An Advanced Algorithm for Air Traffic Flow Management

by

Guglielmo Guastalla

Laurea in Scienze Statistiche ed Economiche, Università degli Studi di Padova (1995)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science in Operations Research

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 1997

Signature of Author.....
Department of Electrical Engineering and Computer Science
22 May 1997

Certified by.....
Amedeo R. Odoni
T. Wilson Professor of Aeronautics and Astronautics
Professor of Civil and Environmental Engineering
Thesis Supervisor

Accepted by.....
Thomas L. Magnanti
George Eastman Professor of Management Science
Professor of Electrical Engineering and Computer Science
Co-Director, Operations Research Center, MIT

An Advanced Algorithm for Air Traffic Flow Management

by

Guglielmo Guastalla

Submitted to the Department of Electrical Engineering and Computer Science
on 22 May 1997, in partial fulfillment of the
requirements for the degree of
Master of Science in Operations Research

Abstract

The increasing volume of air traffic in recent years has led to heavier use of airports and airways, while their capacities have not grown accordingly. This leads to a situation of congestion in air traffic networks, with departure delays and queues before landing, causing great economic losses to air carriers and potentially affecting air traffic safety. A way of reducing congestion is to adopt a Ground Holding policy, i.e., delay some aircraft before departure in order to avoid airborne delay. This thesis focuses on the static/deterministic Multi Airport Ground Holding Problem (MAGHP) under the assumption of insufficient capacities at arrival airports. Different approaches for the solution of the MAGHP are presented: (i) three alternative integer linear programming models; (ii) a heuristic algorithm based on priorities; (iii) a new algorithm based on the integration of the heuristic algorithm with one of the integer linear programming models. A comparison of their performance based on 39 test cases is provided. The integrated algorithm provides exact solutions in a much shorter time than previous algorithms proposed in the literature.

Thesis Supervisor: Amedeo R. Odoni
Title: T. Wilson Professor of Aeronautics and Astronautics
Professor of Civil and Environmental Engineering

Acknowledgments

I would like to thank Professor Amedeo Odoni and Professor Giovanni Andreatta for their continuous guidance and support, and the C. S. Draper Laboratories for funding my work.

Friends on both sides of the ocean helped me go through the tough moments and shared the happy ones. A huge *grazie* goes to all of them, but in particular to Susan, Mina, Abdoul, Arni, Amy and Bonfy.

Finally, I would like to thank all my family and Barbara, for everything.

Contents

Chapter 1

Introduction

1.1 Background

The air traffic network consists of two subsets: sectors and airports. Their capacities are determined by the maximum number of aircraft that air traffic controllers can safely handle during a given time period. For airports we can further distinguish between departure capacity (number of take-offs per unit of time) and arrival capacity (number of landings per unit of time).

The increasing volume of air traffic in recent years has led to heavier use of airports and airways, while their capacities have not grown accordingly. As a result, air traffic congestion is a critical problem in North America, Western Europe and East Asia. Congestion arises whenever the capacity of airports or sectors is exceeded. Usually it is caused by a reduction of capacity due to bad weather and the problem is more severe when this reduction coincides with peak traffic time.

Possible solutions may be, according to the time frame considered ([7]):

- Long–term: Construction of new airports and/or runways and advancements in air traffic control.
- Medium–term: Congestion pricing, i.e., pricing strategies leading to a more even distribution of demand; use of larger aircraft.
- Short–term: Implementation of Traffic Flow Management (TFM) strategies, whose goal

is to optimize the flow of aircraft in the air traffic network. The two main tools are *ground-holding* and *redistribution of flows in the airspace.*

TFM optimization models may be classified in ([4]):

- Ground Holding Problem (GHP): The objective is to minimize the total cost of delays *by absorbing airborne delays on the ground.* Depending on the number of airports considered, we have the Single Airport Ground Holding Problem (SAGHP) or the Multi Airport Ground Holding Problem (MAGHP). This thesis will focus on this latter problem, of which we give a detailed description in Section 1.2.
- The Generalized Tactical TFM Problem (GTFMP): In addition to ground delays, we consider the possibility of assigning airborne delays to flights, either at the arrival airport or in a sector (e.g., through speed reduction).
- The Traffic Flow Management Rerouting Problem (TFMRP): A further extension of the GTFMP, in which the possibility of modifying the flight's path is also considered.

Different versions of the GHP exist:

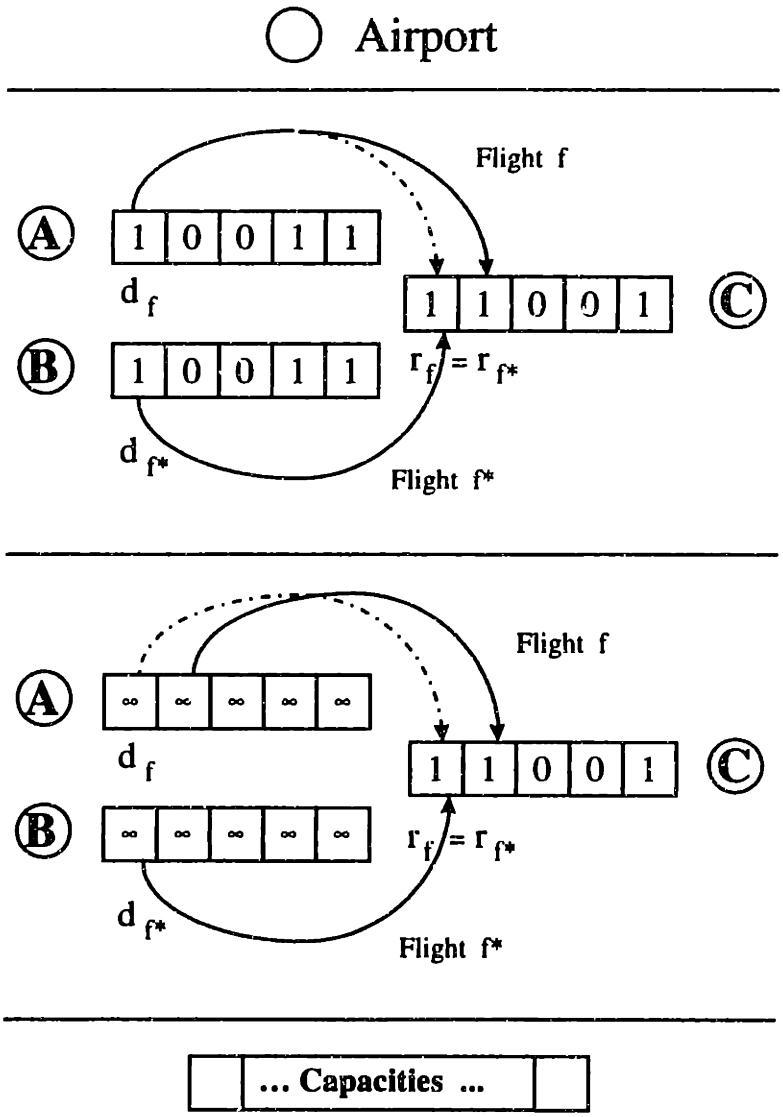
Static vs. Dynamic. Static versions allocate slots once before the schedule is actually flown, while the dynamic versions update the allocation at different stages, in accordance to updated information.

Deterministic vs. Probabilistic. Airports' capacities are constant in the deterministic versions and random variables in the probabilistic versions.

Note that the deterministic GHP, focusing on ground delays, implicitly assumes that there are no restrictions on sector capacities or on departure capacities. When these are considered, in fact, it is not necessarily true that an optimal solution consists of ground delays only.

Figure 1-1 gives an example in which only airport capacities are considered. Each square represents a (departure or arrival) time slot; the number inside the square is the airport's capacity for that time slot. In the example flight f is scheduled for departure from airport A at time d_f and arrival at airport C at time r_f, while flight f^* is scheduled for departure from airport B at time d_{f^*} and arrival at airport C at time r_{f^*} $(= r_f)$.

Figure 1-1: Optimality of ground delays under the assumption of unlimited departure capacities



Denote with c^a and c^g the cost, respectively, of an air and ground delay, and assume $1 < \frac{c^a}{c^g} < 4$ ($\frac{c^a}{c^g} \simeq 2$ is considered a reasonable estimate). The optimal solution to the limited departure capacities example (top part of the figure) assigns an air delay to one of the flights (f in the figure). If we were to consider only ground delays, we would assign 4 units of ground delay (even though there is a departure slot available at A at time $d_f + 3$, the arrival capacity for C at time $r_f + 3$ is equal to 0). On the other hand, in case of infinite departure capacities (bottom part of the figure), the optimal solution assigns 1 unit of ground delay. Since there is always capacity to accommodate a departure, we can always replace an airborne delay with a less expensive ground delay.

The assumption of limited capacities only on arrival is not unreasonable. Sectors are seldom a cause of delay (especially in the US traffic system). Moreover, safety concerns (the minimum separation between aircraft is larger for landings than for take-offs) lead to much higher values for departure capacities than for arrival capacities. In addition, air carriers have been pushing towards the concept of *free flight*. Under free flight, airlines are assigned an arrival time slot for each flight and, after that, they are free to select the time, route and speed for the flight as long as they arrive at the assigned time.

1.2 The Multi Airport Ground Holding Problem

This thesis discusses the static/deterministic version of the problem, under the assumption of unlimited departure and sector capacities. The Multi Airport Ground Holding Problem will be defined as follows:

> *Given a schedule for a set of flights F, and arrival capacities for a set of airports A, assign to each flight f an amount of delay δ_f so as to minimize the total cost of the delays in the network.*

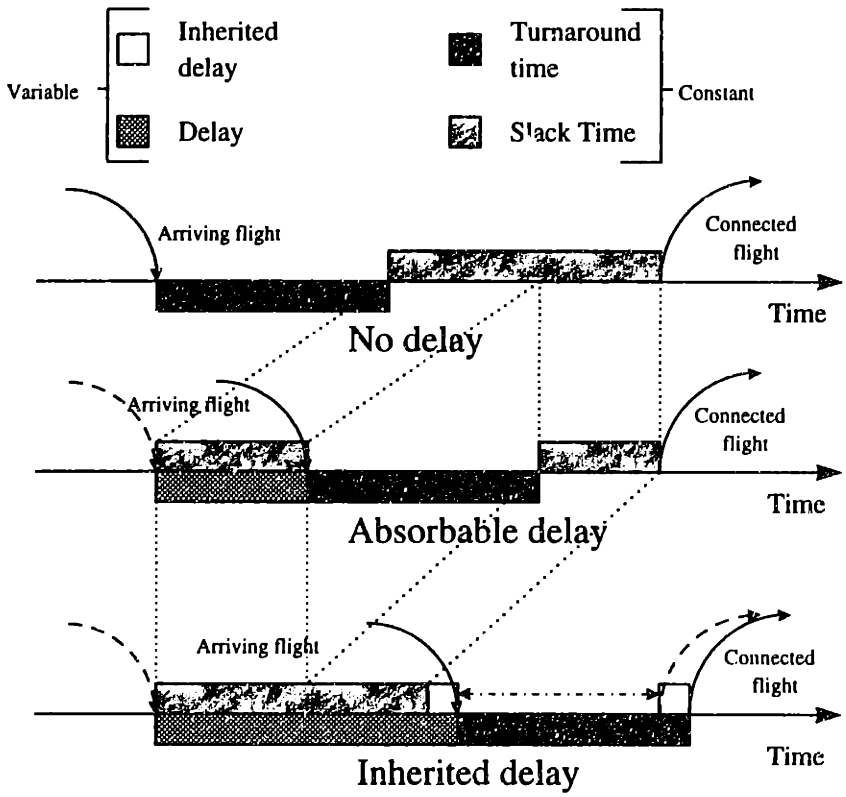
Table 1.1 presents the notation that will be used. It should be stressed that, since capacity is expressed per time unit, time is discretized, i.e., time t refers to the t^{th} *time interval* (similarly a delay δ means a delay of δ *units of time*). We explain the variables and parameters in more detail below.

