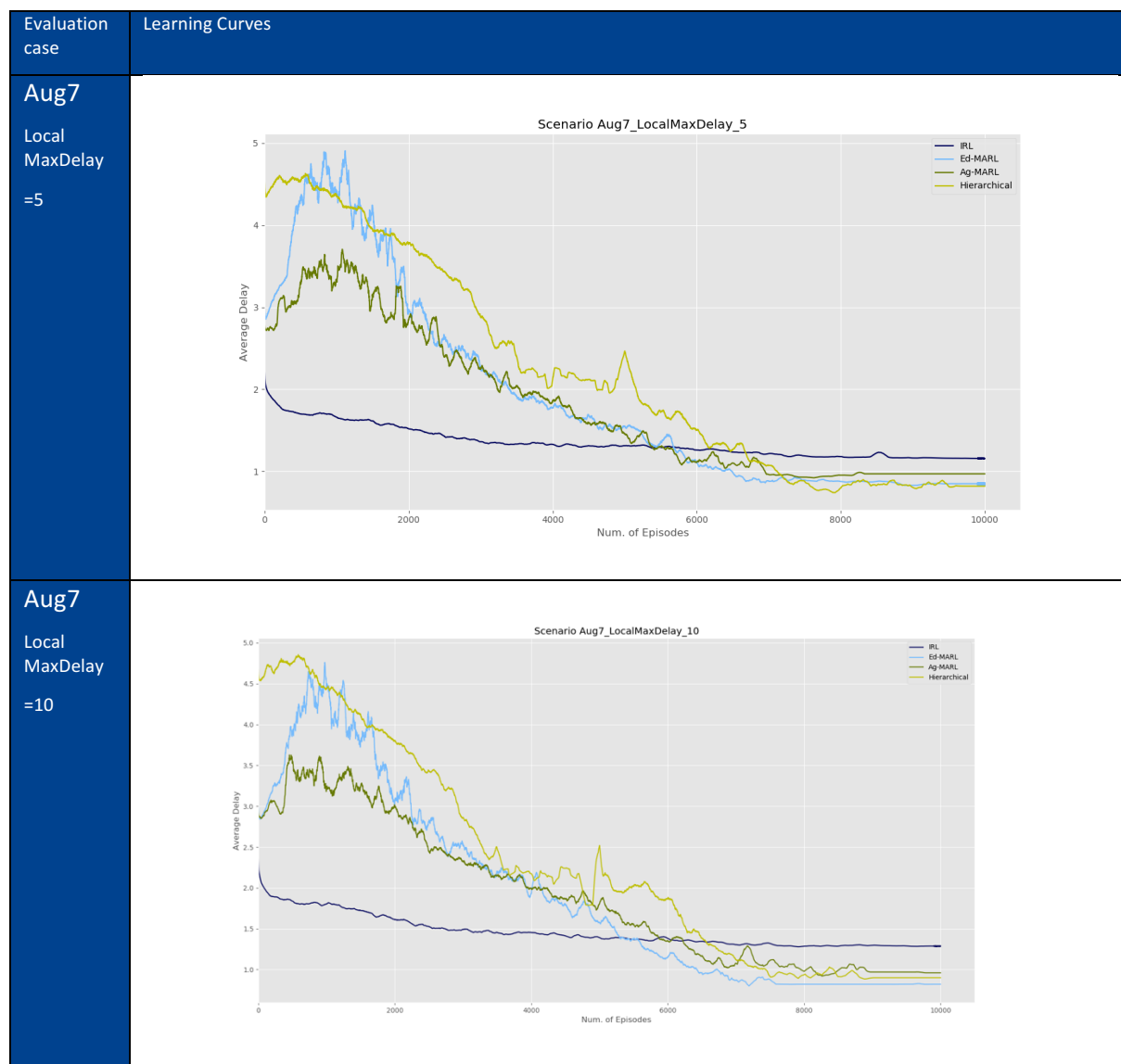
The results are similar when we restrict the max delay for a subset of the flights: In that case we may require these flights (nearly 30% of flights participating in hotspots) to have Max Delay (Local Max Delay) even equal to 15' without affecting considerably the total number of regulated flights and the average delay for all regulated flights.

Evaluation case Name (Aug7 - LocalMaxDelay)	Number of Resulting Hotspots (IndLearners)	Number of Resulting Hotspots (EdgeBased)	Number of Resulting Hotspots (AgentBased)	Number of Resulting Hotspots (Hierarchical)	Number of Regulated Flights (IndLearners)	Number of Regulated Flights (EdgeBased)	Number of Regulated Flights (AgentBased)	Number of Regulated Flights (Hierarchical)
5	2-3	3	6	3	954	<u>789.5</u>	849.5	417
10	2	2	4	2	1003.4	<u>833.5</u>	846.5	502
15	0	0	4	0	998	<u>883</u>	899	455
25	0	0	0	0	989.4	<u>861</u>	909	437
35	0	0	0	0	986.4	<u>864</u>	979	449.5
45	0	0	0	0	986.6	<u>894</u>	970.4	455.5
55	0	0	0	0	988.4	<u>911.5</u>	980.5	449

Table 11: The number of regulated flights per method and evaluation case when there are strict Local Max Delays (i.e. strict Max Delay preferences for some of the flights): Methods can not always resolve DCB problems with strict Local MaxDelays (indicated in red for the Aug7 evaluation case). Bold indications show the best results, while the underlined ones show the second best.

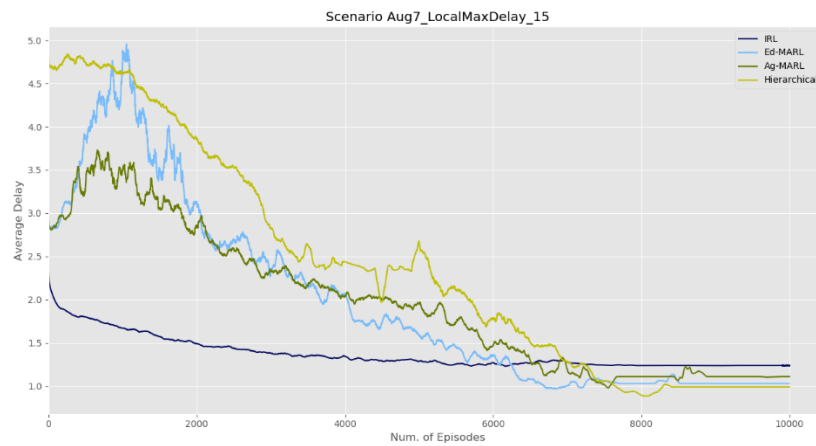
Evaluation case Name (Aug7 - LocalMaxDelay)	Average Delay for regulated flights (according to CFMU data) (min/max)	Average Delay for regulated flights (IndLearners)	Average Delay for regulated flights (EdgeBased)	Average Delay for regulated flights (AgentBased)	Average Delay for regulated flights (Hierarchical)
5	17.54	7.1	6.09	6.6	11.54
10	17.54	7.54	5.89	6.65	10.47
15	17.54	7.27	6.86	7.23	12.61
25	17.54	7.28	6.13	7.49	13.45
35	17.54	7.23	6.03	8.31	14.15
45	17.54	7.3	5.7	8.27	14.05
55	17.54	7.48	5.86	8.36	12.69

Table 12: The average delays achieved for all regulated flights per method and evaluation case when there are strict Local Max Delays (i.e. strict Max Delay preferences for some of the flights) compared to CFMU average delays per evaluation case: All methods manage to considerably reduce the average delays for the regulated flights, compared to CFMU values. Bold indications show the best results.



