Dynamic Arrival Prioritization With Target Time Management and Deep Reinforcement Learning

Leonardo Caranti, Marta Ribeiro, and Marie Carré

***Abstract*— The European Air Traffic Management system is among the most complex systems in the world. Due to the dense nature of the European network, consequences of disruptions are often catastrophic. In particular, disruptions altering the expected flying time tend to pose great challenges to the arrival management of busy hubs. In response, EUROCONTROL released the Target Time Management (TTM) system, allowing airlines to issue Target Times of Arrival (TTA) even before depart. The TTM system helps hubs airports coordinate arrivals and departures. From the point of view of airlines, the advantage resides in being able to prioritize early arrivals of critical flights. Nevertheless, real-time prioritization is not trivial. Many studies have focused on this problem but with results limited to slot swapping in a tactical context. This is less effective compared to airlines having the ability to select a new slot at the pre-tactical level. This work covers this gap, allowing airlines to select the desired TTA even before departure. We use Deep Reinforcement Learning to create a dynamic arrival allocation model capable of prioritizing flights in terms of passenger connecting time, curfew performance, rotation delay, and fairness to other airlines. Additionally, the model is capable of adapting and react to the uncertainty in responses from the TTM. In the real-world, large anticipations in TTAs are often rejected. The model is tested with real data from SWISS International Airline. Results show an improvement of 5.9 minutes for critical passenger connection and 4.8 minutes for rotation delay versus a deterministic approach.**

***Index Terms*— Reinforcement learning, air traffic management, target time management, soft-actor critic, proximal-policy optimization, mixed-integer linear programming.**

I. Introduction

THE European Air Traffic Management system is one of the most complex structures in the world. Due to the dense nature of Europe and its network structure, the consequences of disruptions are often catastrophic. These affect particularly hubs which have a high level of inflow and outflow traffic. Hubs at the centre of Europe, a connection point for many flights, are especially affected in the case of disruptions.

Disruptions often lead to arrival regulations, leading to reducing the inflow capacity of a hub. In these occasions, EUROCONTROL will assign delays to all incoming aircraft, such that the new capacity limit can be met. The consequences of these delays on operations are disastrous. Unfortunately, passenger connections are missed, flights are cancelled, bags are lost, and the profits of airlines take a hit. As a consequence, airlines often try to prioritize critical flights, moving them ahead in the arrival order, taking the turn of other flights.

Research directed at arrival prioritization first started with slot-swapping, which entails swapping around the arrival order of flights [1], [2], [3]. The approach is often tactical and mostly targeted for long-haul flights, since it is possible to adapt the cruise speed to change the arrival order. However, for short-haul flights the tactical range is very narrow and needs to be expanded to pre-tactical to allow for more larger sequence changes. This is why EUROCONTROL released the Target Time Management (TTM) concept [4], which allows airlines and Air Navigation Service Providers (ANSPs) to submit their wished Target Times of Arrival (TTA) even before the flight themselves depart.

SWISS International Airlines, hereafter referred to as SWISS, was the first airline to test the TTM. With a hub in Zurich at the center of Europe, SWISS is a prime candidate for this system. In 2019 alone, SWISS had a total Air Traffic Flow Management (ATFM) delay of 4500 hours alone due to arrival regulations in Zurich [5]. Nevertheless, to properly take advantage of the TTM system, airlines require a model to decide upon the priority of flights. The first efforts encompassed an Integer Linear Programming (ILP) model to rearrange the arrival sequence [6]. While promising, an interesting founding was made during its testing. Requested TTAs sent to the TTM, controlled by EUROCONTROL, do not often matched the slot received. Additionally, the likelihood of a TTA being accepted depends on the type of request: TTAs which delay flights have a higher chance of being accepted by the system than anticipation attempts. As such, the need for a more dynamic and future-looking model is required.

Current research has often addressed this problem using linear, non-dynamic approaches. Studies that incorporate airline data to account for the dynamic nature of flight prioritization remain limited. This study addresses this gap by performing TTA using real airline network data, explicitly considering the uncertainty in TTA acceptance rates across different types of requests. The results provide insights into whether communication with the TTM can be enhanced through a dynamic decision-making process. This process must take into account the environment in which it takes actions to be able to optimize its operational performance. This study compares two Multi-Agent Deep Reinforcement Learning (MARL) algorithms, with the objective of improving performance in terms of passenger connections and rotation delays. The algorithms

Received 9 September 2024; revised 13 March 2025 and 21 April 2025; accepted 4 July 2025. Date of publication 28 July 2025; date of current version 3 November 2025. The Associate Editor for this article was X. Sun. *(Corresponding author: Marta Ribeiro.)*

Leonardo Caranti and Marta Ribeiro are with the Control and Operations Department, Aerospace Faculty, Delft University of Technology, 2629 Delft, Netherlands (e-mail: M.J.Ribeiro@tudelft.nl).

Marie Carré is with Operations Research and Air Traffic Management department, Swiss International Airlines Ltd., (SWISS), 8302 Kloten, Switzerland.

Digital Object Identifier 10.1109/TITS.2025.3589857

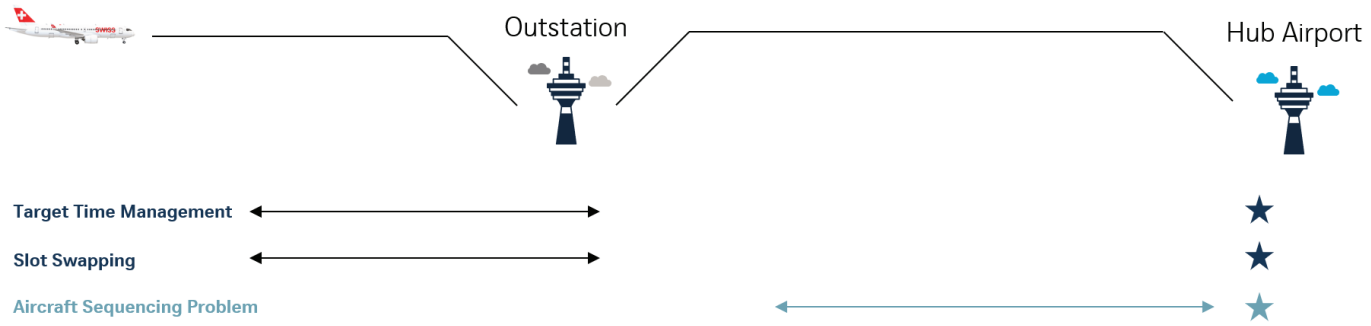


Fig. 1. Overview of all the problems/decisions that can be made prior to the beginning of the turnaround procedure that effect the flight prioritizing process and their timeframes. The arrows represent the timeframe in which the problem is solved, while the star represents the time(s) when the decision's outcome is realized.

are directly compared with the baseline ILP approach produced at SWISS.

This paper first starts with the literature review in Section II, where the literary gap is identified and the research question is outlined. The methods with which it will be answered are described in Section III. The results of the experiments are outlined in Section IV, and discussed in Section V. Finally, conclusions regarding the overall goal of the thesis and initial hypotheses are described in Section VI.

II. Literature Section

Figure 1 provides an overview of the three primary methods for flight prioritization. Instances that vary in terms of the duration of their resolution and impact are displayed here. Aircraft Sequencing Problem (ASP) was the first method to be explored [1]. It involves a sequence of flights while waiting to land. The aim of the ASP is to assigning a landing sequence that optimizes a specific measure for the benefit of the airline, air traffic control, or the airport. This broad definition has been applied in a variety of contexts, including fuel minimization [3], [7], delay [8], [9], deviation from the original schedule [10], and passenger connections [11], [12], [13]. Table I contains a summary of common objectives. However, the primary issue with the literature about the ASP is its lack of dynamicity: the priorities are shifted and the solution must be modified when a new flight joins the group that is already waiting. This, together with the lack of availability of airline-sensitive data, as stated in [11], was the main reason these approaches were not employed in operations [2], [3].

Compared to ASP, slot swapping is a more modern method of flight priority. Conceived by EUROCONTROl, it is intended to lessen the effects of situations in which the capacity for a certain airspace sector is exceeded by demand [18]. This is accomplished by adjusting each flight's departure time using a Calculated Take Off Time (CTOT), which modifies the arrival sequence. This method seeks to have zero holdings by design, in contrast to the ASP. As the order needs to be determined before the flight takes off, it means that the priorities are decided much earlier. However, it can only be used for short-haul flights within the European airspace. The work [19] is one of the most significant literary works in the context of slot swapping. This method attempts to adjust passenger connections and delays by automatically switching planes' slots, another term for CTOTs. However, their method only permitted switching between pairs of flights, which can result in a less-than-ideal order. The work [20] employs evolutionary algorithms and a privacy-preserving strategy to enable the sharing of sensitive data. Work [21] defines the Slot Swapping problem as a MILP that included only 15 turnarounds as a test case. Chen et al. [22] considers the preferences of airlines and passengers in the recovery process. Yang et al. [23] considers airline fairness in the slot allocation decision on a strategical level. Nguyen-Duy et al. [24] also developed reinforcement learning methods for aircraft scheduling. However, this work did not consider uncertainties in the acceptance of slot requests. Finally, it should be noted that all previous works cannot yet be implemented in an operational department, as more thorough evaluation of the network effects is necessary. Additionally, these are often at a strategical level. Furthermore, regrettably, neither author had access to sufficient airline data.

In turn, TTM, ideated by EUROCONTROL, is similar to slot swapping regarding timeframes and effects [4]. However, a series of differences. First, it allows for an unlimited number of swaps for the same flight (up to 30 per minute), to allow for dynamicity (for CTOTs to be revised). Moreover, slot requests can be sent a lot in advance (five hours instead of two hours like slot swaps). Besides, the way one requests a change of CTOT is via a TTA, which then EUROCONTROL calculates back to a CTOT. This is highly functional for the purpose of affecting the arrival order. The CTOT (and thus this arrival time) can be changed up to the start of the Departure Planning Information sequence, 10 minutes before the Target Start-up Approval Time [25].

TTM was first researched at SWISS, Skyguide, and Zurich Airport [6]. The goal was to modify the sequence of inbound traffic at Zurich Airport in order to optimize for passenger connections and rotation delays. This was discovered to be achievable with the employment of a ILP model [6], although it, too, suffered from a lack of dynamicity. EUROCONTROL's slot assignment algorithm does not always award the necessary CTOT due to shifting demand in the airspace. Furthermore, due to the linear and non-stochastic structure of ILPs, long-term network impacts, as well as future effects of present actions, were not completely taken into consideration in the decision-making process.

A. Research Gap

Taking into account the previously mentioned works, the following research gap is identified:

TABLE I
OBJECTIVES APPLIED IN WORKS COVERING THE AIRCRAFT SEQUENCING PROBLEM

Work	*Primary Delay*	*Reactionary Delay*	*Passenger Connections*	*Fuel*	*Difference to Schedule*
Pawelek (7)	✓			✓	✓
Ma (9)	✓				✓
Du (10)					✓
Cecen (8)	✓				
Cecen (14)	✓				
Hoogendoorn (3)	✓		✓	✓	
Vervaat(2)			✓	✓	
Ikli (15)					✓
Montlaur (11)	✓	✓	✓		
Ng (16)	✓			✓	
Furini (17)					✓

The inability to conduct a network-level effect analysis with precise airline data for a system that also takes into account the dynamic nature of flight prioritization. This study will cover this gap. As such, the research question is formulated:

To what extent can Target Time Management be improved by a decision-making process which takes into account a network-level dynamic environment?

While traditionally, similar problems have been solved with linear or with Dynamic Programming, they often lacked dynamicity [2], [6], [26]. It is important to find a method which can adapt to the dynamics of an environment (here, the slot allocation environment) and that can take into account future states. Preferably, this should be modelled using multi-agent systems, since in arrival management there are always a varying number of flights. As such, this study employs Deep Reinforcement Learning (DRL). The latter has the ability to adapt to complex environment dynamics and take into account future states [27]. Moreover, it is possible to structure multi-agent DRL systems, which are widely used within the complex air traffic management (or more broadly, aviation) domain, as reviewed in [28].

