École Polytechnique Fédérale de Lausanne
School of Computer and Communication Sciences
PUJOL Guilhem

# Master thesis

Confidential

- Internship title: Optimization of arrivals and departures flow management at congested airports
- Supervising professor: Michel Bierlaire
- Supervisor at the company: Olivier Ratier
- Internship date: 18th of August 2014 - 13th of February 2015
- Name and address of the company:
  Amadeus S.A.S.
  485 route du Pin Montard BP 69,
  06902 SOPHIA ANTIPOLIS CEDEX
  FRANCE
- Student address:
  85 rue Henri Poincaré
  Résidence Oxford
  Appartement B120
  06410 BIOT
  FRANCE

# Acknowledgments

I would like to thank the École Polytechnique Fédérale de Lausanne for giving me the opportunity to do a research internship, and more particularly Michel Bierlaire and Shadi Sharif Azadeh from the Transportation and Mobility Laboratory who accepted to supervise my master thesis.

My gratitude also goes to my manager at Amadeus SAS, Olivier Ratier, for his constant advise and support, and more generally to the whole RMS team of the Airport IT division.

# Contents

# Introduction

The air traffic is increasing in many European airports. As a consequence, delays become more frequent, which is an important issue for passengers, but also for airports, airlines and ground handlers (because of the refunding cost and the additionnal use of ressources they cause).

Various approaches exist to minimize congestion. One of them is to increase the airport capacity by building new infrastructures. However, as the runway system is often the main bottleneck, this would require the construction of new runways, which is often a subject of contention with the nearby residents, and is in some case physically impossible. A striking example is London Heathrow, the busiest European airport. It has only two operationnal runways that are used to handle more than 1000 operations (arrivals or departures) every day. Though it is operating very near its maximum capacity, and still experiences yearly traffic increases, every project of runway addition has so far been opposed by the local authorities.

Therefore, another interesting approach is to try to optimize the airport policy and particulary everything that regards the use of the runway system without considering the addition of ressources. Indeed, the airport controllers have to manage simultaneously the communication with all nearby aircrafts and the airport operations scheduling at the same time. As the best scheduling can sometimes be intricated and constructed only by taking into account operations that occurs several hours in the future, it is nearly impossible for a human being to find optimal policies in real time, so computer assistance may be beneficial.

We choose to focus on the runway operations, which have been proven to be the main bootleneck in the most congested airports in the world. The limitation of the runway availability causes a correlation between the number of achievable arrivals and departures: allowing more arrivals has an impact on the number of possible departures. This can be the case if we make aircrafts land on a runway that is also used for take-off, if the incoming taxiways cross a departure runway, or if the landing runway is too close to the departing runway, in which case a gaps between operations must be enforced even if they do not take place on the same runway.

The achievable departure and arrivals rates are constrained by the runway configuration, that is the choice of which runways are used, in which direction, and if they are used for departures, arrivals, or both. In an ideal case, choosing the configuration would be a subproblem of the choice of arrival/departure rates (one could determine which couple of arrival/departure rates are possible for each runway configuration, then choose the departure and arrival rates and fix the runway configuration accordingly). However:

- there are meteorological constraints (mostly the wind speed and direction) that can restrict the choice of runway configuration

- a runway configuration change is likely to have a negative impact on the operations unless perfect synchronization is achieved (all aircrafts would have to be positionned to use the new configuration at the same time), which seems difficult in a system with so much incertitude, so it is in some cases better to remain in a sub-optimal configuration to avoid this.

This works has two main objectives revolving around providing assistance to airport controllers on these matter. The first one is to study a model of airport operations that will allow to find good choices for arrival and departure rates, along with the best timing for runway configuration changes. The second goal is to assess the value of indicators of the airport congestion (queue lengths, delay), assuming that the proposed strategy are followed, or when they are subject to modifications by the airport controller.

# Chapter 1

# Literature review

Various approaches have already been developped to tackle the problem of airport congestion. They can be regrouped in 3 categories: microscopic, macroscopic and stochastic approaches [3].

The microscopic approaches try to find an optimized planning for each flight, adjusting for instance the time at which each departing aircraft leave its block so as to minimize the congestion on the taxiways or building an optimized sequencing for departures at each runway extremity in order to have low inter-departure gaps [1].

While these works are extremely valuable, the total number of possible scenarios in a microscopic model is huge, which often prevents us to solve the problem exactly. Moreover, a tool based on a microscopic approach would have to take into account many constraints specific to each airport since its output is a detailed planning of operation which needs to be acceptable to the airport controller. A complete list of these constraints may be difficult to obtain and would require extensive discussions with many of the actor involved in each airports.

It is also interesting to consider macroscopic-level model, where each day of operation is divided into time slots of a small size (10 or 15 minutes) and all the flights in a time-slot are treated as a whole, which means that only the total number of operations occuring during the time slot is relevant to the model. In this case the output of the model is only a macroscopic variable, such as the advised number of departures to process during a time-slot. A mix integer programming model that solves the problem of runway configuration management and of the choice of arrival and departure rates using a macroscopic approach has been developped and works quite well if we assume there are no uncertaineties [2].

Finally, a stochastic approach is a model that tries to take uncertainety into account. It will often be macroscopic, since aggregating events will make it possible to manipulate meaningful probabilities and expectations. The model of airport operations that we will consider in this internship is DELAY [7], which has recently been used to propose a solution to the problem we are considering in this internship [5] [6]. This model was develloped and tested using data from American airports.

This study carries three main contributions:

- as there are no European equivalents of the American ASPM database, determining how to gather sufficient information to calibrate the DELAY model for a European airport
- seeing how well the model performs on medium-sized airports where the congestion mechanics may differ from those of the biggest American airports
- studying the impact of stochasticity by comparing the results with those of a purely deterministic model

# Chapter 2

# Work on available data

## 2.1 Data sources

Several types of data are required in order to apply a model of runway congestion on an airport:

- a history of the operationnal rates achieved by the airport, which can be deduced from the actual take-off and landing times of the aircrafts
- the queue lengths and delays, for which we will need the operational schedule, so that we can compare it with the actual timings of the operations
- categorization data: the capacity of the airport is affected by the runway configuration and the weather (among other factors), so knowing the value of these variables throughout our operational history will help increasing the precision of the model

### 2.1.1 Eurocontrol data

Amadeus was recently granted access to a Eurocontrol service called B2B NM (Business-to-business network manager). Using it, one can retrieve data about the flights that take-off, land, or flight over Europe. At any moment, only data regarding flights scheduled after 9pm on the day before are available, and there are also limitations on the number of queries allowed each day.

The data provided by Eurocontrol contain the initial flight plan for each flight, all updates to this plan that occurred before take-off, and the Flight Updated Messages (FUM) sent by the aircraft during and after the flight.

For the purpose of this work, a script collecting the data of all incoming and outcoming flights at seven European airports was run regurarly, so as to compile a history of the operations that have occured at the targeted airports during several months.

Eurocontrol is the main source of data for this work, since it gives access to both scheduled and actual timings of arrivals and departures. More precisely, the flight plans and FUMs provided by Eurocontrol allow us to retrieve, for each departing flight:

- the **EOBT** (Estimated Off-Block Time) which is the time when the aircraft is scheduled to leave the block and start taxiing
- in five of the seven target airports, the **AOBT** (Actual Off-Block Time) when the aircraft actually leaves the block
- the **ATOT** (Actual Take Off Time) which is the time at which the aircraft takes off

For the incoming flights, we get:

- an **ELDT** (Estimated LanDing Time), which is an estimated time of arrival computed at takeoff
- the **ALDT** (Actual LanDing Time), which the time at which the aircraft lands at the destination airport

- in most case, the runway used for landing

Because Eurocontrol data are not publicly available and as of our work is based on them, we will not name the seven studied airports and will instead refer to them as airport 1, airport 2, ..., airport 7.

### 2.1.2 Amadeus data

Amadeus has direct access to the operational data of one of the seven studied airports. This enabled us to assess the reliability and completness of Eurocontrol data. The result was satisfying, in that the vast majority of flights recorded in Amadeus databases also existed for Eurocontrol, and the timing retrieved in the flight plan matched the operational data quite accurately.

The only issue is that automatically matching flight plans from Eurocontrol with flights in Amadeus database is not trivial, as the system to attribute callsigns (unique identifiers) to flights slightly differs for the two organizations. As the two sources of data gives an ELDT, the departure airport and the destination of each flight, it is in many cases possible to use this information to match Amadeus flights with their Eurocontrol flight plans.

### 2.1.3 Meteorological data

As previously mentioned, the meteorological conditions have a great impact on the activities of any airport: the wind speed and its direction constrain the runway utilization, and inter-departures and inter-arrivals windows are longer when the visibility is insufficient. The weather conditions are recorded in every airport, and often publicly accessible in real-time, but finding historical data can be difficult. That-is-why we decided to build the history of the weather conditions at the studied airports by regularly scraping the daily weather reports available on a public website. [4]

The visibility conditions can be divided into 4 categories, which are from the most to the least favorable: VFR (Visual Flight Rule), MVFR (Marginal Visual Flight Rule), IFR (Instrumental Flight Rule), LIFR (Low Instrumental Flight Rule). The extracted weather reports directly provide the category that is applied, so we did not need to deduce it from more complex meteorological data. For some airports, it was very rare to observe any other category than VFR, leading to few information about the operational rates achievable under bad weather conditions.

## 2.2 Data analysis

### 2.2.1 Inference of information

Some information presented as being part of the Eurocontrol data feed were in fact not explicit, and had to be inferred by comparison with the data available to Amadeus in one of the airports. For instance, the ATOT and AOBT are never given as such in flight plans or flight update messages. However, the timings at which the first and second flight update messages are sent correspond, with a margin of error of less than a minute, to the AOBT and ATOT recorded in Amadeus database at airport 1, so this gave us a way to induce those two timings from Eurocontrol data.

It is also possible to determine which runway was used by some incoming flights. Indeed, the majority of arriving aircrafts send a GPS coordinate corresponding to their position only one or two minutes before landing in there last Flight Update Message. This can be used to compute the runway axis to which the aircraft was the closest just before landing, and therefore, which runway was used and in which direction.

For instance, if we have an airport with two parallel runways as in figure 2.1 and the recorded position of the arriving aircraft is point $M$, the distance between $M$ and the runway axis of runway $i$ is given by:

$$d_i = \frac{|\overrightarrow{A_iB_i} \wedge \overrightarrow{A_iM}|}{||\overrightarrow{A_iB_i}||}$$

Once we know which axis is the best, we can compute the angle between the runway axis and the straight line that join the position of the plane and one extremity of the runway with:

$$\alpha_i = \arccos(\frac{\overrightarrow{A_iB_i}.\overrightarrow{A_iM}}{||\overrightarrow{A_iB_i}|| \times ||\overrightarrow{A_iM}||})$$

We decide not to conclude on the runway that was used if this angle is too high, since this means that the aircraft was not yet aligned with the landing runway at that moment.

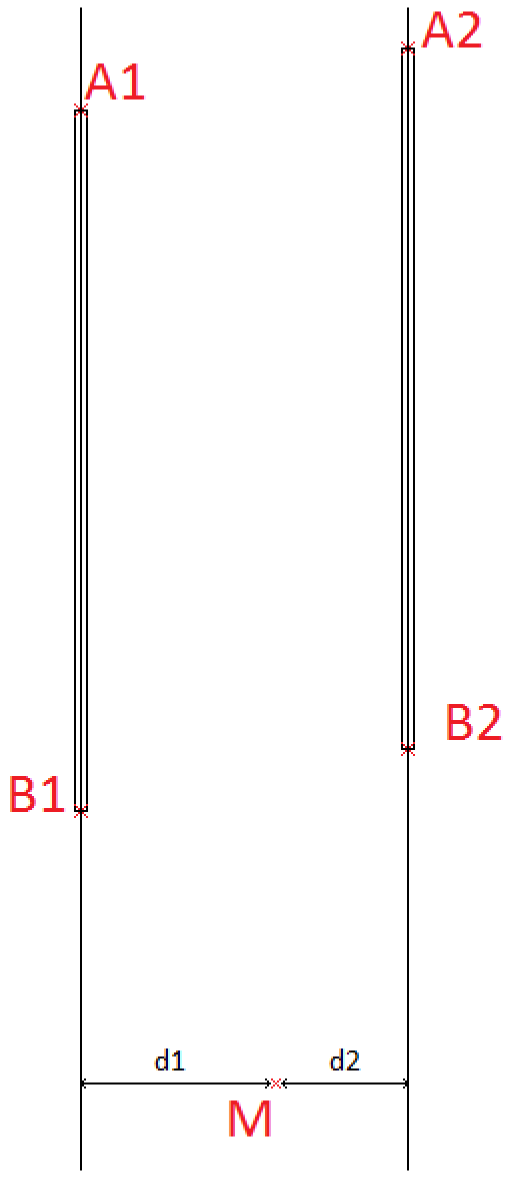


Figure 2.1: Finding the landing runway

### 2.2.2 Airport capacities

From the list of the actual timings of most operations that took place at a given airport, we want to deduce information regarding the capacity of the airport, that is, the maximum number of operations that can be achieved per unit of time. This formulation is a bit imprecise, because in most cases, there is a tradeoff between arrivals and departures: the airport controllers can decide to allow more departures, at the cost of performing less arrivals, and vice-versa. However, deciding to increase the number of departures by 1 will not always cause a decrease of exactly 1 in the maximum number of arrivals and therefore, there is no relevant single maximum number of operations that can be reached per unit of time. Instead, we want the list of couples of numbers of arrivals and departures that are simultaneously achievable during a given period of time.

