Dynamic Target Time Management with Reinforcement Learning

A case study on Zurich short-haul regulated arrivals

Master Thesis Report

Aerospace Engineering - Control & Operations
Sustianable Air Transport

Leonardo Caranti

TU Delft Delft University of Technology

SWISS

Dynamic Target Time Management with Reinforcement Learning

A case study on Zurich short-haul regulated arrivals

Master Thesis Report

Aerospace Engineering - Control & Operations
Sustianable Air Transport

by

Leonardo Caranti

to obtain the degree of Master of Science

at the Delft University of Technology,

to be defended publicly on 26th January 2024.

Student number: 4796594

Project duration:	April 2023 - January 2024
Thesis committee:	Dr. Marta Ribeiro
	Dr. Bruno Lopes Dos Santos
	Dr. Erik-jan van Kampen
	Dr. Junzi Sun
	Dr. Marie Carré

An electronic version of this thesis is available at http://repository.tudelft.nl/.

TU Delft

Acknowledgements

This thesis marks the end of my journey as a Master Student at the TU Delft. It has been an exciting two and a half years, filled with highs and lows, but most importantly filled with memories I will bring with me through the years.

I would firstly like to thank my supervisor from the TU Delft, Marta Ribeiro, whose commitment to my supervision is indescribable. The shear amount of effort she devoted to allow the success of this thesis cannot be overstated. I would also like to thank my supervisor from SWISS International Airlines, Marie Carré, whose highly professional and scientific approach to operational insights proved fundamental to project.

I also want to thank Bruno Santos, whose research vision allowed for this thesis to be streamlined towards the best literary and operational goals. On top of this, I would like to thank Dominique Heilmann and the whole Operations Research & Air Traffic Management team at SWISS, which besides granting me the opportunity to research such a valuable topic, also nurtured a beautiful working environment. I would also like to express my gratitude to Evgeni Todorov, Daniel Bogado Duffner and Alvaro Tomas Gil, for the supervision and recurring meetings throughout my thesis. I would also like to thank Richard Stevens and Ramon Dalmau-Codina for the patience and support in setting up my models.

Finally, I would like to thank my friends and family for the support and patience given to me during the course of this thesis, as well as the rest of my academic journey.

Leonardo Caranti
Delft, January 2024

Contents

List of Figures

List of Tables

List of Abbreviations

ANSP	Air Navigation Service Provider
ASP	Aircraft Sequencing Problem
ASSP	Aircraft Sequencing and Scheduling Problem
ATFCM	Air Traffic Flow and Capacity Management
ATFM	Air Traffic Flow Management
ATM	Air Traffic Management
CASA	Computer Assisted Slot Allocation
CTOT	Calculated Take-Off Time
DCB	Demand-Capacity Balancing
DDPG	Deep Deterministic Policy Gradient
DDPPO	Deep Deterministic Proximal-Policy Optimization
DQN	Deep Q-Network
ECDF	Empirical Cumulative Distribution Function
ECTL	EUROCONTROL
ETA	Estimated Time of Arrival
ETO	Estimated Time Over
FCFS	First Come First Served
LSTM	Long-Short Term Memory
MAE	Mean Average Error
MARL	Multi-Agent Reinforcement Learning
MDP	Markov Decision Process
MILP	Mixed-Integer Linear Programming
NN	Neural Network
PDF	Probability Density FUnction
PPO	Proximal Policy Optimization
RL	Reinforcement Learning
SAC	Soft Actor-Critic

Seq2Seq	Sequence-To-Sequence
SESAR	Single European Sky ATM Research
TBO	Trajectory-Based Operations
TTA	Target Time of Arrival
TTMS	Target Time Management System
UDPP	User-Driven Prioritization Process

Introduction

With up to 91% percent of the 2019 flights being flown in 2023, the European airspace has nearly fully recovered from the COVID-19 pandemic. Yet, Air Traffic Management (ATM) capacity still lags behind (EUROCONTROL, 2023). This results in a recurring pattern of ATM-related delays for all stakeholders. On top of these demand-driven delays, SWISS International Airlines (a participant in this thesis) is also heavily affected with weather and noise-related ATM regulations due to the topography of Zurich Airport and its runway configuration. In turn, this leads to a necessity for airlines like SWISS to optimize for these events.

There exist several approaches to mitigate this problem. The most common one, which most airlines take part in, is a reactive one: operations controllers, on the day of operations, attempt to modify schedule, tail assignments and other processes to improve performance. Nevertheless, since this is a reactive method and it requires manual labor, it is not easily scalable nor as efficient as a proactive method. This is why approaches such as the Aircraft Sequencing Problem (ASP) or Slot Swapping have emerged - here, the flights on sequence of arrivals are reordered based on their importance to the operations, either in the air (ASP) or on the ground (slot swapping). While this has proved a valid approach within literature, such as by Hoogendoorn (2022) or by Pilon et al. (2021), it has a series of limitations. Firstly, ASP sometimes involves holdings, in which fuel burn is increased, since the amount of time saved by slowing down a short-haul flight is often not enough. Moreover, both methods are often not fully evaluated at network-level, either because of a lack of dynamic modelling within the decision-making process or a lack of airline-sensitive data, or both.

A new approach to flight prioritization, named Target Time Management, was ideated by EUROCONTROL (while EUROCONTROL (2014) is not the original paper, it provides the best explanation). This is approach allows airlines and ANSPs to request for a Target Time of Arrival (TTA) for each flight in the case of an Air Traffic Flow Management (ATFM) regulation. This can be requested up to five hours before take-off, and once the flight's arrival time is adjusted, it receives a corresponding departure slot. SWISS, in the research undertaken by Caranti et al. (2023), was the first airline to influence TTAs, but did so using a static approach with Mixed-Integer Linear Programming. Like also seen in previous work by Pilon et al. (2021) and Vervaat (2020) on similar topics, the ATFM delay environment is highly dynamic, and a static approach has a series of limitations which affect its decision-making abilities.

The aim of this thesis is to investigate a dynamic decision-making approach to improve the usage of Target Time Management for SWISS's short-haul fleet in arrival regulations. The methods used are two Reinforcement Learning algorithms, Soft-Actor Critic (SAC) (Haarnoja et al., 2018) and Proximal-Policy Optimization (PPO) (Schulman et al., 2017), which are compared to the MILP approach described in Caranti et al. (2023). Approximately five years of data are used to train the models which are tested on May 2023 data. After this, PPO is tested in a shadow mode trial for five days and compared to the MILP.

Research Objective

The research question, which reflects the previously outlined research goals, is formulated as follows:

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

Report Structure

The report is structured in two sections. Firstly, Part I comprises of the self-contained scientific article where the research is fully outlined. Following this, Part II contains the literature study which laid the foundations for this project.

I

Scientific Paper

Dynamic Target Time Management with Reinforcement Learning

Leonardo Caranti*

Delft University of Technology, Delft, The Netherlands

Abstract

This Master Thesis investigates the possible improvements to the Target Time Management concept to optimize the arrival flows for SWISS International Airlines. The aim is to improve operational performance based on the current model used, as well as prove that Target Time Management constitutes a valuable system to improve operations in a dynamic way. To leverage the dynamic nature of slot assignment, an environment model is created and used as training base for two Multi-Agent Reinforcement Learning algorithms. These two algorithms, Soft-Actor Critic (SAC) and Proximal Policy Optimization (PPO), are then tested against the baseline model currently used in operations at SWISS (based on Mixed-Integer Linear Programming). The four domains to measure the algorithms' performance are passenger connecting time, curfew performance, rotation delay and fairness to other airlines. The algorithms were trained in a simulation environment based on statistical representations of the dynamics of the slot allocation system of EUROCONTROL. They were then tested with new data, where they outperformed a MILP implementation in passenger connecting time and rotation delay metrics (curfew and fairness were comparable in magnitude, since the MILP was slightly unfair for SWISS and RL was slightly unfair for other airlines). PPO was then also tested on the real slot assignment environment hosted by EUROCONTROL and once again compared to a MILP approach. Here, it was found that the improvement in critical passenger connecting time was 5.0 minutes for the MILP, and 5.9 minutes for PPO. Rotation delay was improved by 0.9 minutes by the MILP, and by 4.8 minutes by PPO. PPO also made the highest delays higher and the lowest delays lower, which would require EUROCONTROL or SkyGuide representatives to interpret and make conclusions on fairness and safety. Curfew performance was optimal for both methods. In conclusion, it is proven that Reinforcement Learning techniques can aid the dynamicity of decision-making within Target Time Management. It is also proven that Target Time Management with a dynamic decision making approach can improve operational performance compared to a static one.

1 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 due to its network structure, the consequences of disruptions are often catastrophic. At SWISS, since Zurich is located in the center of Europe, these disruptions are often heavy. While en-route disruptions are very common, in the case of SWISS, the ones in Zurich have a much larger effect. This is partly because Zurich is a hub, and partly also because the disruption intensity is often quite severe due to local weather and airport geometry. The type of disruptions which mostly affect SWISS are arrival regulations. These are when the demand to land at Zurich exceeds the reduced capacity due to external circumstances, and as such EUROCONTROL and SkyGuide (the Swiss Air Navigation Service Provider, ANSP) assign delays to all incoming aircraft, such that the new capacity can be met.

In 2019 alone, SWISS had a total Air Traffic Flow Management (ATFM) delay of 4500 hours only due to arrival regulations in Zurich. Smoothing this out, one can find that it equates to approximately 12 hours of delay per day. The consequences of these delays on operations are disastrous. Unfortunately, passenger connections are missed, flights are cancelled and bags are lost, and the profits of SWISS take a hit. As such, there is a very strong interest from SWISS to be able to better operate under these conditions, and somehow reduce the impact of the arrival regulations.

In the past there have been a lot of research attempts at arrival prioritization, which is the main proactive tool to reduce impacts of heavy arrival delays. These attempts first started with slot-swapping, which entails swapping around the arrival order of flights, such as in Psaraftis [1978], Vervaat [2020] and Hoogendoorn [2022]. The approach was 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

*Msc Student, Sustainable Air Transport, Faculty of Aerospace Engineering, Delft University of Technology

be expanded to pre-tactical to allow for more larger sequence changes. This is why EUROCONTROL released the Target Time Management concept, which allows airlines and ANSPs to submit their wished Target Times of Arrival (TTA) to the departure slot allocation algorithm even before the flight themselves depart.

SWISS has been the first airline to test the Target Time Management concept (during a project prior to this thesis) by building an operational Mixed-Integer Linear Programming (MILP) model to rearrange the arrival sequence. While it was possible to improve passenger connections significantly, the model had some major drawbacks. First of all, the setup of a sequence did not allow a lot of flexibility - it may be that at certain times of the day, there are only low priority flights, and at other times, there are only high priority ones. Moreover, it was also found that the system was highly dynamic, mostly because the requested TTAs sent to EUROCONTROL, not often matched the slot received. Often times, it was found that the type of TTA sent affected the one received: TTAs which delay flights had a higher chance of being accepted by the system than anticipation attempts. As such, the need for a more dynamic and future-looking algorithms is required.

This Master Thesis attempts to evaluate whether Target Time Management can be improved by 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 is why the approach taken will compare two Multi-Agent Reinforcement Learning (RL) algorithms to the baseline MILP approach. A simulation environment of the slot allocation system will be made, where the algorithms can train by performing actions. These algorithms will be then tested against the MILP first on new data (still within the simulation environment) and then also inside the real slot allocation environment. It is expected that the RL algorithms will increase the operational performance in terms of passenger connections, but most of the improvement is expected in rotation delays. This is because it was found that the MILP formulation had a trade-off between rotation delays and passenger connecting time, and it was chosen to shift this in favour of passenger connections during the summer for operational purposes. In RL this trade-off is expected to be less present due to the ability to consider future rewards. However, since the RL algorithms do not use a swapping process, it is also expected that they might struggle more with maintaining inter-airline fairness.

The thesis first starts with the literature review (section 2), where the literary gap is found. In here, the research question is outlined, and the methods with which it will be answered are described in section 3. Then, the results of the experiments are outlined (section 4), and discussed in section 5. Finally, conclusions regarding the overall goal of the thesis and initial hypotheses are described in section 6.

2 Literature Review

There are several methods to approach or formulate the flight prioritization problem. It is important to look into the format of this problem to see if there is any literary gap, as it is in SWISS's best interest to research this.

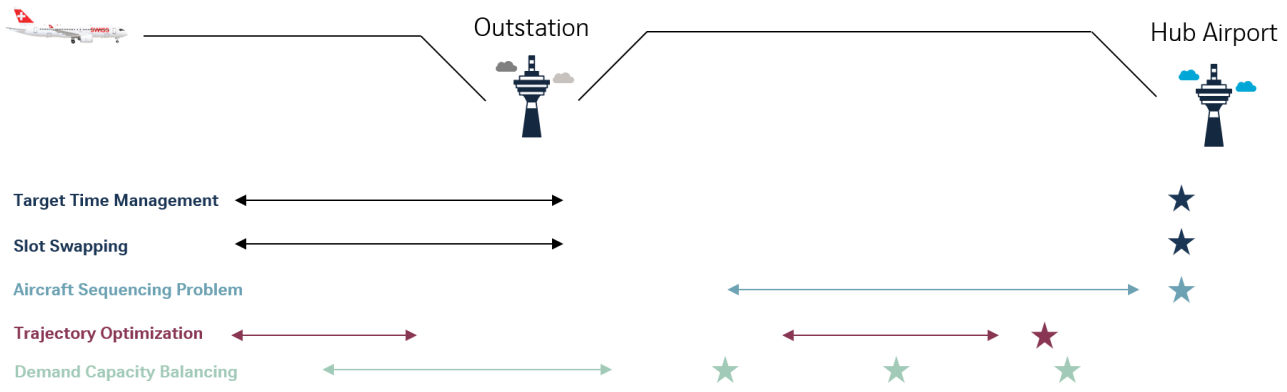

Figure 1: Overview of all the problems/decisions that can be made prior to the beginning of the turnaround procedure that effect the flight prioritizing process and their timeframes. The arrows represent the timeframe in which the problem is solved, while the star represents the time(s) when the decision's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

Figure 1 provides an overview of the five primary methods for flight prioritization. Instances that vary in terms of the durations of their resolution and impact are displayed here. The Aircraft Sequencing Problem (ASP) is the first method to be explored, which is also by far the oldest, initially implemented by Psaraftis [1978]. 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 (see Hoogendoorn [2022]), delay (see Cecen and Durmazkeser [2022]), deviation from the original schedule (see Du et al. [2023]), and passenger connections (see Montlaur and Delgado [2017]). Table 1 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 by Montlaur and Delgado [2017], was the main reason these approaches were not employed in operations, as noted by both Hoogendoorn [2022] and Vervaat [2020].

Table 1: The following is an overview of several objectives applied in literature regarding the Aircraft Sequencing Problem.

Publication	*Primary Delay*	*Reactionary Delay*	*Pax Connections*	*Fuel*	*Diff. to Schedule*
Du et al. [2023]					✓
Cecen and Durmazkeser [2022]	✓				
Cecen [2022]	✓				
Hoogendoorn [2022]	✓		✓	✓	
Vervaat [2020]			✓	✓	
Ikli et al. [2020]					✓
Montlaur and Delgado [2017]	✓	✓	✓		
Ng et al. [2017]	✓			✓	
Furini et al. [2015]					✓

Compared to the ASP, slot swapping is a more modern method of flight priority. It was conceived by EUROCONTROl and is intended to lessen the effects of situations in which the capacity for a certain airspace sector is exceeded by demand (while EUROCONTROL [2014] is not the original paper, it provides the best explanation). 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. Because the order needs to be determined before the flight takes off, it means that the priorities are decided much earlier. It can only be used for short-haul flights within the European airspace at this time.

Pilon et al. [2021] is one of the most significant literary works in the context of slot swapping. This method, which was used in a shadow mode trial with SWISS, attempted to adjust passenger connections and delays by automatically switching planes' slots, which is another term for CTOTs. Additionally, some slots were "extended" past their regular switching window. Although there were still a lot of improvements to be made, their efforts showed a reduction in the amount of passengers missing connections by as much as 12.5%. However, it was clear that the decision-making tool lacked dynamicism and assumed that EUROCONTROL would accept every CTOT request; nevertheless, SWISS subsequently discovered that the approval rate is only about 80%. Furthermore, their method only permitted switching between pairs of flights, which can result in a less-than-ideal order. Evler et al. [2021] and Schuetz et al. [2022] made other noteworthy literary projects. While the latter employed evolutionary algorithms and a privacy-preserving strategy to enable the sharing of sensitive data, the former represented the Slot Swapping problem as a MILP that included only 15 turnarounds as a test case. Both authors, however, faced a similar problem: in order to ensure that these technologies can be implemented in an operational department, a more thorough evaluation of the network effects is necessary. Regrettably, neither author had access to enough airline data to do this.

The third approach to flight sequencing is Target Time Management. This, ideated by EUROCONTROL and best described in the most up-to-date version EUROCONTROL [2023], is the same as slot swapping when it comes to timeframes and effects (see Figure 1), but with a series of differences. First of all, 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 this, the way one requests a change of CTOT is via a Target Time of Arrival, which then EUROCONTROL calculates back to a CTOT. This is highly functional for the purpose of affecting the arrival order. Finally, there is no literature available for the Target Time Management concept, which is both a blessing in terms of opportunity but also a curse in terms of known dynamics. The CTOT (and thus this arrival time) can be changed up until the start of the Departure Planning Information sequence, which means 10 minutes before the Target Start-up Approval Time [EUROCONTROL, 2022].

Prior to this thesis, the notion of Target Time Management was researched at SWISS, SkyGuide, and Zurich Airport [Caranti et al., 2023]. Even though Heathrow Airport and Paris Charles de Gaulle Airport are known

to use this concept to optimize runway throughput, there are no publications or mentions of this available to the public - this meant that SWISS' investigation by Caranti et al. [2023] marked the first time an airline was able to influence Target Times of Arrivals. The tool's 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 Mixed-Integer Linear Program, although it, too, suffered from a lack of dynamicity. This is especially true because EUROCONTROL's slot assignment algorithm does not always award the necessary CTOT due to shifting demand in the airspace. Furthermore, due to the necessary linear and non-stochastic structure of a Mixed-Integer Linear Program, long-term network impacts, as well as future effects of present actions, were not completely taken into consideration in the decision-making process. However, it is both in SWISS's and literature's interest to improve this tool further, given that the issues encountered with the current Target Time of Arrival approach are very similar to those encountered with many other flight prioritization approaches previously mentioned.

In Figure 1, two more problem examples are shown: Demand Capacity Balancing and Trajectory Optimization. Despite the fact that they are crucial in terms of flight prioritization, they are significantly different from the problem SWISS is attempting to tackle and are also extensively documented in the literature. As a result, they have been left out of this literature section because they are too far outside the focus of this study.

Overall, major literary gaps were discovered as a result of 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 flightăprioritization. Furthermore, in order to truly benefit from this, validation appears to be a critical step, as most publications were unable to establish an operational system. The ability to validate the thesis using a live trial and compare it to the live system established during the project prior to this thesis at SWISS [Caranti et al., 2023] would add a lot of literary appeal. Finally, the opportunity to put the Target Time Management strategy to the test aids literary efforts in this relatively unexplored subject. If successful, this would not only demonstrate the possibility of a dynamic flight prioritization process via Target Time Management, but would also improve SWISS International Airlines' operations.

As such, the research question was formulated as follows:

RQ: 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 MILP or with Dynamic Programming, they often lacked dynamicity. This can be seen in Vervaat [2020] (MILP), in Muharremoglu [2000], or even in the previous work done by Caranti et al. [2023] at SWISS on Target Time Management. As such, 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. Possibly, it would be great if this method could also be modelled using multi-agent systems, since in arrival management there are always a varying number of flights.

Because of all the above reasons, an approach using Reinforcement Learning (RL) will be taken, and described further below. As explained in Sutton and Barto [2018], this is a perfect tool to adapt to complex environment dynamics as well as taking into account future states. Moreover, it is possible to structure multi-agent RL systems, which are widespread choice within the complex air traffic management (or more broadly, aviation) domain, as reviewed in Razzaghi et al. [2022].

3 Methodology

To ensure a method which attempts to answer the objective of this thesis, a step-wise approach is taken. Firstly, the relevant data is gathered, explained in section 3.1, then the environment is modelled around it (section 3.2). This ensures a possibly realistic modelling of the real slot-allocation dynamics. After this, one or more suitable algorithms are chosen for the environment, outlined in section 3.3, which are then trained and tested.

3.1 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.

The days are filtered based on the quality of the data and their representativeness to situations where the tool would be used. For instance, 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. Moreover, 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, since the TTA procedure in place at EUROCONTROL can only affect short-haul flights, these are the only ones which can receive TTAs.

3.2 Reinforcement Learning

Reinforcement Learning is a machine learning technique which is characterised by a unique set of elements, differentiating it to other optimization approaches such as MILP. 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. On the other hand, a MILP can only have simpler, linear objectives and constraints, and the environment dynamics have to be engrained in its formulation and inputs. However, while RL will output the most likely action(s) it thinks will maximize future rewards, a MILP will always find the optimal action for the conditions it is formulated in, despite typically having to take more assumptions in the process.

In this section, each part that has to be formulated for a RL algorithm to be trained is outlined and analysed. Firstly, the environment and its dynamics are analyzed in section 3.2.1. Then, how each agents is assigned to flights is described in section 3.2.2, followed what the state comprises for each agent in section 3.2.3. Following this, the approach to modelling actions in outlined in section 3.2.4, finally followed by the modelling of the reward in section 3.2.5.

3.2.1 Environment

The environment where the Target Times of Arrival can be set through agent's actions represents the real operational environment of SWISS. Here, requests are made to Eurocontrol, but sometimes the CASA algorithms 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. Besides this, it is also important to describe how a stepwise increment in time looks like in the simulation within the environment. Both of these two elements will be described in this subsection.

Environment Step Behaviour
Most environments are discretized into timesteps where agents can make actions. In standard reinforcement learning applications, environments are often standardied to be used with python packages which aid training. In order to do so, they are often configured using two functions: a reset and a step function. Since it is in the interest of the authors of this paper to use Reinforcement Learning as a tool to solve a larger problem and rather not as the main content of the research, a python package will also be used to include the RL algorithms: RLLib Liang et al. [2018]. Another reason why this package was used is that it allows for switching between algorithms without having to code them all from scratch. To use this, it is necessary to also configure the environment into a step and reset function.

While the reset function only brings back the environment to the start of a randomly chosen simulation day and resets all agents to the first flights of the day, the step function is more complex. Its details in algorithmic format are described in Algorithm 1, but its general structure is outlined as follows. First, the function takes the state and set of active flights as input, and returns the new state at the next timestep. Within the function, the following processes take place: for each flight (and thus for each agent), the RL algorithm is sampled and an action is retrieved (more detailed information on the action formulation can be found in section 3.2.4). Then, the time t is increased by a timedelta td, and the response by the CASA algorithm is simulated. Then, the rewards are computed, and finally the new active flights are found. Like this, the next state can be built and returned by the function. For RL, the timestep td was set to 30 minutes, while for MILP it was set 60. This is because 60 is the value used currently in operations, but it is expected that by having more future-reward considerations, more frequent actions will create more opportunities for improvements for RL. On the other hand, a timestep as small as 30 minutes would either create redundancy or increase oscillations in actions in the case of MILP, since it does not take into account future states. Note that in Algorithm 1, STA_f is the Scheduled Time of Arrival of flight f.

Environment Regulation Behaviour

Algorithm 1 Process to take a step 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: **for** each flight f in F **do**
4: a_f := Sample RL algo for an action, based on current state S.
5: $TTA_f := \max(STA_f,\ STA_f + a_f \cdot d)$
6: **end for**
7: $t = t + td$
8: Compute response by CASA algorithm:
9: Separate delayed flights from anticipated ones
10: p := sample probability of achievement of the TTA
11: x := sample from $X \sim U(0, 1)$
12: If $x > p$:
13: Find new TTA for the flight, not achieved
14: Else:
15: Accept TTA
16: Update estimated arrival times
17: Get rewards by comparing to original schedule
18: Update schedule:
19: F := New active set of flights if any
20: $F_{dep} \in F_{orig} \setminus F$ (flights in F_{orig} but not in F because they have departed)
21: Remove agents assigned to departed flights F_{dep}, assign them to other new flights.
22: S := new observations, making up the new state
23: **Output** S

Since RL is designed to perform under dynamic environments, it is a great opportunity to model one for this use case. More specifically, the slot assignment algorithm which receives the requested TTAs tends to be highly dynamic. This was one of the major disadvantages when using MILP, which assumed that each requested TTA would be assigned, as outlined in section 2. Because of this, specific dynamics have to be first modelled, then taken into account in the environment.

The location within the step function where these dynamics are included is in line 10 of Algorithm 1. The slot allocation dynamics are modelled in two parts: 1) first find the probability that the requested Target Time of Arrival will be exactly accepted, and then 2) in case it was *not* accepted, find the new received slot instead of the requested one. As a first step, 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. Data was collected between the first of July and the 15th of September 2023 to evaluate these.

Starting with the probability of acceptance of a TTA, the data regarding the time elapsed before it was accepted was fitted to an Empirical Cumulative Distribution Function (ECDF), in such a way that given a specific time it is possible to read the probability directly from this graph. The results of this are shown in Figure 2. 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. If the model learns, it is expected that it compensates this imbalance by requesting more anticipation 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 3, 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

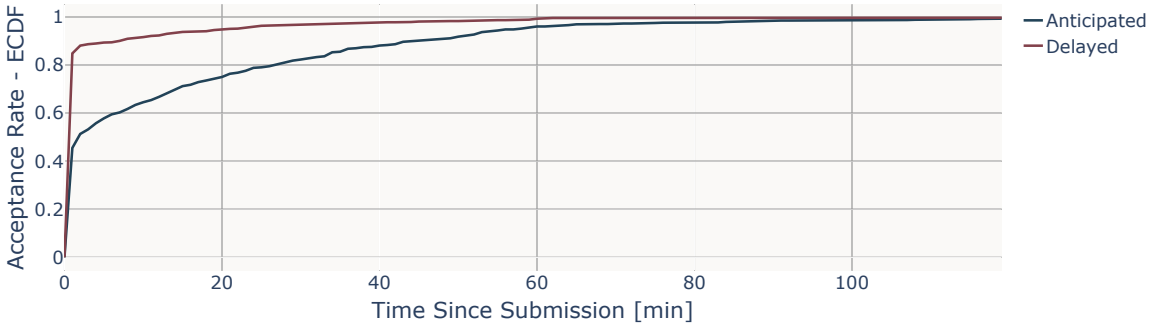

Figure 2: Empirical Cumulative Distribution function for probability of acceptance of a TTA, fitted on data from the 1st July to the 15th September 2023.

have less anticipation than requested up to 120 minutes (factor is less than one). The standard deviation plot in Figure 4 describes the confidence level of each value in Figure 3. 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 in Figure 4 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.