Finally, the research objective of this study is thus to employ DRL to send TTAs to the TTM system, taking into account the uncertainty in the acceptance rate of TTAs per type of request. The objective is to prioritize flights in terms of passenger connecting time, curfew performance, rotation delay, and fairness to other airlines.

III. METHODOLOGY

To ensure a method which attempts to answer the objective of this work, a step-wise approach is taken. This section describes this approach. First, the overall structure is defined and presented in Section III-A. Then, Section III-B describes the relevant data. The environment is described in Section III-C. This ensures a possibly realistic modeling of the real slot-allocation dynamics. Finally, the chosen RL algorithms are outlined in Section III-D.

A. Structure

Figure 2 displays the overall structure of the developed model. First, from the available flights, a list of active flights is selected. Flights closer to their arrival time are selected. Information about these active flights constitutes the state of the environment. The composition of the state is described in Section III-C3.

In response to the current state of the environment, the RL agent produces an action (Section III-C4), which represents the new TTAs requested for each active flight. These requests are sent to the simulated TTM system. The latter is a representation of the real EUROCONTROL response, where acceptances and denials are constructed based on historical data.

The acceptance or denial response is used to formulate the reward for the agent, as described in Section III-C5. The objective is for the RL model to learn to produce TTA requests that are more likely to be accepted, that decrease delay, respect airport curfews, and improve passenger connections and fairness between airlines.

Finally, after each action, the environment is moved forward a predefined td amount of time. In total, one day of operations is simulated before the process is terminated.

B. Data

The data used for the modelling of the environment consists of the following sources: daily schedule, passenger connections and connecting times, arrival regulation evolution, and response to TTAs. The first two are used both in the modelling of the environment as well as the building of the inputs of the decision-making algorithm. The period chosen for training the algorithm is from April 2018 until the end of April 2023, while May 2023 is used a validation set. This is a great validation set, since it includes some very high traffic days (due to vacations), as well as more routine, calmer days.

No preprocessing of the data was performed. However, days are filtered based on the quality of the data and their representativeness to situations where the tool would be used. Days with less than 100 connections (on both SWISS or non-SWISS arrival flights), or 100 flights, or less than 50 connections on the SWISS flights incoming to Zurich, were discarded. This also enables to discard the days within the COVID-19 period where travelling was low, while still maintaining the valuable change of passenger booking behavior and numbers. Note that days within the COVID-19 period where travelling is higher than 100 connections are still included in the training data.

Finally, both short and long haul flights are considered in the data, as both are important for the network-wide simulation of events (passengers often connect from a short-haul flight to a long-haul ones). However, in the scope of our work, the

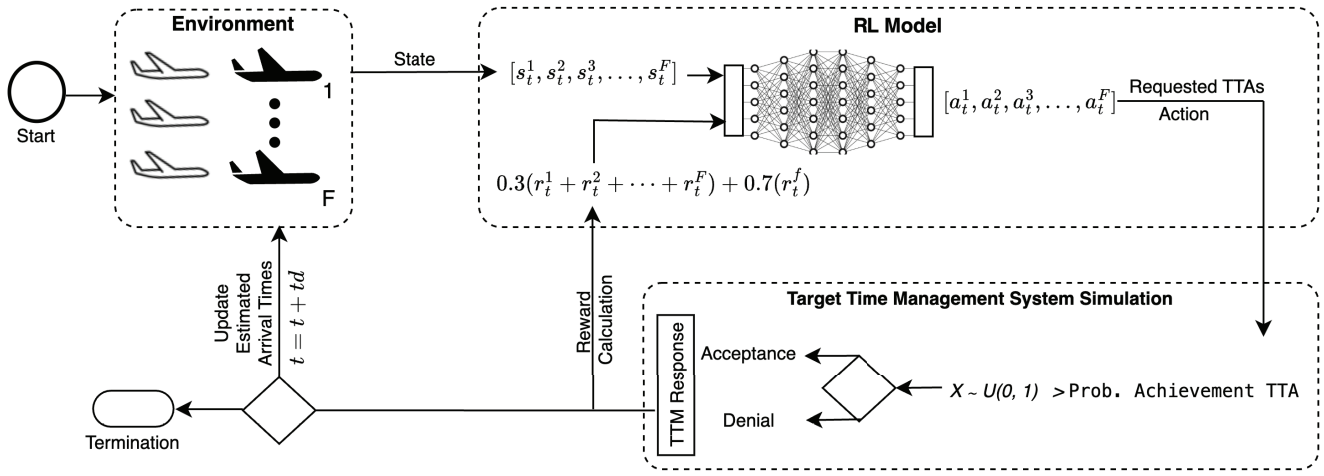


Fig. 2. General structure of the employed reinforcement learning model.

model only sends for TTAs for short-haul flights, as these are the focus of pre-tactical TTAs assessment.

C. Reinforcement Learning (RL) Implementation

RL is structured with one or more agents interacting with an environment, receiving rewards based on their actions. The agents, whose decision-making process is often based on neural networks, learn to maximize both current and future reward through training. To choose an action effectively, agent(s) must have sufficient information from the environment. The reward type allows for complex, non-linear reward functions, and also for complex environment dynamics.

In this section, each part that has to be formulated for a DRL algorithm to be trained is outlined and analysed. Firstly, the environment and its dynamics are analyzed in Section III-C1. How each agents is assigned to flights is described in Section III-C2, followed by what the state comprises for each agent in Section III-C3. Next, the approach to modelling actions in outlined in Section III-C4, finally followed by the modelling of the reward in Section III-C5.

1) Environment: The environment where the TTAs can be set through agent's actions represents the real operational environment of SWISS. Here, requests are made to EUROCONTROL, but, often, it does not assign slots based on the requests. As such, it is important to understand how to model this exchange in the best possible way. Furthermore, the stepwise increment in the simulated environment must be defined. Both of these two elements are described in this subsection. A timestep in the simulated environment is described in Algorithm 1.

a) Environment response regulation behaviour: The slot assignment algorithm by EUROCONTROL which receives the requested TTAs tends to be highly dynamic. The location within the step function where these dynamics are included is in line 11 of Algorithm 1. The slot allocation dynamics are modelled in two parts:

1) Find the probability that the requested TTA will be exactly accepted.
2) In case it was *not* accepted, find the new received arrival slot instead of the requested one.

Algorithm 1 A Timestep in the Simulation Environment

1: **Input** d (normalization value for each action a), current time t, list of active flights F, current state S (shape $n \times |F|$ with n the number of inputs per agent).
2: F_{orig}:= original set of active flights before actions
3: STA_f:= Scheduled Time of Arrival of flight f
4: **for** each flight f in F **do**
5: a_f:= Sample RL action, based on current state S.
6: TTA_f:= $\max(STA_f,\ STA_f + a_f \cdot d)$
7: **end for**
8: $t = t + td$
9: Compute response by EUROCONTROL (slot allocation algorithm):
10: Separate delayed flights from anticipated ones
11: p:= sample probability of achievement of the TTA
12: x:= sample from $X \sim U(0, 1)$
13: If $x > p$:
14: Find new TTA for the flight, not achieved
15: Else:
16: Accept TTA
17: Update estimated arrival times
18: Get rewards by comparing to original schedule
19: Update schedule:
20: F:= New active set of flights if any
21: $F_{dep} \in F_{orig} \setminus F$ (flights in F_{orig} but not in F because they have departed)
22: Remove agents assigned to departed flights F_{dep}, assign them to other new flights.
23: S:= new observations, making up the new state
24: **Output** S

First, the acceptance probability of a TTA was made a function of two variables, a discrete one and a continuous one: whether a flight is delayed or anticipated, and the minutes elapsed since the TTA was requested, respectively. Data was collected between July 1^{st} and 15^{th} September, 2023 to determine the distribution of these two variables. The minutes elapsed were fitted into an Empirical Cumulative Distribution Function

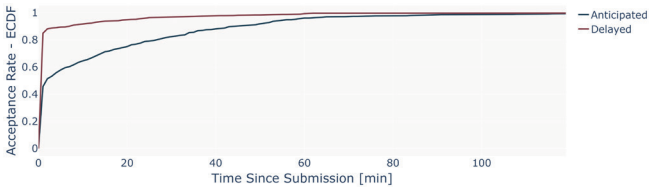


Fig. 3. Empirical cumulative distribution function for probability of acceptance of a TTA, fitted on data from the 1st July to the 15th September 2023.

(ECDF), as shown in Figure 3. It is necessary to remember that the flights' arrival times can be affected from around five hours prior to departure until off-block time. It is possible to see that anticipated flights tend to take significantly more to achieve their TTAs. Within the first minute, about 45% of the anticipated flights achieve their TTA, and about 85% of the delayed flights achieve it as well. However, to get to 95% of the flights achieving their TTAs, it takes 22 minutes for delayed flights, and 58 minutes for anticipated ones. It is expected that the DRL model will learn to compensate this imbalance by requesting more arrival anticipations than delays.

To find the new TTA, in case the requested one was not achieved, two functions have to be fitted. Each function once again depends on whether the flight has been delayed or anticipated, and on how much time has elapsed since the TTA was requested. For each new TTA to be sampled, a mean and standard deviation were taken around a specific point. This point was chosen as the Divergence Factor from the TTA requested, mainly the delay given divided by the delay requested. The ratio is always positive or zero, since in the case of an anticipation, the delay requested is negative. This divergence factor, like previously mentioned, is assumed to be a normal distribution at each point in time, and as such, the mean and standard deviation are fitted. In Figure 4, the mean divergence factor is presented. Here it can be seen how within the first 20 minutes, delayed flights tend to have more delay than requested (factor is greater than one), while anticipated flights tend to have less anticipation than requested up to 120 minutes (factor is less than one). The standard deviation plot describes the confidence level of each value. It can be seen how the standard deviation is decreasing from high to low for delayed flights, while it is increasing from low to high for anticipated flights. This might be because when a flight is delayed, it will stay in the system longer and thus have more datapoints, making the deviation lower. The amount of datapoints after 90 minutes since the TTA request decreases greatly, causing a series of sharp jumps due to a coarser mesh of data. This could be solved by having more data, or by applying a spline to smooth out the curve. In truth, the effect of this is minimal since the probability that a flight stays more than 90 minutes without a new TTA request are equally small.

Finally, each flight subject to a regulation must be delayed. The delay is static and calculated once. For this, simple statistical distributions were used, based on realistic regulation values. The regulations chosen were with a mean delay varying between 20 minutes and 50 minutes. Each simulated day, a random value between 20 and 50 is sampled, and set as the average delay. The delays are assumed to be normally distributed around this value, and each individual flight's delay is sampled from a normal distribution. These values were chosen with SWISS and EUROCONTROL operation representatives which deemed them not only realistic but operationally challenging (arguably more challenging than the average arrival regulation in Zurich). The result obtained when this set of distributions is simulated for 500,000 days with 300 flights per day is shown in Figure 5.

b) Environment step behaviour: An episode of the environment represents a day of flights. Each environment starts with the first flights of the day. At every step, time t is increased by a timedelta td, and the response by EUROCONTROL's slot allocation algorithm is simulated. Then, rewards are computed and, finally, the new active flights are found. After, the next state can be built and returned by the function. For the DRL, the timestep td was set to 30 minutes. At each time step, the RL algorithm is sampled for each flight and an action is retrieved until the end of the day.

Note that this timedelta differs with the ILP at SWISS with which the developed DRL model is directly compared. The ILP timestep is set to 60 minutes in operations. However, as DRL considered future-rewards, more frequent actions are expected to create more opportunities for improvement.

2) Agents: The core of RL requires a Markov Decision Process (MDP). This, in practice, is often modelled with an environment where the agent makes action based on a state, and receives a reward from it. In this case, the interest lies in modelling each action as either an anticipation or a delay of a flight. This means that each agent has to represent one flight. The total number of flights in one day is large, yet the number of active flights which can receive a TTA at the same point in time is still limited. Thus, agents will be reused when a flight becomes inactive. A flight is considered active three hours before scheduled time of departure (at the outstation), and is considered inactive once it has off-blocked, departing towards the hub in Zurich.