For this, we divide our whole history in 15 minute time slots, and count the number of arrivals and departures in each time slot (that is, the number of flights with an ALDT or ATOT in the time slot). We then plot a graph were each point represents a couple of number of arrivals and departures achieved during at least one slot, as in figure 2.2. In this figure, the colors of the points give an indication of the number of times they were reached during the period we are studying. The brighter points were reached 1, 2, 3 or 4 times, whereas the darkest one were reached 5 times or more.

We remove outliers by erasing all points that were not reached at least 5 times throughout the entire history. Finally, we add all points that are dominated by at least another point is the graph. More precisely:

$$\text{newPointSet} = \{(a, d) \in \mathbb{N}^2 | \exists (a', d') \in \text{oldPointSet} : a \leq a' \wedge d \leq d'\}$$

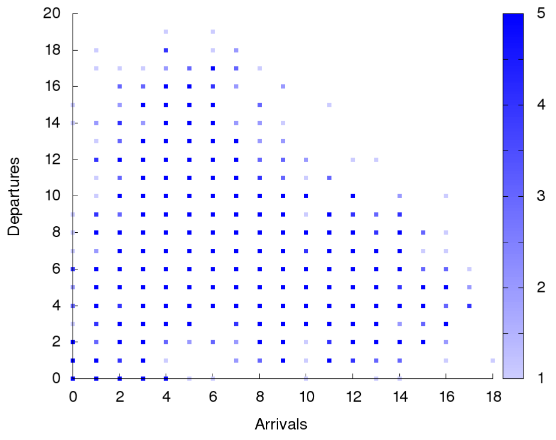


Figure 2.2: Achieved arrival and departure rates

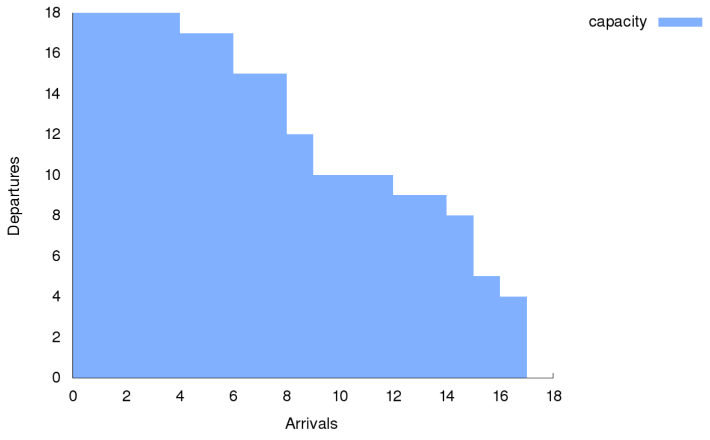


Figure 2.3: Induced Runway Configuration Capacity Envelop

This gives us an admissible operational domain (figure 2.3), which we will refer to as a RCCE (Runway Configuration Capacity Envelop), following the litterature. This name highlights the fact that this domain will depend on the runway configuration in use, as some configurations can achieve more or fewer departures or arrivals than others. Another factor which influence the RCCE is the weather, and particularly the visibility: the reglementation imposes longer gaps between arrivals and departures under bad visibility conditions.

For these reasons, we annotated each time slot with the flight rule that was applied during the slot (when we were sure that it did not change during the slot) by using the meteorological data corpus. This allowed us to make the operation described above separately for each flight rules and therefore to get the four corresponding RCCE.

The problem of the differences between runway configuration is a bit harder to handle, because we only have the runway information for most of the arrivals. Whether this is enough to determine the full runway configuration depends on the airport. Examples of encountered infrastructure are given in table 2.4.

| Runway infrastructure | Possible | Comment |
|---|---|---|
| Single runway in mixed mode | yes | Use the arrival data to determine the direction |
| Two parallel runways in mixed mode | yes | Use the arrival data to determine the direction |
| Two parallel runways in segregated mode | yes | Use the arrival data to determine the direction and the arrival/departure runway |
| Two main parallel runways + third runway | partial | No way to know whether the third runway is active if it is not used by arrivals |
| Four parallel runways in segregated mode | yes | Use the arrival data to determine the direction and the arrival/departure runways |
| Three parallel runways in segregated mode + three other runways | partial | Main runways only |

Figure 2.4: Runway configuration determination for each studied airport

### 2.2.3 Conclusions on the project relevance

Having RCCE gives us a set of allowed decisions for the problem that we want to address, namely choosing the best arrival and departure rates. These graphs can also confirm or infirm the correlation between the maximal arrivals and departures rates, which is what makes the problem interesting (otherwise, we would choose to always operate at maximum departure rate and maximum arrival rate).

The result depends on the considered airport. Airports with a single runway obviously have high correlation between arrivals and the maximum number of departures. Airports with more than three runways ore more have configurations that allow more departures or more arrivals, depending on how many runways are assigned to each operations. Some airport with two runways choose to operate in mix-mode, which means that each runway is used for both arrivals and departures. In this case, we have the same kind of correlation between the arrival and departure rates as the one observed when there is only one runway. For airports which decide to use one runway for arrivals and one runway for departure, if the distance between the two runways is low, the operations on one of the runway can influence what is allowed on the other runway. In all these cases, we can observe the correlation we expected and get RCCE looking like 2.5, though to a lesser extent in the last scenario.

In fact, only one of the studied airports, which had two runways separated by a large distance, showed a rectangle-shaped RCCE (figure 2.6) which is a sign of independency between the arrival and departure processes. For this airport, the benefit of our project would be limited to the help provided to choose the best timing for an inversion of the runways direction or any other operations requiring a temporary traffic interruption.

In the two RCCE bellow, we added red crosses to depict the scheduled number of operations that were encountered during at least a slot. This shows that the scheduling levels sometimes exceed the airport capacity, leading us to expect some congestion dued to the limited runway capacity. This hints that the airport situation can be improved by an optimal rate decision policy.

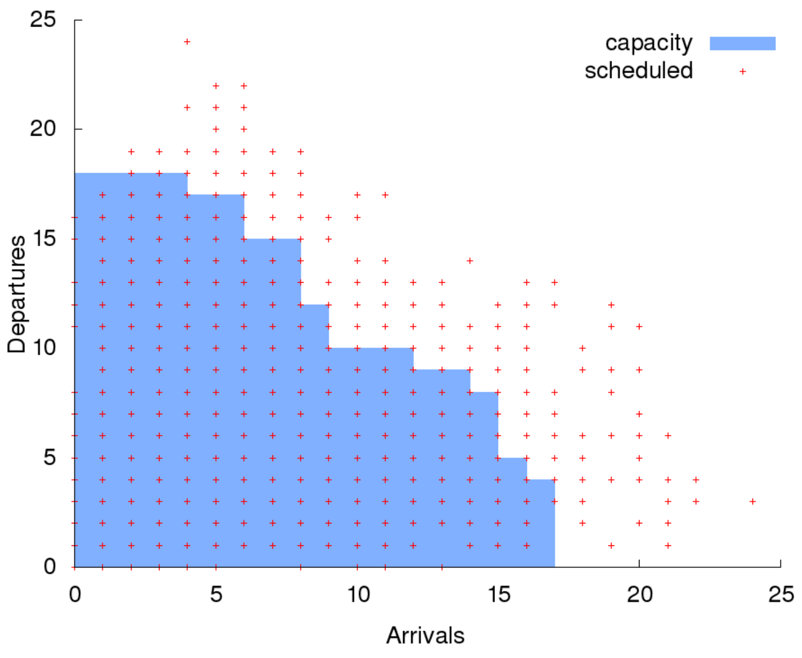


Figure 2.5: Typical RCCE example plotted in VMC, with scheduling levels

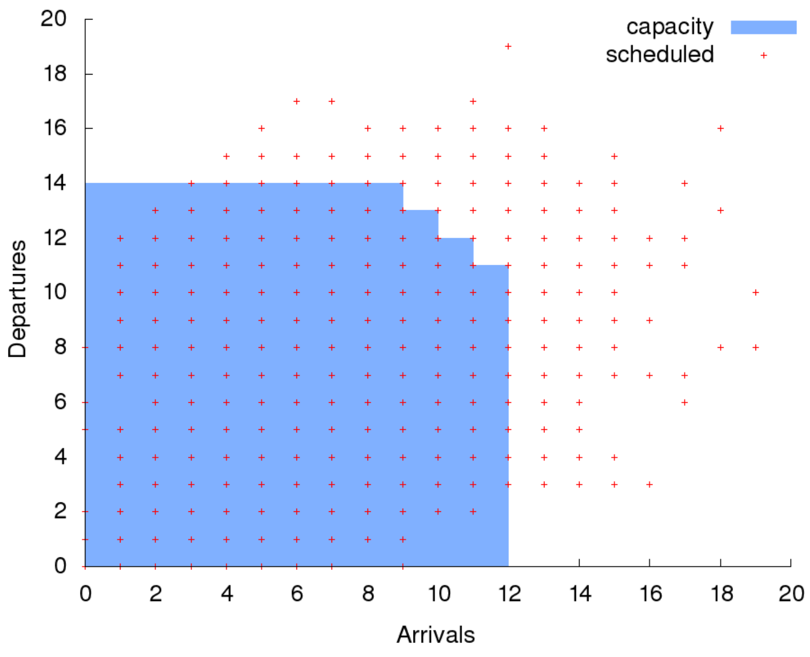


Figure 2.6: RCCE of an airport with very low correlation between arrivals and departures, plotted in VMC, with scheduling levels

# Chapter 3

# Modelization

## 3.1 Runways as a queuing system

Our model of airport operations is very close to the one used by A. Jacquillat [6], except that we also considered a deterministic evolution function to assess the interest of taking stochasticity into account. A detailed description of this model follows.

We consider a queuing system with two servers: one for arrivals and one for departures, which have to separate queues. The two queues have a maximum capacity of $N = 30$ (this is to have a finite state space, so that the problem is tractable).

At the beginning of each slot, our programm will be informed of the current length of both queues, and will look at the most up-to-date version of the schedule it has accesss to. It will contain the scheduled number of arrivals and departures for the current slot but also for all remaining slot of the day, though the reliability of the value decrease with time. The program is then expected to give a recommendation regarding the arrival and departure rate for this slot.

We tried two possible ways of computing the new queue length at the end of the selected slot as a function of the situation at the beginning of the slot and the choice made by our program:

- with a deterministic evolution function, which is the same for departures and arrivals:

$$\text{newQueue} = \text{initialQueue} + \text{scheduledOperations} - \text{chosenServiceRate}$$

- with a stochastic evolution function. In this case, we consider that the arrivals in the queue follows a Poisson law which rate $\lambda$ is the scheduled number of operations during the time slot, and that the service follows an Erlang law of parameter $k = 4$, which rate $\mu$ will be the decided service rate.

It is worth noting that in the deterministic model, the value of the service rate is not allowed to exceed the sum of the initial queue and of the number of scheduled operations, whereas this is not an issue with the stochastic evolution function.

