

A heuristic approach to solving the deterministic and stochastic air crew pairing problem



Lehlohonolo Masipa

Thesis presented in fulfillment of the requirements for the degree of
MSc (Operations Research)
in the Faculty of Science at Stellenbosch University

Supervisor: Dr Linke Potgieter
Co-Supervisor: Prof. Mapundi Banda

April 2019

Declaration of Authorship

By submitting this thesis electronically, I declare that the entirety of the work contained therein is my own, original work, that I am the sole author thereof (save to the extent explicitly otherwise stated), that reproduction and publication thereof by Stellenbosch University will not infringe any third party rights and that I have not previously in its entirety or in part submitted it for obtaining any qualification.

Date: April 2019

“And, when you want something, all the universe conspires in helping you to achieve it.”

Paulo Coelho

Abstract

The air crew pairing problem is a subproblem of the integrated airline management problem and one of the more complicated problems to solve in the field of transport logistics. There have been various approaches both in literature and in industry for solving the deterministic crew pairing problem. More recently there have been developments in solving stochastic instances of the problem. This study presents a heuristic approach to solve the crew pairing problem, for the one day planning horizon. An exact solution approach is also presented to generate the entire feasible region and produce the optimal solution, however this approach is computationally tasking for larger problem sets. To address the need to generate solutions in reasonable time for industry requirements, two heuristic solution approaches are presented; the first is based on a greedy tree search heuristic and the second is based on a random tree search with logical constraints to ensure feasibility in all solutions. A comparison is made with results for a benchmark study in literature in order to test the performance of the algorithms used in the deterministic crew pairing problem. The results are competitive for larger problem sets. Sensitivity analysis is also performed to generate "what-if" scenarios to test for algorithm robustness. The two heuristic approaches are also adjusted to solve the stochastic crew pairing problem, where weather delays are introduced in the flight network. The results illustrate that in most network problems given a delay as applied from a given distribution, an airline would need to employ additional crew to service a network of scheduled flights. The results also indicate that a greedy based algorithm performs better than a completely random monte carlo approach for both the deterministic and stochastic crew pairing problems.

Opsomming

Die skedulering en toekenning van bemanning aan vlugte is 'n subprobleem in die geïntegreerde lugrederyskeduleringsprobleem en ook een van die meer ingewikkelde probleme om op te los in vervoer logistiek. Daar is verskeie benaderings, beide in die literatuur en in die bedryf, vir die oplossing van 'n deterministiese bemanningskeduleringsprobleem. Daar is onlangs ook verwickelinge in die oplossing van stogastiese gevalle van die probleem gewees. Hierdie studie bied 'n heuristiese benadering tot die bemanningskeduleringsprobleem vir 'n een dag beplanningshorison. 'n Eksakte oplossingsbenadering word ook aangebied deur die hele toelaatbare gebied te genereer en die optimale oplossing te vind, maar hierdie benadering is berekeningintensief vir groter probleemversamelings. Om die behoefte aan te spreek om oplossings vir die industrie in 'n redelike tyd te vind, word twee heuristiese benaderings aangebied; die eerste is gebaseer op 'n gulsige boom soektog heuristiek en die tweede is gebaseer op 'n ewekansig lukrake boom soektog met logiese beperkinge om die toelaatbaarheid van alle oplossings te verseker. Resultate word vergelyk met die van 'n maatstaf studie in die literatuur om die gehalte-prestasie van die algoritmes te bepaal wat gebruik word in die deterministiese bemanningskeduleringsprobleem. Die algoritmes is mededingend vir groter probleem stelle. Sensitiwiteitsanalise word ook uitgevoer om verskillende scenarios te toets om die robuustheid van die algoritmes te bepaal. Die twee heuristiese benaderings word ook aangepas om die stogastiese probleem op te los, waar vertraginge lukraak in die vlug netwerk veroorsaak word as gevolg van weerstoestande.

Acknowledgements

My sincerest thanks in completing this Masters goes to the following people:

My family and friends, especially my mother who has always been supportive and enthusiastic throughout all my studies.

My supervisors; Dr Linke Potgieter and Prof Mapundi Banda for their immense patience and understanding as I took on this research whilst working full time. They helped me challenge own thoughts and ideas by ensuring that I had a thorough understanding of what I was doing at each step of the way.

Lehlohonolo Major Masipa and Richard Mhango for their technical support and assistance in programming and encouraging me to learn java and interact independently on stackoverflow to figure out code script.

Contents

Declaration of Authorship	iii
Acknowledgements	ix
1 Introduction	1
1.1 Motivation	1
1.2 Problem description	4
1.3 Scope and objectives	5
1.4 Thesis outline	5
2 Air Crew Pairing and Optimisation of the Duty Generation Problem	7
2.1 Input considerations and terminology	7
2.2 Cost considerations	9
2.3 Air crew pairing problem formulation	9
2.3.1 Generation phase	10
2.3.2 Optimisation phase	10
2.3.3 Variations of CPP formulation	10
2.4 Solution approaches	12
2.4.1 Exact approaches	12
2.4.2 Heuristic and meta-heuristic approaches	13
2.4.3 Stochastic approaches	16
2.5 Considering uncertainty in the CPP	16
2.5.1 Expected value linear programming	17
2.5.2 Stochastic programming	17
2.5.3 Stochastic CPP formulation and solution approaches	18
2.6 Duty generation and crew pairing in industry	21
2.7 Chapter Summary	22
3 Methodology and Results - Deterministic Problem	23
3.1 Informal problem description	23
3.2 Modelling approach and assumptions	23
3.2.1 Problem assumptions	25
3.2.2 Modelling objective	25
3.3 Data description	27
3.4 Mathematical formulation	27
3.4.1 Generation of feasible duties	28
3.4.2 A theoretical framework to generate the entire feasible region	28
3.4.3 Minimum Cost Pairing Algorithm 2	33
3.4.4 Minimum Cost Pairing Algorithm 3	33
Monte Carlo Concepts and Convergence	36
3.5 Computer Implementation	36
3.5.1 Data Transformations	37
3.6 Results	38

3.7	Sensitivity analysis	40
3.8	CPU time	42
3.9	Chapter Summary	44
4	Methodology and Results - Stochastic Problem	47
4.1	Delays in air crew planning	47
4.2	Modelling approach	49
4.2.1	Problem Assumptions	50
4.2.2	Randomness and sampling	50
4.2.3	Delay function	52
4.2.4	Optimisation objective	53
4.2.5	Mathematical formulation	53
4.3	Minimum Cost Pairing Algorithm 4	54
4.3.1	A note on Monte Carlo results	57
4.3.2	Results - Stochastic problem	58
4.3.3	Sensitivity analysis	59
4.4	Chapter summary	61
5	Conclusion	65
5.1	Summary of research	65
5.2	Contribution to Literature	66
5.3	Recommendation for future work	66
5.3.1	Recommendation for computer language implementation of the air crew pairing problem	68
A	Graph Theory	69
A.1	Graph representation of a flight task	69
B	Data Inputs from OR library	71
B.1	Input data for adjacency matrix	71
B.2	Transformed adjacency matrix	71
B.3	Input data for start times	72
B.4	Transformed Start time matrix	72
C	Code extracts	73
C.1	Extracts of flight task objects	73
C.2	Extracts of duty objects	74
	Bibliography	75

List of Figures

2.1	Duty period example	11
2.2	<i>A taxonomy of Stochastic Programming solution methodology [79].</i>	18
3.1	Feasible Duty Region	24
3.2	Graph network of duties	24
3.3	Non-optimal crew allocation	26
3.4	Optimal crew allocation	26
3.5	Comparison of average number of crew.	39
3.6	Comparison of variable connection cost between Beasley, MCP-2 and MCP-3.	39
3.7	Comparison of variable connection cost between Beasley, MCP-3 min, max and avg iteration values	41
3.8	Comparison of variable connection cost between Beasley, MCP-3 min and max iteration values	42
3.9	MCP-2 results - no delay, early vs late index sequence start.	44
3.10	Adjacency matrix for network size $\mathcal{F} = 50$	44
4.1	Delay by cause, 2008 to 2013	48
4.2	<i>Flight delays due to weather and other events, from October 2015 [16].</i>	49
4.3	Distribution of flight delays due to weather in minutes.	51
4.4	MCP-4a, MCP-4b and MCP-2(no delay) comparison of C_f	59
4.5	MCP-4a, MCP-4b and MCP-2(no delay) comparison of C_k^v	60
B.1	CPP-50 raw file in .txt format	71
B.2	CPP-5 transformed adjacency matrix	71
B.3	The original file formats from OR library in .txt format	72
B.4	The corresponding start time matrix for the CPP-5 network	72
C.1	Object class extract for flight tasks	73
C.2	Object class extract for duty	74

List of Tables

1.1	Integrated problem taxonomy	3
1.2	South African airlines delay statistics	3
3.1	Allocation Costing Comparison	27
3.2	Example CPP-5, duration table	30
3.3	Temporary feasible duty set 1	30
3.4	Temporary feasible duty set 2	31
3.5	Temporary feasible duty set 3	31
3.6	Temporary feasible duty set 4	31
3.7	Temporary feasible duty set 5	31
3.8	Temporary feasible duty set 6	31
3.9	Temporary feasible duty set 7	31
3.10	Temporary feasible duty set 8	32
3.11	Temporary feasible duty set 9	32
3.12	Temporary feasible duty set 10	32
3.13	Temporary feasible duty set 11	32
3.14	Temporary feasible duty set 12	32
3.15	Temporary feasible duty set 13	32
3.16	Temporary feasible duty set 14	33
3.17	MCP-2 Results, Example CPP-5 - feasible duties	34
3.18	Program Layout, duration table	37
3.19	MCP-3 results - no delay	40
3.20	MCP-2 results - no delay, random starting point	42
3.21	MCP-3 results (p=1000) - no delay, random start point	43
3.22	Adjacency matrix sparsity	43
3.23	CPU time - comparison	45
4.1	A 45 minute delay at flight task i and delay at possible adjacent tasks	53
4.2	Example CPP-5, duration table	55
4.3	Example CPP-5, moderate delay impact for all flight tasks in network	56
4.4	MCP-4, 45 min single point delay, $\alpha = 0.5$	56
4.5	MCP-4a 15 min single point delay	61
4.6	MCP-4a 45 min single point delay	61
4.7	MCP-4a 90 min single point delay	62
4.8	MCP-4a 120 min single point delay	62
4.9	Different delay function output for varied α values	62
4.10	MCP-4a with different values of recovery parameter α	62
4.11	MCP-4b results - delay at an early vs late index sequence start	63

List of Abbreviations

SP	Stochastic Programming
CSP	Crew Scheduling Problem
CPP	Crew Pairing Problem
DCPP	Dynamic Crew Pairing Problem
SCPP	Static Crew Pairing Problem
LP	Linear Program
ILP	Integer Linear Program
GDP	Gross Domestic Product
MCP	Minimum Cost Pairing
OOP	Object Oriented Programming
CPU	Central Processing Unit
NP	Non-deterministic Polynomial-time

List of Variables

\mathcal{F}	a network of flight tasks
i	flight task in a network \mathcal{F}
j	flight task connected to i in a $duty_k$
k	a duty of sequential flight tasks
K	total number of duties required to service a network of flight tasks in \mathcal{F}
C^f	fixed cost for assigning a crew to a duty, described as a duty cost
C_k^v	variable connection cost for a duty k
S_i	start time of flight task i
E_i	end time of flight task i
T_i	duration of flight task i
T_k	duration of duty k
α	airline recovery parameter
d	delay duration
p	iteration counter
P	total number of iterations

Dedicated to my sister Emma, RIP...

Chapter 1

Introduction

The global airline industry is one of the most competitive industries in the world, with very tight profit margins but large and increasing flight volumes. The revenue generated by this industry globally in 2018 was \$834 billion [39]. In South Africa alone, the airline industry contributes approximately 3.5% to the annual GDP and created 230 000 jobs in 2014 [63]. Due to the level of competitiveness in the industry, revenue growth is driven by low cost flight carriers, optimal fleet, crew and demand management. Airline planning is a large integrated problem that has many components to it that need to be managed. Each of these components have certain cost considerations, for example, fuel, aircraft rental or amortisation, crew salaries, ground staff salaries, craft maintenance, catering, insurance, airport fees, taxes and administration costs. Optimal management of the different components in airline planning, can lead to increasing profit margins as costs are minimised, and airline companies are continuously searching for new ways to improve their planning methodology. One of the main cost drivers include the cost of labour [35, 30]. Airlines attempt to manage this cost by effective work scheduling of their staff. In this thesis, the airline crew pairing problem (CPP) is addressed. Specifically the crew pairing problem is considered for both the deterministic and stochastic cases in which delay scenarios are introduced.

1.1 Motivation

Airline companies have to manage a number of components in order to satisfy customer demand for flights to multiple destinations at various times throughout the day, week and year. As such, the planning and scheduling of aircraft and crew is a complex, multi-stage task. There are four main stages in the airline planning process, namely schedule design, fleet assignment, maintenance scheduling, and crew scheduling. Crew scheduling is further divided into two stages, namely crew pairing and crew assignment.

The integrated planning problem is usually solved in stages, as each of the planning components have their own sets of constraints and input parameters. The integrated airline planning problem can further be classified into short, medium and long haul problem types, depending on the time horizon used by the airline company for their planning routines. At a high level some of the considerations at each stage of the problem are [30]:

- **Schedule Design** - This is where a schedule or timetable for the various routes is generated. Demand considerations are based on market research, gap in the market and capital availability to service a chosen new or existing route between one or more destinations.
- **Fleet Assignment** - This stage requires the optimal assignment of the right size fleet type to meet the scheduled demand requirement. Fleet are the actual planes required to service a route. Domestic fleet range from small passenger planes to larger airbus

fleet types. Airline companies will usually purchase some of their own branded fleet or charter (rent) planes to support their fleet requirements.

- **Maintenance Scheduling** - This stage is also referred to as *maintenance routing*. Regular fuel and maintenance checks are required to service the fleet, and these need to occur during non-operational times. Alternatively the time taken for maintenance may be optimised to ensure that some fleet are available for duty.
- **Feasible duty creation** - This is the creation of sequential flight tasks within regulatory constraints, usually solved simultaneously in the pairing problem as one of the constraints of the pairing problem.
- **Crew Pairing** - This is the creation and optimisation of feasible sequential duties over a time horizon to service short, medium and long haul schedule designs.
- **Crew Assignment/Rostering** - This is the process of assigning pairings/duties to specific crew members based on their duty requirements and the skills of the crew member to service the flight task and fleet type. In general, cockpit crew are usually licensed to fly specific aircraft type, whereas cabin crew can be qualified to fly more than one type of aircraft [30].

To illustrate this complex multi-stage task, consider a single flight operator that would like to service 3 destination routes, namely, Johannesburg, Cape Town and Durban. The operator will need to design a schedule based on the market demand and market gap, such that the schedule is profitable. The operator may choose to have numerous flights at fixed times on a daily basis between each of the cities. The operator will need to first ensure that they have the required number of fleet to service these flights at the various times. In order to save costs the operator may set the flight times between the cities such that a single aeroplane may do multiple trips sequentially starting from a base. In so doing the operator can 'optimise' their aeroplane capacity usage to meet customer demand. The fleet will require servicing from time to time, in order to comply with aviation safety regulations. Maintenance schedules will need to be drawn up for all aeroplanes within the fleet, such that they can still service the targeted flight requirements as per the design schedule. Once the aeroplanes have been allocated to the various flight and maintenance plan, human resources must be assigned to service the flights per the fleet training and airline regulatory requirements. This means that the correctly qualified staff must be assigned to the relevant fleet as per the flight schedule. In addition to specific training requirements for staff, there are regulatory requirements that provide restrictions and guidelines that govern the time structure of crew schedules in line with labour laws. These guidelines include considerations such as rest, maximum duty and layover periods to name but a few. A flight operator needs to consider the above regulations when setting a crew schedule for staff that covers all the flights that have been scheduled over a given time horizon. The design of feasible duties or pairings that cover the flight tasks in a given network and meet the regulations is known as the air crew pairing problem (CPP). As with any schedule based organisation, certain crew may have preferences with regards to which flight tasks they would like to service. Additionally, there are leave requirements that may need to be considered. The process of assigning specific individuals to duties as per team preferences is known as rostering or crew assignment.

Given all the above requirements the planning department for any airline is faced with a complex task in order to manage their operations optimally and minimise their costs, let alone maintain or grow market share in a multi operator environment. In Table 1.1, the various components in the integrated airline planning problem are listed, along with an example with the cost or revenue input that is optimised.

Integrated planning component	Example	Cost or revenue input
Schedule Design	<i>Time:</i> 12:00 <i>Flight Task:</i> JNB - DBN	Revenue generated by demand for route
Fleet Assignment	<i>Flight Task:</i> JNB - DBN <i>Fleet:</i> A340	Fixed cost of fleet type (rental or amortised cost)
Maintenance Routing	<i>Fleet:</i> A340 <i>M schedule:</i> Mon - 19:00	Fuel and maintenance cost
Feasible duty creation	<i>Duty_k:</i> [JNB - DBN] → [DBN - CPT] → [CPT- PE]	Cost of a feasible duty
Crew Pairing	<i>Pairing_p:</i> <i>Duty₁</i> → <i>Duty₂</i> <i>Crew:</i> Team-1	Cost of a feasible pairing

TABLE 1.1: An integrated problem taxonomy table that details the various components of the integrated problem of which crew pairing is a subset.

In addition to the large planning requirement, as with most industries and especially with flying, there are unforeseen circumstances that may disrupt a pre-determined schedule and cause delays. These may include, maintenance, problems with aircraft, extreme weather conditions, late arrival of passengers, cabin crew, pilots and security issues. Such delays lead to a ripple effect in a network of scheduled flights across the country. The flight arrival and departure statistics for a local, South African airline carrier are given in Table 1.2. For a single year, 17% of scheduled flights were late.

Status	Observations
On-time	83%
Late	8%
Very Late	3%
Excessive	3%
Cancelled	2%
Diverted	1%

TABLE 1.2: South African airlines delay (weather and non-weather) statistics for the period October 2014 to September 2015 [57].

Costs associated with flight delays include direct costs such as customer compensation, overtime, extra crew, additional fleet requirements and other indirect costs such as reputational damage that may lead to subsequent market share loss. On average, airlines can lose \$32 million per year due to the various delay factors [63]. The cost implications vary with the severity of the delay and the affected demand (number of customers). Airlines still need to maintain profitability and minimise costs due to delays caused by uncertain events.

One of the major costs for airlines is their crew resources. This includes pilots, co-pilots, ground staff and cabin crew. Typically, the average cost per flight for a captain is \$430 and \$40 per cabin crew member [63]. This can quickly become quite an expensive overhead even for domestic flights, the cost of which can be up to \$120,000 per day on air crew alone for a domestic carrier in South Africa¹. Given that crew represents such a large cost factor for airlines, the effective assignment of crew to flight tasks, is one of the most important air

¹Given 64 domestic flights a day on 2 hourly flights across the country, each with 2 pilots and 4 cabin crew members

planning aspects that contributes to cost minimisation. In addition, by also considering the crew cost implication due to delays in their pre-determined flight schedules, airlines may reduce expenditure on excess crew by a substantial margin.

1.2 Problem description

Airline operations are growing with the increasing level of globalisation. Airlines need to manage their capital and labour costs effectively in order to remain profitable. In particular, the effective management of staff to service the growing flight demands is important to ensure good customer service.

The effective management of staff has to be in line with labour regulations and ensure that staff are not over-stretched, especially given the non-standard hours for flying. Staff also need to be allocated to services for which they have been trained. This means that pilots should fly aeroplanes for which they are licensed, and crew should service fleet based on their qualifications. Whilst ensuring good labour practices in terms of working hours and qualifications, airline companies must also optimise the productivity of their air crew employees whilst on duty in adherence to a fair remuneration structure. Furthermore, airline companies must ensure that they employ enough crew to meet their flight schedule and fleet type demands. At the same time, airline companies should not employ too many crew members if they will not be working long enough hours to get suitable wages or spending too much time in between flights idle. These are some of the regulatory constraints and employment considerations taken into account in the air crew pairing problem which make the problem computationally difficult to solve efficiently, especially for very large airline operations that span across many flight destinations.

Many algorithms have been developed to address the air crew pairing problem, and to solve for different 'what if' scenarios for airlines. The air crew pairing schedules that are generated can be subject to a very wide range of 'what if scenarios'. The sensitivity analysis in this problem has many adjustable parameters, including changes in number of available staff and crew, changes in the start time of airlines due to many delay factors and changes in the number of available flights due to maintenance issues. Chapter 2 provides an extensive review of some of the algorithms presented in literature and those used successfully in industry by many commercial airlines.

There is limited literature that explores and provides approaches to the stochastic CPP or literature that extensively considers simple perturbations for "what if analysis" that is typically provided in the crew pairing systems used in industry. Additionally, the gap in literature currently (to knowledge) is the implementation of monte carlo simulation to provide approximations to a feasible solution of the CPP given a delay in a single or multiple flight task start times in a flight network. Most literature provide solutions that may not always be feasible or at least a subset of solutions where not all the regulatory constraints and parameters are adhered to. This study explores a heuristic approach based on monte carlo simulation to ensure that regardless of the size of the flight network, or a given delay, a feasible crew pairing solution is generated for the deterministic case and the stochastic equivalent of the problem.

1.3 Scope and objectives

There has been a wealth of literature that explores the air crew pairing and assignment problems using various methodology. In this study, the scope is limited to the daily air crew pairing problem, in which daily duties are generated and matched to crew, rather than pairing crew to sequences of duties spanning more than one day. Delays are considered for the stochastic case, however, only weather is considered as the possible cause of delay.