Table 1.1: Notation

F	:	Set of flights;
f	:	Generic flight ($f \in F$);
F_c	:	Set of couples of connected flights ($(f, f') \in F \times F$);
A	:	Set of airports;
a	:	Generic airport ($a \in A$);
T	:	Set of times;
t	:	Generic time ($t \in T$);
$K_{a,t}$	:	Arrival capacity for airport a at time t;
δ^*	:	Maximum delay allowed;
Δ	:	Set of feasible delays $\{0, 1, \ldots, \delta^*\}$;
δ	:	Generic delay ($\delta \in \Delta$);
δ_f	:	Delay of flight f;
r_f	:	Scheduled arrival period for flight f;
a_f	:	Arrival airport of flight f;
T_f	:	Set of feasible times for flight f ($T_f = \{r_f, \ldots, r_f + \delta^*\}$);
$s_{ff'}$	:	Slack time between flights f and f' ($(f, f') \in F_c$);
η_f	:	Inherited delay of flight f;
c_f	:	Cost of delaying flight f for a time period;
x_{ft}	:	Binary decision variables in the VBO and BS models $((f,t) : f \in F, \quad t \in T_f)$;
$x_{f\delta}$	:	Binary decision variables in the ABGE model $((f,\delta) : f \in F, \quad \delta \in \Delta)$;

The feature that distinguishes the MAGHP from the SAGHP is the presence of "inherited" delays (Figure 1-2). For every flight we have a *turnaround time*, i.e., the minimum amount of time that the aircraft must spend on the ground before performing another flight. Turnaround time is needed for loading/unloading passengers, cleaning the aircraft, refueling, etc.. With respect to the time between the scheduled arrival and departure of the aircraft, the complement of the turnaround time is the *slack time*. If the delay for the arriving aircraft is less than or equal to the slack time there is no consequence for the departing flight. When the delay δ_f is greater than the slack time $s_{ff'}$, there is not enough time to complete the turnaround operations before the scheduled departure time: the departing flight "inherits" from the arriving one a delay $\eta_{f'}$ equal to the difference $\delta_f - s_{ff'}$.

Figure 1-2: Definition of turnaround time, slack time and inherited delay



The key elements of a mathematical programming model for the MAGHP are:

- **Objective function:** We want to minimize the sum of the costs of delays;
- **Capacity constraints:** A limited number of planes may land at a given airport in any period of time;
- **Assignment constraints:** Every flight must land at the destination airport no earlier than the desired arrival time r_f and no later than $r_f + \delta^*$. Note that a limit is set on the amount of delay that can be assigned to a flight. This may be motivated, for example, by equity reasons.
- **Coupling constraints:** Arrivals of two connected flights f and f' must be separated by at least the sum of the turnaround time for flight f plus the traveling time and delay of flight f'; this is equivalent to the condition $\delta_{f'} \geq \eta_{f'}$, i.e., the delay of flight f' must be greater than or equal to its inherited delay.
- **Integrality constraints:** Every flight must be assigned to land at the destination airport in one and only one period of time.

We conclude by noting that if coupling constraints were dropped, the problem could be decomposed into $|A|$ SAGHP's. These could be cast as transportation problems; hence the constraint matrix would be totally unimodular and the integrality constraints would no longer be needed.

1.3 The test cases

All the algorithms that are presented in this thesis were tested on two sets of instances already used in the existing literature ([2], [3], [8]). We will refer to cases V1, ..., V7 and BGN1, ..., BGN32, as V and BGN test cases, respectively. The first set was created *ad hoc* by P. D. Vranas for his Ph.D. thesis. The second set was created with POAGG (Pseudo OAG Generator). This code, developed at Charles Stark Draper Laboratories, generates schedules with characteristics similar to actual OAG (Official Airline Guide) schedules.

Table 1.2: Test cases description

Case	$\|A\|$	$\|F\|$	$\|F_c\|$	$\frac{\|F_c\|}{\|F\|}$	Number of itineraries with					
					1	2	3	4	5	≥ 6
					legs					
V1	2	1 000	200	20.00%	600	200	0	0	0	0
V2	2	1 000	400	40.00%	200	400	0	0	0	0
V3	2	1 000	600	60.00%	100	0	300	0	0	0
V4	2	1 000	800	80.00%	0	0	0	0	200	0
V5	4	2 000	400	20.00%	1 200	400	0	0	0	0
V6	4	2 000	800	40.00%	500	600	100	0	0	0
V7	4	2 000	1 200	60.00%	200	300	100	100	100	0
BGN1	2	1 004	496	49.40%	50	431	24	1	0	6
BGN2	3	1 172	767	65.44%	77	119	114	30	29	70
BGN3	3	1 181	773	65.45%	77	124	100	47	27	68
BGN4	2	1 342	935	69.67%	67	71	130	49	40	97
BGN5	9	1 403	729	51.96%	70	525	52	18	3	10
BGN6	9	1 419	733	51.66%	76	514	73	20	2	1
BGN7	3	1 593	1 043	65.47%	120	139	148	62	36	98
BGN8	5	1 760	1 370	77.84%	23	67	65	56	57	289
BGN9	4	1 854	1 339	72.22%	86	89	129	66	46	214
BGN10	2	1 909	956	50.08%	90	784	73	4	0	6
BGN11	4	1 940	1 229	63.35%	93	291	184	71	32	69
BGN12	3	1 945	1 134	58.30%	66	470	199	49	20	11
BGN13	3	1 989	1 316	66.16%	124	192	179	69	42	123
BGN14	4	2 366	1 507	63.69%	98	360	217	88	57	65
BGN15	9	2 396	1 313	54.80%	130	735	128	52	28	14
BGN16	3	2 526	1 272	50.36%	127	1 023	81	15	2	10
BGN17	3	2 527	1 273	50.38%	134	1 006	94	11	3	10
BGN18	3	2 530	1 276	50.43%	132	1 010	90	12	4	10
BGN19	3	2 532	1 278	50.47%	126	1 020	86	12	4	10
BGN20	3	2 534	1 280	50.51%	128	989	120	17	0	0
BGN21	6	2 546	1 539	60.45%	147	464	243	83	31	60
BGN22	4	2 672	1 418	53.07%	107	915	199	29	3	2
BGN23	4	2 806	1 689	60.19%	95	612	260	90	36	47
BGN24	5	2 882	2 196	76.20%	44	148	132	114	60	450
BGN25	6	3 034	2 277	75.05%	109	93	184	95	65	427
BGN26	10	3 142	1 722	54.81%	197	902	201	70	42	8
BGN27	5	3 192	2 443	76.54%	63	99	183	122	75	484
BGN28	5	3 805	2 319	60.95%	185	700	366	134	55	81
BGN29	5	3 823	2 794	73.08%	149	144	276	148	118	406
BGN30	6	4 523	2 953	65.29%	227	549	393	173	118	187
BGN31	5	4 773	2 888	60.51%	240	921	433	153	79	90
BGN32	10	5 005	2 935	58.64%	11	1 481	366	142	56	16

Table 1.2 gives a summary of the different instances. In the first 5 columns the following data are shown:

- the name given to the instance;
- $|A|$, the number of airports;
- $|F|$, the number of flights;
- $|F_c|$, the number of couples of connected flights;
- $\frac{|F_c|}{|F|}\%$, the percentage of connected flights.

The last 6 columns give a description of the different "itineraries" composing the instances, e.g., an aircraft scheduled to perform a 2–leg itinerary will perform a first flight, make a connection in an airport and perform a second flight. Itineraries with more than 5 legs are grouped together (the maximum number of legs is 11).

The value of $s_{ff'}$ is equal to 1 unit of time for all couple of connected flights $(f, f') \in F_c$ in the V test cases, while in the BGN test cases the value of $s_{ff'}$ depends on the time between the scheduled arrival and departure of connected flights (in these cases it is the turnaround time that is equal to 1 unit of time for all $(f, f') \in F_c$).

Some characteristics are common to all instances:

- $c_f = c = \$50$ for all flights;
- The number of time slots considered for each airport is 96, corresponding to the number of 15 minute intervals in 24 hours;
- The maximum admissible delay δ^* is equal to 4 units of time, i.e., one hour;
- Capacity is constant for each airport throughout the day, i.e., for time slots 1, ..., 96. However, in order to cover the case of delayed flights originally scheduled at the end of the day, some slots with "infinite" capacity are added after the 96^{th} slot.

Chapter 2

Integer programming models

In this chapter we present three different integer programming formulations for the solution of the MAGHP as defined in this thesis.

2.1 The Vranas, Bertsimas and Odoni model

The Vranas, Bertsimas and Odoni (VBO) model was the first one proposed in the literature ([9]) for solving the MAGHP. The decision variable x_{ft} is set equal to 1 if flight f is scheduled to arrive ***at*** time t. Figure 2-1 illustrates the different definition of the decision variables in the VBO, BS and ABGE models. For each flight f there are $|\Delta|$ decision variables, one for each possible delay $(0, 1, \ldots, \delta^*)$.

$$\min \quad \sum_{f \in F} c_f \left[\sum_{t \in T_f} t\, x_{ft} - r_f \right] \tag{2.1}$$

s.t.

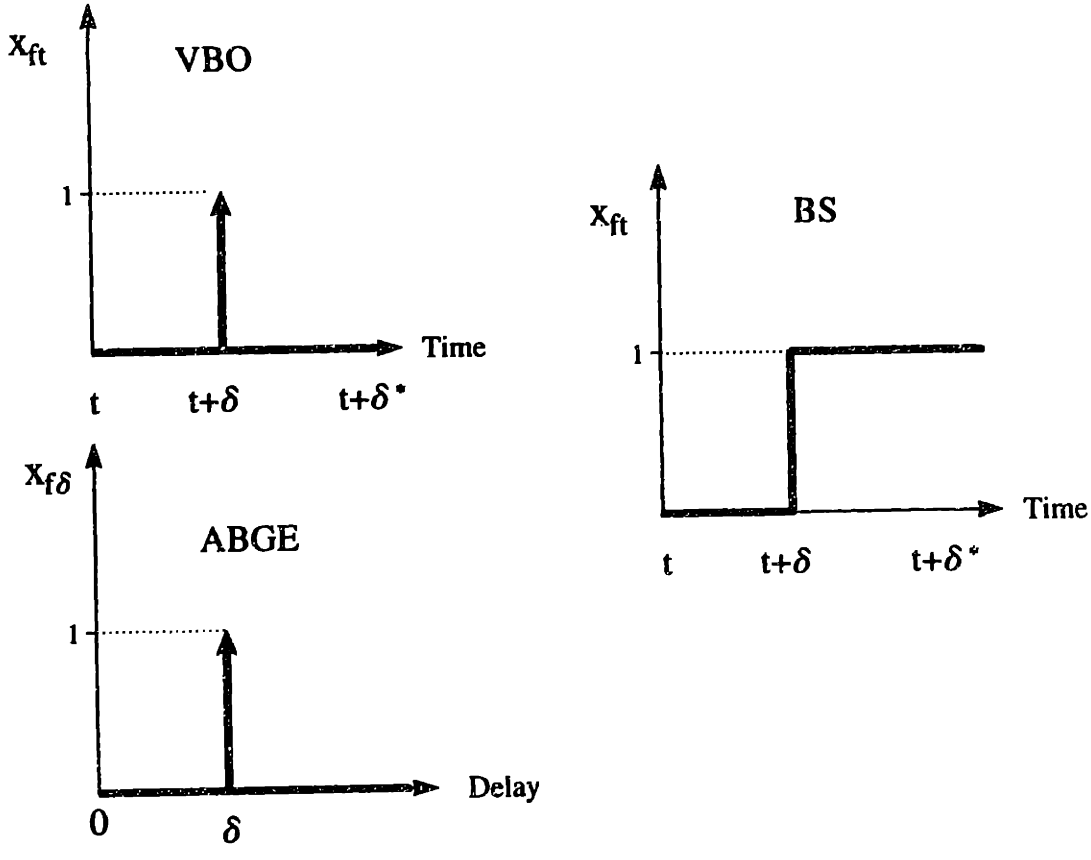
$$\sum_{\substack{(f,t): \\ (a_f = a, t \in T_f)}} x_{ft} \leq K_{a,t} \qquad \forall a \in A \quad \forall t \in T \tag{2.2}$$

$$\sum_{t \in T_f} x_{ft} = 1 \qquad \forall f \in F \qquad (2.3)$$

$$\sum_{t \in T_f} t\, x_{ft} - \sum_{t \in T_{f'}} t\, x_{f't} \leq r_f - r_{f'} + s_{ff'} \qquad \forall (f, f') \in F_c \qquad (2.4)$$

$$x_{ft} \in \{0, 1\} \qquad \forall f \in F \quad \forall t \in T_f \qquad (2.5)$$

Figure 2-1: Different definition of the decision variables



The term $\left[\sum_{t\in T_f} t\,x_{ft} - r_f\right]$ is equal to the amount of delay assigned to flight f (i.e., δ_f); hence the objective function (2.1) minimizes the total cost of delay in the network. Capacity constraints (2.2) ensure that the number of flights landing in airport a at time t does not exceed the capacity. Each flight must land during its feasible time window T_f; this is achieved through the assignment constraints (2.3). The network effects are taken into consideration by the coupling constraints (2.4); the constraint may be rewritten as $\sum_{t\in T_{f'}} t\,x_{f't} - r_{f'} \geqslant \sum_{t\in T_f} t\,x_{ft} - r_f - s_{ff'}$ or $\delta_{f'} \geqslant \delta_f - s_{ff'}$ or $\delta_{f'} \geqslant \eta_{f'}$, i.e., the delay of flight f' must be greater than or equal to its inherited delay. Constraints (2.5) are simply the integrality constraints.