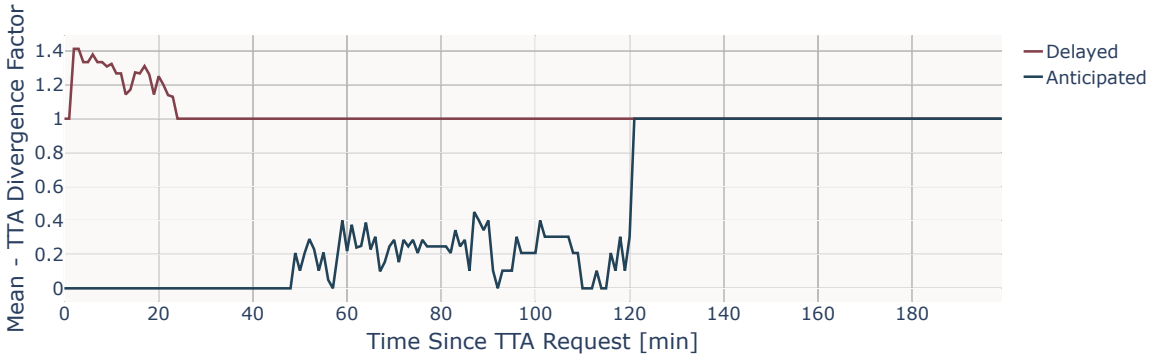

Figure 3: Mean of the Divergence Factor (ratio) between the requested TTA and the actual one.

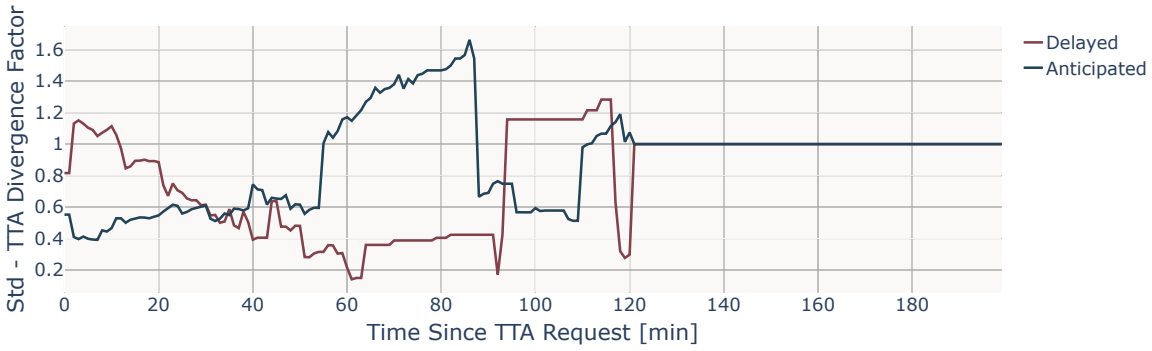

Figure 4: Standard deviation of the Divergence Factor (ratio) between the requested TTA and the actual one.

Besides the slot allocation dynamics to a requested TTA, for a valid simulation one also needs to apply the slots in the first place. This means that given a schedule, each flight subject to a regulation must be delayed. In order to do 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

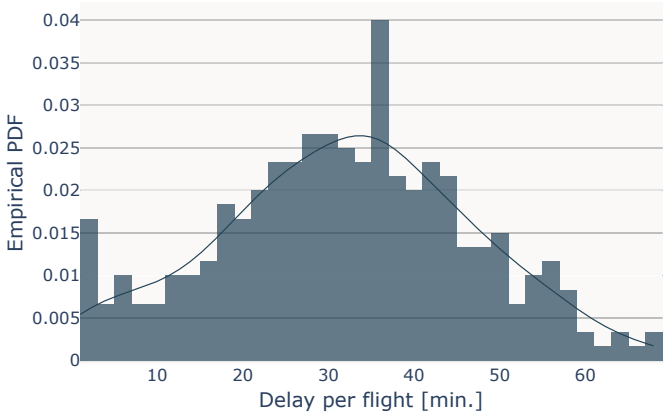

Figure 5: Stochastic simulation of Algorithm 2 for 500,000 days with 300 flights per day.

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 full procecure is seeded for reproducibility. The procedure for assigning the delay is outlined in Algorithm 2. The result obtained when this set of distributions is simulated for 500,000 days with 300 flights per day is shown in Figure 5 with an empirical Probability Density Function (PDF).

Algorithm 2 Process to initially assign delays to flights.

seed := 0
for each day t in T **do**
  Sample mean delay $\mu_d \sim U(20, 50)$
  for each flight f in F **do**
    Sample flight delay $d_f \sim N(\mu_d,\ \mu_d/2)$
    $d_f = \max(1\ [min],\ d_f)$
    Assign delay d_f to flight f.
  end for
end for

3.2.2 Agents

The core of reinforcement learning 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. Since the total number of flights in one day is very large, yet the number of active flights which can receive a Target Time of Arrival at the same point in time is still limited, the 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 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. Then, not all sixty agents are activated at the same time. Whenever a flight is activated, the next available agent is set to it, and when the flight is deactivated the agent is detached from it. This implies that there is no difference in the agents between the flights. This is because, while 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 very 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.

The disadvantage of having sixty rotating agents is that it might take more time to train the algorithm. An alternative solution to this is to take the five or ten most critical flights and have a fixed amount of agents, assigned to those. However, this not only is highly sub-optimal (since it reduces the number of flights that the tool can affect), but it also tends to captivate less the effects on future reward of the algorithm, again because

many less critical flights might be affected in future states. As such, the approach using sixty agents which are not all active at the same time is chosen.

3.2.3 State

The state of the environment is set up in a way that each agent is not only informed about his own state, but has an idea of other states of neighboring flights as well. This is very important 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 be informed about their states as well. The state, with inputs described in Table 2, 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).

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

Table 2: List of inputs which make up the state vector in the environment. It is important to note that some of these are included also as 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.

As described in Table 2, passengers are weighted. In order to calculate their weightings, they are first grouped based on their flight duration (short-haul, long-haul) and booking class (economy, premium economy, business, first). Then, their business value is computed. The discussion of how these values are achieved is beyond the scope of this article, and the actual values are deemed as sensitive commercial information, and as such will not be disclosed. However, it can be said that their values are correlated, but not equal to, the fares that the passengers pay to the airline. Besides this, non-monetary value is also included in the weighting. For instance, a happy customer which recommends the airline to their friends is more valuable. Since SWISSs offer on economy long haul may have a different quality than business short haul, their non-monetary value is also different. This should also be included in the weight of each passenger. For each connection, this weight is multiplied by the number of passengers in the connection, and then used in the state under the *Number of Critical Connections* input.

3.2.4 Action

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 teh 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. For more information on the algorithmic formulation, see Algorithm 1.

3.2.5 Reward

The other important part of the MDP is the reward function. This is meant to encompass all the aspects of the feedback system which the agent should learn how to maximize. 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 the global weight being 0.3 and the local one 0.7. These values were found by starting with a high local reward ratio (0.9), which is easier for the tool to optimize, and decreasing it until 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 one is just calculated by taking the average across agents. Moreover, the normalizing or scaling factors for each reward segment are shown in Table 3. These numbers also show the apparent equivalence rations of each reward type when compared: for instance, like will be discussed in the following sections, 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.

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	-

Table 3: List of scaling or normalization parameters used in the environment reward function.

These values, along with many more normalization and weighting values which will be discussed in this section, were chosen according to manual hyper-parameter tuning or by consulting operations controllers at the Operations Control Center of SWISS. It is also important to mention that all the rewards were negative and flights were penalized when performing poorly, but not directly rewarded when performing well, with the exception for passenger connections and delays. This is expected to work 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.

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. The only connections considered are the ones which either used to be less than 45-minutes before the action of the tool, or which are less than 45 minutes after the decision of the tool. For every connection, the change (new connecting time minus the old one) in minutes is multiplied by the amount 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 3.2.3, 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 as described above.

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.

Curfew

Zurich Airport, like other airports in Europe like Amsterdam Schiphol or Frankfurt, 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.

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.

In this paper, since this is well beyond the scope of this research, this process is modelled in a simple way. Whenever a flight ends up with more delay than it had initially, the difference is store in 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. This system would prove to be fair only in the case where the initial allocated slots to flights are fair. Since both in the real world and in the environment they are assigned independently of the airline, but solely on the regulation window and time of arrival of the flight, this assumption proves to be true. Since the time of arrival of a flight is a choice of the airline, any remaining concept of unfairness, using this method, can be proved as the result of an airlines own choices, and not because of Target Times of Arrival.

Based on personal discussions with members of EUROCONTROLs Network Manager department, in their eyes fairness only become an issue when parties not using Target Times of Arrival are penalized. This means that whenever the delay credit is negative (so, SWISS has anticipated too many flights), there is a fairness issue. This means that 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 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 3 for the equivalence between reward types) and effects on training performance of the algorithm (evaluated using operationally valid metrics).

3.3 Algorithms Chosen

The algorithms chosen to carry out the Reinforcement Learning are two. Firstly, the model should be trained with a simpler algorithm, possibly deterministic, which ensures that the environment can be learnt. Even though the extent to which this simpler algorithm can understand complex dynamics is more limited than others, it may be that this algorithm obtains a high level of optimality. Then, this should be compared to a more complex algorithm, possibly one with a stochastic policy. This would allow to see if there is the need for agents to learn more complex dynamics, at the expense of possibly more training time.

The two algorithms that were chosen for training are Proximal Policy Optimization (PPO, originally developed by OpenAI) and Soft-Actor Critic (SAC, originally developed at UC Berkeley) [Schulman et al., 2017] [Haarnoja et al., 2018]. PPO is considered a simpler algorithm, because it only optimizes one policy and is deterministic. On the other hand, SAC optimizes both the actor's policy and the critic's, while being stochastic in nature. It is expected that for operational reasons, PPO might be seen as more stable due to its deterministic nature, but SAC might be able to generalize better. Besides this, PPO is also an on-policy algorithm, while SAC is off-policy. This difference lies in the fact that while PPO trains and takes actions using the same policy, off-policy algorithms have two policies: one to training on and one to sample actions. After a number of steps, the action policy is updated with the trained one. In Haarnoja et al. [2018], it was found that the main advantages of off-policy algorithms like SAC are related to a better exploration-exploitation balance and higher sample efficiency. On top of this, 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 [Haarnoja et al., 2018]. In contrast, in Kalidas et al. [2023] it was found that PPO was the least successful RL algorithm when it came to complex, dynamic environments, and that off-policy algorithms had the upper hand. However, all these benefits of off-policy algorithms come at the cost of a longer training time, since it is required for more exploration.

It is important to note that with the nature of the problem, a multi-agent system is favorable, mainly due to

two reasons: firstly, Multi-Agent Reinforcement Learning (MARL) has a better ability to achieve cooperation between flights compared to classical RL and secondly, the number of flights is always varying. 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 3.