The following have been identified as the research objectives of this study:

- I. Review literature on air crew pairing solution methodology.
- II. Review literature on various approaches used to solve the dynamic air crew pairing problems.
- III. Develop a methodology to solve the daily air crew pairing problem,
 - i. for the deterministic case;
 - ii. for the stochastic case where weather attributed delays are introduced randomly.
- IV. Analyse and compare results to benchmark literature studies. For both the deterministic and stochastic problems, sensitivity analysis is performed to assess the impact of perturbations in some of the decision parameters;
- V. Provide directions for future research.

1.4 Thesis outline

In Chapter 2, the air crew pairing problem is presented along with the terminology and concepts used in this study. In addition, a literature review of the air crew pairing problem is presented including various studies that have solved both the integrated problem and components of it for the static and dynamic case. A brief review is also provided for the various approaches to solving stochastic linear programming problems, in particular, heuristic techniques applied in solving air crew scheduling problems under uncertainty. In Chapters 3 and 4 the solution methodology is discussed for the deterministic problem without delay, and stochastic instances of the air crew pairing problem, respectively. In both chapters, some sensitivity analysis is performed by varying some parameters in the model and comparing the results to a base case. The data used and results attained by the methodology implementation for both the deterministic and stochastic problems are detailed and discussed in their respective chapters along with a comparison against benchmarked approaches that have been used to solve the problem using the same data set. Concluding remarks on the results and discussions of future work are provided in the final chapter.

Chapter 2

Air Crew Pairing and Optimisation of the Duty Generation Problem

In this chapter, the air crew pairing problem for various time horizons is defined along with important industry specific terminology that differs from the general crew pairing problem. The typical model formulation is presented, along with a discussion on the generation phase and optimisation phase of the problem. Different solution approaches are discussed as well as the application of optimisation technology in industry. An explanation of the cost considerations specific to the air crew pairing problem is also provided.

2.1 Input considerations and terminology

In the air crew pairing problem (CPP), each crew is assigned a sequence of flight tasks in such a way as to ensure a minimum cost. A crew is regarded as an entity, and the individual members of the crew are not considered in this problem.

The CPP has many inputs and terminology used to describe the various elements in the problem. In addition, there are inputs that need to be considered before attempting to attain a feasible pairing solution that meets regulatory and commercial demand requirements. The list below provides some of the key terms that will be used throughout the thesis and in the literature study:

- *Flight task* - a non-stop flight between two destinations. Each flight task has a specified *duration*, along with fixed and variable costs. A flight task may also be referred to as a *flight leg*.
- *Connection* - when two flight tasks are adjacent and can potentially be completed sequentially, the end time of the first flight task and the start time of the connecting flight task are chronological.
- *Connection cost* - Also referred to as an *adjacency cost*, is the cost of connecting two sequential flight tasks that are connected as per the requirements for a *connection*. This cost includes the transport cost and possible idle time between flight tasks.
- *Duty period* - a legal sequence of consecutive flight tasks over the span of one work day [31]. A duty period has a maximum *duration* along with fixed and variable costs depending on the number and length of all the constituent flight tasks. A duty period is also referred to as a *duty*, or *duty path*.
- *Pairing* - an assignment of a sequence of duties to a crew. Crew pairings are such that crew may return to base or end up at a destination of choice. A pairing can span over one or more days, depending on the planning horizon. In this study, the planning horizon for the pairing is one day, therefore only the design of duties are considered.

- *Deadhead* - a flight task used to transport an off-duty crew or crew member from one destination to another in order to fulfil a flight duty as per the crew assignment schedule. In this study the assumption made is that deadheads are implemented prior to commencing a duty.
- *Brief time* - the time used to brief crew at the start of a duty period.
- *Flying time* - the duration of a single flight task, that typically excludes brief time, delays or boarding.
- *Connection time* - the time between two consecutive flight tasks in a duty period. During this time, aircraft refuelling, and preparation for the next flight task is done by crew on duty. Crew may switch between aircrafts if necessary.
- *Idle time* - time intervals during a duty period where crew are not assigned to a flight task. Idle time occurs between subsequent flight tasks periods.
- *Roster* - a sequence of pairings and other flight duties (not necessarily in flight) that would be assigned to a specific crew member.
- *TAFB* - time away from base.

The construction of feasible duties and pairings are subject to various regulatory labour and capital constraints. A typical crew would include pilot, co-pilot and air hostess(es). The different skills of each type of crew member need to be matched to ensure that a crew is adequately staffed to service a flight task, duty and pairings. This requires the consideration of labour laws, much like any other form of employment, to ensure the fair treatment of staff and the safety of all on board. In addition, an airline company might have a limited number of staff to allocate to the various flight tasks and duties in their schedule design. If the problem includes long-haul operations there are many more considerations to be made that ensure that crew preferences and safety regulations are met. Cabin crew member cannot cover more flights in a given period, than he/she is legally permitted to. Some important regulatory constraints include:

- *Maximum flying time* - limited to 8 hours for pilots. This may be a single flight task or a sequence of flight tasks and connection times.
- *Maximum duty period duration* - includes flight time and non-flying time, and is usually restricted to 8 - 10 hours. There may be different policies on the maximum duty period in different parts of the world. Furthermore, there is a restriction on the total elapsed time a crew member may be on duty during a duty period [43].
- *Maximum sit time* - the maximum period that a crew member may be away from their domiciled base but not on flight duty.
- *Maximum duty length* - the maximum number of flight legs in a duty.
- *Maximum pairing length* - the maximum number of duties in a pairing.
- *Maximum and minimum rest period* - during a pairing, in between duties, pilots and staff are required to have a minimum rest period depending on the duration of the preceding duty period. Idle time between consecutive flights is allowed not shorter than a minimum sit time and not longer than a maximum sit time. Only regulatory compliant pairings should be feasible, i.e. crew are not flying a continuous set of duties without the required rest or leave period and there should be no overlaps in the pairing.

- *Minimum allowance* - a cost incurred during flight pairings when crew are away from base.

In this study a pairing is completed for a specified time horizon. If crew are not back at their home-base at the end of the horizon, they are allowed to be passengers on a connecting flight back to base after their duties' end. These constraints will change depending on the time horizon used in planning a pairing cycle. Some pairing cycles are as short as 2 to 3 days and can also get as long as 1 week cycles. For longer pairing time horizons, more planning is required to ensure that all the regulatory constraints are met. Where possible, the pairing cycles should be optimised given the limited resources. The cycles should also consider overnight and stoppage times that occur when there is a period of 8 hours or more within a day cycle that has no scheduled flight tasks (usually at night) especially in the domestic planning problem.

2.2 Cost considerations

The cost of crew can either be calculated on a credit guarantee based system [30] or crew can be paid a fixed salary. The calculation of the credit guarantee based system is given as:

$$\max(F_d \times W_k; M \times W_k; I_d \times W_k)$$

where the F_d is the total actual flying time in a duty period d , M is the minimum duty period, W_k is the crew (k) type specific rate which would be different for cockpit and cabin crew depending on their qualification, I_d is the idle time in duty period d . The $M \times W_k$ is the guaranteed income for a duty, which is usually (5/8) of the length of the duty. An ideal pairing is one such that the duties and pairings minimise the idle periods (non-flying time in a duty) and optimise the minimum and maximum sit periods within a pairing. This would be particularly important for a fixed cost structure, to avoid too many hours where staff are not working or non-productive but are being paid. In general the cost of a duty or pairing is directly proportional to the wages of the crew and a function of the duration of the duty or pairing. The deadhead cost may be included as a penalty cost in the objective function for a pairing or duty problem formulation [41]. If airlines are seeking to reduce costs and maximise profits, the consideration of a scheduling technique to optimise their crew is important, regardless of their pay structure that is also influenced by the jurisdictions in which they operate.

2.3 Air crew pairing problem formulation

The objective of the CPP is to find a feasible minimum cost assignment of crew that cover all flights in the flight schedule. The CPP is typically formulated as a set partitioning problem. More formally, for a set of feasible flight pairings \mathcal{P} , and a set of flight legs, \mathcal{F} , to be covered, the CPP is

$$\begin{aligned} \min \quad & \sum_{p \in \mathcal{P}} c_p x_p \\ \text{subject to :} \quad & \sum_{p \in \mathcal{P}} a_{ip} x_p = 1, \quad \forall i \in \mathcal{F} \\ & x_p \in \{0, 1\} \quad \forall p \in \mathcal{P} \end{aligned} \quad (2.1)$$

where c_p denotes the cost for the p^{th} pairing, x_p is the binary decision variable where $x_p = 1$ if pairing p is selected and 0 otherwise, and $a_{ip} = 1$ if flight i is covered by pairing p and 0 otherwise. In this formulation, the explicit enumeration of all the possible pairings is

required to construct the set of \mathcal{P} . Two distinct phases are therefore defined, namely the generation phase and the optimisation phase. The problem is NP-hard to solve. The cost c_p may be based on a fixed amount or a credit guarantee formulation as presented in §2.2.

2.3.1 Generation phase

Depending on the planning horizon (which may be a daily plan or span across multiple days), the generation phase may either involve the enumeration of feasible duties for daily planning problems or the enumeration of feasible pairings for longer pairing periods. Before pairings can be made, feasible duties that meet regulatory labour working hour restrictions and connections need to be generated. Although a distinction is made between generating duties and pairings, the formulation of the optimisation problem for different planning periods is similar, except for the definition of the feasible set, \mathcal{P} .

A feasible pairing will have a sequence of duties with minimum sit periods (up to a maximum). The sit periods may also fall over day breaks, overnight layovers and weekly cycles. In the pairing problem, pairings require that the first task of the first duty in a pairing start off at the crew base and the last flight task in the last pairing end off at the crew base starting point. However, for duty periods, the start and end point of the duty need not be at the same place, or even at the crew base. This makes the duty problem slightly more tractable.

In Figure 2.1, the various elements in a duty generation phase are illustrated. From the diagram, a duty period can constitute of more than one sequential flight tasks, however, if a flight task is long, a single duty period may contain just one flight task as illustrated in blue in Figure 2.1. There is a connection cost involved between two consecutive flights as explained in §2.1. As may be seen from the illustration, duty period 1 end point and duty period 2 start point are the same (JNB). This means that a possible feasible pairing may have duties 1 and 2 as sequential duties.

2.3.2 Optimisation phase

Once all feasible duties or pairings have been generated as per the problem constraints, the optimal duties or pairings can be found. This may be done simultaneously with the generation phase or after the fact. This is the phase that addresses a particular objective function, i.e. minimise the cost of the crews, which may be some linear or non-linear function. Alternatively, the objective may be to maximise the capacity utilisation of crew during a duty period, by minimising the idle time within a duty. Different solution approaches have been followed in literature to find optimal duties or pairings, as described in the following section.

2.3.3 Variations of CPP formulation

The CPP has been very well documented in literature especially in the fields of operations research and logistics. The complex problem structure, large planning requirements and constraints have led to the development of many mathematical optimisation techniques in this field. The CPP can become quite a large problem depending on the number of flights that need to be covered and the number of work rules; various approaches have been applied to solve or improve an initial schedule. Given the two phases to solving CPP, there are various ways in which good or optimal solutions may be obtained.

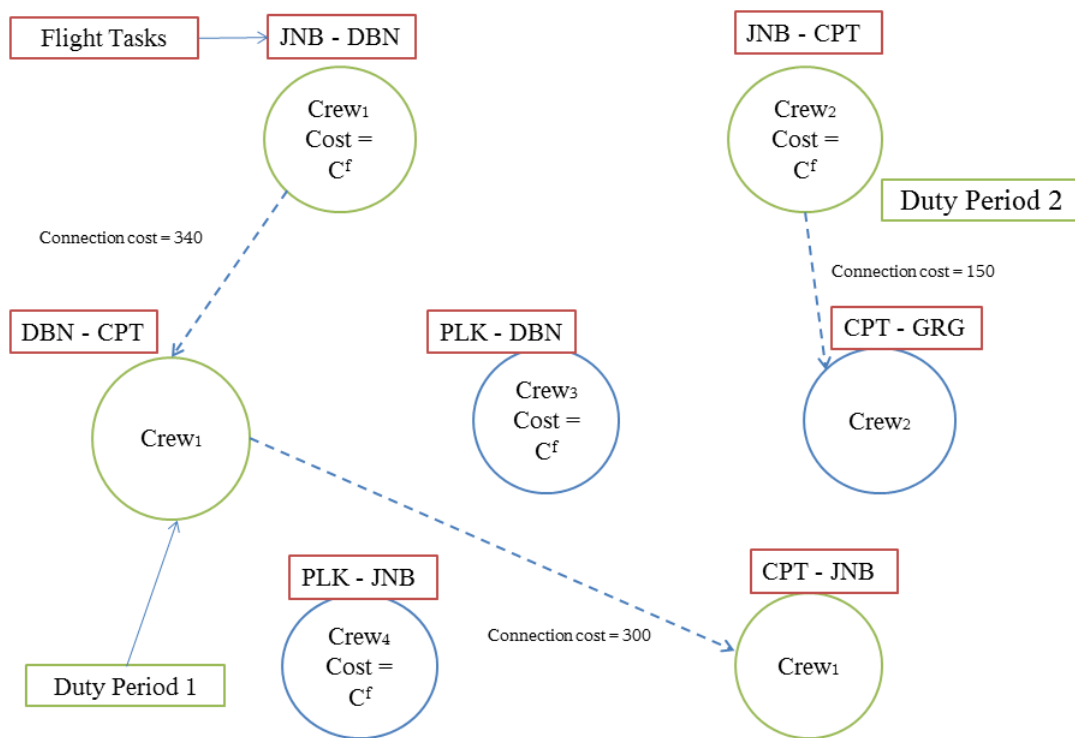


FIGURE 2.1: Duty period example for a small network of flight tasks.

There are many possible approaches to solving the CPP depending on the chosen formulation. In §2.3 the set partitioning problem formulation, typically used to formulate CPP, was presented in Equation 2.1.

Most CPP problems have been formulated either as [44]:

- set partitioning;
- packing or;
- set covering problems.

Consider problem formulation of (2.1) which is a set partitioning problem; if the equality constraints become \geq inequality constraints it becomes a set covering problem in the current 0-1 integer format. If the inequality is \leq , then the problem is a set packing problem. Hofman and Padberg [44] present a survey of algorithms that have been used to solve these types of problem formulations for mass transit and network problems. Some industry problems with a similar structure to the CPP include:

- Crew and Maintenance Scheduling for railway and airline problems;
- Delivery and routing problems;
- Switching Theory;
- Political district assignment problem.

The criteria for the format of the problem is driven by allocating a single or multiple users / customers to a resource. Solution approaches include relaxations of the 0-1 LP, heuristic approaches such as genetic algorithms, simulated annealing and neural network methods. Exact approaches have also worked well to solve all three types of the problem, where the use of Lagrangean relaxations and sub-gradient optimisation, surrogate relaxation [49] or branch and cut methods being particularly popular in literature as will be discussed in the sections below.

Caprara and Toth [18] also conduct a survey specifically for algorithms that have been used on set covering problems. In particular they discuss exact and heuristic solution approaches that had been used predominantly on the test data sets provided by the Beasley OR data library [58] which is also used for the data inputs in this study. According to the survey, branch and bound algorithms that use LP relaxations to solve for lower bounds are considered most effective when solving set covering problems.

2.4 Solution approaches

The next section provides an overview of some of the approaches used to solve the CPP in literature, including; exact, heuristic and stochastic approaches. The approach used is typically driven by the size of the problem and the quality of solutions required.

2.4.1 Exact approaches

Exact approaches to solve the CPP include mixed linear and integer programming methods. Hoffman and Padberg [43] describe a branch-and-cut algorithm which can solve large set partitioning problem formulations such as the CPP. The algorithm presented generates cutting planes based on the structure of the polytopes. The cuts are then used in conjunction with a tree search and other heuristic techniques. The branch and cut solver developed was used on real life large scale crew scheduling problems. All the pure set-partitioning problems that had less than 5000 variables were solved within 4 seconds with optimal solutions. The solutions provided were schedules that crew members were happy with and schedules that provided cost savings to airline companies.

Tree search techniques are easier to conceptualize when thinking about a network of flights to be covered. Beasley and Cao [12] present a zero-one set covering formulation of a crew scheduling problem. Lagrangian relaxation and sub-gradient optimisation are used in order to attain a lower bound on the solution. They then use a tree search technique which incorporates the lower bound to get to an optimal schedule. The branching technique described was able to solve relatively large problems. The formulation presented was restricted to only considering maximum duty time constraints. Other crew bidding or regulatory constraints are not included. Furthermore, the above formulation would require that the schedule be fixed for a time without any room for unexpected events that could change the schedule.

The air crew scheduling problem forms part of a larger set of problems that need to be optimised in the airline and airport management industry. Each of the various planning stages are interdependent and as such in some instances an integrated airline planning formulation may be used. Mercier *et al.* [53] consider an integrated planning approach to simultaneously solve the aircraft routing and the air crew scheduling problem. They use a combination of Benders decomposition and column generation algorithm to solve for the minimum cost route and pairing schedule. In the Benders algorithm the crew scheduling

problem is the master problem and the air craft routing is represented by the sub-problems. The algorithm and formulation yielded better results in comparison, based on identical input data to results on an integrated problem formulation presented by Cordeau *et al.* [28] where the air-craft routing problem is the master and the crew scheduling the sub-problem. Both these cases look only at solving a deterministic formulation and finding a suitable way to decompose the problem.

Rasmussen *et al.* [75] was one of the first papers in literature that focussed on the 'subsequence' generation problem, or as described in this study, the generation phase. They used a dual vector form to solve the relaxed LP by generating negative reduced cost columns and a shortest path solver to generate subsequence in the set of possible/candidate subsequence set.

2.4.2 Heuristic and meta-heuristic approaches

As airline companies grow and cover more routes possibly both international and domestic, the problem sizes become larger and more difficult to solve, as such heuristic approaches have been developed.

Air crew scheduling entails many considerations that can lead to highly constrained optimisation problems; as such they can vary from NP-complete to combinatorial. Heuristic approaches are often used to solve computationally difficult problems when classical methods are too slow or when exact solution approaches are not attainable. The objective with heuristics is to find feasible solutions in the least number of steps and at the least cost. This is determined by the heuristic function value $h(i)$ which is evaluated at each iteration $i \in N$.

A heuristic search technique uses domain specific information about the problem that will direct the search into spaces that can yield feasible solutions. A good heuristic should be able to find decent (not necessarily optimal) solutions for NP-complete problems in efficient time with limited computation capacity. Given the potential scale of air crew scheduling problems, global optimisation techniques can be beneficial when trying to solve for larger flight network problems. Heuristic algorithms can be divided into the following types of search techniques [51]:

- Decomposition - decompose the problem into smaller more tractable components or sub-problems.
- Induction - generalise the problem and solve for smaller instance from which properties of the larger problem may be deduced.
- Reduction - reduce the search space however, there is the possibility of excluding global optimum.
- Construction - step-by-step method to finding a solution that works best for deterministic problems.
- Local Search - start off with a feasible initial solution and progressively improve with each iteration of algorithm. The initial solution may be randomly chosen or estimated from prior information.

Anbil *et al.* [6] use a local improvement heuristic solution approach, as described in approach 4 mentioned above, which starts off by taking a subset of pairings from a feasible pre-determined schedule (manually constructed in most cases), then tries to solve for a better solution for the flights covered by the chosen subset.

Cabral *et.al* [78] present a construction heuristic approach to solving the air crew scheduling problem. A bin packing approach is used to ensure all flight rotations are covered and is typically used for deterministic problems as it is constructive in nature. In this instance crew are considered as the bins (with variable capacity), with the objective being to minimize number of crew (bins) used without having uncovered rotations. The heuristic starts by assigning crew with more schedule impediments first, then the more flexible crew members are assigned. The objective function value is evaluated for each schedule determined with critical tasks being determined. In order to ensure a balanced assignment of tasks among the crew the standard deviation of minutes among all crew is determined and crew are re-assigned until the standard deviation of time and kilometres limit has been reached. This approach yielded good schedules in a good time (90 - 600 minutes) for the assignment of 350 crew to 2000 pairings.

Ozdemir [60] presents a heuristic branch and bound approach which are reduction type heuristic, to solve the pairing problem. He illustrates standard branching principles and highlights the challenges they pose for the crew pairing problem mostly due to slow convergence especially when the possibilities of pairings increases. To address this concern a branching technique proposed by Ryan and Foster [66] is used to get the fractional solution to the set partitioning problem. Based on this rule they developed the Branch and follow-on method to be combined with previously used models in a hybrid manner. The use of Branch and follow-on method was not as fast as the Carmen algorithm, but it yielded much better solution results.

For extremely large problems, meta-heuristic approaches may be considered, especially where there may be incomplete information as in the case of stochastic/dynamic problems. Popular meta-heuristic algorithms include tabu search, simulated annealing, hill climbing, variable neighbourhood search and evolutionary algorithms [40]. A simulated annealing approach is, for example, presented by Thomas Weinert and Mark Proksch [36], which yielded successful results for solving short to medium haul real world CPP's.

There is extensive literature on evolutionary algorithms and their applications in network type problems. Evolutionary algorithms were developed in the 1960's when computer scientists like Von Neumann and Weiner were exploring artificial intelligence techniques that would improve computer programs and the efficiency thereof [54]. In particular, biological systems and natural strategies were of interest to computer scientists and engineers with early applications in machine learning and neural networks. John Holland [45] invented Genetic Algorithms (GA) during his exploration of adaptive strategies in nature, that can be formally described and used to improve search algorithms and computer programs in general. There are many variations on the implementation of GA's, however, most implementations have the following key concepts:

- Chromosomes - initial solution structures with underlying parameters and attributes
- Fitness selection criteria - based on feasibility and optimality goals
- Crossover function - the pairing of parent chromosomes to create new offspring which will have attributes from both parents
- Random mutation - when new offspring have attributes that are not mapped back to parent attributes

GA would typically perform crossovers over multiple populations or iterations of search until a target or stopping criteria has been met. GA's are commonly used in the field of

optimisation where the goal is to find a set of parameters that will minimise or maximise an objective function. This meta-heuristic method provides an adaptive technique that continuously evaluates and assesses the solution spaces and prior solutions. This flexibility lends the GA methods to solving problems with dynamic features. As such the GA has regularly been chosen as the meta-heuristic approach of choice when solving crew pairing problem.

