MASTER IN COMPUTER SCIENCE

Computer-Aided Flight Slot Allocation

Othmane, Amine

Award date:
1996

Awarding institution:
University of Namur

Amine Othmane
$2^{ème}$ *Licence & Maitrise*

Computer-Aided Flight Slot Allocation

1995-96

Acknowledgements

The author would like to thank especially Pr. P.-Y. Schobbens and Ph. Waroquiers. Without their help and advises, achieving this thesis would have been impossible. He will also remenbers the friendly help that TACT team members have given him.

Contents

Introduction

The purpose of this thesis is the analysis of the Flight Slot Allocation Problem. Briefly, it consists of finding a slot i.e. a time window for departure in order to ensure that the flight will comply with all the procedures and rules that regulate the air traffic. Among other rules, the most important is that the numbers of flights which enter a sector or an area shoud never exceeds the sector capacity. Therefore CASA is a network problem but of a particular and sensitive nature.

Analysis means for us :

1. Describing the problem, i.e summarizing the state-of-the-art : Which rules apply, which problems arise, which solutions are used ?
2. Giving a formal model, i.e rewriting it as a mathematical problem, namely a dynamical optimization problem.
3. Proposing new solutions or improving existing ones by the use of appropriate (?) mathematical algorithms, namely the simulated annealing and/or the tabu search methods.
4. Comparing with other models and methods, namely constraint programming and 0-1 linear programming.

The case reviewed in detail in this study is the European case. Currently, air traffic in Europe is regulated by EUROCONTROL, the European Organization for the Safety and Control of Air Navigation. The purpose of this Body is to manage the traffic on real-time basis to avoid traffic jams (Yes, they exist even in the air !), excessive delays and more generally every situation or condition which could lead to hindering security with the help of a fully automated system.

The structure of this thesis is composed of four chapters and follows the structure of the objectives:

- The first chapter is devoted to a description of the current state of air traffic air and a review of the operational requirements.

- In the second chapter, we formalize the main concepts and rewrite CASA as an optimization problem.
- Thirdly, we review two mathematical methods which have proved useful to tackle such complex combinatorial problems : Simulated annealing and tabu search.
- The fourth chapter describes two other models : The Vranas & Bertsimas model which use the 0-1 linear programming formalism and was developed at the beginning for the american case, and the ILOG model which implements somewhat the constraint programming technique.

For the readers who are not at ease with the rather esoteric language of air controllers, a useful list of acronyms have been added to this thesis.

Part I

Description and Formalizing of CASA

Chapter 1

Air Traffic Control in Europe

1.1 Limited Capacity, Growing Demand

People could wonder why, in the air, without policemen, highways, lights, borders and walls, planes are not allowed to fly freely ? Instead of it, air traffic is submitted to restrictions that the common car commuter can hardly imagine :

- Minimal delay between two departures.
- Minimal distance between two planes in the three directions.
- Forbidding of crossing some special areas (big cities, industrial resorts, nuclear power plants, military bases, ...).
- Obligation to use predefined paths.
- Mandatory identification when entering a national airspace.

Problems arise when to control that the flights actually comply with all these regulations and procedures. In Europe, such controls are processed by EUROCONTROL, the European Organization for the Safety and Control of Air Navigation. Actually, a increasing part of controlling has been "computerized". The main principle of Air Traffic Regulation is achieved by giving to each flight a slot, i.e. a short time period for taking off such that the flight will comply with (almost) all regulations and required procedures.

The aim of this report is to study and propose improvements of the core of the system, namely CASA (Computer-Aided Slot Allocation) which is embedded in the TACT (TACTical) system which is merely the interface between CASA and the users, both internal (the controllers) and external (the Air Operators).

We start by listing what we call the *operational constraints*, i.e. the constraints imposed by the very nature of the Slot Allocation Problem itself and not by the peculiar algorithm or method we would like to implement to face the issue. In the following, we will also summarize the state of the current implementation at EUROCONTROL.

1.2 The Operational Constraints

As a general remark, the ATC (Air Traffic Control[1]) has to be achieved under the responsibility of human beings. This has at least two consequences :

- The users should be confident about the system, which means practically a quite complex implementation to face all the possible issues.
- The users should have a clear knowledge of the structure and the behavior of the system, which implies practically the opposite of the previous requirement.

In addition, the demand of the AO (Air Operators) and customers is rapidly growing, but this demand is threefold :

- The number of available destinations.
- The number of places available, i.e. how many people can travel during a year ?
- A more flexible scheme, i.e. smaller flights which can departure at any time. This is in opposition to the traditional system of the sixties-seventies, when there were only long-hauls which were taking off on a regular, fixed basis.

[1]See the list of acronyms at the end of the report

These three features of the current demand increase the total demand of flights, leading to air traffic jams. The observable consequences are bigger delays on the ground but also in the air. Air Traffic Regulating Bodies had to set up one-way paths (in the air !), "flying round-abouts" around international airports, etc ...

1.2.1 Some Terminology

Flights A flight is known by its identification (ID), its departure airport (ADEP) and its destination airport (ADES). It can cross and follow many locations and areas :

- Paths (Uxyy) where x is a letter and yy an integer.
- Unit Information Regions (UIR).
- Flight Information Regions (FIR).
- Aerodroms (AR).
- Navigation Points (NAVAIDs) and Way-points, etc ...

Slots Each flight is given a slot, i.e. a departure time interval which length is rarely over 5 minutes. As flight procedures are normalized, it is easy to compute the crossing and arrival times of the flight with a reasonable precision.

Traffic Volumes To simplify and uniformize ATC, the concept of Traffic Volumes has been developed : It consists of a piece of airspace of limited capacity, the capacity being the number of flights allowed to cross the traffic volume per hour. It is identified by an alphanumeric string[2]. It can be one airport, a set of airports, a path, a family of paths, an UIR, etc ...

[2]Very often, it consists of four letters with the following meaning : the first one denotes a continent or a part of it, e.g. E = Northern Europe, L = Latin Europe, U = Soviet Union; the two first letters combined denote the country, LF = France, EB = Belgium, LS = Switzerland, ES = Sweden; the third and fourth letters are up to the relevant National Regulating Body. Many small countries like Belgium, Netherlands, the Baltic Republics are covered by *one* Traffic Volume.

Regulations A regulation is defined by an alphanumeric string[3] and consists of a Traffic Volume, a validity period (some hours) and a maximum rate of planes allowed to enter the area. Usually, the rate is expressed in flights per hour, and this rate may does not have to remain constant over the validity period, it can even be updated *en-route.*

1.2.2 Regulating the Demand

The core of the method is to avoid traffic jams and delays in the air, for two considerations : financial (fuel is burned, maintenance costs are proportional to the number of flight hours) and security. Taking into account the maximum capacity of Traffic Volumes, ATC Centres *impose ground* delays to flights. But when the actual demand exceeds the capacity, controllers have to define Regulations on the concerned overcrowded Traffic Volumes.

1.2.3 The Constraints

Flights have to comply with many constraints that we systematically state hereinafter.

1. **Sub-periods** A regulation period is divided in sub-periods. Each sub-period has its own rate. The rate defines the "maximum" number of flights in the regulation for the relevant sub-period.

2. **Slots** Each sub-period is divided in slots. The currently used division consists of equally distributed slots with one flight per slot. However, we could consider slots with variable time length and capacity, e.g. 5 flights within a 20' slot.

3. **Exempted flights** "Exempted flights" means flights which cannot be subject to a regulation. They consume resources and slots as others, they are taken into account to compute the loads but they cannot be delayed as others. This includes official, humanitarian, emergency flights and flights coming from outside the European Air Traffic Flow Management (ATFM) zone.

[3]Very often by six letters : the first four letters being the identification of the relevant Traffic Volume, the two last identify the time.

4. **Minimal overloads** The load, i.e. the number of flights in a regulation, should not be higher then the rate of the concerned sub-period. All the flights are considered, both exempted and regulated ones.
   CAUTION : This is not an absolute criterion (*should not be* instead of *has to be*).

5. **Delaying flights : The core of the method** In order to fulfill the previous constraint (4), flight departures are delayed in the future. As a consequence, delays are positive numbers.

6. **Forcing exempted flights** Exempted flights are "forced" to their ETO. In other words, they must be given a slot within short time window around their ETO.

7. **Pending rates** In order to give some flexibility, a small amount of the controlling capacity is devoted to a pending rate. This special rate is mainly used by late filers and short notice flights. By opposition, the rate defined in constraint (1) will now be referred as *normal* rate.

8. **Conversion of pending rates** In order to limit the loss of capacity, pending slots which are close to the current time are converted to normal ones. As a remark, the criterion is much more complex due to the unclear definition of time-bands.

9. **Shallow modification** On request of a Central Executive Unit (CEU) controller, a new regulation could be achieved. If the only change is in the modification of the rate, it is called shallow rectifying. It consists of finding better slots for flights which CTO is after the current time. Allocated flights have precedence over pre-allocated ones[4].

10. **Deep modification** By comparison with the previous constraint, when other changes occur in the regulation definition, the modifying is referred as deep rectifying. It consists in :

    - For flights which Computed Take-Off Time (CTOT) is before the deep rectifying start-time : Nothing changes.

[4]For the difference between allocated flights and pre-allocated ones, see the Message Procedures section

- For flights which have received a Slot Improvement Proposition[5] (SIP) : If the CTOT is after the deep rectifying start-time and the flight is subject to this regulation, the SIP is canceled.
- For other flights : A new regulation is restarted from the beginning.

As CTOT is obviously before CTO, the set of flights subject to modification is much bigger, hence the qualification of "deep".

11. **Regulation canceling** On request of a CEU controller, a regulation can be canceled. This can be due to many factors : strikes, bad weather, riots, wars, etc ...
It consists of :

    - For flights which have been forced to a slot by a CEU controller : Nothing changes.
    - For flights which are airborne or presumed so, i.e. their CTOT is after the current time : Nothing changes *excepting* when the regulation is the last one which the flight is subject to. In this case, the flight may be deleted from the CASA database.
    - For flights which are in any regulation any more : The flight may be deleted from the CASA database.

12. **Flight Plan Revision** The data about a flight can be updated by reception of a message of the outside world (AO, ATC, etc ...). Following rules apply :

    - For flights which are airborne or presumed so : Nothing changes.
    - For flights with a changed profile : A new reservation procedure is made for that flight. Even if it has received a SIP, the corresponding slot is freed. *However*, if the flight has been manually forced by a CEU controller and the new ETOT is still before the old CTOT, nothing changes.
    - For flights with a changed EOBT : A new reservation is normally made. (Still to argue about, cfr. [4])

[5] See the Message Procedures section

13. **Filtering Slot Improvements** During the slot allocation optimization process, only delay improvements which are higher then a given threshold will be taken into account. In other words, if the difference between the old CTOT and the new one is too low, no changes are made.