Algorithm 3 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
```

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 [Haarnoja et al., 2018], [Schulman et al., 2017]. The chosen hyperparameters can be seen in Table 4 for PPO and in Table 5 for SAC. For both algorithms, the strategy used for exploration is Stochastic Sampling [Huang et al., 2022], while the optimizer used was Adam [Kingma and Ba, 2017].

Hyperparameter	*Value*
Learning rate	$5 \cdot 10^{-5}$
Discount factor	0.9
Training batch size	4000
No. layers	2
Layer size	256
Nonlinearity function	ReLU
Clipping parameter	0.3
KL Coefficient	0.2
KL Target	0.01
GAE λ	1.0

Table 4: Chosen hyperparameters for PPO.

Hyperparameter	*Value*
Learning rate	$5 \cdot 10^{-5}$
Discount factor	0.9
Training batch size	256
No. layers	2
Layer size	256
Nonlinearity function	ReLU
Initial α	1.0
Target smoothing coeff. (τ)	$5 \cdot 10^{-3}$

Table 5: Chosen hyperparameters for SAC.

4 Results

The results section is divided into four parts. Firstly, section 4.1 describes the analysis steps taken to evaluate the results. The second section (section 4.2) focuses on the training of PPO and SAC. The third (section 4.3) focuses on the comparison between these two and the MILP on new data (but still in a simulation environment), and the fourth tries to compare the performance of the PPO algorithm when tested in a real environment to the MILP that has been used in operations in the last 6 months.

4.1 Analysis Methods

To analyse the results, it is important to monitor a series of metrics. Initially, all of the reward components have to be analysed, to see whether the tool is learning at all. These can be found better outlined in section 3.2.5. Moreover, more metrics have to be analysed, to see whether the reward setup then translates into operationally valuable decisions. Just like the reward, these operational metrics are also split in sections. Firstly, the passenger connecting time performance has to be analysed, followed by delay performance and fairness. Then, it would be interesting to see how the tool decides to take actions: for each operationally relevant input (critical connections, connecting time, delay), it would be great to check if the tool prioritizes a flight which is more critical than the other flights in the same group.

Another very important comparison is one with the Mixed-Integer Linear Programming formulation developed by SWISS. This is because, besides validating the method, it is possible to compare whether the trained models perform better than the ones already in operations. If it does, besides being a great research achievement, it would make it interesting also for SWISS to be able to deploy the models in operations (after sufficient testing). The basis of the comparison will be using the metrics described in the previous paragraph, with special focus on the action choice and model behaviour. The MILP model formulation can be found in Appendix A.

Finally, it is important to analyse the performance of the trained RL models outside the simulation environment, 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. If the tool also performs well under these conditions, it would prove that the environment which it was trained on is sufficiently similar to reality.

4.2 Training Results

During the training procedure, both PPO and SAC were run on data from the 9th May 2018 until the 20th 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, like 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 20th April 2023 until the 26th May 2023 was used as further testing for comparison of the MILP to the RL algorithms. 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 be seen from 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. Also the total number of iterations trained is around 15 times more, as can be seen from Figure 7, 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), it can be seen from Figure 7 that the reward tends to grow slower over time. This is because, while PPO optimizes the first best policy it can find, SAC has a stronger focus on exploration, hence sometimes deviating from the best policy.

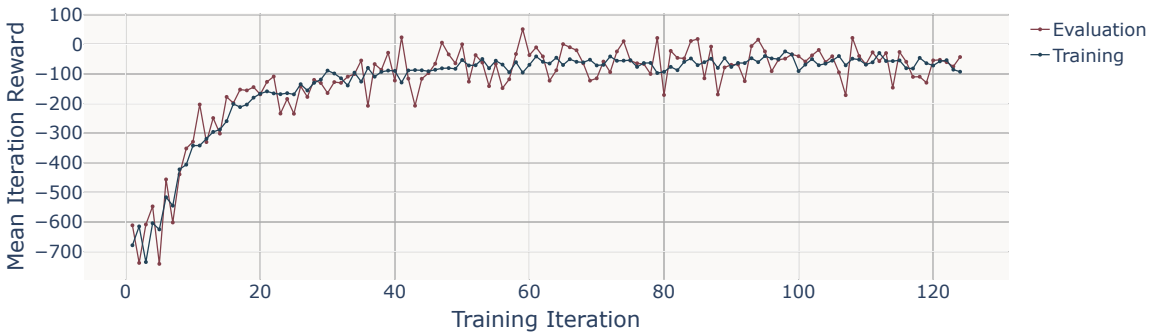

Figure 6: Training overview of the PPO algorithm.

What is interesting to look at is how the reward evolves during training. Since as explained in section 3.2.5, the reward function has various parts, it is important to see the evolution of each one of those parts. This is shown in Figure 8 for PPO and in Figure 9 for SAC. 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

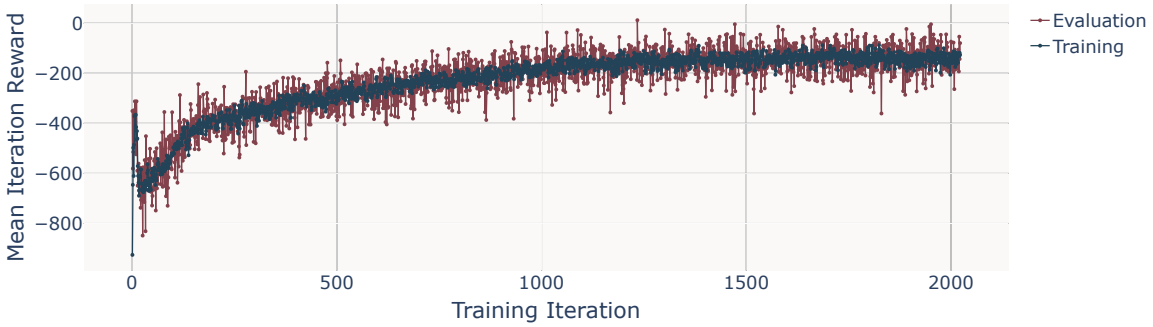

Figure 7: Training overview of the SAC algorithm.

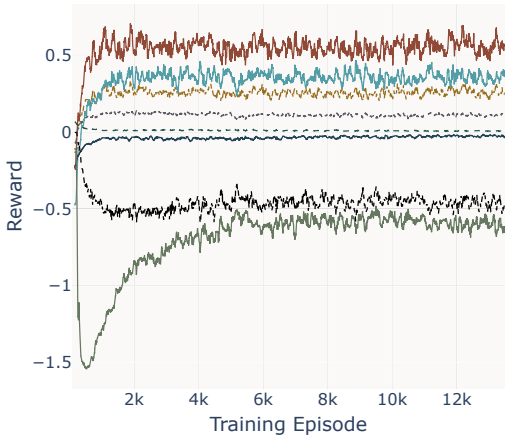

Figure 8: Evolution of each reward element during training for PPO.

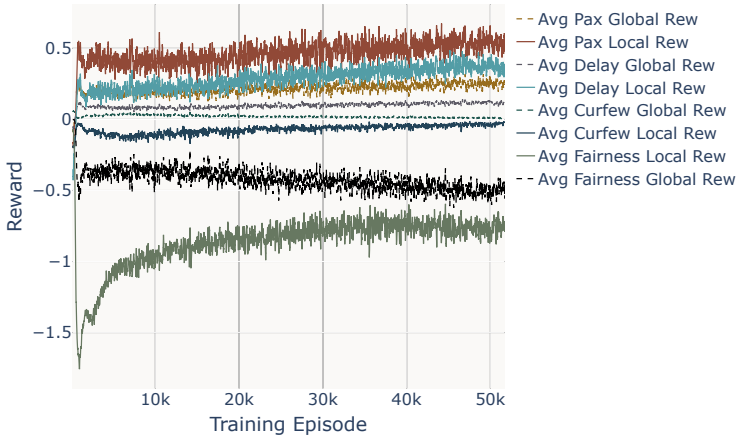

Figure 9: Evolution of each reward element during training for SAC.

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 reward for PPO seems stable after the initial peak. This is because 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 10 for PPO and in Figure 11 for SAC, 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, it can be seen from Figure 10 that 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 aggresively 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 2, two are specifically interesting, and are both inputted as individual and average values. This means that 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. The two are balanced for each agent, and this is compared to the action that the agent takes. In

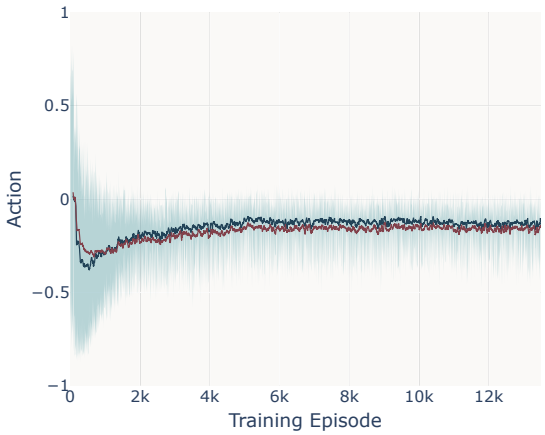

Figure 10: Man action range evolution during training for PPO.

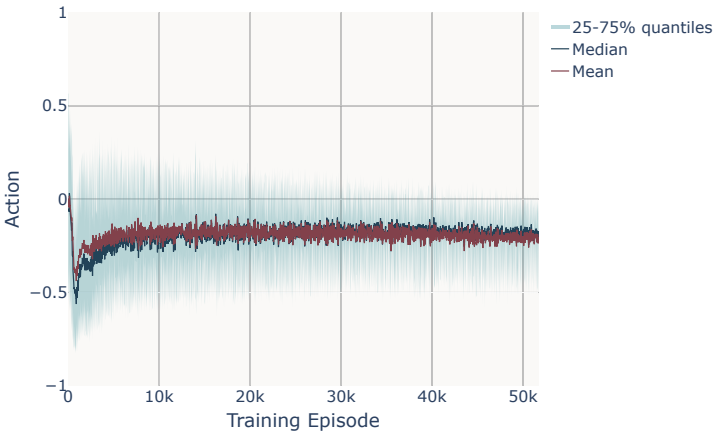

Figure 11: Mean action range evolution during training for SAC.

Figure 12 the results are shown for PPO, and SAC is shown in Figure 13. 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 like after episode 11,000 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 after training iteration 100 being seemingly flat in Figure 6. 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.

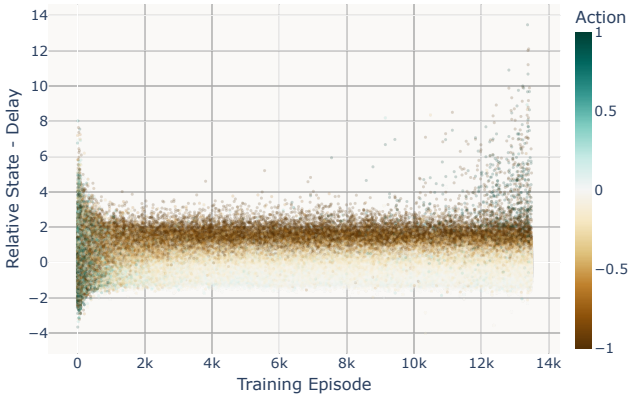

Figure 12: Action sensitivity to relative difference to batch mean delay during training for PPO.

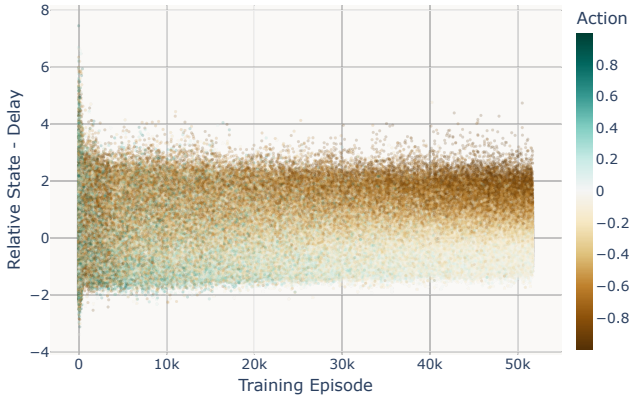

Figure 13: Action sensitivity to relative difference to batch mean delay during training for SAC.

Another input which can be analysed in a similar format is the number of critical connections. The results are shown in Figure 14 (PPO) and Figure 15 (SAC), where 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 flight 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 decides to not change those flights' delays (action is around zero), while SAC tends to delay those flights slighlty further (notice a greenish line in Figure 15).

It has to be pointed out that there seems to be a orange line at the very lowest end of the y-axis values for both

Figure 14 and Figure 15. This could be explained by flights with zero critical connections, which have very high delays. Even though 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).

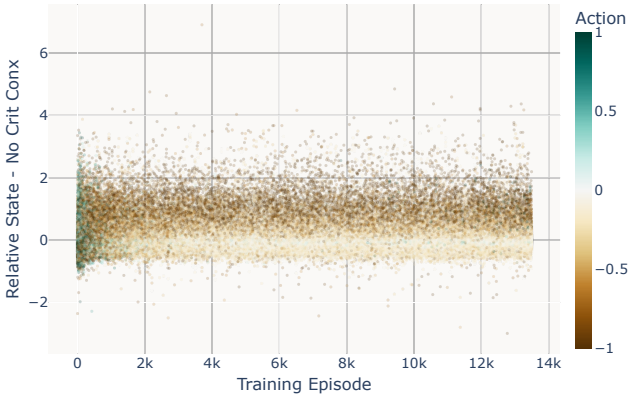

Figure 14: Action sensitivity to relative difference to batch mean number of critical connections during training for PPO.

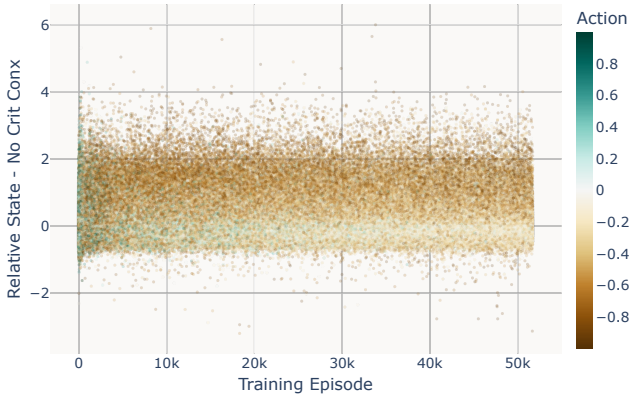

Figure 15: Action sensitivity to relative difference to batch mean number of critical connections during training for SAC.

Finally, it is also interesting to see how the mean delay credit at the end of each episode evolves through training. The results in Figure 16 (PPO) and Figure 17 (SAC) both show how 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 (Figure 8 and Figure 9), delay credits remain more or less constant for PPO and slightly decreasing for SAC. This is because delay credits are essentially proportional to global fairness, which also is shaped similarly for both algorithms. It is important to point out that having a lot of anticipation is not bad, as long as it is not asked for in a way which disadvantages other airlines. Whether this holds in a real environment is a very interesting question which shall be analysed at a later stage.

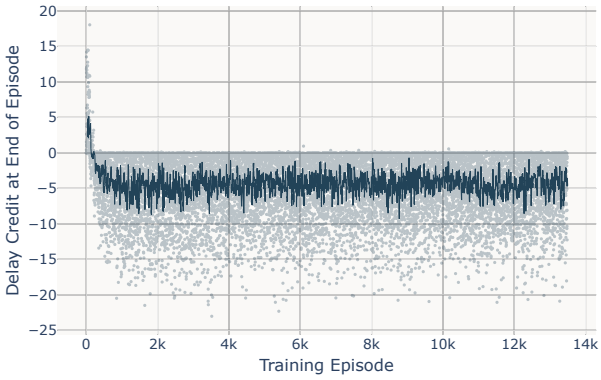

Figure 16: Delay credit at end of episode evolution during training for PPO.

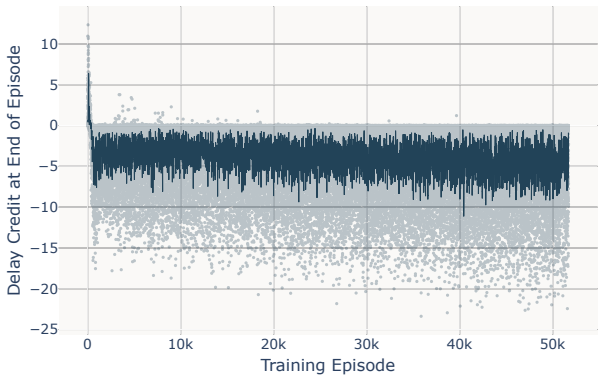

Figure 17: Delay credit at end of episode evolution during training for SAC.

4.3 Testing Results

After training, it is important to see how the PPO and SAC algorithms perform when tested on new data. When doing this, it is also worthwhile to compare them to the tool which is currently in operations (MILP approach). For this, data between April 23rd and May 26th 2023 was used. First, it is interesting to look at how the algorithms scored for each reward type, shown in Figure 18. Starting from the left, it is possible to observe how while the MILP has relatively high fairness rewards, the RL algorithms both struggle, especially SAC. While this may seem a fundamental aspect for the tool to be operational, it is important to consider how this is still a simulation environment designed to penalise this very much. The MILP, 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 18 that the RL algorithms both outperform the MILP. Interestingly, it also appears that SAC always has an edge on PPO when it comes to unseen data, since the rewards are always slightly higher, despite the training reward being slightly lower.

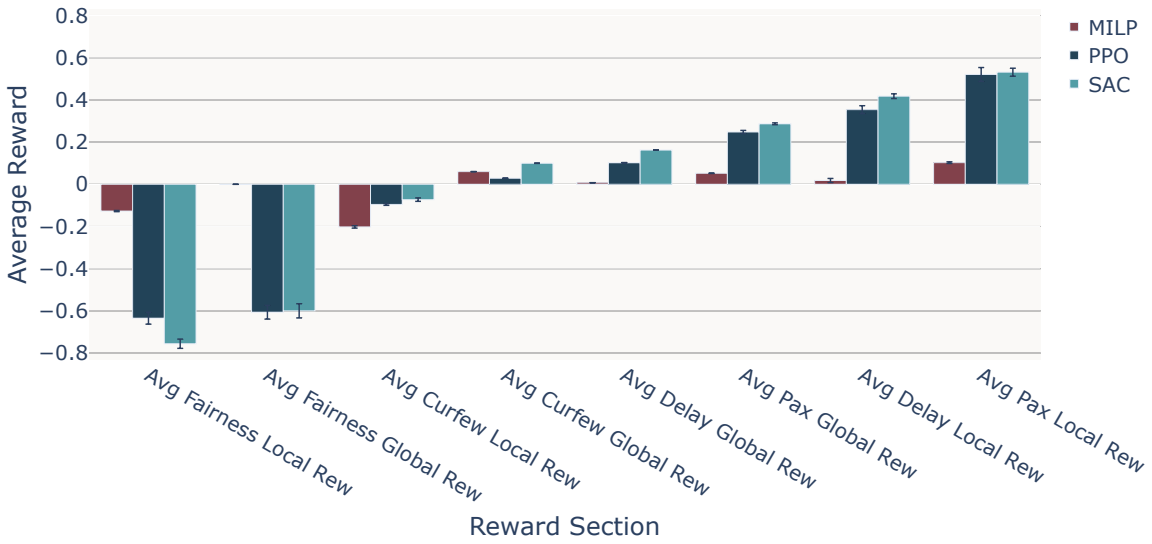

Figure 18: Average reward comparison during testing, split by reward section.

Reward is important for model evaluation, but it only represents a mixed approximation of valuable operational outcomes. Because of this, it is also important to evaluate the models in terms of true operational metrics. This is why an analysis has been made for improvements in critical passenger connecting times (shown in Figure 19) and improvements in rotation delay (shown in Figure 20). 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 PPO (5.1), and MILP is last, with about half of the improvement of the RL 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 9th, for instance, where PPO performs best. On the other hand, in days like May 18th, SAC performs better than the other two by a large margin. Because of this, it would be very interesting to analyse at daily level what the differences between the algorithms is. This will be further developed in section 4.3.1.

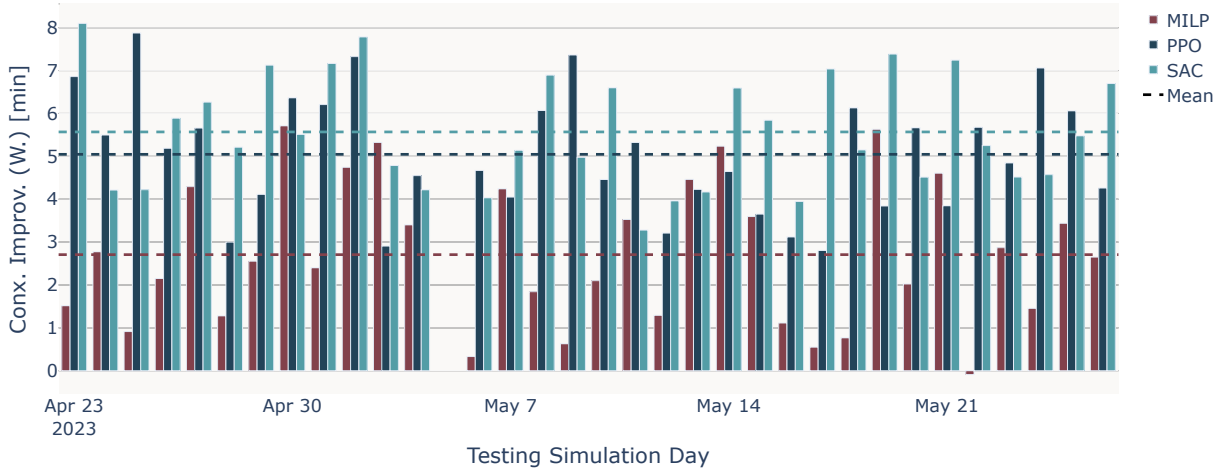

Figure 19: Improvement Minutes per Weighted Passenger for every simulation day, split by algorithm type.

It is also possible to observe the rotation delay improvements for each testing simulation day in Figure 20. 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 MILP barely has any improvement (0.2 minutes). The poor performance of the MILP 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 10th), days where PPO performs much better (May 9th), and days where SAC is better performing (May 18th). The reasons behind this will be investigated further in section 4.3.1.

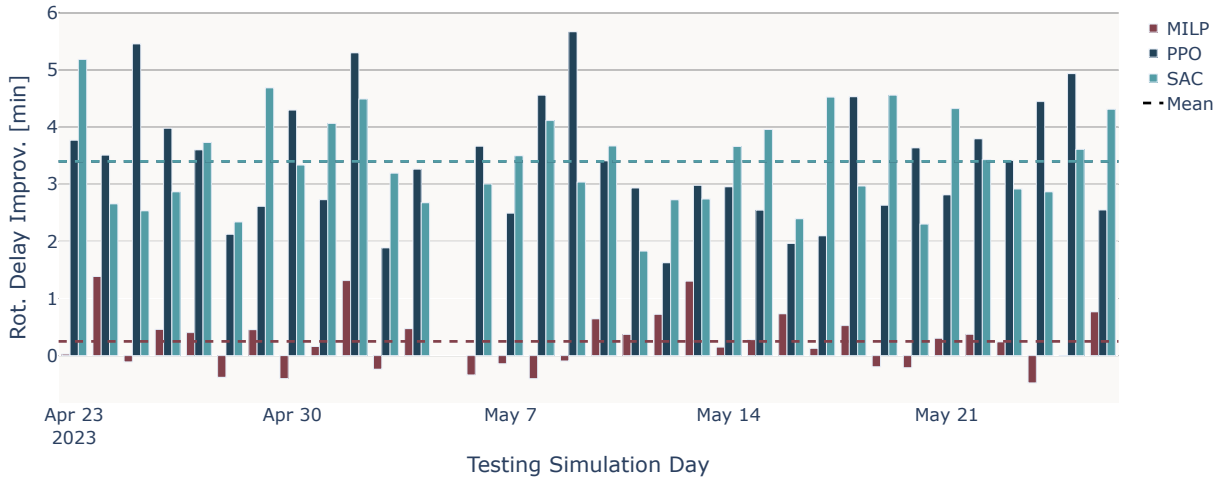

Figure 20: Rotation Delay improvement for every simulation day, split by algorithm type. The PPO and SAC mean dashed lines overlap, and because of this only the latter is visible.

One of the main reasons behind MILP 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 21, Figure 22 and Figure 23. As can be seen, there is a weak Pearson's correlation coefficient (r=0.11) for MILP 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 RL 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 MILP, 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 RL instead of MILP due to the ability to consider future reward.

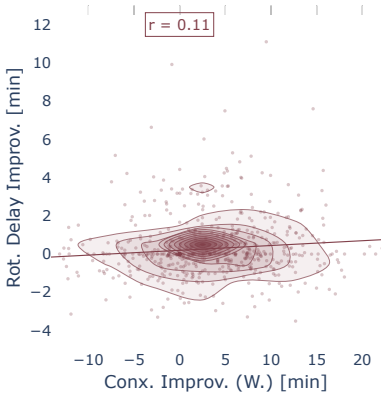

Figure 21: Correlation plot between connection and rotation delay improvement for MILP.

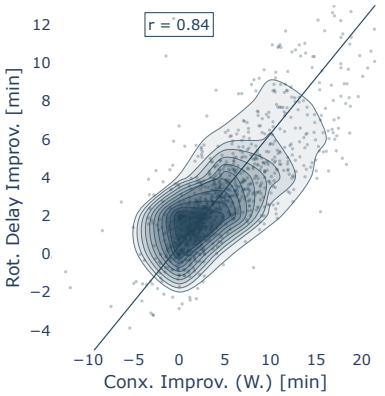

Figure 22: Correlation plot between connection and rotation delay improvement for PPO.

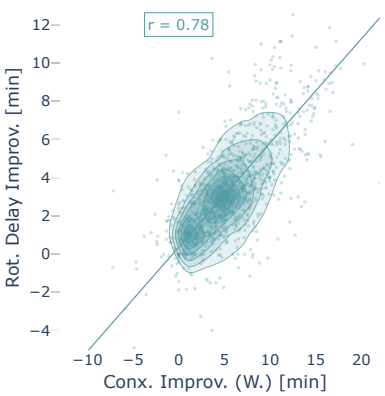

Figure 23: Correlation plot between connection and rotation delay improvement for SAC.

Another interesting analysis can be done regarding the type of actions made by each of the three algorithms. As can be seen in Figure 24, the three have very different action distributions. This plot shows the probability distribution function for the whole action range, for each algorithm. Starting with the MILP, it can be seen how it well distributed along the whole range of possible outcomes, with peaks around -1 (30 minutes of anticipation)

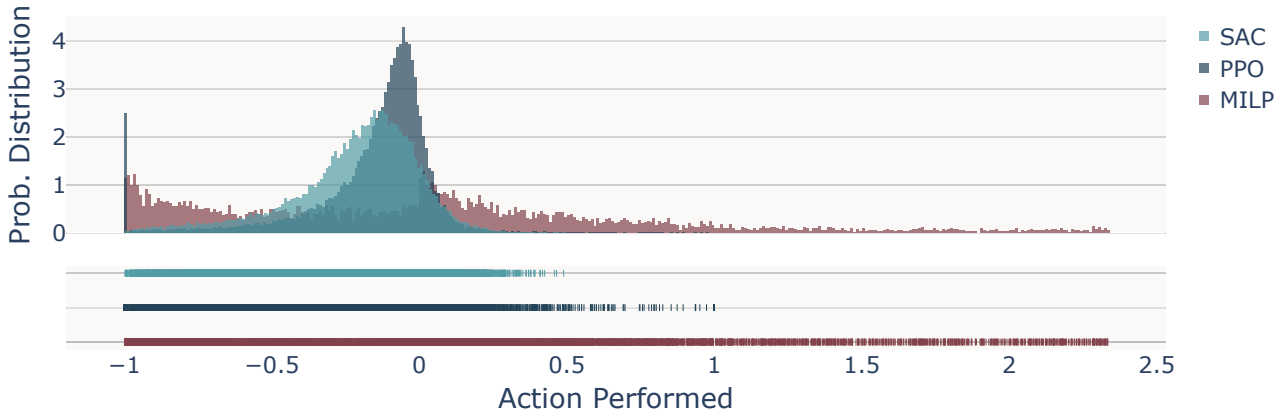

Figure 24: Action probability distribution (probability density) during testing.

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 this 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, initially described as more aggressively increasing delays in Figure 13 and in Figure 15, 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 MILP 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.

4.3.1 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 (9th May and 18th May respectively), one can look for flights which appeared in comparable ways to the algorithms, but the algorithms but behaved differently. Starting with the 9th of May, when PPO performed better, flight LX1629 (from Milano Malpensa to Zurich) is chosen as an example. From Figure 25, an overview of the behaviour of the algorithms for this flight is seen. 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 anymore. 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. During the simulation of May 18th, flight LX1679 is analysed in Figure 26 (from Florence to Zurich). 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 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.

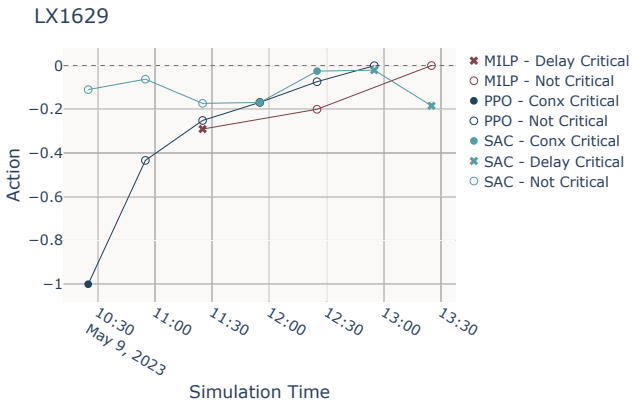

Figure 25: Actions performed on flight LX1629 on the simulation day of May 9th 2023.

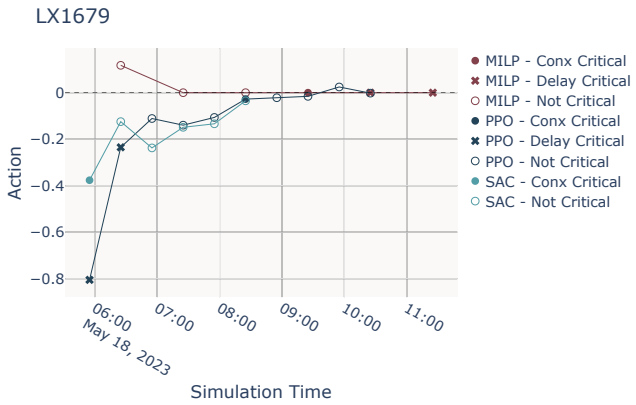

Figure 26: Actions performed on flight LX1679 on the simulation day of May 18th 2023.

A final note has to be made regarding their difference in performance which cannot be seen from these plots. Being a deterministic model, PPO will always choose the same action for the same inputs. Nevertheless, being a stochastic model, SAC might not do so. Because of this, even though SAC's performance is slightly superior, PPO is preferred for operational reasons. This is because the added advantages in reliability in stability (for what a RL model can guarantee) outweigh the marginal increase in operational performance. As such, PPO is chosen for testing in a more realistic environment than the one modelled for training.

4.4 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 EUROCONTROL [2023]. 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 MILP 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 MILP 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.

The two algorithms were tested and compared on EUROCONTROL's PreOps environment between Monday 4th December and Friday 8th 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 MILP 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 6.

Parameter	*MILP*	*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%)

Table 6: Real environment testing conditions for the MILP and PPO algorithms.

From Table 6 it is already possible to see how the two approaches differ greatly. First of all, while the MILP 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 MILP 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 by the PPO algorithm was nearly three times as much as the ones sent by the MILP one (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 MILP approach, operations controllers at SWISS decided that it would only send updates once every 60 minutes.

One last comment regarding the values in Table 6 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 believe 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 7 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 MILP 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), the improvement in rotation delay is nearly 5-fold. This is mainly due to the dynamics observed in Figure 22, 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.

Key Performance Indicators	*MILP*	*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%

Table 7: Operational Key Performance Indicators.

Even though the operational performance improvement does translate to real-life, this is not enough to say that the environment created is a good representation of real life. For the purpose of training the model for operations, the environment was possibly enough, but from a literary perspective there is a lot more to investigate. This can be seen by how the model changed the delay of SWISS with respect to the delay of other airlines. To analyse this, the delay distributions of SWISS flights over time is 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 7, most of the time, for both MILP and PPO, there was no significant difference between the flights. This makes sense and is what 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 MILP 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 16). 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). In order to account for this, further studies into CASA dynamics with respect to TTAs should be investigated, and the environment remodelled.

5 Discussion & Future Work

By analysing the results it was found that modelling a dynamic environment is of great aid to improving Target Time Management. The pairing with a Multi-Agent Reinforcement Learning algorithm can also aid this, due to the ability of such algorithms to account for future states and rewards. It was found that a dynamic slot allocation environment, paired with the successive nature of flights that populate the European ATM network, 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 MILP. 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 22), while MILP considers them as totally independent variables (r=0.11 in Figure 21).

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.

Both SAC and PPO seem to be strong choices for RL algos. 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 differencesarives 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, and is so it is seen as a more operationally dangerous tool. This is because in very rare cases it could make an action very different to the ones it has taken before. Moreover, since for the same inputs it might not do perform same action, 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 deterministic algorithms which are more advanced than PPO, such as Deep Deterministic PPO (DDPPO) or Deep-Deterministic Policy Gradient (DDPG). These might generalize better like SAC does, but still have the deterministic advantages of PPO.

Other ideas for future work imply adding more inputs which unfortunately were not available for 5-year historical data within SWISS. It seems like the tool 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 distance between the current decision, the last decision it can make before it takes off and the take-off time is the estimated taxi time, since this reflects the elapsed time between off-block and take-off.

Besides more inputs, a more complex version of rewards should be implemented. Involving more complex rewards metrics than outlined was deemed beyond the purpose of this paper, since the goal was to evaluate whether RL provides an improvement at all. However, 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.

Another idea for future work would be involving a EUROCONTROL or SkyGuide representative to comment

on the fairness performance of the RL algorithms in practice. This would allow a confirmation of whether or not the fairness was modelled correctly within the environmental reward. The results from testing in a true, live environment show that the delays are more polarized, but this is done symmetrically (both SWISS and other airlines seem to have higher peaks and lower dips). Whether this is fair or not, or whether it is operationally safe and acceptable, are considerations that are beyond the role of SWISS, yet would be very interesting from both an operational perspective and a literary one. In the results, it is obvious that this element struggles the most: while the MILP is known to be unfair for SWISS as explained in Caranti et al. [2023], it seems like it is unfair for other airlines in section 4. This might imply that the formulation of fairness, based on change in average delay over the course of the usage of the tool, is harsh and not representative of reality. In an environment where both SWISS and non-SWISS flights can be modelled, it would be great to have metrics besides the average delay to properly compare the fairness or equity between the airlines. For instance, the THEIL Index developed by Theil [1967] or would have been a valid metric to do so. By also putting a better price index to equity, with the help of EUROCONTROL, the efficiency-equity trade-off would ultimately have higher fidelity.

If the tool was to be deployed in operations, a series of steps have to be taken to ensure a successful implementation. First of all, a higher level of explainability should be attempted - even thought there is no tool to accurately perform feature importance for Reinforcement Learning like there is for Supervised Learning, an attempt at permutation importance could be made. This would allow to see, to some extent, how the tool would perform in the case where its inputs are shuffled across samples. Besides this, a series of edge-cases where the optimization is particularly difficult or has a high risk-reward ratio should be tested. This is because by analysing unseen, edge-cases it is possible to truly find any missing analysis from this article. On top of this, the algorithm would require ordinary monitoring and maintenance. It is advised to observe weekly reports regarding the performance of the algorithm, to find mishaps as fast as possible. Moreover, keeping the MILP active in the background, and only activating it when the MILP's performance exceeds RL's would prove as a stable fallback option. Moreover, retraining either manually every two ro three months, or automatically at the end of every day with the newly generated data proves to be a proactive solution to potential changes to the environment over time.

If the tool was to be implemented in operations in a setting where multiple airlines are considered, it would first be necessary to formulate a more long-term approach to equity. This is because while SWISS has a lot of daily flights to Zurich, other airlines do not, so it would be quite harsh to measure equity or fairness at daily level. Besides this, the architecture of the tool would have to be adapted in one of two ways. The first possible way is to allow each airline to submit data regarding how important for them each flight is with regard to delay, passengers, and other factors and let a centralized method do the decision-making. While this is the most theoretically optimal solution, it is also unrealistic for two reasons: firstly it is necessary to find an unbiased company which airlines are willing to submit their sensitive data to and secondly, airlines may not even know the necessary values on a daily basis. The other option in which something similar could be achieved is by airlines submitting the desired Target Times of Arrival for each flight to an unbiased company, and having a centralized tool which aligns them in a fair way minimizing the difference to the requested TTAs before sending them to EUROCONTROL. While this is more realistic, it is sub-optimal and adds noise in the requests made from the airlines.

Finally, a last recommendation for future work relates to the fidelity of assigning delays. A large assumption was made in this study, which is that delays are normally distributed within a regulation around the mean. In truth, internal studies within SWISS show that they are not, but rather more similar to a β-distribution skewed to the left (many flights with low delays and a few flights with values much higher than the mean delay). Assuming a normal distribution like was done in this study actually makes the environment more harsh than it is in reality, since it assumes that there are as many flights with low delays as ones with high delays. This is also reflected in the MILP performance, which seemed to drop in the simulation environment. Another effect of this assumption is that in reality, the end of the day is less penalized than in the simulation (since there is less demand, delays are typically lower). As such, a different distribution is recommended (like a β-distribution or even an exponential one). An even better alternative to this would be to completely simulate the Demand-Capacity Balancing (DCB) process to more realistically assign delays. This would however require knowledge of other airlines' flights - which pairs very well with the recommendations in the previous paragraphs regarding the inclusion of other flights to better estimate CASA's responses.

6 Conclusions

The goal of this thesis was to answer the following research question: to what extent can Target Tim Management be improved by a decision-making process which takes into account a network-level dynamic environment? The reason behind the need to answer this questions arise from SWISS airlines' struggle with heavy delays due to

arrival regulations in Zurich. In 2019 alone, a total of 4500 delay hours due to these regulations was measured. As such, the tool which is currently used to aid SWISS (based on the concept of Target Time Management) was built. Nevertheless, the tool is not optimal - the environment is highly dynamic and the tool, which uses an MILP formulation, does not account for this.