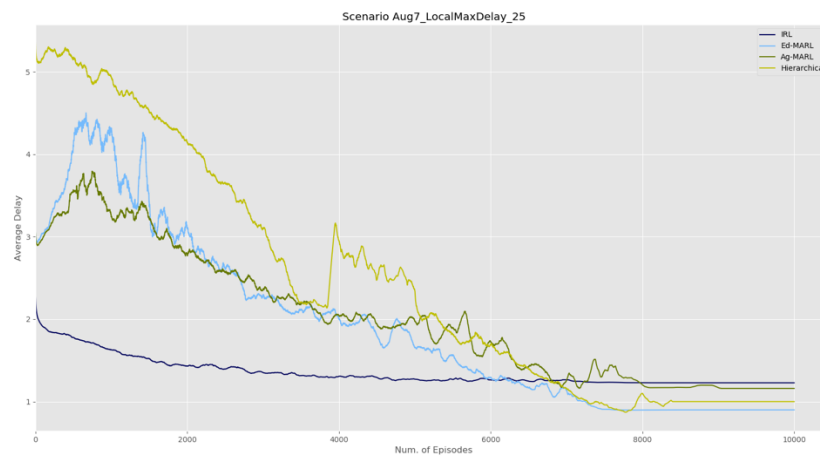
Aug7

Local
MaxDelay
=15



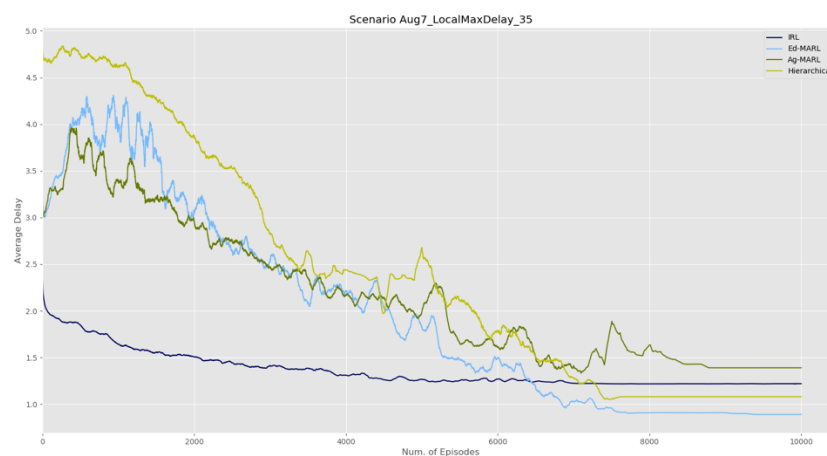
Aug7

Local
MaxDelay
=25



Aug7

Local
MaxDelay
=35



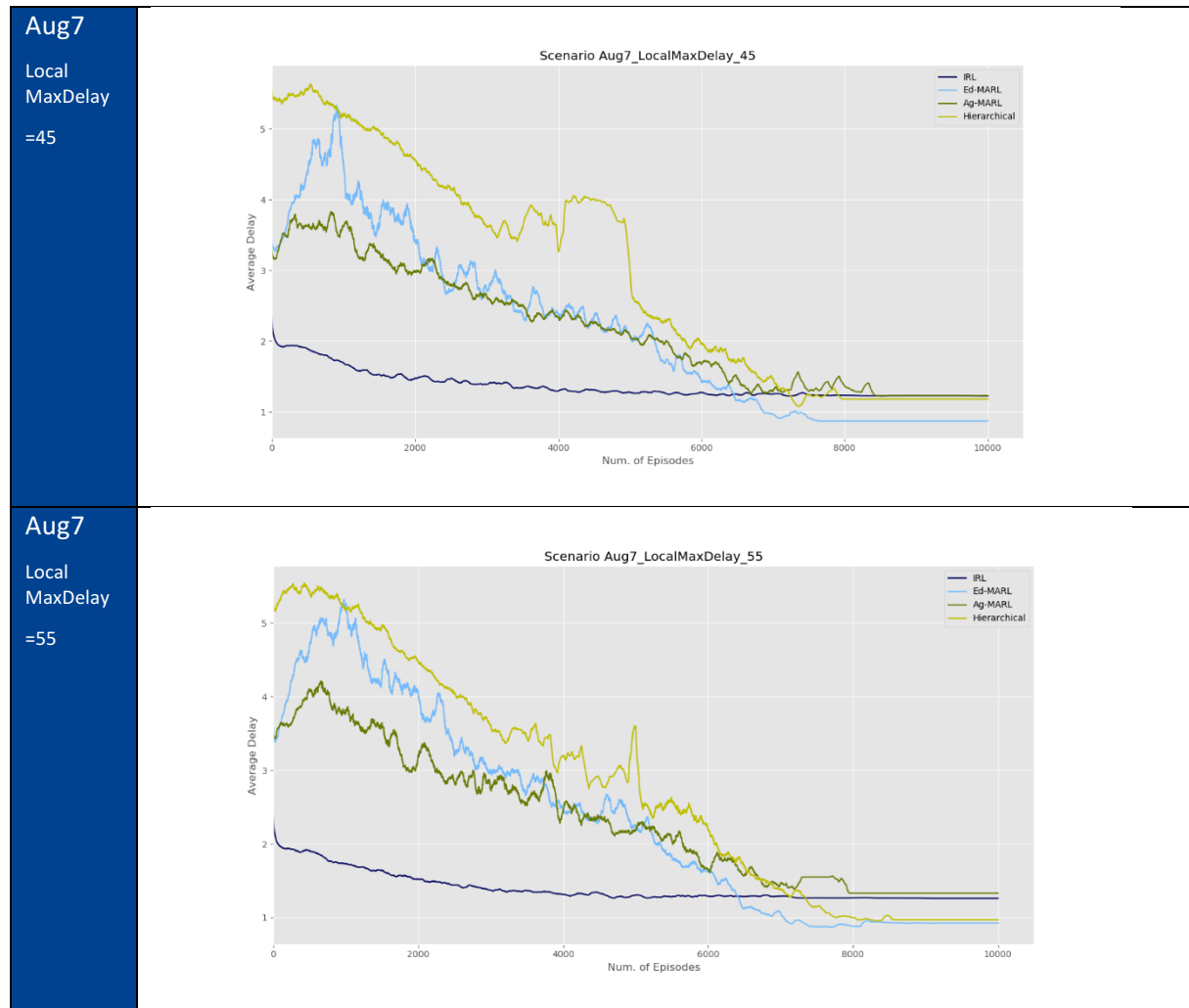


Table 13: The learning curves of all methods for the Aug7 evaluation case, showing how methods manage to learn agents' (flights) joint policies to resolve DCB, when requirements for the delays for some of the flights are strict. Local MaxDelay indicates the MaxDelay for some of the flights, while the Max Delay for the rest of the flights is equal to that in the original evaluation case. The x axis corresponds to the episodes, while the y axis to the average delay reported for all flights.

4 Visualizations of solutions overview in space and time

This section provides visualizations of solutions overview in space and time in one of the evaluation cases: Jul2, which is the “hardest” among cases. However, results shown are representative of other cases and indicative of the benefits and limitations of proposed methods.

4.1 Time series of sector loads

Flights’ trajectories provided by the initial problem state (Original), after CFMU regulations (CFMU), and after the regulations prescribed by the methods (IndLearners, EdgeBased, AgentBased, Hierarchical) were aggregated by the sectors and time intervals of length 60 minutes with a shift of 30 minutes, resulting in time series of sector entry counts attached to the sectors (Figure 10).

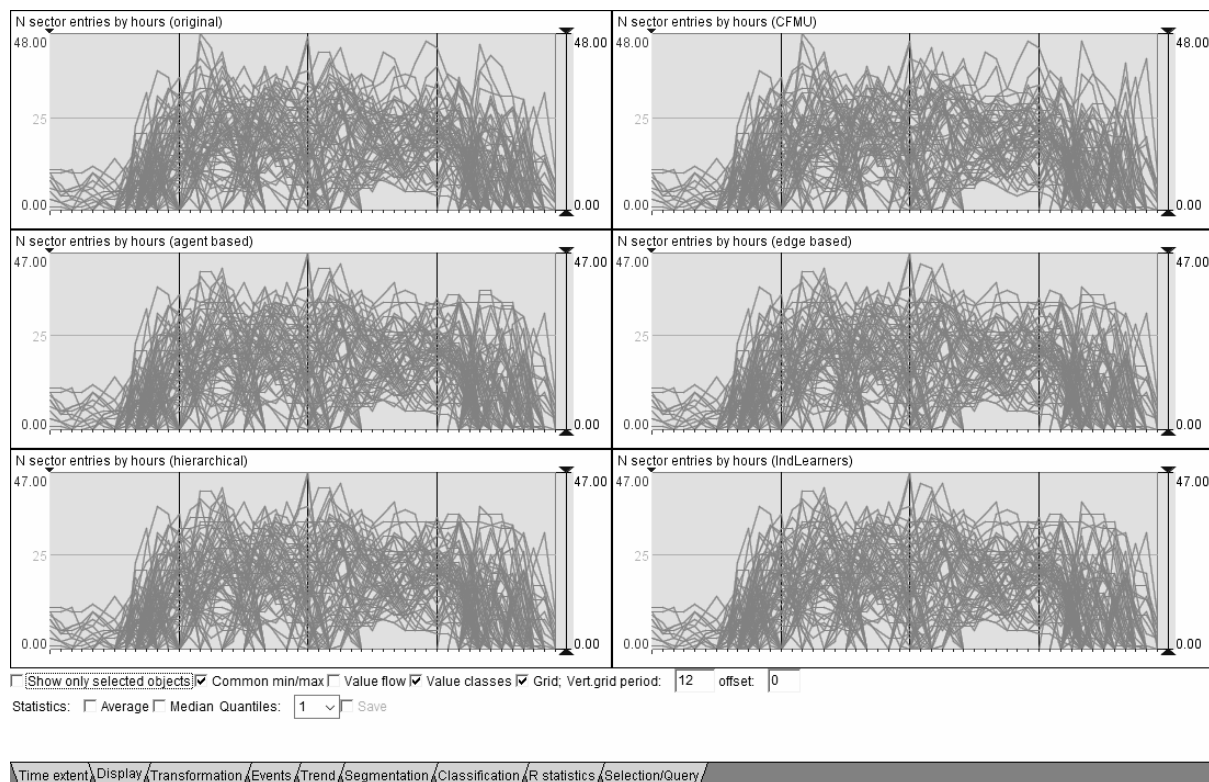
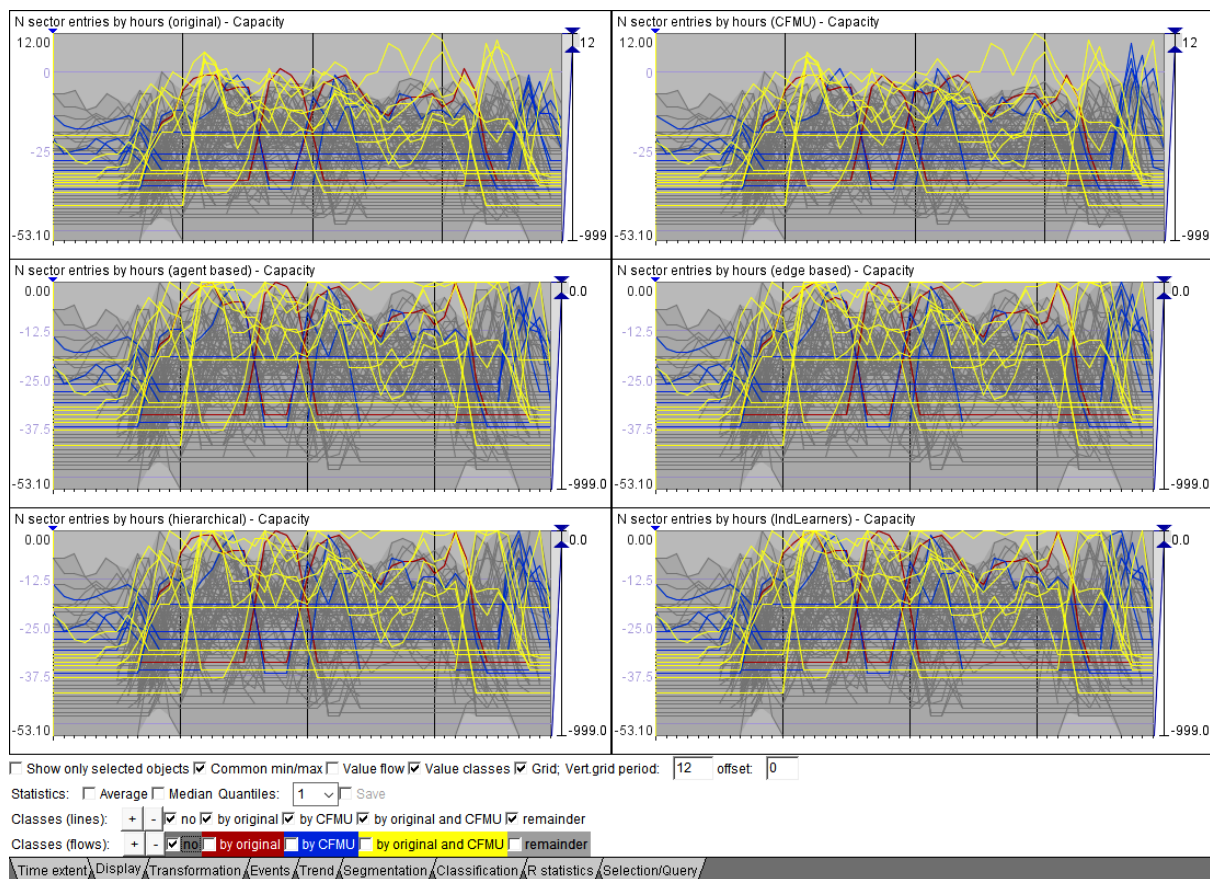


Figure 10. Time series of sector entry counts under different regulation scenarios.

The hourly values were compared with the sector capacities (the capacities were extracted from the hourly values); see Figure 11. There were 19 sectors whose capacities were exceeded at least once at least by 1 flight per hour either by the original flights or by the CFMU-regulated flights. Among them, the capacities of 2 sectors were exceeded only by the original trajectories, capacities of 7 sectors only by the CFMU trajectories, and the capacities of 10 sectors were exceeded by both original and CFMU trajectories. In Figure 11, the time series of these sectors are shown in red, blue, and yellow, respectively. Figure 12 shows only the excesses of the sector capacities, i.e., the positive differences between the loads and the capacities. It includes the graphs only for the original and CFMU-regulated trajectories. There were no excesses of sector capacities in the regulation scenarios Agent Based, Edge Based, Hierarchical, and IndLearners.