The number of agents available is set to the maximum number of active flights which could receive a TTA within the last five years - 60 flights. Not all sixty agents are activated at the same time. Whenever a flight is activated, it is connected to the DRL model through the next available 'agent slot' in the model. When the flight is deactivated the agent is detached from it. This way, the limited agent slots in the DRL model can be used for an unlimited number of real flights. Additionally, this implies that there is no significant difference between agents. This is because, although each flight may have different inputs to the agent, their behaviours do not differ greatly. From a decision-making perspective, a flight coming from Amsterdam should not behave differently to a flight coming from Rome, if all inputs remained the same for both flights. This is why reassigning agents to flights once they depart is possible, and this is also why training all agents on the same RL policy is a valid choice. By doing this, training time decreases as the complexity is lower.

3) State: Agent coordination is crucial for the flight prioritization problem: since in this environment setup, each flight could theoretically always ask for an anticipation (which is in most cases beneficial for an inbound Zurich flight), the agent has to learn when to prioritize other flights, and thus has to

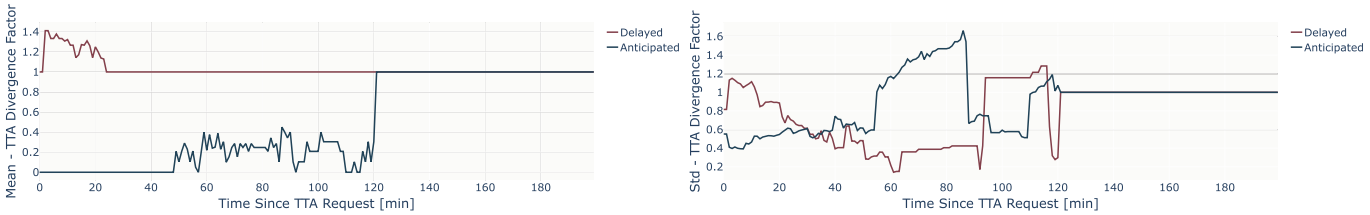


Fig. 4. Mean and standard deviation of the divergence factor (ratio) between the requested TTA and the actual one.

TABLE II

STATE VECTOR IN THE ENVIRONMENT. SOME OF THE PARAMETERS ARE AN AVERAGE ACROSS THE SET OF ACTIVE FLIGHTS, AND THESE ARE HIGHLIGHTED WITH A CHECKMARK (✓). THIS BRINGS THE TOTAL TO 15 INPUTS, INSTEAD OF 10

Input	*Average*	*Variable*	*Explanation*
Delay Credit		x_0	Total amount of delay credited since start of the day.
ATFM Delay	✓	x_1	Delay due to an arrival regulation.
Original ATFM Delay	✓	x_2	Initial ATFM delay prior to any action.
Other Delay		x_3	Delay due to rotation of delayed aircraft.
Flight Time		x_4	Predicted flight time at time of action.
Time of Day		x_5	Time of day in UTC.
Number of Critical Connections	✓	x_6	Weighted number of passengers with $\leq$45 minutes.
Average Connecting Time	✓	x_7	Average connecting time of all passengers.
Minimum Connecting Time	✓	x_8	Minimum connecting time between all passengers.
Time Since Last TTA Sent		x_9	Elapsed time since the last TTA for this flight was sent.

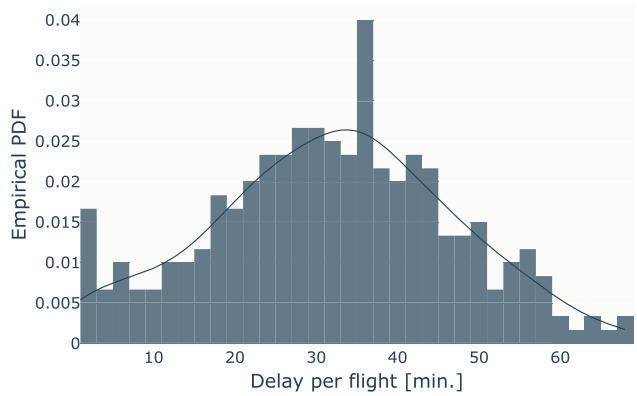


Fig. 5. Stochastic simulation of aircraft delay for 500,000 days with 300 flights per day.

be informed about their states as well. The state, with inputs described in Table II, includes elements of delay, connections, and time. Besides a few simpler inputs like the current time of the day or the flight time, there are a few specific ones which should be further described. For instance, the concept of delay credit is defined as the total amount of delay minutes that SWISS has gained or lost since the start of the actions by the tool. On the other hand, critical connections are defined as having less than 45 minutes connecting time (chosen in accordance with Operations Controllers at SWISS). Formally, the state s at time t is defined as follows:

$$S_t = [s_t^1, s_t^2, s_t^3, \ldots, s_t^F]. \tag{1}$$

The size of the state formulation equals $n \times F$ where F represents the list of active flights. The list of inputs per agent, n, is defined as:

$$s_t^f = [x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9]. \tag{2}$$

Variables $x_0 - x_9$ correspond to the elements in Table II. The state has a fixed size of 10 elements.

4) Action: The action by RL algorithm at time t is formulated as follows:

$$A_t = [a_t^1, a_t^2, a_t^3, \ldots, a_t^F], \tag{3}$$

where each element a represents the action for each active flight, 1 to F. The actions chosen by the algorithms translate into new arrival times. The way this is done is by setting the output of the algorithm to the requested delay with respect to the current slot, scaled by a factor of 30. This means that an action of -1 is set as an anticipation by 30 minutes, an action of 0 is set to no change with respect to the current slot, and an action of 1 is equivalent to a delay of 30 minutes. Even though this may seem limiting in extreme cases where anticipations of more than 30 minutes are required, since the tool recomputes every 30 minutes, flights can typically receive TTAs for more than 2 hours, the 30-minute threshold is not expected to be limiting. It was also the threshold recommended by Network Operations Controllers at SWISS. Of course, there are a few constraints to be taken into account when choosing the TTA: the tool cannot anticipate the flight before its scheduled arrival time, and as such, the requested anticipation must reflect this: the value taken is always the maximum between the computed arrival time and the scheduled arrival time.

5) Reward: The aspects of the reward considered in this tool as passenger connecting time, delay, curfew and inter-airline fairness. For each aspect, the reward is split both in a global part and a local part. Each flight receives a weighted sum between these two parts, with a global and local weight being 0.3 and 0.7, respectively. These values were empirically determined to lead to the best balance between convergence and operational performance in all metrics was found. The local reward is the computed reward per agent, while the global

TABLE III
NORMALIZATION PARAMETERS IN THE REWARD FUNCTION

Scaling Parameter	*Value*	*Unit*
Passenger Connections	45	$pax * min$
Rotation Delay	3	min
Curfew Flights	4	$flights$
Fairness (Delay Credit)	6	min
Global reward	0.3	-
Local reward	0.7	-

one is just calculated by taking the average between agents. The reward at time step t, for each flight f, is formally defined as:

$$R^f = 0.3(r_t^1 + r_t^2 + \cdots + r_t^F) + 0.7(r_t^f), \tag{4}$$

where r represents the reward for each active flight, 1 to f. The global reward is the sum of the reward for all flights. r_f represents the local reward for flight f, defined as:

$$\begin{aligned} r^f = & \begin{cases} \dfrac{pax \times \Delta connection}{45}, & \text{if previous conn} \\ & < 45 \text{ m or new conn} < 45 \text{ m} \\ 0, & \text{otherwise} \end{cases} \\ & + \begin{cases} \dfrac{\text{new delay - old delay}}{3}, & \text{new delay} < \text{ old delay} \\ 0, & \text{otherwise} \end{cases} \\ & + \begin{cases} -4, & \text{if curfew exists and is infringed} \\ 0, & \text{otherwise} \end{cases} \\ & + \begin{cases} \dfrac{\text{new delay - old delay}}{6}, & \text{new delay} > \text{ old delay} \\ 0, & \text{otherwise} \end{cases} \end{aligned} \tag{5}$$

where $\Delta connection$ represents the new connection time minus the previous in minutes. More information on each element of the equation is given in the following paragraphs.

Moreover, the normalizing or scaling factors for each reward segment are shown in Table III. These numbers also show the apparent equivalence rations of each reward type when compared: for instance, 3 long haul passengers with a short connection (below 45 minutes) being delayed 15 minutes is equivalent in terms of environment reward to decreasing the (already short) turnaround of a flight by 3 minutes.

Note that all the rewards are negative and flights are penalized when performing poorly, but not directly rewarded when performing well, with the exception for passenger connections and delays. This is expected to lead to a successful result, since the presence of strong trade-offs is also expected. For instance, the strongest trade-off expected is between fairness and all the other reward elements. Also, a strong trade-off between global and local reward is expected. Finally, a weaker trade-off between passenger connections and rotation delay performance is also expected. Nevertheless, the measurability of these trade-offs might be complex. Each element of the reward formulation is further developed in the following.

a) Passenger connections: The tool is penalized every time a passenger's connecting time is decreased, if the new connecting time is less than 45 minutes, and rewarded whenever the connecting time is increased, if the old connecting time is less than 45 minutes. We only considered connections which either used to be less than 45-minutes before the action of the DRL model, or which are less than 45 minutes after the decision of the model. For every connection, the change (new connecting time minus the old one) in minutes is multiplied by the number of passengers on the connection, yielding a value in $[pax * min]$. In order to obtain the passenger reward, this value is then multiplied by a scaling factor, which weights this with respect to other reward types. For this reward type, the scaling factor used was 45 $[pax * min]$: the logic used was to normalize the value based on 3 long haul passengers being delayed by 15 minutes.

As previously explained in Section III-C3, each passenger connection is weighted. This same weighting also applies to the computation of passenger connecting times in the reward function. For each connection and respective connecting time, this weight is multiplied by the number of passengers in the connection, to then be used in the reward.

b) Rotation delay: Let us imagine a situation where a flight lands with a delay and has to depart again from the airport soon after. If the turnaround time is greater than the minimum one, and if the crew is as fast as possible, the flight could depart with a lower delay than when it landed. The delay which is 'inherited' from the previous flight is defined as rotation delay. The concept of Target Time Management could also be used to optimize for these kind of delays, prioritizing flights with tight turnarounds. This is why a positive reward is given to an agent when its decision decreases its own rotation delay, and viceversa. The reward is computed by dividing the change in minutes by 3 $[min]$. This would entail that in this setup, one flight worsening its turnaround time by 3 minutes is equally detrimental as 3 economy long-haul passenger worsening their short connection by 15 minutes.

c) Curfew: Zurich Airport has a night ban. No flights are allowed to depart after 23:00 local time, and no flights are allowed to land after 00:00 local time. Moreover, for flights between 21:00 and 00:00 there are significant fines for both landing and taking off, which increase exponentially as the night creeps in. This is why a penalty (negative reward) is given to all flights arriving after 23:00, as even one hour before the theoretical limit, the fines are very harsh. Whenever an agent delays a flight beyond 23:00 local time, it is given a reward of -4 (per flight), while it receives a reward of zero for landing before curfew.

d) Fairness: While only SWISS flights are modelled within this thesis, fairness with respect to other airlines is key for this research. This is because, in case of any possible operational implementation, EUROCONTROL would only accept a tool which considers this aspect. In short, it is essential for the actions of each agent to not systematically disadvantage one or more airlines. The way this was often done previously was to only allow swaps as actions between SWISS flights. However, due to real-world environment dynamics within the slot allocation process, this actually brings SWISS to systematically disadvantage themselves. As such, in this approach the fairness will be quantified in an attempt to maintain a

comparable level of average delay between SWISS and other airlines over one day of operations.

Whenever a flight ends up with more delay than it had initially, the difference is stored in the form of credits. The opposite is also true: when a flight has been anticipated, credits are taken away. These delay credits can take up negative values. If an agent makes an action which greatly impacts the delay credits in a negative way, it should receive a penalty for this. The opposite is not true: if an agent decides to delay all of the flights, it should learn that it is a bad idea to do so for operational reasons, not due to fairness, and as such no fairness penalty is given. Whenever an agent tries to anticipate itself and the delay credit is already negative, it should be penalised. The magnitude of the penalty should be proportional to the change in the delay credit caused. As such, the reward is the total minutes of delay credit delta divided by a factor of 6 minutes. Again, this number was chosen by a combination of discussion with operations controllers (see Table III for the equivalence between reward types) and effects on training performance of the algorithm (evaluated using operationally valid metrics).