To answer this research question and to ultimately improve SWISS' operational performance with an updated tool, an environment which resembles a portion of the European Air Traffic network was made. Here, the slot allocation allocation procedure of EUROCONTROL, via Computer Assisted Slot Allocation (CASA) was modelled using statistics. In this environment, just like in Target Time Management, flights are allowed to request for Target Times of Arrival (TTAs), and CASA might decide to accept these or not. The environment was used to train two Multi-Agent Reinforcement Learning decision-making algorithms. Proximal-Policy Optimization and Soft Actor Critic are chosen as the two algorithms. These were then tested on new, untrained data against the baseline MILP approach, and compared using metrics related to improvements in passenger connection, rotation delays, curfew performance and fairness. The same tests were also repeated in a realistic environment, by connecting to EUROCONTROL's CASA on a Pre-Operational server, where actions do not affect real flights, but the environment is the same as is used in real life. Only one RL algorithm could be tested on live data, so due to SAC's non-deterministic nature and the testing performance between PPO and SAC being comparable, PPO was chosen.

The results of the tests are promising. Both PPO and SAC outperform MILP both on the simulated environment and on EUROCONTROL's one. The improvement in passenger connecting time on the simulated environment was highest for SAC, with 5.7 minutes, closely followed by PPO with 5.1, while MILP only showed an improvement of 2.8 minutes per minutes of delay. When it comes to rotation delay, once again PPO and SAC outperformed the MILP (3.4 minutes improvements versus 0.2), and it was found that the main reason behind the large difference was that RL algorithms understood that certain flights were critical for both passenger connecting time and rotation delay reasons. By identifying and prioritizing these flights, they were 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, where PPO outperformed the MILP, but by a smaller margin than in the simulated environment.

The first of the two main behavioural differences between SAC and PPO was found on determinism. While SAC sometimes outputs different actions for the same inputs, PPO is more stable in this regard, and this is seen as an operational advantage for PPO. On the other hand, when looking at the action distributions, PPO tends to sometimes give large anticipation actions for certain flights. Whenever these are accepted by CASA, it performs better. Nevertheless, since they are larger, their chances of being accepted are lower, and when they are not, the performance of PPO decreases. SAC on the other hand tends to have a more balanced approach, more distributed between flights being anticipated and others being slightly delayed.

Overall it was found that Target Time Management can be greatly improved by a dynamic decision-making process, since trade-offs which traditionally take place (for instance between passenger connections and rotation delays) do not have to be made when the decisions are taken by accounting for future rewards. 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 should be undertaken for this model to become operational. The modelled environment did not account for non-SWISS flights, and as such the concept of inter-airline fairness had to be simplified. This topic has to be investigate further from EUROCONTROL side to conclude whether the fairness simplifications implemented result in appropriate behaviour from the algorithm.

There are also further recommendations for future work. 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. For instance, instead of assuming a normal distribution for each flights' delay, one would use a β-distribution, or even perform an instance of demand-capacity balancing using information from other airlines' flights.

References

L. Caranti, M. Carré, and R. Stevens. Optimization of Regulated Airline Arrival Flows via Target Time Management. pages 1–7, Sevilla, Nov. 2023. SESAR. URL https://www.sesarju.eu/sesarinnovationdays.

R. K. Cecen. A stochastic programming model for the aircraft sequencing and scheduling problem considering flight duration uncertainties. *The Aeronautical Journal*, 126(1304):1736–1751, Oct. 2022. ISSN 0001-9240, 2059-6464. doi: 10.1017/aer.2022.17. URL https://doi.org/10.1017/aer.2022.17.

R. K. Cecen and Y. Durmazkeser. Meta-Heuristic Algorithms for Aircraft Sequencing and Scheduling Problem. In T. H. Karakoc, C. O. Colpan, and A. Dalkiran, editors, *Progress in Sustainable Aviation*, Sustainable Aviation, pages 107–118. Springer International Publishing, Cham, 2022. ISBN 978-3-031-12296-5. doi: 10.1007/978-3-031-12296-5_6. URL https://doi.org/10.1007/978-3-031-12296-5_6.

Z. Du, J. Zhang, and B. Kang. A Data-Driven Method for Arrival Sequencing and Scheduling Problem. *Aerospace*, 10(1):62, Jan. 2023. ISSN 2226-4310. doi: 10.3390/aerospace10010062. URL https://www.mdpi.com/2226-4310/10/1/62.

EUROCONTROL. Innovative Slot Allocation: an Overview, Jan. 2014. URL https://www.eurocontrol.int/node/9977.

EUROCONTROL. Arrival Planning Information (API) implementation guide, May 2022. URL https://www.eurocontrol.int/publication/arrival-planning-information-api-implementation-guide.

EUROCONTROL. ATFCM Users Manual, Mar. 2023. URL https://www.eurocontrol.int/publication/atfcm-users-manual.

J. Evler, M. Schultz, and H. Fricke. Flight Prioritization and Turnaround Recovery - Integrating Tactical ATFCM Slot Swapping into Resource-Constrained Ground Operations. Sept. 2021.

F. Furini, M. P. Kidd, C. A. Persiani, and P. Toth. Improved rolling horizon approaches to the aircraft sequencing problem. *Journal of Scheduling*, 18(5):435–447, Oct. 2015. ISSN 1099-1425. doi: 10.1007/s10951-014-0415-8. URL https://doi.org/10.1007/s10951-014-0415-8.

T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In *Proceedings of the 35th International Conference on Machine Learning*, pages 1861–1870. PMLR, July 2018. URL https://proceedings.mlr.press/v80/haarnoja18b.html.

R. Hoogendoorn. Dynamic Airline Centric Inbound Priority Sequencing: A case study on Westerly morning arrivals for KLM at Schiphol. 2022. URL https://repository.tudelft.nl/islandora/object/uuid%3A4c08bbcf-21c9-4074-81bf-b66099b67734.

W. Huang, C. Zhang, J. Wu, X. He, J. Zhang, and C. Lv. Sampling Efficient Deep Reinforcement Learning through Preference-Guided Stochastic Exploration, June 2022. URL http://arxiv.org/abs/2206.09627. arXiv:2206.09627 [cs].

S. Ikli, C. Mancel, M. Mongeau, X. Olive, and E. Rachelson. Coupling Mathematical Optimization and Machine Learning for the Aircraft Landing Problem. June 2020. URL https://hal-enac.archives-ouvertes.fr/hal-02873423.

A. Kalidas, C. Jackson, A. Md, S. Basheer, S. Mohan, and S. Sakri. Deep Reinforcement Learning for Vision-Based Navigation of UAVs in Avoiding Stationary and Mobile Obstacles. *Drones*, 7:245, Apr. 2023. doi: 10.3390/drones7040245.

D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimization, Jan. 2017. URL http://arxiv.org/abs/1412.6980. arXiv:1412.6980 [cs].

E. Liang, R. Liaw, P. Moritz, R. Nishihara, R. Fox, K. Goldberg, J. E. Gonzalez, M. I. Jordan, and I. Stoica. RLlib: Abstractions for Distributed Reinforcement Learning, June 2018. URL http://arxiv.org/abs/1712.09381. arXiv:1712.09381 [cs].

A. Montlaur and L. Delgado. Flight and passenger delay assignment optimization strategies. *Transportation Research Part C: Emerging Technologies*, 81:99–117, Aug. 2017. ISSN 0968-090X. doi: 10.1016/j.trc.2017.05.011. URL https://www.sciencedirect.com/science/article/pii/S0968090X17301420.

A. Muharremoglu. *The aircraft sequencing problem with arrivals and departures*. Thesis, Massachusetts Institute of Technology, 2000. URL https://dspace.mit.edu/handle/1721.1/9156.

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. *Transportation Research Part E: Logistics and Transportation Review*, 106:115–136, Oct. 2017. ISSN 1366-5545. doi: 10.1016/j.tre.2017.08.006. URL https://www.sciencedirect.com/science/article/pii/S1366554517301771.

N. Pilon, L. Guichard, Z. Bazso, G. Murgese, and M. Carré. User-Driven Prioritisation Process (UDPP) from advanced experimental to pre-operational validation environment. *Journal of Air Transport Management*, 97:102124, Oct. 2021. ISSN 0969-6997. doi: 10.1016/j.jairtraman.2021.102124. URL https://www.sciencedirect.com/science/article/pii/S096969972100106X.

H. N. Psaraftis. A Dynamic Programming approach to the Aircraft Sequencing problem. Technical Report, Cambridge, Mass. : Massachusetts Institute of Technology, Flight Transportation Laboratory, [1978], 1978. URL https://dspace.mit.edu/handle/1721.1/67911.

P. Razzaghi, A. Tabrizian, W. Guo, S. Chen, A. Taye, E. Thompson, A. Bregeon, A. Baheri, and P. Wei. A Survey on Reinforcement Learning in Aviation Applications, Nov. 2022. URL http://arxiv.org/abs/2211.02147. arXiv:2211.02147 [cs, eess].

C. G. Schuetz, T. Lorünser, S. Jaburek, K. Schuetz, F. Wohner, R. Karl, and E. Gringinger. A Distributed Architecture for Privacy-Preserving Optimization Using Genetic Algorithms and Multi-party Computation. In M. Sellami, P. Ceravolo, H. A. Reijers, W. Gaaloul, and H. Panetto, editors, *Cooperative Information Systems*, Lecture Notes in Computer Science, pages 168–185, Cham, 2022. Springer International Publishing. ISBN 978-3-031-17834-4. doi: 10.1007/978-3-031-17834-4_10.

J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal Policy Optimization Algorithms, Aug. 2017. URL http://arxiv.org/abs/1707.06347.

R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction, 2nd ed.* Reinforcement learning: An introduction, 2nd ed. The MIT Press, Cambridge, MA, US, 2018. ISBN 978-0-262-03924-6.

H. Theil. *Economics and Information Theory*. North-Holland Publishing Company, 1967. ISBN 978-0-444-10282-9. Google-Books-ID: VVNVAAAAMAAJ.

R. Vervaat. Airline based priority flight sequencing: of aircraft arriving at an airport. 2020. URL https://repository.tudelft.nl/islandora/object/uuid%3A95dd53f0-511e-4060-9271-aa34bf3237a2.

Appendices

A Appendix 1

The MILP model used as a baseline, and developed by SWISS prior to this thesis in Caranti et al. [2023], is outlined as follows. The decisions variables are shown in Equation 1. Then, the constraints are outlined between Equation 3 and Equation 8. The parameters used in these equations are described in Table 8, while a biref explanation for each constraint is shown in Table 9.

$$x_{ij} = \begin{cases} 1, & \text{if flight } i \text{ is assigned to slot } j \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

$$\min z = \sum_{i \in F} \sum_{j \in S} \underbrace{x_{ij} \cdot c_{ij}}_{\text{Criticality}} + \underbrace{x_{ij} \cdot \frac{|i-j|}{M}}_{\text{Diff. to Schedule}} \tag{2}$$

$$\sum_{j \in S} x_{ij} = 1, \qquad \forall\, i \in F \tag{3}$$

Table 8: Parameter Definitions

Parameter	*Definition*
c_{ij}	Criticality of flight i in TTA slot j.
F	Set of flights, indexed with i.
S	Set of TTA slots, indexed with j. Note: $\|F\| = \|S\|$.
M	Chosen large number, $M >> c_{ij}$.
T_i	Scheduled time of arrival of flight i.
t_j	Hours since start of day for slot j.
a	Chosen maximum anticipation limit.
TaT_i	Minimum turnaround time required by flight i.
d	Chosen maximum delay limit.
e_i	Most constraining time (STD and/or ETD) for flight i.
cut	Curfew limit time.

$$\sum_{i \in F} x_{ij} = 1, \qquad \forall\ j \in S \tag{4}$$

$$T_i - \sum_{j \in S} x_{ij} t_j \leq \min\big(a,\ TaT_i\big), \qquad \forall\ i \in F \tag{5}$$

$$-T_i + \sum_{j \in S} x_{ij} t_j \leq d, \qquad \forall\ i \in F \tag{6}$$

$$\sum_{j \in S} x_{ij} t_j \geq e_i, \qquad \forall\ i \in F \tag{7}$$

$$\sum_{j \in S} x_{ij} t_j \leq cut, \qquad \forall\ i \in F \tag{8}$$

Table 9: Constraints Explanation

Constraint	*Explanation*
Equation 3	Each flight must have one TTA slot assigned.
Equation 4	Each TTA slot must be assigned to one flight.
Equation 5	Each flight cannot be anticipated more than the most constraining time between minimum turnaround and a maximum anticipation time.
Equation 6	Each flight cannot be delayed more than the maximum delay time.
Equation 7	Each flight cannot be anticipated more than the most constraining time between STD and ETD if the flight has one.
Equation 8	Each flight cannot be delayed beyond the limit curfew time.

II

Literature Study
(Previously graded under AE4020)

1

Introduction

In this literature review, the state-of-the-art research of a number of airline and airspace -related topics will be investigated. The goal is to find the current assumptions, research gaps and innovative methods which are related, or potentially related, to the topics of flight prioritization for a hub airline, with an important focus on arrival flights.

This literature search complements the thesis project which I, Leonardo Caranti, will undertake with TU Delft and SWISS International Airlines together. This thesis also builds upon a non-published project: my internship. During September 2022 to February 2023, I worked, together with the Operations Research department at SWISS, to implement a Flight Prioritization process, called Target Time Management System (TTMS), very similar to Slot Swapping. For the purpose of this literature search, it will be assumed that the TTMS is an instance of the Slot Swapping problem. The (small) differences between the two will be further analysed later, yet there is so little research available regarding Target Time management that a lot of conclusions will be drawn from Slot Swapping as well as other similar problems.

The purpose of this literature study is to find the literature gap for Target Time Management, when it comes to being more dynamic, future-looking and predictive. It was seen during my internship that there is interest from SWISS side to see whether there is an opportunity to innovate in this research area. Moreover, it was also found that this was one of the limitations of the TTMS tool built during my internship, because of the method chosen (and time available). This method was not able to properly decide when to prioritize future connections over current ones, as well as not being properly able to future delays over current ones (hence improving curfew performance). Because of this, it is now currently optimizing for the next connections and delays only, and not the subsequent ones. As such, these limitations and known research gaps pose a potentially interesting thesis topic in the eyes of SWISS, as well as the Sustainable Air Transport profile of the Aerospace Faculty of the TU Delft.

The first step presented in this literature review is the background on different airline types and why arrival flights are so crucial (chapter 1). Then, the current status of different flight prioritization approaches is outlined in chapter 2, while the methods commonly used in this type (or similar types) of research are investigated in chapter 4. Finally, the research and chosen methods are described in chapter 5, the Research Proposal.

1.1. Airlines Types

Different airline types can be grouped by various parameters, and the most common one is usually their business model. Based on this, they are usually split into two groups: low cost-carriers and network carriers, also known as Full-Service carriers. Even though more types exist such as charter and touristic airlines, as outlined in Gillen (2006), they will not be considered in this section due to their similarity in operational approach to low-cost carriers. Low-cost carriers tend to attract point-to-point demand, by having lower fares

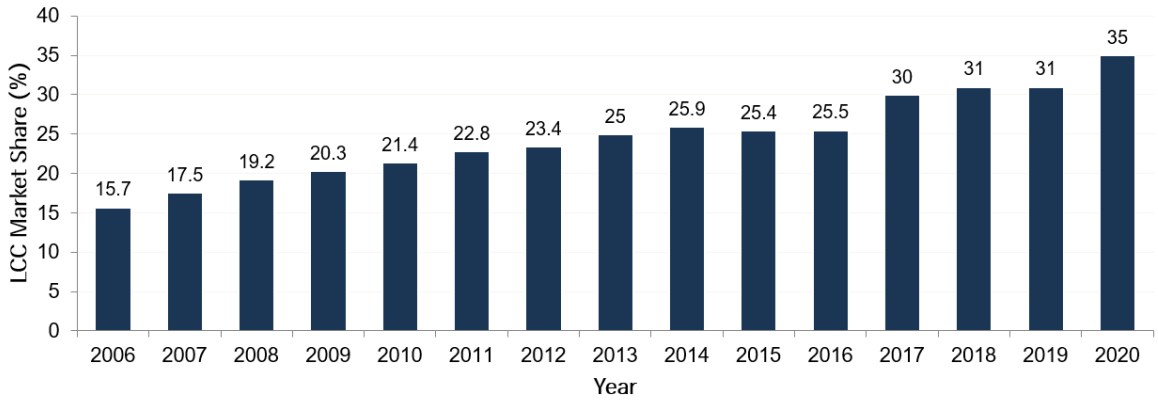

Figure 1.1: Market share of low-cost carriers versus network carriers. (Statista, 2021)

and cutting costs on extra comfort, while network carriers attract demand between two points by bringing the passengers to their hub (Pels, 2008). By doing so, they typically offer more premium services, as well as a larger share of long-haul routes.

As seen in Figure 1.1, the market share of low cost carriers is still lower than network carriers, even though the competition is often fierce, which causes the network and the operational processes of the two to come into contrast. From an operational point of view, low-cost carriers tend to save costs in multiple ways in their turnaround: from having only-bus stands to separate terminals or even having aircraft models with integrated airstairs (Graham, 2013). These approaches are designed to maximize the resources available: low cost carriers are able to minimize their turnaround times, allowing them to have as many flights per day as is physically considered possible. Besides this, low cost carriers also tend to have much fewer connecting passengers and a higher aircraft utilization rate, hence they tend to have shorter turnaround times. On the other hand, as Pels (2008) mentions, the network-carrier's business model is built upon connecting passengers, hence they tend to wait for them longer in case of delays, as well as organize buses and other means of transportation to reduce the connecting times for certain passengers at risk.

Besides presenting a more intriguing problem due to higher complexities, it is in the interest of this thesis to focus on network carriers due to the nature of the company that I am working with. SWISS International Airlines is Switzerland's National Carrier, and is a network carrier with Zurich as its main hub and Geneva as a focus city. Due to their interest in this thesis, low-cost carriers will not be considered.

1.2. Turnaround Process for Network Carriers - Arrival Flights

From the operating regimes and the business models of low-cost and network carriers described in section 1.1, it becomes evident how the turnaround process for a network carrier is significantly more complex and challenging, due to more connecting passengers and higher standards by which the airlines try to adhere to. The sheer amount of efforts to mitigate impacts of disruptions from the turnaround process are numerous, and are mainly generated by prolonged passenger process times, airport capacity or weather constraints, schedule or fleet disruptions (Schmidt, 2017). Moreover, each step in the process which is completed before its scheduled time (shown visually in Figure 1.2) allows to catch-up delay which the aircraft might already have, hence having a great overview of the turnaround processes is crucial.

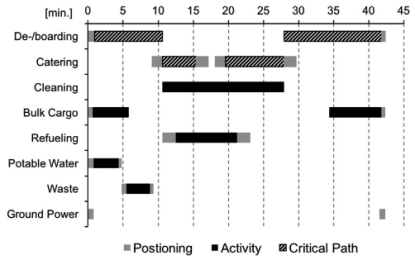

Figure 1.2: Schematic breakdown of a typical turnaround process for an A320. (Schmidt, 2017)

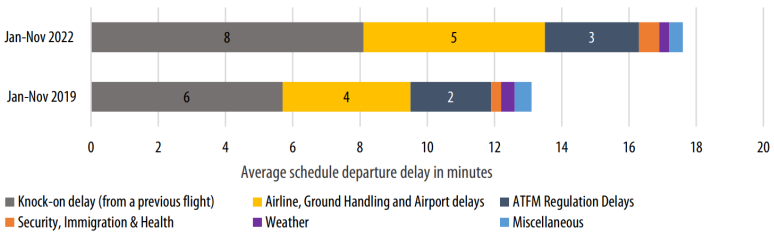

Figure 1.3: Different delay sources for the first three periods of the years 2022 and 2019. (EUROCONTROL, 2022b)

Often time, turnaround processes have a direct impact on an aircraft's delay: either decreasing its current delay, or worsening it. As can be seen in Figure 1.3, the two majour sources of delay, accounting for 77% of total delay in Q1-Q3 2022, were knock-on delay and airline, ground handling and airport delays. (EUROCONTROL, 2022b). These two delay sources can be directly affected (in either way) by the turnaround process, hence the importance of it.

An important item to mention is how the initial stage of the turnaround process, prior to the steps outlined in Figure 1.2, starts with the aircraft landing at a specified time: before then, it is impossible to attempt to catch-up delay, or to ensure that all passengers make their connections. One option would be to let more urgent flights land, and make the others hold or vector while waiting for a landing slot. Even though this is often a solution, it increases costs and emissions due to higher fuel usage. It also increases noise, since the aircraft is likely in the vicinity of the airport. Furthermore, it also increases pilot workload, since the flight time is increased compared to the planned one. Because of these reasons, any attempt possible to improve the status-quo before the flight departs is seen as greatly beneficial when glancing at operations from a network-level perspective. This is exactly why EUROCONTROL (2014) has devised an algorithm (CASA - Computer Assisted Slot Allocation) to assign departure slots which match en-route capacity constraints, which in turn ensures a more fluid arrival process, with often little to no holdings (capacity here defined as the maximum number of flight entries in an hour within a sector which can be safely assigned to a controller). This allows a much better planning for the turnaround of aircraft, since the arrival time is more predictable, as well as significantly decreases the impacts of knock-on delay (2σ statistical significance). (Ivanov et al., 2017)

Besides improving the delay performance, departure slots offer a chance for prioritization. This is why EUROCONTROL has continued to increase efforts in allowing European airlines, airports and ANSP's (Air Navigation Service Providers) to file Target Times of Arrival for each flight of interest. EUROCONTROL (2022a) Then, once the Target Time is received by EUROCONTROL, it is taken into account in CASA, the slot allocation algorithm, which will maximize its efforts to ensure a Target Time of Arrival as close as possible to the requested one. (EUROCONTROL, 2023)

It is the intention of SWISS International Airlines, together with Zurich Airport and SkyGuide, to use this technology to their advantage by sending Target Times of Arrival for each regulated flight incoming into Zurich, maximizing the business benefit of the status-quo before the turnarounds start. However, one must first investigate the current state-of-the-art flight prioritization approaches, methods and solutions that can be undertaken.

2

Flight Prioritization Approaches

There are a series of crucial decision-making steps, prior to touchdown, that an airline or Air Navigation Service Provider can make to affect the turnaround process. These are crucial due to their ability to greatly improve or worsen the status-quo conditions of the network at the start of the turnaround process or inbound wave. The main four processes are shown in Figure 2.1: here, the arrow represents the timeframe when the problem can be solved, and the star represents the moment in which the result of the problem takes effect. Even though all four processes will be described, this chapter will focus on the state-of-the-art techniques related to the Aircraft Sequencing Problem (ASP), Slot Swapping and Target Time Management. This is due to their nature of interest to SWISS, as well as the fact that airlines can have the largest effect on the arrival sequence of aircraft by solving these problems.

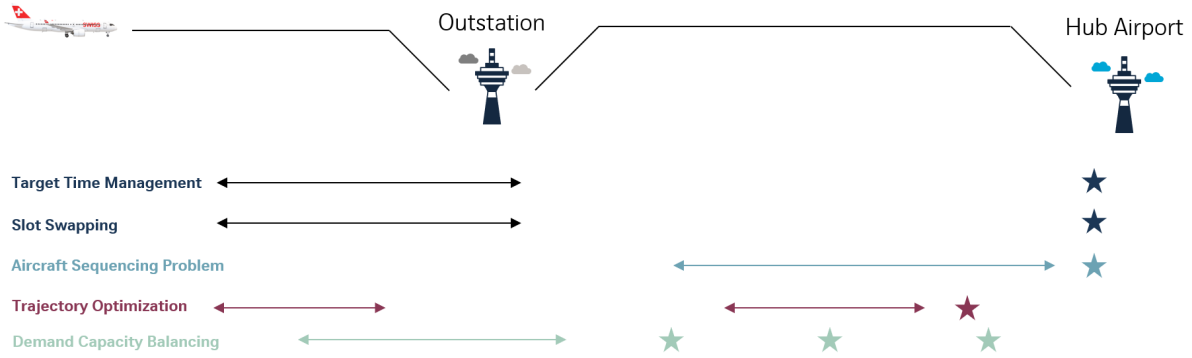

Figure 2.1: Overview of the timeframes of all the problems/decisions which can be made prior to the start of the turnaround process which affect the flight prioritization process. The arrows show the timeframe the problem is solved in, while the star is the time(s) when the result of the decision materializes.

2.1. Arrival Sequencing Problem

One of the most documented and common problems in the ATM (Air Traffic Management) world is the Aircraft Sequencing Problem (ASP), often also referred to Aircraft Scheduling and Sequencing Problem. These are an instance of flight prioritization problems. It entails assigning aircraft on arrival to specific runways based on a set of parameters, such as passenger connections, runway throughput, predicted arrival times, among others. Originally, as early as in Psaraftis (1978), the problem is often rephrased from a travelling salesman formulated as follows: find the optimal sequence of aircraft landings such that all constraints are met while a certain measure of performance is maximized. Often, this performance is measured with metrics such as total delay, total fuel costs, number of missed connections, difference to the original schedule, and others. A visualization of the ASP formulation of the problem can be seen in Figure 2.2, which describes the

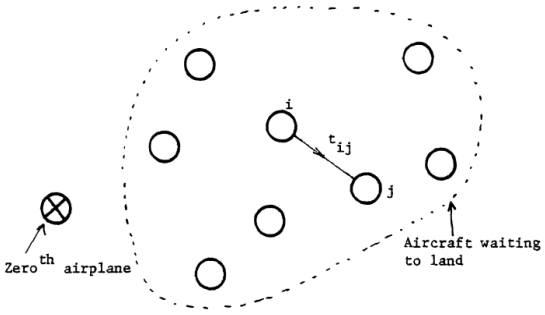

Figure 2.2: An ASP example represented in graph form. (Psaraftis, 1978)

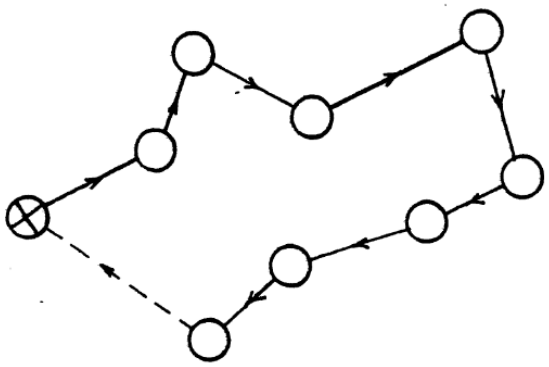

Figure 2.3: An example of a ASP sequence in graph form. (Psaraftis, 1978)

general formulation, and Figure 2.3, which gives an example of what a sequence would look like under this mathematical approach (Psaraftis, 1978).

This problem is characterised by a sequence of states whose transition is deterministic and known. Because of this, Psaraftis (1978) chooses an approach with dynamic programming, with a series of assumptions: the aircraft are all waiting to land, and the air traffic controller can choose which aircraft to prioritize. This approach allowed to optimize passenger waiting times and holdings - and the result was an increased runway throughput. A similar approach some years later, from the same department at MIT, with Muharremoglu (2000) also using dynamic programming to solve the same problem. However, in this case, (weighted) delays were optimized and computational speed was also of primary concern - the dynamic programming approach was augmented with heuristics (greedy policy, time windows). This was then compared to the First Come First Serve (FCFS) policy, usually regarded to as the baseline for the ASP problem, and it was found that delays could be decreased by as much as 40% (Muharremoglu, 2000).

At this point, a valid question arises: if the ASP problem was solved for delays, connecting times and holdings more than 20 years ago, how much more has happened since? Interestingly, due to a number of limitations and assumptions of these older approaches, new research was required.

2.1.1. Mixed-Integer Linear Programming