**Figure 11. Differences between the hourly sector loads (numbers of entries) and the sector capacities.**

Coloured lines correspond to the sectors whose capacities were exceeded by the original (red), CFMU-regulated (blue), or both original and CFMU-regulated flights (yellow). In all four regulation scenarios Agent Based, Edge Based, Hierarchical, and IndLearners, the differences between the sector loads and capacities do not exceed 0, i.e., there are no capacity excesses.

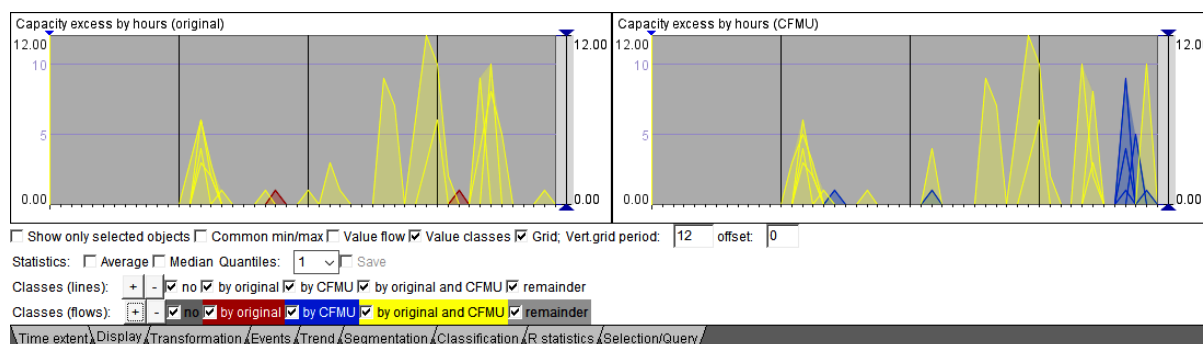


Figure 12. Excesses of sector capacities by the original and CFMU-regulated flights. The colours have the same meanings as in Figure 11.

The following image of a table contains the list of sectors whose capacities were exceeded.

	N capacity excesses (original)	N capacity excesses (CFMU)	Capacity excesses (original), sum	Capacity excesses (CFMU), sum	Capacity excesses (original), max	Capacity excesses (CFMU), max	Capacity excesses (original), min	Capacity excesses (CFMU), min
LECPDWO	1	0	1	0	1	0	1	0
LECPDWX	1	0	1	0	1	0	1	0
LECPGOX	0	1	0	1	0	1	0	1
LECPGIM	0	1	0	1	0	1	0	1
LEMDDNS	0	1	0	1	0	1	0	1
LECMOZI	0	1	0	4	0	4	0	4
LECMSAI	0	1	0	9	0	9	0	9
LEBLALL	0	1	0	1	0	1	0	1
LECBKWK	0	1	0	5	0	5	0	5
LECMSAN	14	10	73	58	12	12	1	2
LECPMXX	3	3	12	11	6	5	3	3
LECLTMS	1	1	10	8	10	8	10	8
LECPAPP	2	2	9	9	6	6	3	3
LECBP1I	1	2	9	12	9	10	9	2
LECBMNU	2	2	8	8	6	6	2	2
LECBMNL	1	1	4	4	4	4	4	4
LECMZML	1	1	1	1	1	1	1	1
LECBCCCL	1	1	1	1	1	1	1	1
LECBKKE	1	1	1	10	1	10	1	10

☒ group by classes Sort by: Capacit Descending ☒ TableLens ☐ condensed Attribute...

Figure 13. For the sectors whose capacities were exceeded, the table shows the numbers of the excess events, the total amounts of excess (sum of all excesses), and the maximal and minimal excesses based on the original and CFMU-regulated flights. The colours have the same meanings as in Figure 11.

The following maps and 3D displays show the sectors; the colouring from yellow to red represents the maximal capacity excess by the original flights (Figure 14 - Figure 16) and by the CFMU-regulated flights (Figure 17 - Figure 19).

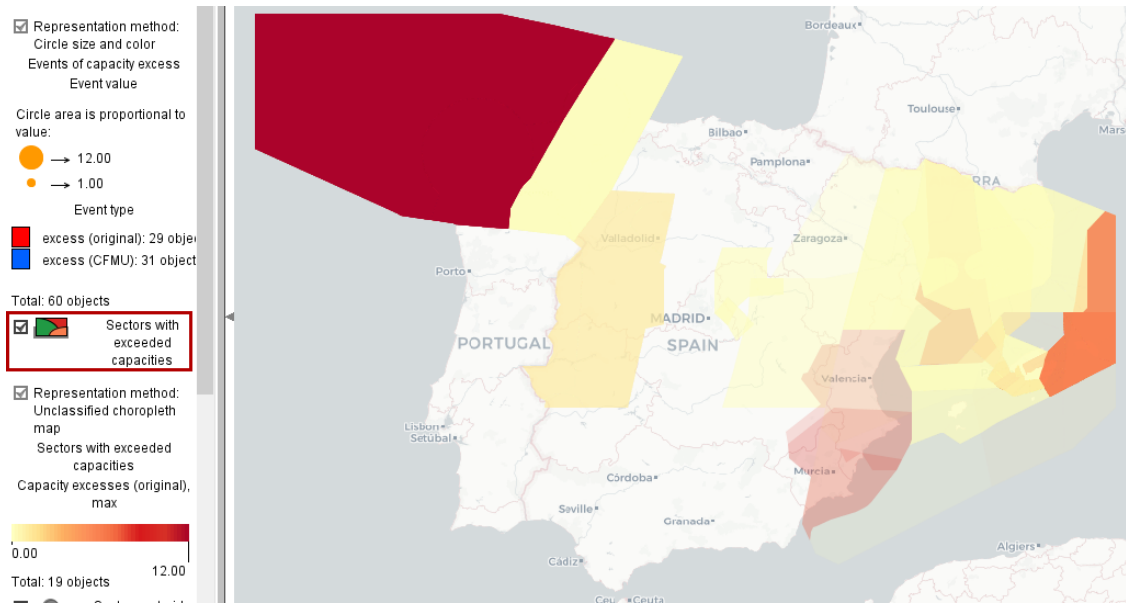


Figure 14. The sectors whose capacities were exceeded by the original or CFMU-regulated flights (19 sectors in total). The colouring from yellow to red represents the maximal capacity excess by the original flights.

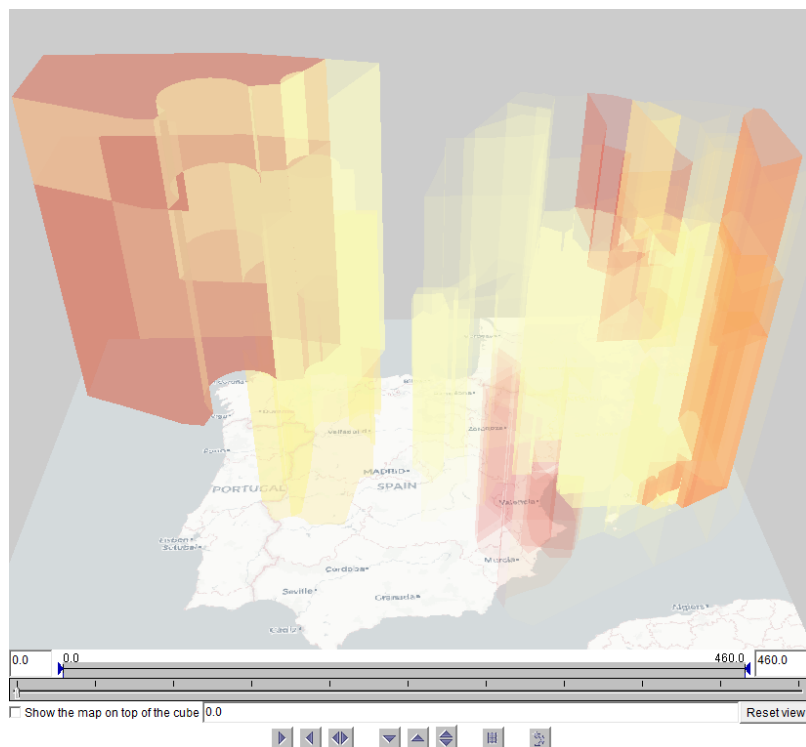


Figure 15. A 3D view shows the 3D shapes of the sectors whose capacities were exceeded. The colouring is the same as in Figure 14.

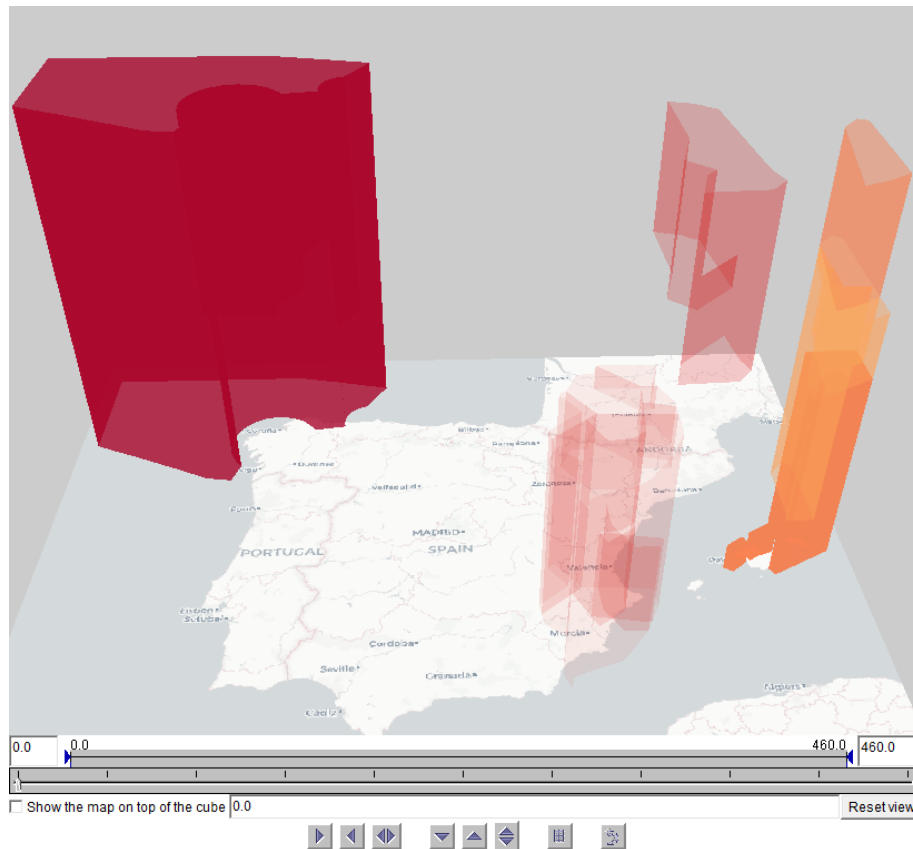


Figure 16. The 3D view shows the sectors whose capacities were exceeded by 3 or more flights per hour based on the original flight data.