D. RL Algorithms Chosen

The Proximal Policy Optimization (PPO, originally developed by OpenAI [29]) and Soft-Actor Critic (SAC, originally developed at UC Berkeley [30]) algorithms were chosen. The following paragraphs first give insight into the different characteristics of the model and then dwell on the reasoning behind the selection of these two algorithms. Note that PPO is built according to the airline's operational guidelines, while SAC serves as a baseline to assess whether these preferences constrain performance.

PPO is an on-policy gradient method, learning directly from the experiences it gathers from the environment. PPO is considered a simpler algorithm, because it only optimizes one policy by learning a function that maps states to actions. Its defining feature is its clipping parameter; PPO prevents drastic policy updates by using a clipped surrogate objective function, which limits the change in policy updates. PPO is widely used due to its stable training, as this clipping mechanism prevents considerable changes [31]. However, PPO does not actively explore the environment, which can be problematic in environments with sparse rewards.

SAC is an off-policy method; the agent learns from stored data encountered in past episodes that are stored in the replay buffer. SAC optimizes both the actor's policy and the critic's. The central feature is its entropy regularization; it tries to maximize the balance between the expected return and a measure of randomness in the policy. SAC uses past experiences and the entropy term to ensure diverse action selection.

The main advantages of off-policy algorithms like SAC are related to a better exploration-exploitation balance and higher sample efficiency [30]. Exploration in SAC is improved compared to older approaches with Entropy Regularization, which maximizes the trade-off between expected rewards and entropy, which is a proxy for the exploration-exploitation trade-off [30]. In contrast, PPO was the least successful RL algorithm when it came to complex, dynamic environments [32] - here off-policy algorithms have the upper hand. However, all these benefits of off-policy algorithms often come at the cost of a longer training time. Off-policy algorithms typically make use of past data stored in a replay memory. The data in the replay buffer contains transitions from older policies. Past research has shown that this creates a distribution mismatch between training data and the current policy, which increases error. Due to this error, off-policy methods need more gradient updates leading to longer training times [33], [34], [35].

Both PPO and SAC are, in their base definition, stochastic in nature. However, in this study, an important modification is made: a deterministic version of PPO is used, where the action with the highest probability is always selected. This choice aligns with the airline's operational preference for stability, predictability, and simplicity. However, a critical question arises of whether a deterministic approach limits the operational performance of the RL model, as it reduces exploration and increases the risk of local optima. To investigate this, the deterministic PPO is compared to the stochastic SAC, which is selected for its ability to enhance exploration through entropy maximization. This exploration is particularly valuable in noisy or unpredictable environments [36].

Additionally, a multi-agent system is favorable for the problem at hand, as it has a better ability to achieve cooperation between flights compared to classical RL. This is also why these two algorithms were chosen, since they can be incorporated within multi-agent RL. For analysing the results, it is important to clarify the difference between episode and iteration. This can be understood from the algorithmic training setup defined in Algorithm 2.

Algorithm 2 Simplified Process to Train PPO and SAC in the Reinforcement Learning Environment

```
for each iteration i in I do
    for each episode e in E do
        Simulate one day by performing actions in the environment.
    end for
    Update learning policy with days simulated in this iteration.
    Simulate one evaluation day to test the updated policy.
end for
```

Finally, each reinforcement learning algorithms has hyperparameters. These are tunable values which change certain behaviours of the algorithms during training. A set of hyperparameters for SAC and PPO was chosen by manual tuning as well as taking values close to the advised ones in their original papers [29], [30]. The chosen hyperparameters can be seen in Table IV. The same learning rates and discount factors are applied to PPO and SAC. These defined how much the policy is adjusted based on new experiences and how much future rewards relative are weighted relatively to immediate rewards, respectively. The batch size was defined based on the best values found empirically for the models. The batch size of

TABLE IV
HYPERPARAMETERS FOR PPO AND SAC

Hyperparameter	Value	
	PPO	SAC
Learning rate	$5 \cdot 10^{-5}$	$5 \cdot 10^{-5}$
Discount factor	0.9	0.9
Training batch size	4000	256
Optimizer	Adam (40)	Adam (40)
No. layers	2	2
Layer size	256	256
Nonlinearity function	ReLU	ReLU
Clipping parameter	0.3	-
KL Coefficient	0.2	-
KL Target	0.01	-
GAE λ	1.0	-
Initial α	-	1.0
Target smoothing coeff. (τ)	-	$5 \cdot 10^{-3}$

PPO is considerably higher than of SAC. This is due to their differences in training. PPO is on-policy, thus training on the most recent batch of data. The larger batch size can stabilize training and prevents excessive policy updates [37]. SAC is off-policy and stores past experiences in the replay buffer. Large batches are not necessarily since it continuously learns from past experiences. Particularly, smaller batches may make SAC more sample-efficient [38]. Additionally, the employed PPO uses both clipping and Kullback-Leibler (KL) divergence to further stabilize training. The former prevents large policy updates in an update, while the latter is an additional constraint that monitors how much the policy diverges from the old policy.

IV. RESULTS

First, the results of training of the two DRL models are shown in Section IV-A. The testing results are presented in Section IV-B and compared with the baseline ILP model built at SWISS with the same objective in the simulated environment. Finally, in Section IV-C, the models are tested in a real-world situation. For this, the tool will be connected to EUROCONTROL's Pre-Operational Computer Assisted Slot Allocation (CASA) environment, and the models will be able to send Target Times of Arrival directly to CASA via a B2B connection. While this is not a fully-operational environment, it uses fully operational data and real flights, but the actions do not result in real flights being delayed or anticipated. However, the slot change due to the TTA is the same that would be in a real operational environment. Hence, the performance of the tool would also be very similar to reality.

A. Training Results

During the training procedure, both PPO and SAC were run on data from 9^{th} May 2018 until 20^{th} April 2023. This data was split into a 80%-20% training-evaluation split. The splits were randomly sampled from the available training days, since allowed both sets to include more difficult days to optimize, such as the COVID-19 peak in 2020, as well as more operationally-typical days in 2019 or 2023. At each iteration, the evaluation dataset was used to check the performance of the algorithm on untrained data. Then, the data from the 20^{th} April 2023 until the 26^{th} May 2023 was used as further testing for comparison of the DRL algorithms with the ILP model at SWISS. First, PPO was trained and set as the target for the new tool. SAC is used to see whether PPO has improvement potential under this environment, but PPO will be prioritized due to its deterministic behaviour and simplicity. These two factors likely contribute to its probabilities of being deployed into operations at SWISS and as such, depending on its performance.

As seen in Figure 6, the total training reward increased in an inverse exponential trend, up to a flat region at around -100. If one looks carefully at the numbers, it is possible to notice how both the training and the evaluation reward do not increase after iteration 100, and that is why it is said to have converged by then. The policy at this episode will be taken as the final trained policy. The total training time was 46 hours on a 64 GB RAM, 16-CPU i9 core (with multi-threading). On the other hand, the training time for SAC was significantly longer (as expected, due to more exploratory behaviour), taking in total around 90 hours on a 128-cores, 512 GB RAM machine. Additionally, the total number of iterations trained is around 15 times more, but it only took around twice as much due to SAC's higher sample-efficiency (hence requiring less episodes per iteration), and the more powerful computer used. Even though the final reward was slightly lower (around -140), the reward tends to grow slower over time. While PPO optimizes the first best policy it can find, SAC has a stronger focus on exploration, hence sometimes deviating from the best policy.

What is interesting to look at is how the reward evolves during training. As explained in Section III-C5, the reward function has various parts and thus it is important to see the evolution of each one of those parts. This is shown in Figure 7. It can be first observed that the overall shape is similar, meaning that the dynamics with which they learnt are also similar. Both algorithms start out by learning how to improve passenger connections, rotation delays, and curfew performance. The way they do this is by anticipating every aircraft, which results in a very unfair process. This is why, in both cases, the average fairness reward dips very low. Nevertheless, besides being an easy first step, this increases the initial reward, and is still better than doing no action. After this, the tool starts to understand the delay credit parameter, and tries to limit the sharp drops in fairness during a simulation day. This causes the local fairness to increase slowly back up.

Some interesting observations can be made regarding the differences between PPO and SAC reward evolution during training. For instance, it seems to be that SAC values global rewards more than PPO does. While the local rewards are lower, the global reward seem to be higher. This is most evidently true for fairness (the green and black line are further apart from each other in SAC), but also notable for passenger connections, rotation delay and curfew. Moreover, it seems like the local passenger connections line (one on top, red) for SAC keeps increasing over training, along with other curfew, and delay rewards. This implies that towards the end of training, when optimizing for fairness, the model also found a way to keep improving other operational metrics like local reward simultaneously. On the other hand, the passenger local

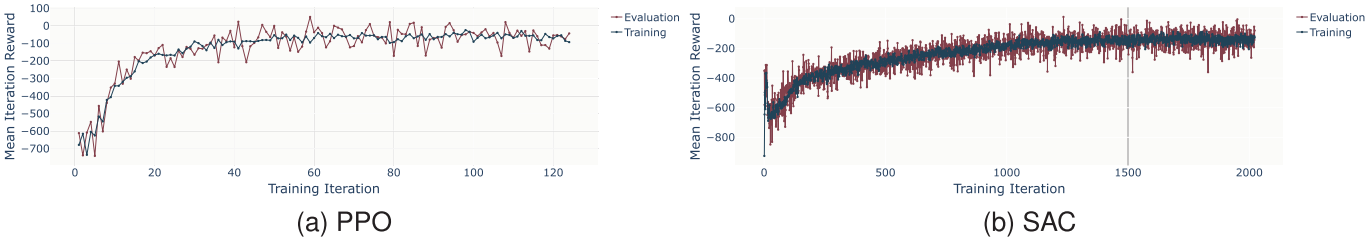


Fig. 6. Training overview of the DRL algorithm.

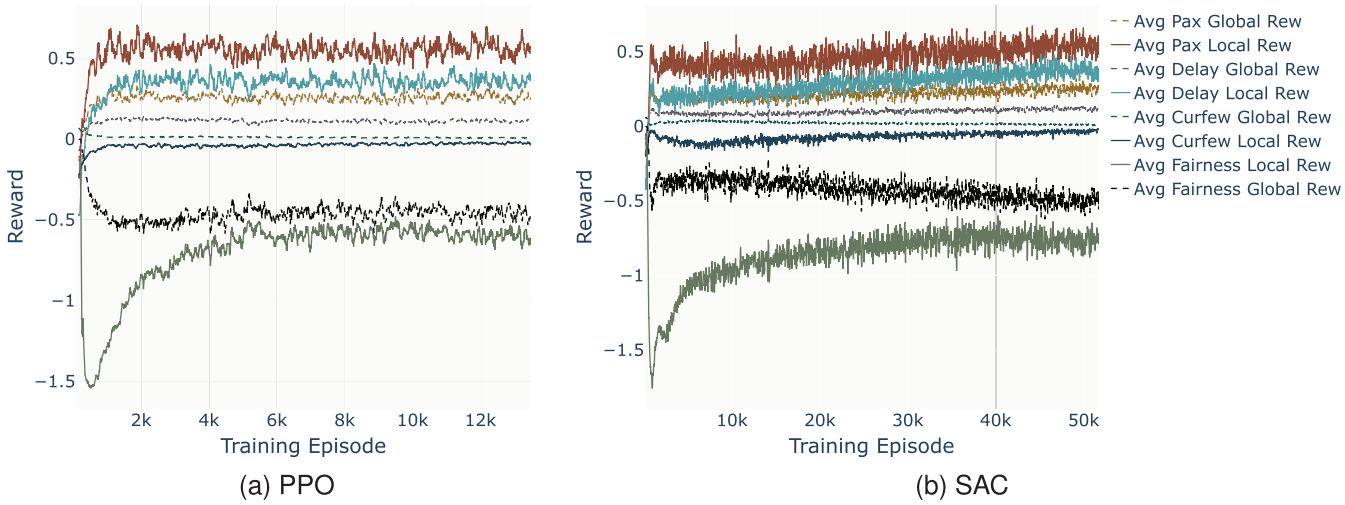


Fig. 7. Evolution of each reward element during training for the DRL algorithms.

reward for PPO seems stable after the initial peak. While PPO optimizes the initial best policy and sticks to it, SAC tries to keep exploring for new, better policies.