After reading this, at least two conclusions should be made : Slot Allocation is subject to various, informal restrictions *and* decisions of human beings have always precedence (cfr constraints 4,9,10,11) on computer ones, hence the complexity of the problem. Moreover, the lack of formality of the operational constraints implies that we cannot use classical optimization algorithms because they tend to comply *strictly* with the constraints.

1.3 The Message Procedures

Various messages are exchanged between the outside world and TACT & CASA. As a general remark, any change in the optimization method should be transparent, hence the Message Procedures have to be left unchanged.

In this section, we will describe in a somewhat chronological way, i.e. the time sequence in which they most often occur.

RPL, IFPL First of all, AO submit flight plans to TACT. These plans are called either Repetitive PLans (RPL) or Initial Flight PLans (IFPL). Repetitive means that the flight occurs on a daily basis. The flight is then recorded in the CASA database.

SAM A fixed time (often one or two hours) before Initial Off-Block Time (IOBT), a Slot Allocation Message (SAM) is sent to the AO. This is the proposition of an available departure time, also called Computed Take-Off Time (CTOT). The flight is said to be allocated. A flight is said to be pre-allocated when CASA has found a suitable slot but has not sent a SAM yet. The minimal delay between the the issuance of the slot and IOBT is also referred as Slot Issuance Time (SIT).

SIP After that, the CASA algorithm can propose better slots to the AO. A Slot Improvement Proposal (SIP) is then sent. The AO has then three possibilities :

SRJ The AO rejects the improvement. It then sends a Slot ReJection Message (SRJ) to TACT. If it does not not answer after a certain time, it is reputed to have rejected the improvement. In both cases, the corresponding slot is freed.

SPA The AO accepts the improvement. It then sends a Slot Proposition Acceptance (SPA) to TACT.

RDY If prior to any improvement, the AO has sent a ReaDY message (RDY), it is reputed to accept all SIPs.

SRM The accepted improvement is then made official by TACT by sending a Slot Revision Message (SRM) to the AO.

SRR If the AO needs to change the departure time (CTOT), it sends a Slot Revision Request (SRR) to TACT. It receives back a SRM.

SLC For various reasons, a CEU controller can cancel a slot, i.e. giving no CTOT to a flight. A SLot Cancellation (SLC) is then sent to the AO.

As a consequence of constraint (13) *Filtering Slot Improvements*, the difference between, respectively, the CTOT of the SAM and the CTOT of the first SIP has to be greater than a given threshold. The same applies to two successive SIPs.

1.4 The Current Implementation of CASA

1.4.1 The Basic Rules

We state here briefly the main principles on which the current implementation of CASA relies in order to fulfill all the previous operational constraints and

message procedures and, in addition, trying to find an optimum distribution of slots and flights.

Delaying flights This is achieved, within EUROCONTROL by CASA. The basic rule is *first planned, first served*, i.e. if flight A has planned its departure before flight B, flight A will actually take off before flight B. This rule is equivalently stated by the following equation :

$$ETO_A \leq ETO_B \Rightarrow CTO_A \leq CTO_B \tag{1.1}$$

where ETO and CTO stand, respectively, for Estimated Time Over and Computed Time Over.

More then one regulation : The most penalizing technique When a flight crosses more than one regulation[6], a problem arises to determine which delay it will eventually suffer of. As overloads are currently computed for each regulation, the actual practice is, for simplicity, to search independently in each crossed regulation a delay. Hence, there are as many delays as crossed regulations. The bigger delay is then selected and the flight is "forced" in the other regulations, i.e. for each relevant regulation, the CTO is the sum of the ETO plus the maximum imposed delay and this, even if these regulations are overloaded. The regulation which has imposed its delay on the flight is referred as the *most penalizing* one.

1.4.2 The Actual Flaws of CASA

Obviously, the current implementation suffers from some flaws. The first one is that the main objectives are nowhere clearly defined. Actually, TACT & CASA have been developed in a purely heuristic way, by implementing the "manual" procedures used by each European ATC Centre. As the regulation of European Air Traffic is planned to be achieved on a global basis, these heuristics do not make the full benefits of the current technology.

[6] A flight crosses in average 2.5 regulations

First planned, first served This principle is too strict and may lead to higher average delays. It can be observed in practice that, by swapping two flights, the average delay could decrease in a noticeable way.

No priority There is no idea of priority between flights, e.g. long-hauls are treated on the same basis as the short-hauls even if they book in advance, big carriers are treated equal to small planes, etc ...

No cost function Oddly enough, there is no idea of a global measure of the delays, no cost function is specified, which is quite strange for what looks like a serious optimization problem.

The most penalizing technique Beside of being suspected of inefficiency, this principle can lead to some strange behaviors :

Chapter 2

Modeling CASA

2.1 Flights in Equations

In this section, the various concepts reviewed in the first chapter will be more formalized. In the following section, we will try to formalize CASA as an optimization problem.

2.1.1 Flights

A flight f is defined by :

- ETOT at time t : $\hat{\tau}^t_{d,f}$
- ETO for regulation r at time t : $\hat{\tau}^t_{r,f}$
- CTOT at time t : $\tau^t_{d,f}$
- CTO for regulation r at time t : $\tau^t_{r,f}$
- Priority coefficient at time t : c^t_f

We recall that a flight has *only one* ETOT, but as many ETOs as regulations it crosses. The same applies for CTOT and CTOs. We will keep the notation of a $\hat{}$ superscript over symbols which refer to estimated values.

2.1.2 Regulations

A regulation r is defined by :

- Start-time at time t : $T^t_{d,r}$
- End-time at time t : $T^t_{e,r}$
- Slot list at time t : $\mathcal{S}^t_r$

A regulation is linked to one and only one Traffic Volume. In other words, at a given time t, only one regulation is active for the considered Traffic Volume.

2.1.3 Slot Lists

A slot list records for each slot s :

- Start-time at time t : $T^t_{d,s}$
- End-time at time t : $T^t_{e,s}$
- Capacity at time t : $\kappa^t_{r,s}$
- Overload at time t : $\epsilon^t_{r,s}$
- *And for each flight f in the slot s :*
  - Identifier of flight f : I_f
  - ETO for regulation r at time t : $\hat{\tau}^t_{r,f}$
  - CTO for regulation r at time t : $\tau^t_{r,f}$
  - Delay of flight f : $d^t_f := \tau^t_{r,f} - \hat{\tau}^t_{r,f}$

A delay shall always be positive, i.e. $\forall f \, \forall t, \, d^t_f \geq 0$

2.2 CASA as an Optimization Problem

2.2.1 The Naive Version

We define :

- The set of all flights at time t : $\mathcal{F}^t$
- The set of all regulations at time t : $\mathcal{R}^t$
- The set of all slots for regulation r at time t : $\mathcal{S}_r^t$

We can partition the set of flights between the regulated ones and the others :

$$\mathcal{F}^t = \mathcal{F}^t_{reg} \cup \mathcal{F}^t_{noreg} \tag{2.1}$$

with $\forall t$:

$$\mathcal{F}^t_{reg} \cap \mathcal{F}^t_{noreg} = \emptyset \tag{2.2}$$

More precisely, if we define $\mathcal{F}t_{r,s}$ as the set of flights which are allocated to the slots s of the regulation r at the instant t, we can state more formally the dynamic feature of our optimization problem. Actually, as explained in section 1.2, the data and the parameters of the problem are eglictible to change continuously on a real-time basis.

Regarding the set of flights, this feature can be elicited as :

$$\mathcal{F}^{t+1}_{reg} = \mathcal{F}^t_{r,s} \qquad \text{Formerly regulated flights} \tag{2.3a}$$

$$+ \bigcup_{\substack{r \in \mathcal{NR}^t \\ s \in \mathcal{S}^t_r}} \mathcal{F}^t_{r,s} \qquad \text{Newly created regulations} \tag{2.3b}$$

$$\setminus \bigcup_{\substack{r \in \mathcal{OR}^t \\ s \in \mathcal{S}^t_r}} \mathcal{F}^t_{r,s} \qquad \text{Terminated/cancelled regulations} \tag{2.3c}$$

$$\setminus \left\{ f \in \mathcal{F}^t_{reg} \mid \hat{\tau}^t_{d,f} > t \right\} \qquad \text{Taken-off flights} \tag{2.3d}$$

$$\setminus \mathcal{ME}^t \qquad \text{Manually exempted flights} \tag{2.3e}$$

The second and third terms of the equation refer to flights which, respectively, are (were) subject to regulations which are newly created (cancelled or terminated). As a flight can be subject to many regulations, it can happen that a flight still remains in $\mathcal{F}^t_{reg}$ after such modifications.

The basic optimization problem can hence be stated as follow :

$$\min \quad \omega^t = \sum_{f \in \mathcal{F}^t_{reg}} c^t_f \cdot d^t_f \qquad d^t_f : \text{decision variables} \tag{2.4a}$$

$$\text{w.r.t. } \epsilon^t_{r,s} < 0 \qquad \forall r \in \mathcal{R}^t, \forall s \in \mathcal{S}^t_r \tag{2.4b}$$

$$d^t_f \geq 0 \qquad \forall f \in \mathcal{F}^t_{reg} \tag{2.4c}$$

This problem has to be solved at each "solving" time t. The cost function is obviously the sum of each flight's delay times its priority coefficient. The basic constraints are that we demand (or we require) that :

- Any slot of any regulation should not be overload.
- Delays are to be positive.

Whether these constraints are requirements or demands is still under discussion. In an ideal world, this should be strictly respected, but as the actual air traffic in Europe is very congested, the current policy is to let the the flights taking off even at the expense of overloading the regulations and hence the capacity of the Control Centres. Otherwise, stronger security could be guaranteed but at the expense of tremendous delays (many hours !).

2.2.2 The Constructive Version

We can notice that the dependency of the ϵ's upon the variables of the problem, namely the delays (d^t_f) is not obvious. This is due to the fact there is no clear relationship between time and the time division in slots.