While the deterministic evolution is straightforward, we need to solve a system of differential equations to get the probability of each outcome at the end of the slot for the stochastic evolution. In fact, since we use an Erlang law to model the rate of service, each state corresponding to a given queue length is divided in $k$ virtual stages of works, and the completion of stages of work is a Poisson process of parameter $k\mu$. This means that there are not $N + 1$ but $kN + 1$ possible states for each of the two queues. The evolution of the probability of each state with time is given by the following system of differential equations:

$$\frac{dP_0(s)}{ds} = -\lambda P_0(s) + k\mu P_1(s)$$
$$\frac{dP_i(s)}{ds} = -(\lambda + k\mu)P_i(s) + k\mu P_{i+1}(s), \forall i \in \{1, \dots, k-1\}$$
$$\frac{dP_i(s)}{ds} = \lambda P_{i-k}(s) - (\lambda + k\mu)P_i(s) + k\mu P_{i+1}(s), \forall i \in \{k, \dots, (N-1)k\}$$
$$\frac{dP_i(s)}{ds} = \lambda P_{i-k}(s) - k\mu P_i(s) + k\mu P_{i+1}(s), \forall i \in \{(N-1)k+1, \dots, Nk-1\}$$
$$\frac{dP_{kN}(s)}{ds} = \lambda P_{k(N-1)}(s) - k\mu P_N(s)$$

We solved this system for all values of $\lambda \in \{0, \dots, 30\}$ and $\mu \in \{0, \dots, 20\}$ and all the initial conditions that represent known initial queue length, that is, for all $n_0 \in 0, \dots, N$:

$$\begin{cases} P_i(0) = 1 \text{ if } i = kn_0 \\ P_i(0) = 0 \text{ if } i \in \{0, \dots, kN\} \backslash \{kn_0\} \end{cases}$$

"Solving" means that we computed the value of all state probabilities at a time equal to the length of a slot, in order to have the probability of each final queue length as a function of the initial queue length and the demand and service rates during the slot. If we denote by $T$ the duration of a slot, the probability of having a queue length of $n$ at the end of a slot is:

$$\begin{cases} P_0(T) \text{ if } n = 0 \\ \sum_{i=1}^{k} P_{(n-1)k+i}(T) \text{ if } n \in \{1, \dots, N\} \end{cases}$$

## 3.2 Comparison with actual data

An important question is to see how we can compare this model to what actually happened in the studied airports using all the data we retrieved. More specifically, we need empirical definitions of the arrival and departure queues. Indeed, we will be interested in minimizing the queue lengths, and this will also allow us to define the number of scheduled operations during a time slot as the number of aircraft joining the queue during this slot.

For the arrival queue, as we are able to retrieve an estimated and actual time for the landing event, we can decide that an arriving aircraft will be considered queueing between ELDT and ALDT, as illustrated by figure 3.1.

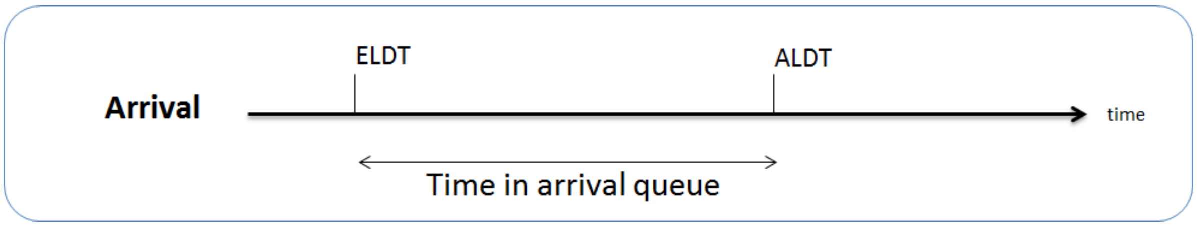


Figure 3.1: Arrival queue empiric definition

Departures are a bit more complicated to handle, as we do not have access to a scheduled time for the takeoff event, but only to the scheduled off-block time. Even is the aircraft leaves the block on schedule, there is an incompressible taxi-time during which it is not yet available for takeoff, so adding aircrafts in the departure queue at EOBT would contradict our model, because we considered that each aircraft in the queue can be served at any time.

As shown in figure 3.2, there are two ways to define our departure queue. The most empirical would be to consider aircraft that are actually queueing at the end of a runway, waiting for the permission to take-off. This means that they would be queuing from AOBT + actual taxi time until ATOT. However, we have not been able to deduce actual taxi-time from the data available to us. The solution we choose is to consider the minimal theoretical time at which each aircraft could be able to take-off, that is the EOBT plus the minimum (also called "unimpeded") taxi-time between its parking position and the runway end used for take-off.

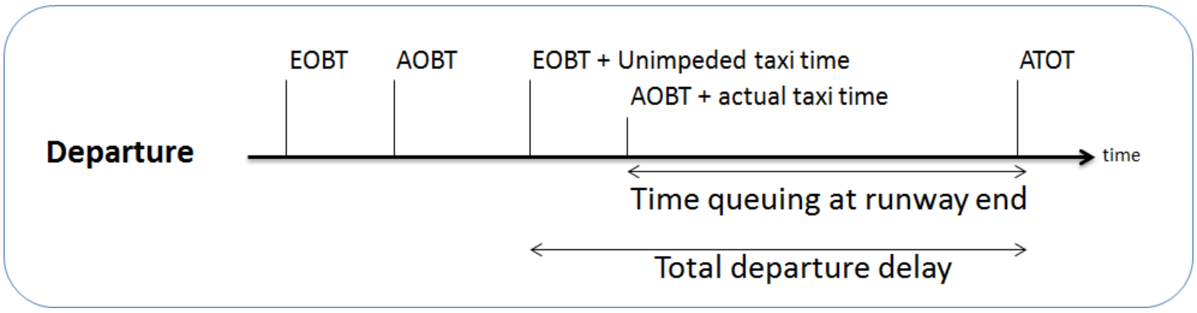


Figure 3.2: Possible departure queue empiric definitions

The computation of the unimpeded taxi-time is not trivial. As a first approach, we only computed a mean unimpeded taxi-time, independent of the parking area and the runway used by the aircraft.

We followed the method presented by Simaiakis [8]. The main idea is to plot the average time between AOBT and ATOT (taxi-time + waiting time at the runway end) as a function of the number of other aircrafts that used the departing runway system during the taxiing process. This number was called the adjusted traffic in the article. The resulting graph for one of the studied airports is shown in figure 3.3. When the adjusted traffic is low, we can consider that the plane was not delayed by interactions with other aircrafts on the ground and that it did not wait, so ATOT - AOBT will be a relevant value in the evaluation of the unimpeded taxi-time. We cannot only keep flights for which the adjusted traffic is 0, as this would introduce a bias toward airplanes that move at higher speeds on the ground. Indeed, all other parameters being equal, an aircraft that moves faster than another one will have a smaller adjusted traffic, because less flights will leave their block while it is moving toward its departing runway.

In the example of figure 3.3, we would keep all flights with an adjusted traffic smaller than 9, after witch the value of ATOT - AOBT starts to significantly increase. The drop of ATOT - AOBT at the very beginning seems to contradict what is explained in the article about the bias toward faster aircrafts. A possible reason for that is that aircraft taxi a bit more slowly at night, which is also the period during which there is the least traffic.

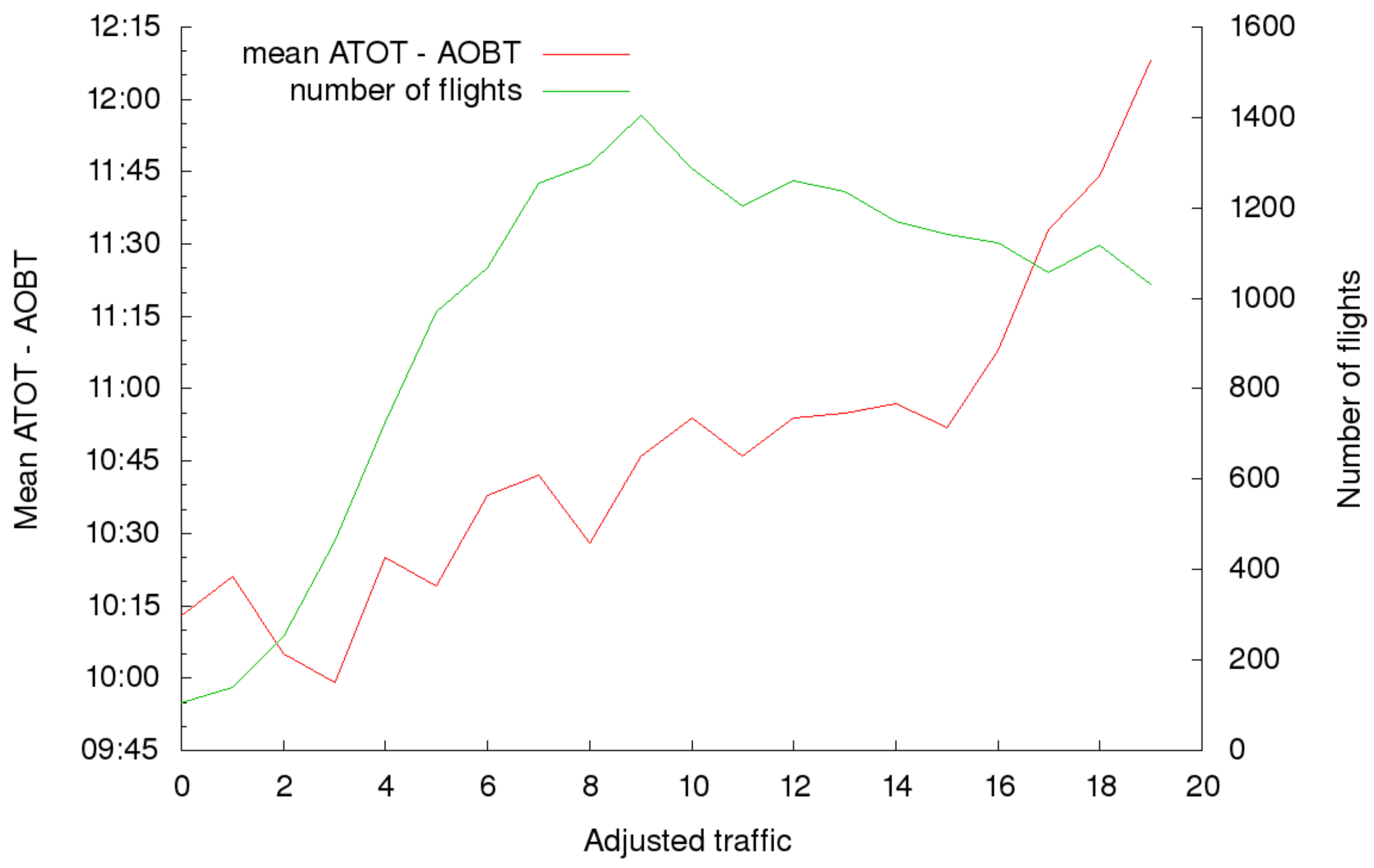


Figure 3.3: Mean ATOT - AOBT as a function of adjusted traffic

## 3.3 Runway configuration changes and traffic interruptions

The impact of the runway configuration on the airport capacity can be taken into account easily by using a different RCCE for each configuration, for the airports where it is possible to determine it. However, the action of changing the runway configuration may induce a short unaivalability of the runway system, because some aircrafts might need to be re-routed to another runway endpoint before they can take-off or land. This period may differ for departures and arrivals, and would also depend on the considered airport, the nature of the runway configuration change (addition of a runway, inversion of runway orientation, . . . ), the degree of anticipation of the change (all aircraft may already have been re-routed when the runway configuration is modified), and probably other factors.

To determine whether this unaivalability period exists at all, and if it is realistic to model it as having a constant length, we identified the moments at which the runway configuration was modified during busy periods of operation (between 8am and 8pm), and computed some statistics on the durations separating the last departure (resp. arrivals) in the old configuration and the first departure (resp. arrival) in the new configuration. Then, we compared them with the durations separating 500 consecutive departures and arrivals chosen randomly (but still between 8am and 8pm). The result is presented in figure 3.4, where the number in parenthesis corresponds to the randomly chosen operations, and the other to the gaps at runway configuration changes.