Ozkol and Zeren [83] solve the CPP using a GA approach with a unique perturbation operator which intentionally makes a chromosome infeasible in order to enhance the possible solution quality. If the search progresses successfully then the chromosome perturbation is maintained, if not the perturbations made to the feasible chromosome are reverted. The experiment used test data from the Turkish Airlines Airbus flight schedule. The proposed algorithm showed improved convergence rates and a reduction in deadhead flights (which subsequently reduces costs). In addition, the combination of larger populations generated in the GA and a perturbation operator that had a bigger probability ratio, resulted in better performance of the algorithm.

Chen *et al.*, [26] solve a short-haul CPP using a multi-objective GA for a Taiwanese airline case study. The CPP may be solved in two stages, where the first is to get a set of feasible pairings and the second is to get optimal pairings that still meet the feasibility criteria. Chen *et al.* combines these two steps in a simultaneous implementation.

Ozdemir and Mohan [59] solve the CPP using a GA on a graph representation of the problem. They formulate a unique graph representation where the nodes represent the flights and the edges represent dependency constraints on other flights, and each path would then represent a feasible sequence of flights to be covered by crew. This is different from the usual graphic representation of the problem in previous literature, where the nodes represent cities or destinations and the edges represent the flights. A GA is then applied to the graph and the results are compared to the greedy algorithm heuristic methodology. The GA technique they used yielded much better results in comparison across various aspects including workload distribution and fewer number of crew required to cover the flights. The technique seemed to be very good at dealing with highly constrained crew scheduling problems and could possibly perform well in highly constrained transport problems in other industries.

Ozdemir [60] studied the subsequence or duty generation problem as a crucial step in obtaining good solutions using a depth first search method before obtaining optimal pairings using the Sprint method, Carmen algorithm and variations of branch and follow on methods. The computing times for the various methodology are also considered, with the largest network of 222 flights taking up to 4 hours in CPU time to get a solution output.

The generation of duties or pairings is usually taken as an implicit step in the CPP, before solving for optimal pairings. Kornilakis and Stamatopoulos [41] describe the process of duty generation by implementing a depth first search to create optimal duties, then using a GA technique to solve for optimal pairings. Test results were generated for actual flight schedule data of Olympic Airways in Greece that contains 2,100 flight legs. The algorithm led to low cost pairing solutions and illustrated improvement on previous solutions using the same data set. The algorithm produced similar quality results when implemented on test data used by Beasley and Chu [11].

Bin packing may be used to solve a whole class of problems such as storage container filling [37], truck loading [4], computer memory and online storage problems [13] and scheduling

problems [72]. CPP would be classified as a scheduling time problem. The number of tasks can be classified as components with specific weights, that need to be completed or packed into bins with a given capacity. In the CPP, the flight tasks and the duration thereof can be considered as the *weights* in the problem and the duties that have a maximum regulatory duration would be the *bins* into which the flight tasks must be fitted. The objective would be to fit the flight tasks into the least number of duties possible, given a set of flight operation and regulatory constraints. Heuristic improvements for maximising the utilisation of a bin have rules on how items are packed into a bin with a given capacity. In bin-packing there are various approaches that can provide packing solutions [3], of which the *best fit* approach is among the best. However, in the CPP, flight task durations can be particularly long in proportion to the duty duration. As such the approach that works best to solve CPP within the bin packing framework is the *first fit* approach.

2.4.3 Stochastic approaches

CPP can be quite difficult to solve especially when running through various scenarios that may affect an airline's profitability. As such, stochastic solution approaches may be used to solve the deterministic problem.

In CPP the occurrence of a delay in a flight network is effectively a random event to some degree that can be forecast using historical data, and the resultant delay may have a financial impact that is uncertain. Monte Carlo simulation, also known as probability sampling, is a forecasting technique that uses the concept of random sampling to generate a series of simulated problem outputs from which an expected value can be determined [65]. Monte Carlo simulation and sampling techniques became very popular in nuclear physics applications in the mid 19th century, with physicists like Von Neumann and Ulam performing the first experiments on the early computers [33].

Monte Carlo can be used to solve various deterministic and stochastic problems across many applications including; Financial modelling, Nuclear Engineering, Biophysics and Cancer Studies [22]. A lot of these problems can be solved using random sampling because their nature is probabilistic and statistical.

The use of Monte Carlo to solve the deterministic equivalent of the CPP, is a stochastic approach to a deterministic problem. This approach is considered in one of the algorithms developed to solve the deterministic CPP in this study.

2.5 Considering uncertainty in the CPP

There are various ways to solving mathematical problems that have uncertainty in the parameters. The four common approaches that have been well studied in literature are, robust optimisation, stochastic programming (SP), stochastic approximation (SA), Monte Carlo simulation and scenario/parametric Analysis. The formulation of the CPP will drive the kind of approach that is best suited to solve the problem. A number of SP formulations have been presented in literature to solve the CPP under uncertainty.

2.5.1 Expected value linear programming

The most simple approach to solving any linear programming problem that has uncertainty in any of the formulation components, technology matrix, right hand side or the cost, is to estimate the expected value of those random elements and solve the deterministic equivalent of the problem. Albert Madansky [50] presents a discussion of the one-stage stochastic programming problem described above and links it to the traditional two stage stochastic programming problem, by the use of pessimistic expected values (worst possible outcomes) in the one-stage problem. The expected value problem of what-if analysis provide opportunities for decision makers to get solutions that would be suitable given the information available at a given point in time. However these approaches do not always provide the most optimal solutions as discussed in [74]. The next section provides an overview of SP and a taxonomy that includes what-if and expected value formulations as the basics of the traditional two-stage programming and chance constrained problems. In the CCP with uncertainty some of the approaches from the most simple expected linear programming to the two stage stochastic programming problem with recourse, have been considered in literature.

2.5.2 Stochastic programming

SP is a framework for solving mathematical programming problems that may incorporate uncertainty in the decision variables or parameters. Dantzig [32] first introduced SP in his paper on linear programming under uncertainty, in which he specifically addressed a class of problems where decision variables need to be determined at multiple stages. Subsequent to this paper, many problem classes were presented and formulated which incorporate different elements of uncertainty. SP allows for uncertainty to be quantified and formally included in the problem statement as a variable to be estimated in order to reach a near optimal solution [67]. The uncertainty may present itself in the objective function as noise or in the set of constraints which may change depending on the expectation of the unknown parameters. Uncertainty can also be present in unrealised decision variables that are time dependent. Given the various types of SP formulations [73, 10, 23, 24] the appropriate formulation may be used to describe different problems depending on the occurrence, distribution and types of random events or variables in the problem description. Powell and Topaloglu [62] describe three scenarios of uncertainty. These are:

- Information not currently known, but will transpire in the future;
- Inaccessible information (perhaps information only known to some people or expensive to acquire);
- Incomplete information, where data for the variable is not measurable or cannot be modelled by some kind of distribution.

Typically in a SP problem a decision needs to be made for a particular point in time, without any information about the future, other than the historical data and experience that is provided. In some instances a decision needs to be taken at multiple points in the future. Single or multi-stage SP is one of the approaches to model such problems. A taxonomy of SP problems and solution approaches is presented by Valente *et al.* [79] and illustrated in Figure 2.2. As indicated in Figure 2.2, the framework for modelling SP problems is split into three distinct approaches that have specific formulations: recourse problems, chance constrained problems and distribution problems. SP with recourse is a formulation that enables reaction to stochasticity, where there are a set of feasible recourse decision variables. Roger Wets provided one of the earlier and widely used algorithms to solve SP's with simple recourse

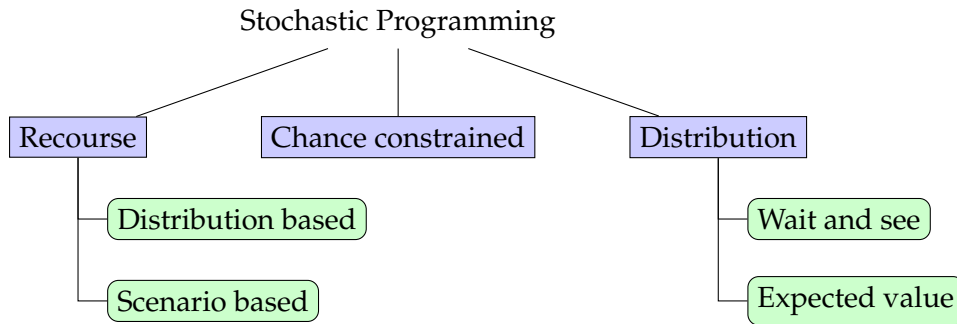


FIGURE 2.2: A taxonomy of Stochastic Programming solution methodology [79].

by reducing the problem to its deterministic equivalent [42, 81].

In a series of papers published by Charnes and Cooper, industrial problems with uncertainty in the constraints are analysed [25]. Optimal sequential decision rules are defined prior to splitting the problem (linear or non linear) into two parts, namely: determination of an optimal probability distribution of the uncertain constraint term and the subsequent approximation of those distributions. This leads to the exploration of chance constrained programming; where violated constraints can be associated with a cost or compensated for by a decision taken in the second stage. Charnes and Cooper illustrate their treatment of chance constraints and normal deviates through a shipping chartering example where the demand is subject to independent normally distributed random deviations. They later formulated a deterministic equivalent to optimise the chance constraint problem formulations and certainty equivalents which were presented as approaches to solving SP problems.

Beale [10] first presented the distribution approach to solving SP's with a simple linear program as given by (2.2), where the value of b is not exact, but can be estimated.

$$\begin{aligned} \min \quad & f(x) \\ \text{subject to: } & Ax \leq b \end{aligned} \tag{2.2}$$

The constant term on the right hand side, b may be modelled by considering probability distributions that would best describe the expected values. Beale defines a quadratic programming algorithm to solve the resultant problem.

2.5.3 Stochastic CPP formulation and solution approaches

The formulation of the stochastic CPP (SCPP) is dependent on the framework of SP in which a particular problem is solved. Stochasticity introduced in a problem, may be for example, in the form of some delay cost, additional crew requirement, changes in deterministic schedule or additional fleet requirement.

Given the many causes of delays as discussed in §1.2, various expansions of the delay in the CPP have been considered in literature. The first is a stochastic integer programming formulation that is reviewed by Birge and Yen [14], where the random event is the possibility of a predecessor flight between two different pairings causing a delay for crew who need to switch. They propose a delay switching non-linear constraint which they solve using a delay branching algorithm that presents decent results for small CPP's. Tam *et al.* [77]

compare the SP approach used by Yen and Birge with a bi-objective integer program optimisation by [34] and provide an improvement on the stochastic formulation by Yen, where their solution methodology was tested using actual flight data from Air New Zealand. One of the observations made with the flight delay data was that if a delay occurred earlier on in the day, given scheduled flights, the delay would likely cause more delays throughout the course of the day than if it occurred later.

Schaefer *et.al* [56] also present a CPP formulation under uncertainty. A penalty method that penalises attributes of a pairing that lead to bad pairings is implemented. Algorithms to solve the optimal crew pairing with k attributes, such as crew changes, rest period and duty flying time, for the objective function

$$\min_{\alpha, \gamma \in Y} s(\alpha, \gamma) \quad (2.3)$$

is proposed where α as the maximum penalty factor for attribute $i \in k$ and γ is the slope a penalty function f^i . The main concern that is raised in Schaefer's paper, even with their own assumptions, is that when there are random events the recovery or push back strategy, as presented in most stochastic formulations, is not applicable in reality. The push back strategy entails that decision makers adjust actual departure and arrival times in order to allow more time for delaying factors to be resolved. In reality this may not always be possible and could result in even more costly impacts on original schedules. A recovery policy is a policy adopted by flight carriers to reduce the impact of delays on original flight schedules. Different types of recovery policies are widely used in practice when there have been delays earlier in the schedule. Some common recovery strategies include:

- Reduce flight time by flying at higher altitudes and higher speeds when safe to do so;
- Take off a few minutes earlier than scheduled flight time, if possible (all passengers board earlier, quicker plane clean up and runway availability).

Caprara *et al.* [19] present a recoverable robust framework (combination of robust optimisation and stochastic programming) to solve event based network problems where time specific delays are propagated through a network. The nodes of the network represent events and the arcs between the nodes illustrate how a delay is propagated from one event to another (precedence relationship). Given a nominal event based problem formulation (NOP), the objective function is to minimise the total propagated delay time over all scenarios in order to obtain the delay robust optimisation problem (DROP) equivalent. Buffers are then created by generating constraints that link the variables of the NOP to the DROP. The approach is applied to solve the deterministic train platforming problem (TTP) for real life data of an Italian railway network. An integer linear programming (ILP) formulation of the TTP is solved using a branch and bound algorithm with a refining heuristic technique that is applied on the continuous relaxation of the problem. The algorithm is designed to find good solutions that are feasible as quickly as possible. Good results are obtained for the problem when the refining heuristic is applied. The output for the number of trains used in the network remains the same for both the optimal solution of the nominal problem and its robust delay equivalent problem. The delay propagation is reduced to ranges of 0 to 35% at various stations in the train network.

Delayed column generation has proven effective in solving the SCPP and has been widely used in industry and in literature [27]. The basic approach for delayed column generation includes:

- Step 1: Generate initial pairing solution (set of columns);

- Step 2: Solve the linear program relaxation of the integer program;
- Step 3: Generate new columns using the dual form of the relaxed LP such that the current solution is improved;
- Step 4: Repeat Step 3, until all columns have positive reduced costs.

Aoun and El Afia [7] present the SCPP as multi-agent Markov decision processes, where stochastic disturbances are considered in the CPP in order to minimise the real operating costs for the pairings. Some of the disturbances included in the stochastic problem include delays due to weather, technical breakdowns and unscheduled maintenance.

Robust methodology are particularly popular in dealing with changes to existing schedules. Shebalov and Klabjan [68] also used a robust method to minimise the cost of replacing crew on a pre-determined plan. Similar to the solution methodology used in solving stochastic programs, a delayed column generation and lagrangian relaxation technique was used. Various perturbations to the existing crew plan were simulated and their solution methodology was implemented as a recovery method. The cost structure considered in determining the effectiveness of the recovery schedule includes dead-heading, uncovered crew and reserved crew, which are all cost elements that should be minimised in a good crew pairing solution.

Muter *et al.* [55] present a robust methodology to solve the CPP which uses an existing crew pairing plan for a Turkish airline company that requires additional flights to be included at short notice. The solution methodology is based on row and column generation and is only implemented when required. The objective is to minimise the impact on the original pairing schedule. The robust approach is considered a deterministic approach to tackling a stochastic problem.

Silverwood [69] uses Monte Carlo simulation to solve the crew scheduling problem (CSP) where stochastic delays to plane departure times are simulated for a given schedule. The SP formulation of the CSP in Silverwood's study is the expected value formulation, where the expected value is used to replace the stochastic distribution of the delay time plus the flight duration variables. In particular, these variables were considered in the stochastic fleet assignment with time windows and route selection model. A combination of scenario analysis and chance constrained programming was used to ensure that the model constraints were maintained when extreme values from the variable distribution were sampled. The study illustrates a varying range of results that were influenced by various parameters, such as the schedule density, number of fleet available and the average delay time. The inclusion of stochastic delays in the model was more effective for dense flight schedules (more flights). Schedules that had a smaller number of flights were less affected by the stochastic delay settings.

Another technique used to deal with delays, is the employment of reserve staff that are specifically called when there are delays or events affect a pre-determined crew pairing schedule. Quantas Airways [38] implemented a reserve crew level as part of their annual planning, with the objective of minimising the total crew cost in the event of an aircraft delay. Conditional probabilities and future schedules were used to determine cost implications and the likelihood of an aircraft delay.

Cacchiani *et al.* [17] provide an overview of various recovery models and algorithms that can be applied to real time railway scheduling. Rail scheduling typically follows cyclical or non-periodic time tables and similar to the CPP, stock (trains) need to be rolled and staffed

by appropriately skilled individuals. The following types of recovery plans for disturbances or delays may be considered as the objective function of a problem:

- Minimize the delays of passengers and freight;
- If there are cancellations, minimise any additional cancellations in the system due to limited resources;
- Minimise total additional carts that need to accommodate overflow from passengers or freight.

The above objectives can be considered in the train time table rescheduling problem, which is further split into re-routing or re-timing class of problems. Various solution and modelling approaches include modelling the problem using graph networks, mixed integer problem formulations and various heuristic techniques that may be applied to CPP.

2.6 Duty generation and crew pairing in industry

Arabeyre *et.al* [8] compiled a survey of the various methods different airline companies have implemented to optimise their crew pairing. The survey unpacks the various stages in the CPP starting with the generation problem, analysis of different cost formulations, the reduction problem and the optimisation thereof. Typically, a planning schedule would be completed for a set number of days and these schedules may be repeated at the end of each period.

In industry, airline companies employ large scale systems to generate feasible rule based crew schedules and assignments. One of the most widely used systems is the Carmen System (also known as Jeppesen), which provides resource optimisation tools for airline companies and other large scale transport companies that benefit from the systems scheduling capabilities [20]. Carmen Systems was purchased by Boeing in 2006 in order to extend the fleet product offering to include a service offering for clients that supports their efficient management strategies in the aviation and transport industry [15]. The systems are used by some of the larger airline companies including British Airways, Green Cargo, Air France, Scandanavian Airlines and KLM [46].

The algorithm used in the Carmen System crew pairing optimiser is a large scale 0-1 integer program presented by Wedelin [80]. The technique starts off with a dual search algorithm on the lagrangian relaxed form of the ILP to find the 0-1 solution. An approximation technique is then used that keeps the cost vector c to a degree determined by a parameter k that determines the degree of 'greediness' or 'optimality' in order to generate unique 0-1 solutions and convergence. The algorithm produces good quality solutions for very large air crew pairing problem sizes.

A Canadian based company, AD-OPT solutions - Division of Kronos Incorporated, [5] also offers system solution support to the airline industry specifically for airline crew pairing and rostering. The optimisation tool used, GENCOL, is based on a column generation approach that generates feasible and optimal pairings and crew schedules. The system also provides 'what-if' scenario analysis, forecasting and cost benefit analytical tools through a user interface that can provide quick solutions. Many of the airline companies that use the AD-OPT

solutions are American based, such as WestJet, Air Canada and Atlas airlines. However, AD-OPT also services larger international clients including, Emirates, EasyJet, Etihad, Qantas and even our local airline, South African Airways.

2.7 Chapter Summary

In this chapter, the CPP was presented. Terminology, definitions and regulatory requirements were described before reviewing the standard mathematical formulation for the CPP in and some of the constraints that may be applicable. Given the deterministic mathematical formulation different solution approaches that have been applied previously in literature were reviewed, namely: exact, heuristic and meta-heuristic, as well as stochastic approaches to the deterministic problem. In addition, a review of stochastic formulations and solution approaches of the CPP under uncertainty was given in order to understand various solution methods that could be applied to deal with possible random delays that are prevalent in the airline industry and in particular those delays that might affect the air crew planning schedule.

Chapter 3

Methodology and Results - Deterministic Problem

This chapter presents a solution approach to solving the CPP for the deterministic case. The methodology, assumptions and problem parameters are described in order to define the scope for the problem. A mathematical formulation is presented based on the standard set partitioning problem formulation as cited in the literature in §2.3. Three main algorithms are presented. First, an exact solution approach is followed to generate the entire feasible region of duties, and then find an optimal duty set from this region. The second and third algorithms are heuristic techniques implemented in a manner that simultaneously generates feasible duties and select a duty set from this region that covers all flight tasks.

3.1 Informal problem description

The deterministic CPP requires that crew are allocated to a set of flight tasks within a given network over a stipulated time horizon. The time horizon for which the CPP is solved in this study is a 1 day time horizon, where the number of flight tasks to be allocated in the most efficient manner ranges from a network size of 50 to 500. These network problems are taken from the OR library and are the same data set used in Beasley's paper [12] where he solves the CPP using a tree search heuristic as discussed in §2.4.1. The network problems were randomly generated with varying sizes starting from 50 flight tasks to 500 (increments of 50). In this chapter the methodology approach will be described using an example network, $\mathcal{F} = 5$, in order to illustrate the solution approaches and output.

3.2 Modelling approach and assumptions

Three solution approaches are proposed in this chapter. First, an exact approach is followed whereby all possible flight connections and duties are compared and a minimum cost solution is found. This is proposed as a benchmark to the two heuristic approaches that follow in the second part of the chapter. The second approach is based on a greedy algorithm, where a duty is created by adding the least cost connection to a path of flight tasks until the given duty has reached its maximum duration. The third approach is based on a Monte Carlo simulation, where the expected cost is determined over multiple simulations of flight duties that have been generated.

As mentioned in Chapter 2, two phases are defined in the CPP, namely the generation phase, and the optimisation phase. In an exact approach, a clear distinction is usually made between the phases, whereas in heuristic approaches, the search technique does not always require that these two phases be solved independently. In Figure 3.1, an illustration of the feasible and optimal regions generated by the different approaches are given. In the

exact approach, all feasible duty periods are generated using enumeration. The heuristic approaches do not generate the entire feasible region, and may therefore not contain the optimal set of duty periods. Since the scope is limited to the daily problem, duty period rather than pairings are generated and matched to crew, however, reference is still made to the CPP since crew are paired with duties in the optimisation similar to how crew would be paired to pairings.