Actually, to state more formally this relationship : Let f be flight belonging to slot s of regulation r at time t, i.e :

$$f \in \mathcal{F}^t_{r,s}$$

If we suppose that f suffers from a delay d_f^t and that $d_f^{t+1} = d_f^t + \delta$, δ being positive or not. If we define

$$\rho^t = \{r \in \mathcal{R}^t \mid \exists! s : f \in \mathcal{F}_{r,s}^t\} \tag{2.5}$$

the set of all regulations which contain one and only one slot allocated to flight f at time t, i.e the regulations f is subject to. Provided that ρ^{t+1} has been computed from the value of δ and ρ^t, we can deduce :

$$\forall r \in \rho^t \backslash \rho^{t+1} \qquad \epsilon_{r,s}^{t+1} = \epsilon_{r,s}^t - 1 \tag{2.6a}$$

$$\forall r \in \left(\rho^{t+1} \cap \rho^t\right) \cup \mathcal{C}_{\mathcal{R}^{t+1}}\left(\rho^{t+1} \cup \rho^t\right) \qquad \epsilon_{r,s}^{t+1} = \epsilon_{r,s}^t \tag{2.6b}$$

$$\forall r \in \rho^{t+1} \backslash \rho^t \qquad \epsilon_{r,s}^{t+1} = \epsilon_{r,s}^t + 1 \tag{2.6c}$$

The first equation (2.6a) means that for a slot which is no more allocated to the flight f, its overload decreases by one unit. The third equation (2.6c) means that for a slot which is newly allocated, its overload increases. The second equation (2.6b) regards all other slots : Those who are still linked to the flight f as shown by the first term $(\rho^{t+1} \cap \rho^t)$, the second term represents the slots which are not concerned by the changing of the delay of f.

We can restate the problem in a more constructive (?) way following an idea in ([11]) by introducing the following auxiliary variables :

$$u_{rsf}^t \in \{0, 1\} \tag{2.7}$$

$$\delta_{rf}^t \geq 0 \tag{2.8}$$

Setting u_{rsf}^t to 1 means that f has been linked to the slot s of regulation r at time t. That is why the u's are often referred as "time stamps". The value of δ_{rf}^t express the difference between the beginning of the slot allocated to f and its computed entrance time (CTO) in the regulation r.

We can then express the relationship between these auxiliary variables and the flight's variables. Hereinafter, $*$ refers to the time horizon, i.e. $* \in \mathbb{N}$ and T_{rs}^t is the duration of the slot s which is often supposed equal[1] within

[1]It is not always so in current practice.

the regulation.

$$\underbrace{\tau^t_{r,f}}_{I} = \underbrace{\hat{\tau}^t_{r,f}}_{II} + \underbrace{d^t_f}_{III} = \underbrace{\sum_{s=1}^{*}\left(\sum_{\sigma=1}^{s-1} T^t_{r\sigma}\right) u^t_{rsf}}_{IV} + \underbrace{\delta^t_{rf}}_{V} \tag{2.9a}$$

$$\text{w.r.t.} \quad \sum_{s=1}^{*} u^t_{rsf} = 1 \tag{2.9b}$$

$$\delta^t_{rf} < T^t_{r,s+1} \tag{2.9c}$$

The system (2.9) should therefore respected for all flights f's and all regulations r's. Its first equation (2.9a) can be explained as follows : The first term is the CTO, i.e the computed entrance time of the flight f in the regulation r, it is by definition of the ground-holding policy equal to the ETO (term II), i.e the intended entrance time plus its delay (term III). It is also equal to the duration of all slots (term IV) preceding the one which is allocated to the flight plus its delay δ^t_{rf} within the allocated slot (term V).

Equation (2.9b) ensures us that one and only one slot is allocated to the flight f within the regulation r. Equation (2.9c) guarantees that the flight's CTO is not pushed beyond the end of the allocated slot.

So the load of a slot can be connected to the time-stamps by observing that the number of flights a slot is allocated to is :

$$\epsilon^t_{rs} + \kappa^t_{rs} = \sum_{f \in \mathcal{F}^t} u^t_{rsf} \tag{2.10}$$

The problem can then be fully stated :

$$\min \quad \omega^t = \sum_{f \in \mathcal{F}^t} c_f^t \cdot d_f^t \qquad \text{cost function} \qquad (2.11a)$$

$$\text{w.r.t. } \epsilon_{rs}^t \leq 0 \qquad \text{no overloads} \qquad (2.11b)$$

$$d_f^t \geq 0 \qquad \text{positive delays} \qquad (2.11c)$$

$$\hat{\tau}_{rf}^t + d_f^t = \sum_{s=1}^{*} \left(\sum_{\sigma=1}^{s-1} T_{r\sigma}^t \right) \quad u_{rsf}^t + \delta_{rf}^t \qquad (2.11d)$$

$$\delta_{rf}^t < T_{r,s+1}^t \qquad s \text{ such } u_{rsf}^t = 1 \qquad (2.11e)$$

$$\sum_{s=1}^{*} u_{rsf}^t = 1 \qquad \text{only one slot allocated to } f \qquad (2.11f)$$

$$\epsilon_{rs}^t + \kappa_{rs}^t = \sum_{f \in \mathcal{F}^t} u_{rsf}^t \qquad \text{slot's overload} \qquad (2.11g)$$

where :

- d_f^t : decision variables (natural numbers)
- ϵ_{rs}^t : constraint variables (natural)
- u_{rsf}^t, δ_{rf}^t : auxiliary variables (u's binary, δ's natural)
- c_f^t, $\hat{\tau}_{rf}^t$, $T_{r,s}^t$, κ_{rs}^t : parameters (natural)

If we want to be complete, we have also to respect the constraint *Filtering slot improvements* (subsection 1.2.3, 13) which leads to the following constraint :

$$\forall t, \ |d_f^{t+1} - d_f^t| > \Theta \qquad (2.12)$$

Θ being the threshold between two successive SIPs.

To get a idea of the magnitude of the figures in the real case : Actually in Europe, more or less 20000 flights a day take off and land, of which a round value of 6000 are regulated. Around 100 regulations are created per day. Taking an average duration of 5 hours and a slot duration of 3', we see that a regulation's slot list contains approximately 100 slots. Hence, our problem contains :

- d_f^t : $|\mathcal{F}^t| = 6000$
- ϵ_{rs}^t : $|\mathcal{R}^t| \cdot |\mathcal{S}_r^t| = 10000$
- δ_{rf}^t : $|\mathcal{R}^t| \cdot |\mathcal{F}^t| = 600000$. Hence, a total of 616000 integer variables.
- u_{rsf}^t : $|\mathcal{R}^t| \cdot |\mathcal{S}_r^t| \cdot |\mathcal{F}^t| = 60$ millions[2] (sic !) binary variables.
- c_f^t : $|\mathcal{F}^t| = 6000$
- $\hat{\tau}_{rf}^t$: $|\mathcal{R}^t| \cdot |\mathcal{F}^t| = 600000$
- $T_{r,s}^t$: $|\mathcal{R}^t| \cdot |\mathcal{S}_r^t| = 10000$
- κ_{rs}^t : $|\mathcal{R}^t| \cdot |\mathcal{S}_r^t| = 10000$. Hence, a total of 622000 integer parameters.

Clearly, direct resolution of Europe air traffic congestion with this model is highly infeasible. However, two remarks can be made :

- A clever, i.e dynamic managing of the variables can noticeably reduce the size of the problem. Actually, a flight is suspectable to be subject only to the regulations linked to the Traffic Volumes it crosses, not *all* the regulations. But as the problem is dynamic, structuring the variables as arrays would lead to the higher values.
- If we suppose that the slot duration is constant, i.e. $T_{rs}^t = T$ and that all the δ's are equal to zero, which means practically that T is the new time unit, (2.11) is a linear 0-1 problem, for which solving methods exist.

Hence, we have to explore new ways of solving these problems, exploration which is left to the next part of this thesis.

[2]These estimates are upper values and are obtained by multiplying the cardinal of each set.

Part II

Solving CASA : The Simulated Annealing and Tabu Search Optimization Techniques

Chapter 3

Simulated Annealing & Tabu Search

3.1 Introduction

In this chapter, we will give a brief description of two of the best known combinatorial optimization techniques : Simulated annealing and tabu search. This is motivated by the problem we have at hand, namely CASA. Obviously, direct resolution of the model (2.11) is infeasible. Resolution of a linear problem in $\mathbb{R}$ with 60 million variables is very demanding in resources[1] so what to say of 0-1 linear problem ...

Such resolutions have been attempted by Vranas ([11, 15, 14]) by using linear relaxation of the 0-1 problem and adding some simplifications but its use in an operational environment is doubtful as the solving time's magnitude is of one hour on a Sun Sparc workstation. However, the description of this attempt and others will be deferred to the next chapter.

In two words, simulated annealing is a combinatorial optimization technique, which means that :

- It is stochastic in nature.
- It gives a near-optimum (if it converges !) of the problem at hand.

[1] We recall that the problem is dynamic, i.e. its parameters change continuously and it should be solved each, say 10' !

As we will see later, only asymptotic convergence can be proven for this technique but no proof of "true" convergence[2] exists, up to now.

Actually, the paradigm of simulated annealing stems from thermodynamics and tries to reproduce the physical phenomenon of annealing, i.e. the slow decreasing of free energy of a material downwards its global maximum by lowering carefully the temperature, hence the name. It is applied in industry in material science to obtain a harder, bigger and more stable crystal of the material. Since the Middle Age, it has been used for e.g. swords in annealed steel.

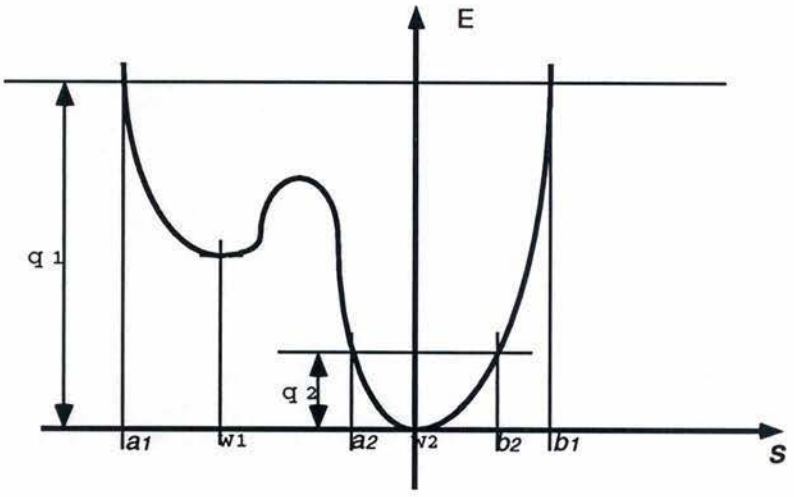

Figure 3.1: Free energy and temperature

As shown on Fig. 3.1, when the temperature is increased to Θ_1, all the states between $[a_1, b_1]$ are allowed because temperature (T) is a measure of the kinetic energy of the molecules and the free energy (E) of a material is somewhat related to it[3]. Actually, the relationship is :

$$E_c = k_B T \quad k_B : \textit{Boltzmann constant} \tag{3.1}$$

so two minima are possible : ω_1 which is local and ω_2 which is global.