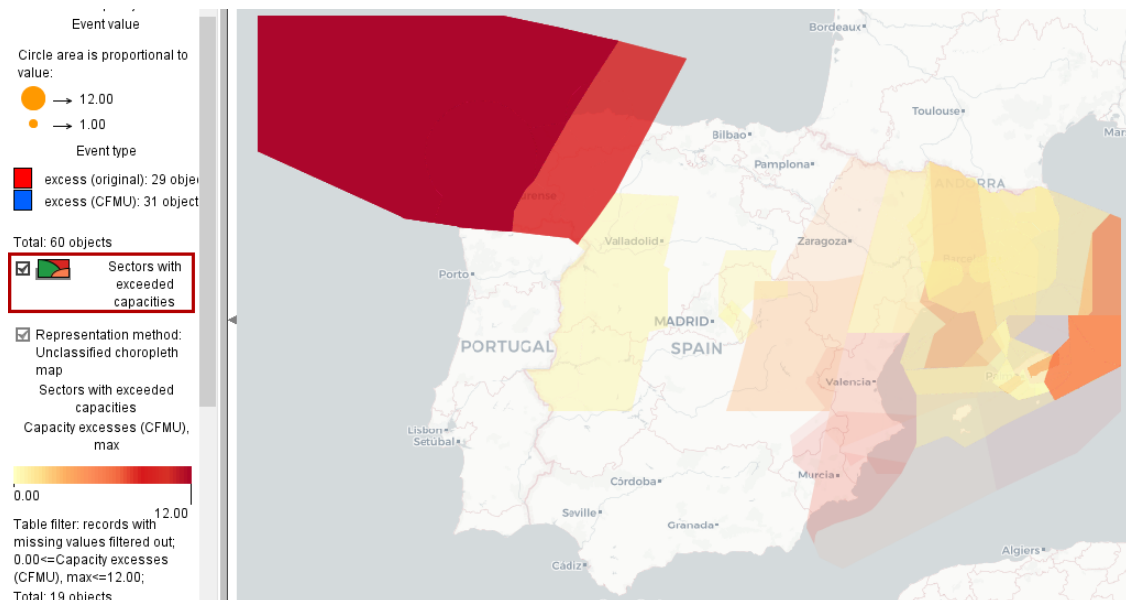


Figure 17. The map shows the same 19 sectors as in the previous figures. The colouring from yellow to red represents the maximal capacity excess by the CFMU-regulated flights.

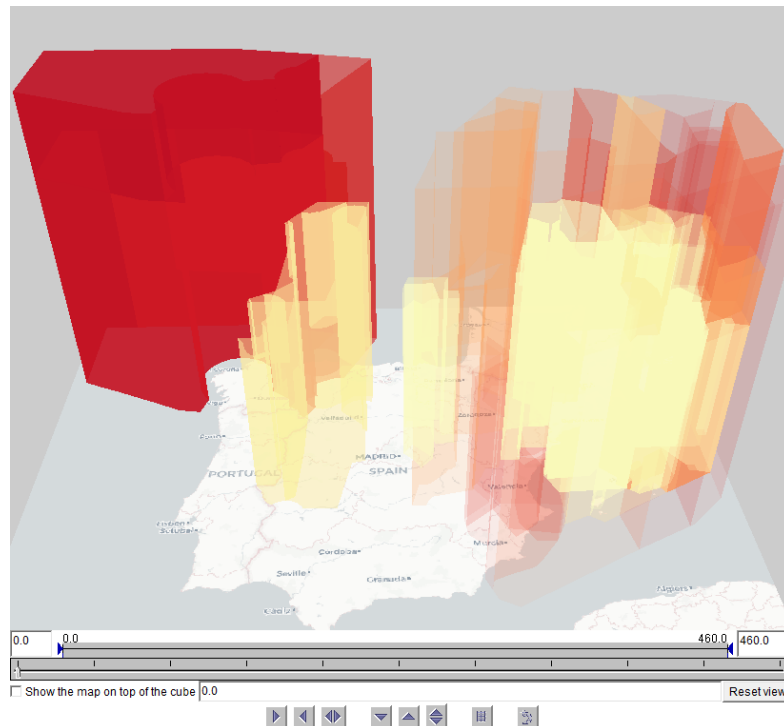


Figure 18. A 3D view shows the 3D shapes of the sectors whose capacities were exceeded. The colouring is the same as in Figure 17.

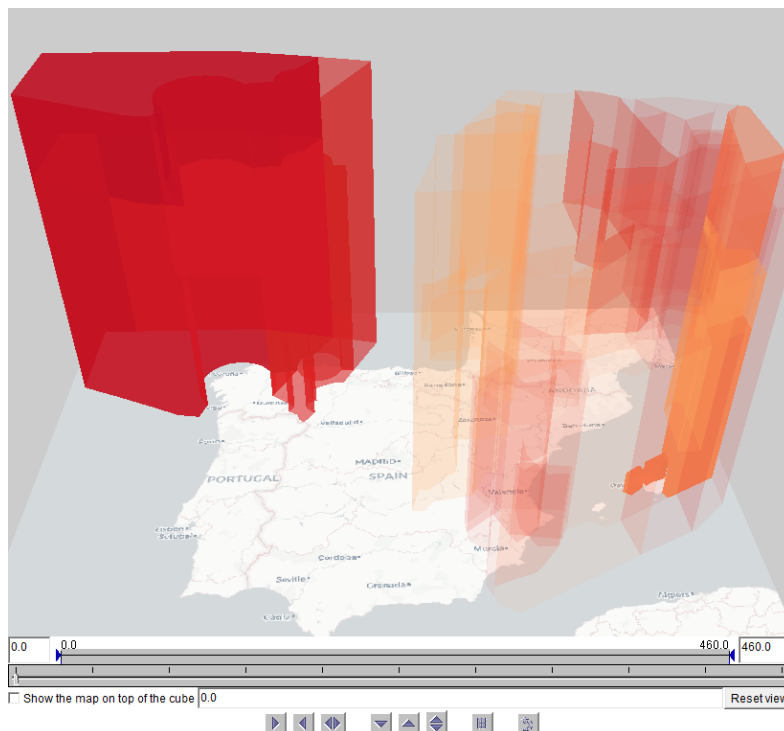


Figure 19. The 3D view shows the sectors whose capacities were exceeded by 3 or more flights per hour based on the CFMU-regulated flight data.

4.2 Events of exceeding sector capacity

We have extracted the events of the sector capacity excess from the time series of the sector load differences to the capacities. The following maps and space-time cubes show the spatio-temporal distributions of the events of sector capacity being exceeded based on the original and CFMU-regulated flight data. The events are represented by circles; the sizes are proportional to the excess values (see the map legend). The spatial positions of the events are the positions of the sector centroids. The temporal axis in the cubes goes from the bottom to the top.

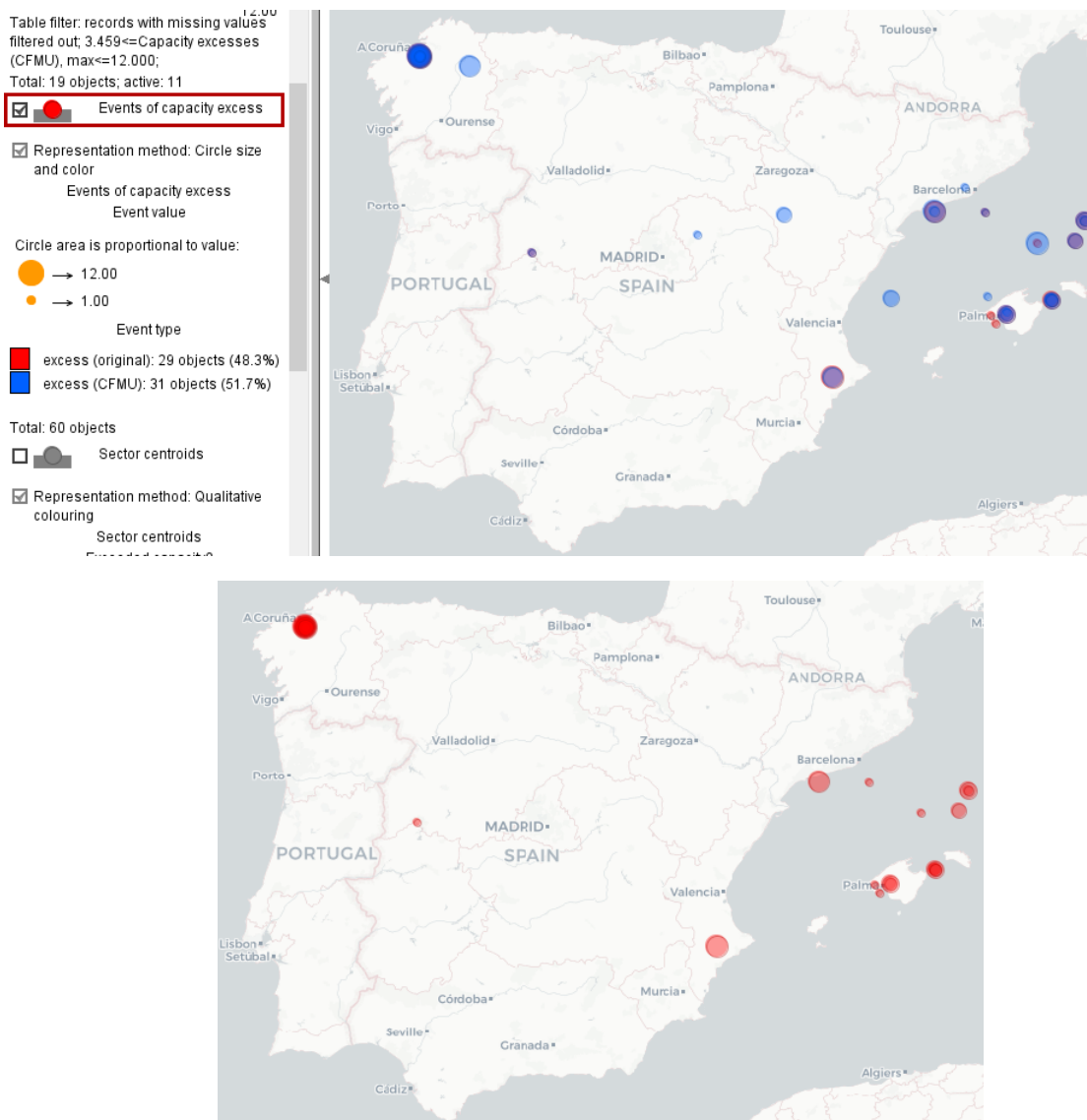




Figure 20. The maps represent the events of sector capacity excess based on the original (red) and CFMU-regulated (blue) flight data. The circle sizes are proportional to the excess amounts.