Furthermore, a graph theoretic approach is followed in the algorithmic implementation, where each node in the flight network represents a flight task, and each edge represents a feasible connection between flight tasks. In Figure 3.2, a visualisation of a flight network containing 50 flight tasks (referred to as CPP-50) is given. The flight network, as illustrated in Figure 3.2, may be translated to an adjacency matrix, where the flights that are adjacent to each other are denoted by a non-zero element in the adjacency matrix, and zero otherwise. Specifically for this study, a network representation of duty tasks is a weighted digraph (see appendix A for a short discussion on graph theory). The weights represent the cost of the connection between two sequential tasks. The directions represent the fact that certain tasks cannot happen before prior tasks due to scheduled duty start times and duration.

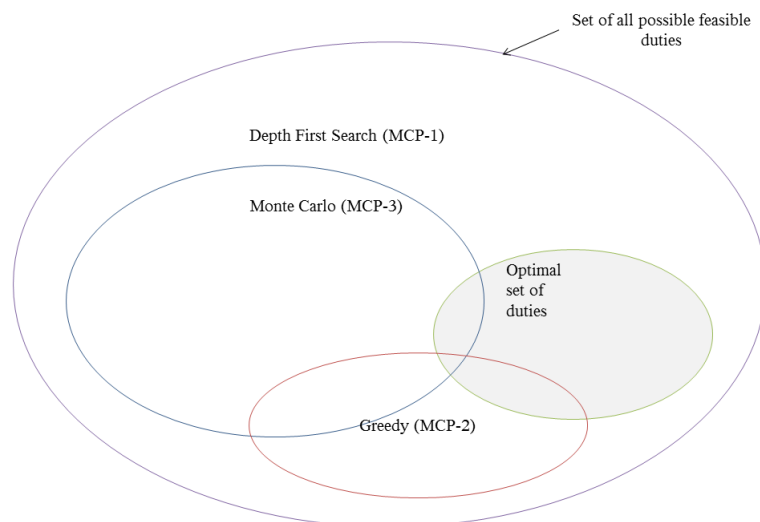


FIGURE 3.1: Feasible duty region as created by the proposed algorithms.

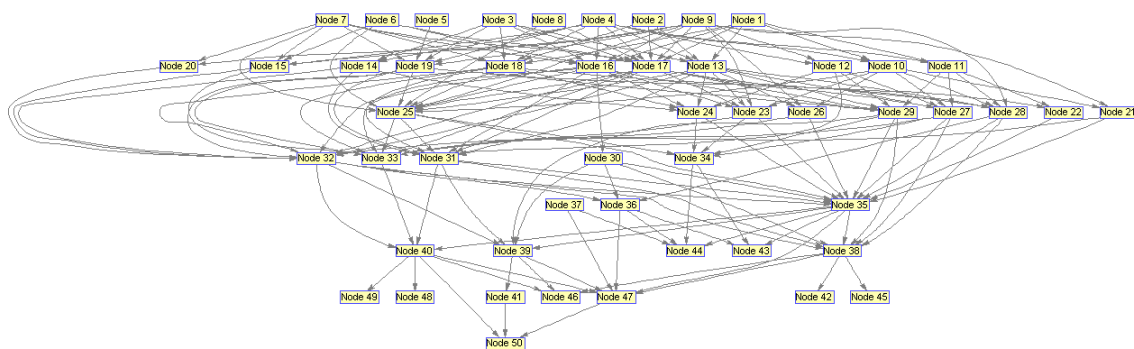


FIGURE 3.2: A bigraph of a CPP network of 50 flight tasks with their respective adjacency connections.

3.2.1 Problem assumptions

Whilst every attempt has been made to simulate the pairing problem in reality, there are assumptions that are made for the static model:

- Duty duration may not exceed 480 min (regulatory duty period).
- Since some tasks are quite long and not in the path of feasible connections, they may be assigned to single task duties.
- The start point of a flight task in a duty does not need to be the same as the end point of the last flight task.
- The crew are multi-skilled and are able to service any of the fleet type in the domestic network described.
- Crew does not have to end at base after duty. ¹
- The connection cost between flight tasks covers expenses such as *deadhead* expenses for crew members to return to base. In addition, those flights that end up at destinations that do not have regular flights to more commercial destination points may result in longer duration crew idle time.
- Sequential flights in a duty period cannot overlap.
- A crew can only perform one duty per day, i.e. one duty per crew.
- Fixed cost for crew salary regardless of how long crew is on duty, within the regulatory duty period.
- *Brief time* and *connection time* are included in the duration. These are standard procedures that are completed within a fixed allocated time regardless of the duration of the flight or size of the plane.

3.2.2 Modelling objective

The CPP requires a solution methodology that will provide a fast and cost effective duty generation and ultimately crew pairing for managerial decision making purposes. In this study, the objective is to minimise the total cost of crew required to service a network of flights by generating optimal or near-optimal duty periods, that can be assigned to limited number of crew during a one day time horizon schedule. In addition, a fixed cost structure is assumed for every crew employed to service the flight network, and a variable connection cost, which is dependent on the connecting flights that are serviced by the crew. For every flight task the crew have to take off from their base, there is a cost and possible time lag in getting the crew back to base or to a connecting flight that can still be serviced within the feasible duty period.

Consider the diagrams given in Figures 3.3 and 3.4. The first diagram illustrates a network of flight legs where crew have not been optimally allocated to service the tasks. It also illustrates an impractical and expensive model for ensuring that all flight tasks are covered by the same number of crew. This network is impractical because even if the flight task is 1 hour long, the crew must still get paid a minimum amount based on the contractual minimum or fixed salary compensation structure as described in §2.2. Table 3.1 gives a breakdown of the

¹This is a distinct assumption for this study as it deviates slightly from the definition of crew pairing in many studies in literature, where crew pairings start and end at a base location [52].

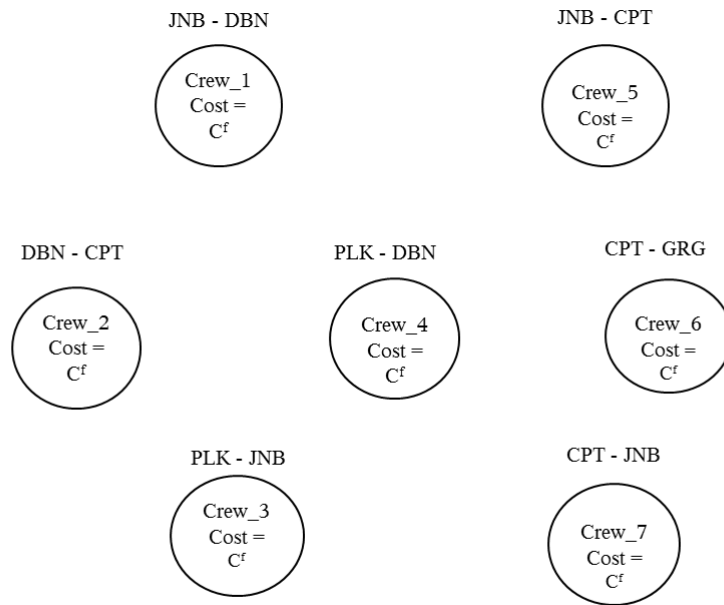


FIGURE 3.3: *Non-optimal crew allocation where every flight task is assigned to a duty managed by a crew.*

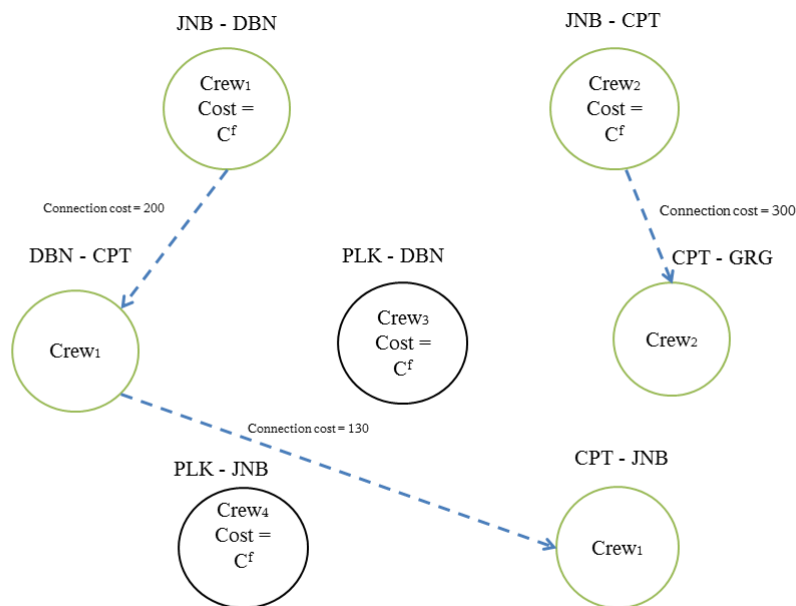


FIGURE 3.4: *Optimal crew allocation where sequential flight tasks are grouped together in a duty. Less crew are required in this crew allocation.*

cost structure difference between the optimal and non-optimal crew allocations based on the illustrations in Figure 3.3 and 3.4. The Fixed cost for the crew is denoted as C^f , where C^f is a constant. Generally the connection cost are much smaller than the fixed cost per crew.

Cost element	Optimal allocation	Non-optimal allocation
Flight tasks	7	7
Duties (D)	4	7
Crew (K)	4	7
Cost per crew	C^f	C^f
Connections Cost	$D_1 = (200 + 130)$ $D_2 = 300$	0
Total Allocation Cost	$630 + 4C^f$	$7C^f$

TABLE 3.1: *Flight task duty assignment costing comparisons of networks described in Figures 3.3 and Figure 3.4.*

3.3 Data description

The test data for algorithmic comparison was taken from the O.R library of test data for Operations Research problems [58]. The data files were originally used by Beasley and Cao [12] using a tree search algorithm to solve the CPP. Beasley and Cao's approach to the problem solution is two phased, with the first phase a column generation to generate and initial assignment of K crew to N tasks, the second phase is an optimisation heuristic tree search where branch and bound cuts are implemented until the cost function is optimised. The benchmark algorithm in this study (MCP-3) performs both the generation of duties (assignment to crew) simultaneously whilst optimising. The data sets include problem sizes (one each) of $\mathcal{F} = 50, 150, 200$ up to 500 flight tasks that need to be covered by available resources. The data in each network size problem .csv file includes:

- Network adjacency matrix detailing the feasible flight task connections with a non-zero value, with the non-zero value referring to the cost of the connection;
- Flight task start and end times (in minutes); and
- Maximum duty duration per network size, which was 480 min for all network sizes.

The comparison with Beasley's tree search algorithm provides a benchmark for the deterministic case of the CPP. In addition, the data sets are constructed to allow one-day time horizons for planning, where maximum duty durations are set to 8 hours (for the standard working day) and the start times of the flight tasks occur within a one-day time horizon without overlap. The data has the potential to be extended for multiple day time horizons for weekly or even monthly schedules, but this is outside the scope of this study.

3.4 Mathematical formulation

The objective function is to minimise the total cost of crew pairings such that all flight tasks are covered exactly once and form part of a path whose duration is less than 480 min (8 hours being the regulatory limit for number of consecutive hours a crew member may work at a time). As mentioned in Chapter 2 the problem may be formulated as a set partitioning problem. We define the set of all feasible duties $k \in \mathcal{K}$, and the set of all flight tasks $i \in \mathcal{F}$. Let $a_{ik} = 1$ if flight i is covered by duty k and 0 otherwise. In addition, C_k^v is the variable cost for the k^{th} duty and C^f the fixed cost associated with each crew. The objective function to minimise the sum of variable connection costs and fixed cost is given by

$$\text{minimize } \sum_{k \in \mathcal{K}} C_k^v x_k + \sum_{k \in \mathcal{K}} C^f x_k \quad (3.1)$$

$$s.t \quad \sum_{k \in \mathcal{K}} a_{ik} x_k = 1, \quad \forall i \in \mathcal{F} \quad (3.2)$$

$$x_k \in \{0, 1\} \quad \forall k \in \mathcal{K} \quad (3.3)$$

to minimize the sum of connection costs, C_k^v for each crew pairing k , as well as the fixed cost, C_k^v , associated with the total number of selected crew.

3.4.1 Generation of feasible duties

In order to solve the CPP a set of feasible duties K is generated as follows: Let T_j be the duration of flight task j and S_j be the start time of flight task j . Note that the use of subscripts $i, j \in \mathcal{F}$ refer to adjacent flight tasks and are used interchangeably in the algorithm and formula statement, where i would typically refer to the current flight task under consideration whose potential adjacent flight task is j . The below rules need to be adhered to in the algorithms in order to generate regulatory feasible solutions.

1. The start time S_j of the next flight task j should be greater than the start time plus the duration of the previous flight task i .
2. All the duties k must have unique starting points. This means that no two duties can start at the same flight task or have the same flight task in both duties.
3. No duty may exceed its maximum capacity.

For the search algorithm, once a task has been completed, it is assigned to a ‘taboo’ or ‘completed’ task list and is removed from the list of possible connections for the remaining search. Once a duty has reached its maximum capacity a new duty search starts. The search ends once all the nodes/tasks have been allocated to duties (covered).

The start time constraint eliminates infeasible connections. From the data provided in the test files per network size, the duration of each flight task was calculated with the given start and end times. This was required to determine which flight tasks could be completed sequentially, such that the start time of the next flight was later than the start time plus duration of the previous task.

3.4.2 A theoretical framework to generate the entire feasible region

From Figure 3.1, it is evident that various algorithms will produce different parts of the total feasible region in the solution space for the optimal crew pairing. Given the finite set of combinations of paths and very tight regulatory and allocation constraints, there is a way of traversing all possible *feasible* paths and comparing them to obtain the optimal solution. The problem is enumerative on a discrete set of possible tasks, albeit large, therefore the various feasible options and the subsequent comparison thereof becomes a combinatoric problem in nature. An approach is presented for a small example to illustrate that it is possible to generate the entire feasible region.

The theoretical approach of the minimum cost pairing is an adaptation of the Depth First Search Algorithm by W. Dijkstra [29]. Depth First Search (DFS) is used to traverse spanning trees such as those described by the flight network. The nodes represent the flight tasks in the spanning tree. The DFS algorithm starts at a root or random node (flight task). The DFS then traverses or explores as far as possible along the branches of the chosen root, then backtracks. In so doing an assessment is made of the shortest path. The traversal can remember

those nodes that have previously been visited and include considerations for the order in which the nodes have been visited.

In the CPP, consider the starting task i as a parent node. For each feasible node j , a traversal is required of the children's feasible nodes. At each traversal, the child nodes become the parent, feasibility criteria is assessed for the subsequent nodes and multiple 'temporary duties' are stored until all the possible, feasible paths of the start node have been traversed. After all task i paths have been explored, a comparison is made of the most affordable path, the temporary duty paths are then deleted and only the cheapest path nodes are listed as 'covered' or 'completed' in the taboo list. The next task with the next start time (but not already covered) is selected and the traversal for optimal paths begins. This continues until all the flight tasks have been assigned a duty path. Multiple duties are created with various paths and stored with each possible duty map for a given network. The search for the optimal set of duties, requires that all combinations of duties are explored. This exploration is costly as temporary data needs to be stored for each branch search, until an optimal set of duties is selected using the comparator function. The cost reduces over the life of the search due to the 'taboo' or 'covered' list, since less options are available for feasible tasks to include in the subsequent duty paths. The pseudo-code for this algorithm is given in Algorithm 3.1 - 3.2 (MCP-1).

```

1 Input: Temporary duties ( $k_{temp}$ );
2 Output: Least cost temporary duty;
3 Temp duty comparator procedure;
4 foreach temp duty  $k_{temp}$  do
5   | sum temp duty cost  $C_k^v temp + C^f$ 
6 end
7 select least cost temp duties;
8 add flight tasks from selected least cost temp duty flight tasks to Covered nodes list;

```

Algorithm 3.1: Temporary duty comparator procedure (TDC).

An illustrative example of the MCP-1 is given below. The MCP-1 Algorithm generates all possible duty sets and stores them temporarily as described in Tables 3.3 to 3.16. A small example is used to illustrate how the algorithm works. Consider a small network problem represented by the adjacency cost matrix C for a CPP-5 problem.

$$C = \begin{pmatrix} 0 & 20 & 0 & 35 & 10 \\ 0 & 0 & 67 & 40 & 0 \\ 0 & 0 & 0 & 0 & 12 \\ 0 & 98 & 40 & 0 & 0 \\ 0 & 0 & 30 & 45 & 0 \end{pmatrix}$$

Also associated with the above network of connections, consider the start times, end times and duration (computed) of the tasks in CPP-5 given in Table 3.2.

From the fourteen temporary solutions generated (as seen in Tables 3.3 to 3.16), it is clear that duty set 13 is the least cost duty allocation, with two duties ($2C^f$) and the least variable connection cost $C_k^v = 72$. The comparator function of MCP-1, evaluates the total cost of all the feasible sets generated and selects the least cost set.

The selection of the starting nodes and determination of whether a node is chosen as a parent node is one of the feasibility considerations. The data used is such that the start times

```

1 Input Network adjacency matrix ( $A$ ) to determine Adjacency( $i$ );
2 flight task start times ( $S_i$ ) and end times ( $E_i$ );
3 Output: Network traversal of flight tasks ;
4 Calculate flight task durations;
5 foreach  $i \in |\mathcal{F}|$  do
6   |  $T_i = E_i - S_i$ 
7 end
8 Create temporary duty sets;
9 for  $\forall i \in |\mathcal{F}|$  do
10  |  $i$  is visited = true ;
11  | DFS procedure  $DFS(\mathcal{F}, A, i, k, S_i, E_i)$  foreach  $j \in \mathcal{F}.Adjacency(i)$  do
12  |   | if  $j$  is visited == false then
13  |     | if duty duration +  $T_j \leq 480$  and  $S_i \leq S_j$  where  $i = k(end)$  then
14  |       | add  $j$  to duty  $k$  ;
15  |       |  $j$  is visited = true;
16  |       |  $DFS(\mathcal{F}, A, i, k, S_i, E_i)$  else
17  |       | start new duty  $k + 1$  ;
18  |       | add  $j$  to  $k + 1$  ;
19  |       |  $j$  is visited = true;
20  |       |  $DFS(\mathcal{F}, A, i, k + 1, S_i, E_i)$  end
21  |     | end
22  |   | end
23  | end
24  Compare temporary duty sets created by DFS using TDC procedure;
25  Print least cost duties, paths and cost
26

```

Algorithm 3.2: Depth First Search - MCP-1 algorithm pseudocode.

Task $i =$	Start time S_i	End Time E_i	Duration (min)
1	1	30	29
2	45	100	55
3	120	150	30
4	130	180	50
5	200	240	40

TABLE 3.2: Example CPP-5

Duty $_k$	Duty Task Path	Duration	Duty Cost
Duty $_1$	1 \rightarrow 2 \rightarrow 3	114 min	$20 + 67 C^f$
Duty $_2$	4	50 min	C^f
Duty $_3$	5	40 min	C^f
Total duty allocation cost			$87 + 3C^f$

TABLE 3.3: Temporary feasible duty set 1

S_i for the nodes (flight tasks i) in the network are in ascending order with each flight task i to \mathcal{F} , where $|\mathcal{F}|$ is the size of the network. Different flight tasks can be selected as starting points, either randomly or based on a set of rules that still ensure that the start time constraint is still maintained.