[2] "Full" convergence implies asymptotic convergence but not the opposite

[3] Please refer to a good book on thermodynamics for a more rigorous and detailed explanation

If temperature is decreased carefully to Θ_2, then only the states between $[a_2, b_2]$ are allowed so only one minimum impossible, namely ω_2. But if the temperature is too quickly decreased, then it can happen that the system is trapped at the local minimum ω_1 which is also referred as a meta-stable state whereas ω_2 is referred as a stable state, following thermo-dynamical terminology. The simulated annealing tries to reproduce the physical phenomenon with the following correspondences :

- Finding a stable state is equivalent to minimize a function : The energy.
- The energy of the system is, in fact, a cost function.
- A state of the physical system is equivalent to a state of the combinatorial problem, i.e. an instantiation of all the variables.

3.2 The Homogeneous Scheme

3.2.1 The Model : Stochastic Transitions

We give here a more formal description of the behavior of the system during the annealing/optimization phase. We suppose that the problem to solve is :

$$\min \omega(x) \tag{3.2}$$
$$\text{w.r.t. } x \in \mathcal{S} \tag{3.3}$$

ω being the cost function and $\mathcal{S}$ the set of "realizable" solutions.

We can model the behavior of annealing (both simulated and physical) by assessing that the system jumps from a state i to a state j until equilibrium is reached. Such transition is accepted with the following probability :

$$\exp\left(\frac{\omega_i - \omega_j}{k_B T}\right) \tag{3.4}$$

This acceptance rule is commonly referred as the *Metropolis* criterion. When equilibrium is achieved the probability that the system is in state i is :

$$P\{X = i\} = \frac{1}{Z(T)} \exp\left(\frac{\omega_i - \omega_j}{k_B T}\right) \tag{3.5}$$

where $Z(T)$ is the normalization constant :

$$Z(T) := \sum_j \exp\left(\frac{-\omega_j}{k_B T}\right)$$

Obviously, a state with lower "energy" has more chances to be selected as the final state t equilibrium.

Following the works of [2], the acceptance rule can be defined as :

$$\begin{aligned} A_{ij} &= P(\text{accept } j \text{ from } i \mid i, j \in \mathcal{S}) \\ &= \begin{cases} 1 & \text{if } \omega(j) - \omega(i) \leq 0 \\ \exp\left(\frac{\omega_i - \omega_j}{c}\right) & \text{if } \omega(j) - \omega(i) \geq 0 \end{cases} \end{aligned} \tag{3.6}$$

where c is the new constant, introduced for the sake of a better generality.

We have now to elicit-ate how the system jumps from one state to another. Actually, many formalisms are allowed but they share a common feature : They are all stochastic. Some authors prefer a constant distribution [2], others prefer Gaussian or Cauchy distributions [5, 9]. The choice is based upon selecting the one which leads to a better rate of convergence or a better ensured convergence[4] or a better ergodicity.

We recall that ergodicity is (one definition among others) the opposite of limit of the correlation between two sequences of consecutive states of the system : i.e. if the behavior of the system at one instant is closely related to the behavior of the system in the future, then the system is weakly ergodic. Strong ergodicity implies two things :

- The final behavior will not depend on the initial conditions. As we do not know the structure of the optimal solutions, we are ensured that the system will converge or, at least, it will not be trapped in a local minimum.
- The averages computed on the ensembles are, at equilibrium, averages of observables' values (Gibbs, 1902).

[4] As we will see later, only asymptotic convergence has been proven. However some schemes have "almost-proven" convergence.

Quite often, it is observed in practice (e.g. Some combinatorial techniques such genetic algorithms, Lester Ingber's Very Fast Simulated Re-annealing, Adaptative Simulated Annealing[5]) that some schemes have better rate of convergence but lack of ergodicity. In other words, the found minimum is often not the global one.

The simplest scheme [2] is the constant distribution : All the states of the neighboring $\mathcal{S}_i$ of the current system's state i are proposed with equal probability :

$$\begin{aligned} G_{ij} &= P(\text{generate } j \text{ from } i \mid i, j \in \mathcal{S}) \\ &= \begin{cases} 0 & \text{if } j \notin \mathcal{S}_i \\ \frac{1}{|\mathcal{S}_i|} & \text{if } j \in \mathcal{S}_i \end{cases} \end{aligned} \tag{3.7}$$

Therefore, the probability that a transition of the system from state i to state j takes place is simply :

$$P_{ij} := \begin{cases} G_{ij} \cdot A_{ij} & \text{if } i \neq j \\ 1 - \sum_{l \in \mathcal{S}, \updownarrow \neq)} P_{il} & \end{cases} \tag{3.8}$$

3.2.2 Convergence & Markov Chains

As mentioned in the subsection's title, we will use the Markov chains theory to assess the asymptotic convergence of the homogeneous scheme. A Markov chain is simply a sequence of transitions of the system where the probability of each transition is specified : $P_{ij}^k = P(X(k) = j \mid X(k-1) = i)$, i.e. the probability that transition $i \to j$ takes place at time k, the matrix $\mathbf{P}$ being commonly referred as the *transition* matrix. If $\mathbf{P}$ does not depend on the "transition time" k, the Markov is called *homogeneous*; if not, it is called *inhomogeneous.*

As we are interested in the long-term/final behavior of the system, we define the stationary distribution (Feller, 1950) :

$$q_i := \lim_{k \to +\infty} P(X(k) = i \mid X(0) = j, \forall j) \tag{3.9}$$

[5]Ingber's simulated annealing schemes have incorporated a lot of tricks and heuristics, so almost no formal proof could have been given, but at least, they are claimed to be very effective operationally.

With this definition, it can be shown that the stationary distribution deserves its name, i.e satisfies $\mathbf{q}^T = \mathbf{q}^T \mathbf{P}$. In other words, this distribution is left unchanged by any transition that would take place and it is the left eigenvector of $\mathbf{P}$ with eigenvalue 1. Moreover, this distribution, if it exists, is also the *final* distribution of the system, i.e. :

$$\lim_{k \to +\infty} a_i := \lim_{k \to +\infty} P(X(k) = i) = q_i \tag{3.10}$$

The conditions of existence of such a stationary distribution are the following (Feller, 1950) :

Theorem 3.1 *Let* $\mathbf{P}$ *be the transition matrix associated to a Markov chain. Then* $\mathbf{P}$ *has a stationary distribution* $\mathbf{q}$ *if :*

- $\mathbf{P}$ *is aperiodic*[6]*, i.e. the greatest period should be 1 for all i's.*
- $\mathbf{P}$ *is irreducible, i.e. any state is reachable from another in a finite number of transitions.*

$$\forall i, j \; \exists n > 0 \; : P_{ij}^n > 0$$

- $\mathbf{P}$ *satisfies the* balance *equation :*

$$\mathbf{q}^T = \mathbf{q}^T \mathbf{P} \tag{3.11}$$

We have now to show that simulated annealing re-defined as a Markov chain (3.8) complies with the required conditions (3.1).

Theorem 3.2 [7] *Let* $\mathbf{P}$ *be the transition matrix associated with the simulated annealing algorithm as previously defined. Then the algorithm has a stationary distribution : the Boltzmann distribution.*

$$q_i(c) = \frac{1}{Z(c)} \exp\left(-\frac{\omega(i)}{c}\right) \tag{3.12a}$$

[6]The period of state i is the greatest common divisor of all the n's such $P_{ii}^n \geq 0$

[7]The reader is deferred to the excellent book of Aarts&Koorts [2] for the complete proofs of these theorems.

with

$$Z(c) := \sum_{j \in \mathcal{S}} \exp\left(-\frac{\omega(j)}{c}\right) \tag{3.12b}$$

Moreover, the algorithm converges asymptotically to $\mathcal{S}_{opt}$ in the following sense :

$$\lim_{c \to 0} q_i(c) = q_{i,opt} = \begin{cases} \frac{1}{|\mathcal{S}_{opt}|} & \text{if } i \in \mathcal{S}_{opt} \\ 0 & \text{otherwise} \end{cases} \tag{3.13}$$

As theorem (3.2) states it, only asymptotic convergence could have been proven, i.e. the stationary distribution can be made as close as intended to the optimal distribution by lowering the "temperature" c, but there is no proof that there exists a given c_{max} such $\forall c : 0 < c < c_{max}$, $\mathbf{q}(c) = \mathbf{q}_{opt}$. Even so, one of the hypothesis of convergence was that the Markov chain was homogeneous, so we would have to set the temperature c at a very low value from the start.

One (big !) caveat of this approach is that the convergence rate is very low :

Theorem 3.3 *Let N be the minimal number of transitions to approximate the stationary distribution with tolerance $\epsilon > 0$, i.e :*

$$|\mathbf{a}(N) - \mathbf{q}(c)| < \epsilon \tag{3.14a}$$

Then we have :

$$N > \left(|\mathcal{S}|^2 - 3|\mathcal{S}| + 3\right)\left(1 + \frac{\ln \frac{\epsilon}{2}}{\ln\left(1 - \gamma^{|\mathcal{S}|^2}(c)\right)}\right) \tag{3.14b}$$

where

$$\gamma(c) := \min\{\lfloor P_{ij} \rfloor \mid i, j \in \mathcal{S}\} \tag{3.14c}$$

And the definition of the "norm" $\lfloor\rfloor$ is

$$\forall a \in \mathbb{R},\ \lfloor a \rfloor = \begin{cases} a & \text{if } a \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

So the convergence rate is inversely proportional to the second power of the states space's size. We recall that *one* state is only an assignment of all the variables so the states space's size is approximately the number of variables times the average number of values a variable can be set to. Actually, it is *at least* proportional to the quadratic size since the second term is always greater then 1 if we suppose that we take $\epsilon < 1$. If the set of allowable values is not enumerable[8], resolution along these lines is hopeless.

3.3 The Inhomogeneous Scheme

3.3.1 Markov Chains Again

In this section, we will introduce an improvement to the previous scheme by assessing that the Markov chain associated with the simulated annealing process would no longer be homogeneous. We allow then that the transition matrix depends on "transition" time. However, such Inhomogeneous Markov chain is actually build as an infinite sequence of finite homogeneous Markov subchains with decreasing temperature c.