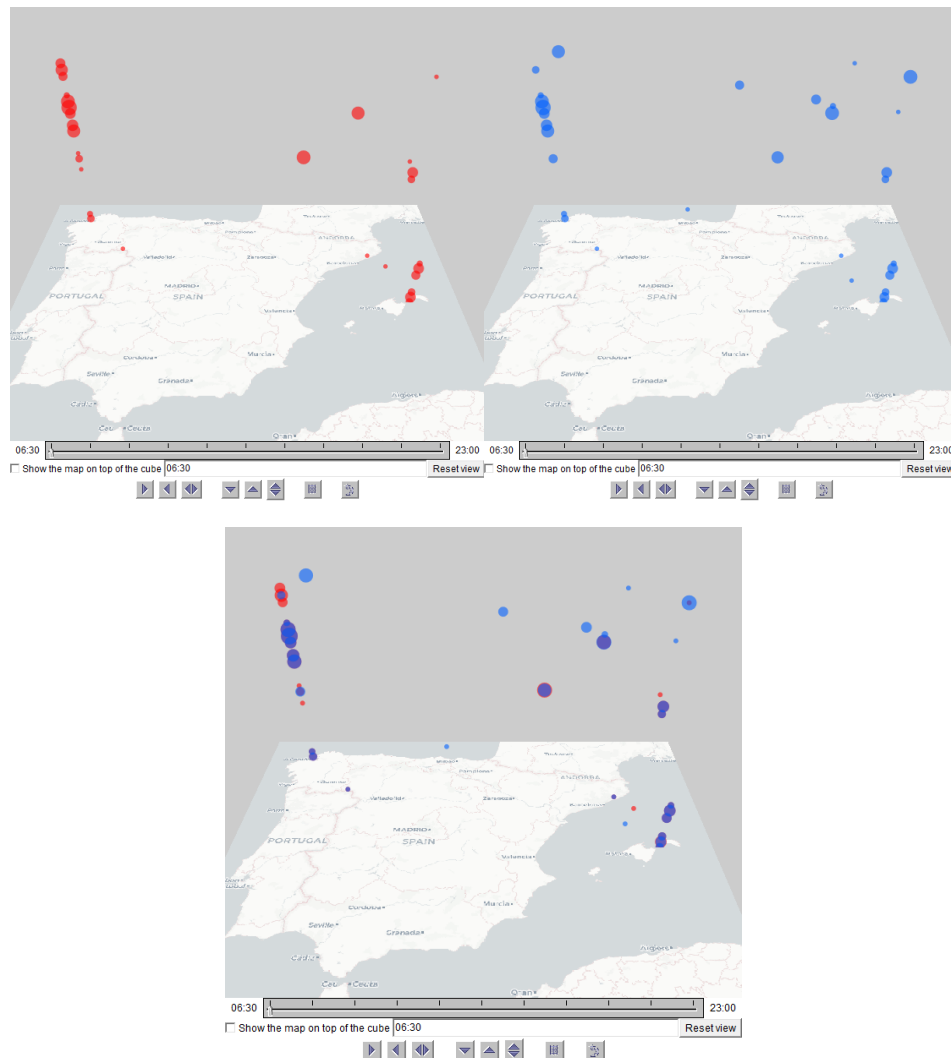
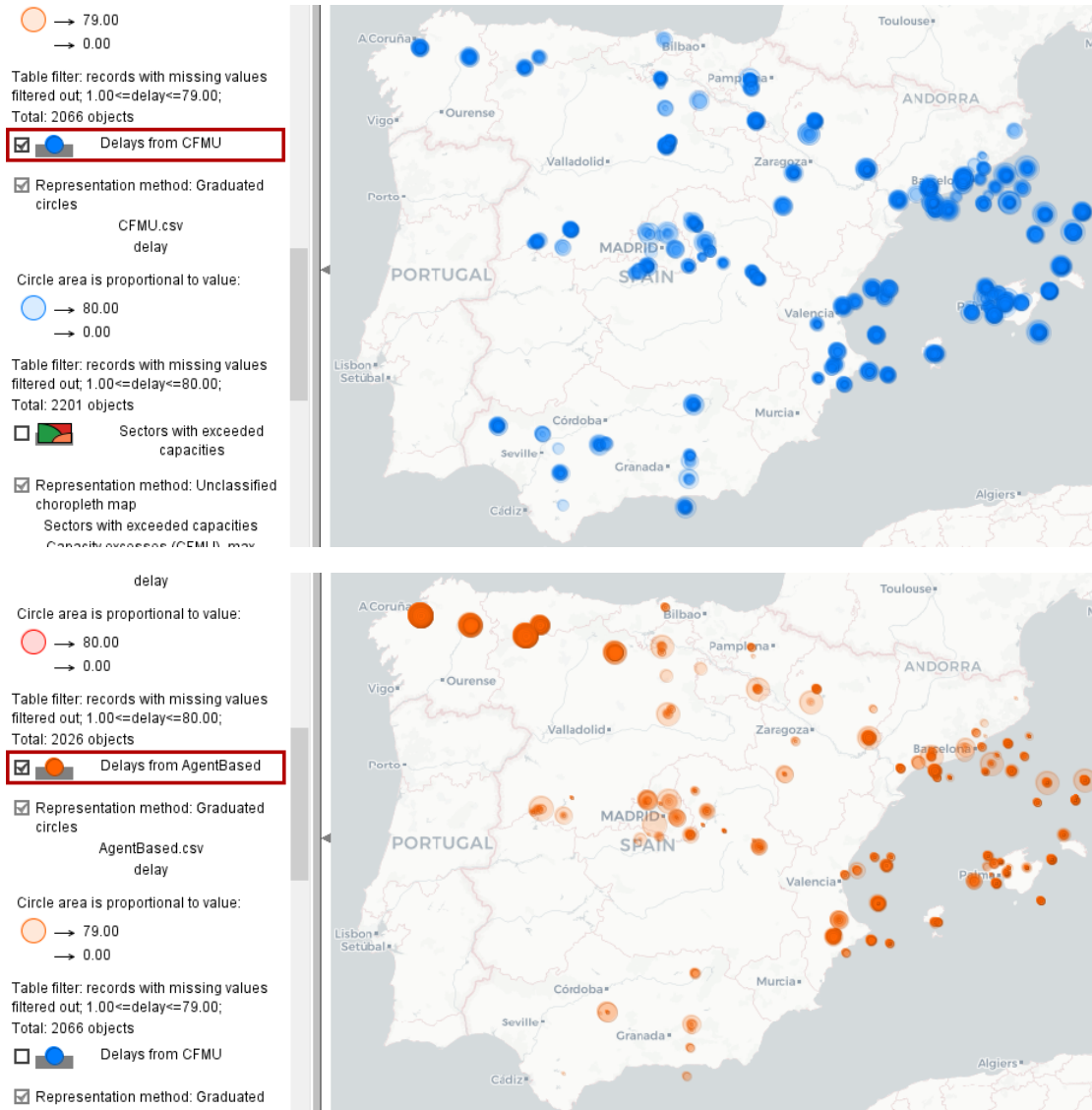


Figure 21. The capacity excess events are shown in a space-time cube. The vertical dimension, from bottom to top, represents time. The events are represented by circles; the sizes and colours are as in Figure 20.

4.3 Flight delays in space and time

In the following maps and space-time cubes, the delays within sectors are represented by circles with the sizes proportional to the delay durations.



Circle area is proportional to value:

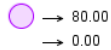



Table filter: records with missing values
filtered out; $1.00 \leq \text{delay} \leq 80.00$;
Total: 1777 objects

☒  Delays from EdgeBased

☒ Representation method: Graduated
circles

EdgeBased.csv
delay

Circle area is proportional to value:

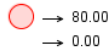

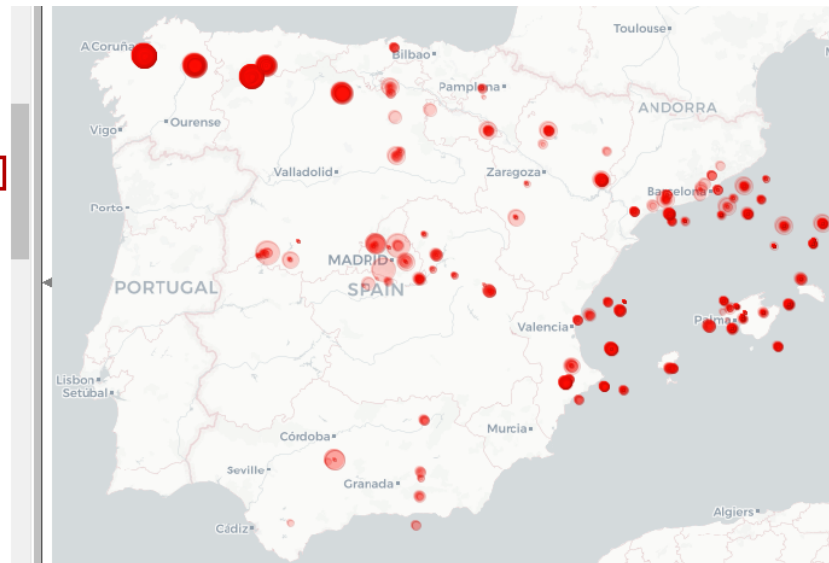


Table filter: records with missing values
filtered out; $1.00 \leq \text{delay} \leq 80.00$;
Total: 2026 objects

☐  Delays from AgentBased

☒ Representation method: Graduated
circles

AgentBased.csv



Circle area is proportional to value:

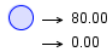



Table filter: records with missing values
filtered out; $1.00 \leq \text{delay} \leq 80.00$;
Total: 1991 objects

☒  Delays from Hierarchical

☒ Representation method: Graduated
circles

Hierarchical.csv
delay

Circle area is proportional to value:

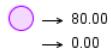

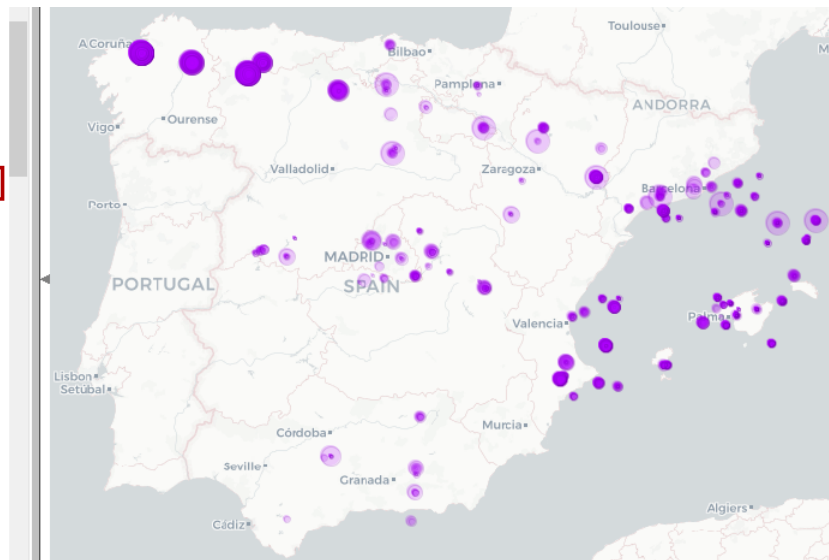


Table filter: records with missing values
filtered out; $1.00 \leq \text{delay} \leq 80.00$;
Total: 1777 objects