2.2 The Bertsimas and Stock model

The Bertsimas and Stock (BS) has been the most efficient MAGHP model to date ([1]). It makes use of a different definition of the value of the decision variables: x_{ft} is equal to 1 if and only if flight f arrives ***by*** time t (Figure 2-1).

The new definition leads to the following formulation.

$$\min \quad \sum_{f\in F} c_f \left[\sum_{t\in T_f} t\left(x_{ft} - x_{f(t-1)}\right) - r_f\right] \tag{2.6}$$

s.t.

$$\sum_{\substack{(f,t):\\ (a_f=a,t\in T_f)}} \left[x_{ft} - x_{f(t-1)}\right] \leq K_{a,t} \qquad \forall a \in A \quad \forall t \in T \tag{2.7}$$

$$x_{ft} - x_{f(t-1)} \geq 0 \qquad \forall f \in F \quad \forall t \in T_f \tag{2.8}$$

$$x_{f(r_f+\delta^*)} = 1 \qquad \forall f \in F \tag{2.9}$$

$$x_{ft} - x_{f't'} \geq 0 \quad \forall (f,f') \in F_c \quad \forall (t,t') \in T_f \times T_{f'},$$

$$t' = t - r_f + r_{f'} - s_{ff'} \tag{2.10}$$

$$x_{ft} \in \{0,1\} \qquad \forall f \in F \quad \forall t \in T_f \tag{2.11}$$

In order to pinpoint the moment of arrival of flight f, given the new decision variables x_{ft}, we evaluate the quantity $[x_{ft} - x_{f(t-1)}]$. This quantity is equal to one only during the time interval t when flight f arrives, and is used in the objective function (2.6) and in the capacity constraints (2.7). The assignment constraints (2.8) and (2.9) take the place of constraints (2.3) in model VBO. Constraints (2.8) guarantee that variables corresponding to time periods after the arrival time will all be set to 1. In (2.9) the value of the variable corresponding to the last admissible delay is set to 1, because in every feasible solution flight f must have landed by this time. We can substitute this value in all other constraints to reduce the total number of variables by $|F|$ (one variable less for each flight). Constraints (2.10) make use again of the new definition of the decision variables to prohibit unacceptable combinations of delays for connected flights. The values of t and t' are such that $\eta_{f'} = \delta_{f'}$; if $x_{ft} = 0$, then it follows $x_{ft'} = 0$, i.e., if flight f has not landed by time t, then flight f' cannot have landed by time t' because the delay of flight f' would be less than its inherited delay. Constraints (2.11) are the integrality constraints.

2.3 The Andreatta, Brunetta and Guastalla Exact model

The Andreatta, Brunetta and Guastalla Exact model (ABGE) is nearly identical to the VBO model.

$$\min \quad \sum_{f \in F} c_f \sum_{\delta \in \Delta} \delta \, x_{f\delta} \tag{2.12}$$

s.t.

$$\sum_{\substack{(f,\delta): \\ (f \in F, \delta \in \Delta, \delta = t - r_f)}} x_{f\delta} \leq K_{a,t} \qquad \forall a \in A \quad \forall t \in T \tag{2.13}$$

$$\sum_{\delta \in \Delta} x_{f\delta} = 1 \qquad \forall f \in F \tag{2.14}$$

$$\sum_{d=0}^{\delta} x_{fd} - \sum_{d=0}^{\delta - s_{ff'}} x_{f'd} \geq 0 \qquad \forall (f, f') \in F_c, \; \forall \delta : (\delta, \delta - s_{ff'}) \in \Delta^2 \tag{2.15}$$

$$x_{f\delta} \in \{0, 1\} \qquad \forall f \in F, \quad \forall \delta \in \Delta \tag{2.16}$$

Constraints (2.13), (2.14), and (2.16) are identical to constraints (2.2), (2.3), and (2.5), except for the use of delays instead of times in the variable definitions (Figure 2-1). Each individual constraint (2.4) is replaced by a *set* of constraints (2.15), one for each value of δ between $s_{ff'}$ and δ^* (for $\delta < s_{ff'}$ we have $\delta - s_{ff'} \notin \Delta$). In the next section, it will be shown that this choice leads to a stronger formulation.

2.4 An example with two flights

Table 2.1 presents the VBO and ABGE formulations for a simple case with two flights: f_1 is scheduled to arrive at airport a_1 at time 12, and f_2 is scheduled to arrive at airport a_2 at time 20. The flights are connected, with slack time $s_{f_1 f_2}$ equal to one. The capacity is such that flight f_1 can arrive with 3 or 4 periods of delay, while flight f_2 can land with 1, 3 or 4 periods of delay. The cost for a period of delay is \$50 and the maximum admissible delay, δ^*, is equal to

Table 2.1: Example of formulation

Model	ABGE	VBO
Minimize	$50x_{11} + 100x_{12} + 150x_{13} +$ $+200x_{14} + 50x_{21} + 100x_{22} +$ $+150x_{23} + 200x_{24}$	$600x_{1,12} + 650x_{1,13} + 700x_{1,14} +$ $+750x_{1,15} + 800x_{1,16} + 1000x_{2,20} +$ $+1050x_{2,21} + 1100x_{2,22} + 1150x_{2,23} +$ $+1200x_{2,24}$
Subject To		
Assign(1)	$x_{10} + x_{11} + x_{12} + x_{13} + x_{14} = 1$	$x_{1,12} + x_{1,13} + x_{1,14} + x_{1,15} + x_{1,16} = 1$
Assign(2)	$x_{20} + x_{21} + x_{22} + x_{23} + x_{24} = 1$	$x_{2,20} + x_{2,21} + x_{2,22} + x_{2,23} + x_{2,24} = 1$
Cap(1,12)	$x_{10} \leq 0$	$x_{1,12} \leq 0$
Cap(1,13)	$x_{11} \leq 0$	$x_{1,13} \leq 0$
Cap(1,14)	$x_{12} \leq 0$	$x_{1,14} \leq 0$
Cap(1,15)	$x_{13} \leq 1$	$x_{1,15} \leq 1$
Cap(1,16)	$x_{14} \leq 1$	$x_{1,16} \leq 1$
Cap(2,20)	$x_{20} \leq 0$	$x_{2,20} \leq 0$
Cap(2,21)	$x_{21} \leq 1$	$x_{2,21} \leq 1$
Cap(2,22)	$x_{22} \leq 0$	$x_{2,22} \leq 0$
Cap(2,23)	$x_{23} \leq 1$	$x_{2,23} \leq 1$
Cap(2,24)	$x_{24} \leq 1$	$x_{2,24} \leq 1$
Coup(1,2)	$\begin{cases} x_{10} + x_{11} - x_{20} \geq 0 \\ x_{10} + x_{11} + x_{12} + \\ -x_{20} - x_{21} \geq 0 \\ x_{10} + x_{11} + x_{12} + x_{13} + \\ -x_{20} - x_{21} - x_{22} \geq 0 \end{cases}$	$12x_{1,12} + 13x_{1,13} + 14x_{1,14} + 15x_{1,15} +$ $+16x_{1,16} - 20x_{2,20} - 21x_{2,21} - 22x_{2,22} +$ $-23x_{2,23} - 24x_{2,24} \leq -7$
Binaries	$x_{10}, x_{11}, x_{12}, x_{13}, x_{14}$ $x_{20}, x_{21}, x_{22}, x_{23}, x_{24}$	$x_{1,12}, x_{1,13}, x_{1,14}, x_{1,15}, x_{1,16}$ $x_{2,20}, x_{2,21}, x_{2,22}, x_{2,23}, x_{2,24}$

4 (as in the V and BGN test cases). It can be seen that the two problems are identical, except for the number of coupling constraints (1 for the VBO model, 3 for the ABGE model) and the coefficients in the objective function (a constant value of $\sum_f c_f r_f = \$50(12+20) = \$1600$ must be subtracted from the objective function value of VBO). The optimal solution of the linear relaxation of VBO is $x_{1,15} = 1$, $x_{2,21} = \frac{2}{3}$, and $x_{2,24} = \frac{1}{3}$ (i.e., $\delta_{f_1} = 3, \delta_{f_2} = \frac{2}{3} \cdot 1 + \frac{1}{3} \cdot 4 = 2$), with a total cost of \$250. The optimal integer solution is $x_{1,15} = x_{2,23} = 1$ (i.e., $\delta_{f_1} = \delta_{f_2} = 3$), with a total cost of \$300. The optimal solution to ABGE is the same, i.e., $x_{13} = x_{23} = 1$, and is obtained directly from the model relaxation. The solution $x_{13} = 1$, $x_{21} = \frac{2}{3}$, $x_{24} = \frac{1}{3}$ does not, in fact, satisfy the second coupling constraint: $-\frac{2}{3} \not\geq 0$.

The unique coupling constraint in the VBO model limits the value of the weighted average of the flights' decision variables, with weights equal to the corresponding times. The set of constraints in ABGE deals with several combinations of delay values. Pursuing the analogy with random variables, we say that VBO places a constraint on the mean, while ABGE places constraints on several ($\delta^* - s_{ff'}$) percentiles. Even though the two formulations are equivalent in terms of integer feasible solutions, they are not so if fractional values are allowed: the ABGE formulation is stronger.

As the example shows, it is possible for the VBO model to have a solution to the relaxed problem with a lower value than that of the integer optimal solution. It will be seen that the ABGE and BS models have always given the same value for the relaxed and integer problems for the tested instances.

2.5 Number of variables and constraints

Since the size of the instances exceeded the capabilities of our model generating software, we implemented a special purpose C code to generate the 3 different formulations and eliminate *redundant* constraints, according to the following rules:

Variables. There are exactly $|\Delta|$ variables for each flight in the VBO and ABGE models. As pointed out in Section 2.2, we can use constraints (2.9) to eliminate one variable for each flight, leaving us with δ^* variables per flight.

Capacity constraints. A capacity constraint was considered redundant whenever the number of variables with positive coefficients was less than or equal to the capacity of the airport. *The number of non redundant capacity constraints is the same for all three models, for* the number of positive coefficient variables is the same in (2.2), (2.7), and (2.13) for every (a,t). The maximum number of constraints is $|A||T|$.

Assignment constraints. There are exactly $|F|$ assignment constraints in the VBO and ABGE models. The number of constraints for model BS is, according to the formulation given, $|F||\Delta|$. *For each flight, however, two constraints are redundant: when* $t = r_f$ we have $x_{f(r_f-1)}$ equal to zero (flight cannot be assigned to a slot before the scheduled arrival time) and $x_{fr_f} \geq 0$ is redundant; when $t = r_f + \delta^\star$ we have $x_{f(r_f+\delta^\star)} = 1$ (substitution of constraints (2.9)) and $x_{f(r_f+\delta^\star-1)} \leq 1$ is redundant. Therefore the number of non redundant constraints for model BS is exactly $|F|(|\Delta| - 2)$.