$Duty_k$	Duty Task Path	Duration	Duty Cost
$Duty_1$	1 → 4	79 min	$35 + C^f$
$Duty_2$	2 → 3 → 5	125 min	$67 + 12 + C^f$
Total duty allocation cost			$114 + 2C^f$

TABLE 3.4: Temporary feasible duty set 2

$Duty_k$	Duty Task Path	Duration	Duty Cost
$Duty_1$	1 → 5	69 min	$10 + C^f$
$Duty_2$	2 → 3	85 min	$67 + C^f$
$Duty_3$	4	50 min	C^f
Total duty allocation cost			$77 + 3C^f$

TABLE 3.5: Temporary feasible duty set 3

$Duty_k$	Duty Task Path	Duration	Duty Cost
$Duty_1$	1 → 5	69 min	$10 + C^f$
$Duty_2$	2 → 4	105 min	$40 + C^f$
$Duty_3$	3	30 min	C^f
Total duty allocation cost			$50 + 3C^f$

TABLE 3.6: Temporary feasible duty set 4

$Duty_k$	Duty Task Path	Duration	Duty Cost
$Duty_1$	2 → 3	85 min	$67 + C^f$
$Duty_2$	1	29 min	C^f
$Duty_3$	4	50 min	C^f
$Duty_4$	5	40 min	C^f
Total duty allocation cost			$67 + 4C^f$

TABLE 3.7: Temporary feasible duty set 5

$Duty_k$	Duty Task Path	Duration	Duty Cost
$Duty_1$	3 → 5	70 min	$12 + C^f$
$Duty_2$	1 → 4	79 min	$35 + C^f$
$Duty_3$	2	55 min	C^f
Total duty allocation cost			$47 + 3C^f$

TABLE 3.8: Temporary feasible duty set 6

$Duty_k$	Duty Task Path	Duration	Duty Cost
$Duty_1$	3 → 5	70 min	$12 + C^f$
$Duty_2$	1 → 2	84 min	$20 + C^f$
$Duty_3$	4	50 min	C^f
Total duty allocation cost			$32 + 3C^f$

TABLE 3.9: Temporary feasible duty set 7

MCP-1 generates the entire feasible set by enumerating through all possible feasible tasks as illustrated in the 14 duties of the MCP-1 enumerated for CPP-5. In addition, the comparator function also highlights the optimal duties from the generated set of feasible solutions. The optimal solutions generated by this algorithm may potentially be used as a benchmark

$Duty_k$	Duty Task Path	Duration	Duty Cost
$Duty_1$	3 → 5	70 min	$12 + C^f$
$Duty_2$	1	29 min	C^f
$Duty_3$	2	55 min	C^f
$Duty_4$	4	50 min	C^f
Total duty allocation cost			$12 + 4C^f$

TABLE 3.10: Temporary feasible duty set 8

$Duty_k$	Duty Task Path	Duration	Duty Cost
$Duty_1$	2 → 4	105 min	$40 + C^f$
$Duty_2$	1	29 min	C^f
$Duty_3$	3	30 min	C^f
$Duty_4$	5	40 min	C^f
Total duty allocation cost			$40 + 4C^f$

TABLE 3.11: Temporary feasible duty set 9

$Duty_k$	Duty Task Path	Duration	Duty Cost
$Duty_1$	3 → 5	70 min	$12 + C^f$
$Duty_2$	2 → 4	105 min	$40 + C^f$
$Duty_3$	1	29 min	C^f
Total duty allocation cost			$52 + 3C^f$

TABLE 3.12: Temporary feasible duty set 10

$Duty_k$	Duty Task Path	Duration	Duty Cost
$Duty_1$	1 → 2	70 min	$20 + C^f$
$Duty_2$	3	30 min	C^f
$Duty_3$	4	50 min	C^f
$Duty_4$	5	40 min	C^f
Total duty allocation cost			$20 + 4C^f$

TABLE 3.13: Temporary feasible duty set 11

$Duty_k$	Duty Task Path	Duration	Duty Cost
$Duty_1$	1 → 2 → 3 → 5	154 min	$20 + 67 + 12 + C^f$
$Duty_2$	4	50 min	C^f
Total duty allocation cost			$99 + 2C^f$

TABLE 3.14: Temporary feasible duty set 12

$Duty_k$	Duty Task Path	Duration DT_k	Duty Cost (C_k)
$Duty_1$	1 → 2 → 4	134 min	$20 + 40 + C^f$
$Duty_2$	3 → 5	70 min	$12 + C^f$
Total duty allocation cost			$72 + 2C^f$

TABLE 3.15: Temporary feasible duty set 13

for the other algorithms. In reality, however, enumerating through a large network size a recursive algorithm like the DFS approach is computationally expensive and time consuming. Experimental implementation of the MCP-1 for problem sizes $N = 50$ took longer than

$Duty_k$	Duty Task Path	Duration	Duty Cost
$Duty_1$	1	29 min	C^f
$Duty_2$	2	55 min	C^f
$Duty_3$	3	30 min	C^f
$Duty_4$	4	50 min	C^f
$Duty_5$	5	40 min	C^f
Total duty allocation cost			$5C^f$

TABLE 3.16: *Temporary feasible duty set 14*

2 hours on a personal computer to generate a solution and larger problem sizes lead to the program hanging due to limited computer resources to store all the temporary paths (originating from a given starting node or flight task) prior to comparison with paths created from an alternate parent node or flight task. Depending on the planning schedule and requirements, this might not always be the most practical solution approach. As such two more approaches are presented that build in a few additional heuristic rules and are less exhaustive in terms of the number of solutions they produce, making them faster to obtain results. The improvement in speed (as a result of using a heuristic approach) however, comes with a reduced probability of generating optimal solutions.

3.4.3 Minimum Cost Pairing Algorithm 2

Minimum Cost Pairing Algorithm 2 (MCP-2) generates duties by using a greedy heuristic to select least cost connections. In this case the generation and optimisation phases are not distinct and only a subset of the feasible region is generated (as illustrated earlier by Figure 3.1).

MCP-2 starts by selecting a start flight task i and assessing all feasible sequential tasks that can be completed after task i . Of these feasible flight tasks, the task that has the least connectivity cost is selected as the next task in the duty path. The feasibility assessment checks the start time of the next flight task and ensures that it occurs at a later time than the end time of the current task. The second assessment is a duty duration assessment, to ensure that adding the duration of the task in the current path does not exceed the regulatory duration of 480 min. If the task does not meet this criteria it is not feasible. Once a duty has reached its maximum duration, a new duty is started by selecting a new start flight task i that has not yet been covered by any other duty. The algorithm iterates over all flight tasks until all the tasks have been covered by a duty path. It is possible to have a duty path that only contains a single task. Consider again the small network problem represented by the adjacency matrix CPP-5 used to illustrate the workings of MCP-1 and its associated network of connections. Consider the duration and start times of the tasks in CPP-5, t_i , for $i = 1, \dots, 5$ given in Table 3.3. Also consider an initial starting point of $i = 1$ (this may be randomly chosen). The feasible duties as per MCP-2 are given in Table 3.17. The total cost for covering the network is $50 + 3C^f$, which happens to be (in this case) the lower bound solution provided by MCP-1. MCP-2 can provide feasibility from a time perspective where the hard constraint is to ensure regulatory compliance, however, there are instances where it does not provide the most cost efficient set of duty classes. The pseudo-code for this algorithm is given in Algorithm 3.3.

3.4.4 Minimum Cost Pairing Algorithm 3

The third approach, Minimum Cost Pairing Algorithm 3 (MCP-3), is based on Monte Carlo sampling, over a series of iterations to get a best outcome. A single iteration of MCP-3

$Duty_k$	Duty Task Path	Duration T_k	Duty Cost (C_k)
$Duty_1$	1 → 5	69 min	$10 + C^f$
$Duty_2$	2 → 4	105 min	$40 + C^f$
$Duty_3$	3	30 min	C^f
Total duty allocation cost			$50 + 3C^f$

TABLE 3.17: Solution output results using MCP-2 on the CPP-5 example.

```

1 Input Network adjacency matrix (A) to determine Adjacency(i);
2 flight task start times ( $S_i$ ) and end times ( $E_i$ );
3 Output: Least cost next flight task(j) ;
4 Selecting next flight task  $j$  from feasible least of adjacent flight tasks to  $i$ ;
5 foreach  $a_{ij} \neq 0 \in \mathcal{F}.Adjacency(i)$  and  $E_i < S_j$  do
6   | if  $a_{ij}$  is min == true then
7   | | select flight task  $j$ 
8   | end
9 end

```

Algorithm 3.3: Greedy Selection procedure (GS) psuedocode.

is based on MCP-2, however instead of using a greedy heuristic to select the next feasible flight task to be added to a duty, the next flight task is selected uniformly random and the procedure is iterative over a chosen number of simulations P to generate a sample of solutions from which a comparison of the average or least cost solution can be made. As with MCP-2, the algorithm starts with an initial start flight task, whereafter all feasible flight connections are assessed for possible inclusion in the duty. A random feasible flight task is added to the duty. All feasible flight connections are again assessed and a randomly chosen feasible flight task is added to the duty. The search continues until no feasible connections exist, whereafter a new duty is started with a new feasible start flight task that has not been covered yet. The algorithm continues by adding more duties, until all flight tasks have been covered. For each iteration of the algorithm, the total cost and duties obtained are stored. This would be repeated over a set number of iterations P , compared and the least cost iteration $p \in P$ results would be selected. The code of this implementation is given by Algorithm 3.5 and 3.6.

The steps of the algorithm are demonstrated by means of the example presented in section 3.4.2. Consider adjacency cost matrix C_a for a CPP-5 problem and the related flight task start time and durations given in Table 3.3. If flight task $i = 1$ is taken as the first selected parent node (note that the starting node can be any point in a network \mathcal{F}), given the feasibility constraints are met as described in §3.1.4, this results in flight tasks 2, 4 and 5 being the possible feasible tasks that can be selected as the next flight task to be included in the current duty, k . The algorithm, then randomly selects one of the three flight tasks as the next task in the duty. This search is recursively conducted until all flight tasks have been assigned to a duty.


```

1 Input Network adjacency matrix (A) to determine Adjacency(i);
2 flight task start times ( $S_i$ ) and end times ( $E_i$ );
3 Output: Feasible duty set, total cost of duty ;
4 Initialise duties  $k = 1$ ;
5 Initialise flight tasks  $i$ ;
6 Declare random number (rand) as an integer ;
7 foreach  $i \in |\mathcal{F}|$  do
8   |  $T_i = E_i - S_i$ 
9 end
10 while  $size(\text{Covered nodes list}) \neq size(\mathcal{F})$  do
11   | GS procedure;
12   | if  $duty\ duration + T_j \leq 480$  then
13     |   add flight task  $j$  to duty  $k$  ;
14     |   current flight task  $i = j$ 
15   | else
16     |   start new duty  $k + 1$ ;
17     |   Starting task for duty  $k = rand \in (1, |\mathcal{F}|)$  with Covered flight task list is taboo ;
18     |   Repeat 13 to 14 ;
19   | end
20 end

```

Algorithm 3.4: MCP-2 algorithm pseudocode.

```

1 Input Network adjacency matrix (A) to determine Adjacency(i);
2 flight task start times ( $S_i$ ) and end times ( $E_i$ );
3 Output: Least cost next flight task(j) ;
4 Selecting next flight task  $j$  from feasible least of adjacent flight tasks to  $i$ ;
5 foreach  $a_{ij} \neq 0 \in \mathcal{F}.Adjacency(i)$  and  $E_i < S_i$  do
6   | if  $a_{ij}$  is  $\neq 0$  then
7     |   select random flight task  $j$ ;
8     |   and add flight task to duty  $k$ 
9   | end
10 end

```

Algorithm 3.5: Random Selection procedure (RS) psuedocode.

```

1 Input Network adjacency matrix (A) to determine Adjacency(i);
2 flight task start times ( $S_i$ ) and end times ( $E_i$ );
3 Output: Feasible duty set, total cost of duty ;
4 Initialise duties  $k = 1$ ;
5 Initialise flight tasks  $i$ ;
6 Declare random number (rand) as an integer ;
7 for  $p=1:P$  (Number of iterations) do
8   foreach  $i \in |\mathcal{F}|$  do
9      $T_i = E_i - S_i$ 
10  end
11  while  $size(\text{Covered nodes list}) \neq size(\mathcal{F})$  do
12    RS procedure;
13    if  $duty\ duration + T_j \leq 480$  then
14      add flight task  $j$  to duty  $k$  ;
15      current flight task  $i = j$ 
16    else
17      start new duty  $k + 1$ ;
18      Starting task for duty  $k = rand \in (1, |\mathcal{F}|)$  with Covered flight task list is
19      taboo ;
20      Repeat 13 to 14
21    end
22  end
23 Return least cost iteration  $p$  feasible duty set and total cost of duty

```

Algorithm 3.6: MCP-3 algorithm pseudocode.

Monte Carlo Concepts and Convergence

The random selection of feasible ‘next’ flight tasks within a duty, in MCP-3, is what makes the algorithm an application on Monte-Carlo [48]. In this application, from a list of feasible adjacent nodes, random numbers are generated to select the next feasible node. If the random number that is generated is not part of the feasible list, the randomizer function, generates another number until a feasible number i is selected. The duties generated by MCP-3 represent a sample population from the exhaustive solution space. Given the random sampling approach, there exists a certain probability of attaining the optimal duty solutions [9].

3.5 Computer Implementation

Given the object-oriented nature and multi-typed input and output requirements of the problem, object oriented programming was used to program the algorithms. The algorithm code was written and implemented using Java with a NetBeans editor.

Object-oriented programming comprises of classes or objects defined with attributes and procedures which are then applied in various ways to return a specified output or solution using algorithms or robust formula. Table 3.18 specifically illustrates the object-oriented program that was written to solve the CPP in this study. The classes represent elements of the program that were defined and could be used by multiple algorithms, these elements include attributes that describe the individual flight legs, their duration and connection costs. Two classes in particular are defined: flight legs and duties. Extracts of the various objects

and main base code are included in **Appendix C**

Program Function	Name	Description
Classes (Objects)	Flight Legs Duties	Duration, Cost, Start Time, Index Value Total duration, Total cost, sequential index path
Input	Start time End time Adjacency matrix	Start times of the flight tasks in a network End times of the flight tasks in the network Adjacency matrix representing flight-tasks and adjacent flight tasks in a network
Methods (Object property)	Feasibility Recursion Heuristic Search	Check duration, check available connection Assign parent node based on feasibility at each iteration or specified interval Generate duties using rule based approach i.e. Min-Cost, Tree Search etc.
Output (Object output)	Final Duties Feasible duty paths	Total duties, paths, costs non-connected tasks Sequential list of feasible flight tasks per duty

TABLE 3.18: A design layout of the inputs, outputs and structure of the Algorithms developed using object-oriented programming in Java.

The methods are a set of functions which can be called on into any algorithm, that implement a set of rules and/or calculations. In particular the rules and calculations are built to be applied on the classes and affect the output of each algorithm. Methods defined for the crew pairing problem are:

- calculation of the duration for the total sequential path;
- logically assessing the availability of possible connections when building a sequential path;
- recursively implementing the assessment of feasibility every time a new node (flight task) is added to the duty path;
- implementing a heuristic that decides how to add new nodes to a sequential path.

Lastly, the output component of the program infrastructure is designed to illustrate the objectives of the problem. The first output is illustrating the adjacency matrix of the network data that has been input from the database. The second output is determined by the feasibility methods which assess which flights can be connected whilst maintaining time and connection feasibility constraints. Once the various methods have been implemented in either an iterative loop (for loop) or while a certain condition is attained (while loop), output files are printed that detail feasible duties, with the duty path maps, total variable flight task connection cost and number of duties. Duties may also consist of single flight tasks that could not be connected in a sequential path.

3.5.1 Data Transformations

The test data used for this study was taken from Beasley's OR Library [58]. Examples of the data input files received are illustrated in Appendix B. The data in the library is provided in .txt format and required character delineation in Microsoft Excel or an equivalent

spreadsheet application. The following transformations were performed once the data was delineated into numeric data and columns:

- The first N rows of the delineated data are separated into another Excel sheet as these represent adjacencies. Adjacency input is then input into a Matlab sparse function that converts the adjacency list as read from Excel using *XLSRead* into an adjacency matrix using a bigraph function.
- The second half of the delineated sheet is extracted and saved as the start times matrix. The first column indicates the start time and the second column indicated the end time, from which the duration is calculated as an additional measure. The top row of each .txt provides the number of flight tasks in the problem and the maximum duty duration (as may be seen in Figure B3).
- The duration of each flight task is calculated as a function of the node (flight task) object in the java code prior to any computation of the implemented algorithms.

3.6 Results

The presented algorithms were tested on a Hewlett-Packard 350 G2 Notebook Computer with Intel(R) core(TM) i5-5200U CPU @2.20GHz, 4GB RAM with 64-bit operating system.

The below terms and abbreviations will be used in the results tables to follow:

- p - number of iterations of algorithm;
- k - number of crew required to service the network;
- C_k^v - variable cost given k number of crew;
- C^f - fixed cost given k number of crew;

In Figures 3.5 - 3.6 and Tables 3.19 the results computed for the various algorithms are included and compared to Beasley's branch and cut methodology. For MCP-2 and MCP-3 algorithm implementations, the initial starting point was $i = 1$.

The results are compared to the output by Beasley and Cao's deterministic tree search algorithm for the same data set [12]. Beasley's solution output generates a total cost of connections and the number of duties for each solution. The output for the number of duties and the total cost of connections in Beasley's solution varies depending on the number of iterations computed for the branch and cut methodology used. In general, there is not too much variance in the number of duties that Beasley uses and the number of duties generated by the deterministic algorithm in this study.

In Figure 3.5, a comparison of the number of duties used for each network problem size N , is made. It may be seen that generally the number of crew for MCP-2 and some iterations of MCP-3 are relatively the same, differing by one or two extra duties per network. The difference becomes more obvious in larger iterations (p) of MCP-3 where a more notable difference in the fixed cost is given for network problem sizes $N = 200, 250, \text{ and } 500$ in Table 3.19.

In Figure 3.6, a comparison between the variable cost obtained by the MCP-2, MCP-3 and

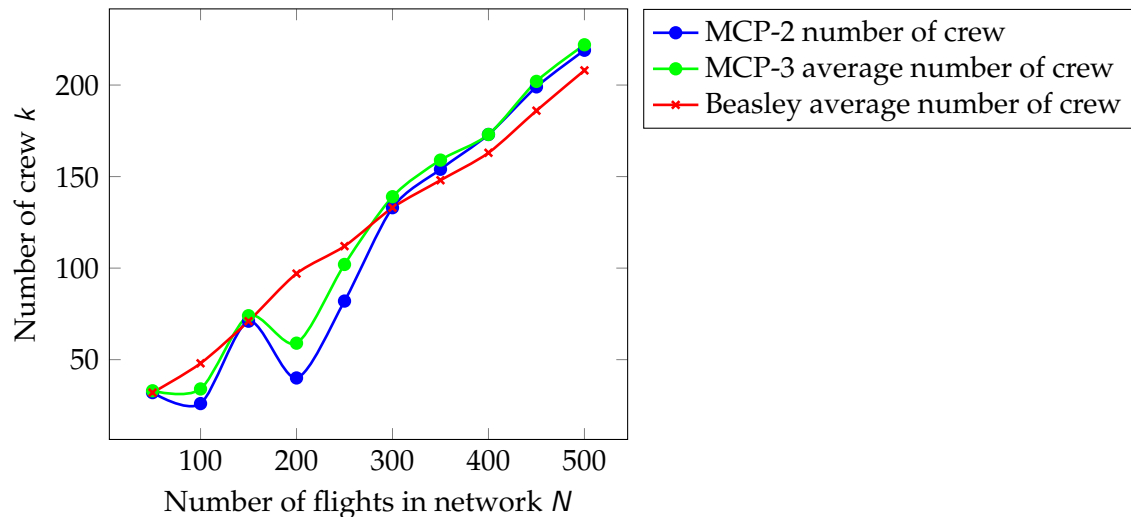


FIGURE 3.5: Comparison of average number of crew obtained between MCP-3 and Beasley's tree search algorithm for various network sizes using the same data set.

Beasley's algorithms are presented for each network problem size. Each algorithm has a distinct logic in selecting the next flight task to be added to a duty. Evidently Beasley's tree search performs much better in terms of variable costs than the MCP-2 algorithm which does not do a complete tree path evaluation before assigning a task to a duty. For larger problem sizes, however, MCP-2 starts to perform better than Beasley for the variable connect cost C_k^v for problem sizes ≥ 400 . The variation in the number of crew is quite low but is important when considering the fixed cost per crew, C^f . Both the MCP-2 and MCP-3 search use fewer crew for the smaller task sizes but eventually converges with Beasley for the larger task sizes. The MCP-3 algorithm performs the worst of the three with significantly higher variable connection costs for each network size compared to MCP-2 and Beasley.

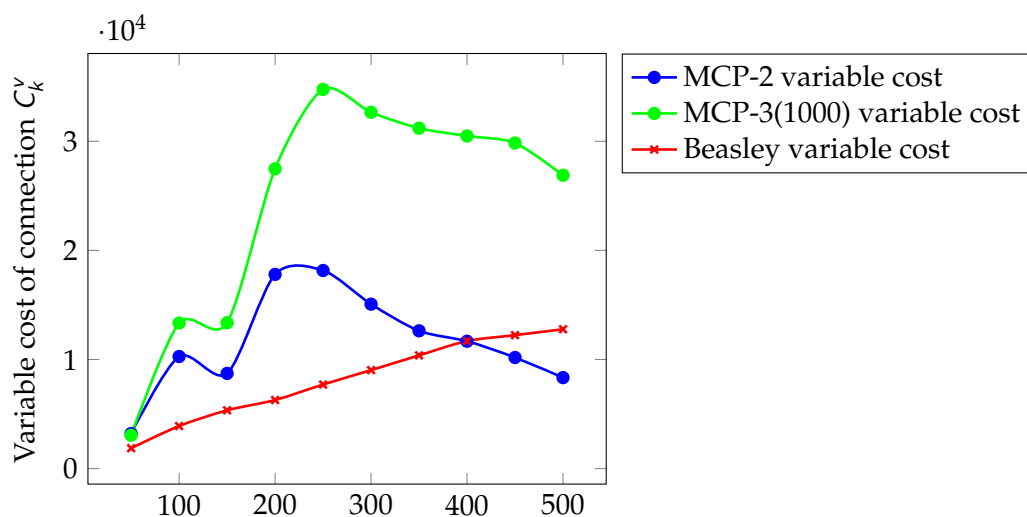


FIGURE 3.6: Comparison of variable connection cost between Beasley, MCP-2 and MCP-3

In Table 3.19, results of different iteration counts of MCP-3 are given. The p-number of iterations generated to produce an average best outcome based on the random flight-task

selection is $p = 10, 100$ and 1000 . The more iterations the higher the probability of generating a good solution that may be in the optimal region as discussed in §3.1.7. It may be noted that with more iterations p , there is a slight but not significant improvement in the total average cost.

Net-work size \mathcal{F}	MCP-3(10) Cost C_k^v	MCP-3(100) C_k^v	MCP-3(1000) C_k^v	Beasley C_k^v	MCP-3 (10) # of Crew	MCP-3(100) # of Crew	MCP-3(1000) # of Crew	Beasley # of Crew
50	3882	3507	3058	1872	33 C^f	34 C^f	33 C^f	31 C^f
100	15260	14670	13344	3905	34 C^f	29 C^f	34 C^f	48 C^f
150	15398	14103	13373	5347	73 C^f	76 C^f	74 C^f	73 C^f
200	30920	29534	27478	6288	59 C^f	54 C^f	59 C^f	97 C^f
250	36505	35445	34755	7707	92 C	97 C^f	102 C^f	112 C^f
300	35913	31211	32658	9026	139 C^f	139 C^f	139 C^f	133 C^f
350	34544	32143	31202	10378	153 C^f	156 C^f	159 C^f	148 C^f
400	33219	31200	30487	11696	175 C^f	175 C^f	173 C^f	163 C^f
450	32397	30720	29841	12232	199 C^f	200 C^f	202 C^f	186 C^f
500	28741	28664	26887	12772	223 C^f	219 C^f	222 C^f	208 C^f

TABLE 3.19: MCP-3 results no delay for different number of iterations p compared with Beasley for the network sizes N

If the start times of the flight tasks were not a consideration, there may have been better solutions attained whereby the larger weighted (longer duration flight tasks) that appear in the list, could have been assigned to duties first. The shorter duration remaining flight tasks would then be placed in the remaining duty capacity and this may have led to the utilisation of fewer duties and hence crew k to service the network as is seen in many bin-packing algorithms [70].