☐  Delays from EdgeBased

☒ Representation method: Graduated
circles

EdgeBased.csv



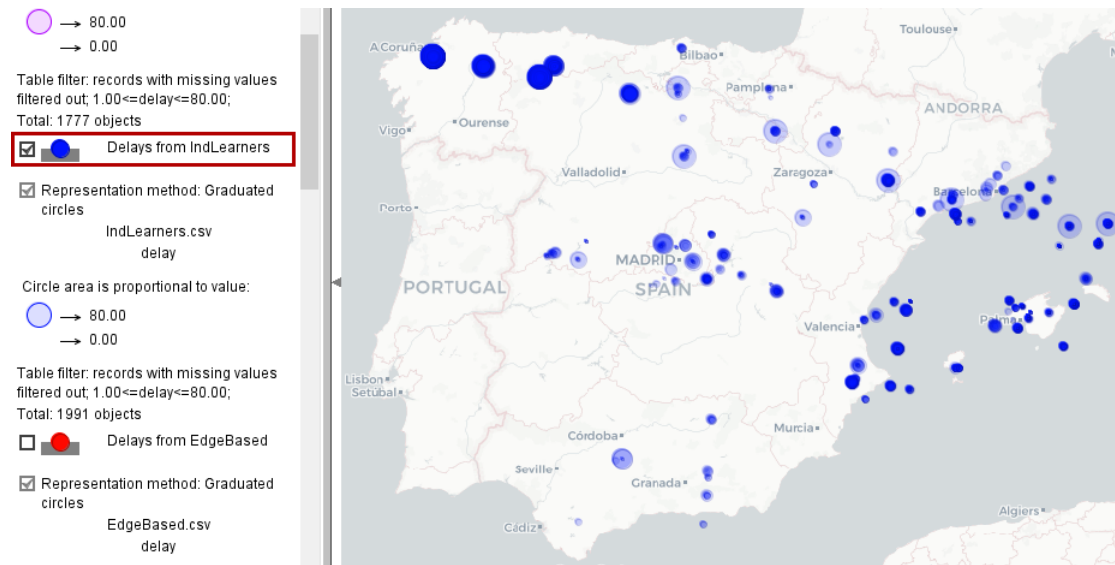


Figure 22. Flight delays are represented by circles positioned at the sector centroids. The sizes are proportional to the delay durations. From top to bottom: delays decided by CFMU, AgentBased, EdgeBased, Hierarchical, and IndLearners.

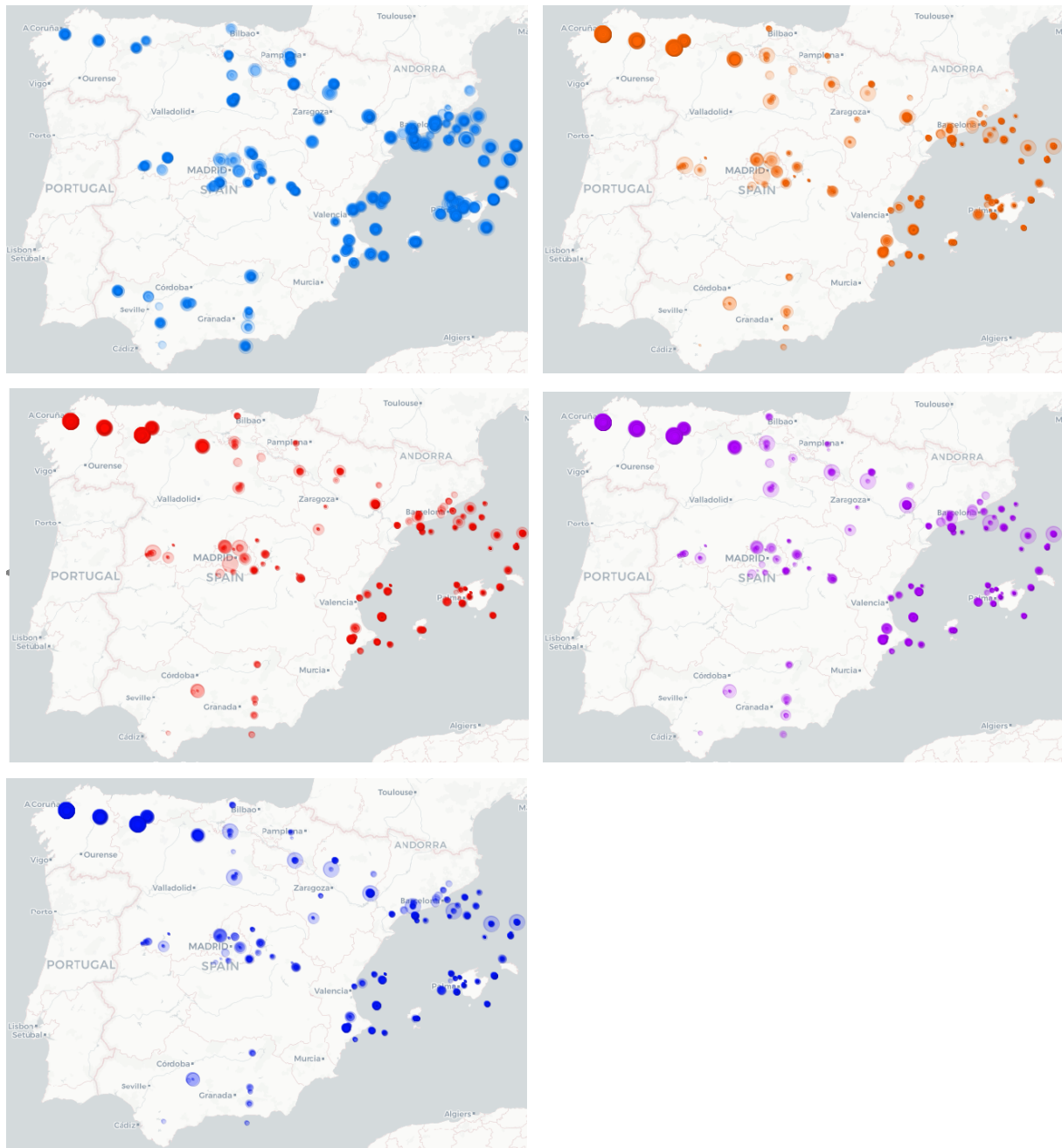


Figure 23. The same maps as in Figure 22 are shown together (without legends) for comparison. From top to bottom and from left to right: CFMU, AgentBased, EdgeBased, Hierarchical, and IndLearners.

The maps show that, compared to the CFMU, the methods AgentBased, EdgeBased, Hierarchical, and IndLearners reduce the delays on the east (areas of Barcelona, Canary Islands, and Valencia) and on the south (Seville and Granada) but increase the delays on the northwest of Spain.

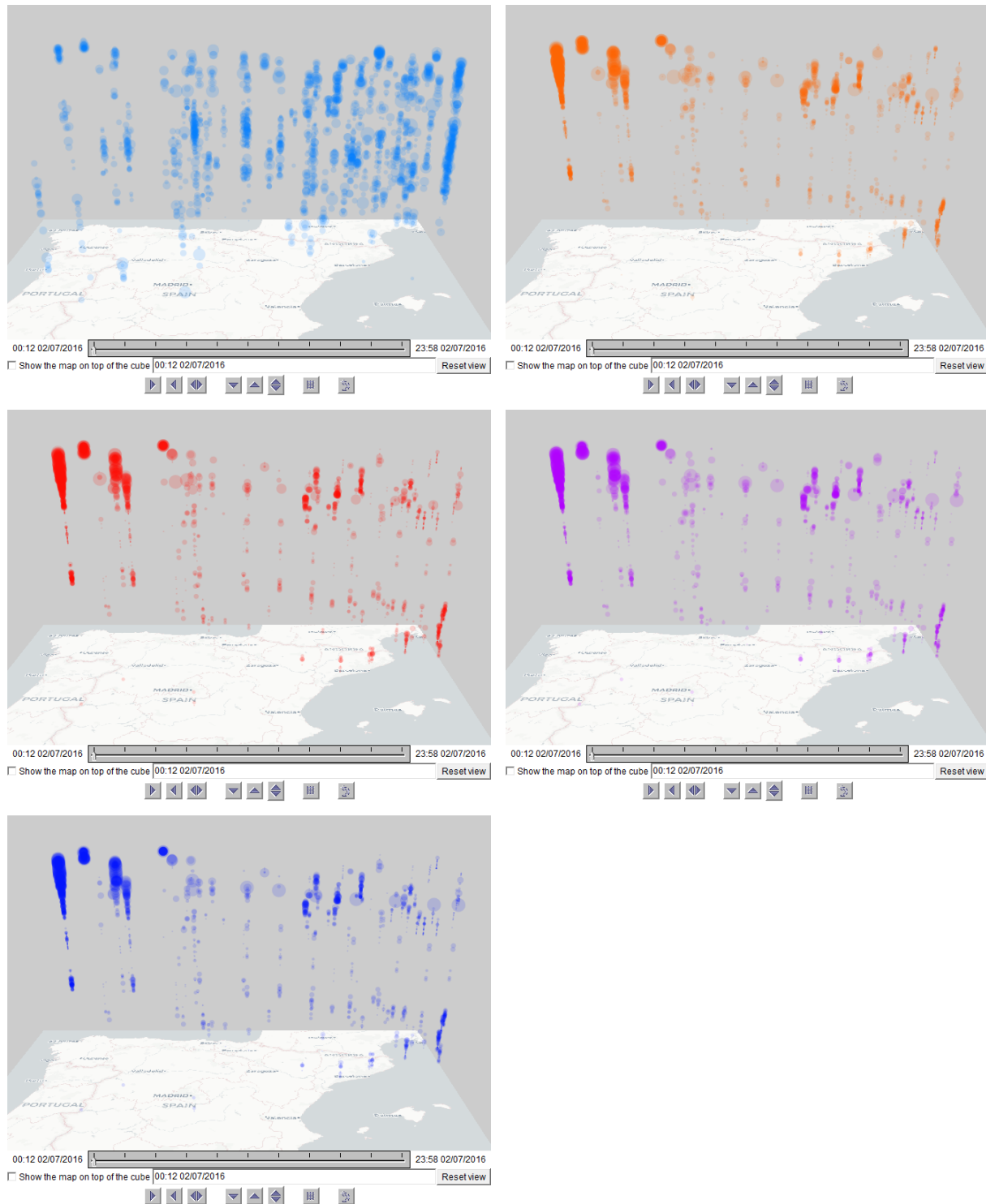


Figure 24. The space-time cubes show the spatio-temporal distribution of the delays. The time axis is oriented upwards. From top to bottom and from left to right: CFMU, AgentBased, EdgeBased, Hierarchical, and IndLearners.

The cubes show that, compared to the CFMU, the methods AgentBased, EdgeBased, Hierarchical, and IndLearners perform notably better in the first half of the day. In all areas except the northwest they also perform well in the second half of the day. The delays in the north-western area significantly increase by the end of the day according to all methods.