Coupling constraints. The number of non redundant coupling constraints is equal to $\delta^\star|F_c| - \sum_{(f,f')\in F_c} s_{ff'}$ for the ABGE and BS models, as we will now prove. First, we will show that the number of constraints is the same. Constraints (2.10) are defined over the set $\{(t,t') \in T_f \times T_{f'} : t' = t - r_f + r_{f'} - s_{ff'}\}$; since $\delta = t - r_f$,this is equivalent to $\{(\delta,\delta') \in \Delta^2 : \delta' = \delta - s_{ff'}\} \equiv \{\delta : (\delta, \delta - s_{ff'}) \in \Delta^2\}$, i.e., the set over which constraints (2.15) are defined. The constraint corresponding to $\delta = \delta^\star$ is redundant in both models: in the ABGE model because the first sum in the constraint is over all values of δ and therefore equal to one (constraint (2.14)), while in model BS because $x_{f(r_f+\delta^\star)} = 1$ (substitution of constraints (2.9)). Therefore for each couple of connected flights we have $\delta^\star - s_{ff'}$ constraints, for a total of $\sum_{(f,f')\in F_c}(\delta^\star - s_{ff'}) = \delta^\star|F_c| - \sum_{(f,f')\in F_c} s_{ff'}$ (this is in $O(\delta^\star F_c)$). This analysis shows also that couple of flights for which $s_{ff'} \geq \delta^\star$ should not be included in the $|F_c|$ set even if performed by the same aircraft, because they don't imply any constraint. The number of coupling constraints for model VBO is exactly equal to $|F_c|$.

Table 2.2 summarizes this information. Since the number of capacity constraints is the same for the three models and the number of coupling constraints in ABGE and BS is the same (this number being greater than or equal to $|F_c|$), the models may be ranked in increasing order

Table 2.2: Number of variables and constraints

Model	Variables	Constraints		
		Capacity	Assignment	Coupling
ABGE	$\|F\|\|\Delta\|$	$O(\|A\|\|T\|)$	$\|F\|$	$O(\delta^*\|F_c\|)$
BS	$\delta^*\|F\|$	$O(\|A\|\|T\|)$	$\|F\|(\delta^* - 1)$	$O(\delta^*\|F_c\|)$
VBO	$\|F\|\|\Delta\|$	$O(\|A\|\|T\|)$	$\|F\|$	$\|F_c\|$

Note: $\delta^* = |\Delta| - 1$;

of the total number of constraints as follows: VBO, ABGE, BS.This ranking is confirmed in table 2.3, which gives information about the constraint matrix given as input to Cplex 3.0©, including the number of constraints (rows), variables (columns) and nonzero elements for each formulation.

2.6 Results

All experiments were performed on a SUN Sparc 20 with 120 Mb of RAM. The integer programming solver was Cplex 3.0©. We set time limits of one hour for the solution of the relaxed formulation and one hour for the branch & bound algorithm, together with a limit of 5000 nodes in the tree for the latter.

Tables 2.4, 2.5 and 2.6 report the results obtained. With the exception of cases V1 and V2, the VBO model was not able to reach an optimal solution. In such cases the last column of Table 2.4 reports the best, if any, integer solution found. For the same reason, in Table 2.7, which compares the running times of the different algorithms, VBO times are reported only for the solution of the relaxed problem. In the same table, the total solution time includes the time to generate the problem (input of data and elimination of redundant constraints) using the C code.

Examining the four tables, we can draw the following conclusions:

- VBO was the fastest model to solve the relaxed problem (with the exception of case V3), while BS was the slowest. This was to be expected since the complexity of the simplex algorithm depends on the number of constraints, of which VBO has the fewest while BS has the most.

Table 2.3: Cplex problem matrices: number of rows, columns and nonzeroes

Case	Constraints			Variables			Nonzeroes		
	ABGE	BS	VBO	ABGE	BS	VBO	ABGE	BS	VBO
V1	1 697	3 697	1 297	5 000	4 000	5 000	12 760	14 810	11 760
V2	2 317	4 317	1 517	5 000	4 000	5 000	16 000	16 400	14 000
V3	2 907	4 907	1 707	5 000	4 000	5 000	18 970	17 550	15 970
V4	3 514	5 514	1 914	5 000	4 000	5 000	22 000	18 800	18 000
V5	3 421	7 421	2 621	10 000	8 000	10 000	26 000	30 400	24 000
V6	4 629	8 629	3 029	10 000	8 000	10 000	32 000	32 800	28 000
V7	5 831	9 831	3 431	10 000	8 000	10 000	38 000	35 200	32 000
BGN1	2 141	4 149	1 626	5 020	4 016	5 020	12 789	12 395	12 694
BGN2	2 696	5 040	2 140	5 860	4 688	5 860	16 826	16 652	17 881
BGN3	2 691	5 053	2 150	5 905	4 724	5 905	16 801	16 641	17 961
BGN4	3 233	5 917	2 420	6 710	5 368	6 710	20 494	19 620	21 104
BGN5	3 283	6 089	2 623	7 015	5 612	7 015	17 837	17 402	18 182
BGN6	3 395	6 233	2 659	7 095	5 676	7 095	18 352	17 718	18 337
BGN7	3 760	6 946	2 839	7 965	6 372	7 965	23 656	22 888	24 266
BGN8	4 189	7 709	3 467	8 800	7 040	8 800	26 998	27 136	30 238
BGN9	4 448	8 156	3 459	9 270	7 416	9 270	28 481	27 904	30 231
BGN10	3 989	7 807	2 992	9 545	7 636	9 545	24 550	23 758	24 345
BGN11	4 578	8 458	3 430	9 700	7 760	9 700	28 583	27 606	28 988
BGN12	4 371	8 261	3 273	9 725	7 780	9 725	27 093	26 078	27 273
BGN13	4 704	8 682	3 509	9 945	7 956	9 945	29 922	28 847	30 527
BGN14	5 605	10 337	4 132	11 830	9 464	11 830	35 193	33 709	35 363
BGN15	5 554	10 346	4 253	11 980	9 584	11 980	32 205	31 067	32 265
BGN16	5 237	10 289	3 985	12 630	10 104	12 630	32 272	31 456	32 372
BGN17	5 178	10 232	3 997	12 635	10 108	12 635	31 882	31 239	32 342
BGN18	5 179	10 239	4 003	12 650	10 120	12 650	31 886	31 251	32 386
BGN19	5 137	10 201	4 000	12 660	10 128	12 660	31 746	31 256	32 451
BGN20	5 345	10 413	4 001	12 670	10 136	12 670	32 810	31 694	32 490
BGN21	6 105	11 197	4 463	12 730	10 184	12 730	37 314	35 543	36 799
BGN22	5 690	11 034	4 316	13 360	10 688	13 360	34 873	33 719	35 093
BGN23	6 461	12 073	4 754	14 030	11 224	14 030	40 471	38 783	40 381
BGN24	7 230	12 994	5 416	14 410	11 528	14 410	46 542	44 662	48 452
BGN25	7 336	13 404	5 699	15 170	12 136	15 170	47 427	46 357	50 627
BGN26	7 246	13 530	5 469	15 710	12 568	15 710	42 666	41 010	42 391
BGN27	7 570	13 954	5 971	15 960	12 768	15 960	49 713	48 938	53 933
BGN28	8 775	16 385	6 451	19 025	15 220	19 025	55 484	53 326	55 459
BGN29	8 849	16 495	6 945	19 115	15 292	19 115	58 112	57 178	62 562
BGN30	10 747	19 793	7 870	22 615	18 092	22 615	68 524	65 634	68 904
BGN31	10 965	20 511	7 995	23 865	19 092	23 865	69 745	66 921	69 335
BGN32	11 463	21 473	8 419	25 025	20 020	25 025	69 583	65 473	69 038

- The ABGE model reached an optimal integer solution for the relaxed problem in 24 out of 39 cases; model BS did so in 23 out of 39 cases. This never happened with VBO.

- In only 5 cases (V1, V2, V4, V5, BGN12) the optimal value of the relaxed problem of model VBO coincide with the optimal integer value. On the contrary, this was always the case for models BS and ABGE.

- Model VBO found an integer solution in only 13 cases and reached optimality in only 2.

- The mean number of non integers in the solution and nodes in the branch & bound search was lower for ABGE ($\frac{1109}{39} \simeq 28.43$ and $\frac{4214}{15} \simeq 280.93$ respectively) than for BS ($\frac{1529}{39} \simeq 39.21$ and $\frac{5169}{16} \simeq 323.06$ respectively).

- With the exception of case V4, ABGE was faster then BS in terms of total times, even when ABGE performed a branch & bound search and the BS model did not.

- The "most difficult" case (BGN31) was solved in about 35 minutes with ABGE and in about 65 minutes with BS. This was the only case that took more than 30 minutes to solve with ABGE, while this threshold was exceeded for BS in cases BGN29 (about 36 minutes) and BGN30 (about 48 minutes) as well.

- On average, ABGE is about 4 times faster than BS (average of $\frac{\text{ABG}}{\text{BS}} \simeq 26.44\%$). In 21 out of 39 cases it is more than 4 times faster, and in one case (BGN30) is more than 10 times faster.

Given these results, ABGE was chosen as the model for use in the integrated algorithm that will be presented in Chapter 4.

Table 2.4: Solving VBO: Detailed results

Case	Relaxed Problem				Branch & Bound			
	Time	Iter.	Non Int.	Optimal Value	Time	Iter.	Nodes	Optimal Value
V1	3.50	1 678	108	71 000.00	21.36	1 828	374	71 000
V2	7.40	2 634	261	56 000.00	92.62	7 758	1 243	56 000
V3	18.27	3 260	523	84 300.00	1 038.35	97 129	5 000	—
V4	23.99	3 499	588	65 000.00	2 695.80	226 490	5 000	—
V5	13.40	2 992	448	96 300.00	674.62	28 320	5 000	—
V6	21.62	3 644	543	89 933.33	1 132.79	61 234	5 000	—
V7	29.55	4 987	627	71 600.00	952.00	47 712	5 000	—
BGN1	2.71	418	168	12 900.00	390.20	36 043	5 000	15 100*
BGN2	9.66	1 799	474	50 512.50	562.01	42 985	5 000	—
BGN3	4.06	642	213	25 400.00	482.04	34 464	5 000	27 850*
BGN4	12.46	2 122	535	50 575.00	973.98	68 407	5 000	—
BGN5	4.00	401	153	10 725.00	468.70	22 782	5 000	13 800*
BGN6	3.41	420	161	10 450.00	526.65	29 225	5 000	12 650*
BGN7	11.60	1 853	664	58 650.00	1 008.06	59 803	5 000	—
BGN8	14.03	1 978	496	62 200.00	917.77	46 092	5 000	—
BGN9	17.11	2 387	693	78 950.00	1 174.42	57 987	5 000	—
BGN10	11.82	1 215	589	41 725.00	792.88	39 731	5 000	—
BGN11	9.83	1 329	653	54 750.00	1 036.36	53 837	5 000	—
BGN12	4.78	722	333	22 400.00	808.88	37 992	5 000	28 900*
BGN13	12.01	1 341	388	44 533.33	901.61	42 878	5 000	52 050*
BGN14	18.72	1 890	647	63 100.00	1 145.20	47 782	5 000	74 150*
BGN15	18.87	2 174	994	68 178.12	924.82	32 785	5 000	—
BGN16	17.21	1 397	608	50 881.25	882.86	30 871	5 000	—
BGN17	10.57	868	391	24 087.50	1 034.12	46 265	5 000	33 900*
BGN18	12.16	907	420	25 937.50	953.32	37 319	5 000	34 750*
BGN19	14.85	1 186	477	31 512.50	1 034.16	42 075	5 000	40 400*
BGN20	15.69	1 237	548	38 712.50	1 150.46	52 632	5 000	50 350*
BGN21	30.12	2 816	863	73 200.00	1 248.44	51 922	5 000	—
BGN22	16.79	1 302	622	42 775.00	1 037.54	36 350	5 000	—
BGN23	28.87	2 415	861	74 650.00	1 354.42	45 617	5 000	—
BGN24	71.14	6 019	1 019	137 654.17	2 666.78	86 684	5 000	—
BGN25	64.41	5 418	997	145 762.50	2 547.28	84 124	5 000	—
BGN26	27.80	2 169	915	78 579.17	1 223.56	28 948	5 000	—
BGN27	114.57	6 980	1 107	188 487.50	3 600.54	76 687	4 762	—
BGN28	57.66	3 569	1 123	112 645.83	2 173.06	62 773	5 000	—
BGN29	54.02	3 502	1 067	123 800.00	3 043.86	76 841	5 000	—
BGN30	70.22	4 283	1 333	136 950.00	3 040.03	71 086	5 000	—
BGN31	92.91	4 792	1 643	171 775.00	2 729.61	57 905	5 000	—
BGN32	90.70	4 628	1 752	129 775.00	2 169.23	40 061	5 000	—
		Total: 26 005				Total: 186 379		