First of all, with the rise of excellent and open source solvers, a lot of effort has gone into modelling the ASP as a Mixed-Integer Linear Programming (MILP) problem. This is because its ability to provide the best solution under a set of constraints makes it very attractive to a lot of operations-research topics. Likewise, Montlaur and Delgado (2017) and Vervaat (2020) directly use these types of approaches to preform flight prioritization in different scenarios. The former compared different objective functions of the MILP (rotation delay, passenger wait times) and different delay timeframes of the problem (on-ground and en-route, usually regarded to as pre-tactical and tactical in the Air Traffic Management (ATM) world) to try to assign arrival times to the aircraft. On the other hand, in the latter, the model was used to optimize for passenger connections, delay and fuel costs, but this time only at en-route level.

Even though these MILP approaches are highly able to model the environment, it comes to a cost: it has to be simplified greatly for the model to be able to find an optimal solution, and the number of flights in each instance has to be moderated for the computation complexity of this NP-hard problem. The limiting factors for Montlaur and Delgado (2017) were the lack of not having enough data on the airline side regarding delay costs and passenger connections, as well as only focusing on only one day's traffic for one airport (12 September 2014), again due to limited schedule data. Another reason why this approach struggles to scale with data is its NP-hardness: doing a schedule-wide simulation for months and months would be very time intensive using a MILP.

Vervaat (2020) did have more data, for instance passenger connection data, as well as more accurate (yet still very simple and linear) cost modelling. Nevertheless, this is also quite a limiting factor when it comes to using such a tool in operations: the cost data for passenger connection delays was only done by connection type

(business vs economy, long vs short-haul) and the cost of a missed connection was kept at a fixed constant. In reality, a re-booking on a very frequent route will be much cheaper than on a less-frequent route, or a re-booking for a flight in the morning is more likely to also be cheaper than one for the evening, since no hotel costs are incurred. Finally, there was again no schedule-wide simulation in the case of Vervaat (2020), where the model was analysed for only 10 days in January 2019, but this is seen as an improvement compared to Montlaur and Delgado (2017), where only one day was used. Nevertheless, in a highly dynamic and seasonal environment, testing on only such a small set of days may also be a limitation as to why these models might have a low implementation rate.

Another set of notable publications have used MILP approaches to model the ASP problem, with additional elements to make it more suitable to solve the problem at hand while being possibly used in operations. Firstly, Furini et al. (2015) used MILP in combination with a rolling horizon and meta-heuristics to be able to limit the computational intensity of the program. Nevertheless, here the objective function was the difference between the original schedule and the schedule with delays - which may not necessarily be the optimal way forward when it comes to the business needs of an airline. Also Cecen and Durmazkeser (2022) used an MILP approach for the ASP, but this time enhanced by meta-heuristic methods: Genetic Algorithms, Simulated Annealing and Tabu Search. This allowed an improvement in both in the computational speed, but unfortunately the paper does not compare the meta-heuristic methods to using the MILP alone, but rather only to the First-Come First-Served baseline. Overall the delay decrease (which was the main objective of the program) was always between 19.9% and 31.5% better than the FCFS baseline, but no passenger connections (or other objectives) were modelled.

Also Ng et al. (2017) compared an MILP approach boosted with meta-heuristics (Efficient Artificial Bee Colony Optimization), with the purpose of decreasing computational time. For the instance described in this paper, even though they were trying to solve a slightly different problem, the Aircraft Landing Problem (mixed-mode parallel runways, attempting to minimize the difference between the current schedule and the original schedule), it was found that while the MILP failed to find a solution within an hour, and therefore was considered unusable for a set of 18 flights, the Efficient Artificial Bee Colony algorithm was able to do so in around one minute, with optimality gaps always ranging between 0.1% and 8% for most runs.

Finally, also Ikli et al. (2020) started with a MILP approach to the ASP. However, this was then used to train a variety of supervised learning models to see how they would compare to the MILP. The results of the MILP were given to the supervised learning models, and then different supervised learning models were compared: Linear Regression (as a baseline, not really a supervised learning model), a Neural Network and a Support Vector Regression. What was interesting, again, is that these were never compared to the MILP approach but to just the FCFS approach, while it was stated that the computational time was significantly improved, enough to be in line with operations according to the authors (around 5 seconds).

In Table 2.1, a summary of the MILP approaches, timeframes and scenarios can be found. It can be seen that overall, very limited data is used in the development of the models, which typically have very little validation due to this. Unfortunately, the MILP approach, besides not allowing dynamic modelling of the environment, is also bound to limitations in computational power, which is one more reason why often smaller validation sets are used. Moreover, the decision horizon of these approaches is relatively consistent with the ASP formulation: the decisions are taken when the flights are within the area covered by the Extended-Arrival Manager. While this definition varies a lot between airports, it is usually between 1 hour and 2 hours before touchdown (SESAR, 2019b). This only allows for limited re-sequencing of the arrival flights without any holdings, as well as little to no effect of playing with flight speed on the arrival time. The only exception to this action timeframe was by Montlaur and Delgado (2017), which besides the tactical approach also considered a prioritization done at pre-departure level (pre-tactical).

Table 2.1: Summary of the MILP literature approaches for the Aircraft Scheduling Problem.

Publication	*Method*	*Processing time*	*Timeframe*	*Scenario*
Cecen and Durmazkeser (2022)	MILP, heuristics	Medium	Tactical	17 Aug 2019
Ikli et al. (2020)	MILP, superv. learn.	Low	Tactical	2 days, Jul-Apr 2019
Vervaat (2020)	MILP	High	Tactical	10 days, Jan 2019
Ng et al. (2017)	MILP, heuristics	Medium	Tactical	18 flights (no dates)
Montlaur and Delgado (2017)	MILP	High	Pre-tactical, Tactical	12 Sept 2014
Furini et al. (2015)	MILP, rolling horiz.	Medium	Tactical	2 days Aug-Sep 2011

2.1.2. Stochasticity

Approaches like the ones previously described do not unfortunately integrate any type of stochasticity. As Hoogendoorn (2022) explains, this is not an optimal approach since the problem is highly dynamic and non-deterministic: once a new aircraft enters the group of flights to be scheduled, the previous solution is non-optimal anymore. Moreover, as Bennell et al. (2011) suggests, a lot of approaches prior to 2011 only assumed a static environment where no disturbances or non-deterministic behaviour was assumed, and this often results in many of these algorithms not being implemented in operations.

Hoogendoorn (2022) used an (enhanced) MILP approach, where the MILP was used in combination with statistics to allow the model to have more dynamics. By allowing chance-constraints (inspired from Khassiba et al. (2020)) and a rolling time window (inspired by Santos et al. (2017)), the model is capable of adding stochasticity to the solution. This is done via constraints of form shown in Equation 2.1, as outlined by Geletu et al. (2013):

$$P(h(x,\xi) \geq 0) \geq p \tag{2.1}$$

Here, h is the original MILP constraint, x is a static input vector, ξ is a random probability vector and p is a threshold probability that the constraint h is greater than zero. This allows events which are stochastic in nature to be modelled. For instance, in Hoogendoorn (2022), they are used in the modelling of the probabilities of flights aircraft making their Estimated Time Over above specific waypoints. Even though this greatly enhances the ability of this approach to be implemented in real life, after a shadow mode trial it was found that the predictions for the ETO points were not accurate enough for this to be actually implemented in operations.

Another research which tried to approach the ASP in a similar approach, with some stochastic behaviour in the prediction of the Estimated Time of Arrival's (ETAs) was done by Du et al. (2023). Here, it was shown that an alternative approach to having a fully static MILP is to recompute the MILP results every time a certain parameter changes. In turn, this parameter is allowed to be dynamic in nature. In their approach, the chosen parameter was the ETA, which coincidentally was the parameter which when poorly predicted did not allow Hoogendoorn (2022) to deploy the algorithm into the operations scenario. Du et al. (2023) used Random Forest to predict the ETA, and showed that this led to a better result in terms of predicted arrival sequence. Unfortunately, while the paper mentions that this will be used for the ASP problem, it only focuses on the quality of the ETA prediction (MAE reduced from 259 to 79) and the quality of the prediction of the initial status of the sequence, rather than the effect of it on the ASP results. Still, it concludes that by having a better prediction of the final arrival sequence and ETAs it is guaranteed to have better results for the ASP (but no quantification given).

Finally, another interesting stochastic approach, still backed up with MILP methods, is presented by Cecen (2022). Here, the Sample Average Approximation algorithm is used to solve the ASP. This algorithm, ideated by Verweij et al. (2003), uses monte-carlo simulations on various uncertain parameters. For each instance of these parameters, a MILP is solved. Then, the optimality of the solution is found by comparing which solutions appeared most times and how different they are. In the case of Cecen (2022), flight duration was used as a varying parameter, and a MILP was solved with the objective of minimizing the total arrival and departure delay for each aircraft. The objective of the research was however to see if adding a heuristic approach would reduce the computational time required for the MILP. Indeed, it was found that this was successful in reducing computational time without reducing the quality of the final results. Nevertheless, it has also indirectly

proven that the Sample Average Approximation algorithm can be successfully applied to the ASP problem, allowing for more dynamicism.

Overall, it can be seen that all the stochastic approaches to the ASP use it to augment the MILP approach. This is often a limitation - it means that not only the problem has to be simplified to linear form (not taking into account any non-linearities due to fuel, for instance), but also the stoachasticity is limited to a set number of parameters which are deemed dynamic. In all three examples provided, it was only one parameter which was varied (flight duration, ETO or ETA). This approach is even more limiting when evaluating the problem at a network-level, something which no paper has yet attempted.

2.1.3. Optimization Objectives

With the gathered literature, it is now possible to compared the different objectives used in the ASP approaches. Not surprisingly, different objectives are used based on the data available and the target stakeholder. When Air Navigation Service Providers (ANSP's) are the target stakeholder, the difference with the original sequence is usually the objective; when the airport is the target stakeholder, delay is usually the primary target; and then airlines are the primary stakeholder, a combination of delay and passenger connections are the objective. An overview of the different approaches can be found in Table 2.2.

Table 2.2: Overview of the various objectives used in the presented literature in the Aircraft Sequencing Problem.

Publication	*Primary Delay*	*Reactionary Delay*	*Pax Connections*	*Fuel*	*Diff. to Schedule*
Du et al. (2023)					✓
Cecen and Durmazkeser (2022)	✓				
Cecen (2022)	✓				
Hoogendoorn (2022)	✓		✓	✓	
Vervaat (2020)			✓	✓	
Ikli et al. (2020)					✓
Montlaur and Delgado (2017)	✓	✓	✓		
Ng et al. (2017)	✓			✓	
Furini et al. (2015)					✓

What was most interesting was that no crew inputs (crew connections, duty time) or curfew performance was ever considered in these approaches. Moreover, reactionary delay, even though as large of a source of delay as primary delay, was very rarely considered as well. Finally, it must be said that a major limitation was the data regarding the airlines' inputs: often this is business-sensitive data which researchers, such as Montlaur and Delgado (2017), do not have access to. And also in the case where the data was accessed, for instance in the case of Vervaat (2020), the business model was only approximated, and the connection costs were not taken into account individually but rather an average was assumed (making the tool not usable for real operations).

2.2. Slot Swapping

Another problem often tackled in the aviation industry is slot swapping. To really understand what this entails, one must conceptualize what a departure slot is. Like explained in section 1.2, a central authority (Eurocontrol in Europe and the FAA in the US) assigns slots to flights when the demand exceeds the capacity in a certain ATM sector or aerodrome (Schummer and Abizada, 2017). Aircraft must then wait for their departure slot, which in turn spreads out the demand and meets the lowered operational capacity.

An example of where this might happen is at Zurich Airport. Even though it is rated for 40 flights (movements) per hour, if there is wind from the wrong direction, due to its structure as seen in Figure 2.4, the operating concepts have to change. In the worst possible scenario, where a type of wind from East to West called Bise is present, the so-called Bise Concept is implemented. This, compared to the North concepts which allows 40 movements per hours, only allows 28. This is due to the crossover of aircraft taking off on runway 10 (28 in the figure) which cross over potential go-arounds on runway 14. (Bogado Duffner, 2019)

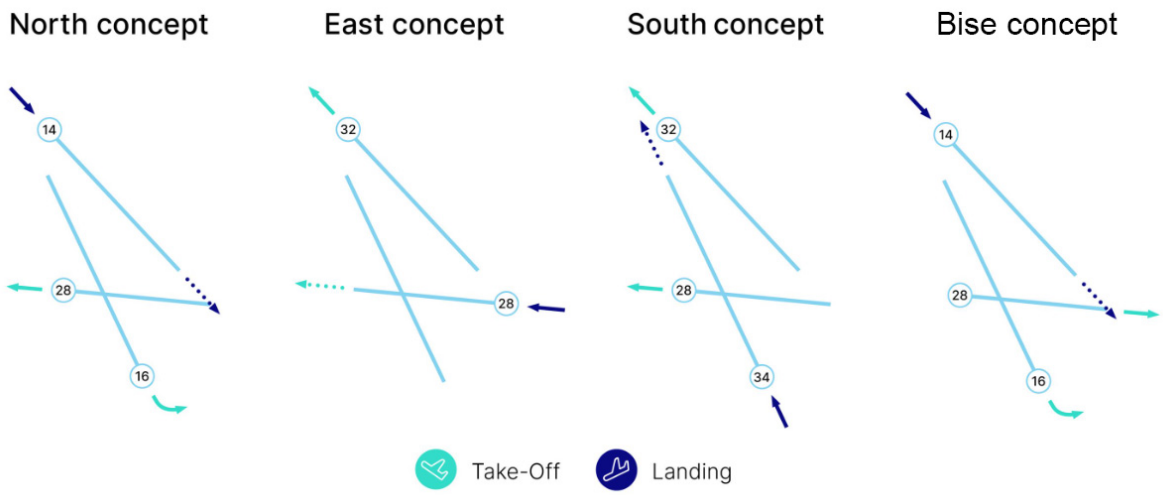

Figure 2.4: Operating concepts at Zurich Airport. The four concepts are all based on wind directions (north, east, south) and Bise (a very strong Swiss wind type) and the numbers reflect the operating runway names and directions. (FlughafenZurich, 2023)

Only due to the Bise concept, which unfortunately happens with a frequency of around once or twice a week (source: controllers in the SWISS International Airlines Operations Control Center), significant delays can be registered. Now, one can imagine that due to a number of other reasons, such as other directional winds, runway closures, bad visibility, or even ATC strikes and maintenance, capacity can be reduced and induce heavy delays. When this happens, EUROCONTROL sets a series of Calculated Take-Off Time's (CTOTs), often denoted as departure slots, which allow the aircraft to arrive with the reduced rate at the airport or airspace sector of concern in theory with no additional holdings needed. (EUROCONTROL, 2023)

The Slot Swapping problem consists of airlines moving around two or more flight's CTOTs to change their arrival order, without affecting the total delay of the regulated traffic. This is done by switching the arrival times of two (or more chosen flights), and then recomputing the CTOT based on the flight time. Like the ASP (section 2.1), it is an instance of flight prioritization, since it allows for a reordering of a sequence based on business objectives. As mentioned in Campanelli et al. (2014), Slot Swapping is regarded as the one of the most effective methods to-date when it comes to decreasing reactionary delay, on-par with reserve aircraft swaps.

Some research, even though less than for the ASP, has been carried out for the Slot Swapping problem, especially at European level. Possibly the most notable and renowned efforts were made by Pilon et al. (2021): in a combined effort between EUROCONTROL, SkyGuide, SWISS International Airlines and other stakeholders within the ATM domain, a so-called User-Driven Prioritization Process (UDPP) was defined. Here, users at the SWISS Operations Control Center were involved in a shadow mode trial of a process which allowed Slot Swapping based on the following set of parameters: delay and passenger missed connections (in terms of a basic cost model). The optimization algorithm (not based on a MILP but rather an exhaustive search, not very scalable algorithm), was able to improve missed connections by 12.5% and overall connections by 40%. Each passenger was weighted by the average business revenue similar passengers would generate (for instance, each business class long haul passenger was weighted the same, and would have the same cost of a missed connection), and the percentages are calculated taking these into account.

What is possibly most interesting about the research of Pilon et al. (2021) is the degree of improvement that was seen in UDPP with respect to the baseline, even though the trial was only in shadow mode, and there were a set of non-trivial limitations:

- Only swaps between pairs of flights were considered.
- Only the current rotation and turnaround were considered, no reactionary delay or subsequent events were modelled (such as curfew performance).

- No network-wide performance was evaluated - this is the largest point of recommendation from the authors' side.
- The problem was solved in a very non-dynamic approach - this required the users to often recompute the results, which would often lead to decreasing stability over time. One more downside of this was that since the algorithm was only trained to work for flights whose most penalizing regulation was Zurich Arrivals, it would struggle to see any effects of flights with multiple regulations.
- It was also assumed that EUROCONTROL was able to guarantee the requested arrival times. In reality, this is sometimes not true, even though the acceptance rate is high. Around 80-90% of the time, the request TTAs were accepted by EUROCONTROL. In the other cases, the flights were still shifted in the direction wished for, so the impacts of this were relatively low.
- The flight duration was assumed as deterministic and known, while in reality it is bound to vary, like seen in the ASP model approaches.

Yet, despite the limitations and the recommendations, UDPP seems to have been very promising on the operations from the shadow mode trial, making it a possibly very interesting point of future research.

Other approaches have since followed from UDPP. Evler et al. (2021) in fact builds upon the research of Pilon et al. (2021), by incorporating ground constraints (mainly the gate allocation problem). To do this in a realistic way, it was also thus necessary to incorporate wishes and schedules from multiple airlines. The chosen use case was Frankfurt Airport, and the schedule used for the analysis was 15 simulated parallel turnarounds. This implies that, unlike UDPP, the business analysis was heaviliy simplified, since no real passenger connections were used, but just simulated assuming average load factors and flights per day.

The major takeaways from Evler et al. (2021) paper are the following. Slot swapping solved with an MILP, including ground constraints (slot allocation) is very efficient as long as the status quo of the downstream network does not vary too much (i.e. subsequent departures are not delayed even more). Nevertheless, it must be noted that as the problem scales with size and as the delays increase (this paper only took 15 turnarounds as a use case), the amount of calculations and data from the airline side is also greatly increased, since a 10-minute delay could cause a lot of missed connections, hence an increased accuracy is required in the slot assignment. In Evler et al. (2021), this type of realistic, large-scale datasets were not present, so it was not possible to do these trade-offs. Lastly, another interesting point which must be mentioned is how there was little regard for the possibility of flights being cancelled or delayed (for instance, due to connections), or aircraft being swapped. The author suggests that this is a major point of improvement for future research.

One of the major troubles of this approach was ensuring equity within different airlines. This was done by keeping the total delay per airline constant. Nevertheless, this then meant that for airlines with very few flights there were very little trading opportunities, limiting the usability of the approach. In the future, it is suggested that using a constant delay per a specific unit of time (for example, 1 month) would allow more slot trading opportunities whilst still ensuring equity.

Two other notable approaches to the Slot Swapping problem must be mentioned. The first one is by Schuetz et al. (2022), as part of the larger European project *SlotMachine* funded by SESAR (2022). Here, the goal of the paper was to evaluate whether it was possible to preserve privacy in a system where airlines could submit their preferences for wished arrival times. In this process, it was also chosen to analyse whether a Genetic Algorithm would be faster in assigning said wished times than a MILP. The main findings were that it was possible to preserve privacy in such a system with Multi-Party Computation, whilst Genetic Algorithms allowed for a 10 to 20 times reduction in computational time with 95% of the optimality of a solution found by a MILP.

However, in the setup of the method by Schuetz et al. (2022), the business-side of the data was not really taken into account to its fullest extent. For instance, the business input, besides being fictitious, was generalised as shown in Figure 2.5: a linear weight was given to the earliest, optimal and latest time for each flight. In reality, the cost for an airline could very well be non-linear, stepwise or completely flat. Moreover, no equity within airlines was insured - this greatly impacted the usability of the algorithm in real world scenarios. Moreover, the authors unfortunately only were able to use fictitious, generated data regarding the flight data, which also

meant that no network-wide approach could have been done. Nevertheless, with this project it can be seen how non-MILP approaches have been tested and do provide advantage.

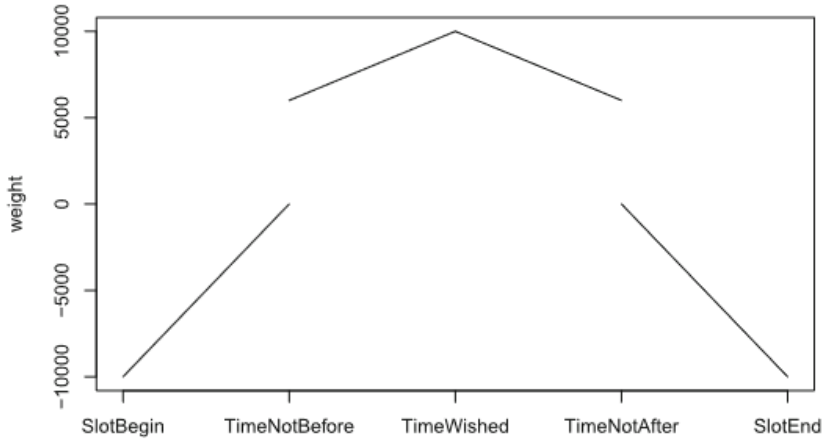

Figure 2.5: Example of the incorporation of airline requests for a specific flight in the weights of the algorithm presented in Schuetz et al. (2022). (Schuetz et al., 2022)

The only research paper found regarding Slot Swapping which did have a network-wide approach was Delgado et al. (2021), also funded by SESAR Joint Undertaking, under the project name Domino. (SESAR, 2019a) What makes this a very interesting paper is the way they modelled each part of the network: taxi times, cruise duration, wind, turnaround time, ATFM delay, connecting times. Each one of these components was given an empirical distribution based on historical data or other publications. Then, the 12th of September 2014 was taken as a sample date (same as in Montlaur and Delgado (2017)) for the simulation, with all of the European flights on that day being part of the data considered. The source of the connecting passenger data, which is typically very business sensitive is not mentioned. Then, an instance of three different problems is solved in the simulation: Slot Swapping, the Aircraft Sequencing Problem, and 4D Trajectory Optimization. This is all done with the use of an agent-based model, where each agent was a single aircraft.

The main takeaways from this approach by Delgado et al. (2021) were very important: 4D trajectory optimization was successful in the reduction of delays and improvement of connecting times, while the ASP and Slot Swapping did not provide any improvements - surprisingly, the ASP actually made the situation worse (popup flights kept appearing and the highly dynamic situation, modelled in a static way, kept changing for the worse). The authors suggest that this is very possibly due to lack of uncertainty modelling. What the authors do not suggest, but may also be likely, is that a rule-based agent based model may not be the best approach to the ASP problem or to Slot Swapping, especially not in a static, rule-based way.

2.3. Target Time Management

Target Time Management is a relatively new concept, developed by EUROCONTROL and explained in EUROCONTROL (2023). This solution approach is essentially a Slot Swap with more flexibility, less constraints, but non-immediate guarantee of success. Like summarized in Table 2.3, there is no time window limit for swaps, while the limit is 20 minutes for Slot Swapping, and a Target Time can be requested at any point after the flight plan is filed, while a slot swap cannot be asked more than two hours before departure. These advantages come with one drawback: while for Slot Swapping the swap is immediate and guaranteed, for Target Time Management it is not - the slot assignment algorithm (CASA) will remember the requested time and try to optimize for it. From internal tests at SWISS, this slot request success rate is estimated to be somewhere around 80-90%, hence proving that this is only a very minor drawback from an operational point of view. This is especially true since the times where the slot is not achieved, the CASA algorithm still manages to move the slot in the desired direction.