Definition 3.1 *Let L be the length of each homogeneous Markov subchain. Then the temperature sequence c_k is taken as being piecewise constant such that :*

$$\forall k,\ lL < k < (l+1)L,\ c_k = \sigma_l \tag{3.15a}$$

Moreover, the σ's must satisfy

$$\forall l,\ \sigma_{l+1} \leq \sigma_l \tag{3.15b}$$

$$\lim_{l \to +\infty} \sigma_l = 0 \tag{3.15c}$$

As already mentioned in subsection (3.2.1), ergodicity is an important feature of a stochastic system, it ensures at least, that the final/regime behavior of the system is not constraint by the initial state. A more precise definition of it is :

[8] A priori, a flight's delay can take any value, which implies that the states space's size is infinite

Definition 3.2 *(Weak ergodicity) A inhomogeneous Markov chain is weakly ergodic if :*

$$\forall i, j, l \in \mathcal{S},\ \forall m > 0\ :\ \lim_{k \to +\infty} (U_{il}(m,k) - U_{jl}(m,k)) = 0 \tag{3.16a}$$

where

$$U_{ij}(m,n) := P(X(n) = j \mid X(m) = i) \tag{3.16b}$$

In other words, $X(k)$ becomes independent of $X(m)$ whatever the later's value is, as $k \to +\infty$.

Definition 3.3 *(Strong ergodicity) A inhomogeneous Markov chain is strongly ergodic if there exists a vector $\mathbf{q}_{opt}$ such :*

$$\forall i, j \in \mathcal{S},\ \forall m > 0\ :\ \lim_{k \to +\infty} U_{ij}(m,k) = q_{j,opt} \tag{3.17}$$

We notice that strong ergodicity implies[9] convergence in distribution, i.e.

$$\lim_{k \to +\infty} P(X(k) = i) = q_{i,opt} \tag{3.18}$$

So to ensure convergence of the Markov chain, we have to ensure its strong ergodicity. We remark that, for homogeneous Markov chains, there is no difference between strong and weak ergodicity.

Theorem 3.4 *An inhomogeneous Markov chain is strongly ergodic if :*

- *It is weakly ergodic.*
- *It verifies the* balance *equation for all k's :*

$$\forall k,\ \exists \mathbf{q}^k\ :\ \left(\mathbf{q}^k\right)^T = \left(\mathbf{q}^k\right)^T \mathbf{P}^k \tag{3.19}$$

- *The sequence $\left(\mathbf{q}^k\right)_k$ must satisfy :*

$$\sum_{k=1}^{+\infty} \mid \mathbf{q}^k - \mathbf{q}^{k+1} \mid < \infty \tag{3.20}$$

[9] It implies also weak ergodicity

And in the same vein as for the homogeneous case, we can prove that the Markov chain associated to the simulated annealing algorithm as defined by (3.6), (3.7) and (3.15) converges asymptotically towards the optimal distribution $\mathbf{q}_{opt}$.

Theorem 3.5 *If the following conditions are satisfied :*

- *The length L of a homogeneous Markov subchain must be such :*

$$L \geq \max_{j \in \mathcal{S}} \left\{ \min \left\{ p \mid \exists i \in \mathcal{S}_{opt},\ P_{ij}^{p} > 0 \right\} \right\} \tag{3.21}$$

- *The temperature sequence $(\sigma_l)_l$ must satisfy :*

$$\sigma_l \geq \frac{\Delta(L+1)}{\log(l+2)} \tag{3.22a}$$

where

$$\Delta := \max_{i,j \in \mathcal{S}} \{ \omega(j) - \omega(i) \mid j \in \mathcal{S}_i \} \tag{3.22b}$$

Then the inhomogeneous Markov chain associated to the simulated annealing algorithm converges towards its optimal distribution $\mathbf{q}_{opt}$, $\mathcal{S}_i$ being the neighboring of the state i and $\chi_{\mathcal{S}_i}$ its characteristic function:

$$\mathbf{q}_{opt} = \frac{1}{|\mathcal{S}_i|} \chi_{\mathcal{S}_i} \tag{3.23}$$

To summarize, each homogeneous subchain should be long enough to allow escaping from the set of optimal solutions $\mathcal{S}_{opt}$ to any other state, so at least, we are sure that the system will not be trapped at a local minimum during the annealing sub-process. And the temperature c_k should be lowered carefully to avoid the same issue.

It is noteworthy to observe that the above theorem states only sufficient conditions, not necessary ones, so actually the sub-chain's length could be lowered or the temperature decrease rate be raised. Some authors [1] have even given sufficient and necessary conditions for asymptotic convergence, so more accurate values of parameters L and c_k are straightforwardly derived

but it is of little operational help : As the given optimization problem is (often) not well known, the shape of its cost function nasty, etc ... so even assessing a coarse magnitude of parameters L and Δ is painful.

Some other authors [12] have also proven more general sufficient conditions, i.e. conditions which hold were the acceptance and generating probabilities are not the same as those chosen in (3.6) and (3.7).

Theorem 3.6 *Let $(\mathcal{S},\omega)$ be a combinatorial optimization problem. Let G_{ij} and A_{ij} be, respectively, the generating and acceptance probabilities of the simulated annealing algorithm associated to it. Let P_{ij} the transition probabilities of the inhomogeneous Markov chain to the simulated algorithm as defined in (3.8).*
If the following conditions are satisfied :

$$\forall i,j \in \mathcal{S},\ \exists p \geq 1\ :\ \exists \{l_0,\ldots,l_p\} \subset \mathcal{S} \text{ such } l_0 = i,\ l_p = j \text{ and } \forall k,\ 0 \leq k \leq p-1,\ G_{l_k l_{k+1}}(c_k) > 0 \tag{3.24a}$$

$$\forall i,j \in \mathcal{S},\ \forall k > 0\ :\ G_{ij}(c_k) = Gji(c_k) \tag{3.24b}$$

$$\forall i,j \in \mathcal{S},\ \forall k > 0\ :\ A_{ij}(c_k) = \begin{cases} 1 & \text{if } \omega(i) \geq \omega(j) \\ \in]0,1[& \text{if } \omega(i) < \omega(j) \end{cases} \tag{3.24c}$$

$$\forall i,j,m \in \mathcal{S} \text{ such } \omega(i) \leq \omega(m) \leq \omega(j)\ :\ A_{ij}(c_k) = A_{im}(c_k)A_{mj}(c_k) \tag{3.24d}$$

$$\forall i,j \in \mathcal{S} \text{ such } \omega(i) \leq \omega(j)\ :\ \lim_{c_k \to 0} A_{ij}(c_k) = 0 \tag{3.24e}$$

$$\sum_{l=0}^{\infty} (\mathbf{A}(\sigma_l))^{L+1} = +\infty \tag{3.24f}$$

where the c_k's and σ_l's are defined by (3.15), the length L by (3.21), $\mathcal{S}_i$ as the neighboring of state i and the $\mathbf{A}$'s as :

$$\mathbf{A}(c) := \min_{i,j \in \mathcal{S}} \{A_{ij}(c) \mid i \in \mathcal{S}, j \in \mathcal{S}_i\}$$

Then the inhomogeneous Markov chain associated to the simulated annealing algorithm converges asymptotically in the following sense :

$$\lim_{k \to +\infty} P(X(k) \in \mathcal{S}_{opt}) = 1 \tag{3.25}$$

A few explanations and useful comments should be given about this (lengthy) theorem.

- The two first conditions express that the generating mechanism is "total", i.e. any state is reachable from another in a finite time, and symmetric, i.e. if transitions $i \to j$ and $j \to i$ are proposed with equal probability.
- Third condition stress that any cost-improving transition is always accepted, any cost-hindering transition might be rejected.
- Fourth condition implies that the acceptance mechanism is somewhat transitive.
- Fifth condition stresses the fact that, at lower temperatures c_k, almost all cost-decreasing transitions are rejected.
- The length of a homogeneous Markov subchain should be high enough to allow, as previously explained, escaping from an optimal state to any other state.

Therefore this theorem states convergence conditions which are very close to the physical conditions observed with (real) annealing as described in the first part of this chapter. Moreover, this theorem ensures the convergence of the algorithm if we decide to use e.g. Gaussian or Cauchy distributions for the generating probabilities.

3.3.2 The Convergence Rate

Despite all the efforts, we were not able to assess full convergence of the (inhomogeneous) simulated annealing algorithm. Moreover, it can be shown that the convergence rate, even if improved by the introduction of the temperature decreasing scheme, is still too low for operational interest. Indeed, a bound has been derived by [10] :

Theorem 3.7 *Let N be the minimal number of transitions to approximate the optimal distribution with tolerance $\epsilon > 0$, i.e. :*

$$| \mathbf{a}_k - \mathbf{q}_{opt} | < \epsilon \tag{3.26a}$$

Then we have :

$$N = \mathcal{O}\left(\left(\frac{1}{\epsilon}\right)^{\frac{1}{\min(\alpha,\beta)}}\right) \tag{3.26b}$$

with

$$\alpha := \frac{1}{(L+1)|\mathcal{S}_i|^{L+1}} \tag{3.26c}$$

$$\beta := \frac{\min\{\omega(i) \mid i \in \mathcal{S} \setminus \mathcal{S}_{opt}\} - \omega_{opt}}{(L+1)\Delta} \tag{3.26d}$$

In practice, β is often observed that $\beta = \mathcal{O}(L+1)^{-1}$, so only the value of a matters for deriving the magnitude of the convergence rate. We see that it is still exponential *but* in the neighboring's size. So, as we have control over it, solving our optimization problem along these lines is not hopeless.

However, some researchers have tried some faster temperature schemes. The scheme we reviewed at extent in previous section is actually a logarithmic scheme :

$$\sigma_l \simeq \frac{\sigma_0}{\ln(l)}$$

for which a convergence proof has been given. But, it has been observed in practice, that adopting quicker schemes[10] [5, 9] could still converge. The problem is that *no* convergence proof have been assessed up to now. Actually, the convergence is achieved, when possible and observed, by careful tuning of the parameters and incorporating some heuristics inside the algorithm. For example, a "linear" scheme could be used $\sigma_l \simeq \sigma_0/0$, or even an exponential scheme as proposed by [5, 9] :

$$\sigma_l \simeq \frac{\sigma_0}{\exp(l)} \tag{3.27}$$

This scheme must be used with great care. Currently, it is used with two extra features : adaptativeness and reannealing. We remember that decreasing too quickly the temperature increases the probability of being trapped in a local minimum. So three heuristic rules are followed (For a longer explanation of this, cfr. [5]) to avoid such phenomenon :

[10] Fast temperature decreasing is also referred as *simulated quenching*, another correspondence with thermodynamics

Broad spectrum : The generation distribution must have a broader spectrum [11]. Intuitively, it is justified by the fact that, if the system is on the edge of being trapped in a local minimum, it would be wise to allow it to escape from it quickly. A constant distribution over the neighboring as (3.7) implies that the system would remain a longer time in the vicinity of the local minimum.