A comparison is also made between the minimum and maximum output values for iterations of Beasley tree search Algorithm (results in Tables 1 and 2 [12]) and MCP-3(1000). In particular, the variable cost of connection is compared in Figure 3.7 and the number of flight tasks used is compared in Figure 3.8. An important observation to note for MCP-3 is that the least variable cost output was not always associated with the least cost number of crew. However, it can be seen that Beasley consistently has lower crew associated with the minimum cost output. This difference may be due to the fact that MCP-3 uses a lot of random decisions in the logical steps and every iteration or set of iterations p can result in different outputs and configurations of the feasible duty sets.

3.7 Sensitivity analysis

Sensitivity analysis is performed to assess how the model output changes to varied changes in the parameters or ways in which the model is implemented.

The results illustrated in Figures 3.5 and 3.6 were based on the initial flight task chosen to start creating duties being $i = 1$. In order to see whether or not there would be a difference in the results or duties created if a different flight task was initialised, MCP-2 and MCP-3 were re-run with a randomised initial flight task and the results are again compared with Beasley and the initial MCP-2 results in Table 3.20 ($MCP - 2_r$) and Tables 3.21 ($MCP - 3_r$) respectively. $MCP - 2_r$ has very little if any variation in both variable cost or fixed crew cost. This may be due to the very robust logic in always looking for the minimum cost next flight task. Where some of the starting points randomly chosen are quite far down the list in terms of start time order, the algorithm would have in any event made those flight

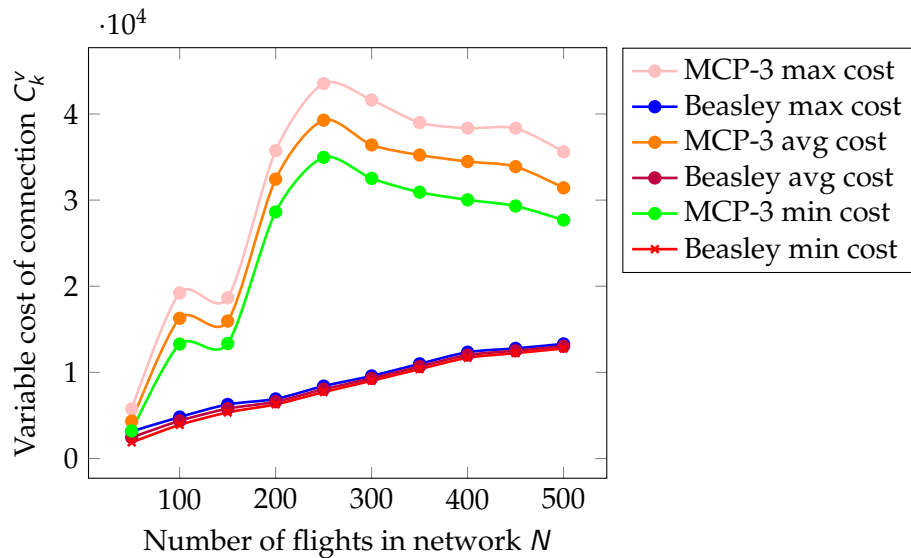


FIGURE 3.7: Comparison of variable connection cost between Beasley, MCP-3 (with 1000 iterations) min, max and average iteration values

tasks as singular duties and it reverts to the chronological logic of assigning duties based on the earliest start time and least cost next flight task. The results are somewhat different for $MCP-3(p=1000)_r$, where there are some differences in both the variable and fixed costs for each network size, this again is attributed to the random nature of the algorithm in choosing the next flight task, coupled now with selecting a random starting point to initialise the duty creation.

The second test that is considered is the impact of creating duty paths by starting at a specific point in the sequence as opposed to randomly selecting which flight task to start creating duties.¹ In particular this is performed to assess whether sequencing the duty paths by starting at the later scheduled flight tasks would be less impactful than starting with flight tasks that are scheduled earlier. For each network tested, a starting index in first quartile in N of flight tasks is compared to an index in the last quartile of flight tasks in the network, i.e. for a network of 100, a starting duties with flight task $i = 25$ (Q1) is compared to starting off at flight task $i = 75$ (Q3).

Figure 3.9 illustrates very little variance (if any) on the impact of starting duty sequencing with earlier index flight tasks versus later index flight tasks. MCP-2 (greedy approach) was the algorithm used and in most instances there is no variance in the number of duties used to cover the entire network. In all network sizes, although very small, there is a difference in the variable connection cost. This is expected given the different starting points for the sequencing of duties. The variance is small, however due to the fact that the flight tasks start times are in ascending order of the flight task index, and ultimately duties will be organised according to the start time feasibility procedure as described in all the algorithms where the start time of index (j) must be later than the start time + duration of predecessor index (i). In addition, flight tasks that are further on in the network typically get allocated to single flight tasked duties. This may be due to the sparsity of the network matrices used from the data set. Furthermore, in the real world country or regional flight networks may show clustering

¹In all the algorithms, there may be a specific manner in which the first task in the first duty is selected, however the first task in the subsequent duties, follows on in a chronological order (earliest possible start time) from the list of flight tasks that are still available.

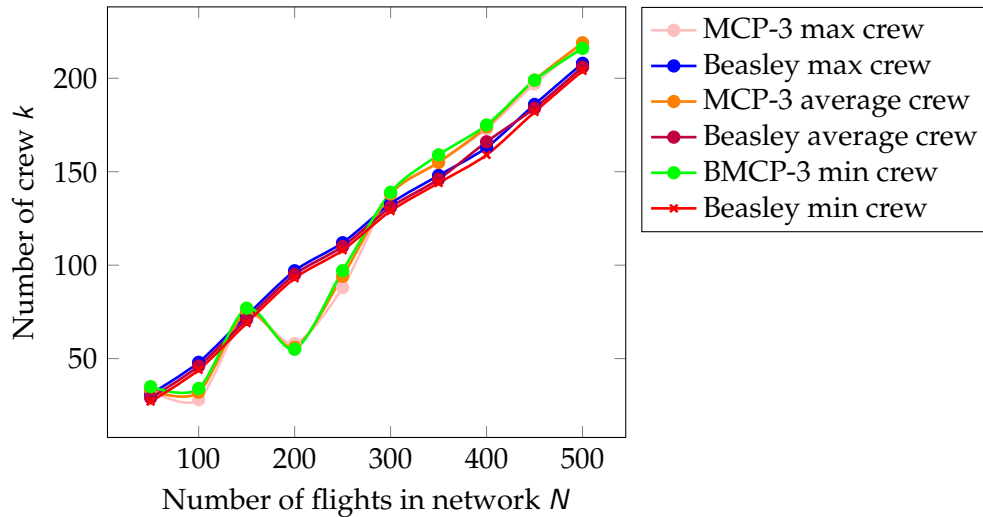


FIGURE 3.8: Comparison of variable connection cost between Beasley, MCP-3 (with 1000 iterations) min, max and average iteration values

Net- work size \mathcal{F}	MCP-2 C_k^v	Beasley C_k^v	MCP-2 _r C_k^v	MCP-2 # of Crew	Beasley # of Crew	MCP-2 _r # of Crew
50	3218	1872	3128	$32C^f$	$31C^f$	$32C^f$
100	10275	3905	9992	$26C^f$	$48C^f$	$27C^f$
150	8727	5347	8727	$71C^f$	$73C$	$71C^f$
200	17801	6288	17614	$40C^f$	$97C^f$	$41C^f$
250	18158	7707	18207	$82C^f$	$112C^f$	$82C^f$
300	15073	9026	15126	$133C^f$	$133C^f$	$133C^f$
350	12631	10378	12732	$154C^f$	$148C^f$	$154C^f$
400	11675	11696	11684	$173C^f$	$163C^f$	$173C^f$
450	10185	12232	10271	$199C^f$	$186C^f$	$199C^f$
500	8341	12772	8341	$219C^f$	$208C^f$	$219C^f$

TABLE 3.20: MCP-2 results with no delay and random starting point.

or more connectivity for popular connections and airports. Table 3.22 illustrates the sparsity of the network adjacency matrices of the data used. An illustration of the CSP-50 adjacency matrix is provided in Figure 3.10 to illustrate the sparsity and the clustering of connections in the earlier start time flight tasks, and the limited connectivity in the later start time flight tasks.

3.8 CPU time

The algorithms developed are also assessed on CPU time per implementation in Table 3.23 as this would be an important consideration for industry application. The computational time is calculated for each implementation for a given network size. In real world scenarios, if there is an eminent delay of known duration, the results of a revised crew pairing for the one day horizon would typically be required between 2 to 6 hours in advance. Therefore, in this study it is ideal to have a run time per network \mathcal{F} that is less than two hours in order for the implementation to be feasible for commercial use. The problem search creates a combinatorial number of paths to be optimised depending on the problem size. This

Net- work size \mathcal{F}	MCP-3 Cost C_k^V	Beasley C_k^V	$MCP - 3_r$ C_k^V	MCP-3 # of Crew	Beasley # of Crew	$MCP - 3_r$ # Crew
50	3048	1872	4539	$33C^f$	$31C^f$	$33C^f$
100	13344	3905	16268	$34C^f$	$48C^f$	$31C^f$
150	13373	5347	15583	$74C^f$	$73C^f$	$76C^f$
200	27478	6288	32307	$59C^f$	$97C^f$	$57C^f$
250	34755	7707	39060	$102C^f$	$112C^f$	$94C^f$
300	32658	9026	35945	$139C^f$	$133C^f$	$138C^f$
350	31202	10378	35304	$158C^f$	$148C^f$	$156C^f$
400	30487	11696	34427	$173C^f$	$163C^f$	$174C^f$
450	29841	12232	34129	$202C^f$	$186C^f$	$199C^f$
500	26887	12772	31343	$222C^f$	$208C^f$	$219C^f$

TABLE 3.21: MCP-3 results ($p=1000$) no delay and a random start point.

Net- work size	adjacency matrix size	Non-zero elements	Sparcity (%)
50	2500	173	93.08
100	10000	715	92.85
150	22500	1355	93.98
200	40000	2455	93.64
250	62500	4150	93.36
300	90000	6107	93.21
350	122500	7882	93.57
400	160000	10760	93.28
450	202500	13510	93.33
500	250000	16695	93.32

TABLE 3.22: Adjacency matrix sparsity measured as a percentage of non-zero elements for all networks sizes used from the data set.

computational cost could be optimised via super computing capability. In Beasley's solution a Fortran processor was used, the high-level coding is computer system independent [12]. The CPU time in Beasley's paper [12] was much higher than CPU recorded for MCP-2 and MCP-3, however the algorithms were implemented on different processors. In addition, the solution quality as illustrated in the comparison between Beasley suggests that Beasley's algorithm provides better connection costs, however this alone would not be a comprehensive assessment of the total cost given that the number of crew in some instances of Beasley's output was higher. Further, a necessary requirement for the solution output of MCP-1 and MCP-2 was that all flight tasks were covered and the assignments met the feasibility requirement, which was not always the case in some iterations of Beasley's solutions. The algorithms here were implemented in a third generation language, where re-factoring was used to define various methods that were called upon during the algorithm to create computational efficiencies. Since MCP-1 requires that each flight task must be visited at least once, the best computation time to solve the problem is at least $O(|V|+|E|)$ time [29]. For both the stochastic and deterministic case, CPU time would be of the same order, given that only the start time files are updated prior to solving the deterministic equivalent using the three algorithms, *cetris parabis*. Processor CPU time output is rounded to the nearest second, as such even the runs that took less than a second to compute, resulted in a 1 second output.

Network size N	MCP-1	MCP-2	MCP-3 p=10	MCP-3 p=100	MCP-3 p=1000
5	20	1	1	1	1
10	420	1	1	1	1
50	>7200	1	1	1	2
100	N/A	1	1	1	2
150	N/A	1	1	1	2
200	N/A	1	2	2	4
250	N/A	1	2	3	6
300	N/A	2	2	3	8
350	N/A	2	2	3	9
400	N/A	2	2	4	11
450	N/A	2	2	5	14
500	N/A	2	2	5	16

TABLE 3.23: A comparison of the CPU time to run the various algorithms for the different network size problems.

the entire region by enumerating through all possible feasible duty sets. The second and third algorithms (MCP-2 and MCP-3) generated parts of the feasible region, using rule based heuristics and sampling techniques. The results for the various algorithms developed are compared with Beasley's results for the deterministic problem of the same data set, solved using a branch and bound heuristic. Beasley's results performed better for the variable costs, the fixed costs were relatively aligned across all three algorithms with little variation. MCP-2 performed better than Beasley for large problem sizes (>400). However the main similarity in the algorithms presented was the fact that all results generated were strictly feasible and calculated within reasonable computational time.

Chapter 4

Methodology and Results - Stochastic Problem

In this chapter, a stochastic formulation of the CPP is presented. A Monte Carlo simulation approach is used to solve for an expected value stochastic CPP (SCPP). A problem formulation is described with an elaboration of the stochastic component in the problem. The algorithm used to solve the SCPP is an adaptation of MCP-3 as described in the earlier chapters (§3.1.7). Worked examples are also provided to illustrate the procedure. The approach presented in this chapter provides airlines with a robust network plan that can withstand expected delays in the network with minimal impact to costs. An airline company may rather want to use the stochastic solution (number of crew), than the deterministic one, as the solution is more robust against delays. The point is not the scheduling, but rather on planning the correct number of crew in advance. The additional crew given a delay event could be considered as the *reserve level* crew previously described in §2.5.2 which some airlines currently implement to manage delays in their air crew planning.

4.1 Delays in air crew planning

Given the competitive nature of the airline industry, it is important that airlines have efficient air crew planning schedules and strategies that minimise the cost implications due to delays. In §2.5.3 various methodology are presented to address possible delays and other elements of uncertainty that may occur in the CPP. Uncertainty in literature has been included in mathematical formulation of CPP, by considering changes in crew [56] or event creating additional crew capacity in the event of a delay scenario [55]. In this study a delay scenario caused by weather is considered and the required number of possible duties to service the flight tasks is calculated for the given delay scenario. The difference in this study to the formulation presented by Silverwood [69] is the fact that this study considers the one day time horizon crew pairing problem with delays, as opposed to complete crew scheduling. Further, the flight task duration is kept constant, which results in changes to the departure and arrival times of the scheduled flight tasks and subsequently, possible changes to the sequential flight tasks that are paired with a given crew. Lastly the delay data used in this study, focusses on weather delay data.

Weather delays constitute a large portion of the delay factors globally as may be seen by the National Airspace statistics presented in Figure 4.1 [21]. This is due to the fact that weather delays not only affect a single carrier, but they affect entire airport operations, with limited grounding capacity, interactions with other airlines and air traffic control. A flight is considered delayed when it arrives 15 minutes later than the schedule [16]. Figure 4.2 illustrates specifically the cost departure delays greater than 15 minutes in Europe for data

collected for the period 2004 to 2008 [61]. Some of the costs would include the items discussed in Chapter 3 for the deterministic CPP and other airline cost implications from the integrated problem.

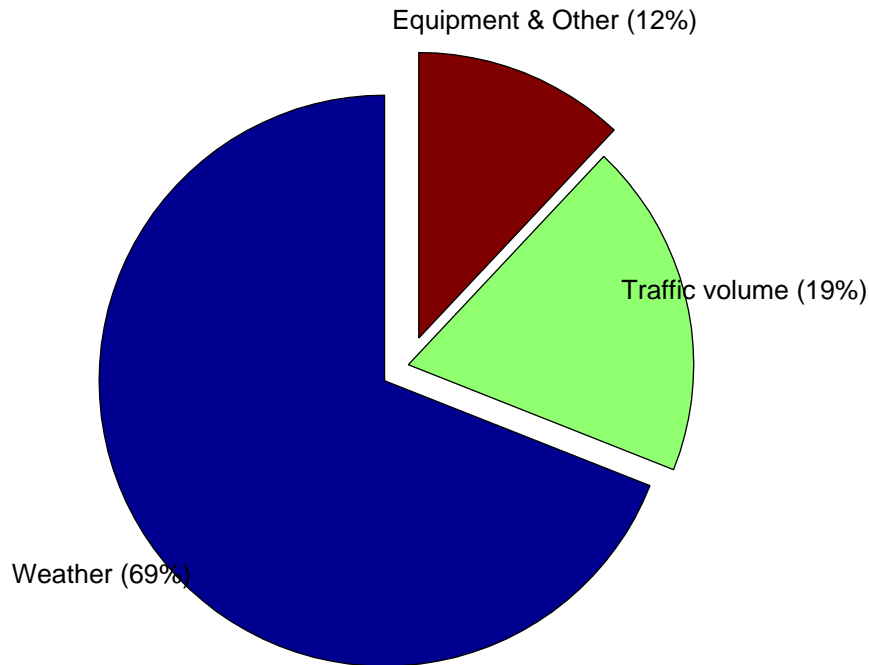


FIGURE 4.1: *Pie chart of flight delays due to various causes including weather, statistics from 2008 to 2013 [21].*

Another consideration for weather delays is the seasonal implication. Different times of the year in different hemispheres are subject to more weather delays than others. This is due to convective weather events being more prevalent during the rainy seasons and winter months. Convective weather events include rain and thunderstorms which are normally accompanied with low ceiling, poor visibility conditions and high winds. Considerations for seasonal weather implications can go a long way in budgeting and planning for delays due to weather. Figure 4.2 illustrates weather delay statistics compared to non-weather delays throughout the course of a given year.

It is also important to note that, although weather is one of the major causes of flight delays, this data may be masked by delays due to late arrival connecting flights. Flights that arrive late and are required for the departure of subsequent flights are classified separately. When en-route flights encounter moderate to severe weather scenarios, passenger safety is of paramount importance to airlines. This entails that flights that have yet to depart or land will have to wait until it is safe to do so. For aircraft that are waiting to land, this typically can cause congestion in the airspace whilst pilots are requested to maintain a holding pattern. Extreme cases may result in aircraft being re-routed to avoid the weather condition. Re-routing to the nearest landing has fuel and passenger delay cost implications for the airlines. In addition this can have a knock on effect on subsequent schedules. The longer the undesirable flight conditions persist, the more severe the impact on the entire network

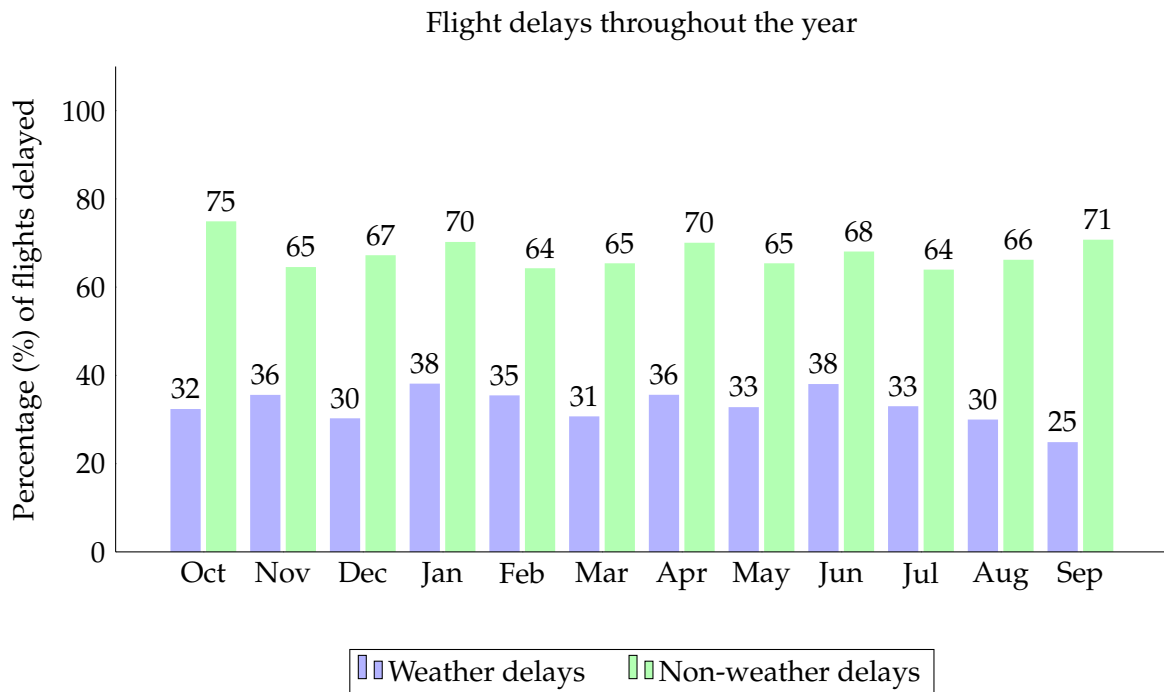


FIGURE 4.2: Flight delays due to weather and other events, from October 2015 [16].

and airspace. Airlines collectively need to have tactical solutions to optimally manage their schedules and the impact given the delays. Weather scenarios and forecasts unfortunately only improve closer to the time, however the later a strategic plan is implemented, the less savings an airline can make. Predictions and amendments to effectively manage delay impact for long haul flights would need to be made at least 4 to 8 hours in advance, and 2-6 hours for short haul flights [16].