4.4 Flight delay statistics

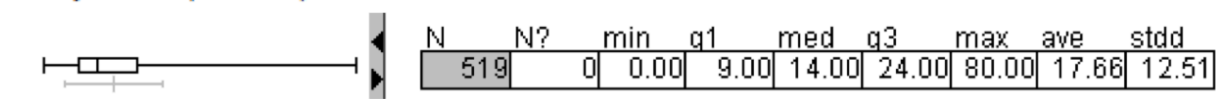
Here we look at statistics of flight delays introduced by different approaches. For each flight according to CFMU, AgentBased, EdgeBased, Hierarchical, and IndLearners, we compute

- The difference of the flight start time with respect to the original (i.e., the start delay)
- The difference of the flight end time with respect to the original (i.e., the end delay)
- The difference of the flight start time with respect to the CFMU (i.e., the increase or decrease of the start delay)
- The difference of the flight end time with respect to the CFMU (i.e., the increase or decrease of the end delay)

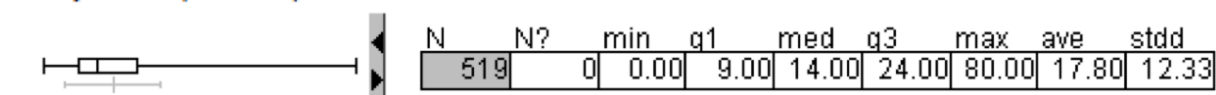
The following statistics have been obtained only from those flights where either the start time or the end time differed from the original.

4.4.1 CFMU flight delay statistics

Delay at start (minutes)

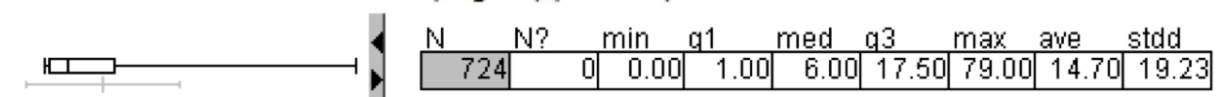


Delay at end (minutes)

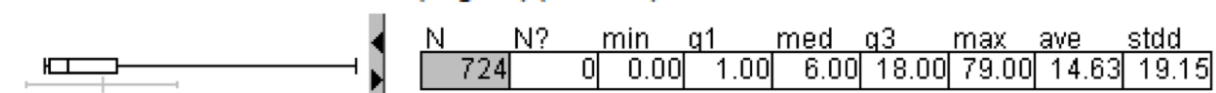


4.4.2 AgentBased flight delay statistics

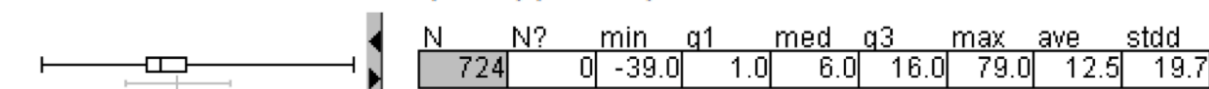
Start date+time - Start date+time (original) (minutes)



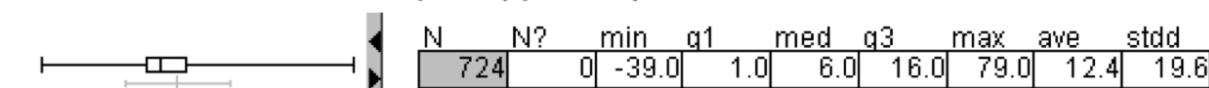
End date+time - End date+time (original) (minutes)



Start date+time - Start date+time (CFMU) (minutes)

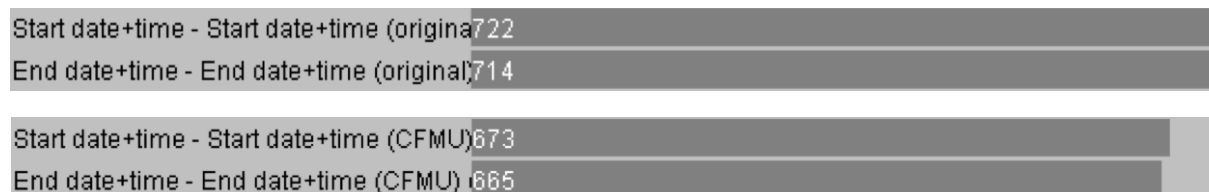


End date+time - End date+time (CFMU) (minutes)



4.4.2.1 Counts of the flights with positive differences to the original times (i.e., increases)

AgentBased:



The figures show that there are 673 flights whose starts are regulated by the AgentBased method more than by CFMU, and there are 665 flights whose ends are regulated more than by CFMU.

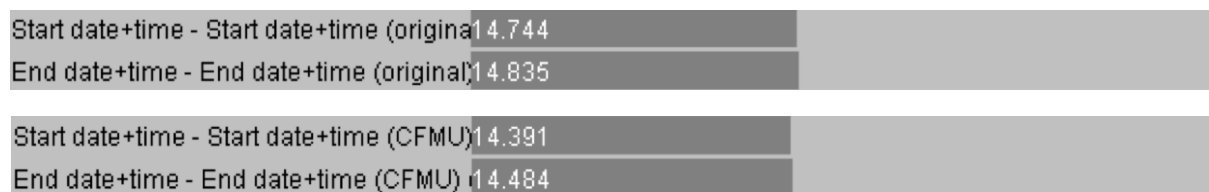
CFMU:



The AgentBased method introduces delays in a larger number of flights than CFMU.

4.4.2.2 Average positive differences on delays:

AgentBased:



CFMU:



4.4.2.3 Counts of the flights with negative differences to the original times (i.e. decreases)

AgentBased:

Start date+time - Start date+time (original)
End date+time - End date+time (original)

Start date+time - Start date+time (CFMU) 48
End date+time - End date+time (CFMU) 48

While the AgentBased method increases the number of regulated flights compared to CFMU, it reduces delays in 48 flights that were regulated by CFMU.

CFMU:

Delay at start (minutes)
Delay at end (minutes)

4.4.2.4 Average negative differences on delays:

AgentBased:

Start date+time - Start date+time (original)
End date+time - End date+time (original)

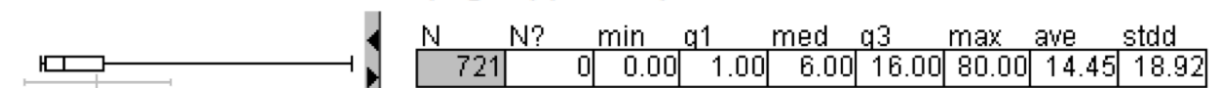
Start date+time - Start date+time (CFMU) -13.792
End date+time - End date+time (CFMU) -13.792

CFMU:

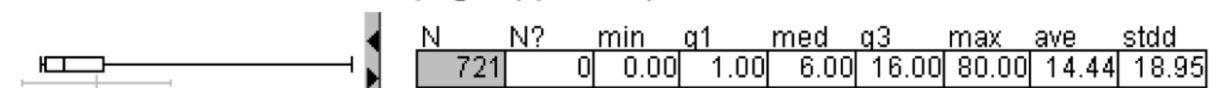
Delay at start (minutes)
Delay at end (minutes)

4.4.3 EdgeBased flight delay statistics

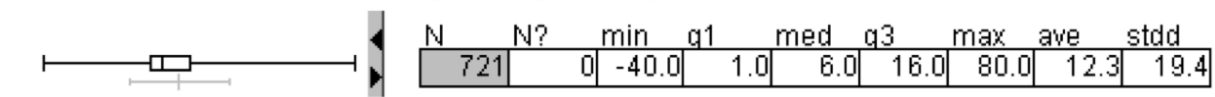
Start date+time - Start date+time (original) (minutes)



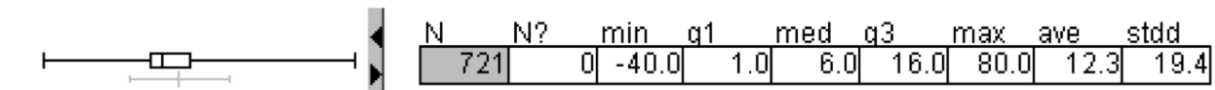
End date+time - End date+time (original) (minutes)



Start date+time - Start date+time (CFMU) (minutes)



End date+time - End date+time (CFMU) (minutes)



4.4.3.1 Counts of the flights with positive differences to the original times

EdgeBased:

Start date+time - Start date+time (original)	719
End date+time - End date+time (original)	711
Start date+time - Start date+time (CFMU)	676
End date+time - End date+time (CFMU)	668

The figures show that there are 676 flights whose starts are regulated by the EdgeBased method more than by CFMU, and there are 668 flights whose ends are regulated more than by CFMU.

CFMU:

Delay at start (minutes)	510
Delay at end (minutes)	518

The EdgeBased method introduces delays in a larger number of flights than CFMU.

4.4.3.2 Average positive differences:

EdgeBased:

Start date+time - Start date+time (CFMU)	14.037
End date+time - End date+time (CFMU)	14.193
Start date+time - Start date+time (original)	14.488
End date+time - End date+time (original)	14.640

CFMU:

Delay at start (minutes)	17.973
Delay at end (minutes)	17.838

4.4.3.3 Counts of the flights with negative differences to the original times and durations (i.e., decreases)

EdgeBased:

Start date+time - Start date+time (original)
End date+time - End date+time (original)

Start date+time - Start date+time (CFMU) 43
End date+time - End date+time (CFMU) 43

While the EdgeBased method increases the number of regulated flights compared to CFMU, it reduces delays in 43 flights that were regulated by CFMU (48 for AgentBased).

CFMU:

Delay at start (minutes)
Delay at end (minutes)

4.4.3.4 Average negative differences:

EdgeBased:

Start date+time - Start date+time (original)
End date+time - End date+time (original)

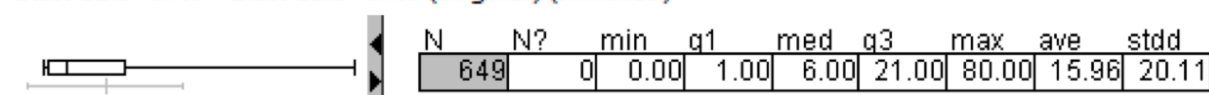
Start date+time - Start date+time (CFMU)	-14.279
End date+time - End date+time (CFMU)	-14.279

CFMU:

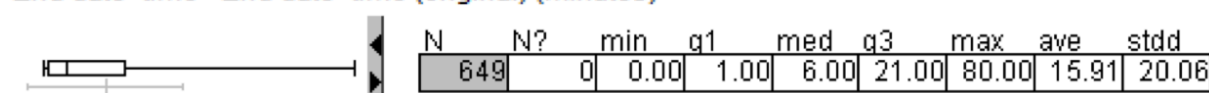
Delay at start (minutes)
Delay at end (minutes)

4.4.4 Hierarchical flight delay statistics

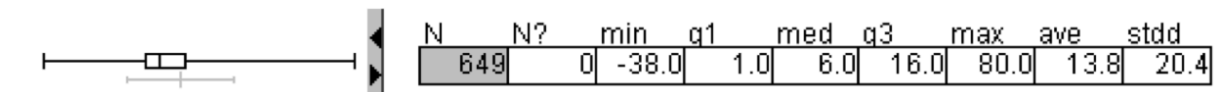
Start date+time - Start date+time (original) (minutes)



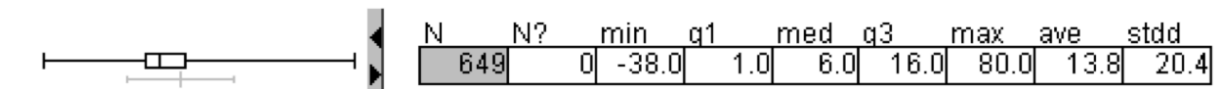
End date+time - End date+time (original) (minutes)



Start date+time - Start date+time (CFMU) (minutes)



End date+time - End date+time (CFMU) (minutes)



4.4.4.1 Counts of the flights with positive differences to the original times and durations (i.e., increases)

Hierachical:

Start date+time - Start date+time (original)	646
End date+time - End date+time (original)	642
Start date+time - Start date+time (CFMU)	607
End date+time - End date+time (CFMU)	603

The figures show that there are 607 flights whose starts are regulated by the Hierachical method more than by CFMU, and there are 603 flights whose ends are regulated more than by CFMU. These figures are better than for AgentBased and EdgeBased.