Reannealing : From time to time, it is necessary to increase the temperature so more cost-hindering transitions are accepted, with the same aim of preventing the system trapping in a local minimum. This method is sometimes referred as VFSR : Very Fast simulated Reannealing.

Adaptativeness : The aim is exploiting the knowledge we have of the cost function (if we can) by imposing different temperatures schemes, one for each dimension/variable[12].

An experimental implementation combining these last features has been developed by Ingber & al. and is claimed to be one of the most powerful combinatorial optimization algorithms, not only by its author, but also by other researchers. Nevertheless, as already said, great care and time are necessary for the fine tuning of such algorithm. Even so, no convergence can be ensured. The code of ASA (Adaptative Simulated Annealing) has been made publicly available by FTP [8].

3.4 The Finite-Time Scheme

We will, in this section, how the simulated annealing algorithm associated to the inhomogeneous Markov chain with the simple "standard" temperature scheme (3.5) could be implemented in practice. Indeed, even the inhomogeneous scheme is too slow for operational use, an optimal state is approached only at infinity. So we have to "truncate" this scheme in a clever way in order to obtain quicker convergence, but not at the expense of eventually finding a

[11]The spectrum's width of a distribution is measured by the ratio (standard deviation/maximum value) or (first momentum/maximum value), e.g. a Gaussian distribution with large standard deviation has a quite large spectrum, on the opposite, a Dirac distribution has no spectrum

[12]We assume that the states space's dimension is finite.

"good" semi-optimum. Practically, there is a trade-off between the accuracy of the computed optimal solution and the time needed to obtain it. Basically, we have to define four parameters :

- The initial state of the algorithm, i.e. which values must be assigned to the variables, what is the initial value of the temperature.
- The temperature decreasing function : At which rate the temperature be decreased and when.
- The stop criterion : When do we estimate that the system is close enough to an optimal solution.
- The length of a homogeneous Markov sub-chain : Too short chains can cause the system being trapped in a local minimum, on the other side, too long chains hinder the convergence speed.

3.4.1 Choosing an Initial State

First of all, the variables can be set to any allowed values. What matters is the initial temperature σ_0 which will be chosen high enough in order to accepting all the generated transitions. This is achieved by computing the acceptance ratio :

$$\chi \simeq \frac{m_\uparrow + m_\downarrow \cdot \exp\left(-\frac{\delta\omega}{\sigma}\right)}{m_\uparrow + m_\downarrow} \tag{3.28}$$

with $m_\uparrow$, $m_\downarrow$ being, respectively, the number of cost-increasing and cost-decreasing generated transitions and $\delta\omega$ the mean cost difference of the $m_\downarrow$ transitions. So the system is started with a low temperature then a sequence of n trials is made and χ is computed by counting the number of accepted transitions over the proposed transitions after *each* trial. The same is made for $m_\uparrow$, $m_\downarrow$ and $\delta\omega$. Inserting these values into (3.28) and solving it gives an estimate for σ. Updating σ after each trial leads to the value of σ_0 at the end of the sequence. A few hundreds runs are enough to obtain convergence. Physically, this corresponds to the melting up of the material, so the state of the system is very disordered.

3.4.2 The Temperature Decreasing Function

The aim is to decrease the temperature slowly enough to ensure that quasi-equilibrium is reached at the end of each Markov sub-chain. By quasi-equilibrium, we mean that, within each homogeneous sub-chain, the system state distribution can be as close as desired to the stationary distribution $\mathbf{q}(\sigma_l)$ (cfr. sect. 3.2.2).

Moreover, we desire that two successive stationary distributions are close enough :

$$(1+\kappa)^{-1} \leq \frac{|\mathbf{q}(\sigma_l)|}{|\mathbf{q}(\sigma_{l+1})|} \leq (1+\kappa) \tag{3.29}$$

with κ being the temperature decreasing parameter. Satisfying these equation leads to the following condition :

$$\sigma_{l+1} \simeq \frac{\sigma_l}{1+\frac{\sigma_l \ln(1+\kappa)}{3\Sigma(\sigma_l)}} \tag{3.30}$$

with $\Sigma(\sigma_l)$ being the approximated standard deviation of distribution $\mathbf{q}(\sigma_l)$.

3.4.3 The Stop Criterion

The algorithm is stopped when the cost of the current system state is supposed close to the optimal cost. More precisely, we require that, in average, the difference between $\omega(\sigma_l)$ and ω_{opt} is small when compared to ω_∞, i.e. the maximum[13] average cost.

$$\frac{|\omega|_{\sigma_l} - \omega_{opt}}{|\omega|_\infty} < \epsilon \tag{3.31}$$

where ϵ is referred as the stop parameter. Satisfying this criterion requires that :

$$\frac{\sigma_l}{|\omega|_\infty} \frac{\partial |\omega|_\sigma}{\partial \sigma}(\sigma_l) < \epsilon \tag{3.32}$$

However, computing the derivative of $|\omega|$ is a sensitive task. It is suggested to smooth the values of $]omega$ prior to approximate the derivative with classical methods [2].

[13] Actually, $|\omega|_\infty := \lim_{\sigma \to \infty} |\omega| = \frac{1}{|\mathcal{S}|} \sum_{i \in \mathcal{S}} \omega(i)$ as $|\omega|_\sigma := \sum_{i \in \mathcal{S}} \omega(i) q_i(\sigma)$

3.4.4 The Markov Length

The length of a homogeneous Markov sub-chain is chosen big enough to allow the system visiting the entire neighboring of a state. It can be proven that is sufficient to choose :

$$L = |\mathcal{S}_i| \tag{3.33}$$

3.4.5 The Convergence Rate Again

Beside the fact that the finite-time scheme previously described is more heuristic then rigorous, it is at least faster : its complexity is proportional to the logarithm of the states space's size. More precisely, we have that the minimal number N of transitions to satisfy the stop criterion (3.31) is :

$$N = \mathcal{O}(\tau\, L\, \ln(|\mathcal{S}|)) \tag{3.34}$$

where τ is the time for computing one transition[14]. The (only) remaining problem is that there is no *a priori* confidence that the eventual solution is an optimal one. Only *a posteriori* simulations can assess it.

3.5 The Tabu Search

It is a technique close to simulated annealing but where the exploration of the neighboring is done in a more "clever" way or, at least, in a less stochastic way. The main difference lies in the acceptance mechanism : Instead of sampling randomly the neighboring of a state i, the algorithm keeps an updated list of the recent transitions. This list is referred as the *tabu* list as the transitions in this list are forbidden for a certain time, i.e. the system cannot make twice the same transition within a certain time window.

However, the system can be a little improved by adding *aspiration* criteria, i.e. if a tabu transition satisfies one of this aspiration criteria, its tabu status is overridden. The idea hidden in these features is still to prevent the system from being trapped in a local minimum.

The algorithm has the following structure :

[14]So it is clever to formulate the cost function in an incremental way to avoid massive re-computations

1. Choose an initial state i, either randomly or by exploiting the knowledge of the system.
2. Generate randomly a list $(T_k)_k$ of transitions $i \rightarrow j$, $j \in \mathcal{S}_i$.
3. Pick up the best[15] transition T_k. Is T_k a tabu transition ?
4. If yes, check if T_k satisfies one of the aspiration criteria $(\mathcal{C}_l)_l$, i.e. if $\exists l : \mathcal{C}_l(T_k)$ is true. If yes, the transition is no longer tabu.
5. If T_k is tabu then go back step 3, else go next step.
6. Apply transition T_k.
7. Update parameters : Tabu list (mandatory), Aspiration criteria (optional), system state.
8. Go back step 2 until a solution is found or time and resources capacities are exceeded.

In practical implementations, people add some extra features as short-term and long-term memory tabu lists to which different aspiration criteria are applied. Updating the tabu-list can be achieved in different ways : managing it as a LIFO pile of constant length, giving different tabu coefficients to the transitions, varying the length of the tabu list e.g. a bigger length at the beginning to ensure full exploration then decreasing it, as the temperature control scheme in simulated annealing, etc...

These technique has the advantage of being much flexible then e.g. simulated annealing or more classical optimisation algorithms, more heuristics can be incorporated in it, but has two major drawbacks : Tuning the parameters is extremely sensitive and there is no general convergence proof.

3.6 Conclusion

To conclude, it is obvious that solving combinatorial problems is intractable by classical methods as these problems are often, in practice, NP-complete

[15]By "best", we mean the transition which leads to the highest decrease of the cost function

and still difficult by using stochastic exploring algorithms such simulated annealing or tabu search. Actually, some general remarks can be made :

- Formalizing the problem is itself a sensitive task. Clearly a problem like optimizing the total delay of Europe's air traffic as modelled in (2.11) with 60 million variables is almost infeasible. Ther is must be a trade-off between the accuracy of the model and its size/simplicity.
- Some algorithms have been developped for solving these difficult optimization problems, almost all of them are stochastic (simulated annealing, tabu search, genetic methods, etc ...). Moreover, only asymptotic convergence can be proven for simulated annealing and genetic methods. Tuning the parameters is a very sensitive task.
- In practical implementations, some heuristics is incorporated inside the algorithms in order to speed up and to exploit the existing knowledge of the system to optimize. However, getting a clear understanding of the algorithm's behavior is henceforth less easy. The tabu method is more a framework for using heuristics rather than an optimization algorithm in the classical sense.
- For the above reasons, practical implementations of the algorithms end up with solutions but there is no warranty that they are optimal !

In two words, it is hard to assert a priori that any of the previously exposed stochastic algorithms could, with reasonable confidence, solve a combinatorial optimization problem. Such algorithms should be considered as frameworks for developing *ad hoc* implementations on a trial $ error basis.

Chapter 4

Solving CASA

The purpose of this chapter is the derivation of approximated models from the general CASA model (2.11). However, it reader should bear in mind that these approximated models will not be equivalent to the general one or to each other. Actually, each model is derived in order to make the full benefits of chosen the method/algorithm. The three models which will be exposed in this chapter make different compromises between speed, optimality, size and flexibility. From an operational point of view, a fast and reliable algorithm is clearly preferred over an optimal slower one. Flexibility as a criterion is less often required but can be fruitful when the requirements are not clearly stated or change regularly.

4.1 Reformulating CASA as a Combinatorial Problem

Reformulating CASA as a combinatorial problem requires three elements to be defined :

The state space : Basically, a state of the system is simply an setting of the system's variables, i.e. a list of pairs (variable/value). As a consequence, only the minimum number of variables should be retained to keep the state space's size low. Moreover, the variables should belong to a *finite countable* set. If the variables are allowed to take any value in

$\mathbb{R}$ or $\mathbb{N}$, then the state space's size is infinite which implies intractability of the problem.