Table 2.3: Overview of the key differences between Target Time Management and Slot Swapping. The green entries are seen as an advantage of the method that they are part of when compared to the other method.

Comparison Point	*Target Time Management*	*Slot Swapping*
Swap window limit	No limit	20 min'
Action time	After flight plan filing (3+ hrs before dep.)	<2 hrs before departure
Request success rate	Estimated 80-90%	100%
Literature volume	No literature	15 publications
Fleet type	Short-haul only	Short-haul only

There are currently no literature accounts of a usage of Target Time Management, yet there are two major accounts of the process being used in operations. London Heathrow Airport used this to optimize runway throughput and decrease arrival delays. Paris Charles de Gaulle Airport also used this, more recently, to optimize runway throughput as well as passenger walking distances. Nevertheless, due to its similarity to Slot Swapping, the literature and methods related to Slot Swapping can also be applied to the problem of Target Time Management. For this same reason, only the Open Points in ASP and Slot Swapping will be discussed in section 2.4. It is in SWISS International Airlines' interest to further the research of the Target Time Management topic due to its promising levels of increased flexibility. This will also be further investigated in the HORIZON project by SESAR, where SWISS is participating, due to start in August 2023 and last for two years.

2.4. Open points in ASP and Slot Swapping

As seen in section 2.1 and section 2.2, the Aircraft Sequencing Problem and Slot Swapping have a lot of similarities. This can also be seen from Figure 2.1, where the time in which their decision has an effect (the touchdown time) is the same, but the moment in which the computation takes place is different (before departure instead of before starting the descent or approach). Hence, even though they often have connections and delays as optimization goals, the ASP has fuel as well, but allows for less flexibility. In turn, Slot Swapping has larger margins to play with but also plays with more uncertainty, and it is expected that for this reason, predictions and uncertainty management will have a stronger influence for this problem. Both problems are equally dynamic in nature, due to the changing status of the environment as well as potential new flights appearing in the list.

It then also makes sense that due to the similarities that they share, the ASP and Slot swapping also share a lot of open points of research:

- *What is the real business impact, in quantitative terms, of performing flight prioritization?* This has not been explored a lot due to limited data, especially on operating delays and connections. This is one of the reasons why very few ASP and Slot Swapping instances are currently in operations, despite the significant amount of research.

- *How can network-level information be best taken incorporated in flight prioritization?* The need for network-wide simulations is significant, since the impacts of flight prioritization are indeed network-wide. This again relies on having realistic, sensitive airline data and, in the best scenario possible, a live trial. Again, this is fundamental to actually being able to use flight prioritization instances in real world scenarios.

- *How do predictions and uncertainties impact the flight prioritization process?* From a lot of the analysed research, the conclusion often was that a dynamic, predictive approach with an inclusion of uncertainties was deemed the best for flight prioritization. Nevertheless, very few of these options were analysed at all - partially because of the large amount of data required, and partially because of how cumbersome it may be for the advantages that it might bring: in certain cases, it may be that uncertainties may not be quantifiable at all, or just too large to be useful information for other systems.

- *How is it possible to speed up the computation of an optimal (or near-optimal) flight sequence?* After seeing lots of MILP approaches struggle with growing problem size, it is necessary to mention the interesting efforts to solve flight prioritization problems with Genetic Algorithms (Schuetz et al., 2022), Agent-Based Models (Delgado et al., 2021), Bee Colony Optimization (Ng et al., 2017), Tabu Search and

Simulated Annealing (Cecen and Durmazkeser, 2022). All of these appeared to sacrifice some optimality as compared to MILP approaches to save a lot of computational time. However, no approach has gone beyond MILP, while still attempting to reduce computational speed: reinforcement learning or neural-network based multi-agent methods, for instance, might even be able to take in fewer assumptions than a MILP while being computationally faster.

- *How does a theoretical approach differ from a live trial validation?* Except for UDPP (Pilon et al., 2021), which however was only tested in a shadow mode trial, no other flight prioritization was actually tested in a real-life situation. As shown for UDPP, this led to great insights and truly showed the potential of what these approaches can achieve. Yet, still a lot of testing has to be done in order to fulfill the needs from the airline and other stakeholders' side.

An important point of attention must be devoted to SESAR's HORIZON project. This, not a public project yet, is a part of SESAR 3 Joint Undertaking, and its purpose is to enhance the Air Traffic Flow and Capacity Management (ATFCM) situation with regard to increasing capacity and delays at a European level. One of the projects within HORIZON is TTMS, merely the Target Time Management System. This project is a multi-stakeholder effort to improve the arrival flight management with the use of Target Times of Arrival. The initial use and test case is Zurich Airport, and the three main stakeholders in TTMS are SWISS International Airlines, SkyGuide (the Swiss ANSP) and Zurich Airport.

As an initial step towards TTMS, an interim tool was tested by me during my internship at SWISS. This tool only takes into account SWISS flights which are regulated with Zurich arrival regulations (LSZHA regulation ID). It then rearranges aircraft on arrival (the same as the Slot Swapping mechanism) based on passenger connections and delays. It does this with the use of an MILP, and takes into account only the next rotation. Once the TTAs are computed, they are then sent onto Eurocontrol, which tries to guarantee the arrivals times requested. Currently, the acceptance rate for TTAs is estimated to be 80-90% after a couple of months of trials. For 20 flights, the TTA tool is capable of improving around 25 to 50 passenger connections (weighted by importance), without making delays worse. On the other hand, it can also improve delays without worsening connections significantly. An important trade-off between delay and passenger connections is noticed in the objective function calibration. Overall, the system could be made more static and influence future actions, for instance by taking into account curfew performance as well as crew and future passenger rotations. This thesis will further develop the TTA tool and is hence supported by SWISS.

3

Problem Outline

SWISS International Airlines's major hub is Zurich Airport. Due to the nature of the runway system, as well as the nature of the congested and dense European airspace, it often happens that a large number of flights gets regulated upon arrival or departure. It can be seen from Figure 3.1 that even though delay has decreased since the start of the COVID pandemic due to decreased demand levels, it is very often the case that in Zurich the demand exceeds the capacity. In 2019 alone, around 4500 hours of ATFCM delay were recorded, which is around an average of 12 hours of delay per day. It can also be seen from Figure 3.2 that delays get significantly worse during summer - hence in some days, a total of 12 is is exceeded. This is a huge bottleneck for all stakeholders in Zurich: if the aircraft arrive late, the delayed turnaround process has huge financial consequences which stem off of passengers missing their connections, curfew fees, bad customer service, increased workload for all stakeholders. Hence, an improved arrival flight prioritization process is the key to reducing this impact, being the best solution to when the reduction of total arrival delays is not a viable alternative.

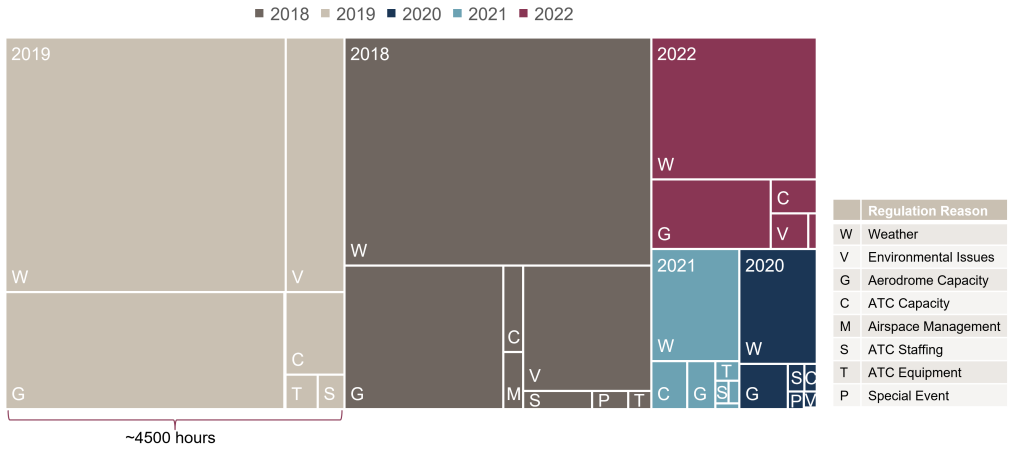

Figure 3.1: ATFCM Delay by year and regulation reason. Data from 2019-2022. (Source: SWISS International Airlines)

After analysing an initial set of literature regarding various types of aircraft prioritization approaches, it is the possible to outline a preliminary problem description. After this, a method will be chosen to solve the problem.

As found from chapter 2, there is significant research regarding the Tactical Phase of Flight Prioritization (the ASP) while much less regarding the pre-tactical phase (Slot Swapping, Target Time Management). The goal of the problem is to address pre-tactical steering of arrival flights (via Target Time Management) with a dynamic system which takes into account uncertainties, as well as possible changes during the tactical phase. As seen, this is a point of struggle for the ASP, Slot Swapping and for Target Time Management. In order to do this, the

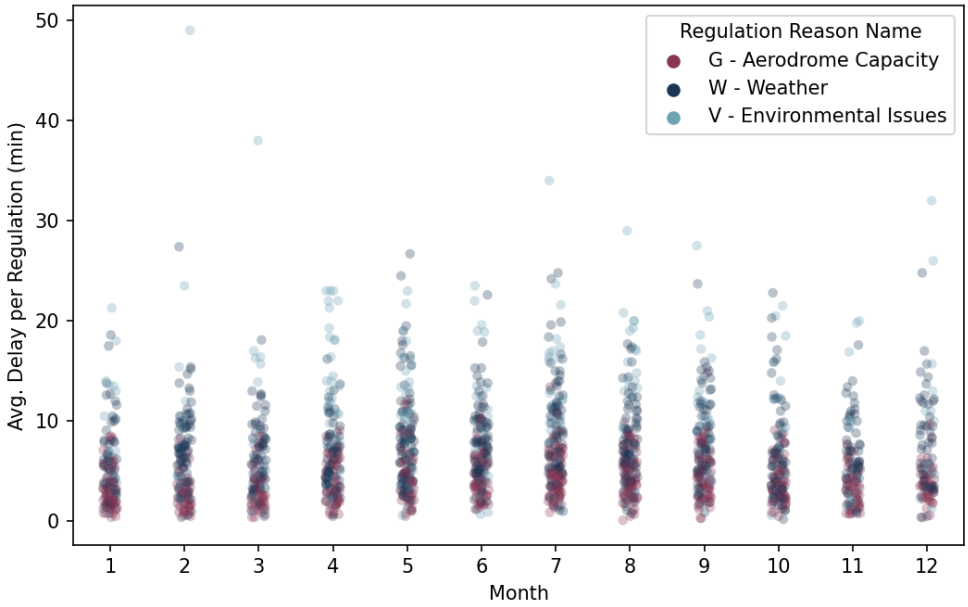

Figure 3.2: ATFCM Delay by month and the top 3 regulation reasons (accounting for 96.6% of the total delay in ZRH). Data from 2019-2022. (Source: SWISS International Airlines)

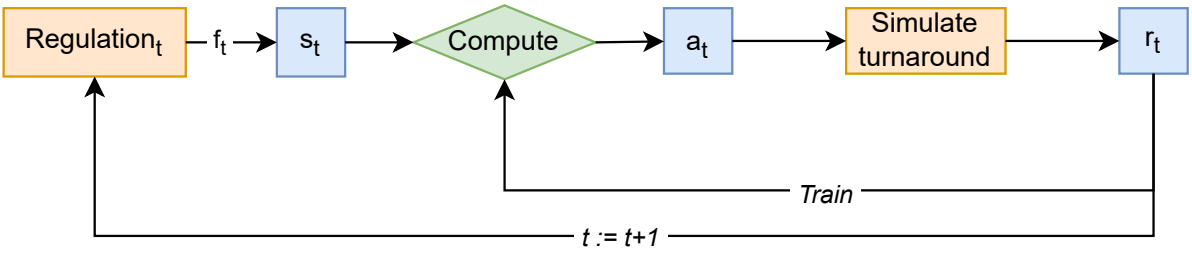

Figure 3.3: Visual depiction of an instance of the problem at a given timestep *t*. Yellow elements are simulation-based, blue are the problem attributes which characterise the current timestep, and green symbolizes a computational decision-making process.

system, besides assigning arrival times to the flights, must also make predictions/uncertainty measurements on one or more of the following parameters:

- Estimated Time of Arrival
- Calculated Take-Off Time
- Which aircraft will wait for which passengers

Besides this, the solution has to optimize, as a minimum, for passenger connections, delays and curfew performance (on a cost basis). Optimally, it should also optimize for minimal holdings (hence minimal difference between the arrival sequence set during the pre-tactical phase and the actual ones), and crew connections. The solution should also optimally undergo a live trial to check its performance with respect to a more static, less forward-looking approach which is currently being implemented at the SWISS International Airlines Operations Control Center: a MILP solution which only takes into account passengers and delay reduction.

The problem can be broken down in various steps. At a given timestep t, the regulation (necessary for having a slot to perform Target Time Management with) is simulated, also shown in Figure 3.3. Then, a set of flights available for swapping is found, denoted as f_t, and data is gathered on them, making up the state space at timestep t: s_t. With this, it is possible to compute an arrival sequence, denoted by action a_t. After this, a turnaround can be simulated, assuming that the sequence remains the same, and then a reward can be

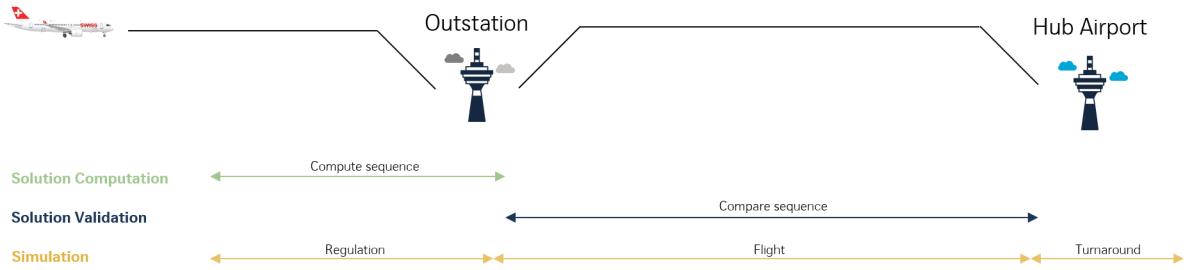

Figure 3.4: Timeline representation of the problem. Yellow elements are simulation-based and green symbolizes a computational decision-making process.

computed (r_t). With this, it is possible to move to the next timestep $t+1$ and repeat the process.

Once the set of aircraft f_t has taken off, it is necessary to see how well the algorithm was able to sequence aircraft. In order to do this, the actual sequence will be compared to the assigned sequence, to see how well aircraft were able to stick to their planned arrival times. This can be seen by the validation phase outlined in Figure 3.4, along with the other problem stages previously outlined in Figure 3.3. Somehow, the algorithm should learn how to assign a sequence which is uncertainty-aware as well as it should be able to learn to maximize its performance at least according to the following given parameters:

- Minimal passenger short connecting times
- Minimal rotation delay
- Minimal curfew fees
- Minimal difference between planned sequence and actual sequence

In order to properly tackle this problem, it is also necessary to understand the uncertainty processes at stake. Besides whether or not a departing flight will wait for its passengers, which is a time-invariant uncertainty but more dependent on the connecting time and passenger quantity/type, the uncertainty regarding the CTOT and the ETA have to somehow be modelled. A fictitious uncertainty graph for the evolution of these two uncertainties can be seen in Figure 3.5, where the best estimate of the values is shown as a line and the uncertainty region (which could for instance be 1 or 2 standard deviations) is shown as a coloured area around the line. It can be seen that the closer we get to the CTOT or the ETA, the more certain we become of the value. It should also be noted how the ETA uncertainty is very probably related to the CTOT itself: once the flight has taken off, there are significantly less variables which can affect the ETA. This is also why the ETA uncertainty region right after the final CTOT is achieved drops significantly in the example shown in Figure 3.5.

The uncertainty in the ETA mostly evolves and changes from a number of events which happen in between, or at least which might cause differences in, the flight phase. The two most notable ones, which are also most documented and explored in literature, are Trajectory-Based Operations (TBO) and Demand-Capacity Balancing (DCB). The first one refers to optimizing the flight plan by changing the trajectory of the aircraft, in any of the 4 dimensions (3 spatial dimensions plus time, by speeding up the aircraft). This may be done in the case of sub-optimal flight plans which circumnavigate an -event (i.e. a storm) which become sub-optimal when such an event disappears (i.e. storm ends or moves). Another example where this may be done is in the case of conflicts which arise, not planned by at flight-plan level (since it is filed very much in advance). Examples of conflict resolution in literature are found in Ribeiro et al. (2022), Cuppen (2022) and Tian et al. (2021), while Pawelek et al. (2017) solves more general trajectory-based problems. Overall, uncertainty is added by the TBO process from the possibility of having to optimize the trajectory in-flight due to fuel inefficiencies or due to other conflicts or constraints (such as weather, other aircraft, among others).

Demand-Capacity Balancing is the other main aspect which affects uncertainty in the ETA. This is a problem which aims to match the capacity of an airspace sector (or aerodrome) and the demand for flying through it at a specific time. Often, due to pop-up flights or aircraft departing with a delay, the original, matched

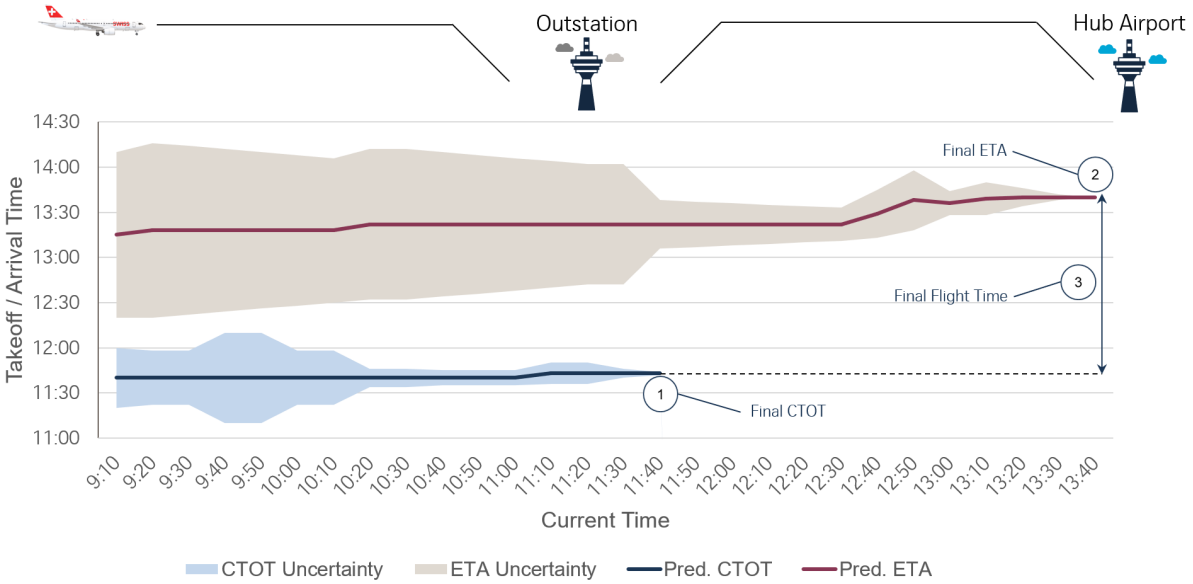

Figure 3.5: Fictitious example of two flights and the uncertainty evolutions across the different flight stages.

Table 3.1: Overview of the data sources and for which steps they would be used.

Data	*Source*	*Live*	*Historic*	*ETA*	*CTOT*	*Pax. Wait.*	*Arr. Sequence*	*Regulation*	*Turnaround*
Schedule	SWISS	✓	✓	✓	✓	✓	✓	✓	✓
Actual Times	SWISS		✓	✓	✓			✓	✓
Regulations Data	ECTL	✓	✓		✓			✓	
Delay Codes	SWISS		✓			✓			
Est. Flight Time	ECTL	✓	✓	✓	✓		✓	✓	
Pax Connex	SWISS	✓	✓			✓	✓		✓

DCB process might not be valid anymore. As such, some aircraft might have to be slowed down, or re-routed in another sector to ensure safety, by not exceeding the capacity of the sector. Examples of efforts in literature where DCB was investigated are Janssen (2019), Huang and Xu (2021) and Tang and Xu (2021). Due to unexpected changes in the timing of aircraft passing through sectors, it is clear that the ETA's uncertainty is affected.

Overall, both TBO and DCB are not planned to be modelled in this problem proposal. This is because the point is to take into account the uncertainties generated by these processes in the pre-tactical phase, rather than optimizing them. As such, patterns will be searched for (sectors which usually might arise the most uncertainty, or flights which often see differences between the filed and flow flight plan), based on a set of input data related to the flight schedule, aerodromes of departure and arrival, among others.

As already hinted at, there are different data sources which appear to be important at the different phases of the problem. Most of the data can be found within SWISS: the Schedule, Actual Times, Delay Codes and Passenger Connection data. Of these, every data type is sensitive, with the exception of the passenger connection data, which is highly sensitive. On top of this, regulation data is also required, to be able to model and validate the regulations in the process - this is not sensitive data, as it is semi-public (has to be requested through the Eurocontrol NM Portal). An overview of what data type and source is used for each part of the program, as well as which data sources are to be used live and which ones only for historical data, is shown in Table 3.1. The table is divided in four sets of columns: the first to show the source of each data type, the second to show which data is historical, the third to show which data type is used for each prediction process within the environment, and the last shows which data is used to compute the action and environment events: the arrival sequence, the regulation severity and the turnaround process (and any potential delays).

4

Methods

Besides the problem outline, a crucial part of the research proposal and literature search is to outline important and possible methods to use. As seen in chapter 2, a wide variety of methods was used to solve the Aircraft Sequencing Problem and the Slot Swapping Problem. Yet, while these methods seem to all have their advantages, they may not be enough for the problem proposed. This statement will be challenged in this section, where each method will be investigated when it comes to its advantages, disadvantages, assumptions, limitations.

4.1. Mixed-Integer Linear Programming

As hinted at in subsection 2.1.1, Mixed-Integer Linear Programming is by far the most common approach to solving the ASP and Slot Swapping Problem. This entails, just like all optimization methods, to propose an objective function to maximize or minimize and a set of constraints to abide to. The program then, using a set of linear algebra rules such as matrix multiplication, the simplex method (Dantzig, 1990) and column generation (Lübbecke, 2011), moves through the feasible solutions until it cannot improve the objective function anymore.

The concept of linear programming, envisioned by Dantzig (1982), allows the guarantee of finding the global maximum, given enough computational time. Nevertheless, to be able to do this, the program must be expressible in a set of strict mathematical rules: it requires the objective function and the constraints to be linear with respect to the variables the program is optimizing for. The main difference between linear programming and mixed-integer linear programming is that in the latter, variables can continuous, integers and binary in nature.

The main advantages of MILP are that it is simple to code, and if no assumptions have to be taken in the mathematical formulation of the problem, it can truly be the best way to find a solution. It is also relatively fast: while Linear Programming (only integer variables) can be solved in polynomial time, Mixed-Integer Linear programming can be solved in exponential time by using the branch-and-bound technique (Land and Doig, 1960). Still, this is much better than many other solutions methods which are either NP-hard or which are not even guaranteed to yield an optimal solution. Finally, one more advantage of MILP approaches is their availability: even for non-coding tasks, MILP solutions are available in a wide variety of commercial and open-source tools, including Microsoft Excel, making it a very available technique.

On the other hand, this simplicity comes at a price. The main drawback is that both the dynamics and the goal of the problem have to be decomposed mathematically into a linear form, which is often difficult or impossible. This is especially true in large and uncertain environments. Moreover, the environment is assumed to be static and, as seen in Hoogendoorn (2022), the modifications necessary to model one small stochastic process (with the use of change constraints) require significant efforts, not to mention a whole stochastic environment. As such, it is often the case that MILP problems are shaped to consider all time instances at once and solve for them in one go (Urbanucci, 2018). This then leads to a further common issue which MILP prob-

lems often face: to avoid simplifications, more and more variables are often added, increasing the cardinality and dimensionality of the problem, and in turn reducing its speed.

While solving the proposed problem in chapter 3 with Mixed-Integer Linear Programming would contribute significantly to research, this method provides a great baseline for comparison. As such, the model developed in my internship at SWISS International Airlines, or something very similar to it, will provide a good threshold for comparison to any other methods chosen.

4.2. Dynamic Programming

Dynamic Programming is a technique first envisioned by Bellman (1954), which entails taking a problem and simplifying it into smaller, sequential sub-problems which can be individually solved to find the optimal solution. When it comes to flight prioritization approaches, dynamic programming has its fair share of literature, with the first dating back to the efforts by Psaraftis (1978) and the MIT research group led by Amedeo R. Odoni. Here, the main problem was to choose a sequence of aircraft landings, and each landing was phrased as a sub-problem: at each step, which aircraft should land first? Once this aspect is figured out, the next sub-problem will see one less aircraft, until the last sub-problem which only has one aircraft left.

Even though this process is described as a sequence of states starting from one starting point to the end, it is often solved backwards, starting from the end and working backwards to the initial solution, with the use of the Bellman Equation. (Bellman, 1954) This yields a great advantage: it is able to optimize for a given cost function at each sub-problem, without having to analyse all the outcome options. Nevertheless, the main advantage of dynamic programming over Mixed-Integer Linear Programming is its ability to handle non-linearities while still performing relatively well: besides the ability to be able to solve NP-hard problems, it can often also reduce their time-complexity. For instance, for the knapsack np-hard problem, dynamic programming can find a solution in pseudo-polynomial time. (Yang et al., 2018)

Another advantage of dynamic programming is that, in contrast to MILP approaches, it can decode sequential processes in stepwise approach (like they are in nature), rather than having to solve them in one go. As such, it is not subject as much as MILP to high-dimensionality issues. Also, it is excellent at modelling complex and known transitions between these sequential states, which in the reality can often be highly non-linear. This makes it typically the best candidate to problems which can be outlined as a sequence of states with complex but deterministic, known transitions.

An important point of dynamic programming is its relationship to the Markov property. It can be said dynamic programming can easily handle instances where each state only depends on the previous state, and not the ones before it. This is typical of Markov processes, and is mathematically expressed as shown in Equation 4.1. This is one of the key assumptions of dynamic programming. Here, X_n is the random variable at sequence step n, while x_n is the outcome that the random variable X_n takes. (Nakaoka, 2018) This is often an advantage - Markov processes occur everywhere - but in reality, for the ASP problem it may not hold: due to the propagative nature of secondary delay, the current actions (new sequence) may affect more than one state ahead.

$$P(X_n = x_n | X_{n-1} = x_{n-1}, ... X_0 = x_0) = P(X_n = x_n | X_{n-1} = x_{n-1}) \tag{4.1}$$

The fact that dynamic programming is suited to handle complex but known transitions is both a blessing and a curse - in many cases, state transitions are not so deterministic, but rather stochastic, or in worse cases, completely random. Moreover, even though it is very fast for smaller instances, or for certain instances where the number of variables in a MILP approach would be too large, dynamic programming still tends to explode in complexity when the number of state transitions is very large. As such, the ability in handling more complex problems yields unfortunately a large time-complexity.

4.3. Supervised Learning

Supervised Learning is a Machine Learning technique which bases itself on predicting a label based on a set of inputs. The usefulness of these algorithms is to be able to infer the relationship between the inputs and

outputs without explicitly outlining its dynamics. The goal is to find a pattern which maximizes the prediction accuracy, often measured with the root mean-square for continuous outputs (regression problems) and the log-likelihood for binary outputs (classification tasks). (Singh et al., 2016)