CFMU:

Delay at start (minutes)	510
Delay at end (minutes)	518

The Hierachical method introduces delays in a larger number of flights than CFMU but in a smaller number of flights than AgentBased and EdgeBased.

4.4.4.2 Average positive differences:

Hierachical:

Start date+time - Start date+time (original)	16.034
End date+time - End date+time (original)	16.083
Start date+time - Start date+time (CFMU)	15.603
End date+time - End date+time (CFMU)	15.652

CFMU:

Delay at start (minutes)	17.973
Delay at end (minutes)	17.838

4.4.4.3 Counts of the flights with negative differences to the original times Hierarchical:

Start date+time - Start date+time (original)
End date+time - End date+time (original)

Start date+time - Start date+time (CFMU) 39
End date+time - End date+time (CFMU) 39

While the Hierarchical method increases the number of regulated flights compared to CFMU, it reduces delays in 39 flights that were regulated by CFMU (48 for AgentBased, 43 for EdgeBased).

CFMU:

Delay at start (minutes)
Delay at end (minutes)

4.4.4.4 Average negative differences:

Hierarchical:

Start date+time - Start date+time (original)
End date+time - End date+time (original)

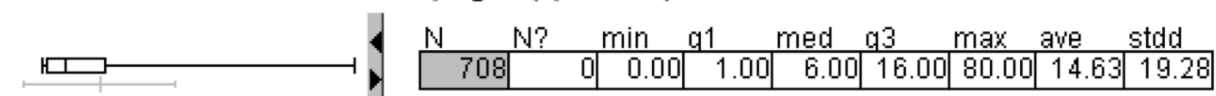
Start date+time - Start date+time (CFMU)	-12.590
End date+time - End date+time (CFMU)	-12.590

CFMU:

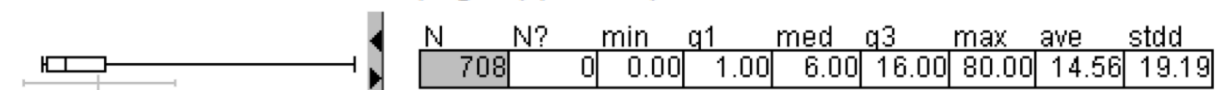
Delay at start (minutes)
Delay at end (minutes)

4.4.5 IndLearners flight delay statistics

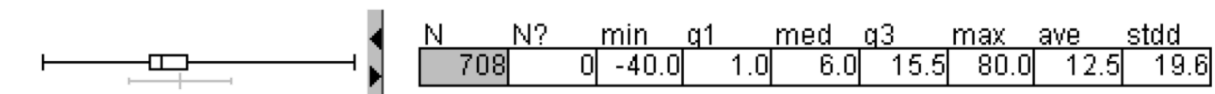
Start date+time - Start date+time (original) (minutes)



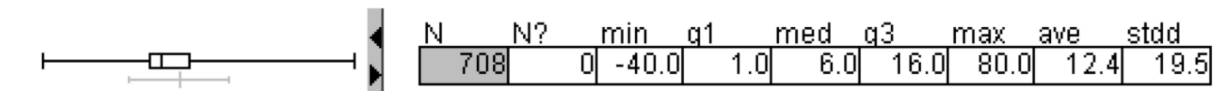
End date+time - End date+time (original) (minutes)



Start date+time - Start date+time (CFMU) (minutes)

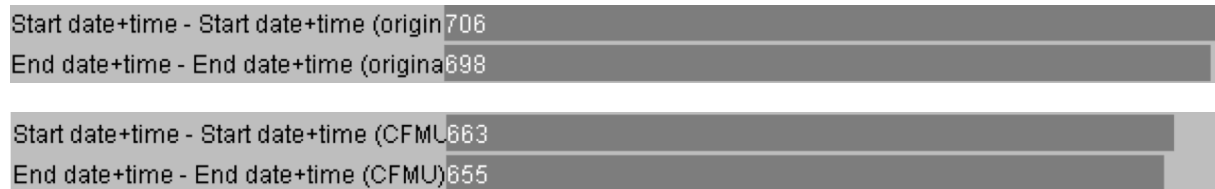


End date+time - End date+time (CFMU) (minutes)



4.4.5.1 Counts of the flights with positive differences to the original times and durations (i.e., increases)

IndLearners:



The figures show that there are 663 flights whose starts are regulated by the IndLearners method more than by CFMU, and there are 655 flights whose ends are regulated more than by CFMU. These figures are very close to AgentBased and EdgeBased and worse than for Hierarchical.

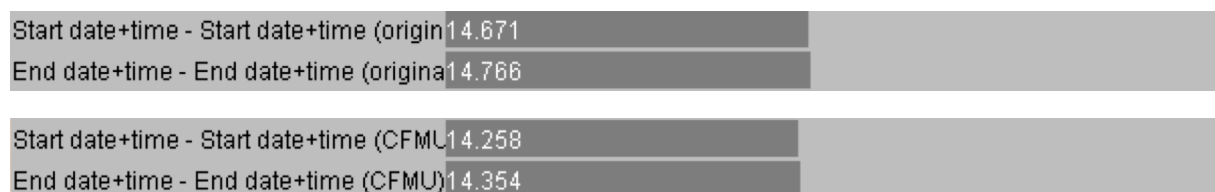
CFMU:



The IndLearners method introduces delays in a larger number of flights than CFMU.

4.4.5.2 Average positive differences:

IndLearners:



CFMU:



4.4.5.3 Counts of the flights with negative differences to the original times

IndLearners:

Founding Members

Start date+time - Start date+time (origin)
End date+time - End date+time (origina

Start date+time - Start date+time (CFMU)	42
End date+time - End date+time (CFMU)	42

While the IndLearners method increases the number of regulated flights compared to CFMU, it reduces delays in 42 flights that were regulated by CFMU (48 for AgentBased, 43 for EdgeBased, 39 for Hierarchical).

CFMU:

Delay at start (minutes)
Delay at end (minutes)

4.4.5.4 Average negative differences:

IndLearners:

Start date+time - Start date+time (origin)
End date+time - End date+time (origina

Start date+time - Start date+time (CFMU)	-14
End date+time - End date+time (CFMU)	-14

CFMU:

Delay at start (minutes)
Delay at end (minutes)

4.5 Comparison between the four methods and with the CFMU

4.5.1 Counts of regulated flights

4.5.1.1 AgentBased:

Start date+time - Start date+time (original)	722
End date+time - End date+time (original)	714

Start date+time - Start date+time (CFMU)	673
End date+time - End date+time (CFMU)	665

4.5.1.2 EdgeBased:

Start date+time - Start date+time (original)	719
End date+time - End date+time (original)	711

Start date+time - Start date+time (CFMU)	676
End date+time - End date+time (CFMU)	668

4.5.1.3 Hierarchical:

Start date+time - Start date+time (original)	646
End date+time - End date+time (original)	642

Start date+time - Start date+time (CFMU)	607
End date+time - End date+time (CFMU)	603

4.5.1.4 IndLearners:

Start date+time - Start date+time (original)	706
End date+time - End date+time (original)	698

Start date+time - Start date+time (CFMU)	663
End date+time - End date+time (CFMU)	655

4.5.1.5 CFMU:

Delay at start (minutes)	510
Delay at end (minutes)	518

5 Conclusions

The results reported for the four Collaborative Reinforcement Learning methods show their potential to solve effectively the DCB problems at the pre-tactical stage, assessing the delays of regulated flights. Specifically:

- They manage to find solutions to all cases – i.e. they do manage to regulate flights crossing an operational space in a day so as to resolve all hotspots.
- They manage to find solutions effectively: They do converge to solutions quite fast, few rounds after exploration, in most of the cases.
- They manage to reduce the average delay for the regulated flights considerably, compared to the average delay for the regulated flights reported by CFMU. The same holds for the average delay considering all flights.
- The Hierarchical method, has the potential to reduce significantly the regulated flights, and thus the average delay for all flights, compared to the other methods and of course CFMU. However, in its current implementation is not that efficient, and the average delay to the regulated flights is much higher than that reported by the other methods.

All methods, across the different evaluation cases, follow the same patterns for regulations prescribed and number of regulated flights: While the EdgeBased method manages to achieve a good and effective balance between the average delay to regulated flights and the number of regulated flights, the Hierarchical method manages to reduce significantly the number of regulated flights, while increasing the average delay to these flights. We need to delve into the details of that method in order to explore its potential to further reduce the average delay without increasing considerably the number of regulated flights.

A major issue that needs to be explored is “what constitutes a difficult case”? The evaluation cases characteristics shown in Table 1, do not seem to provide an answer: Jul2, being the most computationally demanding among the cases has a low average degree per flight, few hotspots with a low number of interacting flights. However, methods (as well as CFMU) need to search deep into solutions in order to reach one that resolved the DCB problem encountered.

Finally, all methods manage to incorporate, as one of their inherent features, airlines preferences to the delays of some of the flights: This is a significant issue, showing that our methods can contribute to prescribing solutions to DCB problems, taking into account stakeholders preferences and constraints.

Visualizations provide a comprehensive way to summarize results in space and time, while – and more importantly- they provide to a certain degree justifications of the “reasoning” behind decisions on



regulations: For instance, while aggregated results on average delays and regulated flights show the potential of the methods, delving into details of their efficacy requires inspecting the spatio-temporal distribution of capacity excess events and their intensity (shown in Figures 14-19), as well as the spatio-temporal distribution of delays (shown in Figure 24): These visualizations for instance provide justifications on the need for increased delays for the north-western Spain at the end of the day for July 2, 2016. These justifications are further backed up with the evolution of demand for that day (provided in Table 7, row “July 2”) for the most demanded sector (corresponding to the one indicated in red in Figure 14). These visualizations provide firm evidence that at the end of that day, that sector had large demand for a large number of periods. Nevertheless, such visualizations provide the means to compare solutions and guide human decision and/or preferences on solutions generated.

6 References

[Cook et al, 2015] Andrew J Cook and Graham Tanner. 2015. European airline delay cost reference values. (2015). <http://www.eurocontrol.int/publications/european-airline-delaycost-reference-values>.

[Guestrin, 2002] Carlos Guestrin. 2003. Planning Under Uncertainty in Complex Structured Environments. Ph.D. Dissertation. Stanford, CA, USA. Advisor(s) Koller, Daphne. AAI3104233.

[Kok et al, 2006] Jelle R. Kok and Nikos Vlassis. 2006. Collaborative Multiagent Reinforcement Learning by Payoff Propagation. J. Mach. Learn. Res. 7 (Dec. 2006), 1789–1828. <http://dl.acm.org/citation.cfm?id=1248547.1248612>