This clearly shows that the gaps tend to be higher at runway configuration changes. However, it is hard to determine if controllers find it convenient to use already existing gaps in the schedule to insert runway configuration changes because this impacts fewer aircrafts, or if the gaps were actually created by the runway configurations changes. Moreover, the high standard deviation in the gaps lengths leads to believe that using a constant value would be an over-simplification.

| Airport | Operation | Shortest gap | Longest gap | Mean gap | Median gap | Std deviation |
|---|---|---|---|---|---|---|
| Airport 1 | Arrivals | 40” (0”) | 26’42” (24’39”) | 8’13” (2’03”) | 7’55” (1’27”) | 5’17” |
| | Departures | 1’50” (0”) | 24’47” (18’48”) | 6’39” (2’00”) | 5’18” (1’24”) | 4’47” |
| Airport 2 | Arrivals | 2’06” (1”) | 22’02 (35’08”) | 10’14” (3’08”) | 9’06” (2’22”) | 5’19” |
| | Departures | 3’30” (11”) | 13’30” (20’42”) | 8’03” (2’54”) | 6’49” (2’25”) | 3’33” |
| Airport 3 | Arrivals | 4” (1”) | 28’34” (15’03”) | 10’18” (2’24”) | 9’52” (1’58”) | 5’55” |
| | Departures | 2’52” (0”) | 13’39” (20’14”) | 7’07” (2’22”) | 5’50” (1’55”) | 3’15” |
| Airport 4 | Arrivals | 16” (0”) | 20’10” (8’48”) | 9’06” (1’49”) | 9’00” (1’27”) | 4’11” |
| | Departures | 2’18” (1”) | 8’54” (9’32”) | 4’38” (1’53”) | 4’35” (1’30”) | 1’31” |
| Airport 6 | Arrivals | 23” (0”) | 10’58” (28’13”) | 3’06” (1’57”) | 2’42” (1’31”) | 2’24” |
| | Departures | 55” (0”) | 36’28” (11’26”) | 8’45” (1’42”) | 7’46” (1’23”) | 6’19” |
| Airport 7 | Arrivals | 2’06” (1”) | 22’02” (35’08”) | 10’14” (3’08”) | 9’06” (2’22”) | 5’19” |
| | Departures | 3’30” (11”) | 13’30” (20’42”) | 8’03” (2’54”) | 6’49” (2’25”) | 3’33” |

Figure 3.4: Statistics on the gaps at runway configuration changes

Another difficulty that arises if we want an algorithm that chooses the best runway configuration for each time slot is that the set of allowed runway configurations depends on the meteo. Therefore, we would need a model of the weather throughout the day to account for this. There have been attempts in this direction where the weather conditions evolution was modelled as a Markov chain [5], but we were not able to find any evaluation on their relevance. Moreover, this causes a dramatic increase of the state space size (both visibility and wind need to be part of the model), which has an unavoidable impact on the computation time of any algorithm tackling the problem.

Finally, as European airports have less runways that the largest American airports, the best ordering of runway configurations throughout the day is often already clear for the controller (with three runways, you can basically decide to favor arrivals or departures according to the schedule, and with two parrallel runways, you just need to choose the direction so that the wind constraints are satisfied)

For these reasons, we decided to solve a slightly different problem. Even if a controller decide to operate a runway configuration change, the question of the optimal timing for the change will still arise. We propose to let the controller input a timeframe during which the change must occur (because of traffic or meteorological constraint), along with the time during which the runway system will not be able to operate because of the change (if any), and we will try to take optimal decisions with this additionnal constraint.

Since this model actually allows to schedule any kind of traffic interruption, and not only those potentially caused by the configuration changes (for instance, the deicing of the runways), and since the gain of not having to model the meteorological state will allow us to choose a finer resolution for the time slots, we might have to deal with interruptions longer than a slot. As described in the next section, we created a modified version of the algorithm to take this into account.

# Chapter 4

# Algorithms

We have a mathematical model for the departure and arrival processes that we would like to use to take decisions regarding the choice of the arrival and departure rates, and the scheduling of traffic interruptions, each potentially resulting in an update of the runway configuration.

Our algorithm will be provided with a schedule of operations for a long period of time, for instance, a whole day, and a list of the interruptions to schedule during this period. For each interruption, the duration and the timeframe during which it can begin is specified. Because the evolution of the airport state is not deterministic (even if we can use a deterministic model to compute our solution), it is not suitable that the algorithm returns a fixed schedule for interruptions and operationnal rates. Indeed, for some particular realization of the airport operations, these decisions might be way off the optimal choices. Instead, the algorithm returns a decision matrix that, for each slot and each initial conditions associated to the slot, gives the best decision that can be taken in this situation.

We define the following notations:

- $E_s$ is the set of all time slots (if we plan a day with time slots of 15 minutes, $E_s = \{1, 2, \ldots, 96\}$)
- for $s \in E_s$, $RCCE(s)$ is the set of Pareto points of the RCCE associated with slot $s$, depending on the weather condition at this slot
- for $s \in E_s$, $schArr(s)$ is the scheduled number of arrivals at slot $s$, and $schDep(s)$ is the scheduled number of departures at slot $s$
- As in section 3.1, $N$ is the maximal length of the queues handled by the stochastic model
- $E_q = \{0, 1, \ldots, N\}$
- $E_r = \{0, 1, \ldots, 20\}$ is the set of rates for a single operation for which we solved the system of differential equations of chapter 3
- $E_{scheduled} = \{0, 1, \ldots, 20\}$ is the set of scheduled numbers of operations for which we solved the system of differential equations of chapter 3
- $N_{op}$ is the total number of operations scheduled during the day, typically of the order of 1000
- $S$ is the state space: $S = E_s \times E_q \times E_q$
- $D$ is the decision space: $D = E_r \times E_r$
- a strategy is a function from $S$ to $D$
- $T$ is the transition matrix computed by solving the system of differential equation of chapter 3. Under the stochastic framework, for all $(q, q') \in E_q \times E_q$, $r \in E_r$ and $sch \in E_{scheduled}$, $T[q][r][sch][q']$ is the probability to have a queue $q'$ qt the end of a slot for which the initial queue is $q$, the chosen rate for the operation is $r$ and the scheduled number of operations is $sch$ (this works for both arrivals and departures).

The states and decisions definitions need to be clarified: a state is a triplet containing a slot, along with the arrival queues and departure queues at the beginning of the slot, and a decision contains recommended arrival and departure rates.

In addition to the perturbation caused by the stochasticity of the model, the schedule itself may be modified during the day, for instance if some flights are cancelled. As the schedule cannot be included in the state space (the number of possible scheduling is of order $O(|E_s|^{N_{op}})$), the strategy will be to re-run the algorithm at the beginning of each time slot to take into account the schedule update. For this, we need the execution time to be short in comparison with the duration of a time slot.

Our programm is, as in [8], a dynamic programming algorithm which find the decisions that minimizes a given cost function.

## 4.1 Without interruptions

### 4.1.1 Deterministic framework

In order to check whether the impact of incertitude is worth being taken into account, we observed what would happen in a deterministic framework, where all flights become available for takeoff or landing at the exact time the operation is scheduled and where the number of operation that take place during a time slot is equal to the rate decided by the controller at the beginning of this slot.

The problem we want to solve is to find which departure and arrival rates minimize the total cost over a given set of slots, under the constraint that the departure and arrival rates must be in the RCCE at each slot. We choosed to solve it using a dynamic programming algorithm.

To define our cost function, we first introduce the notion of cost for a single state. For a state $(s, iaq, idq) \in S$:

$$stateCost(s, iaq, idq) = \alpha iaq^2 + (1-\alpha) idq^2$$

$\alpha \in [0;1]$ is a parameter that reflects the relative costs of the arrival and departure queue. As making an incoming aircraft wait in the air is more ressource-consuming than delaying an aircraft on the ground, it is very common to take $\alpha > 0.5$.

Once we have chosen a strategy $strat$, and as the framework is deterministic, the state that will be encountered at each slots if we follow the strategy is fully determined under given initial conditions. We will assume that both queue are initially empty, and denote by $deterministicState_{strat}(s)$ the state that will be encountered at slot $s$ for the strategy we want to evaluate.

The value of the cost function associated to the strategy is:

$$TotalCost(strat) = \sum_{s \in E_s} stateCost(deterministicState_{strat}(s))$$

The algorithm will fill a 3-dimensional matrix $M$. What we want to have at the end of the execution is:

$$\forall (s, iaq, idq) \in S, M[s][iaq]idq] = (c, ar, dr)$$

where $s$ corresponds to a slot, $iaq$ to a potential length of the arrival queue at the beginning of this slot, $idq$ to a potential length of the departure queue at the beginning of this slot. $(c, dr, ar)$ is the triplet which last two elements are the departure and arrival rates that should be chosen at this slot and which first element $c$ is the value of the remaining cost if we follow the optimal policy from this state on.

M can be computed from the last to the first slot. First, in order to ensure that the queues will be empty at the end of the time period, we introduce an additionnal slot at the end, and set:

$$M[|S|+1][0][0] = (0,0,0)$$
$$\forall (iaq, idq) \neq (0,0) M[|S|+1][iaq][idq] = (\infty, 0, 0)$$

Now we consider $s \in E_s$ and suppose that $M[s+1][.][.]$, $M[s+2][.][.]$, $\ldots$, $M[|S|+1][.][.]$ are already filled. We want to fill $M[s][.][.]$.

For each $(s, iaq, idq) \in S$ and each decision $d = (arrRate, depRate) \in RCCE(s)$, we compute the cost of taking decision $d$ in state $(s, iaq, idq)$ as:

$$cost(s, d) = slotCost(iaq, idq) + M[s+1][iaq + schArr(s) - arrRate][idq + schDep(s) - depRate][0]$$

Then, we select one of the decisions that minimize $cost(s, d)$ and put this cost along with the decision in $M[s][arrq][depq]$. There might be several decisions that minimizes the expression. This means that there are several optimal strategies, and we can pick any of them to fill $M$.

Once $M$ is completely filled, it is easy to compute an optimal strategy. If we assume that both queues are initially empty, the optimal arrival and departure rates for the first slot are the two last values in $M[1][0][0]$. If we apply these rates, and as we are in a deterministic framework, we know what the queue lengths will be at the end of slot 1, so we can look at the matching position of $M[2]$ which rate should be applied at slot 2, and so on until the end of the day.

This algorithm was implemented and runs almost instantaneously if we ask it to compute a strategy for a whole day (96 slots of 15 minutes). It is therefore perfectly reasonable to run it at the beginning of each slot in order to update the strategy.

### 4.1.2 Stochastic framework

The strategy is similar to the one used for the deterministic framework, with some modifications.

The definition of the global cost function needs to be updated, because we can no longer deduce the list of states that will be encountered from the strategy that we want to evaluate. If we consider a strategy *strat*, then for each slot $s$, $state_{strat}(s)$ is the random variable reflecting the state we will encounter at slot $s$ if we follow the strategy *strat*. The cost function is now:

$$TotalCost(strat) = \mathbb{E}(\sum_{s \in E_s} stateCost(state_{strat}(s))$$

The meaning of the data stored in matrix $M$ is almost the same: at the beginning of a slot $s$, we will observe the lengths of the arrival and departure queue $iaq$ and $idq$ and look for what to do in $M[s][iaq][idq]$. The difference is the cost stored in the first item of a matrix element which is updated according to the new cost function definition: it is now the expectation of the sum of the *stateCost* function over all slots after and including $s$ assuming we follow the computed strategy.

We also define cost values for $M[|S|+1]$, but we cannot afford to put an infinite cost somewhere, because we can never guarantee that a given state will not be reached (so all costs in $M$ would be infinite if we did that). Therefore, we just penalize the final queue as we would penalize the queues in an ordinary slot:

$$\forall (arrq, depq) \in E_q \times E_q, M[|S|+1][arrq][depq] = (stateCost(1, arrq, depq), 0, 0)$$

For each initial state $(s, arrq, depq) \in S$ and each decision $d = (arrRate, depRate) \in RCCE(s)$, the probability to reach state $newState = (s+1, arrq', depq')$ at next slot is given by:

$$p(newState) = T[arrq][arrRate][schArr][arrq'] \times T[depq][depRate][schDep][depq']$$

Then the contribution of the state $newState$ to the cost of taking decision $d$ is:

$$f(newState) = p(newState)M[slot+1][arrq'][depq'][0]$$

And finally, the total remaining cost that we expect if we take decision $d$ is:

$$remainingCost(d) = stateCost(s, arrq, depq) + \sum_{(s+1, newArrQ, newDepQ) \in S} f(s+1, newArrQ, newDepQ)$$

Then, we select one decision that minimizes the expected cost and put this cost along with the decision in $M[s][arrq][depq]$

If M is computed before the day begins, it gives a strategy that minimizes the cost function over the day: at the beginning of any slot $s$, if the arrival and departure queue lengths are equal to $iaq$ and $idq$, then fixing the arrival and departure rates to $M[s][iaq][idq][1]$ and $M[s][iaq][idq][2]$ is an optimal decision to minimize the cost function.