In the CPP, a delay of any form has a ripple effect on pre-determined schedules. Some of the implications of delays, re-routing or grounding due to weather events specifically on CPP include:

- Grounded crew may be late for a connecting flight task in their duty
- An existing duty period may already be quite close to the 480 min limit and delays could result in overtime re-imbursements for duties > 480 min.
- Re-routed crew may have to deadhead to the original location at a later stage in order to continue a duty or pairing.
- Additional reserve crew may need to be called upon to fulfill the service requirements for existing flight schedules, where original crew are delayed or not available.

4.2 Modelling approach

Cook and Tanner [1] provide a diagrammatic categorisation of the various phases in which airlines can manage costs due to delays. The first phase of possible cost mitigation is in the strategic or scheduling phase where airlines pre-empt possible delays and put in place mitigates to reduce or pre-empt the potential costs due to delays. The second phase is a tactical phase which is further split into three stages: pre-departure, airborne and post-flight.

The approach presented in this chapter to solve the stochastic case can be implemented as a strategic pre-departure solution method where contingencies are forecasted based on demand and forecast weather data.

In §2.4.4 various approaches to deal with uncertainty in the CPP were considered. In this study, uncertainty is considered by formulating the problem as an expected value problem. This is a common approach or alternative to the two-stage stochastic programming problem formulation with recourse. The modelling approach presented in this chapter for the SCPP builds on the formulation and approaches described in the deterministic methodology section. Random delay events are introduced by changing the start time data inputs. Weather statistics are reviewed in order to determine the probability of a delay that is introduced into the deterministic problem. A Monte Carlo based algorithm is used to solve for feasible duties given a change to the start time data and this results in changes to the feasible region generated. The results of the Monte Carlo provide an expected number of crew and expected variable costs per network size N that airlines can use in their strategic planning or air crew pairing cost forecasts, given a distribution of delays.

4.2.1 Problem Assumptions

The problem assumptions used in the SCPP presented in this study are built on those assumptions described in §3.2.1 and a few more considerations that relate to the delay scenarios in the model:

- A distribution of flight delays due to weather is determined from flight delay data due to weather.
- A random delay with a probability of occurrence, based on the delay distribution, will impact the start time of a flight task. The duration of the flight is not impacted, therefore the flight task end time is adjusted by the same delay amount.
- The additional cost due to delay covers any administrative costs which are incurred to re-assigning crew or crew having to work longer hours. Some of these may include, accommodation, additional per diem for late duties or overlays in the event that late flight tasks are only completed after midnight. This is not calculated explicitly but is implied in the difference between the deterministic and stochastic results.
- The recovery strategy includes common airline strategies used to catch up on lost time due to delays. Some of these strategies include pilots flying faster within safety conditions, quicker turn around times when preparing the plan between flight tasks and reduced *briefing time*.
- A randomly selected delay event can occur at one or more randomly chosen flight tasks in the network.

4.2.2 Randomness and sampling

In order to simulate the impact of uncertainty in the CPP and evaluate the average cost of randomness occurring in a CPP system, the concepts of randomness and sampling are defined. In this study, the sample considered is the set of flight tasks $i \in \mathcal{F}$ in a network of size N , where $N = |\mathcal{F}|$. For each network size, the complete population is available and finite. In probability sampling theory, there are a couple of sampling techniques that may be used, some of which include [71]:

- simple random sampling;

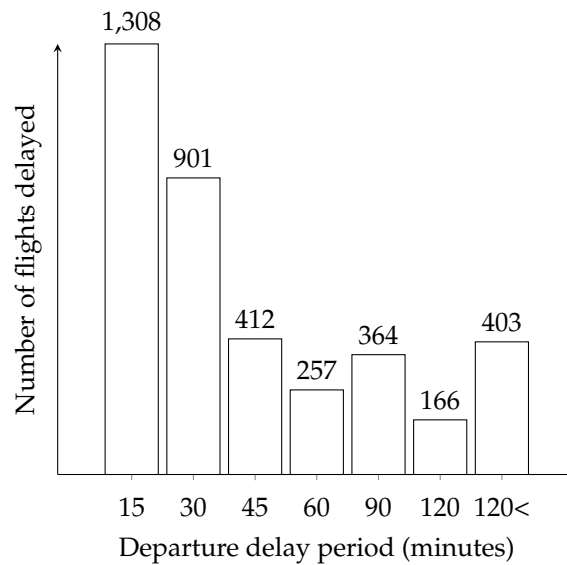


FIGURE 4.3: A distribution of flight delays due to weather in minutes [2]

- clustering;
- multi-stage sampling;
- systemic sampling;
- stratified sampling.

There are also a few sampling techniques that can be used for non-probabilistic sampling and more suited for qualitative or judgement based studies [76]. For the CPP under uncertainty (SCPP), the simple random sampling technique is selected. Simple random sampling can be used for CPP because the following assumptions are made in the model approach under uncertainty:

- There is a population of size N of flight tasks i .
- Each flight task i has an equal probability ($1/N$) of having a delayed start time, therefore,
- the occurrence of a delay at any flight task i is identically and independently distributed.
- For each iteration of a random simulation of delay on the flight network, N , the initial delay affects only one flight task. Subsequent delays of a smaller degree (as a function of the initial delay) will only affect a connecting flight task if it is chosen as the next j in a duty path.

The delay period to be applied on a given flight task, is also randomly selected from a distribution of delays as described in Figure 4.3. The probability of randomly selecting a given delay is given by the distribution D .

The generation of a random number $i \in N$ which would be the selected flight task in the network where a delay would occur can be based on a pseudo-random number generator, typically found in most computer programs [82]. Pseudo-random number generators (PRNG) use deterministic recursive formula to generate sequences of numbers that can be

used in experiments where Monte Carlo simulation is implemented. The starting number chosen as the initial formula input in many of these deterministic models is called the *seed*. In theory, two sequences created using the same seed and the same PRNG would be identical [64]. Common PRNG's include;

- Mersenne Twister - linear recurrence generator;
- Well - linear recurrence generator;
- Micali-Schnorr - cryptographically secure generator;
- Blum Blum Shub - cryptographically secure generator.

Sequences produced by PRNG's can be tested for the quality of their randomness using a variety of statistical tests.

4.2.3 Delay function

A series of random delays is simulated such that the total cost of the duties can change depending on the delay on the flight task start time, where the delay occurs and the recovery strategy employed by the airline company for adjacent flight tasks. The following assumptions are made to simulate the delays:

- Depending on a shock event the time multiplier on the starting time is large at the time and starting point of the shock, but due to integrated operations, management efficiencies and overnight effects this can be recovered before the end of a flight duty cycle.
- The delayed start time for adjacent subsequent flight tasks to those tasks that have initial delays, can be reduced given various airline recovery strategies. This recovery can be approximated by a function.

Therefore, for a given shock the start times are subject to a delay depending on the severity of the scenario and the shock function of the scenario as determined by historical recovery data. If a task start time is affected by a cause of delay, the start times of adjacent tasks to which there are possible transitions are also delayed by a fraction of the initial scenario time factor. Let $S_i^{delayed}$ be the updated delayed start time of task i and S_i be the original start time of task i and d be the delay in minutes. Also define an adjustable delay parameter α denoting the efficiency and efficacy of an airline's recovery strategy given delays to a schedule. Then the delayed start time for all possible subsequent tasks may be given by:

$$S_i^{delayed} = S_i + d \times \exp(-\alpha \times i) \quad (4.1)$$

Subsequent flight tasks may also be affected by at a smaller fraction than the previous task depending on the severity of the scenario impact. Table 4.1 illustrates the impact of a 90 minute delay with recovery parameter $\alpha = 0.5$ on a given flight task. The delays to subsequent connections are also illustrated with the exponential decay function propagating through the duty path.

In this study, a single point delay will be considered in the flight network for a given schedule. However, it is possible to have delays occur at multiple points in the network where the delays may be due to different causes other than weather, as such they may be subject to different delay distributions.

The delay function is included in the solution approach by updating the start time of the

Delay (i)	45 min	60 min	90 min	120 min
1st connection	27.29	36.39	54.59	72.78
2nd connection	16.55	22.07	33.11	44.15
3rd connection	10.04	13.39	20.08	26.78
4th connection	6.09	8.12	12.18	16.24
5th connection	3.69	4.93	7.39	9.85
6th connection	2.24	2.99	4.28	5.97

TABLE 4.1: A 45 minute delay at flight task i and delay at possible adjacent tasks, given a delay function ds_i and recovery parameter $\alpha=0.5$.

affected flight in the input file, and then continue to solve with the applicable algorithm. The start time of any subsequent flight tasks are updated during the algorithmic procedure only once they are included in the same duty path.

In the SCPP where the expected delays are introduced, the connection costs between subsequent flight tasks remain the same. However, a delay in the start time of any or several flights results in a change in the duties obtained, and would, therefore, affect the total cost obtained.

4.2.4 Optimisation objective

The objective function for the SCPP is to minimise the average cost of the crew pairing given an expected delay, $d \in D$. The delay distribution data used in this study is taken from an observation data set of delays at a regional airport in the states with circa 75 000 observations of scheduled flight delays [16]. The inclusion of delays and application of random sampling to ascertain the average cost over a number of iterations is an application of a stochastic approach to solving the stochastic problem.

4.2.5 Mathematical formulation

In §2.5.2 various frameworks and approaches that dealt with uncertainty were presented. The approach used to formulate SCPP, is an expected value approach which would fall under the Monte Carlo simulation framework. The approach chosen for modelling SCPP is appropriate because the cardinality of the set of possible delay from the delay distribution $d \in D$ that can *randomly* occur is finite. Alternatively, if a pre-determined average delay (of flight task start time) were to occur at a random point, the possible number of points $i \in \mathcal{F}$ at which a delay can occur is also finite, this would also be an appropriate expected value problem. In this study a randomly selected delay d from a given distribution D is replaced with the expected value delay period, however the selection of delay d is performed over a number iterations P such that over the iterations, the expected delay is determined and an expected cost and expected number of duties can be determined.

A stochastic mathematical formulation of the CPP is presented in (4.2) to (4.4), as an expected value linear programming problem. Recall the deterministic formulation presented in (3.1) to (3.3). The SCPP includes the ω scenarios, where there exists some uncertainty in the occurrence of a delay at one or more of the flight task start times. The value of the random parameter may not be known, but there is a known distribution for ω which describes the random parameter. The objective is to optimise the crew pairings such that the regulatory and airline crew constraints are adhered to. Consider the set of all feasible duties $k \in K$ under delay scenario d , and the set of flight tasks $i \in \mathcal{F}$, $a_{ik} = 1$ if flight i is covered

by duty k and 0 otherwise. Define C_k^v and C^f as the variable connection and fixed duty cost. Let d be the randomly selected delay with known distribution D . The formulation of SCPP is given by:

$$\text{minimize} \quad \mathbb{E} \left[\sum_{k \in K} C_k^v * (x_k, d) + \sum_{k \in K} C_k^f(x_k, d) \right] \quad (4.2)$$

$$\text{s.t} \quad \sum_{k \in K} a_{ik} * (x_k, d) = 1 \quad i \in F \quad (4.3)$$

$$x_k \in (0, 1) \quad k \in K \quad (4.4)$$

If the stochastic simulation in the CPP is to have a fixed delay event randomly occur at one of the flight task points given a randomly selected $d \in D$, for a given CPP network size N , the probability of selecting delay d is determined by the frequency described by the distribution D . Given that the delay event is simulated on a random flight task point, sampling over P iterations and calculating the cost for each iteration, the expected cost over the sample P can be determined.

4.3 Minimum Cost Pairing Algorithm 4

Minimum cost pairing algorithm 4 (MCP-4) is based on a Monte Carlo approach in order to simulate a series of solutions through iterative sampling of a fixed delay on randomly selected flight task. As discussed in §2.5.3 a Monte Carlo approach is one of the approaches that can be used to solve a stochastic problem. In particular, the objective as per the formulation described in Equations (4.2) to (4.4), seeks to evaluate the least expected cost CPP duty paths given a random delay scenario. The procedure steps for MCP-4 to solve the SCPP are described below, given delay distribution D :

1. Use a random number generator to select a flight task $i \in \mathcal{F}$ with probability $1 \setminus N$.
2. For a randomly selected delay $d \in D$ add the delay duration to the start time of randomly selected flight task i as per Step 1. Also update the possible connecting child and grandchild flight task start times as per delay function. The *subsequent delay* to the connecting child flight task is only applied if that flight task is chosen as part of the duty path sequence from flight task i . The delay will only impact the duty of the delayed flight task. New duties will not be impacted by the delay, unless multiple delays are simulated in the Network.
3. Assess feasibility of possible connections and eliminate non-feasible connecting flight tasks
4. Select the next flight task from the feasible list, randomly (MCP-4a) or using a greedy approach (MCP-4b) to be added to the current duty. The greedy approach adds an additional heuristic to the way in which the next flight task is selected as opposed to a completely random selection.
5. Create duties using Steps 1 to 5, until all flight tasks are allocated to feasible duties.
6. Store the total cost.
7. Repeat Steps 1 to 6 for P iterations (each iteration will have randomly chosen starting flight task where delay scenario is applied)
8. From the stored cost and path data in Step 6, evaluate average cost and average number of duties used in each iteration $p \in P$.

The output from this procedure would allow an airline company to determine and budget for an average expected cost of pairing should there be a given delay $d \in D$ on the network, where the delay occurs at one network point (flight task). Step 8 indicates that an average cost is determined from all the possible paths that have been temporarily stored, as such the optimisation is not explicit. If only the minimum duty path solutions were considered at this stage of the solution approach, airlines would risk the possibility of under budgeting crew and perhaps number of duties required in the event of a delay. The average provides a good approximation to adequately prepare for delay events. The accuracy of the expected delays is the parameter that can lead to more optimal solutions to prepare for delays given good quality historical data.

In step 4, two variations are presented for selecting the next feasible flight task, MCP-4a and MCP-4b. The first approach is completely random and will provide expected value results over a number of iterations P . The second approach seeks to apply additional heuristics that may have more optimal results, as seen in the more optimal deterministic results of MCP-2 as compared to results from MCP-3 which was also random in §3.6. Therefore for the results of the SCPP, the aim will also be to observe if the introduction of the greedy heuristic provides improved solutions to a completely random approach.

A small network (CPP-5) example is used to illustrate the procedure for MCP-4a. Consider the network problem presented in §3.1.5 with cost matrix C and deterministic start times:

$$C = \begin{pmatrix} 0 & 20 & 0 & 35 & 10 \\ 0 & 0 & 67 & 40 & 0 \\ 0 & 0 & 0 & 0 & 12 \\ 0 & 98 & 40 & 0 & 0 \\ 0 & 0 & 30 & 45 & 0 \end{pmatrix}$$

The associated start times and duration of the tasks in CPP-5 are given in Table 4.2.

Task $i =$	Start time s_i	Duration T_i
1	1	30
2	45	100
3	120	150
4	130	180
5	200	240

TABLE 4.2: Example CPP-5 start time and duration

Also consider an arbitrary delay (45min) at flight task $i = 1$ with an airline recovery parameter $\alpha = 0.5$. Let the delay scenario be a single point delay d . Then, Table 4.3 illustrates the possible delayed start time for each flight task, should it be selected as the point of initial delay in the flight task network.

Using a random number generator and the algorithm as described at the start of §4.2.2 step 1 through to step 8, yields the results in Table 4.4. A pseudo-algorithm is also provided in Algorithms 4.1 - 4.3 for the implementation. In this example, the initial point of delay is at flight task $i = 1$, whose start time is delayed by 45 min. However since flight task $j = 2$ is no longer an option, given that the start time is earlier than the end time for $i = 1$, the two remaining feasible connections are $j = 4$ or 5. The random number generator selected flight task $j = 5$ as the next task in the path. Since there are no feasible connections, not already in a duty, a new duty is started, $Duty_2$, the starting point of which is selected as $i = 2$. The

Feasible connecting flight task $i =$	Delayed Start time dS_i	Duration T_i
1	46	30
2	90	100
3	165	150
4	175	180
5	245	240

TABLE 4.3: Example CPP-5, moderate delay impact for all flight tasks in network

possible connections are $j = 3$ or 4 , of which $j = 4$ is randomly selected. Since no feasible connections can take place after flight task $j = 4$, $Duty_3$ is created and includes the last remaining flight task $i = 3$. Note the delay only affects $Duty_1$ and the 45min delay from flight task $i = 1$ is propagated to a smaller delay at the next flight task in the duty of 27.29min. Additionally, the other duties are not impacted by the delay from flight task $i = 1$ because they have independent paths. This process is run for a couple of iterations, with each iteration selecting a random delay that is propagated through the system at a randomly selected start point. The average results are collated to get an expected cost and number of duties for the crew pairing for a given network N with delay distribution D .

Data: Delay distribution D

Result: Apply random delay d to selected flight task i

```

1 ;
2 Declare random number (rand) as an integer;
3 for  $l \in \text{length}(D)$  do
4 |   select rand() index  $l$  and return index value  $d(\text{delay})$ 
5 end

```

Algorithm 4.1: Random delay propagator from distribution (DP)

```

1 Input Network adjacency matrix ( $A$ ), to determine Adjacency( $i$ ) ;
2 flight task start times ( $S_i$ ) and end times( $E_i$ );
3 Output Next flight task  $i$ ;
4 foreach  $a_{ij} \in \mathcal{F}.\text{Adjacency}(i)$  do
5 |   if  $a_{ij} \neq 0$  then
6 | |   select random or minimum  $j$  and update flight task  $j$  start time by
6 | |   propagated delay  $d$ 
7 |   end
8 end

```

Algorithm 4.2: Random next flight task selector (RS)

$Duty_k$	Duty Task Path	Duration DT_k	Total delay(m)	Duty Cost (C_k)
$Duty_1$	$1 \rightarrow 5$	69 min	45 + 27.29	$10 + C_f$
$Duty_2$	$2 \rightarrow 4$	55 min	0	$40 + C_f$
$Duty_3$	3	0	30	C_f
Total duty allocation cost				$50 + 3C_f$

TABLE 4.4: MCP-4, 45 min single point delay, $\alpha = 0.5$

4.3.1 A note on Monte Carlo results

The objective of simulating a random delay $d \in D$ over many random starting points is to provide the airline decision makers with an expected cost they can pre-empt in the crew pairing problem given a particular delay scenario. This gives the airline the opportunity to make a more robust selection in terms of number of crew in the event of a delay. Based on the algorithm and the output thereof, a particular path is not expected, but rather an average cost over many different paths that can be taken and the connection cost thereof. This is due to the fact that the algorithm computes a path and a cost for each iteration p and temporarily stores the path map and cost until all iterations P have been simulated with different starting points chosen as the delay point. Once this is done, the average cost is computed from the temporarily stored paths and costs, by simply summing the total cost over the number of stored iterations. The paths however cannot be averaged as these are maps with unique paths and not of the type 'integer' or 'double' but rather are defined as *string* objects. The Monte-Carlo simulation effectively establishes the expected total connection cost and fixed costs given a delay.

```

1 Input Network adjacency matrix(A) to determine Adjacency(i);
2 flight task start times ( $S_i$ ) and end times ( $E_i$ );
3 Output Minimum cost duties;
4 Initialise duties  $k = 1$ ;
5 Initialise flight tasks  $i$  (current visited task);
6 Declare random number (rand) as an integer ;
7 Initial  $i = rand \in (1, |\mathcal{F}|)$ ;
8 foreach  $i \in \mathcal{F}$  do
9   |  $T_i = E_i - S_i$ 
10 end
11 Fetch delay  $d \in D$  using ((DP));
12 for  $p = 1: P$  do
13   | while  $size(Covered\ flight\ task\ list) \neq |\mathcal{F}|$  do
14     | propagate initial delay  $d$  as per procedure ((RS));
15     | update flight task  $i$  start time by  $d$  (for first flight task only);
16     | apply (RS) to get next flight task  $j$ ;
17     | update next flight task  $j$  start time by delay function;
18     |  $S_{j\text{delayed}} = S_j + d \times \exp(-\alpha \times j)$  (for tasks in first duty only);
19     | if  $duty\ duration + T_j \leq 480$  and  $S_i \leq S_j$  then
20       | add flight task  $j$  to duty  $k$  and Covered nodes list;
21       |  $j \rightarrow next\ i$ , repeat 16 to 21;
22     | else
23       | start new duty  $k + 1$ ;
24       | Initial  $i \in k + 1 = rand \in (1, |\mathcal{F}|)$  Covered flight task list is taboo ;
25       | Repeat 16 to 19 ;
26     | end
27   | end
28 end
29 Calculate average cost of duties  $\forall p \in P$ ;
30 Calculate average duty length  $\forall p \in P$ 

```

Algorithm 4.3: MCP-4 algorithm pseudocode.

4.3.2 Results - Stochastic problem

The stochastic problem solutions are compared to Beasley's results [12] and the stochastic problem results are compared to their deterministic equivalent for each scenario d and recovery parameter α .

The importance of considering delays is still necessary as described by the worked example in §4.2.1, more so in the evaluation of the expected cost due to delays. Airline companies can use this approach to gain approximations to budget costs for crew with the potential to build in additional buffers in order to reduce unexpected cost implications given weather delays (or any other delays for that matter). It is important to remember that even though the number of duties and connection costs may not be too far off the deterministic outputs, the actual duty paths will be very distinct and the indirect costs (reputational as a result of customer service) will need to be accounted for. The distinct nature of the duty paths is attributed to the random selection of the next feasible flight-task. The output for MCP-4 provides an average cost over P -iterations in order to provide an approximate cost for delays. However, the airline company would still need to decide on a crew pairing schedule and to some extent maintain it or apply other recovery techniques at the point where the delay occurs. Tables 4.5 illustrate the results for a single point delay with the weather attributed flight task delay distribution as illustrated in Figure 4.3. The results are compared to the deterministic results solved using MCP-2 with no delay, to illustrate the difference in the variable and fixed costs that an airline may need to consider when expecting delays. Since MCP-4 solves for the expected value problem, the average cost and average number of crew is calculated. Therefore, an airline may want to consider using the results on MCP-4 to have an expected cost of crew pairing for various scenarios given a delay. As can be seen in the results, the expected number of crew increases for some network size problems. See in particular $N = 100, 150, 200, 400$ and 500 . For most of network problem sizes, regardless of the delay, there is an expected increase in the variable cost of connection C_v when compared to the results of the deterministic problem.