The actual cost function : It differs from the primary cost function, i.e. the function defined in the general model by the fact that the constraints in (2.11) are incorporated in it. Moreover, different penalties can be assigned to each primary constraint. More precisely, if the primary problem is of the general form :

$$\min \quad f(\,, x_i, \,) \tag{4.1a}$$
$$\text{w.r.t.} \forall j,\ c_j(\,, x_i, \,) = 0 \tag{4.1b}$$
$$\forall i,\ x_i \in \mathbb{N} \tag{4.1c}$$

then the combinatorial version will be :

$$\min \omega(\mathbf{k}) = f(\mathbf{k}) + \sum_j \pi_j c_j(\mathbf{k}) \tag{4.2a}$$
$$\mathbf{k} \in \mathcal{S} \tag{4.2b}$$

where the π's are the penalty coefficients, $\mathbf{k} = (x_i)_i$ is a state of the system and $\mathcal{S} = \mathbb{N}^n$ the state space. It is wiser to do so otherwise because each generated transition should be checked if it does not lead to a forbidden state, i.e a state which does not satisfy the primary constraints c_j, prior to being proposed to the acceptance mechanism, checking which could be achieved by evaluating *all* the primary constraints. In addition, evaluating ω should be incremental, i.e. evaluating the cost's difference of the modification of a variable's value should not imply recomputing the whole cost-function. More rigorously, the term $\omega(\mathbf{k}+\delta_i)-\omega(\mathbf{k})$, where δ_i is the modification of x_i should remain simple or, even better, depends only on x_i, i.e. $\frac{\partial}{\partial x_j}\omega(\mathbf{k}+\delta_i)-\omega(\mathbf{k}) = 0, \forall j \neq i$. The second term in (4.2a) should not always be linear in the constraints or the variables. However, it is the simplest way to achieve incrementality of the cost function. We notice that incorporating the primary constraints in ω has the advantage of some flexibility in the model. Some strong constraints can be given higher penalties or to link the "cost" of constraint violation to the actual primary cost[1]. Tuning the

[1]In the case of CASA, it is equivalent to say that an overload of one flight over one regulation costs as much as e.g. a 60' delay.

penalty coefficients is sensitive as too low values would (in most cases) speed up the algorithm but the found solution would be irrealistic, i.e. too much constraints would be violated.

The neighboring structure : As this is the main element which drives the simulated annealing convergence, it should be chosen carefully. If the neighboring's size is too large, convergence would be be delayed as shown by the convergence rate's estimates (3.3, 3.7 & 3.34).

For the above reasons, we propose the following combinatorial version which is derivated in a natural way from (2.11) :

$$\min \quad \omega^t = \sum_{f \in \mathcal{F}^t} c_f^t \cdot d_f^t + \sum_{\substack{\forall r \in \mathcal{R}^t \\ \forall s \in \mathcal{S}_r^t}} p_{rs}^t \frac{\lfloor \epsilon_{rs}^t \rfloor}{\kappa_{rs}^t} + q_{rs}^t \left(\frac{\lfloor \epsilon_{rs}^t \rfloor}{\kappa_{rs}^t} \right)^2 \tag{4.3a}$$

$$\mathcal{S}^t = \left\{ (d_f^t)_f \mid f \in \mathcal{F}^t \right\} \tag{4.3b}$$

with the d_f's such :

$$\forall f \in \mathcal{F}^t, \forall t > 0 \; : \; d_f^t \in \{0, \cdots, d_{max} - 1\} \tag{4.3c}$$

where :

- p_{rs}^t is the primary constraints violation penalty.
- q_{rs}^t is the secondary constraints violation penalty.
- $\frac{\lfloor \epsilon_{rs}^t \rfloor}{\kappa_{rs}^t}$ is the slot's relative overload.

and the $\epsilon_r s^t$ are considered as functions of the delays d_f^t.

This version exhibits the required features :

Flexibility : One can balance the cost of delaying a flight versus. the relative overloading of a slot. The linear term in ϵ express that overloads are to be avoided, the quadratic term has been integrated to avoid "bursty" overloads. Actually, it is preferred to have a 5% overload for one hour rather then a 30 % overload for 15'.

Incrementality : Proving incrementality of (4.3a) in a rigorous way is not obvious due to the complex relationship between the ϵ's and the delays d_f. However, it is intuitively the case : Modifying a flight's delay is naturally incremental (linear !) in the cost term $c_f \cdot d_f$, but the second and third terms, we should notice that only a few slots are concerned. Actually two slots per regulation are concerned, the one which was allocated to the flight before the transition and the one after. As a regulated flight crosses in average 2.5 regulations, only 5 slots' overloads have to be evaluated.

Reasonable size : We observe that the problem contains, hopefully, a bit less then 60 million variables. As already mentioned (4.3c), a delay can take d_{max} values. As a delay is rarely (at least, up to now) over 4 hours, the size of the state space $\mathcal{S}$ is approximately :

$$|\mathcal{S}^t| = |\mathcal{F}^t| \cdot d_{max} = 6000 \cdot 240 \simeq 1.5 \cdot 10^6 \tag{4.4}$$

Moreover, we impose that a transition could change only *one* flight's delay. The delay modification must belongs to $\{-\delta_{max}, \delta_{max} - 1\}$. So if we take $\delta_{max} = 5$', the neighborhood's size is :

$$|\mathcal{S}_i| = 2 \cdot |\mathcal{S}_i| \cdot \delta_{max} \simeq 60.000 \tag{4.5}$$

4.2 Convergence Estimates

Recalling the expressions of the convergence rate magnitude in the homogeneous (3.3), inhomogeneous (3.7) and finite-time (3.34) schemes, we can derivate in a straightforward way the actual magnitudes where N^t stands for the minimal number of transitions at time t to achieve convergence :

Homogeneous

$$N^t \geq \mathcal{O}\left(d_{max}^2 |\mathcal{F}^t|^2\right) \mathcal{G}\left(\epsilon', |\mathcal{F}^t|^2\right) \tag{4.6a}$$

with

$$\lim_{\substack{\epsilon' \to 0 \\ |\mathcal{F}^t| \to \infty}} \mathcal{G}\left(\epsilon', |\mathcal{F}^t|^2\right) = +\infty \tag{4.6b}$$

So, if we desire a higher optimality on big-sized problem as CASA is, the computation time is exponential and therefore solving these model is not realistic in operational use.

Inhomogeneous

$$N^t \geq \mathcal{O}\left(\exp\left(-(L+1)\cdot\left(2\cdot|\mathcal{F}^t|\cdot\delta_{max}\right)^{L+1}\right)\ln(\epsilon')\right) \tag{4.7a}$$

with

$$L \leq 2\cdot|\mathcal{F}^t|\cdot\delta_{max} \tag{4.7b}$$

Actually, giving an upper bound on L is not obvious. We recall that is the length of a homogeneous Markov sub-chain. In practice, L is set to the value of the neighborhood's size $|\mathcal{S}_i|$. The problem is still exponential in size but, as previously noted, it is the neighborhood's size which plays the key role. So by varying δ_{max}, we can modify the asymptotic convergence rate.

Finite-Time

$$N^t \geq \mathcal{O}\left(L\ln\left(|\mathcal{F}^t|\,d_{max}\right)\right) \tag{4.8}$$

So the finite-time is less time consuming as it is logarithmic in the state space's size. However, it should be reminded that there is no warranty that the algorithm ends up with an optimal solution.

4.3 Some Other Strategies

4.3.1 Constraint Programming : The ILOG Study

This section is devoted to two other strategies which, in practice, opposite goals. Constraint programming is a technique for "quick" resolution of constraints. 0-1 linear programming is a powerful tool for formalizing complex problems and finding an optimal or near-optimal solution but is as the expense of resolution speed. We will present here briefly the works of two teams which are currently working also on the Slot Allocation Problem. These two

teams have already carried some simulations of CASA but only in the static case, i.e. when all the information is available at start up[2].

Basically, constraint programming, also referred as CLP(fd)[3] : Constraint Logical Programming (finite domain), contains three elements :

Value List : The algorithm keeps and updates for *each* variable a list of allowed values. In CLP(fd), it consists often of a chained list of integer intervals. If the variable's valuelist contains only one value, it is set otherwise it is free. If at any instant, there exists a variable with an empty valuelist, the problem is unsolvable.

Instantiation : A free variable is is chosen and instantiated to an allowed value. Choosing the variable and its value is a sensitive task. In practice, a lot of heuristics is incorporated in this part of the algorithm. In the ILOG implementation, the user can even defining himself the instantiation strategy. Without tuning, the problem can remain unsolvable.

Propagation Mechanism : Each time a variable is instantiated, its value is propagated i.e. for each other variable, the system eliminates the imposssible values. The idea is to reduces back-tracking[4].

The algorithm stops when a variable's valuelist becomes empty (the system has no solution) or when all the variables have been instantiated.

The model derived by ILOG [13] differs somewhat of the general model (2.11) by considering two sets of constraints : the hour constraints and the interval constraints. From (2.10), we derive :

$$\epsilon^t_{rs} + \kappa^t_{rs} = \sum_{f \in \mathcal{F}^t} \leq \kappa^t_{rs} u^t_{rsf} \tag{4.9}$$

As usually the capacities are defined on an hour basis, the first set of constraints has been expressed for "slots" with duration $T = 60'$ and $\kappa^{t,h}_{rs}$ set

[2]Currently, only two thirds of the flight plans are registered before starting the regulations of the current day.

[3]Actually, constraint can be defined on real numbers. The nuance between constraint programming and pure CLP is that CLP use a Prolog-like syntax

[4]Remember that, in Prolog, back-tracking is considered as the major cause of its poor time performance.

to its hour values. The second set of constraints is expressed for "slots" with duration $T = \theta$, usually 10' with $\kappa_{rs}^{t,i} = \lceil \frac{\kappa_{rs}^{t,h}}{\Theta} \rceil$[5]. As the interval capacities are rounded up, there is no real redundancy between the two sets.

As the ILOG implementation allows the user to define an instantiation strategy to speed up the algorithm, it would be wiser to use it. Two strategies are proposed :

Chronological scheduling : Given an enumeration of the flights, the algorithm tries to instantiate all the flight delays in the order specified. Given a flight, it tries to instantiate its delay to the minimum value and increases it if it leads until no more constraint are violated.

Reducing overcharges first : The algorithm select the most overloaded (violated) constraint first. It selects then, among the flights linked to that constraint, the one whose delaying leads to the smallest increase of cost. Given that flight, the algorithm tries to push it out of the constraint. If it fails, it decreases the flight's delay until no more constraints are violated. Estimating in advance what would be the effect of a flight's delaying on the cost function is difficult, so ILOG has proposed an ad hoc heuristics [13].