The action range also describes the behaviour of the algorithm. In this case, the action is allowed to be between -1 and 1, where a negative value of -1 corresponds to an anticipation of 30 minutes. As can be seen in Figure 8, the action range changes significantly during training. The overall pattern is similar for both algorithms: the algorithms tends to initially anticipate all the aircraft quite aggressively, then slowly optimizes for fairness by avoiding excessive anticipations, bringing the mean action closer to zero, and by narrowing the action range. Nevertheless, there are two significant differences between the two algorithms. First of all, SAC has a wider action range, meaning that it, on average, anticipates and delays more aggressively. On the other hand, the mean value for PPO is always lower than the median, suggesting that there are outliers or some kind of skew in the negative direction. This means that at times, PPO finds a few flights which it decides to anticipate much more aggressively than others. This might be PPO's way of compensating for SAC's more aggressive strategy overall.

The action taken can also be analysed in relation to the inputs that the algorithm takes into account. Out of all 15 inputs outlined in Table II, two are specifically interesting, and are both inputted as individual and average values. With these the algorithm should have enough information to understand when the agent is more critical than other agents in the same batch. The first input which is analysed is the delay versus the average delay. In Figure 9 the results are shown. Here, when the y-axis has a positive value, it means that the agent has more delay than the average. Here a value of 1 corresponds to 30 minutes of delay more than average. It can be seen how both algorithms tend to anticipate (orange) flights which have more delay than the average. In both plots it can be seen how the threshold for anticipation is around 30 minutes (y-axis value of 1), but the anticipation is maximum for flights with 60 minutes of delay. Below this anticipation threshold, there is no significant action for non-critical flights. The reason why these are not anticipated could be because the algorithm understands the consequences of very high delays. Another note can be made on how it seems that after episode $11k$ there are some flights with very high delay with respect to the average that get delayed even more by PPO. This proves that overfitting could be one plausible explanation for PPO's reward. This may partially be due to PPO attempting to over-optimize its initial policy in the small cases when for similar inputs and the same action it receives a different reward.

The second input, the number of critical connections, is shown in Figure 10. A positive value in the y-axis corresponds to a flight having more critical connections than the average. The magnitude of the y-axis is 1 for every 10 long-haul economy class passengers more than the average. It is evident how both algorithms tend to prioritize flights with a lot of critical connecting passengers (orange data points for positive y-axis values). Then, when the number of connections is low compared to the average, it seems that the algorithms decide that this flight is not so important, which is a logical choice. However, they do so in different strategies: PPO

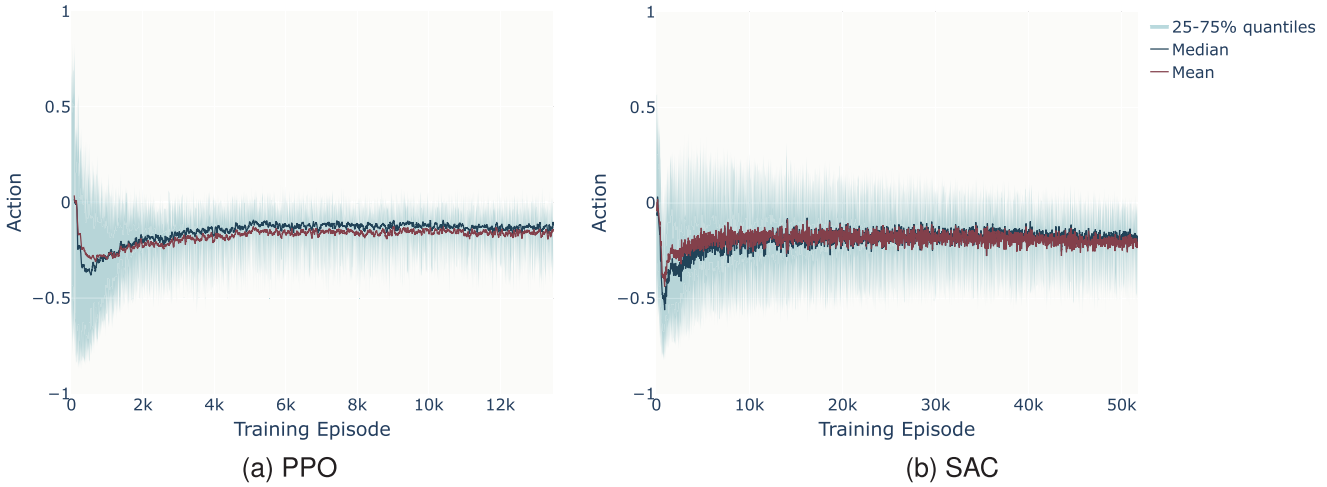


Fig. 8. Mean action range evolution during training for the DRL algorithms.

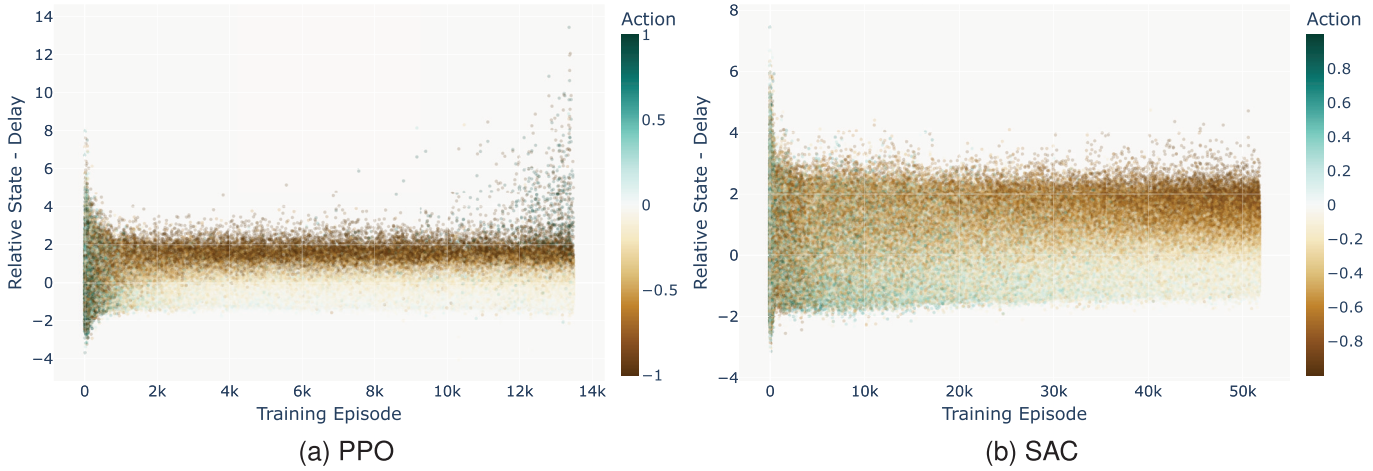


Fig. 9. Action sensitivity to relative difference to batch mean delay during training for both RL algorithms.

decides to not change those flights' delays (action is around zero), while SAC tends to delay those flights slightly further (notice a greenish line in Figure 10). Finally, there seems to be an orange line at the very lowest end of the y-axis values for both cases. This could be explained by flights with zero critical connections, which have very high delays. Even though the number of critical connections and delays are somewhat correlated inputs, whenever this is not the case due to zero critical connections, the tool understands that it should look at other inputs which might be more critical, such as the time of day (curfew) or the delay credit (fairness).

Finally, Figure 11 shows how the mean delay credit at the end of each episode evolves through training. In both cases, the delay credits evolution between the two algorithms is similar, with SAC being more varying due to its exploration-exploitation balance. In both cases, the algorithm starts off with a lot of positive delay credits, and then tends to anticipate more than it delays flights (hence having negative delay credits). What is interesting to see is that while fairness tends to increase over training, delay credits remain more or less constant for PPO and slightly decreasing for SAC. This is due to delay credits being essentially proportional to global fairness. Note that having a lot of anticipation is not bad, as long as it is not requested in a way that disadvantages other airlines.

B. Testing Results

Data between April 23^{rd} and May 26^{th} 2023 is used for testing. Figure 12 shows the algorithms scored for each reward type. Starting from the left, it is possible to observe how while the the baseline ILP model has relatively high fairness rewards, the DRL algorithms both struggle, especially SAC. The ILP, which from previous testing is known to be unfair for SWISS and not to other airlines, still ranks low for fairness, meaning that in reality it is likely that the two RL are more fair than they may seem. Besides curfew performance and fairness, it appears from Figure 12 that the DRL algorithms both outperform the ILP. Interestingly, it also appears that SAC always has an edge on PPO when it comes to unseen data.

Figure 13 shows improvements in critical passenger connecting times, and Figure 14 improvements in rotation delay. Focusing on passenger connections first, one can first notice how the mean improvement is highest for SAC (5.7 minutes improvement in connecting time), very closely followed by

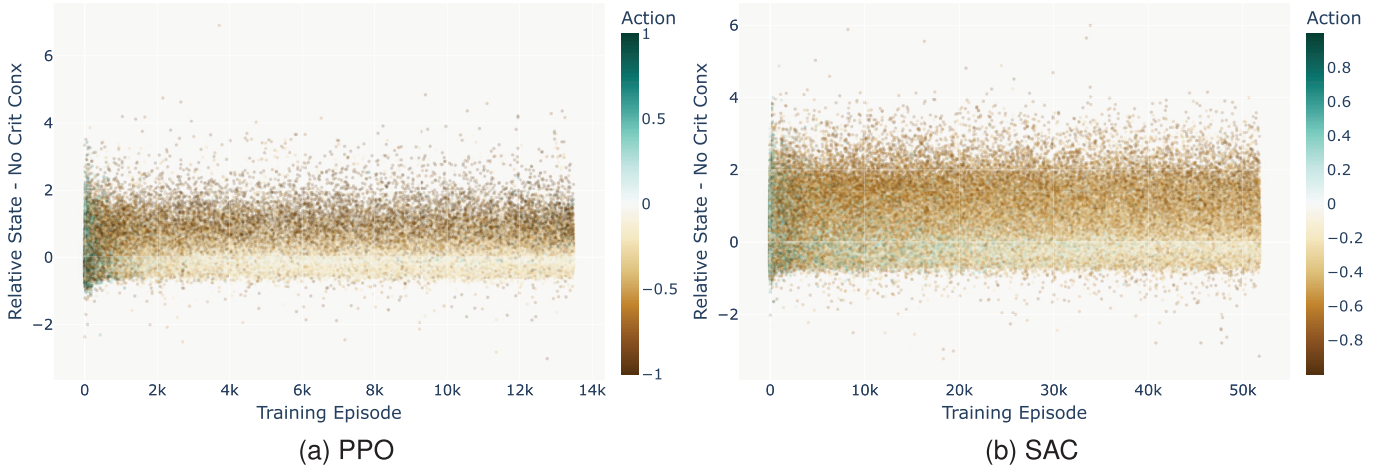


Fig. 10. Action sensitivity to relative difference to batch mean number of critical connections during training for both DRL algorithms.