* Best integer solution

Table 2.5: Solving BS: Detailed results

Case	Relaxed Problem				Branch & Bound			
	Time	Iter.	Non Int.	Optimal Value	Time	Iter.	Nodes	Optimal Value
V1	14.00	3 403	0	71 000.00	—	—	—	71 000
V2	28.36	4 741	0	56 000.00	—	—	—	56 000
V3	37.73	5 061	73	84 700.00	112.85	4 468	410	84 700
V4	62.93	6 679	43	65 000.00	16.56	639	43	65 000
V5	101.71	8 687	0	96 300.00	—	—	—	96 300
V6	144.28	10 499	0	92 200.00	—	—	—	92 200
V7	221.72	12 969	0	72 200.00	—	—	—	72 200
BGN1	15.93	3 243	0	13 200.00	—	—	—	13 200
BGN2	37.82	5 302	8	51 000.00	0.44	6	1	51 000
BGN3	34.68	4 925	0	26 150.00	—	—	—	26 150
BGN4	70.83	7 566	18	51 500.00	30.53	1 126	99	51 500
BGN5	30.82	4 771	0	11 250.00	—	—	—	11 250
BGN6	32.60	4 770	0	10 750.00	—	—	—	10 750
BGN7	97.06	8 502	92	58 900.00	36.32	1 091	96	58 900
BGN8	140.49	10 286	12	62 400.00	0.64	2	1	62 400
BGN9	173.29	11 955	67	79 350.00	211.24	5 687	242	79 350
BGN10	72.90	6 927	0	42 700.00	—	—	—	42 700
BGN11	144.29	10 626	71	54 750.00	191.01	5 022	298	54 750
BGN12	102.78	8 342	0	22 400.00	—	—	—	22 400
BGN13	139.81	9 930	0	45 150.00	—	—	—	45 150
BGN14	235.48	13 048	28	63 550.00	14.05	365	16	63 550
BGN15	121.13	9 507	0	70 500.00	—	—	—	70 500
BGN16	102.90	8 536	0	52 150.00	—	—	—	52 150
BGN17	97.55	8 496	0	25 250.00	—	—	—	25 250
BGN18	97.81	8 442	0	27 000.00	—	—	—	27 000
BGN19	106.23	8 435	0	32 600.00	—	—	—	32 600
BGN20	111.90	8 539	0	39 800.00	—	—	—	39 800
BGN21	309.03	15 816	58	74 250.00	103.87	2 467	96	74 250
BGN22	125.64	9 063	0	43 900.00	—	—	—	43 900
BGN23	287.86	14 713	0	75 300.00	—	—	—	75 300
BGN24	453.17	19 647	195	138 350.00	691.24	11 892	508	138 350
BGN25	486.32	20 731	219	146 400.00	858.30	14 039	485	146 400
BGN26	204.39	11 983	0	80 950.00	—	—	—	80 950
BGN27	563.64	19 012	207	189 600.00	1 023.81	11 271	570	189 600
BGN28	625.99	21 898	0	113 600.00	—	—	—	113 600
BGN29	771.47	23 542	234	124 000.00	1 419.96	18 156	764	124 000
BGN30	1 098.24	30 092	152	137 800.00	1 801.97	19 124	632	137 800
BGN31	1 164.47	31 227	52	172 450.00	2 663.61	25 786	908	172 450
BGN32	610.48	20 920	0	134 500.00	—	—	—	134 500
		Total: 1 529				Total: 5 169		

Table 2.6: Solving ABGE: detailed results

Case	Relaxed Problem				Branch & Bound			
	Time	Iter.	Not Int.	Optimal Value	Time	Iter.	Nodes	Optimal Value
V1	3.51	1 619	0	71 000.00	—	—	—	71 000
V2	8.51	2 358	0	56 000.00	—	—	—	56 000
V3	16.47	2 863	129	84 700.00	135.60	7 504	428	84 700
V4	26.46	3 782	74	65 000.00	157.38	7 980	392	65 000
V5	16.46	3 375	0	96 300.00	—	—	—	96 300
V6	36.88	4 342	0	92 200.00	—	—	—	92 200
V7	41.23	4 419	26	72 200.00	1.74	29	7	72 200
BGN1	3.20	614	0	13 200.00	—	—	—	13 200
BGN2	14.65	2 413	0	51 000.00	—	—	—	51 000
BGN3	6.11	942	0	26 150.00	—	—	—	26 150
BGN4	19.89	2 585	121	51 500.00	25.93	1 236	131	51 500
BGN5	4.52	534	0	11 250.00	—	—	—	11 250
BGN6	4.80	606	0	10 750.00	—	—	—	10 750
BGN7	22.59	2 756	0	58 900.00	—	—	—	58 900
BGN8	23.80	2 566	18	62 400.00	0.78	19	2	62 400
BGN9	38.31	3 664	52	79 350.00	90.90	3 438	186	79 350
BGN10	20.73	2 191	0	42 700.00	—	—	—	42 700
BGN11	24.25	2 426	78	54 750.00	145.46	5 425	288	54 750
BGN12	10.80	1 311	0	22 400.00	—	—	—	22 400
BGN13	18.12	1 831	22	45 150.00	6.39	255	24	45 150
BGN14	37.73	3 332	36	63 550.00	88.29	2 793	187	63 550
BGN15	34.06	3 460	0	70 500.00	—	—	—	70 500
BGN16	27.01	2 295	0	52 150.00	—	—	—	52 150
BGN17	17.15	1 451	0	25 250.00	—	—	—	25 250
BGN18	18.36	1 633	0	27 000.00	—	—	—	27 000
BGN19	18.34	1 572	0	32 600.00	—	—	—	32 600
BGN20	25.42	2 089	0	39 800.00	—	—	—	39 800
BGN21	57.87	4 239	0	74 250.00	—	—	—	74 250
BGN22	27.12	2 089	0	43 900.00	—	—	—	43 900
BGN23	57.27	3 675	16	75 300.00	2.55	58	7	75 300
BGN24	117.23	7 128	0	138 350.00	—	—	—	138 350
BGN25	117.07	6 790	130	146 400.00	602.78	15 052	428	146 400
BGN26	52.79	3 516	0	80 950.00	—	—	—	80 950
BGN27	155.23	7 573	82	189 600.00	937.84	17 085	590	189 600
BGN28	123.25	5 882	48	113 600.00	3.18	58	5	113 600
BGN29	127.28	5 902	134	124 000.00	1 378.76	25 568	741	124 000
BGN30	237.04	8 852	0	137 800.00	—	—	—	137 800
BGN31	256.94	9 114	143	172 450.00	1 852.06	26 243	798	172 450
BGN32	168.36	6 927	0	134 500.00	—	—	—	134 500
		Total: 1 109				Total: 4 214		

Table 2.7: Comparison of times

Case	Solving Relaxation			B & B		Total*		
	ABGE	BS	VBO	ABGE	BS	ABGE	BS	$\frac{ABGE}{BS}$
V1	3.51	14.00	3.50	—	—	3.56	14.05	25.34%
V2	8.51	28.36	7.40	—	—	8.58	28.42	30.19%
V3	16.47	37.73	18.27	135.60	112.85	152.14	150.65	100.99%
V4	26.46	62.93	23.99	157.38	16.56	183.94	79.57	231.17%
V5	16.46	101.71	13.40	—	—	16.58	101.85	16.28%
V6	36.88	144.28	21.62	—	—	37.04	144.40	25.65%
V7	41.23	221.72	29.55	1.74	—	43.13	221.86	19.44%
BGN1	3.20	15.93	2.71	—	—	3.24	15.98	20.28%
BGN2	14.65	37.82	9.66	—	0.44	14.73	38.34	38.42%
BGN3	6.11	34.68	4.06	—	—	6.20	34.74	17.85%
BGN4	19.89	70.83	12.46	25.93	30.53	45.90	101.46	45.24%
BGN5	4.52	30.82	4.00	—	—	4.60	30.90	14.89%
BGN6	4.80	32.60	3.41	—	—	4.89	32.66	14.97%
BGN7	22.59	97.06	11.60	—	36.32	22.68	133.48	16.99%
BGN8	23.80	140.49	14.03	0.78	0.64	24.69	141.24	17.48%
BGN9	38.31	173.29	17.11	90.90	211.24	129.35	384.66	33.63%
BGN10	20.73	72.90	11.82	—	—	20.85	73.01	28.56%
BGN11	24.25	144.29	9.83	145.46	191.01	169.85	335.40	50.64%
BGN12	10.80	102.78	4.78	—	—	10.94	102.90	10.63%
BGN13	18.12	139.81	12.01	6.39	—	24.64	139.94	17.61%
BGN14	37.73	235.48	18.72	88.29	14.05	126.20	249.67	50.55%
BGN15	34.06	121.13	18.87	—	—	34.21	121.25	28.21%
BGN16	27.01	102.90	17.21	—	—	27.15	103.02	26.35%
BGN17	17.15	97.55	10.57	—	—	17.29	97.69	17.70%
BGN18	18.36	97.81	12.16	—	—	18.51	97.97	18.89%
BGN19	18.34	106.23	14.85	—	—	18.49	106.36	17.38%
BGN20	25.42	111.90	15.69	—	—	25.58	112.01	22.84%
BGN21	57.87	309.03	30.12	—	103.87	58.02	413.03	14.05%
BGN22	27.12	125.64	16.79	—	—	27.28	125.80	21.69%
BGN23	57.27	287.86	28.87	2.55	—	60.01	288.03	20.83%
BGN24	117.23	453.17	71.14	—	691.24	117.42	1 144.58	10.26%
BGN25	117.07	486.32	64.41	602.78	858.30	720.08	1 344.82	53.54%
BGN26	52.79	204.39	27.80	—	—	52.99	204.57	25.90%
BGN27	155.23	563.64	114.57	937.84	1 023.81	1 093.32	1 587.68	68.86%
BGN28	123.25	625.99	57.66	3.18	—	126.68	626.21	20.23%
BGN29	127.28	771.47	54.02	1 378.76	1 419.96	1 506.30	2 191.67	68.73%
BGN30	237.04	1 098.24	70.22	—	1 801.97	237.37	2 900.48	8.18%
BGN31	256.94	1 164.47	92.91	1 852.06	2 663.61	2 109.28	3 828.37	55.10%
BGN32	168.36	610.48	90.70	—	—	168.67	610.74	27.62%

* Including generating time

Geometric mean: 26.44%

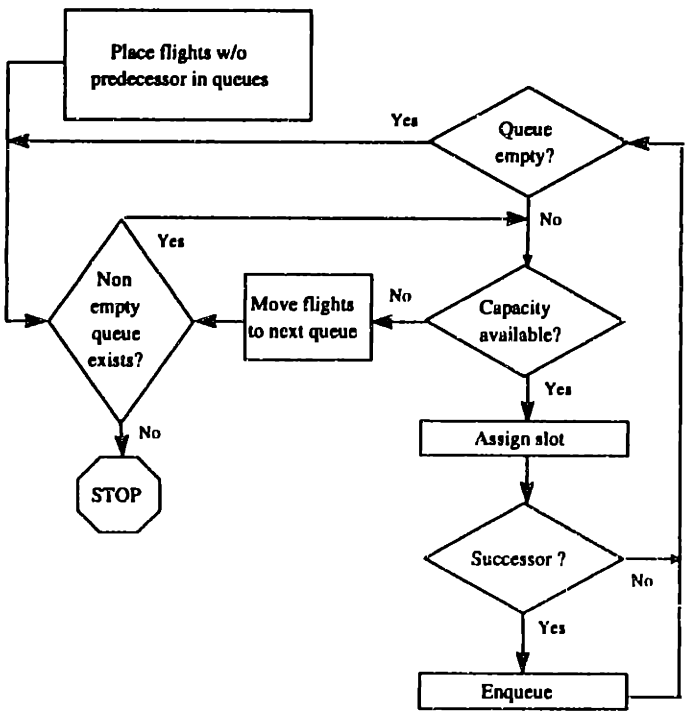
Chapter 3

The heuristic

3.1 The basic heuristic

The basic heuristic is a greedy algorithm based on dynamic priorities, of which we present a flow chart (Figure 3-1).