To get the cost of a given decision, we obviously need to loop over all possible new arrival queue lengths and all possible departure queue lengths to compute the sum of the contribution of each possible new state. To get a lower computation time, there is a way to ignore some states that have a probability lower than $10^{-6}$.

Indeed, while looping over the arrival or departure queue length in increasing order, we will encounter up to three distinct phases:

- the phase where we are exploring new queue lenghts that are very unlikely, because they are too low
- the phase where we are exploring plausible values of the new queue lenght
- the phase where we are exploring new queue lenghts that are very unlikely, because they are too high

It is easy to skip the first and third phase when they exist, by doing no computation until the value in the transition matrix reaches $10^{-6}$, and by breaking out of the loop as soon as it has exceeded $10^{-6}$ at least once and goes below it again. This way, we avoid most of the iterations of the loop while loosing minimal precision, and get an running times of approximately 5 seconds on a recent computer to handle 96 slots.

We can no longer access predictions of the queue lengths at every time slot as in the deterministic framework. However, it is possible to compute the expected queue length at each slot (assuming that the strategy proposed by the algorithm is followed), just by assuming that both queues are initially empty (this is not really a simplification, as most airports have several hours with none or very few operations during the night) and by computing the probability of each queue length at the beginning of each slot in chronological order.

## 4.2 Interruptions handling

The solver was modified in order to be able to schedule interruptions optimally. For this, each decision must reflect our choice to start or not an interruption, and to take this decision, we need more information than only the length of the queues that are contained in the state: if there is an interruption to schedule during a certain period, we need to know whether it has not started yet, has already started, or is already over to take a decision.

First, we add a list of interruptions to the input of the solver. An interruption is specified by a timeframe during which it is allowed to begin and by its duration.

For this, we define the notion of "state" of an interruption at a certain slot.

The length in slots of interruption $I$, that we will denote by $n_I$ is given by:

$$n = \lceil \frac{duration(I)}{slotDuration} \rceil$$

The state of interruption $I$ at slot $s$ is then defined as:

- 0 if $I$ starts after $s$
- 1 if $I$ starts at the beginning of $s$
- 2 if $I$ started at the beginning of the slot before $s$

- ...
- $n_I$ if $I$ started n-1 slots before $s$
- $n_I + 1$ otherwise (even if $I$ started more than $n_I$ slots before $s$)

Depending on when the interruption $I$ is finally scheduled, the state of $I$ at slot $s$ may take different values. We denote by $allowedStates(I, S)$ the set of states that $I$ may have at slot $s$.

We do not allow the input to contain interruptions that may overlap, that is two interruptions such that there exists a legal scheduling and at least one slot such that the two interruptions will be ongoing during the slot. With this restriction, it is impossible to find a slot $s$, and two interruptions $I_1$, $I_2$ such that $allowedStates(I_1, S)$ and $allowedStates(I_2, S)$ both contain more than one state. This means that we can encode all the information regarding interruptions in the state space by adding a single dimension containing an interruption state: for slots for which there is an interruption that has several allowed states, we will store its state in the state space by using this additionnal dimension.

We define $N_i$ as:

$$N_i = \max(\bigcup_{I,S} allowedState(I, S))$$

The new state space $S'$ is:

$$S' = S \times \{0, 1, \ldots, N_i\}$$

The last element of $s$ correspond to the state of the interruption associated with the previous slot, at the previous slot. If this interruption is still ongoing during the current slot, we will either force the departure and arrival rate to be 0 if the interruption ends after the slot, or modifiy the RCCE for this slot by applying an homothetic transformation which factor depends on the percentage of the slot covered by the remaining of the interruption.

The new decision space $D'$ is:

$$D' = D \times \{0, 1, \ldots, N_i\}$$

The state that we put in the decision must be coherent with the previous state and belong to $allowedState(I, S)$. In fact, the only case where we have a choice for the new state is when the previous state was 0, in which case the new state may be either 0 (wait) or 1 (start the interruption now). In all other cases, the new state is just $\min(n_I, previousState + 1)$.

With these new cost function adn state and decision spaces, the algorithm runs exactly as described in the previous part.

## 4.3 Handling fixed rates constraints

If an aircraft controller decides to use a program to assist him in the decision of the operational rates, it is very likely that he will want to be able to fix the arrival or departure rate on some slots, and optimize the other choices with this constraint. One advantage of this algorithm is that it can easily find the optimal solution under such constraints. In fact, this reduces the number of cases that have to be examined for the slot where there is a constraint. This was implemented as follows:

1. if both the arrival and departure rates are fixed at a given slot, we put these rates in the decision matrix, compute the associated cost, and do not look at other possible rates
2. if only the arrival rate is specified, we find the point of the RCCE with this arrival rate and the highest departure rate and proceed as in 1 with this point. If there are no points with this arrival rate in the RCCE, we fix the departure rate to 0
3. same process if only the departure rate is fixed

4. if an interruption is occuring during a slot where the rate is specified, the rate is still multiplied by a factor proportionnal to the proportion of the slot not covered by the interruption

## 4.4 Choice of arrival queue weight in the cost function

### 4.4.1 Deterministic model

For the deterministic model, the choice of the parameter $\alpha$ which determines the weight of the arrival queue in the cost function has a high impact on the decisions suggested by our algorithm. Indeed, both queues are empty most of the time in the deterministic framework, so the value of $\alpha$ determines which of the two operations will have to be delayed during peak hours. As an example, we plotted the queue lengths obtained with the deterministic dynamic programming algorithm for $\alpha = 0.1$ (figure 4.1) and $\alpha = 0.9$ (figure 4.2) with the data of airport 3 on the 22nd of September 2014:

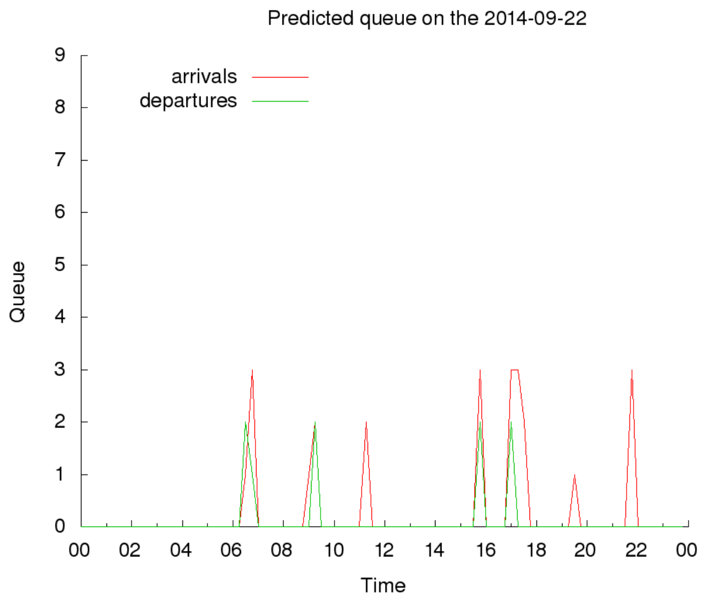


Figure 4.1: Queue length predicted by our algorithm with $\alpha = 0.1$

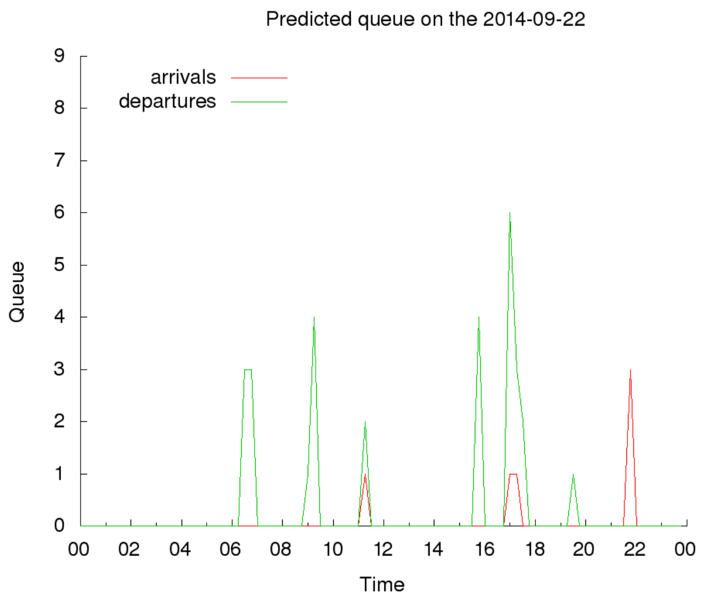
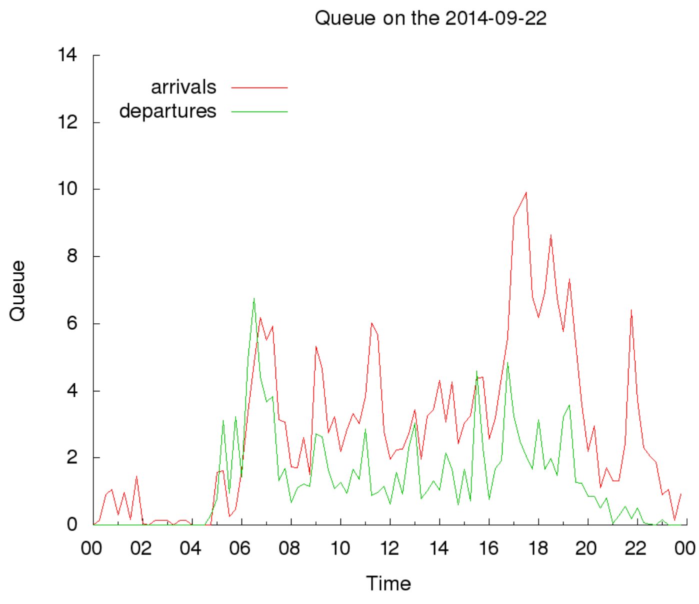


Figure 4.2: Queue length predicted by our algorithm with $\alpha = 0.9$

### 4.4.2 Stochastic model

The choice of $\alpha$ also matter. For the same operational day, we also plotted the two graphs obtained with $alpha = 0.15$ and $alpha = 0.9$:



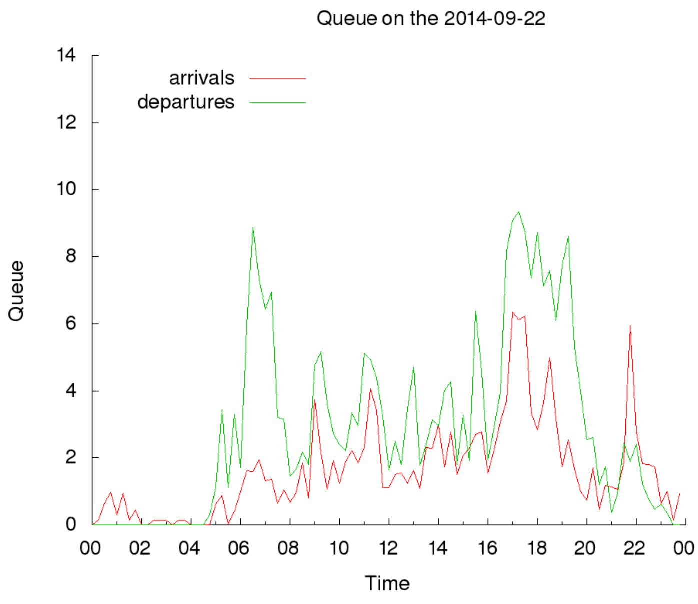
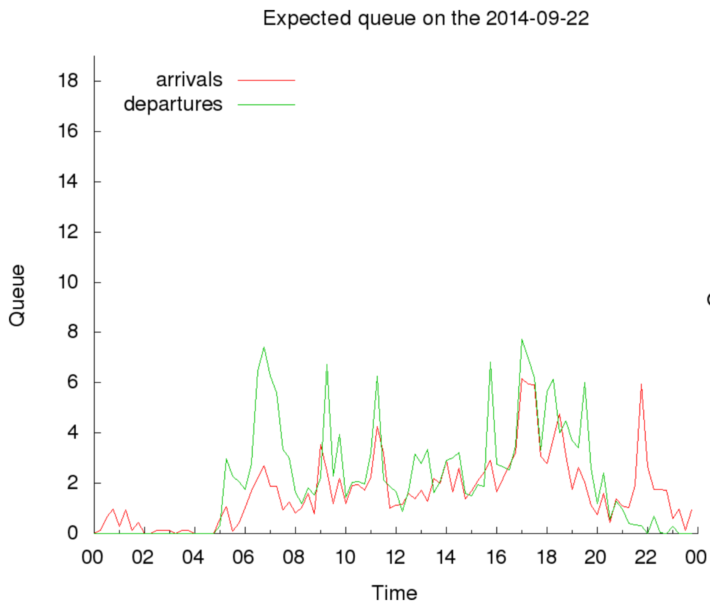


Figure 4.3: Expected queue length with $\alpha = 0.1$

Figure 4.4: Expected queue length with $\alpha = 0.9$

We do not plot these graphs for $\alpha = 0$ or $\alpha = 1$. For these value, the algorithm completely ignores one of the queue, so it always outputs the maximal possible rate for the other queue, along with the associated rate for the queue that has no weight, which will be very low. The queue for the operation that is ignored by the optimizer rapidly reaches the maximal lengths that we can handle (N = 30) and the result has therefore no meaning.