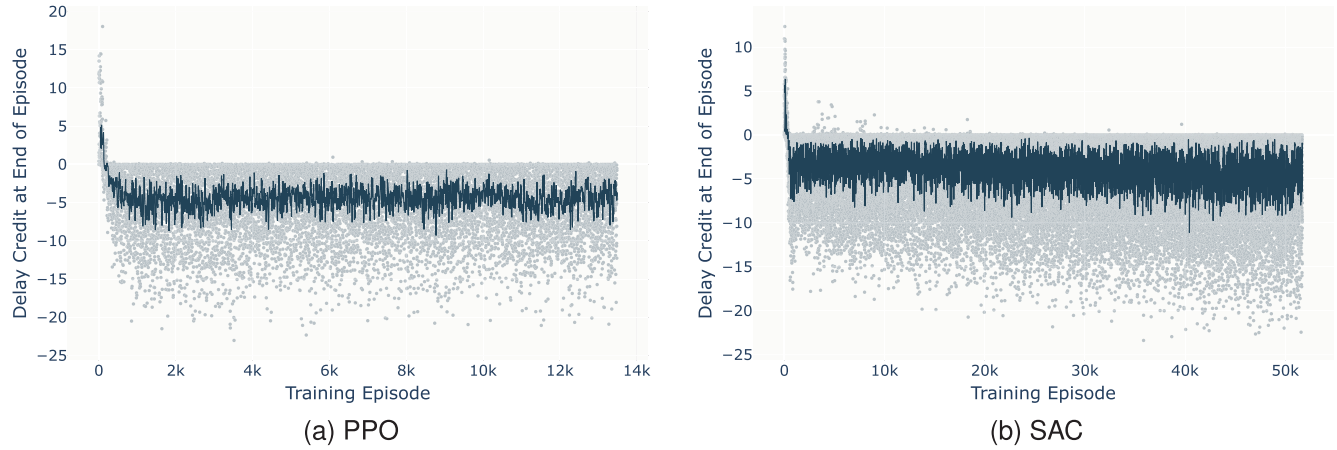


Fig. 11. Delay credit at end of episode evolution during training for both DRL algorithms.

Fig. 12. Average reward comparison during testing, split by reward section.

PPO (5.1 minutes), and ILP is last, with about half of the improvement of the DRL algorithms (2.8 minutes per minute of delay). It is also observable how the variation around these mean values is quite large - there a few some days where PPO outperforms SAC. This can be seen on May 9^{th}, for instance, where PPO performs best. On the other hand, in days like May 18^{th}, SAC performs better than the other two by a large margin. Section IV-B1 further

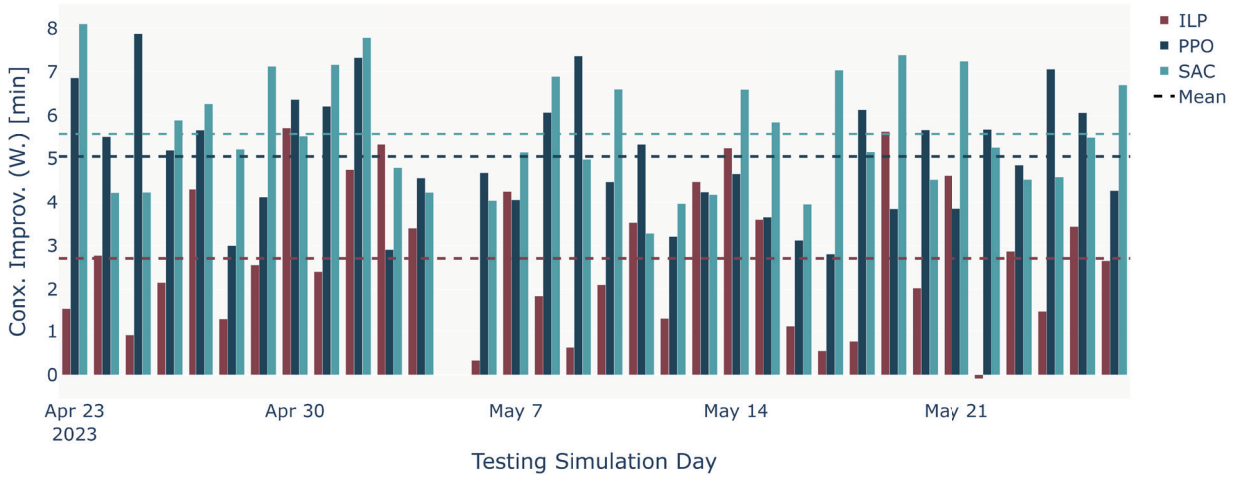


Fig. 13. Improvement minutes per weighted passenger for every simulation day, split by algorithm type.

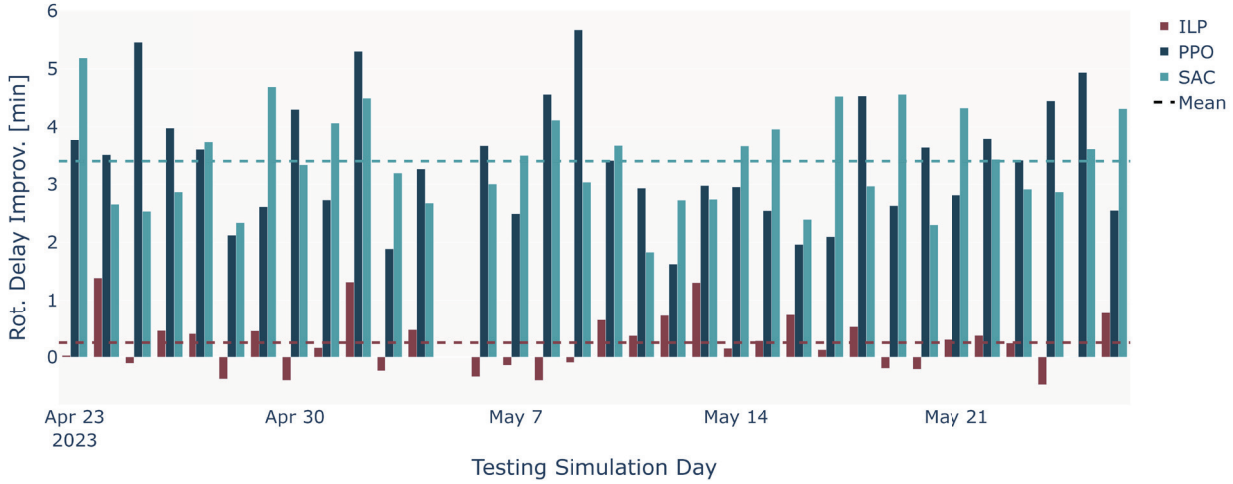


Fig. 14. Rotation delay improvement for every simulation day, split by algorithm type. The PPO and SAC mean dashed lines overlap, and because of this only the latter is visible.

analysis the differences between the algorithms at a daily level.

The rotation delay improvements for each testing simulation day is shown in Figure 14. Here, SAC and PPO are much closer in terms of operational performance SAC (both exactly at 3.4 minutes, overlapping in the figure), while the ILP barely has any improvement (0.2 minutes). The poor performance of the ILP is unfortunately by design: in the objective function, there is a trade-off between delay and passenger connections, and since in the summer the latter was SWISS's maximum priority, delays were not greatly improved. It is also interesting how there are days where SAC and PPO perform equally well (May 10^{th}), days where PPO performs much better (May 9^{th}), and days where SAC is better performing (May 18^{th}). The reasons behind this will be investigated further in Section IV-B1.

One of the main reasons behind ILP not performing very well is that its decision-making process only relies on optimizing for connections. This can be investigated by finding the correlation between the connection time and rotation delay improvement for each reward received. The results of this are shown in Figure 15. As can be seen, there is a weak Pearson's correlation coefficient ($r = 0.11$) for ILP between the two parameters, while it is strong for PPO and SAC ($r = 0.84$ and $r = 0.78$, respectively). The implications of this are that PPO and SAC identify flights which are critical both in terms of rotation delay and passenger connections and optimize for those. This proves that the DRL algorithms understand that these are the most important elements for future reward. This also ensures that when the flights come back to Zurich, they do not come back too delayed. This is the key to also having a higher improvement in passenger connections overall compared to the ILP, since the effect of not including delay in the decision-making mechanism is compounding. If a flight is delayed, it is likely to have more missed connections, which means it will be prioritized more compared to delayed flights, which will cause even more missed connections further in the future. This also proves the advantage of using DRL instead of ILP due to the ability to consider future reward.

As can be seen in Figure 16, the three methods have very different action distributions. This plot shows the probability

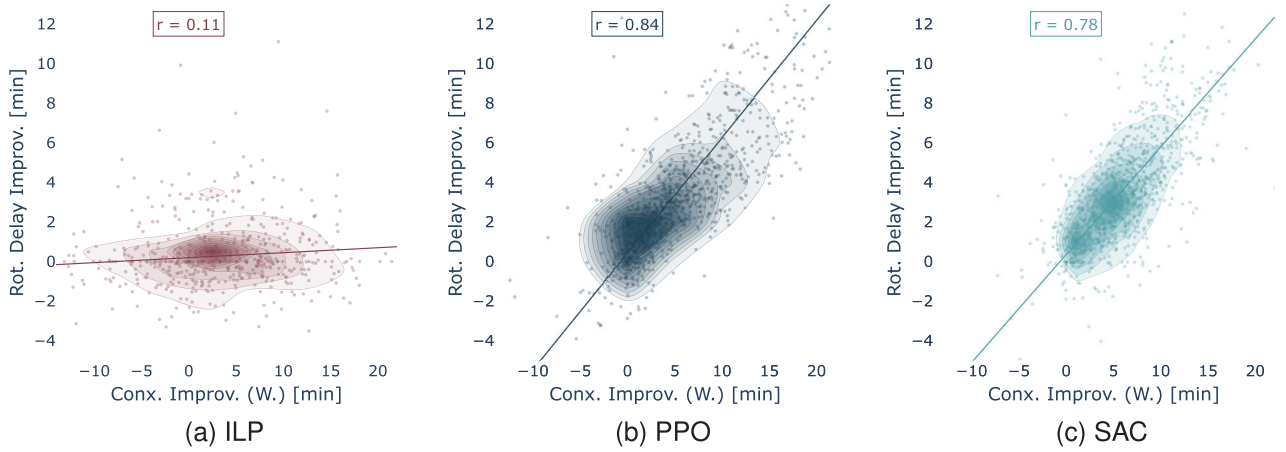


Fig. 15. Correlation plot between connection and rotation delay improvement for the three models.

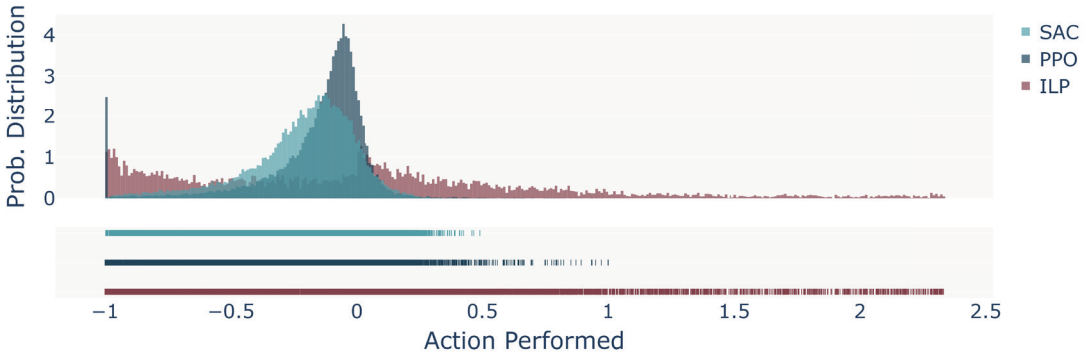
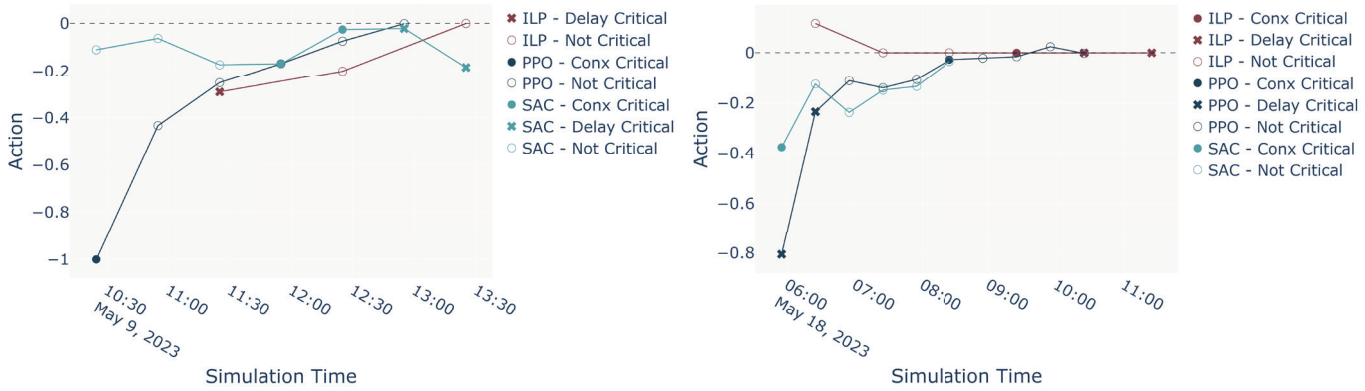


Fig. 16. Action probability distribution (probability density) during testing.

distribution function for the whole action range, for each algorithm. Starting with the ILP, it can be seen how it well distributed along the whole range of possible outcomes, with peaks around -1 (30 minutes of anticipation) and 0 (no change in Arrival Time). On the other hand, PPO's distribution is shaped like a β-distribution with a peak right on the negative side of the zero. This allows it to counter the environment which sometimes does not accept the requested TTA. Moreover, it is very interesting how there are a few flights which often receive a -1 value, because the algorithms understands their high priority. While this may seem at first a great strategy, it might actually not be. The probability that CASA accepts it is lower, and as such, it may be that by asking a more moderate anticipation, the flight might depart earlier. Finally, SAC has a much wider distribution with a peak around -0.2, and with tails extending much further than PPO. This behaviour causes SAC to both anticipate and delay flights more often than PPO. Yet, because of PPO's peak at -1, it seems like PPO is even more aggressive with anticipating very specific flights than SAC. The ILP, on the other hand, has tails on the delay side (action$>$0) extending way beyond SAC's and PPO's, delaying flights by much larger margins.