Supervised Learning works by training on a dataset to be able to make predictions. In this dataset, supervised learning assumes that the target (the solution or output) is known. Hence, to use a supervised learning algorithm, it is of primary necessity to know the "correct answer" of the past. This does not make it a suitable solution for ASP, Slot Swapping or Target Time Management, since often it is the case that the perfect solution is unknown. There are exceptions to this rule, which are those cases in which the model is trained on the results of a MILP to enhance computational speed in the operational environment, like done in Ikli et al. (2020).

On the other hand, the ASP and Slot Swapping problems greatly benefit from better stochasticity and predictions process, like seen in Hoogendoorn (2022) and Du et al. (2023) respectively. As such, as previously outlined, it would be very interesting to include uncertainty estimation and predictions on the following three processes: ETA prediction, CTOT prediction and predicting which aircraft wait for passengers. These are three instances where supervised learning is a possible approach, since the final ETA (continuous, regression task), the final CTOT (continuous, regression task) and whether or not an aircraft waited for the passengers (binary, classification task) can be known with full certainty.

Interestingly, Du et al. (2023) attempted to enhance the ASP with exactly this approach, even though only for the prediction of ETAs. It appeared to enhance the ASP solutions by giving them more stability with respect to the final arrival times. In this paper, the algorithm chosen was Random Forest, originally published by Breiman (2001). Being one of the first modern supervised learning algorithms (excluding linear and logistic regression from the modern algorithms), it proved to be quite powerful and often used also in more recent literature. Often times, it is even chosen over neural network approaches due to its flexibility, ease of use and performance. (Singh et al., 2016)

Yet, since the advent of Random Forest, supervised learning algorithms have evolved. Different classes of algorithms have come to life: besides decision-tree approaches (such as Random Forest), Neural Networks (NN) are very common, support vector machines have a very well-established basis, as well as more traditional regression approaches like linear or logistic regression. Nevertheless, as of the present day, there are three similar algorithms which share the spotlight when it comes to performance, and they are all decision tree-based. (Singh et al., 2016) The oldest one of the three, XGBoost (eXtreme Gradient Boosting) was envisioned by Chen and Guestrin (2016) and is currently supported partly by Microsoft and by Baidu (Chinese search engine). Just like Random Forest, it creates a large set of decision trees to then infer a solution, but instead of taking the mean of the outputs of each tree, it uses a variant of gradient descent (Newton-Raphson method) to optimize for the solution during training. Moreover, it offers a set of other advantages to optimize the learning process: parallel training of trees, and optionally also features such as bagging, regularization, early stopping, and multiple loss functions. (Chen and Guestrin, 2016)

The slightly more recent but equally famous supervised learning technique worth mentioning is LightGBM (Light Gradient Boosting Machine). This method, devised by Ke et al. (2017) is currently maintained mainly by Microsoft and partly also by Google. It has nearly all of the features that also XGBoost has, but with the difference that instead of growing trees level-wise at each decision step, it grows then leaf-wise: instead of trying to creating branches which minimize loss, it creates leafs which minimize it. This appears to draw advantages to it such as memory consumption and efficiency, hence the name "Light".

The last supervised learning algorithm which is worth mentioning is CatBoost, originally published by Prokhorenkova et al. (2019). It was developed and is maintained by the research group at Yandex (Russian search engine). Compared to LightGBM and XGBoost, it provides and edge when it comes to input data, since it can also handle video and audio data, even though non-important for this research proposal. Moreover, it also provides and edge with respect to the other algorithms as it can output the expected variance (uncertainty) of each prediction. The interesting part is that, like explained in Malinin et al. (2021), CatBoost can differentiate between data uncertainty (due to the fact that given similar data, there have been very different outcomes in the training data), and knowledge uncertainty (the model has not seen similar inputs ever before). This last property is the reason why it makes it more interesting for the ETA, CTOT and waiting aircraft predictions

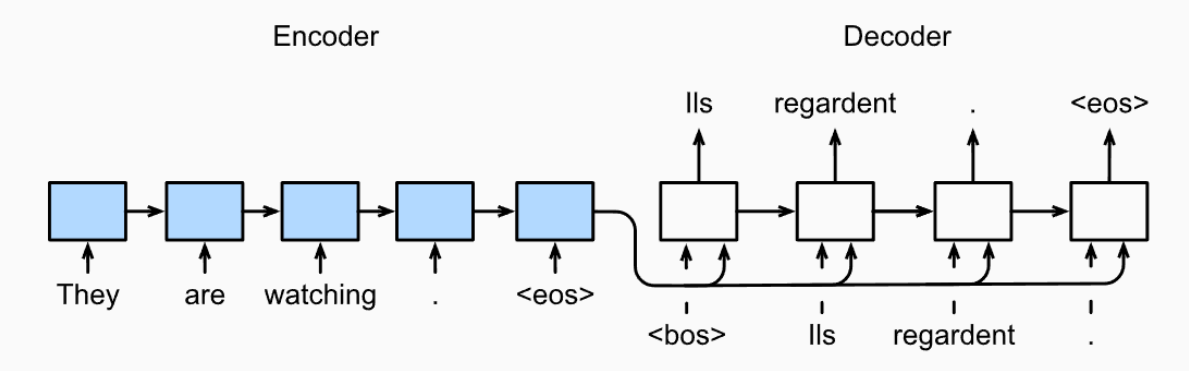

Figure 4.1: Example of a Sequence-to-sequence translation problem. (Learning, 2020)

tasks. These uncertainty values can then be taken into account in the assignment of an arrival sequence, with the knowledge that certain flights are more likely to move around the sequence than others.

4.4. Unsupervised Learning

A brief mention must be made to a specific method of unsupervised learning: transformers. These are an enhanced type of sequence to sequence (Seq2Seq) networks. The concept of Seq2Seq2 networks was first documented by Sutskever et al. (2014), when two neural networks were combined with a Long-Short-Term Memory (LSTM) cell (originally proposed by Hochreiter and Schmidhuber (1997)). These two neural networks (referred to as encoder and decoder) are able to take a variable input size, encode it to a fixed-size state space, and decode it to a variable output size. Thanks to the LSTM cell, the process can be repeated for every input, and the algorithm "remembers" some of the previous inputs and outputs in order to produce even better results. This is a typical architecture used in text-to-text or text-to-image translation, like is shown in the example in Figure 4.1.

The architectural difference between a transformer and a Sequence to Sequence model is its ability to work with longer sequences due to a property called "Attention". This, first outlined by Vaswani et al. (2017), is the modern concept upon which some of the most revolutionary algorithms in the artificial intelligence world have been developed on, including BERT (Devlin et al., 2019) and GPT (Brown et al., 2020). Due to the ability of transformers to handle sequential data very well, Dalmau Codina and Herrema (2019) decided to use such an architecture to model the sequence of arrivals at an airport. Besides the problem that they were solving, which is substantially different to Slow Swapping, the concept of modelling arrivals as a sequence and feeding them into an encoder-decoder architecture is an excellent one, due to its flexibility and easy of adaptation to various sized-scenarios. Nevertheless, even though it can handle very complex relationships, this approach may also struggle to be dynamic, since it also attempts to find a one-shot solution to a sequencing problem.

4.5. Reinforcement Learning

Reinforcement Learning is a large class of machine learning algorithms which are also based on the Markov Decision Process assumption. However, besides this, there are no other assumptions which they take. The main reason why this set of algorithms has become so popular in recent years is that it is able to find a strategy, rather than a solution, to thrive in a dynamic environment. Moreover, the environment dynamics do not have to be given to the model: with enough information about the current state, reinforcement learning techniques provide winning strategies which maximize future rewards by making numerous actions in a given environment.

Based on the above premises, reinforcement learning is particularly interesting for the Target Time Management problem for three reasons. Firstly, it is capable of handling uncertainties well - the time-sequential nature of a reinforcement learning environment, where actions succeed one another, fits particularly well with the concept of decreasing uncertainties when the final landing time approaches. Moreover, and this is possibly the most important of the three reasons, reinforcement learning provides a strategic solution to a dynamic environment, which is one of the main causes of operational implementation failures of many MILP or

Dynamic Programming solutions to the Aircraft Sequencing Problem or Slot Swapping. Finally, this approach of maximizing future rewards should be better at minimizing propagated delays and curfew fees (since they both occur in the future as a result of the current action). Besides these three reasons, reinforcement learning is an attractive solution for operational purposes also when it comes to its ability to be very fast once trained, and due to the lack of assumptions it takes when modelling an environment.

When used in common jargon, Reinforcement Learning typically refers to model-free, control problems. Model-free entails that the algorithm is given no information about the environment dynamics: in order to have a feeling of what they are like, it must interact with it by making an action. This is different in the case of model-based problems, like Dynamic Programming, where the environment dynamics are embedded into the system. (Wang et al., 2018) On the other hand, control problems differ from prediction problems by, like previously hinted, finding a strategy to maximize future rewards rather than finding a solution. Due to these features, Target Time Management can be phrased as an instance of a model-free, control problem.

A very important notion when comparing Reinforcement Learning algorithms is the policy. This, formally defined as in Equation 4.2, is the strategy followed to pick an action. Here, a represents the action, s represents the state and π the policy function. There can be both deterministic policies as well stochastic policies (the one outlined in Equation 4.2 is an example of a stochastic policy). In reinforcement learning the policy is often approximated with a neural network, but any function which takes as input the current state is by definition sufficient. Another important notion in reinforcement learning is the state-value function: this is the expected total reward given by following a certain policy. This is formally expressed as in Equation 4.4, where V^{π} is the state-value function, R is the total reward, r_t is the reward at step t and γ is the discount rate (a zero-one value which signifies how important future rewards are with respect to the current one). Finally, instead of basing the value only on the state, it is also possible to make it a function of also the action: the state-action-value function. The formal definition of this can be seen in Equation 4.3, where Q^{π} is the state-action-value function. Generally, using the state-action-value is seen as a more modern and more powerful approach when compared to the state-value function. (Sutton and Barto, 2018)

$$\pi(a,s) = P(a_t = a | s_t = s) \tag{4.2}$$

$$Q^{\pi}(s,a) = E[R|s,a,\pi] \tag{4.3}$$

$$V^{\pi}(s) = E[R|s_0 = s] = E\left[\sum_{t=0}^{\infty} \gamma^t r_t | s_0 = s\right] \tag{4.4}$$

Reinforcement learning algorithms can have different approaches with regard to the policy, the state-value function and the state-action-value function. The two main groups of algorithms are Policy-based or Value-based. The goal of both algorithms is to find the optimal policy, since this ensures the best strategy for the problem at hand. Obviously, it is possible to try out different policies and use optimization techniques to refine the policy which maximizes the rewards: this is the basis of policy-based algorithms. On the other hand, it is also possible to optimize the state-action-value function to reflect the environment in the best way possible, to choose a policy which maximizes the value-function: these are value-based algorithms. In deep reinforcement learning, these functions are all approximated by deep neural networks. Some examples of the most documented sub-classes of algorithms for policy-based methods are the Policy Gradient (Sutton et al., 1999) and Actor-Critic (Sutton, 1988), and the equivalent for the state-value algorithm class is Deep Q Networks (Mnih et al., 2013). A last mention to the different approaches to reinforcement learning goes to off- versus on-policy algorithms. Here, on-policy refers to the fact that the algorithm directly changes the policy to optimize for rewards, which allows it to find better optimums as compared to off-policy algorithms, which use two policies (one for exploration, one for exploitation), trying to find a better balance between space exploration and good optimums.

Reinforcement learning is one of the fastest developing fields in the computer science industry, and as such, the state-of-the-art algorithms are always changing. For the purpose of this literature study, three algorithms have been chosen to represent the state-of-the-art. (Henderson et al., 2019) It is important to note that the algorithms which are built to deal with image data and language processing have been discarded due to the different nature of the problem at hand. The first chosen algorithm, proposed by Schulman et al. (2017), is Proximal Policy Optimization (PPO). This is a policy-based, on-policy algorithm which can handle both continuous and discrete action space. PPO uses a stochastic policy, similarly outlined as in Equation 4.2.

Table 4.1: Overview of the most prominent Reinforcement Learning algorithms and their features.

	Policy/Value Basis		*Action space*		*Policy Type*	
Algorithm	Policy	Value	Continuous	Discrete	Off-policy	Stochastic
PPO	✓		✓	✓		✓
DDPG	✓	✓	✓		✓	
SAC	✓	✓	✓	✓	✓	✓

Moreover, PPO has the advantage of sometimes being combined with LSTM (Long-Short Term Memory) cells to enhance the "memory" ability of the algorithm, as shown in August and Hernández-Lobato (2018).

The second algorithm worth mentioning, first presented by Lillicrap et al. (2019), is Deep Deterministic Policy Gradient (DDPG). This is a hybrid algorithm, since it uses two networks, an actor and a critic, and the actor is policy-based, while the critic is value-based. Moreover, in contrast to PPO, this is an off-policy algorithm, and uses a deterministic policy instead of a stochastic one. However, is not capable of handling discrete action spaces, as it can only model continuous ones. As found by Henderson et al. (2019), DDPG tends to perform better (and train faster) than PPO in higher dimensions for continuous control tasks.

Finally, the last reinforcement learning algorithm presented in this literature review is Soft-Actor Critic (SAC), originally proposed by Haarnoja et al. (2018). Just like DDPG, this is also a hybrid (partly policy-based, partly value-based) algorithm which uses an actor-critic framework. However, SAC can also handle discrete action space as well as continuous ones. Just like DDPG, it is an off-policy algorithm, but it uses a stochastic policy instead of a deterministic one. The combination of these features allows it to reportedly have a better performance than DDPG when it comes to finding the right balance between exploration and exploitation, like shown in Saaybi et al. (2022). In general, both DDPG and SAC are significantly more sample-efficient than PPO (due to being off-policy and being able to explore the whole space faster), to the price of being more difficult to tune. A comparison of the features of each algorithm can be seen in Table 4.1.

With regard to flight prioritization approaches using reinforcement learning, there are no papers found which tackle either the Aircraft Sequencing Problem, Slot Swapping or Target Time Management. However, there is some usage in similar topics, where the is still the concept of a sequence of aircraft, but the goals, dynamics and constraints are too different from Target Time Management to really consider them in the same research class. As Zhang et al. (2021) suggests, a possible use case for cooperative reinforcement learning is Air Traffic Management in Unmanned Aerial Vehicle scenarios. An example is for instance by Ribeiro et al. (2022), which combined Soft-Actor in a multi-agent system to improve ATM conflict resolution methods. This entailed modelling each flight as an agent, in an environment with a decentralized decision-making process, which would return the same cumulative reward to all agents. A visual depiction of a centralized versus a decentralized system can be seen in Figure 4.2 and Figure 4.3 respectively. Multi-agent reinforcement learning systems are highly scalable - in Ribeiro et al. (2022), nearly 600 instantaneous aircraft were used for the training process, and nearly 900 for testing. In the case of Target Time Management, these are not the numbers of interest, but it could very well be that in a mid-day wave in Zurich, the regulation affects up to 40 aircraft within the timespan of one hour.

Another notable effort with regard to Reinforcement Learning in the ATM domain, even though still far from Target Time Management, is by Vonk (2019). Here, the authors explored different separation and sequencing methods by augmenting them with DDPG and Deep Q-Networks (state-action-value, off-policy method) in a multi-agent scenario. In this case however, no optimal policy was found, and the reason for this is attributed to sub-optimal reward function and exploration strategy. This highlights the importance of the latter two elements when building reinforcement learning systems. Besides Vonk (2019) and Ribeiro et al. (2022), there are other papers dealing with multi-agent reinforcement learning systems for similar ATM topics, but no approach was found within similar ATM topic, Slot Swapping, Target Time Management or Aircraft Sequencing which used more centralized or single-agent reinforcement learning methods.

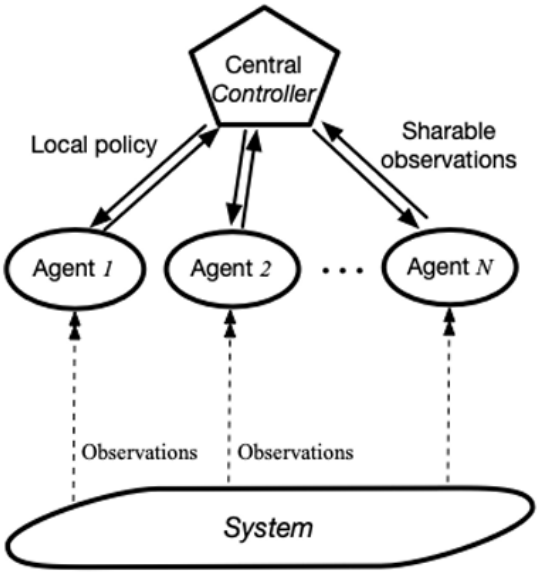

Figure 4.2: An example of a centralized multi-agent system. (Zhang et al., 2021)

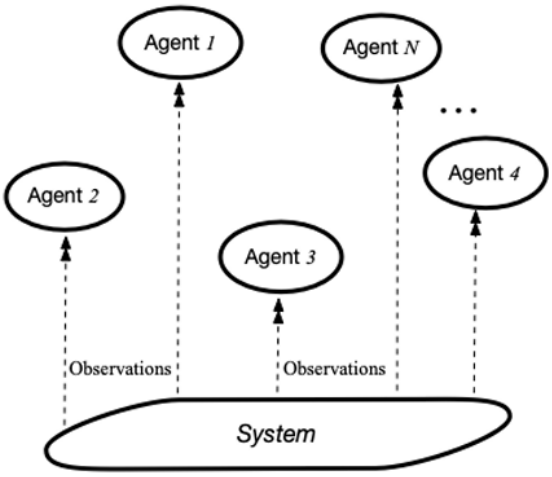

Figure 4.3: An example of a decentralized multi-agent system. (Zhang et al., 2021)

4.5.1. Reinforcement Learning for Sequences

When it comes to state-of-the-art techniques which could be interesting for Target Time Management, Reinforcement Learning frameworks which are modelled for sequential data are the most interesting. In essence these combine the unsupervised learning methods for sequences outlined in section 4.4 with the reinforcement learning frameworks provided in section 4.5. This idea was initially proposed by Keneshloo et al. (2019), which analysed various different reinforcement learning frameworks and connected them to a basic Seq2seq model. This proved that the frameworks were capable of exceeding normal RL methods for tasks which were sequential in nature (mainly specific chosen Gym tasks).

Later, both Giuliari et al. (2020) and Chen et al. (2021) used this approach to connect RL frameworks to transformers instead of Seq2seq models (encoder-decoder structures with attention, see section 4.4 for more details). The former was able to prove that this structure outperformed traditional RL algorithms with regard to two known issues in RL: the credit assignment and the sparse rewards problem. On the other hand, Chen et al. (2021) was able to prove the superiority of this framework with regard to the incorporation of long-horizon.

Besides these three papers, very little research has been done to further the development of this emerging technology. One last yet important publication which approached RL for sequences is by Taghian et al. (2021), which is currently the only publication (to the authors' knowledge) which attempts to use RL for sequences in a non-standardised environment such as a Gym instance. Here, the authors modelled an encoder-decoder structure together with the decoder being tested as various RL algorithms (Deep, Q-Network, Multi-Layer Perceptron, among others). It was found that for volatile markets, this structure even exceeded the time-series methods which are common in most econometrics-based approaches.

Reinforcement Learning for sequences, even though never used before in the aviation industry, poses as a great method for decision-making for the slot-swapping problem. Nevertheless, due to its early stages in the development, it may require significantly more training time and effort to produce a valid result. With enough time, it would be interesting (in literary terms) to compare the performance of a normal reinforcement learning algorithm versus a RL instance for sequences for the Target Time Management problem.

4.6. Methods for Target Time Management

While Slot Swapping (hence Target Time Management) was traditionally approached with MILP methods, after this chapter it can be seen that different techniques could yield interesting prospects for the steps in the Target Time Management or Slot Swapping process. Mixed-Integer Linear Programming would provide

Table 4.2: Overview of the most interesting algorithms for Target Time Management.

Algorithm	*Type*	*Task*	*Notes*
MILP	Linear Optim.	Baseline approach	Static, non-forward looking, stable
CatBoost	Supervised L.	Predictions, uncertainties	Predictions within environment model
NN + LSTM	Supervised L.	Predictions	Predictions with variable-sized inputs
Transformers	Unsupervised L.	Aircraft Sequence	Good at modelling sequences
PPO with LSTM	Reinforcement L.	Aircraft Sequence	Good at remembering past actions
SAC	Reinforcement L.	Aircraft Sequence	Excellent for dynamic environment
Multi-Agent SAC	Reinforcement L.	Aircraft Sequence	For many flights, more difficult to train
Transformer + DQN	RL for Sequences	Aircraft Sequence	State-of-the-art, most difficult to train

a good baseline for comparison for any other more refined, dynamic and predictive techniques. Supervised Learning provides an excellent framework for the inclusion of predictions regarding the Estimated Time of Arrival, Calculated Take-Off Time, and flights which wait for passengers, as well as the uncertainties associated with each prediction. Unsupervised learning, in the form of encoder-decoder or transformer architectures, would provide an excellent framework for mathematically modelling the arrival sequence in a variable input-output size. Finally, reinforcement learning provides an excellent approach for the logic and choices within the flight prioritization process due to its ability to handle complex and dynamic environments. A more in-depth description of the most interesting algorithms for Target Time Management are outlined in Table 4.2.

With this, an order of the utilization of the methods can be proposed. This, visualized in Figure 4.4, includes two supervised learning methods (CatBoost and a Neural Network paired with a LSTM cell), two Reinforcement Learning methods (one more traditional - either SAC or PPO, and one for sequences specifically) and one MILP method. The reason behind having two RL methods is such that one could choose to investigate a more complex yet state-of-the-art one, or opt for a simpler and more documented model. The MILP on the other hand, would not be used for decision-making but only for validation. Finally, the CatBoost model would be used to estimate both the ETA and the uncertainties related to it, while the Neural Network combined with the LSTM would allow for a better handling of non-fixed size input.

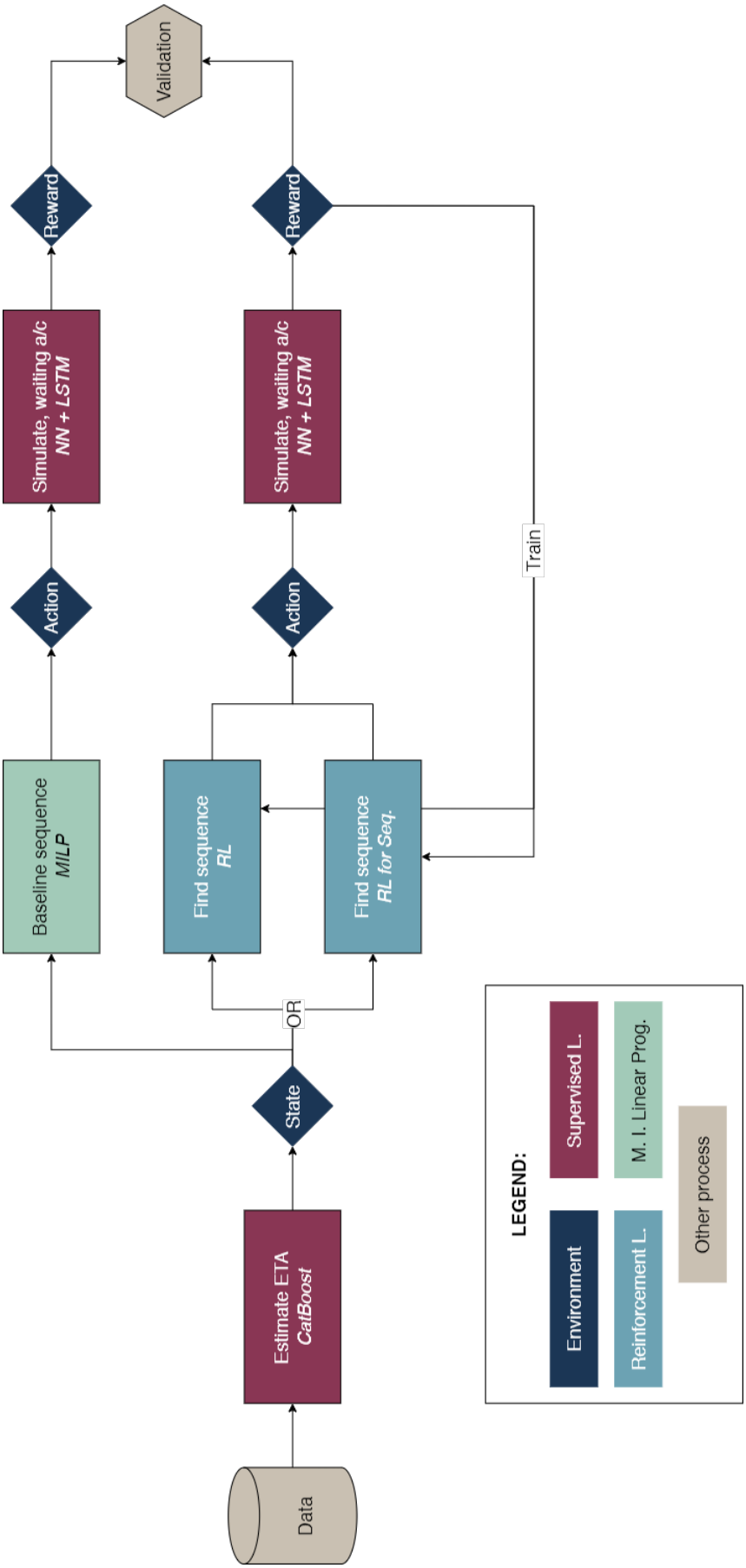

Figure 4.4: Flow diagram of how the methods and processes could be combined to solve the Target Time Management problem.

5

Research Proposal

Based on chapter 2, chapter 3 and chapter 4, a great deal of unanswered questions from literature and potential methods have been raised and discussed. Especially in section 2.4, the true gaps in the literature have been presented. Summarising these, the lacking contents were found to be the inclusion of uncertainties and predictions, network-level approaches, quantification of business impact, while possibly doing a trial and keeping computational times within an operational timeframe. This leads to the following research question:

***RQ1**: To what extent can Target Time Management be improved by a decision-making process which takes into account network-level environment dynamism, uncertainties and predictions?*

This can be broken down in various sub-questions to simplify the process of answering such a broad research question:

- **RQ1.1** *How much does adding uncertainties and predictions (regarding the Estimated Time of Arrivals and waiting aircraft for connecting passengers) into the Target Time Management process improve it?*
  This will be answered in a two-step manner, by first including the predictions of estimated time of arrivals of aircraft, and then by including the predictions on which aircraft will wait for connecting passengers. It is expected that the former will improve the algorithm's performance, while the latter will bring the modelled environment much closer to reality (hence easier to validate).

  - **RQ1.1.1** *To what extent is it possible to predict the estimated time of arrival as well as the data and knowledge uncertainty related to it?*
    In order to answer this sub-sub-question, data regarding the scheduled and actual flight times will be organized, and then fed into a CatBoost, supervised learning algorithm to be trained. Once trained, the outputs of this algorithm (the estimated time of arrival and uncertainty related to it) will be then used as inputs to the process in sub-question RQ1.3.

  - **RQ1.1.2** *To what extent is it possible to predict which aircraft are going to wait for which passenger connections?*