Figure 3-1: The basic heuristic: flow chart



Queues are ordered according to the flight priorities (descending order); every time we place a flight into a queue, its priority π_f is updated. A flight is not placed into a queue until its preceding flight has been assigned a time slot, so that we are guaranteed to have the correct value for the inherited delay. As a computational by-product, the number of flights in the queue is always kept at a minimum (corresponding to the number of itineraries), improving the algorithm's running time.

We make two remarks:

- The algorithm does not guarantee feasibility: there is no attempt to limit the amount of delay assigned to a flight.
- Nothing is specified about how to compute priorities π_f. Obviously, a flight's priority must depend on its delay, especially considering the previous remark. The definition of delay, however, is left to the "user". For example we may consider the delay already suffered, or the delay we would obtain by moving the flight to next available time slot. Furthermore, the priority could also be a function of other parameters or variables. We will give some examples of priority rules in Section 3.4.

3.2 The modified heuristic

A simple modification of the previous algorithm aims toward an improvement in terms of feasibility and of number of delays assigned.

In order to describe the modified heuristic we present a *pseudo-code* description (Tables 3.1 and 3.2) and a flow chart (Figure 3-2). The difference from the basic heuristic is the introduction of local optimization through the **swap** function.

The **swap** function relies on the assumption of unlimited capacities for departures. As pointed out in the introduction, this assumption gives us the freedom to reschedule flights without worrying about the departure capacities. The only restrictions left are those regarding the connections.

Table 3.1: *Pseudo-code* description of the modified heuristic: main program.

```
FOR EVERY FLIGHT f WITHOUT PREDECESSOR
  COMPUTE PRIORITY π_f
  PLACE f IN QUEUE(a_f, r_f)
FOR EVERY TIME t
  FOR EVERY AIRPORT a
    IF QUEUE(a,t) IS NOT EMPTY
      ASSIGN FIRST FLIGHT (f) TO SLOT (a,t)
      IF THERE IS A SUCCESSOR f'
        COMPUTE INHERITED DELAY η_f'
      IF η_f' > 0 OR t − r_f > δ*
        IF SWAP(f,t) FINDS f⃗
          SWAP SLOTS OF f AND f⃗
      IF THERE IS A SUCCESSOR f'
        UPDATE INHERITED DELAY η_f'
        COMPUTE PRIORITY π_f'
        PLACE f' IN QUEUE(a_f', r_f'+η_f')
    ELSE MOVE ALL FLIGHTS TO QUEUE(a, t + 1)
  NEXT AIRPORT
NEXT TIME
```

Table 3.2: *Pseudo-code* description of the modified heuristic: swap procedure.

```
PROCEDURE SWAP(f,t)

  FOR EACH TIME t̄ : t̄ ∈ T_f^↔ AND t̄ < t
    FOR EACH FLIGHT f⃗ ALREADY ASSIGNED TO (a_f, t̄)
      IF t ∈ T_f⃗^↔
        RETURN f⃗ (LEAVE PROCEDURE)
    NEXT FLIGHT
  NEXT TIME
  RETURN ∅ (NOT FOUND)
```

We consider the time window $T_f^{\leftrightarrow}$ within which we can freely move the flight:

$$T_f^{\leftrightarrow} = \begin{cases} \{r_f + \eta_f, \ldots, r_f + s_{ff'} + \delta_{f'}\} \cap T_f & (f, f') \in F_c \\ \{r_f + \eta_f, \ldots, r_f + \delta^*\} & (f, f') \notin F_c \end{cases}$$

This time window (i) is composed of feasible times ($\cap T_f$), (ii) begins with $r_f + \eta_f$ because we cannot ignore inherited delays, (iii) ends with $r_f + s_{ff'} + \delta_{f'}$ because this is the last time for which we can ignore effects on flight f' (if flight f' doesn't exist the window ends at the last feasible time, i.e., $r_f + \delta^*$).

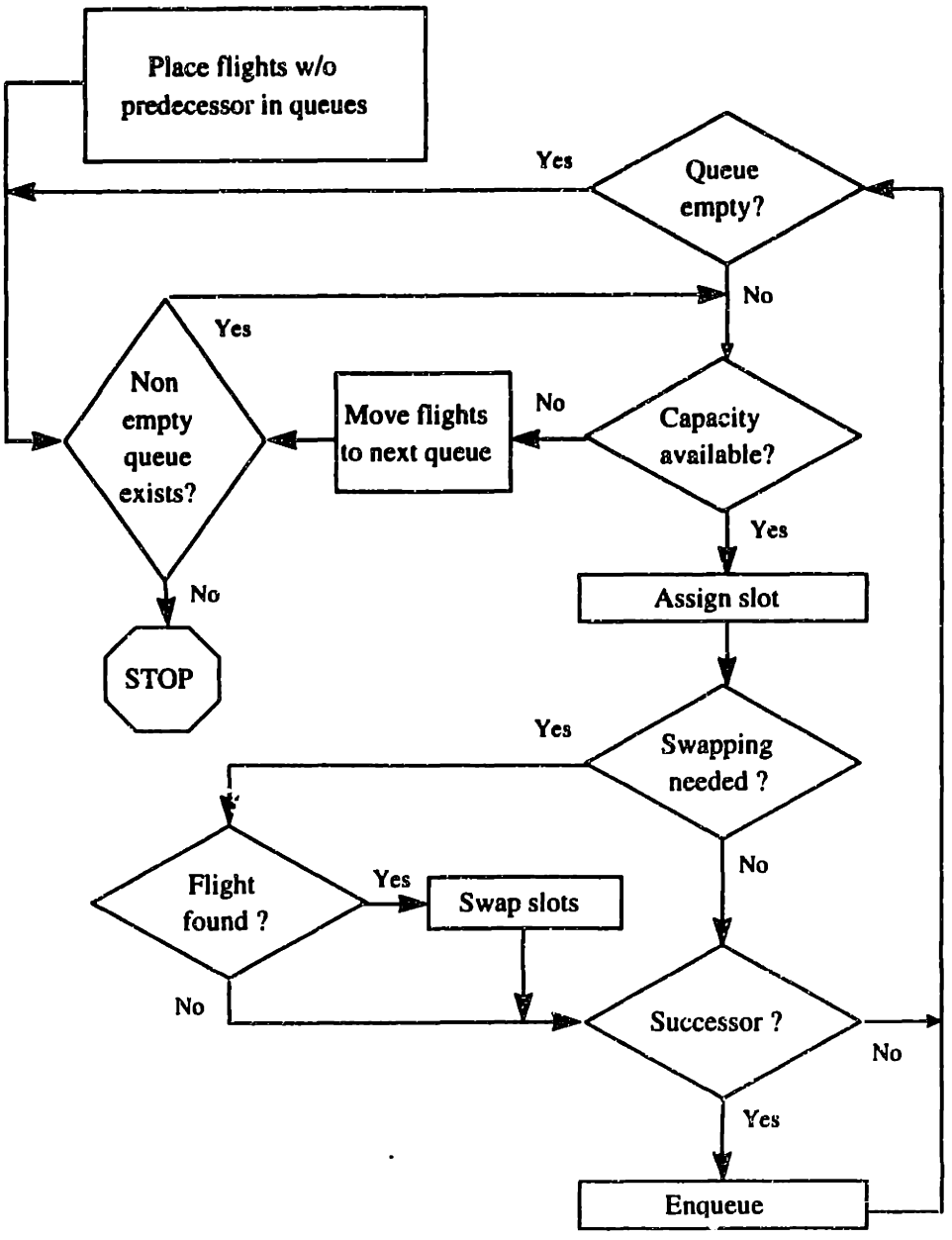
Thus, $T_f^{\leftrightarrow}$ may be considered a *dynamic feasible window*. Note that the window depends on both the previous flight (setting the value of η_f) and the successive one (through $\delta_{f'}$). When we assign a delay ($\delta_{f'} > 0$) to a flight, we are also expanding the dynamic feasible window of its predecessor.

Suppose we are about to assign a flight f to a time slot t and that this will cause either inherited delay or infeasibility. Then we check for the existence of flight $\vec{f}$, previously assigned to a time $\bar{t}$ ($\bar{t} < t$, $\bar{t} \in T_f^{\leftrightarrow}$), such that $t \in T_{\vec{f}}^{\leftrightarrow}$ (i.e., flight $\vec{f}$ can be assigned to time t without worrying about its successor). If such a flight exists, we can assign f to time $\bar{t}$ and $\vec{f}$ to time t; we call such an operation *swapping*.

Swapping has the following effects:

- The sum of the delays assigned to $\vec{f}$ and f does not change: an increase of the first is compensated by a decrease of the latter.
- If the **swap** function has been called because of inherited delay for f', the successor of f, then the inherited delay has certainly been reduced ($\bar{t} < t$).
- If the **swap** function has been called due to feasibility problems and $\bar{t} - r_f \leq \delta^*$, i.e., if the delay assigned to f is less or equal than the maximum delay allowed, then feasibility has been "restored".

Figure 3-2: The modified heuristic: Flow chart



3.3 Implementation

The heuristic requires fast access to data for each flight (that must be updated during the algorithm) and queue capabilities.

The first requirement calls for a static data structure, e.g., a matrix; the latter calls for a dynamic data structure, e.g., a linked list.

We implemented the heuristic in `C`, making use of the following data structures:

- A matrix, of dimension $|F|$, of **`structure flight`**, each with the following type of fields:
  - Data fields, e.g. $a_f, r_f, \delta_f, \eta_f, \pi_f$, etc.
  - Index fields, giving the position in the matrix of the preceding flight ($f^* : (f^*, f) \in r_c$) and the successive flight ($f' : (f, f') \in F_c$). This allows to retrieve the information on these flights, needed to compute the dynamic time window $T_f^{\leftrightarrow}$, in $O(1)$.
  - *Pointer* to **`structure flight`**: The flight will always be part of a list (specifically the queue of flights requesting landing or the list of assigned flights); this field points to the next element in the list.
- A matrix (double dimension: $|T| \times |A|$) of **`structure time_slots`**, each with the following fields:
  - Capacity.
  - Number of flights assigned.
  - *Pointer* to **`structure flight`**, pointing to the first element in the queue of flights requesting landing. This queue is ordered according to the priorities.
  - *Pointer* to **`structure flight`**, pointing to the first element in the list of flights assigned to the slot. This list is used during the search for flights available for swapping.

We conclude this section with a simple assessment of the heuristic's complexity:

- Each flight must be assigned once and may be placed in several queues before being assigned.

- The queues are maintained as ordered lists. In the worst case (i.e., all flights in the same queue), the enqueuing operation is in $O(|F|)$.
- There is no need to maintain the list of assigned flights in a specific order. Therefore the assignment operation is in $O(1)$.
- The enqueuing operation is executed at most $|T| \cdot |F|$ times.
- Since $|T| \ll |F|$, it follows that the complexity of the algorithm is in $O(|F|^2)$.

3.4 Some examples of priorities

In this section we give some examples of possible priority rules. All of them were used in testing the heuristic on the V and BGN instances presented in Section 1.3. In defining the priority rules, we will use a connection indicator function:

$$I_f = \begin{cases} 1 & \exists f' : (f, f') \in F_c \\ 0 & \nexists f' : (f, f') \in F_c \end{cases}$$

Priority D : $\pi_f^D = 2\delta_f + I_f$.

Our primary interest is in the amount of delay assigned; therefore higher delays correspond to higher priorities. When two flights are tied in delays, we give priority to a flight with a successor over one without (Table 3.6 shows the possible values given by the priority when $\delta^* = 4$).

Priority H : $\pi_f^H = \begin{cases} \delta_f + I_f \cdot \lceil \frac{\delta^*+1}{2} \rceil & \delta_f \leq \frac{\delta^*}{2} \\ 2\delta_f + I_f & \delta_f > \frac{\delta^*}{2} \end{cases}$.