The value of $\alpha$ that seems to minimize the absolute differences between the queues predicted by the stochastic model and the actual queues is 0.8. For instance, during the same day as above, we plotted the expected queue length with $\alpha = 0.8$ (figure 4.5) and the actual queue length (figure 4.6).



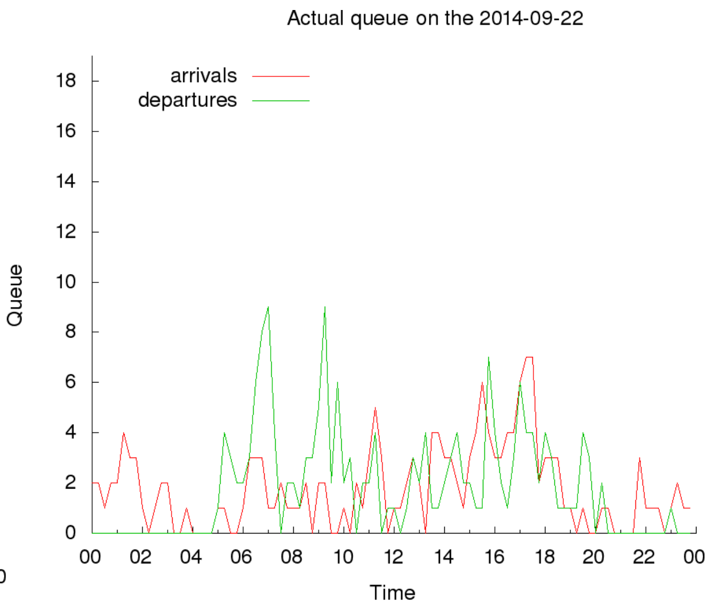


Figure 4.5: Expected queue length with $\alpha = 0.8$

Figure 4.6: Actual queue length

## 4.5 Effect of meteorological conditions

The following step was to take into account the flight rules that are applied during each slot. RCCE for airport 1 in both VFR (Visual Flight Rules) and IFR (Instrumental Flight Rules) are given bellow (figure 4.7) as an illustration of the difference that the meteorological conditions can make. As VMC is by far the most frequent weather category that we encountered in our data set, the RCCE plotted for the three other categories probably gives an underestimation of the achievable rates.

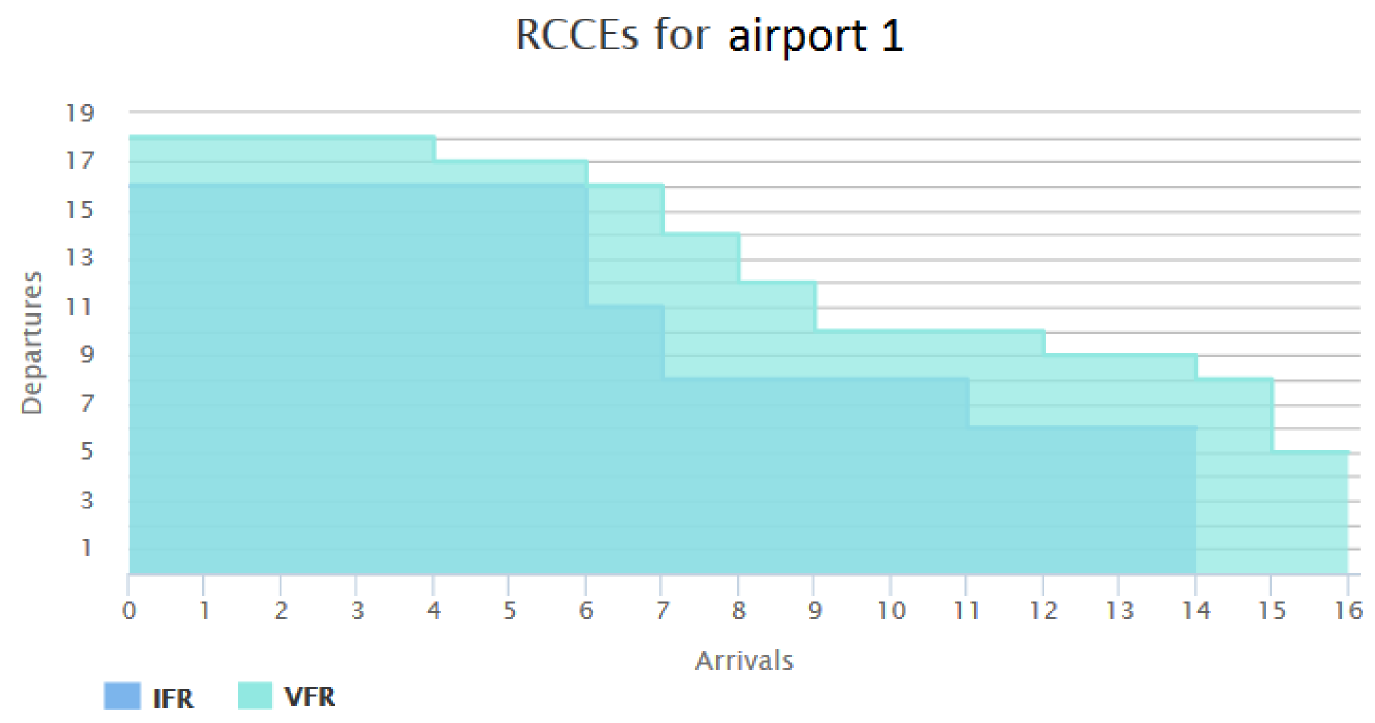
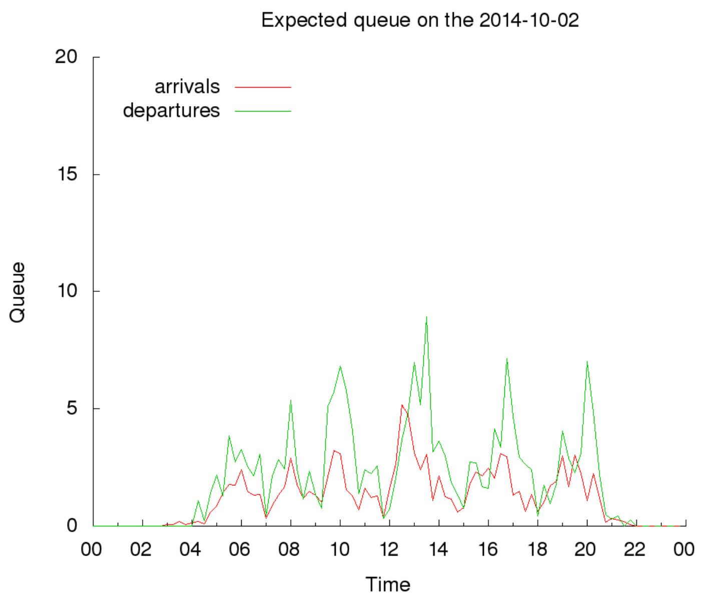


Figure 4.7: RCCE of airport 1 in VFR and IFR conditions

For instance, here are the expected and actual queue lengths at airport 1 on the 2nd of October 2014, obtained by solving and applying the stochastic model without taking the weather categories into account (all slots were considered as VFR):



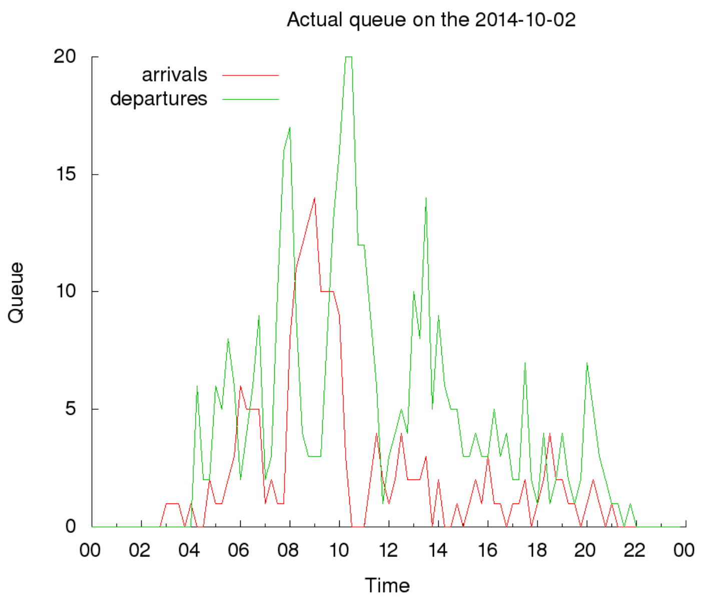


Figure 4.8: Expected queue on a day with a bad weather period

Figure 4.9: Actual queue on a day with a bad weather period

A look at the meteorological data confirmed that IFR were applied at airport 1 between 0:45am and 11:00am, which is the moment from which the actual departure and arrival queue start to recover. This confirm the importance of taking at least the flight rule into account in our model. If we switch to the IFR RCCE during the period of bad weather while applying our model, we get the following result:

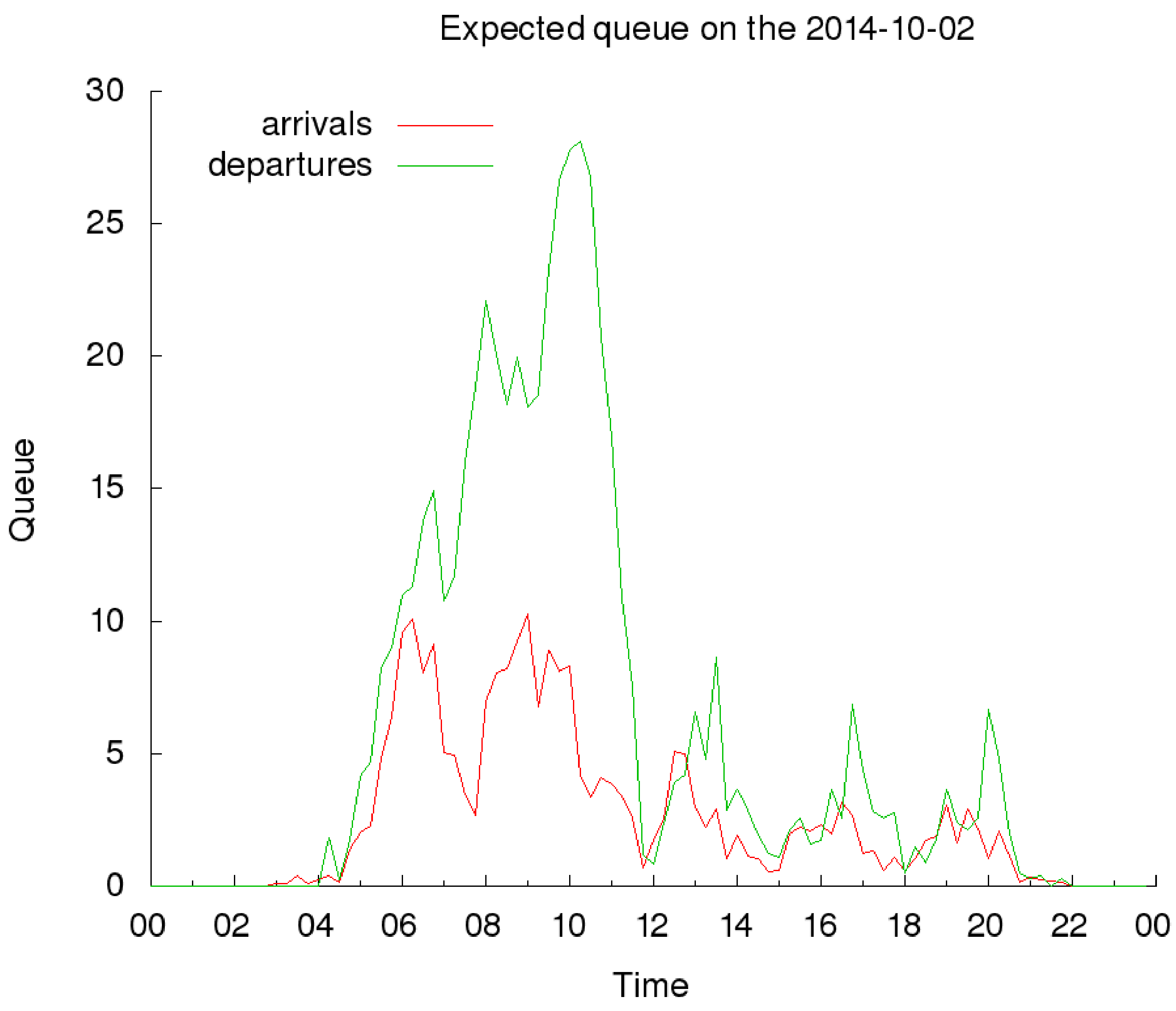


Figure 4.10: Expected queue with IMC RCCE during bad weather period

The predicted departure queue length at the peak exceeds the actual maximal queue length. There are two likely explanations for this:

- we lack IFR data to build a reliable RCCE, so we do not allow some couples of arrival and departure rates that are in fact achievable
- when the departure queue becomes really high, the airport controllers start favoring departures a little more. This would explain why the actual arrival queue at the peak is higher in the actual data than in our model

## 4.6 Simulations

Predicting queue lengths at each time slot is already interesting, but providing an estimation of the delay that should be expected would be a better indicator of the upcoming congestion. The issue is that the notion of delay is not well captured by our stochastic model: we don't know which aircrafts are in the queue at each step, and in fact, a specific aircraft has no representations in the model which is macroscopic.