Obviously, the second strategy is less naive then the first one and gives better results. Simulations have been carried on a set of 17 regulations and 1991 flights of which a large amount of "combined" flights, i.e. flights which cross more then one regulation. CASA gives a total delay of 40.000', the first strategy decreases it to 32000' then the second strategy improves it to 21000', hence a relative reduction of 40% ! Moreover, solving time is of the order of a second on SunSparc 20 workstation. However, it should be reminded that CASA solves the *dynamic* problem which is more constrained then the static one, so the real benefits would be less then 40%.

4.3.2 $0-1$ Linear Programming : The Vranas & Bertsimas Study

The models defined by Vranas [11] can be derived from model (2.11) by making the hypothesis that all the slots have the same duration T and that

[5] $\lceil \rceil$ is the round-up function : $\lceil a \rceil = n+1$ if $a \in [n, n+1]$, $a \in \mathbb{R}, n \in \mathbb{N}$

the flights enter in the regulation at the *beginning* of the slot. Hence the δ's in (2.11) are set to 0. It is the same as saying that the slot's duration is the new time unit.

Given these hypothesis, it is easy to notice that (2.11) can be transformed in a 0-1 linear problem in the time-stamps u_{rsf} :

$$\min \omega = \sum_{f \in \mathcal{F}} c_f \cdot \left(T \sum_{s=1}^{*} (s-1)\, u_{rsf} - \hat{\tau}_{rf} \right) \quad \text{cost function} \tag{4.10a}$$

$$\text{w.r.t.} \sum_{f \in \mathcal{F}} u_{rsf} \leq \kappa_{rs} \quad \text{slot's overload} \tag{4.10b}$$

$$T \sum_{s=1}^{*} (s-1)\, u_{rsf} \geq \hat{\tau}_{rf} \quad \text{positive delays} \tag{4.10c}$$

$$\sum_{s=1}^{*} u_{rsf} = 1 \quad \text{only one slot allocated to } f \tag{4.10d}$$

where the subscripts^t have been dropped since only the static case has been reviewed. We remark that the problem still contains 60 million binary variables but is now linear. Vranas & Bertsimas use then the classical method of linear relaxation of the integer problem, method which gives near-integer (in this case, binary) solutions of the problem (4.10).

However, the figures given by the simulations are less hopeful then ILOG's figures are. Simulations were carried on various sets ranging from 2293 flights and 25 regulations to 1808 flights and 16 regulations, the later set containing more then 50% of combined flights. If we examine more carefully the worst case with 50% of combined flights, which is the more realistic[6] one, we observe only a 10% improvement of the total delay. Moreover, the computation time is about 7400", almost 2 hours on a SunSparc 20 workstation ! In the dynamic case, this method is of no operational use. If we look at better figures, the total delay decreasing can be up to 40% but the computation time remains of the same magnitude.

[6]If it is not realistic today, it will be so tomorrow

4.4 Conclusion

It is difficult to draw some definite conclusions of the material exposed above. We can remind that each of the presented methods (0-1 linear programming, simulated annealing and constraint programming) make different compromises between speed and optimality. In theory, 0-1 linear programming finds an optimal solution even it takes ages for achieving it. On the opposite, ILOG's constraint programming as quickly as possible a solution which satisfies the constraints without regards for optimality. In practice, things are a bit different. Using linear relaxation in integer programming hinder optimality since it is unlikely that all variables would be set to non-integer values. Tuning of the strategy in constraint programming allows the user to solutions which are thought to be close to optimality. Simulated annealing or tabu search are more flexible and are expected to be optimal *and* fast but are quite difficult totune.

Reviewing the figures given by the two methods leads to perplexity. Although the test cases are similar (around 18 regulations, 1900 flights of which 50% are regulated, the workstation is a SunSparc 20), there is a little discrepancy with the theory : If the 0-1 linear programming is actually much slower then constraint programming, it gives in practice worse value of total delay. But concluding that constraint programming is the effective method is too early as only a few simulations have been done.

In two words, solving the CASA problem seems is not obvious as many strategies are possible but no one has proven to be the most accurate. Further models should be developped and tested on a *standard* base of sets.

Conclusion

The best way to conclude is perhaps to review the initial objectives and examine how they have been achieved if they were.

- A description of CASA has been given and a list of operational requirements/constraints has been laid down. However, this is a rather informal list, some points as the existence of sub-periods, pending rates, the concept of a slot,etc... have not been properly addressed. We add that the above list has been obtained by talking with people, reading the software documentation [4], even reading the code sources.

- A fairly general mathematical model of CASA has been developed. Even if it is intractable in its present form, it provides a basis from which the other models have been derivated in a natural and straightforward way.

- A thorough explanation of the principles of simulated annealing has been achieved. The author stressed the point that for a combinatorial problem as CASA is, there is no warranty of solving it in a reasonable time. Only extensive simulations can assess the strength and qualities of a method. Heuristics have to be incorporated in the solving algorithm to speed it up.

- A comparison have been made between ILOG and Vranas models. Vranas model is very time-consuming, so its operational use is doubtful but it should in theory be useful for calibrating the other methods and models. ILOG model is powerful and quick and even achieves good figures in finding a close-to-optimal solution. Results are in little discrepancy with the theory as the most optimal solutions have been found by ILOG and not by Vranas. However, as the test cases were not exactly the same and only a few simulations have been carried out, concluding that ILOG is the most effective might be early.

- What lacks is the development of the prototype which would have implemented the simulated annealing technique to assess its qualities in

optimizing CASA. Actually, the prototype is still in its infancy and will remain in it for a while. The major reason is that the author has attempted from the beginning to develop an interface with the current implementation of CASA to solve directly the dynamic case. A bit of modesty would have, maybe, given more concrete estimates on the static case.

The author is convinced that stronger support for this fields of research will be necessary. As the air traffic grows up by 10% a year, needs for more cost-effective and optimal traffic management policies would become more and more stringent.

The author pleases the readers to be indulgent with the style of this thesis. As the author knows his audience ranges from hard-liner mathematicians to confirmed hackers and has himself a bakcground in physics and thermodynamics, choosing the level of rigour and formality was difficult. Moreover, some sentences could have been written in a surprising or, at least, innovative English.

The author has been however very pleased to study this problem. It has proven to be useful and instructive to work on a practical case and to confrontate it with the theories he has learned.

List of Acronyms

As normal human beings do not speak often this language...

AO : Air Operators. The air companies.

AR : AiRports. A yet common concept.

ATC : Air Traffic Control. The centre which manages air traffic over a given sector.

ATFM : Air Traffic Flow Management. The global European traffic management policy.

CASA : Computer-Aided Slot Allocation. The algorithm reviewed in this thesis.

CEU : Central Executive Unit. The operational unit of ATFM.

CTOT : Computed Take-Off Time. Take-off time imposed by CASA.

CTO : Computed Time-Over. Imposed entrance time in a regulation.

EOBT : Estimated Off-Block Time. Intended time for leaving the airport's main building. The difference with ETOT is the time to access the runway.

ETOT : Estimated Take-Off Time. Intended take-off time.

ETO : Estimated Time-Over. Intended entrance time in a regulation.

FIR : Flight Information Regions. Somewhat related to ATC.

ID : Flight IDentification. To identify a flight.

IFPL : Initial Flight PLan. The first flight plan submitted to CASA.

IOBT : Initial Off-Block Time. Somewhat related to EOBT.

LBBR : Identifier of Brussels International Airport (Latin europe, Belgium, BRussels).

NAVAID : Navigation points. Entrance points, gates, round-abouts, ...

RDY : ReaDY message. The flight will accept any SIP.

RPL : Repetitive Flight Plan. A flight which take-ff on a regular (daily) basis.

SAM : Slot Allocation Message. Sent by CASA, first proposition of a slot.

SIP : Slot Improvement Proposal. Sent by CASA if a better slot has been found.

SIT : Slot Issuance Time. After this time, the flight's slot is "frozen".

SLC : SLot Cancellation. Sent by CASA, the flight's slot is cancelled.

SPA : Slot Proposition Acceptance. The flight accepts the proposed slot.

SRJ : Slot ReJection message. The flights rejects the proposed slot

SRM : Slot Revision Message. Sent by CASA, the newly proposed slot is "frozen".

SRR : Slot Revision Request. The flight requests a new slot.

TACT : TACTical System. The system in which CASA is embedded. Serves as an interface betwen CASA and the users (AO, CEU).

UIR : Unit Information Regions. Somewhat related to the FIR.

SIP : Slot Improvement Proposal. Sent by CASA, a better slot is proposed.

Bibliography

[1] Hajek B. Cooling schedules for optimal annealing. *Mathematics of Operations Research*, 13, 1988.

[2] Aarts E. and Koorts J. *Simulated Annealing and Boltzmann Machines.* John Wiley & Sons, 1989.

[3] EUROCONTROL. *Tactical System Software Requirements*, 1.0 edition, 1995. Part : Interface to Casa.

[4] EUROCONTROL. *Tactical System Software Requirements*, 1.0 edition, 1995. Part : Computer Assisted Slot Allocation.

[5] Ingber L. Very fast simulated reannealing. *Journal of Mathematical & Computer Modelling*, 12, 1989.

[6] Ingber L. Simulated annealing : Practice versus theory. *Journal of Mathematical & Computer Modelling*, 18, 1993.

[7] Ingber L. Adaptative simulated annealing : Lessons learned. *Journal of Control & Cybernetics*, 1995. To be published.

[8] Ingber L. Asa code public archive. ftp://ftp.alumni.caltech.edu, 1996.

[9] Ingber L. and Rosen B. Genetic algorithms and very fast simulated reannealing. *Journal of Mathematical & Computer Modelling*, 16, 1992.

[10] Romeo F. Mitra D. and Sangiovani-Vincentelli A. Convergence and finite-time behavior of simulated annealing. *Advance of Applied Probabilities*, 1986.

[11] Vranas P.B. Optimal slot allocation for european air traffic flow management. *To be published*, 1996.

[12] Anily S. and Federgruen A. Simulated annealing methods with general acceptance probabilities. *Journal of applied probability*, 1987.

[13] ILOG S.A. Air traffic allocation module based on ilog solver. Technical report, ILOG S.A., 1995. Unpublished.

[14] Bertsimas D.J. Vranas P.B. and Odoni A.R. Dynamic ground-holding policies for a network of airports. *Transportation Science*, 1995. To appear.

[15] Bertsimas J.D. Vranas P.B and Odoni A.R. The multi-airport ground-holding problem in air traffic control. *Operations Research*, 42(2), 1994.