If the delay suffered is more than half of the maximum feasible delay, we follow the same reasoning of the previous rule. For shorter delays, the importance of the delay and the existence of a connection are inverted: The priority is based on the existence of a successor; in case of tie, the delay already suffered by the flight is considered. There is an attempt to get rid of connecting flights as soon as possible, because they are the "difficult" ones to handle. Once the delay is large, however, maintaining feasibility become

Table 3.3: Priority table D

δ_f	$(f, f') \notin F_c$	$(f, f') \in F_c$
0	0	1
1	2	3
2	4	5
3	6	7
4	8	9

Table 3.4: Priority table H

δ_f	$(f, f') \notin F_c$	$(f, f') \in F_c$
0	0	3
1	1	4
2	2	5
3	6	7
4	8	9

as important as avoiding inherited delay. The threshold delay of $\frac{\delta^*}{2}$ is just a matter of convenience (Table 3.4).

Priority N : $\pi_f^N = \delta_f + I_f$.

The existence of a connection is considered as important as a unit of delay (Table 3.5).

Priority I : $\pi_f^I = I_f + 2 \cdot \min\{\delta^*, I_f \cdot \delta_{f'}\} + 2\delta^* \cdot \min\{\delta^*, \delta_f\}$.

This priority considers the amount of delay for the connected flight as well. In fact, the other priority rules may be considered to be "nearsighted", because the presence of a successor does not necessarily imply that there is inherited delay. Priority I uses information about inherited delay to choose between flights that have suffered equal delay.(Table 3.6).

It seems reasonable, when considering network effects, to consider how many legs follow the flight in the itinerary, because the more flights are still to be performed by the aircraft, the more potential effects a delay has. Previous work ([2]), however, suggests that priority rules based on the number of legs following the flight in the itinerary do not lead to improvements over priority rules based on the simple connection indicator function.

Table 3.5: Priority table N

δ_f	$(f, f') \notin F_c$	$(f, f') \in F_c$
0	0	1
1	1	2
2	2	3
3	3	4
4	4	5

Table 3.6: Priority table I

δ_f	$I_f = 0$	$\delta_{f'}$				
		0	1	2	3	4
0	0	1	—	—	—	—
1	8	9	11	—	—	—
2	16	17	19	21	—	—
3	24	25	27	29	31	—
4	32	33	35	37	39	41

3.5 Results

The heuristic was tested on the V and BGN cases using the priority rules presented in the previous section. Priorities D, H and I always led to feasible delay assignments, while priority N gave infeasible results in 11 cases for which the maximum delay was 5, i.e., one more than the maximum allowed.

In Table 3.7, we summarize the results in terms of distance from optimality. With the exception of one case (BGN2) the modified heuristic gave results within 5% of the optimal, averaging between 1.5% and 2% from optimal. The use of the **swap** function implied an improvement of a little more than 1% for all the rules.

The running time was always less then 0.1 CPU seconds, with the smaller instances running in 0.01 CPU seconds. In some cases the modified heuristic ran faster than the basic heuristic. The modified heuristic assigns fewer delays in total, therefore reducing the number of enqueing operations. This reduction may compensate for the time spent looking for swap possibilities.

Table 3.7: Comparison of performances

Heuristic		Distance from optimum						Average
		0%	$(0,1)$%	$[1,2)$%	$[2,3)$%	$[3,5)$%	$[5,\infty)$%	
H	No swap	4	4	6	11	9	5	2.63%
	Swap	8	8	9	6	7	1	1.60%
D	No swap	2	3	6	12	10	6	2.97%
	Swap	6	7	9	7	9	1	1.92%
N	No swap	2	4	3	10	4	5	2.75%
	Swap	6	6	8	3	4	1	1.50%
I	No swap	2	3	5	13	10	6	2.95%
	Swap	6	8	8	8	8	1	1.89%

Chapter 4

The integrated algorithm

4.1 Description of the algorithm

We now introduce the integrated algorithm, which uses the heuristic presented in the previous section to "help" the best integer programming model of Chapter 2 (i.e., the ABGE model). The idea stems from the following considerations:

- The heuristic is very fast and provides a near optimal solution. The value of this solution can be used as an upper bound for the branch & bound procedure, whenever the latter is required.

- The simplex method goes from one basic feasible solution to the other, improving the value of the objective function. If we provide an initial basic feasible solution, i.e., an advanced basis, we automatically reduce the number of solutions to be examined. Furthermore, our initial feasible solution is relatively close to the optimal; the reduction may be significant.

- If the coupling constraints were ignored, then the optimal solution of the relaxation would be integral. Therefore we may consider these constraints as the ones which lead to non-integrality. The solution of the heuristic satisfies these constraints, thus we hope that by starting from the heuristic solution, integrality will be more probably maintained.

- All variables in the ABGE model are binary. This means that every variable is always at its lower or upper bound and may therefore be considered basic or at its bound. The

solution provided by the heuristic will be an extreme point of the feasible region. Hence there will be no cross–over time (i.e., the time needed to "go" from a solution inside the feasible region to a solution corresponding to an extreme point).

The above considerations are clearly heuristic in themselves. The results of the next section show, however, that this kind of approach led to very good results.

The integrated algorithm (which we will refer to as ABGI) consists of two distinct stages: in the first stage the heuristic algorithm is run using different priority rules, while in the second the best result of the heuristic is given in input to Cplex©, which converts this result to an initial feasible solution and solves the problem. More specifically, at the end of the heuristic the decision variables corresponding to the delays assigned by the heuristic ($x_{f\delta^*}$ if the delay assigned is greater than δ^*) are passed to Cplex© as "variable at upper bound", while all the other variables are passed to Cplex© as "variable at lower bound". It is left to Cplex© internal algorithm to decide which variables to consider basic.

4.2 Results

Table 4.1 shows how the algorithm performed on the V and BGN test cases. The second column reports the time spent to run the heuristic with the 4 different priority rules (HDNI). In the third column we have the heuristics that achieved the best objective function value (that may be found in next column). Only one solution is given in input to the ABGE model, and this corresponds to the priority rule out of the parenthesis. In the fifth column, $\Delta_H^*\%$ is the percentage error relative to the optimum, i.e. $\frac{C_H - C^*}{C^*}\%$, where C_H is the cost given by the heuristic and C^* is the optimal cost.

Figure 4-1 compares the total running times for the BS, ABGE and ABGI models.

It is clear from Table 4.1 (compare with Table 2.7) and Figure 4-1 that the integration of the heuristic and the ABGE model substantially improved the performance of the algorithm. With only one exception (case BGN7), ABGI gave an integer solution without resorting to branch & bound. Even in case BGN7, the first node of the branch & bound resulted in an integer solution which took only 5 iterations to solve. It is worth noting that this case did not require branch & bound with ABGE and that in two cases (BGN15 and BGN26) ABGI

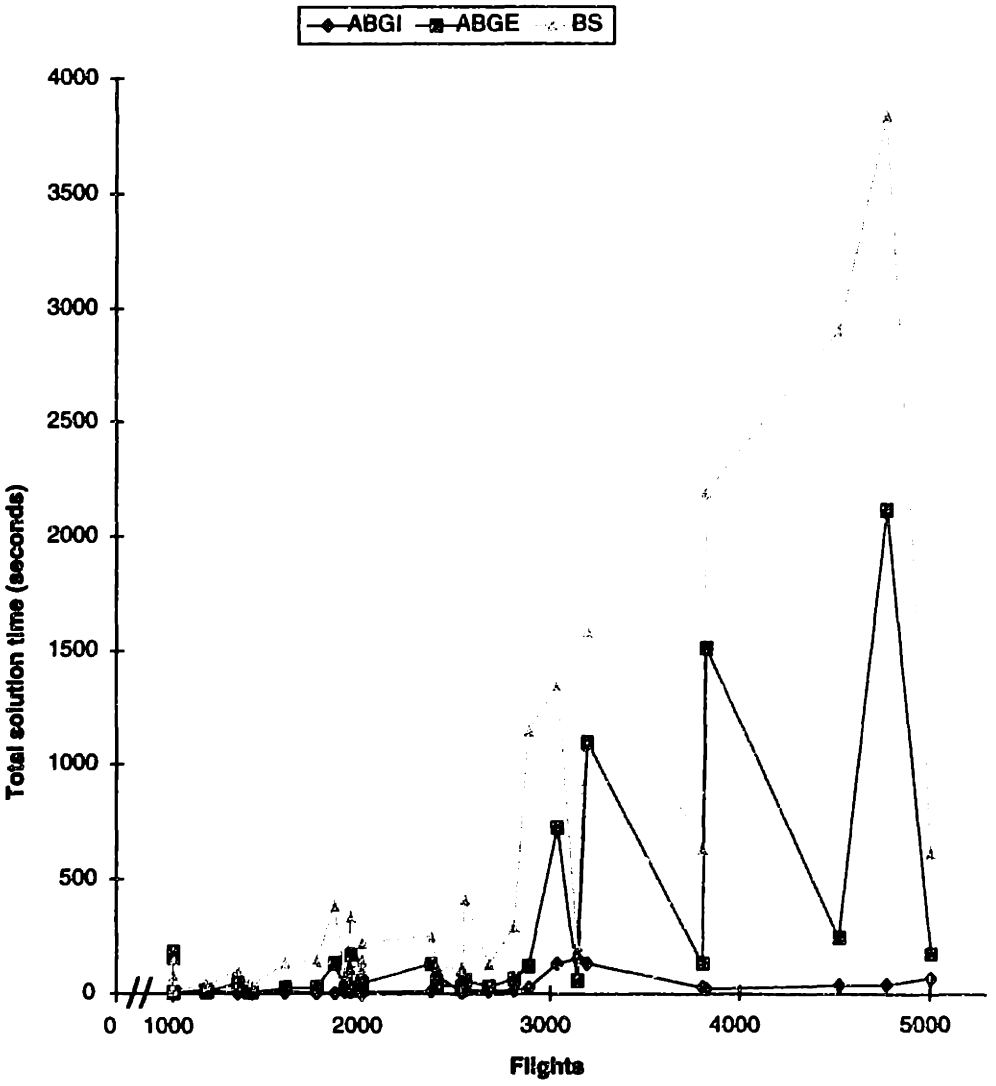
Table 4.1: Solving with ABGI: Detailed results

Case	Heuristic				Relaxed Problem				Total Time
	Search Time	Chosen (equiv.)	Obj. Value	$\Delta_H^*\%$	Time	Iter.	Non Int.	Optimal Value	
V1	0.04	H	71 000	0.00	0.61	176	0	71 000.00	0.70
V2	0.04	H (DNI)	56 000	0.00	1.50	324	0	56 000.00	1.61
V3	0.07	H (DNI)	84 800	0.11	2.57	466	0	84 700.00	2.71
V4	0.04	H (DNI)	65 000	0.00	2.27	373	0	65 000.00	2.41
V5	0.10	H	96 300	0.00	4.63	729	0	96 300.00	4.85
V6	0.09	H	93 300	1.19	10.02	1 190	0	92 200.00	10.27
V7	0.09	H (N)	72 400	0.28	13.10	1 354	0	72 200.00	13.35
BGN1	0.04	H (DNI)	13 200	0.00	0.75	138	0	13 200.00	0.83
BGN2	0.07	I	54 600	7.05	3.88	722	0	51 000.00	4.03
BGN3	0.04	I	26 500	1.34	1.53	245	0	26 150.00	1.66
BGN4	0.09	H (DN)	52 300	1.55	3.92	573	0	51 500.00	4.09
BGN5	0.06	H (DNI)	11 250	0.00	1.44	182	0	11 250.00	1.58
BGN6	0.06	H (DNI)	10 750	0.00	1.46	180	0	10 750.00	1.61
BGN7*	0.07	H (N)	60 500	2.71	5.20	609	14	58 900.00	5.84
BGN8	0.08	H (DN)	63 750	2.16	5.18	562	0	62 400.00	5.37
BGN9	0.10	H (D)	81 500	2.71	6.59	699	0	79 350.00	6.83
BGN10	0.08	I	43 150	1.05	4.42	539	0	42 700.00	4.62
BGN11	0.08	H (DNI)	55 300	1.00	3.50	332	0	54 750.00	3.72
BGN12	0.09	H (DNI)	22 400	0.00	1.69	115	0	22 400.00	1.92
BGN13	0.09	H (DI)	46 350	2.66	4.46	465	0	45 150.00	4.68
BGN14	0.09	H (DI)	64 150	0.94	5.82	509	0	63 550.00	6.09
BGN15	0.10	H (DNI)	71 550	1.49	71.64	4 077	0	70 500.00	71.89
BGN16	0.12	H (D)	52 550	0.77	7.81	775	0	52 150.00	8.07
BGN17	0.08	H (DNI)	25 400	0.59	3.57	313	0	25 250.00	3.79
BGN18	0.08	H (DNI)	27 150	0.56	3.84	350	0	27 000.00	4.07
BGN19	0.09	I	32 900	0.92	4.27	378	0	32 600.00	4.51
BGN20	0.12	H (DNI)	39 950	0.38	5.18	466	0	39 800.00	5.46
BGN21	0.10	I	77 050	3.77	11.72	946	0	74 250.00	11.97
BGN22	0.10	H (DNI)	44 250	0.80	7.34	638	0	43 900.00	7.60
BGN23	0.10	I	76 600	1.73	8.45	626	0	75 300.00	8.74
BGN24	0.16	I	142 500	3.00	19.86	1 415	0	138 350.00	20.21
BGN25	0.20	I	151 250	3.31	127.09	5 159	0	146 400.00	127.52
BGN26	0.13	I	81 900	1.17	147.17	5 584	0	80 950.00	147.50
BGN27	0.21	I	193 450	2.03	128.82	4 843	0	189 600.00	129.28
BGN28	0.18	I	117 150	3.12	21.72	1 234	0	113 600.00	22.15
BGN29	0.17	H (DNI)	126 600	2.10	14.53	789	0	124 000.00	14.96
BGN30	0.21	N	142 250	3.23	29.00	1 359	0	137 800.00	29.54
BGN31	0.24	I	178 150	2.24	30.15	1 374	0	172 450.00	30.67
BGN32	0.25	H (DN)	140 000	4.09	59.69	2 661	0	134 500.00	60.25