1) Comparison of PPO and SAC Behaviour: In order to correctly identify the differences in behaviour between PPO and SAC, one needs to dive deep at flight-level. With two days identified where PPO and SAC performed best (9^{th} May and 18^{th} May, respectively), one can look for flights which appeared in comparable ways to the algorithms but elicited different responses from them. Starting with Figure 17a, where PPO performed better. Here, the actions performed are shown against the simulation time. For each action in the plot, the observations are analysed and the flight is categorized either as non-critical, critical for delay or critical for passengers. It is possible to see how PPO tends to quickly assign actions of -1 to the flight until it is not critical any more. It then proceeds to decrease the anticipation requests to smaller values, until it takes off around 13:00. On the other hand, SAC anticipates it less at the beginning and then around 12:00, the flight appears to be critical once, and it decides to keep anticipating it until it takes off later than for PPO (around 13:30). Because PPO tended to successfully anticipate it by larger amounts at the beginning, the flight was able to take off earlier with PPO.

On the other hand, one can also look at a day where SAC performed better in Figure 17b. Here, PPO also starts off by attempting to anticipate the flight heavily, with an action of -0.8. On the other hand, SAC does not do so, and only starts with an action of -0.4. The following actions by both SAC and PPO are closer to zero, with PPO always anticipating more heavily than SAC. Nevertheless, it appears that the SAC flight takes off more than 1.5 hours before PPO's. This must mean



(a) Example A (b) Example B

Fig. 17. Actions performed on different flights used as example.

that some of the initial attempts of PPO to anticipate the flight were unsuccessful, while SAC's were. This is because more extreme anticipations are less likely to be accepted by CASA (and by the simulation environment as well).

In conclusion, the behaviour difference between PPO and SAC mainly arises from the former more harshly anticipating flights (with an action of -1, which corresponds to 30 minutes of anticipation) hours prior to their departure. On the other hand, SAC tends to anticipate flights step-by-step. Whenever PPO is successful with the first anticipations, it performs best. However, it often is not due to the environment not being able to fulfil its requests, hence performs slightly worse.

A final note has to be made regarding their difference in performance which cannot be seen from these plots. Even though SAC's performance is slightly superior, the deterministic version PPO is preferred for operational reasons. Additionally, results show that of PPO has a smaller action range than SAC (see Figure 8) and, in general, a less aggressive approach. This is preferred by the airline as, the added advantages in predictability and reliability in stability (for what an RL model can guarantee) outweigh the marginal increase in operational performance by SAC. As such, PPO is chosen for testing in a more realistic environment than the one modelled for training.

C. Testing in a Real Environment

To see the effect of the program in a non-simulated environment, it was connected via B2B to EUROCONTROL's Network Manager PreOps environment [4]. This is a copy of the of the ATFM live status, with real-time updates to all events. However, actions regarding TTAs within this environment do not affect real flights, but rather only affects fictitious slots assigned to the real flights. The system which assigns these fictitious slots is the same which assigns them in operations (CASA). This allows to simulate slot assignment in a very real environment without having repercussions on real flights. This is a perfect test case for the developed Reinforcement Learning tool, even more so because the ILP model is still running in operations, affecting real flights. This allows them for to be tested in parallel, on the same regulation, with the same flights. The only difference is that the ILP actually affects real flights and passengers, but as far as performance analysis goes, this does not affect the test's fidelity. Due to higher stability, simplicity and an overall good performance, PPO was tested instead of SAC.

TABLE V
REAL ENVIRONMENT TESTING CONDITIONS

Parameter	*ILP*	*PPO*
Total Flights	193	269
Average Delay	9.9 min.	12.3 min.
Total TTAs Sent	613	1694
Achieved Anticipated TTAs	57/148 (38.5%)	221/1119 (19.7%)
Achieved No-Change TTAs	272/315 (86.3%)	181/210 (86.2%)
Achieved Delayed TTAs	102/150 (68.0%)	194/365 (53.2%)

The two algorithms were tested and compared on EUROCONTROL's PreOps environment between Monday 4^{th} December and Friday 8^{th} December, 2023. During these 5 days, there were a total of four regulated days with arrival regulations in Zurich, which allowed for a large number of flights for both the ILP and PPO algorithms to be tested. Moreover, on a fifth day, a simulated regulation was included on the PreOps test system, allowing for even more flights to be tested for the PPO algorithm. An overview of the total amount of flights, along with other parameters, is shown in Table V.

From Table V it is already possible to see how the two approaches differ greatly. First of all, while the ILP has a relatively symmetrical distribution between TTAs which anticipate, delay or cause no change to flights, PPO is significantly more skewed. This is because PPO knows that the chances of an anticipation being accepted are significantly lower than a delay. However, what is interesting to realize the achievement percentage of anticipations also depends on the total amount of anticipations asked. It appears than PPO has only half of the anticipation achievement rate (19.7%) that ILP has. Even though the first value might seem a disadvantage for PPO, it is not necessarily the case. This is because the improvement towards the desired TTA still yields positive operational results: the percentage of achieved TTAs is only an indication of the behaviour of the environment rather than an operational metric. As such, other metrics will be later used to truly evaluate the performance of the two algorithms. It is also interesting to note how the total amount of TTAs sent

TABLE VI
OPERATIONAL KEY PERFORMANCE INDICATORS

Key Performance Indicators	*ILP*	*PPO*
Weighted Improvement Conx Time per Pax	5.0 min	5.9 min
Unweighted Improvement Conx Time per Pax	1.6 min.	2.6 min.
Rotation Delay Avg. Improvement	0.9 min	4.8 min
Share of SWISS Higher Delay	5.4%	17.2%
Share of Non-SWISS Higher Delay	0.5%	3.8%
Share of No Difference in Higher Delay	94.2%	72.3%

by the PPO algorithm was nearly three times as much as the ones sent by the ILP (1694 versus 613). This is also due to the fact that PPO was allowed to send one set of TTAs every 30 minutes since it is dynamic and tends to be more forward-looking. On the other hand, upon seeing the lack of dynamicity of the ILP approach, operations controllers at SWISS decided that it would only send updates once every 60 minutes.

One last comment regarding the values in Table V must be made. It seems like PPO increased the average delay from around 10 minutes to 12.3. Even though without information on other airline's slots it is difficult to truly understand why, there are two possible explanations. Firstly, a very plausible one is that the difference is not significant: only two minutes, with a sample size of only five days, is not sufficient to make any conclusions. While this may be true, a second analysis should also be made. It is believed by both experts at SWISS and at EUROCONTROL that once the TTA tool delays a flights, CASA then works to optimize by filling empty slots and improving flights' delays. What is unknown is how much time this takes. If this was to take a significant amount of time, in the order of tens of minutes, there would be a gap where the average delay is higher than it should be. If the flights both take off before improving, the average delay remains high. This would explain why asymmetrical TTAs like in PPO could increase delays overall, but it should be further analysed with EUROCONTROL data.

To truly understand the performance of the two algorithms in the same conditions, it is necessary to look at operational performance metrics. The most important ones (with which the current operational tool is also measured) are shown in Table VI and reflect connecting passengers, rotation delay and some aspects of fairness. Unfortunately curfew performance was not able to be tested since the regulations within the testing period did not extend late enough in the evening. This being said, the performance was still very interesting: it appears that PPO is superior to ILP for both rotation delay performance and for passenger connection improvement times. While the difference in passenger connection improvements is only around 20% (5.9 minutes weighted improvement versus 5.0 minutes), the improvement in rotation delay is nearly 5-fold. This is mainly due to the dynamics observed in Figure 15, where also during testing within the simulation environment it was seen that RL algorithms tended to optimize flights which were both critical in rotation delay and in passenger connections. This is a confirmation that the environment created was a good enough representation for the simulated operational improvements to translate to real life.

The delay distributions of SWISS flights over time was compared to the distributions of other flights incoming to Zurich. Their distributions are compared every two minutes, and with a two-tailed t-test it is possible to see whether the means of the distributions are statistically different or not. This was tested at 2-σ significance level ($p < .05$). As shown in Table VI, most of the time, for both ILP and PPO, there was no significant difference between the flights. This is expected - no airline should have significantly more ATFM delay in case of an arrival regulation. However, the ratios of SWISS and non-SWISS flights having higher delay shares are very different between ILP and PPO. As partially anticipated, the share of time that non-SWISS flights have higher delay increases when using PPO. This may be due to the fact that the delay credit value is negative at the end of the day (also seen in Figure 11). This could be solved by adding a stronger penalty for fairness during training. However, it seems like also SWISS's delay portion has increased from 5.4% to 17.2%. This implies that the PPO algorithm has somehow increased the polarization of the delays - the spread between the highest and lowest delays has increased. This is due to the oversimplification of CASA's dynamics: it must be that to compensate for a higher number of anticipations requests, CASA assigns higher delays to low-priority flights, and much lower delays to high-priority flights (here, priority both entails the priorities assigned by TTAs as well as first-filed-first-served for non-TTA flights from other airlines).

V. DISCUSSION

Modelling a dynamic environment is of great aid to improving TTM. The ability to account for future states was the key reason behind the success of both PPO and SAC when compared to the current method using ILP. The operational performance increase by 20% for passenger connections and by 5-fold for rotation delay was mainly due to the ability to identify flights that are both critical for connections and passengers at the same time ($r = 0.84$ in Figure 15), while ILP considers them as totally independent variables ($r = 0.11$ in Figure 15). Nevertheless, many question remain before implementation of this model in a fully operational real-environment. Some of which are discussed in the following sections.

A. DRL Algorithms

Both SAC and PPO seem to be strong choices for DRL algorithms. SAC performs slightly better when it comes to untrained data, since even though it had lower rewards during training (around -140 versus -100), it had higher overall rewards during testing in virtually every reward element besides fairness. Their behavioural differences arrives from PPO tending to harshly anticipate a small set of flights, while SAC tends to have a more balanced overview between the priorities of flights. This results in more TTAs being accepted with SAC, hence a slightly better performance. On the other hand, SAC is non-deterministic, which makes it a potentially more operationally risky tool. This is because in very rare cases it could make an action very different to the ones it

has taken before. Moreover, since it may produce different actions for the same inputs, it is likely to be less accepted by users in the operations control center, who are ultimately responsible for the tool (even though it runs automatically). Despite generalizing better to the 35-day batch it was tested on, it would need further testing to really be operationally stable. Hence, PPO was chosen for the time being as the most operationally viable tool. It would be interesting as future work to test other algorithms which are more advanced than PPO, such as Deep Deterministic PPO (DDPPO) or Deep-Deterministic Policy Gradient (DDPG).

B. Explainability & Predictability

To successfully deploy the tool in operational settings, several steps must be taken to ensure effective implementation. First, a higher level of explainability should be attempted. Second, a series of edge-cases where the optimization is particularly difficult or has a high risk-reward ratio should be tested. Third, the algorithm would require ordinary monitoring and maintenance. For example, keeping the ILP active in the background, and only activating it when the ILP's performance exceeds RL's would prove as a stable fallback option. Moreover, frequent retraining of the model proves to be a proactive solution to potential changes to the environment over time.