    To answer this, data regarding delay codes would have to be combined with passenger connection data, and then fed into a Neural Network to be trained. At a second stage, the inclusion of an LSTM cell within this network will be evaluated. Once the network is trained and validated with unseen data, its outputs will be connected with the modelling of the environment, part of RQ1.2.

- **RQ1.2** *To what extent can a dynamic approach minimize the difference between planned sequence and flown sequence, as well as improve passenger connections and minimize rotation delays?*
  To answer this question, the decision-making environment has to be modelled first. Here, the regulation delays data is retrieved and a statistical distribution is fitted to the average delay per flight. Then, for every day where schedule data is available, the regulation data is fitted to simulate a ficti-

tious regulation on real schedule data. Together with the passenger connections, this constitutes the state, The action and reward will be modelled next. To model the reward, the turnaround process has to be simulated (using the predictions of which aircraft will wait as well as the estimated time of arrival from RQ1.1.1 and RQ1.1.2). Then, a Reinforcement Learning algorithm will be used as the dynamic, decision-making tool. This will have to be coded, trained and validated with the use of the simulation environment.

- **RQ1.3** *To what extent can a reinforcement learning algorithm improve Target Time Management compared to a more static approach using a Mixed-Integer Linear Program?*
  This first validation question will be set up by first connecting the MILP approach devised during my internship to the environment created in RQ1.2. Then, the inputs will be taken and fed into the MILP algorithm, whose answers will constitute the actions in the environment. Once this is done, the KPIs used to evaluate and train the Reinforcement Learning algorithm will then be used to compare it with the MILP approach.

- **RQ1.4** *To what extent does the difference in performance between a MILP, static approach and a Reinforcement Learning approach for Target Time Management change between a simulated environment and a real-life trial?*
  To fully understand the potential of this technology, the reinforcement learning approach could be connected to the already existing frontend of the MILP tool built during my internship. If the algorithm appears to work better than the MILP approach, it would be great to test it in live action during the course of a regulation. Besides technical considerations, this would require some proof, in the form of a presentation to the Network Operations Control users, that this algorithm would probably deliver better results if deployed in operations. This live trial testing would be the perfect way to validate the results and analyses done with only post operational data.

In the investigation of Target Time Management approaches, it should be possible to include more information about the uncertainty and future states (via predictions) to further enhance the dynamic approach - justifying sub-question *RQ1.1*. *RQ1.1.1* and *RQ1.1.2* directly propose the parameters which are planned to be predicted (the ETA and the waiting aircraft) as well as the uncertainties which will be considered (data and knowledge uncertainty for the ETA prediction). Moreover, *RQ1.2* attempts to bridge the gap between the planned and the flown sequence of arrivals with the use of the dynamic approach, which should be able to predict this better. Sub-question *RQ1.3* aims at building a dynamic, reinforcement learning approach to be compared to the baseline, static MILP approach which was built during my internship at SWISS. Due to the dynamic modelling of the environment, it is expected that the latter could struggle more in terms of long-run and total rewards. Finally, *RQ1.4* relates to the opportunity to do a testing trial to compare the dynamic strategy to the static one in a more realistic environment. Even though this may not be the most crucial part of the research, it would provide immensely important validation as to whether or not the environment was modelled appropriately or not.

An important topic which was often mentioned in the literature study but not in this chapter is the prediction of the Calculated Take-Off Time (CTOT). This is because, from a literature perspective, it is a very interesting problem: it is often the case that ATFCM delays are time-variant and unstable, and having a better prediction of them would be of great value for multiple stakeholders in the aviation domain. An important research paper in this domain was presented by Dalmau-Codina et al. (2021), where the predicted CTOTs based on different rates are calculated. It could be that, for instance, the weather that was causing an arrival regulation in Zurich gets worse, and the arrival rate is reduced from 28 to 25 movements per hour - this increases the average delay for each aircraft hence worsening all CTOTs. While this can be taken into account by the research in Dalmau-Codina et al. (2021), there are a lot more factors which have to be taken into account: for instance, if another airline files their flight plans very late, this might shift the already planned sequence of arrivals. Similarly, if an aircraft has an unplanned delay and cannot take off on time, hence they receive a new CTOT, this would also shift the whole sequence. Since predicting the weather and the behaviour of other airlines is seen as beyond the scope of this thesis, CTOT prediction will not be carried out.

The results of this thesis would be used similarly to the ones of my internship - if it is successful, the algorithm could be added to the tool which has been created to sequence aircraft on arrivals. Instead of using a MILP approach, it would be possible to use this new algorithm. The user would be able to set manual Target Times

of Arrivals, if wished, and would be presented with the calculated TTAs by the algorithm. If required (this is a feature in testing in the Operations Control Center as of now), it should be possible to automate the computation with the click of a button. Once clicked, this should allow for automatic sending of TTAs to EUROCONTROL every set number of minutes. With this, the user should not even be impacted by the change in logic of the computation between the MILP and Reinforcement Learning. This computation of TTAs and their quality is then reported monthly to a team between SWISS, SkyGuide and EUROCONTROL, to ensure that from a business point of view the TTAs make sense, as well as that there were no ATC sector overloads.

Bibliography

M. August and J. M. Hernández-Lobato. Taking gradients through experiments: LSTMs and memory proximal policy optimization for black-box quantum control, Apr. 2018. URL http://arxiv.org/abs/1802.04063.

R. E. Bellman. The Theory of Dynamic Programming. Technical report, RAND Corporation, Jan. 1954. URL https://www.rand.org/pubs/papers/P550.html.

J. A. Bennell, M. Mesgarpour, and C. N. Potts. Airport runway scheduling. *4OR*, 9(2):115–138, June 2011. ISSN 1614-2411. doi: 10.1007/s10288-011-0172-x. URL https://doi.org/10.1007/s10288-011-0172-x.

D. Bogado Duffner. Delay Recovery and Reduction of Night Ban Critical Flights at SWISS. Master's thesis, ETH Zurich, Institute for Dynamic Systems and Control, Sept. 2019. URL https://www.research-collection.ethz.ch/handle/20.500.11850/371111?show=full.

L. Breiman. Random Forests. *Machine Learning*, 45:5–32, Oct. 2001. doi: 10.1023/A:1010950718922.

T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language Models are Few-Shot Learners, July 2020. URL http://arxiv.org/abs/2005.14165.

B. Campanelli, P. Fleurquin, V. Eguíluz, J. J. Ramasco, A. Arranz, I. Extebarria, and C. Ciruelos. Modeling reactionary delays in the European air transport network. Nov. 2014.

L. Caranti, M. Carré, and R. Stevens. Optimization of Regulated Airline Arrival Flows via Target Time Management. pages 1–7, Sevilla, Nov. 2023. SESAR. URL https://www.sesarju.eu/sesarinnovationdays.

R. K. Cecen. A stochastic programming model for the aircraft sequencing and scheduling problem considering flight duration uncertainties. *The Aeronautical Journal*, 126(1304):1736–1751, Oct. 2022. ISSN 0001-9240, 2059-6464. doi: 10.1017/aer.2022.17. URL https://doi.org/10.1017/aer.2022.17.

R. K. Cecen and Y. Durmazkeser. Meta-Heuristic Algorithms for Aircraft Sequencing and Scheduling Problem. In T. H. Karakoc, C. O. Colpan, and A. Dalkiran, editors, *Progress in Sustainable Aviation*, Sustainable Aviation, pages 107–118. Springer International Publishing, Cham, 2022. ISBN 978-3-031-12296-5. doi: 10.1007/978-3-031-12296-5_6. URL https://doi.org/10.1007/978-3-031-12296-5_6.

L. Chen, K. Lu, A. Rajeswaran, K. Lee, A. Grover, M. Laskin, P. Abbeel, A. Srinivas, and I. Mordatch. Decision Transformer: Reinforcement Learning via Sequence Modeling, June 2021. URL http://arxiv.org/abs/2106.01345.

T. Chen and C. Guestrin. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794, Aug. 2016. doi: 10.1145/2939672.2939785. URL http://arxiv.org/abs/1603.02754.

D. Cuppen. Conflict Prioritization with Multi-Agent Deep Reinforcement Learning. 2022. URL https://repository.tudelft.nl/islandora/object/uuid%3A02c5d5a2-a203-46f9-8f75-1be017d5675e.

R. Dalmau Codina and F. Herrema. Encoder-Decoder Approach to Predict Airport Operational Runway Configuration A case study for Amsterdam Schiphol airport. pages 1–8, 2019. URL https://upcommons.upc.edu/handle/2117/181178.

R. Dalmau-Codina, B. Genestier, C. Anoraud, P. Choroba, and D. Smith. A Machine Learning Approach to Predict the Evolution of Air Traffic Flow Management Delay. page 8, Sept. 2021. URL https://www.researchgate.net/publication/354809858_A_Machine_Learning_Approach_to_Predict_the_Evolution_of_Air_Traffic_Flow_Management_Delay.

G. B. Dantzig. Reminiscences about the origins of linear programming. *Operations Research Letters*, 1(2):43–48, Apr. 1982. ISSN 0167-6377. doi: 10.1016/0167-6377(82)90043-8. URL https://www.sciencedirect.com/science/article/pii/0167637782900438.

G. B. Dantzig. Origins of the simplex method. In *A history of scientific computing*, pages 141–151. Association for Computing Machinery, New York, NY, USA, June 1990. ISBN 978-0-201-50814-7. URL https://doi.org/10.1145/87252.88081.

L. Delgado, G. Gurtner, P. Mazzarisi, S. Zaoli, D. Valput, A. Cook, and F. Lillo. Network-wide assessment of ATM mechanisms using an agent-based model. *Journal of Air Transport Management*, 95:102108, Aug. 2021. ISSN 0969-6997. doi: 10.1016/j.jairtraman.2021.102108. URL https://www.sciencedirect.com/science/article/pii/S0969699721000910.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, May 2019. URL http://arxiv.org/abs/1810.04805.

Z. Du, J. Zhang, and B. Kang. A Data-Driven Method for Arrival Sequencing and Scheduling Problem. *Aerospace*, 10(1):62, Jan. 2023. ISSN 2226-4310. doi: 10.3390/aerospace10010062. URL https://www.mdpi.com/2226-4310/10/1/62.

EUROCONTROL. Innovative Slot Allocation: an Overview, Jan. 2014. URL https://www.eurocontrol.int/node/9977.

EUROCONTROL. Arrival Planning Information (API) implementation guide, May 2022a. URL https://www.eurocontrol.int/publication/arrival-planning-information-api-implementation-guide.

EUROCONTROL. New EUROCONTROL Analysis Paper: 2022 The year European aviation bounced back, Dec. 2022b. URL https://www.eurocontrol.int/publication/eurocontrol-analysis-paper-2022-year-european-aviation-bounced-back.

EUROCONTROL. ATFCM Users Manual, Mar. 2023. URL https://www.eurocontrol.int/publication/atfcm-users-manual.

EUROCONTROL. Forecast Update 2023-2029, Oct. 2023. URL https://www.eurocontrol.int/publication/eurocontrol-forecast-update-2023-2029-autumn-2023.

J. Evler, M. Schultz, and H. Fricke. Flight Prioritization and Turnaround Recovery - Integrating Tactical ATFCM Slot Swapping into Resource-Constrained Ground Operations. Sept. 2021.

FlughafenZurich. Operating concepts Flughafen Zuerich, Jan. 2023. URL https://www.flughafen-zuerich.ch/en/company/media-policy-and-investors/politics-and-business/operating-concepts. Publication Title: Operating Framework.

F. Furini, M. P. Kidd, C. A. Persiani, and P. Toth. Improved rolling horizon approaches to the aircraft sequencing problem. *Journal of Scheduling*, 18(5):435–447, Oct. 2015. ISSN 1099-1425. doi: 10.1007/s10951-014-0415-8. URL https://doi.org/10.1007/s10951-014-0415-8.

A. Geletu, M. Klöppel-Gersdorf, H. Zhang, and P. Li. Advances and applications of chance-constrained approaches to systems optimisation under uncertainty. *International Journal of Systems Science*, 44:1209–1232, July 2013. doi: 10.1080/00207721.2012.670310.

D. Gillen. Airline Business Models and Networks: Regulation, Competition and Evolution in Aviation Markets. *Review of Network Economics*, 5(4), Dec. 2006. ISSN 1446-9022. doi: 10.2202/1446-9022.1103. URL https://www.degruyter.com/document/doi/10.2202/1446-9022.1103/html.

F. Giuliari, I. Hasan, M. Cristani, and F. Galasso. Transformer Networks for Trajectory Forecasting, Oct. 2020. URL http://arxiv.org/abs/2003.08111.

A. Graham. Understanding the low cost carrier and airport relationship: A critical analysis of the salient issues. *Tourism Management*, 36:66–76, June 2013. ISSN 0261-5177. doi: 10.1016/j.tourman.2012.11.011. URL https://www.sciencedirect.com/science/article/pii/S0261517712002245.

T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In *Proceedings of the 35th International Conference on Machine Learning*, pages 1861–1870. PMLR, July 2018. URL https://proceedings.mlr.press/v80/haarnoja18b.html.

P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger. Deep Reinforcement Learning that Matters, Jan. 2019. URL http://arxiv.org/abs/1709.06560.

S. Hochreiter and J. Schmidhuber. Long Short-term Memory. *Neural computation*, 9:1735–80, Dec. 1997. doi: 10.1162/neco.1997.9.8.1735.

R. Hoogendoorn. Dynamic Airline Centric Inbound Priority Sequencing: A case study on Westerly morning arrivals for KLM at Schiphol. 2022. URL https://repository.tudelft.nl/islandora/object/uuid%3A4c08bbcf-21c9-4074-81bf-b66099b67734.

C. Huang and Y. Xu. Integrated Frameworks of Unsupervised, Supervised and Reinforcement Learning for Solving Air Traffic Flow Management Problem. In *2021 IEEE/AIAA 40th Digital Avionics Systems Conference (DASC)*, pages 1–10, Oct. 2021. doi: 10.1109/DASC52595.2021.9594397.

S. Ikli, C. Mancel, M. Mongeau, X. Olive, and E. Rachelson. Coupling Mathematical Optimization and Machine Learning for the Aircraft Landing Problem. June 2020. URL https://hal-enac.archives-ouvertes.fr/hal-02873423.

N. Ivanov, F. Netjasov, R. Jovanovi, S. Starita, and A. Strauss. Air Traffic Flow Management slot allocation to minimize propagated delay and improve airport slot adherence. *Transportation Research Part A: Policy and Practice*, 95:183–197, Jan. 2017. ISSN 0965-8564. doi: 10.1016/j.tra.2016.11.010. URL https://www.sciencedirect.com/science/article/pii/S0965856416300052.

J. Janssen. Capacity-based Arrival Sequencing and Trajectory Optimization for Continuous Descent Operations. 2019. URL https://repository.tudelft.nl/islandora/object/uuid%3Af483be8e-522b-473c-8e83-08cd514915f7.

G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://papers.nips.cc/paper_files/paper/2017/hash/6449f44a102fde848669bdd9eb6b76fa-Abstract.html.

Y. Keneshloo, T. Shi, N. Ramakrishnan, and C. K. Reddy. Deep Reinforcement Learning For Sequence to Sequence Models, Apr. 2019. URL http://arxiv.org/abs/1805.09461.

A. Khassiba, F. Bastin, S. Cafieri, B. Gendron, and M. Mongeau. Two-Stage Stochastic Mixed-Integer Programming with Chance Constraints for Extended Aircraft Arrival Management. *Transportation Science*, 54(4): 897–919, July 2020. ISSN 0041-1655. doi: 10.1287/trsc.2020.0991. URL https://pubsonline.informs.org/doi/10.1287/trsc.2020.0991.

A. H. Land and A. G. Doig. An Automatic Method of Solving Discrete Programming Problems. *Econometrica*, 28(3):497–520, 1960. ISSN 0012-9682. doi: 10.2307/1910129. URL https://www.jstor.org/stable/1910129.

Learning. 10.7. Encoder-Decoder Seq2Seq for Machine Translation Dive into Deep Learning 1.0.0-beta0 documentation, Jan. 2020. URL https://d2l.ai/chapter_recurrent-modern/seq2seq.html.

T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning, July 2019. URL http://arxiv.org/abs/1509.02971.

M. Lübbecke. Column Generation. Jan. 2011. ISBN 978-0-470-40053-1. doi: 10.1002/9780470400531.eorms0158.

A. Malinin, L. Prokhorenkova, and A. Ustimenko. Uncertainty in Gradient Boosting via Ensembles, Apr. 2021. URL http://arxiv.org/abs/2006.10562.

V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing Atari with Deep Reinforcement Learning, Dec. 2013. URL http://arxiv.org/abs/1312.5602.

A. Montlaur and L. Delgado. Flight and passenger delay assignment optimization strategies. *Transportation Research Part C: Emerging Technologies*, 81:99–117, Aug. 2017. ISSN 0968-090X. doi: 10.1016/j.trc.2017.05.011. URL https://www.sciencedirect.com/science/article/pii/S0968090X17301420.

A. Muharremoglu. *The aircraft sequencing problem with arrivals and departures*. Thesis, Massachusetts Institute of Technology, 2000. URL https://dspace.mit.edu/handle/1721.1/9156.

S. Nakaoka. Chapter 3 - Data-Driven Mathematical Modeling of Microbial Community Dynamics. In A. S. R. Srinivasa Rao and C. R. Rao, editors, *Handbook of Statistics*, volume 39 of *Integrated Population Biology and Modeling, Part A*, pages 93–130. Elsevier, Jan. 2018. doi: 10.1016/bs.host.2018.08.001. URL https://www.sciencedirect.com/science/article/pii/S0169716118300282.

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. *Transportation Research Part E: Logistics and Transportation Review*, 106:115–136, Oct. 2017. ISSN 1366-5545. doi: 10.1016/j.tre.2017.08.006. URL https://www.sciencedirect.com/science/article/pii/S1366554517301771.

A. Pawelek, R. Dalmau, P. Lichota, and X. Prats. Assessment of arrival traffic synchronisation with RTAs and fuel-efficient trajectories. In *17th AIAA Aviation Technology, Integration, and Operations Conference*, AIAA AVIATION Forum. American Institute of Aeronautics and Astronautics, June 2017. doi: 10.2514/6.2017-3770. URL https://arc.aiaa.org/doi/10.2514/6.2017-3770.

E. Pels. Airline network competition: Full-service airlines, low-cost airlines and long-haul markets. *Research in Transportation Economics*, 24(1):68–74, Jan. 2008. ISSN 0739-8859. doi: 10.1016/j.retrec.2009.01.009. URL https://www.sciencedirect.com/science/article/pii/S0739885909000171.

N. Pilon, L. Guichard, Z. Bazso, G. Murgese, and M. Carré. User-Driven Prioritisation Process (UDPP) from advanced experimental to pre-operational validation environment. *Journal of Air Transport Management*, 97:102124, Oct. 2021. ISSN 0969-6997. doi: 10.1016/j.jairtraman.2021.102124. URL https://www.sciencedirect.com/science/article/pii/S096969972100106X.

L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin. CatBoost: unbiased boosting with categorical features, Jan. 2019. URL http://arxiv.org/abs/1706.09516.

H. N. Psaraftis. A Dynamic Programming approach to the Aircraft Sequencing problem. Technical Report, Cambridge, Mass. : Massachusetts Institute of Technology, Flight Transportation Laboratory, [1978], 1978. URL https://dspace.mit.edu/handle/1721.1/67911.

M. Ribeiro, J. Ellerbroek, and J. Hoekstra. Improving Algorithm Conflict Resolution Manoeuvres with Reinforcement Learning. *Aerospace*, 9(12):847, Dec. 2022. ISSN 2226-4310. doi: 10.3390/aerospace9120847. URL https://www.mdpi.com/2226-4310/9/12/847.

S. Saaybi, A. Majid, V. Prasad, A. Koubaa, and C. Verhoeven. *Covy: An AI-powered Robot for Detection of Breaches in Social Distancing*. July 2022. doi: 10.48550/arXiv.2207.06847.

B. F. Santos, M. M. E. C. Wormer, T. A. O. Achola, and R. Curran. Airline delay management problem with airport capacity constraints and priority decisions. *Journal of Air Transport Management*, 63:34–44, Aug. 2017. ISSN 0969-6997. doi: 10.1016/j.jairtraman.2017.05.003. URL https://www.sciencedirect.com/science/article/pii/S0969699716302514.

M. Schmidt. A review of aircraft turnaround operations and simulations. *Progress in Aerospace Sciences*, 92:25–38, July 2017. ISSN 0376-0421. doi: 10.1016/j.paerosci.2017.05.002. URL https://www.sciencedirect.com/science/article/pii/S0376042117300039.

C. G. Schuetz, T. Lorünser, S. Jaburek, K. Schuetz, F. Wohner, R. Karl, and E. Gringinger. A Distributed Architecture for Privacy-Preserving Optimization Using Genetic Algorithms and Multi-party Computation. In M. Sellami, P. Ceravolo, H. A. Reijers, W. Gaaloul, and H. Panetto, editors, *Cooperative Information Systems*, Lecture Notes in Computer Science, pages 168–185, Cham, 2022. Springer International Publishing. ISBN 978-3-031-17834-4. doi: 10.1007/978-3-031-17834-4_10.

J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal Policy Optimization Algorithms, Aug. 2017. URL http://arxiv.org/abs/1707.06347.

J. Schummer and A. Abizada. Incentives in landing slot problems. *Journal of Economic Theory*, 170:29–55, July 2017. ISSN 0022-0531. doi: 10.1016/j.jet.2017.04.003. URL https://www.sciencedirect.com/science/article/pii/S002205311730042X.

SESAR. Domino - Novel tools to evaluate ATM systems coupling under future deployment scenarios, Dec. 2019a. URL https://domino-eu.com/.

SESAR. SESAR Joint Undertaking \ E-AMAN common service, Jan. 2019b. URL https://www.sesarju.eu/sesar-solutions/e-aman-common-service.

SESAR. SESAR Joint Undertaking \ SlotMachine - A Privacy-Preserving Marketplace for Slot Management, Dec. 2022. URL https://www.sesarju.eu/projects/SlotMachine.

A. Singh, N. Thakur, and A. Sharma. A review of supervised machine learning algorithms. In *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*, pages 1310–1315, Mar. 2016.

Statista. Low cost carrier market share worldwide 2007-2020, Jan. 2021. URL https://www.statista.com/statistics/586677/global-low-cost-carrier-market-capacity-share/. Publication Title: Statista.

I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to Sequence Learning with Neural Networks. In *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. URL https://proceedings.neurips.cc/paper_files/paper/2014/hash/a14ac55a4f27472c5d894ec1c3c743d2-Abstract.html.

R. S. Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3(1):9–44, Aug. 1988. ISSN 1573-0565. doi: 10.1007/BF00115009. URL https://doi.org/10.1007/BF00115009.

R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction, 2nd ed.* Reinforcement learning: An introduction, 2nd ed. The MIT Press, Cambridge, MA, US, 2018. ISBN 978-0-262-03924-6.

R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour. Policy Gradient Methods for Reinforcement Learning with Function Approximation. In *Advances in Neural Information Processing Systems*, volume 12. MIT Press, 1999. URL https://papers.nips.cc/paper_files/paper/1999/hash/464d828b85b0bed98e80ade0a5c43b0f-Abstract.html.

M. Taghian, A. Asadi, and R. Safabakhsh. A Reinforcement Learning Based Encoder-Decoder Framework for Learning Stock Trading Rules, Jan. 2021. URL http://arxiv.org/abs/2101.03867.

Y. Tang and Y. Xu. Multi-Agent Deep Reinforcement Learning for Solving Large-scale Air Traffic Flow Management Problem: A Time-Step Sequential Decision Approach. In *2021 IEEE/AIAA 40th Digital Avionics Systems Conference (DASC)*, pages 1–10, Oct. 2021. doi: 10.1109/DASC52595.2021.9594329.

Y. Tian, C. Xu, M. Sun, C. Li, and R. Sun. Study on Arrival Aircraft Sequencing Based on Optimization of Point Merge Procedure. *Discrete Dynamics in Nature and Society*, 2021:e6663161, Apr. 2021. ISSN 1026-0226. doi: 10.1155/2021/6663161. URL https://www.hindawi.com/journals/ddns/2021/6663161/.

L. Urbanucci. Limits and potentials of Mixed Integer Linear Programming methods for optimization of polygeneration energy systems. *Energy Procedia*, 148:1199–1205, Aug. 2018. ISSN 1876-6102. doi: 10.1016/j.egypro.2018.08.021. URL https://www.sciencedirect.com/science/article/pii/S1876610218303072.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention Is All You Need, Dec. 2017. URL http://arxiv.org/abs/1706.03762.

R. Vervaat. Airline based priority flight sequencing: of aircraft arriving at an airport. 2020. URL https://repository.tudelft.nl/islandora/object/uuid%3A95dd53f0-511e-4060-9271-aa34bf3237a2.

B. Verweij, S. Ahmed, A. J. Kleywegt, G. Nemhauser, and A. Shapiro. The Sample Average Approximation Method Applied to Stochastic Routing Problems: A Computational Study. *Computational Optimization and Applications*, 24(2):289–333, Feb. 2003. ISSN 1573-2894. doi: 10.1023/A:1021814225969. URL https://doi.org/10.1023/A:1021814225969.

B. Vonk. Exploring reinforcement learning methods for autonomous sequencing and spacing of aircraft. 2019. URL https://repository.tudelft.nl/islandora/object/uuid%3A2e776b60-cd4e-4268-93e3-3fcc81cd794f.

X. Wang, W. Xiong, H. Wang, and W. Y. Wang. Look Before You Leap: Bridging Model-Free and Model-Based Reinforcement Learning for Planned-Ahead Vision-and-Language Navigation. pages 37–53, 2018. URL https://openaccess.thecvf.com/content_ECCV_2018/html/Xin_Wang_Look_Before_You_ECCV_2018_paper.html.

F. Yang, T. Jin, T.-Y. Liu, X. Sun, and J. Zhang. Boosting Dynamic Programming with Neural Networks for Solving NP-hard Problems. In *Proceedings of The 10th Asian Conference on Machine Learning*, pages 726–739. PMLR, Nov. 2018. URL https://proceedings.mlr.press/v95/yang18a.html.

K. Zhang, Z. Yang, and T. Baar. Multi-Agent Reinforcement Learning: A Selective Overview of Theories and Algorithms. In K. G. Vamvoudakis, Y. Wan, F. L. Lewis, and D. Cansever, editors, *Handbook of Reinforcement Learning and Control*, Studies in Systems, Decision and Control, pages 321–384. Springer International Publishing, Cham, 2021. ISBN 978-3-030-60990-0. doi: 10.1007/978-3-030-60990-0_12. URL https://doi.org/10.1007/978-3-030-60990-0_12.