*B&B: Time 0.48, Iter. 5, Nodes 1

was slower than ABGE. This confirms the fact that the integrated algorithm cannot guarantee the improvement over the mathematical model: the solution of the heuristic may lead on "the wrong path" from the optimum.

Figure 4-1: Comparison of running times for ABGE, BS and ABGI.



Nonetheless, we were able to solve every test case in less than 2.5 minutes. Given these results, we decided to test the ABGI on more "realistic" instances. The next section describes the new instances and the results.

4.3 New test cases based on OAG data

The new cases are based on real OAG data, corresponding to January 13^{th} and July 3^{rd} 1993. The two data sets comprise respectively 63,455 and 55,106 flights, respectively, on a network of more than 100 airports.

For each flight, we had information on the official number, the departure/arrival airports, the departure/arrival times, and the equipment (i.e., the aircraft type). The information on connections is kept confidential by airlines and is not available in the OAG. For this reason, we implemented a C program to generate couplings. The program takes in input the entire OAG schedule, a list of airports of interest and two time thresholds, defined as minimum and maximum connection times. Two flights are considered connected if they are performed with the same equipment, by the same carrier and if the departure of the second flight is between the time of arrival of the first plus the minimum connection time and the arrival of the first plus the maximum connection time.

The minimum and maximum connection times were set to 20 and 40 minutes respectively. In order to increase the percentage of connections, we first relaxed the constraint about the equipment. Then we relaxed the constraint about the carrier. Thus we obtained three different instances for each day. The January 13^{th} instances comprise 22,522 flights with the percentage of connections equal to 69%, 72%, and 77%. The July 3^{rd} instances comprise 20,220 flights with the percentage of connections equal to 67%, 71%, and 77%.

The list of airports was the same for all the instances. It included the 23 airports that exceeded 20,000 hours of aircraft delay in 1993 ([6]): Atlanta Hartsfield, Boston Logan, Charlotte/Douglas, Washington National, Denver Stapleton, Dallas–Ft.Worth, Detroit, Newark, Honolulu, Houston Intercontinental, New York JFK, Los Angeles, New York La Guardia, Orlando, Miami, Minneapolis–Saint Paul, Chicago O'Hare, Philadelphia, Phoenix, Pittsburgh, Seattle–Tacoma, San Francisco, St. Louis.

For each airport, the value of the capacity was found running the heuristic on a problem in which the only capacitated airport was the one of interest and choosing the last value for which a feasible solution was obtained.

Table 4.2 reports the results on the new test cases. ABGI solves all instances in less than 20 minutes, without ever requiring branch & bound.

Table 4.2: Solving the OAG instances with ABGI

Case	Heuristic				ABGE			Total Time
	Search Time	Chosen	Objective Value	$\Delta_H^*\%$	Time	Iter.	Optimal Value	
Jan13.69	1.34	H	666 700	3.45	657.22	5 847	644 450	662.42
Jan13.72	1.44	H	674 750	4.60	701.13	6 288	645 100	706.21
Jan13.77	1.46	H	709 350	9.32	1 099.07	9 591	648 900	1 104.13
Jul03.67	1.40	H	639 250	3.93	606.04	6 176	615 100	610.58
Jul03.71	1.42	H	643 850	4.55	665.35	6 764	615 850	670.04
Jul03.77	1.42	I	669 650	7.87	884.45	8 799	620 800	889.20

Chapter 5

Conclusions and further research

The results of the last chapter showed that the integration of heuristic methods with good mathematical programming formulations can lead to substantial performance improvements in the solution of the MAGHP.

The good performance depends on two factors, and each of them deserves further attention:

- The heuristic gives surprisingly good results, especially considering its simplicity. It would be interesting to provide bounds on the distance from the optimal solution and characterize under which conditions good results are achievable.
- The value of the objective function for the relaxed ABGE and BS problem was always equal to the integer optimum, suggesting that this could be an inherent characteristic of these two formulations. Is it possible to prove such a result, or, again, identify the conditions under which this happens?

We believe the first phenomenon to depend on the *flatness* of the feasible region around the optimum: the problem has several equivalent (in terms of objective function) optimal solutions, and many other near–optimal solution.

Similar questions to the one investigated in this thesis could also be posed with regards to limiting capacities for other elements of the air traffic network.

The ABGE model can be easily modified to consider the case of limited departure capacities. We denote by $K^D_{a,t}$ the departure capacity of airport a at time t, c^g_f and c^h_f the cost of ground

and airborne delay respectively and consider the decision variables $y_{f\delta}$ equal to one if the delay of flight f on arrival is equal to δ. The extended model is:

$$\min \quad \sum_{f \in F} \left[c_f^g \sum_{\delta \in \Delta} \delta \, x_{f\delta} + c_f^b \left(\sum_{\delta \in \Delta} \delta \, y_{f\delta} - \sum_{\delta \in \Delta} \delta \, x_{f\delta} \right) \right] \tag{5.1}$$

s.t.

$$\sum_{\substack{(f,\delta): \\ (f \in F, \delta \in \Delta, \delta = t - d_f)}} x_{f\delta} \leq K^D_{a,t} \qquad \forall a \in A, \quad \forall t \in T \tag{5.2}$$

$$\sum_{\substack{(f,\delta): \\ (f \in F, \delta \in \Delta, \delta = t - r_f)}} y_{f\delta} \leq K_{a,t} \qquad \forall a \in A, \quad \forall t \in T \tag{5.3}$$

$$\sum_{\delta \in \Delta} x_{f\delta} = 1 \qquad \forall f \in F \tag{5.4}$$

$$\sum_{\delta \in \Delta} y_{f\delta} = 1 \qquad \forall f \in F \tag{5.5}$$

$$\sum_{d=0}^{\delta} y_{fd} - \sum_{d=0}^{\delta} x_{fd} \geq 0 \qquad \forall f \in F, \, \forall \delta \in \Delta \tag{5.6}$$

$$\sum_{d=0}^{\delta} y_{fd} - \sum_{d=0}^{\delta - s_{ff'}} x_{f'd} \geq 0 \qquad \forall (f, f') \in F_c, \, \forall \delta : (\delta, \delta - s_{ff'}) \in \Delta^2 \tag{5.7}$$

$$x_{f\delta} \in \{0, 1\} \qquad \forall f \in F, \quad \forall \delta \in \Delta \tag{5.8}$$

$$y_{f\delta} \in \{0, 1\} \qquad \forall f \in F, \quad \forall \delta \in \Delta \tag{5.9}$$

Note in (5.1) that the cost of airborne delay c_f^b is multiplied by the *difference* between the total delay and ground delay and that we need constraints (5.6) to ensure that the delay on arrival will be at least equal to the assigned ground delay, i.e., the delay at departure.

The model can be similarly extended to consider en route sector capacities (the actual formulation has not been included because it implies the use of cumbersome notation):

- The objective function is still (5.1) (under the reasonable assumption that the cost of airborne delay does not depend on *where* the airborne delay is incurred);
- Capacity constraints similar to (5.2) and (5.3) are added for each sector/time combination;

- Assignment constraints similar to (5.4) and (5.5) are added for each flight/sector traversed by the flight combination;
- Constraints similar to (5.6) are added to guarantee that the delay of flight f is not decreasing: one constraint for each couple of consecutive elements of the air traffic network used by the flight, starting from the couple (departure airport, first sector) and ending with the couple (last sector, arrival airport).

We suspect that the formulation will still be strong enough to avoid long branch & bound solution times, at least in the case with limited capacities in departure and unlimited en route sector capacities. However, it should be stressed that any constraint of type (5.6) could be considered equivalent to a coupling constraint of two copies of the same flight with slack time equal to zero. This means we are adding "difficult" constraints.

We can consider two different strategies for the implementation of the integrated algorithm in the case of limited capacities. The first strategy uses the same heuristic presented in this chapter as a starting point for the mathematical model, while the second strategy will consider the development of new heuristics.

The first strategy is based on the consideration that departure (and sector) capacities, as stated in the introduction, are seldom a problem. Therefore the solution of the heuristic will (hopefully) be feasible most of the time and slighly infeasible in other cases.

The second strategy aims at reaching a solution that satisfies the coupling constraints immediately (this gives also a starting bound for the branch & bound procedure). We believe that the development of the new heuristic will require much more ingenuity if one wants to achieve a performance similar to that of the heuristic presented in this thesis.

Bibliography

[1] G. Andreatta and L. Brunetta, 1995. "Multi–Airport Ground Holding Problem: A Computational Evaluation of Exact Algorithms," *Operations Research*, to appear.

[2] G. Andreatta, L. Brunetta and G. Guastalla, 1996. "Multi–Airport Ground Holding Problem: A Heuristic Approach Based on Priority Rules." In *Air Traffic Management '95*, L. Bianco, P. Dell'Olmo and A. Odoni (eds.), Springer Verlag, Berlin, Germany (to appear).

[3] L. Brunetta, G. Guastalla and L. Navazio, 1996. "Solving the Multi–Airport Ground Holding Problem," *Annals of Operations Research*, to appear.

[4] D. J. Bertsimas, A. R. Odoni, G. Perakis and K. Zografos, 1996. "A Critical Review of Optimization Models and Their Potential Application to the European Air Traffic Flow Management System," prepared for the NOAA Project of the European Commission (unpublished).

[5] D. J. Bertsimas and S. Stock, 1994. "The Air Traffic Flow Management Problem with Enroute Capacities," *Working Paper no. 3726-94 MSA*, Sloan School of Management, Massachusetts Institute of Technology, Cambridge, MA, USA.

[6] Federal Aviation Administration, 1994. "1994 Aviation Capacity Enhancement Plan," *Technical Report no. DOT/FAA/ASC-94-1*, FAA, Office of System Capacity and Requirements, Washington, DC, USA.

[7] P. B. Vranas, 1996. "Optimal Slot Allocation for European Air Traffic Flow Management," to appear in *ATC Quarterly*, 1997.

[8] P. B. Vranas, 1993. "The Multi–Airport Ground Holding Problem in Air Traffic Control," *Ph.D. Thesis*, Dept. of Ocean Engineering and Operations Research Center, Massachusetts Institute of Technology, Cambridge, MA, USA.

[9] P. B. Vranas, D. J. Bertsimas and A. R. Odoni, 1994. "The Multi–Airport Ground Holding Problem in Air Traffic Control," *Operations Research 42*, 249-261.