C. Future Work

The environment created using statistics on TTA acceptance rates was sufficient to improve the baseline model for operations, but it was not enough to truly be a realistic, high fidelity model of CASA. This is because the complexity of CASA are also arises from flights of other airlines, which SWISS does not have information of. This also affects the TTA acceptance rates as well as the slot allocation process. This lack of modelling also caused the delay distributions to be more polarized, ultimately increasing the gap between the highest and the lowest delays. As future work, it is recommended to model other aircraft within the system (either with other airlines, or simplifying their behaviour within CASA) or directly train with the CASA system. The latter would be the best approach, but with the scale of data required for Reinforcement Learning, some type of offline version of CASA would have to be available.

Other ideas for future work imply adding more inputs. The DRL model was able to identify which flights were critical for turnaround just based on their current delay, time of day and other patterns between inputs. While this is possible, it could certainly be improved by also giving inputs such as the scheduled turnaround time or the predicted turnaround time. Another input which might help the model understand the time to the last decision it can make, before take-off time, is the estimated taxi time, since this reflects the elapsed time between off-block and take-off.

Finally, a more complex version of rewards should be implemented. The current setup assumes that three flights delayed by one hour are equivalent to one flight delayed by three - and this is operationally not true, since most airlines would cancel flights with a delay of three hours. This could be solved in one of two ways: firstly, it would be possible to weigh the change in delay of a flight with its magnitude when using it in the reward function (would not be a totally new concept since connections are also weighed, but using passenger importance instead). Or, one could also fully integrate cancellations in the environment, and giving a large (reward) penalty for every cancelled flight.

VI. Conclusion

This study created a Multi-Agent RL decision-making model for aircraft prioritization, based on passenger connection, rotation delays, curfew performance and fairness. The model has to adapt to the uncertainties of whether requests for Target Times of Arrival are accepted or not. The model achieved an improvement in passenger connecting of 5.9 minutes and in rotation delay of 4.8 minutes. This is an improvement compared to a deterministic approach. The DRL model understood that certain flights were critical for both passenger connection time and rotation delay reasons. By identifying and prioritizing these flights, it was able to increase their future reward, since these flights would not come back delayed from their outstations. This proves that a dynamic, forward-looking system can greatly improve decision-making within Target Time Management. These results were also backed by similar numbers in the live trial.

This dynamic decision-making process can improve the usage of the Target Time Management system created by EUROCONTROL. Moreover, it was also proved than Multi-Agent RL is a great tool to achieve this, since it can be modelled on a per-flight basis, as well as take into account complex environment dynamics. Nevertheless, further testing on fairness for all involved airlines should be undertaken for this model to become operational. Additionally, due to the lack of historical data on certain features, these could not have been included. Nevertheless, after the analysis of the behaviour of the algorithms, it can be stated that these features could have improved the performance even more. These are scheduled turnaround time, predicted turnaround time and taxi times. If these were included in the model, it could allow it to further its environment-inference abilities. Finally, allowing a different delay-assignment distribution in the environment would also increase its fidelity.

References

[1] H. N. Psaraftis, "A dynamic programming approach to the aircraft sequencing problem," Dept. Flight Transportation Laboratory, Massachusetts Inst. Technol., Cambridge, MA, USA, Tech. Rep. R78-4, 1978.

[2] R. M. Vervaat, "Airline based priority flight sequencing: Of aircraft arriving at an airport," Master's thesis, Delft Univ. Technol., Delft, The Netherlands, 2020.

[3] R. Hoogendoorn, "Dynamic airline centric inbound priority sequencing: A case study on Westerly morning arrivals for KLM at Schiphol," Master's thesis, Delft Univ. Technol., Delft, The Netherlands, 2022.

[4] *ATFCM Users Manual*, Eurocontrol, Brussels, Belgium, Mar. 2023.

[5] *Network Operations Report 2019-Main Report*, Eurocontrol, Brussels, Belgium, 2020.

[6] L. Caranti and M. Carré, L. Weber, D. Heilmann, and R. Stevens "Optimization of regulated airline arrival flows via target time management, in *Proc. SESAR Innov. Days (SID)*, Seville, Spain, Nov. 2023.

[7] A. Pawełek and P. Lichota, "Tactical and strategic air traffic sequencing with minimum-fuel trajectories," *J. Theor. Appl. Mech.*, vol. 63, no. 1, pp. 27–36, 2025.
[8] R. K. Çeçen and Y. Durmazkeser, "Meta-heuristic algorithms for aircraft sequencing and scheduling problem," in *Progress in Sustainable Aviation*, T. H. Karakoc, C. O. Colpan, and A. Dalkiran, Eds., Cham, Switzerland: Springer, Jan. 2022, pp. 107–118.
[9] J. Ma, D. Delahaye, and M. Liang, "Arrival and departure sequencing, considering runway assignment preferences and crossings," *Aerospace*, vol. 11, no. 8, p. 604, Jul. 2024.
[10] Z. Du, J. Zhang, and B. Kang, "A data-driven method for arrival sequencing and scheduling problem," *Aerospace*, vol. 10, no. 1, p. 62, Jan. 2023.
[11] A. Montlaur and L. Delgado, "Flight and passenger delay assignment optimization strategies," *Transp. Res. C, Emerg. Technol.*, vol. 81, pp. 99–117, Aug. 2017.
[12] Y. Ding, S. Wandelt, G. Wu, Y. Xu, and X. Sun, "Towards efficient airline disruption recovery with reinforcement learning," *Transp. Res. E, Logistics Transp. Rev.*, vol. 179, Nov. 2023, Art. no. 103295.
[13] Y. Xu, S. Wandelt, and X. Sun, "A distributionally robust optimization approach for airline integrated recovery under in-flight pandemic transmission risks," *Transp. Res. C, Emerg. Technol.*, vol. 152, Jul. 2023, Art. no. 104188.
[14] R. K. Çeçen, "A stochastic programming model for the aircraft sequencing and scheduling problem considering flight duration uncertainties," *Aeronaut. J.*, vol. 126, no. 1304, pp. 1736–1751, Apr. 2022.
[15] S. Ikli, C. Mancel, M. Mongeau, X. Olive, and E. Rachelson, "Coupling mathematical optimization and machine learning for the aircraft landing problem," in *Proc. 9th Int. Conf. Res. Air Transp.*, Jun. 2020, pp. 1–8.
[16] K. K. H. Ng, C. K. M. Lee, F. T. S. Chan, and Y. Qin, "Robust aircraft sequencing and scheduling problem with arrival/departure delay using the min-max regret approach," *Transp. Res. E, Logistics Transp. Rev.*, vol. 106, pp. 115–136, Oct. 2017.
[17] F. Furini, M. P. Kidd, C. A. Persiani, and P. Toth, "Improved rolling horizon approaches to the aircraft sequencing problem," *J. Scheduling*, vol. 18, no. 5, pp. 435–447, Oct. 2015.
[18] *Innovative Slot Allocation: An Overview*, Eurocontrol, Brussels, Belgium, Jan. 2014.
[19] N. Pilon, L. Guichard, Z. Bazso, G. Murgese, and M. Carré, "User-driven prioritisation process (UDPP) from advanced experimental to pre-operational validation environment," *J. Air Transp. Manage.*, vol. 97, Oct. 2021, Art. no. 102124.
[20] C. G. Schuetz et al., "A distributed architecture for privacy-preserving optimization using genetic algorithms and multi-party computation," in *Proc. Int. Conf. CoopIS*, in Lecture Notes in Computer Science, M. Sellami, P. Ceravolo, H. A. Reijers, W. Gaaloul, and H. Panetto, Eds., 2022, pp. 168–185.
[21] J. Evler, M. Schultz, and H. Fricke, "Flight prioritization and turnaround recovery," in *Proc. 14th ATM Res. Develop. Seminar*, 2021.
[22] K. Chen, J. Wu, and L. Yang, "Reassigning departure slots with preferences of the airline and passengers," *Transp. Res. Rec.*, vol. 2678, no. 3, pp. 786–804, Mar. 2024.
[23] R. Yang, M. Le, and Q. Wang, "Multi-objective airport slot allocation with demand-side fairness considerations," *Aerospace*, vol. 12, no. 2, p. 119, Feb. 2025.
[24] A. Nguyen-Duy, D.-T. Pham, J.-Y. Lye, and D. Ta, "Reinforcement learning for strategic airport slot scheduling: Analysis of state observations and reward designs," in *Proc. IEEE Conf. Artif. Intell. (CAI)*, Jun. 2024, pp. 1195–1201.
[25] *Arrival Planning Information (API) Implementation Guide*, Eurocontrol, Brussels, Belgium, May 2022.
[26] A. Muharremoglu, "The aircraft sequencing problem with arrivals and departures," M.S. thesis, Massachusetts Inst. Technol., Cambridge, MA, USA, 2000.
[27] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed., Cambridge, MA, USA: MIT Press, 2018.
[28] P. Razzaghi et al., "A survey on reinforcement learning in aviation applications," 2022, *arXiv:2211.02147*.
[29] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*.
[30] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1861–1870.
[31] L. Engstrom et al., "Implementation matters in deep policy gradients: A case study on PPO and TRPO," 2020, *arXiv:2005.12729*.
[32] A. P. Kalidas, C. J. Joshua, A. Q. Md, S. Basheer, S. Mohan, and S. Sakri, "Deep reinforcement learning for vision-based navigation of UAVs in avoiding stationary and mobile obstacles," *Drones*, vol. 7, no. 4, p. 245, Apr. 2023.
[33] J. N. Tsitsiklis and B. Van Roy, "An analysis of temporal-difference learning with function approximation," *IEEE Trans. Autom. Control*, vol. 42, no. 5, pp. 674–690, May 1997.
[34] R. Munos, "Error bounds for approximate policy iteration," in *Proc. 19th Int. Conf. Mach. Learn.*, vol. 3, 2003, pp. 560–567.
[35] C. Gelada and M. G. Bellemare, "Off-policy deep reinforcement learning by bootstrapping the covariate shift," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 3647–3655.
[36] T. Haarnoja et al., "Soft actor-critic algorithms and applications," 2019, *arXiv:1812.05905*.
[37] Z. Li, T. Chen, Z.-W. Hong, A. Ajay, and P. Agrawal, "Parallel Q-learning: Scaling off-policy reinforcement learning under massively parallel simulation," in *Proc. Int. Conf. Mach. Learn.*, 2023, pp. 19440–19459.
[38] C. Wang and K. Ross, "Boosting soft actor-critic: Emphasizing recent experience without forgetting the past," 2019, *arXiv:1906.04009*.
[39] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017, *arXiv:1412.6980*.

Leonardo Caranti received the bachelor's degree in aerospace engineering from Delft University of Technology (TU Delft) and the master's degree in aerospace engineering from TU Delft in 2024. Since then working as a Data Scientist at SWISS International Airlines. During his studies, he mostly worked on the combination of machine learning, optimization and aircraft operations topics, especially from the perspective of Air Traffic Management and Airlines. Besides an internship at SWISS, he also developed his thesis in collaboration with the airline.

Marta Ribeiro received the Ph.D. degree from Delft University of Technology, where she specialized in using deep reinforcement learning to enhance the safety of autonomous flying vehicles. She is currently an Assistant Professor with the Operations and Environment Section, Aerospace Engineering Faculty, Delft University of Technology. Her research focuses on leveraging artificial intelligence and machine learning to optimize aviation operations, with a particular emphasis on condition-based aircraft maintenance.

Marie Carré received the Diploma degree in engineering from French Institute of Advanced Mechanics and the Ph.D. degree in computer science, on the operations optimization of multi-hubs airlines during disruption. She has been working at SWISS Operations Research and ATM team for over seven years. She is responsible for all SWISS participation to SESAR projects and research initiatives. Her current projects ranges from trajectory-based operations (NM-TBO and ATC TBO), optimization of arrival sequence through multi-stakeholder optimization (HARMONIC), and flight trajectories optimization to reduce both CO2 and non-CO2.