A possible solution is to simulate the whole day of operations according to the model rules, by generating aircrafts as they enter the queue and keeping track of them until they leave it. The simulation is performed as follow:

1. Initialize two empty queues for departures and arrivals, and set the current slot to 1
2. Generate demands for departures and arrivals for the current slot according to two Poisson laws which parameters are the number of scheduled operations during the slot (therefore, each incoming demand will have a precise timing of arrival in the queue)
3. Deliver services according to two Erlang laws of parameter $k = 4$ and with an expectation equal to the rate indicated by the optimal strategy that we computed for the slot and initial queues that we are encoutering.
4. Increment the current slot and go back to step 2

To generate the service according to an Erlang law, we just need to generate ”stages of works” according to a Poisson law which parameter is $k$ time the expectation of the given Erlang law. Each of the aircrafts that we put in a queue has a counter of the remaining stages of work that need to be carried out before it can be considered as served and leave its queue. As the Poisson processes are memoryless, the fact that we allow the generation of stages of work even when there are no aircrafts in the queue introduces no bias in the simulation.

The mean delay for departures or arrivals at a specific slot can then be seen as the average of the time that each demand generated during the slot stayed in the queue.

The variance in the simulation output is very significant: figure 4.11 shows the results of 4 simulations performed with the same data and strategy for airport 1 on the 22/09/2014. Therefore, we decided to perform 1000 simulations for each operational days, after which we could observe that the queue mean of the queue lengths encounterered in the simulations matched almost perfectly the expected queue lengths computed directly from the model (see figures 4.12 and 4.13).

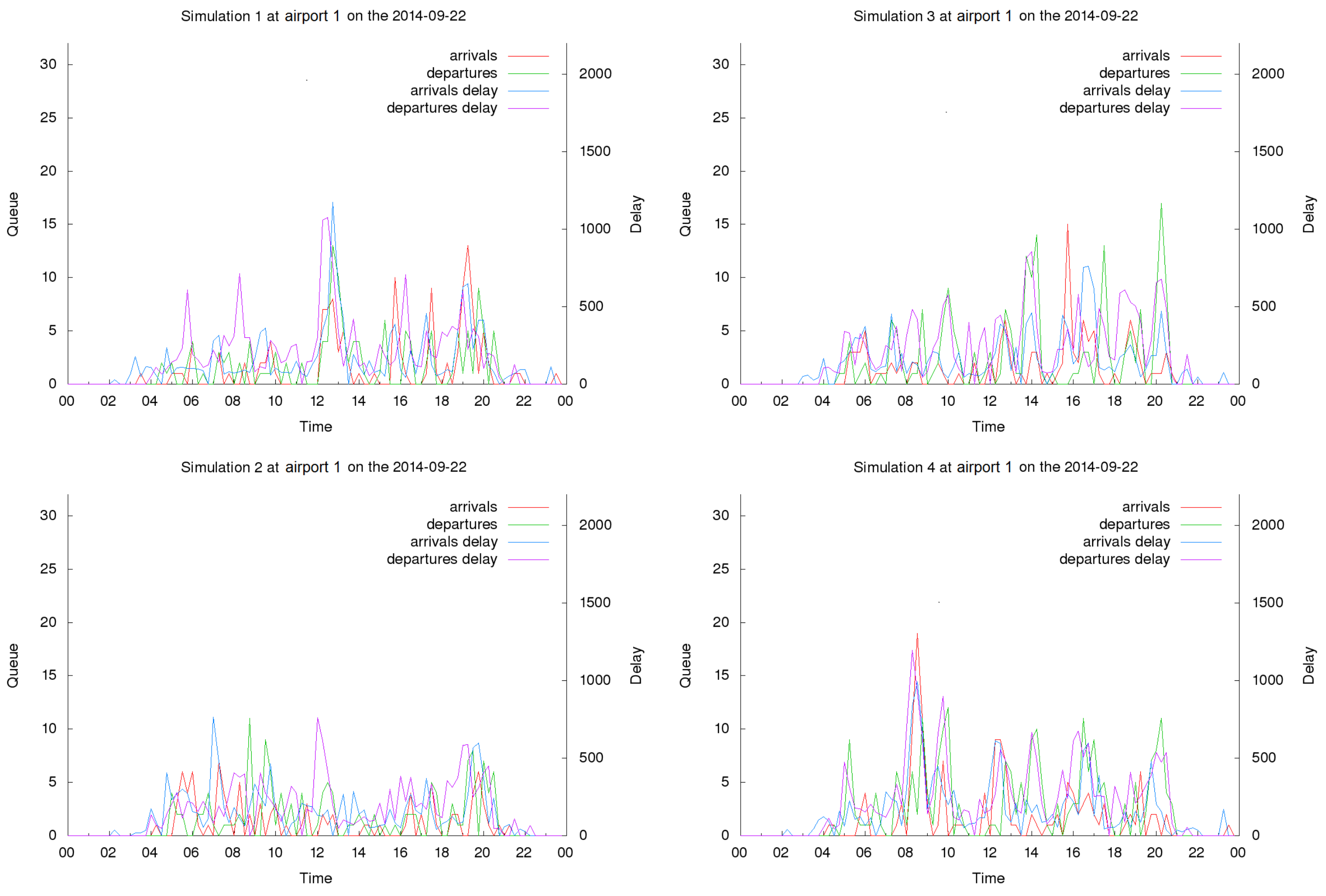


Figure 4.11: Examples of simulation outputs

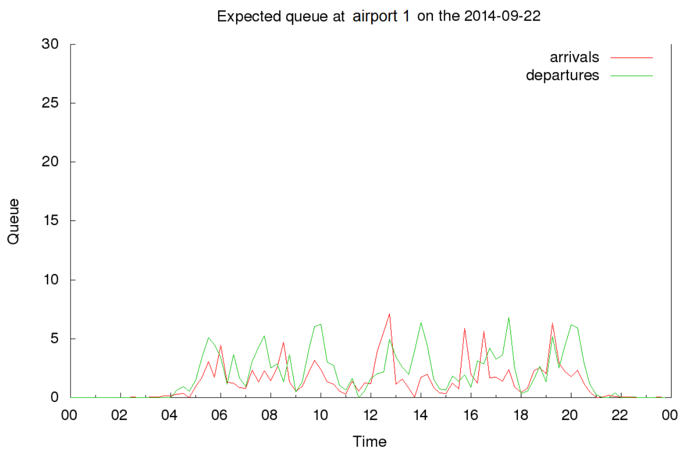


Figure 4.12: Expected queue lengths

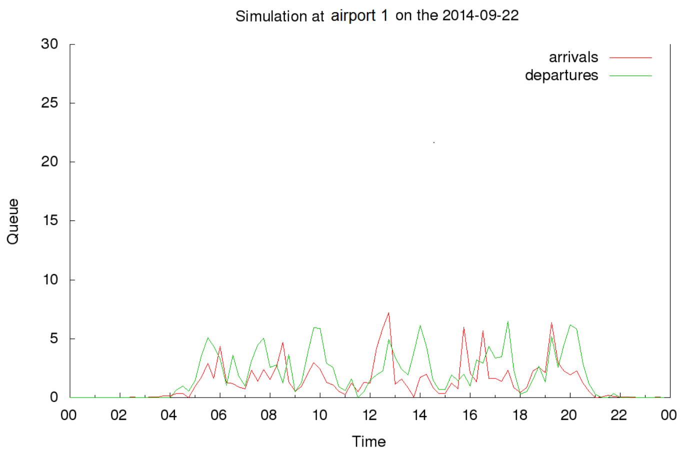


Figure 4.13: Simulations average queue lengths

# Chapter 5

# Evaluation and comparison of the two models outputs

Though we would like to evaluate the strength of the strategy proposed by the two models, it is almost impossible to compare them with the current strategies used by the airport controllers. Indeed, we cannot test our models in real conditions, and there are no widely accepted models of the airport controllers actions to which we could confront our algorithms. Moreover, the two algorithms return by design what our framework perceives as optimal strategies, so we would need to use another model of airport operations to compare them to any other one.

There are however two weaker forms of evaluation that we were able to perform:

- assessing the predictive value of the models: if we assume that both the strategies proposed by our models and the actual decisions of the airport controllers are not far from the optimal choices, then there is a point in comparing the expected queue lenghts and delay that we predict and there value throughout the actual day of operation. Similarities will show that our models are not far from the reality, and therefore give some confidence in the strategy they propose.
- computing an approximated value of the gain brought by the stochastic model by comparing the evaluation of the cost function on the stochastic and deterministic strategies

## 5.1 Model predictions

For all days of operations on which it was run (all the day of Octobers on 5 of the 7 airports), the deterministic framework predicted queue length that were way bellow the actual values (see 4.1 or 4.2 and figure 4.6). In fact, depending on the value of parameter $\alpha$, one of the two queue is kept empty most of the time, while the other remains very low. Because of this result, we did not implement an algorithm to assess delays in the deterministic framework (assuming that the service is delivered uniformally during a time slot, it would be quite easy to have a prediction of the delay of each aircraft in the deterministic model) and directly concluded that the predictive value of the model is low.

The stochastic model performs better. For instance, we computed the average of the mean squared error for each day of October in airport 1 for its queue lenghts predictions. We can see in figure 5.1 that it stays relatively low (we lacked data for some day, for which the error is displayed as being 0, the are actually no days for which we have a perfect match). More visually, we can compare figures 4.5 and 4.6 to see that the expected queue lengths are not far from the actual values. Generally, the error is an under-prediction of the queue value. We identified 3 possible reasons for this:

- The queing model does not perfectly reflect the mechanic of the runway congestion
- There are other sources of delays than runway saturation in the airport
- The model might actually take better decisions than the airport controllers, leading to shorter queues

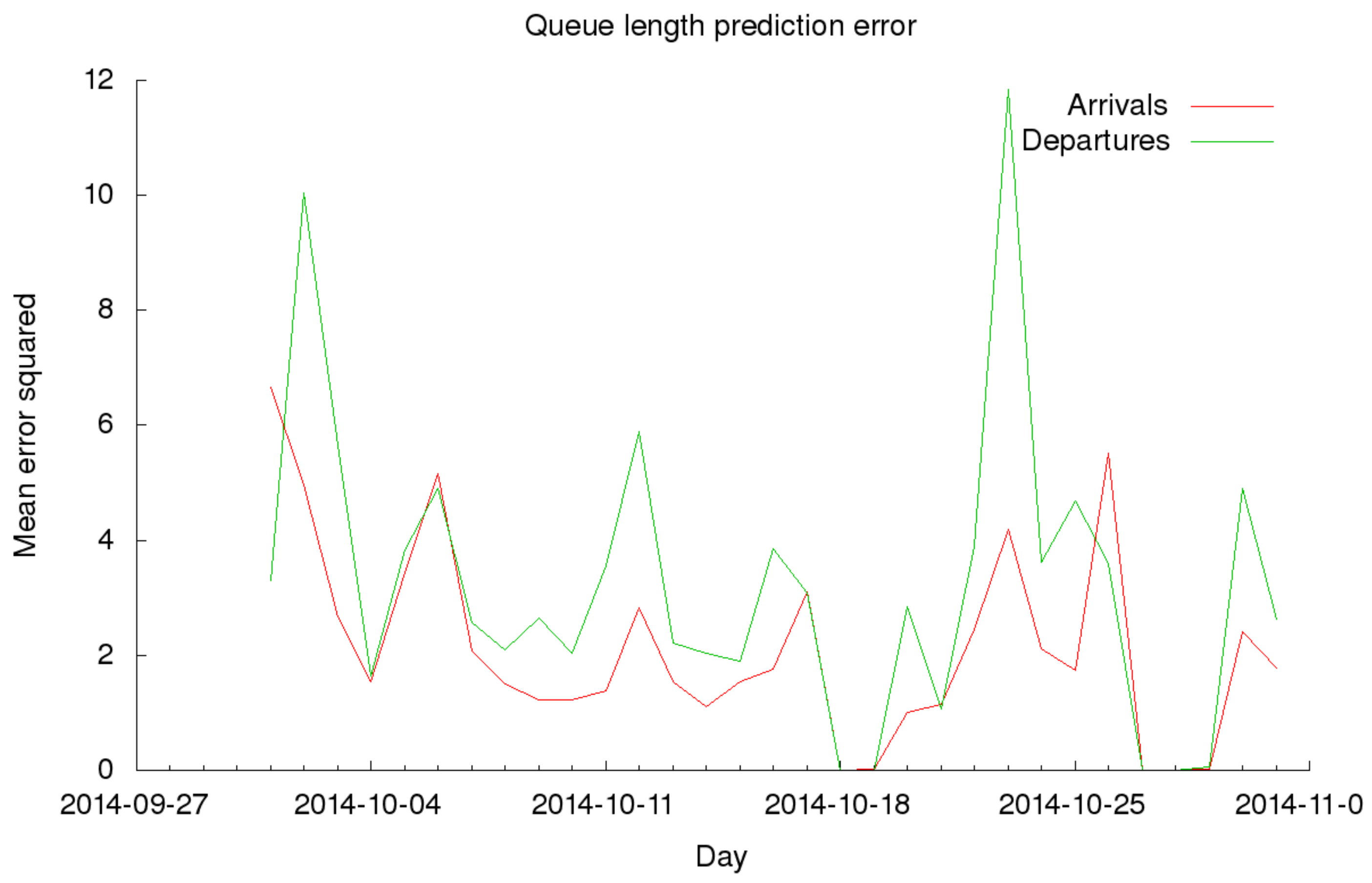


Figure 5.1: Mean error squared of the predictions of the stochastic model at airport 1 in October

One of the other sources of delay that we were able to observe is de-icing. At airport 1, the predictions of the model dramatically lost accuracy after December. While it is true that bad weather categories are more frequent during winter and that our model is less precise under these condition, the main reason for this is more likely the beginning of de-icing operation that have to be performed on many airplanes just before they arrive on their departing runway. As we understood, this can become the main bottleneck of the airport and would therefore need to be taken into account. The solution could be to compute a new RCCE for de-icing operations, but we were not able to isolate the time slots during which de-icing was undergoing with the data available to us.

Regarding the delay predictions that we can make by simulating the day of operations, they are surprisingly much lower than the actual delay. As the queue lengths predicted by our model are quite close to reality, we were not able to determine the reason for this.

## 5.2 Comparison of deterministic and stochastic framework decisions

We applied the stochastic simulations on the strategies output by the deterministic model so as to have an evaluation of the queue lengths that should be expected if we follow these strategies (since the stochastic framework is good at predicting queue lenghts). With this, we were able to compute the stochastic cost function on the deterministic model strategies. We could therefore see that depending on the airport, the cost function is between 3% and 7% lower if we apply the stochastic model. This is significant, but a bit surprising if we consider the poor quality of the deterministic model predictions, which suggests that a short-term vision of the upcoming operations can be sufficient to take good decisions in the studied airports.

Moreover, we computed the expected difference between the rates (both departure and arrivals) recommended by each algorithms at each slot. In figure 5.2, we show the average on all days of October

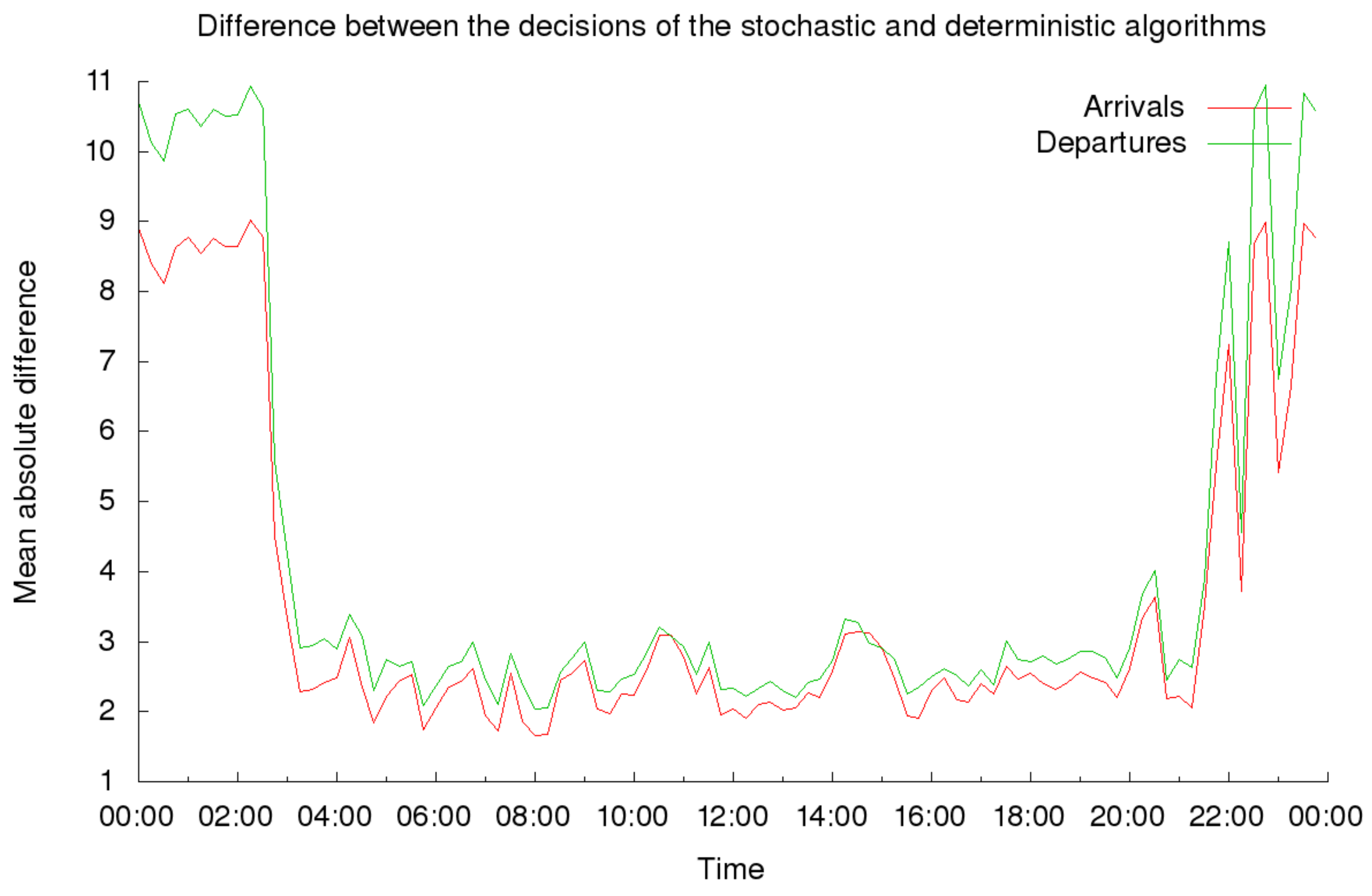


Figure 5.2: Average on all days of October of the expected absolute difference between the advised rates of the two algorithms

of these differences for each time slots. The high value at the beginning and the end are only caused by the fact that the algorithms decision is not really specified when the schedule is very low, such as during the night. The significant difference proves that the stochastic model not only makes more accurate predictions, but that these predictions have an impact on its decisions which are quite different from those selected by the deterministic model.

We also compared the way that the two algorithms schedules interruptions. The result was that for all input cases we tried, the deterministic algorithm scheduled the interruption begin at the time slot where the stochastic algorithm considers that it is the most likely to start. No significant evolutions in the difference between the cost function of the two algorithm was observed when they both have to schedule one interruption.

# Chapter 6

# Possible improvements

## 6.1 De-icing

De-icing is a major source of congestion in some European airports, to the extent that it breaks our model predictions, and it would therefore be worth taking it into account. The first step would be to find a way to identify the days and time slots during which de-icing is undergoing. Then it may be sufficient to build a new RCCE corresponding to the operational capacities when de-icing takes place. If this is unsuccessful, it is likely that the model has to be modified, perhaps by adding a queue for planes that are waiting for de-icing if it is found that both the runway and the de-icing pads are limiting ressources in the operations.

## 6.2 RCCE definition

Our definition of the RCCEs is not completely satisfying: we keep the points that have been reached at least 5 times, which means that the RCCE can only grow with the size of the dataset. We unsuccesfully attempted to compute it as the mean value of the departure rate that is observed under high departure demand and a given arrival rate [8]. The problem is to find slots during which the departure demand is saturating and the arrival rate is also high. In the studied airports, this seems to be unfrequent as the schedule tends to be balanced between periods of peek arrivals and peek departure. For this reason, the RCCE we obtained did not allow high departure rates.

## 6.3 Finding the source of error in delays predictions

As the queue predicted by the stochastic model are realistic, it is surprising that the delays are not. The fact that the expected queue lengths computed directly from the optimizer output are very close to the mean queue lengths of the simulations tends to prove that the simulation was correctly designed and implemented. The argument that even if we predict correctly how many aircrafts are queuing, we do not know which one are in the queue does not seems to hold, because the average delay on the whole day should still be directly correlated to the average queue length on the whole day. However, we predict a correct average queue length along with an average delay that is too low.

# Conclusion

We applied a stochastic model of airport operations designed with very congested airports in mind to European airports that are under slightly less pressure. The model is quite accurate regarding queue lenght prediction when the main bottlebneck of the airport operations is the runway system. It was also possible to compute optimal strategies according to this model in a matter of seconds for a whole operational day, and to allow constraints on the proposed solution, such as interruption scheduling, or pre-decided operationnal rates.

We also saw that trying to apply a deterministic model results in a prediction of the congestion that is by far too optimistic, and in a strategy that is likely to cause of order 5% more delays.

# Bibliography

[1] Jason Adam David Atkin. *On-line decision support for take-off runaway scheduling at London Heathrow Airport.* PhD thesis, University of Nottingham, 2008.

[2] Dimitris Bertsimas, Michael Frankovich, and Amedeo Odoni. Optimal selection of airport runway configurations. *Operations research*, 59(6):1407–1419, 2011.

[3] Mark Hansen, Tasos Nikoleris, David Lovell, Kleoniki Vlachou, and Amedeo Odoni. Use of queuing models to estimate delay savings from 4d trajectory precision. In *Eighth USA/Europe Air Traffic Management Research and Development Seminar*, 2009.

[4] Bluepony Technologies Inc. *CheckWX*, 2009.
`http://www.checkwx.com/`.

[5] A Jacquillat, A Odoni, and M Webster. Airport congestion mitigation through dynamic control of runway configurations and of arrival and departure service rates under stochastic operating conditions. Technical report, ESD Working Paper Series, Available at http://esd. mit. edu/WPS/2013/esd-wp-2013-14. pdf, 2013.

[6] Alexandre Jacquillat. *A queuing model of airport congestion and policy implications at JFK and EWR.* PhD thesis, Massachusetts Institute of Technology, 2012.

[7] Peeter Andrus Kivestu. *Alternative methods of investigating the time dependent M/G/k queue.* PhD thesis, Massachusetts Institute of Technology, 1976.

[8] Ioannis Simaiakis. *Analysis, modeling and control of the airport departure process.* PhD thesis, Massachusetts Institute of Technology, 2013.