Figures 4.4 and 4.5 illustrate the results of the two variations of MCP-4 a) and b) as compared to MCP-2 results with no delay as a benchmark to assess the impact of a delay and how many more additional crew may be required to service a given network. Recall, MCP-4a is a completely random monte carlo simulation where the selection of the next feasible flight task is random, compared to MCP-4b which has a greedy approach. From the results it is clear that MCP-4b has better results compared to MCP-4a for the variable cost and the fixed costs across most network sizes, this may be attributed to the greedy approach in selecting the next feasible flight task, resulting in duties where connection costs are minimised. Interestingly the MCP-4b also has better connection costs than MCP-2 without delay on all network sizes greater than 50. This may be attributed to the fact that there is no iterative process involved in MCP-2 and the greedy process is implemented just once. However in MCP-4b, although delays are sampled from a distribution, the iterative nature in having different starting points to start generating duties and that delays only affect a single duty, followed by a strict greedy approach in assigning remaining flight tasks; all these factors contribute to the lower connection costs. The lower connection cost however does not take away the requirement for additional crew to service most of the network sizes given delay events as can be seen in flight networks of size 100, 150, 200, 250, 400 and 500.

One of the observations made in the results for the SCPP is that there are some network size problems where even with a relatively small delay i.e. 15min, the additional number of crew required to service the network is quite large (see results in Figures 4.4 and 4.5). The

nature of the large problem networks is that the adjacency matrix easily becomes a *sparse* matrix, where some rows have very few non-zero feasible connections. Flight tasks in rows with the aforementioned characteristics are more likely to be allocated to their own duties with no other flight tasks meeting the start time feasibility requirement to be added to the path. Additionally, given that the flight tasks start times are scheduled in chronological order, the later flight tasks are generally only able to be grouped in a duty with flight tasks that are further down the list than they are. This means that the starting point of the algorithm will have little or no effect on the results of the variable and fixed costs, given the start time feasibility criteria and the chronological order of the flight task start times playing a key part in how duties are created as was seen in the comparative results of the deterministic implementation on MCP-2 (see Tables 3.22).

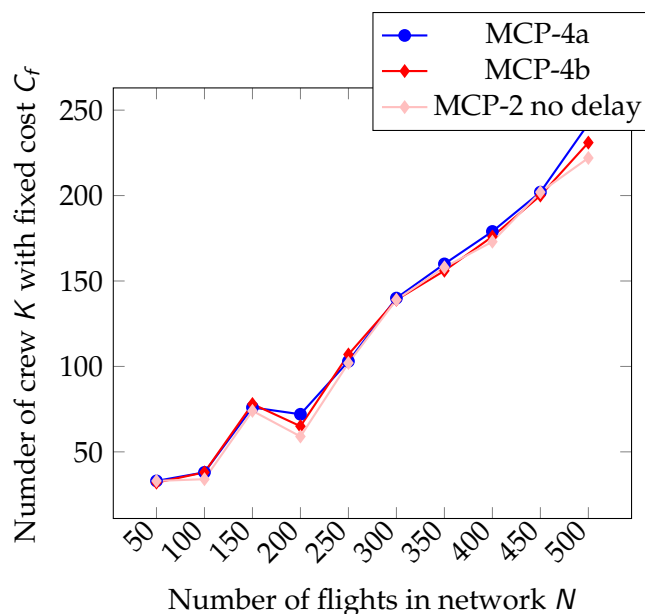


FIGURE 4.4: MCP-4a, MCP-4b and MCP-2(no delay) comparison of C_f .

4.3.3 Sensitivity analysis

In this section the robustness of the proposed algorithms is tested. This is done by introducing various changes in the parameters and assessing the variability in the model output to changes in the input.

The selection of the delays in the expected value problem is random therefore it may be tricky to isolate how different degrees of delays impact the variable and fixed costs as compared to the deterministic problem. This may be a useful *what if analysis* for airlines to determine thresholds for which they would need to drastically change pre-determined pairings. In order to test this, a few categories of delays are described (also taken from the weather attributed flight task delay data). All of the assumptions about the SCPP delay and the impact thereof to crew as described in §4.2.3 are maintained.

The delay function used to simulate an expected delay is a general time decay function with three scenarios selected as per the below based on the tail end of the weather delay statistics histogram in Figure 4.3:

- ω_1 . Slight delay = 15min

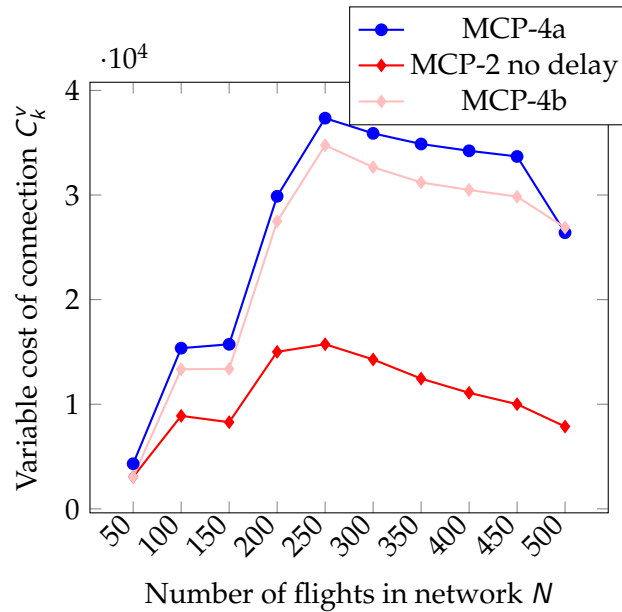


FIGURE 4.5: MCP-4a, MCP-4b and MCP-2(no delay) comparison of C_k^v .

- ω_2 . Moderate delay = 45min
- ω_2 . Bad delay = 90min
- ω_4 . Severe delay = 120min

The only difference in the methodology to evaluate the sensitivity of the model to varying degrees of delays, is that a fixed delay applied to a randomly selected flight task, as opposed to having a randomly selected delay applied to a random flight task. Tables 4.5 to 4.8 illustrate the output of the MCP-4a. The results show that with the increasing delay scenario ω , there is an increase in the number of duties (although incremental) required to service some of the networks. The variance of the variable connection cost is not material enough to draw conclusions of an increasing connection cost with the increasing delay.

The second test for model sensitivity is performed to assess the impact of the recovery parameter α . The impact of a delay propagated from the initial flight task where the delay occurs is such that each subsequent flight tasks start time is delayed by a decaying function, $S_j \times \exp(-\alpha)$. In the main implementation of the delay function discussed in §4.2.3, an α value of 0.5 was used and described as an efficient recovery parameter. Table 4.9 illustrates the actual delay value of the selected α to be used in the sensitivity test to follow, where a delay start time (S_j) of 45min is used to measure the subsequent delay. Table 4.9 illustrates that an increasing delay value for a given delay function $S_j \times \exp(-\alpha)$ results in a smaller delay impact of the subsequent flight task in the path. A couple of different values of α are compared in Tables 4.10. Increasing values of α (larger decreasing exponential function) result in the delay propagating through the system with a quicker recovery, such that the number of additional duties required is less than when the value of α is larger.

The third test for model sensitivity is to evaluate the impact of the timing of the delay, i.e. if the delay happens earlier in the day or later, what is the difference in the expected additional cost due to delay for the airline. Since the flight task start times are in chronological order, the approach to set delays at fixed points within the earlier and later quantiles would be appropriate.

The comparison of the results in Tables 4.11 where a delay is propagated earlier or later in the

Net- work size	MCP-4a Cost C_k^v delay	MCP-2 C_k^v no delay	MCP-4a C_f delay	MCP-2 C_f no delay
50	4330	3048	$33C_f$	$33C_f$
100	15782	13344	$32C_f + 2C_{f(extra)}$	$34C_f$
150	15854	13373	$74C_f + 2C_{f(extra)}$	$74C_f$
200	30661	27478	$59C_f + 10C_{f(extra)}$	$59C_f$
250	38211	34755	$99C_f$	$102C_f$
300	36219	32658	$139C_f$	$139C_f$
350	36305	31202	$138C_f$	$158C_f$
400	34405	30487	$173C_f + C_{f(extra)}$	$173C_f$
450	33947	29841	$199C_f$	$202C_f$
500	26400	26887	$240C_f + 18C_{f(extra)}$	$222C_f$

TABLE 4.5: MCP-4a 15 min single point delay

Net- work size	MCP-4a Cost C_k^v delay	MCP-2 C_k^v no delay	MCP-4a C_f delay	MCP-2 C_f no delay
50	4342	3048	$33C_f$	$33C_f$
100	15637	13344	$34C_f + 3C_{f(extra)}$	$34C_f$
150	15861	13373	$74C_f + C_{f(extra)}$	$74C_f$
200	30401	27478	$59C_f + 11C_{f(extra)}$	$59C_f$
250	37951	34755	$100C_f$	$102C_f$
300	36084	32658	$139C_f$	$139C_f$
350	35260	31202	$157C_f$	$158C_f$
400	34423	30487	$173C_f + 2C_{f(extra)}$	$173C_f$
450	33843	29841	$200C_f$	$202C_f$
500	26399	26887	$222C_f + 18C_{f(extra)}$	$222C_f$

TABLE 4.6: MCP-4a 45 min single point delay

network, illustrates higher sensitivity for the smaller sized networks $\mathcal{F} = 50, 100, 150, 200$ and 250 for both the fixed and variable connection costs. This may be due to the fact that in a smaller network there are less possible connections between flight tasks as compared to larger sized networks. Therefore, a delay earlier in the network leads to a bigger change in the possible duties created than in a larger network where duty lengths are longer and list of feasible next flight tasks is longer. There also seems to be a relationship between the change in connection cost and the change in fixed cost. If the change in connection cost is small, so is the related change in the number of flight tasks used to service the network. First and third quartiles were used in this comparison as a reasonable spread when considering the flight task at which a delay should be propagated. An alternative for future research would be to use quantiles that have a wider spread such as 10th and 90th percentile.

4.4 Chapter summary

In this chapter, the stochastic crew pairing problem (SCPP) was presented as an expected value problem to approximate the extra cost due to possible delays in the flight network. Data was sourced from an airport weather station where observations were recorded of the departure times compared to the scheduled times. In particular, delays due to weather events were recorded, where the delays above 15min were considered in estimating the distribution of the delay variable. Two algorithm variations were presented to solve the expected value problem using a monte carlo simulation approach: one with a greedy heuristic

Net-work size	MCP-4a Cost C_k^v delay	MCP-2 C_k^v no delay	MCP-4a C_f delay	MCP-2 C_f no delay
50	4349	3048	$33C_f$	$33C_f$
100	15250	13344	$34C_f + 5C_{f(extra)}$	$34C_f$
150	15866	13373	$74C_f + 2C_{f(extra)}$	$74C_f$
200	29894	27478	$59C_f + 13C_{f(extra)}$	$59C_f$
250	37696	34755	$102C_f$	$102C_f$
300	35864	32658	$139C_f + C_{f(extra)}$	$139C_f$
350	35082	31202	$157C_f$	$158C_f$
400	34406	30487	$173C_f + 2C_{f(extra)}$	$173C_f$
450	33797	29841	$200C_f$	$202C_f$
500	26398	26887	$222C_f + 18C_{f(extra)}$	$222C_f$

TABLE 4.7: MCP-4a 90 min single point delay

Net-work size	MCP-4 Cost C_k^v delay	MCP-2 C_k^v no delay	MCP-4 C_f delay	MCP-2 C_f no delay
50	4324	3048	$33C_f$	$33C_f$
100	15060	13344	$34C_f + 6C_{f(extra)}$	$34C_f$
150	15743	13373	$74C_f + 3C_{f(extra)}$	$74C_f$
200	29662	27478	$59C_f + 14C_{f(extra)}$	$59C_f$
250	37587	34755	$102C_f$	$102C_f$
300	35905	32658	$139C_f + C_{f(extra)}$	$139C_f$
350	35037	31202	$157C_f$	$158C_f$
400	34314	30487	$173C_f + 2C_{f(extra)}$	$173C_f$
450	33631	29841	$200C_f$	$202C_f$
500	26350	26887	$222C_f + 18C_{f(extra)}$	$222C_f$

TABLE 4.8: MCP-4a 120 min single point delay

α	0.05	0.1	0.3	0.5	1.4
delay	42.81	40.72	33.34	27.3	11.10

TABLE 4.9: Different delay function output for varied α values

Net-work size	$\alpha=0.05$		$\alpha=0.1$		$\alpha=0.3$		$\alpha=1.4$	
	Cost C_k^v	C_f	C_k^v	C_f	C_k^v	C_f	C_k^v	C_f
50	4305	$33C_f$	4322	$33C_f$	4329	$33C_f$	4314	$33C_f$
100	14760	$42C_f$	14888	$41C_f$	15065	$40C_f$	15602	$37C_f$
150	15610	$77C_f$	15563	$77C_f$	15690	$77C_f$	15777	$76C_f$
200	29087	$76C_f$	29011	$76C_f$	29594	$74C_f$	30524	$70C_f$
250	36302	$108C_f$	36745	$107C_f$	37160	$105C_f$	38199	$100C_f$
300	35455	$141C_f$	35409	$142C_f$	35657	$141C_f$	36176	$139C_f$
350	34390	$160C_f$	34484	$160C_f$	34789	$158C_f$	35137	$156C_f$
400	33716	$179C_f$	34177	$176C_f$	34122	$176C_f$	34456	$175C_f$
450	33417	$202C_f$	33435	$201C_f$	33611	$201C_f$	33778	$200C_f$
500	26177	$242C_f$	25950	$243C_f$	26205	$241C_f$	26405	$240C_f$

TABLE 4.10: MCP-4a comparison with different values of recovery parameter α

to select the next flight task in a duty path, the second with a random selector. A brief

Net- work size	MCP-4b C_k^v Q1	MCP-4b C_k^v Q3	% change in cost (C_k^v)	MCP-4b # of Q1 Crew	MCP-4b # of Q3 Crew	% change in cost (C_f)
50	3309	3128	3.12%	31 C_f	32 C_f	5.79%
100	8941	8966	0.29%	37 C_f	36 C_f	2.78%
150	8361	8504	1.68%	73 C_f	73 C_f	0 %
200	15782	15513	1.73%	61 C_f	63 C_f	3.17%
250	16962	16448	3.12%	100 C_f	101 C_f	1.00%
300	14977	15069	0.62%	134 C_f	134 C_f	0 %
350	12681	12736	0.43%	154 C_f	154 C_f	0 %
400	11542	11573	0.03%	174 C_f	174 C_f	0%
450	10183	10185	0.03%	199 C_f	199 C_f	0%
500	7677	7884	2.60%	232 C_f	231 C_f	0.43 %

TABLE 4.11: Computational results using MCP-4b on various network sizes with delay at an early vs late index sequence start.

discussion is provided on the random number generator and sampling techniques most selected for use in the MCP-4 approach. The average or mean expected cost was calculated over P number of simulations. This would give the airline an indication of how much they can expect to spend given a delay on a pre-determined/deterministic schedule. The delay d is implemented only on a single point of the network and impacts the single duty that is generated from the delayed point. Based on the MCP-4 algorithm and child task selection procedures, the *subsequent delay* as per the delay function is implemented on those tasks that are subsequently chosen to connect to the initial delay task in the duty. The results illustrate that in most network problems, given a delay as selected from the distribution, an airline would need to employ additional crew to service the network. Good results are shown where a greedy heuristic (MCP-4b) is used to select the next flight task to add to the network, as compared to the random approach in MCP-4a. Lastly, sensitivity analysis is used to assess the impact of adjusting the recovery parameter α and how different delay severities would impact the types of duties generated.

Chapter 5

Conclusion

In this chapter, a summary of the research is presented and how each objective, as given in 1.3, was achieved. In addition, possible future expansion of the work is discussed, mainly based on the limitations of the current research and implemented algorithms. Recommendations are also provided on the computer language to be used in replicating the study based on some of the implementation challenges and successes for the algorithms presented.

5.1 Summary of research

The main objective of this study was to investigate methodology for solving the airline crew pairing problem, with particular insight on the stochastic version of the problem. The air crew pairing problem is presented in Chapter 2 and an industry and literature review of the problem is completed in fulfillment of Objective I. Terminology, definitions and regulatory requirements were described before reviewing the standard mathematical formulation (set partitioning problem) for the crew-pairing problem and some of the constraints that may be applicable. Given the mathematical formulation different solution approaches that have been applied previously in literature were reviewed in fulfilment of Objective-I. In addition, a review of stochastic approaches was done in order to understand various solution methods that could be applied to deal with possible delays that are prevalent in the air-line industry and in particular those delays that might affect the air crew planning schedule; this was done in fulfilment of Objective II.

In chapter 3 a solution approach to solve the air crew pairing problem for the 1 day duty planning horizon was presented specifically for the deterministic case in fulfilment of Objective III(i). The mathematical formulation was described as a set partitioning problem with a set of base constraints included in the algorithms that were implemented in order to address the regulatory considerations. Furthermore, the objective function had an additional cost element which considered a fixed pay structure for the crew pairing problem based on the total number of crew that would actually be employed to service the optimal duties that were generated. Three algorithms were presented to generate components of the feasible region described by the mathematical formulation. The first algorithm MCP-1, generated the entire region by enumerating through all possible feasible duty sets. The second and third algorithms (MCP-2 and MCP-3) generated parts of the feasible region, using rule based heuristics and sampling techniques. The results illustrated that MCP-2 (greedy approach) performed much better than the MCP-3 algorithm. Both algorithms had similar fixed duty costs when compared to Beasley, but produced significantly higher connection costs for most network problem sizes. MCP-2, however, performed better than Beasley for large problem sizes. This was attributed to the fact that the feasibility constraint is a hard constrained, which was not the case with Beasley (who had some solutions that were infeasible). Sensitivity analysis was performed by testing in the robustness of the algorithms presented. The start times and starting points for creating duties were chosen as the main

parameters for perturbation. The sensitivity analysis illustrated very little variation in the results when different starting points were considered. This was due to the fact that flight task start times were chronological in order, and duties would ultimately be organised according to the start time feasibility constraint.

In chapter 4, the SCPP was presented as an expected value problem, and solved using a monte carlo simulation approach. Stochasticity was introduced into the problem by means of a stochastic delay applied to a random flight in the network. The distribution of the delay parameter was approximated using historic weather and delay data from an airline. Two algorithm variations were presented, based on a Monte Carlo simulation approach; one with a random selector, the second a greedy heuristic to select the next flight task in a duty. A brief discussion is provided on the random number generator and sampling techniques most selected for use in the MCP-4 approach. The average or mean expected cost was calculated over P number of simulations. This would give the airline an indication of how much they can expect to spend given a delay on a pre-determined/deterministic schedule. The delay d is implemented only on a single point of the network and impacts the single duty that is generated from the delayed point. Based on the MCP-4 algorithm and child task selection procedures, the *subsequent delay* as per the delay function is implemented on those tasks that are subsequently chosen to connect to the initial delay task in the duty. The results illustrate that in most network problems, given a delay as selected from the distribution, an airline would need to employ additional crew to service the network. Good results are shown where an additional greedy heuristic (MCP-4b) is used to select the next flight task to add to the network, as compared to the random approach in MCP-4a. Lastly, sensitivity analysis is used to assess the impact of adjusting the recovery parameter α and how different delay severities would impact the types of duties generated. This was completed to fulfill Objectives III.ii and the discussion and comparison of the results thereof to fulfill Objectives IV.

5.2 Contribution to Literature

The main contributions of this dissertation are the following:

- Survey of literature in the field of CPP with an extensive review of various approaches used to solve the stochastic CPP, both heuristic and exact including common approaches and algorithms that have subsequently been implemented in industry.
- Two heuristic approaches that always provides feasible solutions to both the CPP and SCPP. The monte carlo based heuristic applied to the crew pairing problem is not well explored in literature.
- Extensive recommendation for future work (§5.3) are identified in the study from the computational experience, problem formulation approaches and heuristics that may provide better solutions.

5.3 Recommendation for future work

Stochasticity is generally complex to simulate as it is typically influenced by prior knowledge or historical information which may not always accurately predict future events. A first recommendation for future work would be to develop a live management system that uses both geographical informatics systems (GIS) and weather monitoring systems to plot weather trends on a more live basis and feed this information into the algorithm, which can

be solved hours before results may be required to implement tactical changes to an airline schedule. This may require an industry change as possible implementation at a consolidated level, since airlines and airport companies would need to be in positions to react to tactical adjustments in advance. Some of the tactical adjustments may include, quicker taxiing off runways or if all passengers manage to board in time, to allow planes to depart earlier to free up airspace and queuing traffic before weather systems affect pre-determined schedules.

The Monte Carlo strategy used in this study considered a uniform random selection of next flight task in particular for MCP-3 in the deterministic approach and MCP-4 in the stochastic approach. Alternative strategies could have also been used such as:

- 1. Restrict the selection of most expensive flight task only when no other option is available, and then continue with the uniform random selection of the remaining flight tasks.
- 2. Biased random selection of flight tasks, with the cheapest the highest probability to select, and the most expensive the least chance of selection.

In this study a delay was initiated at a single point in the network that would affect the duty in which the delayed flight task would be allocated. For a very large network there is potential for independent delays to occur at more than one point; multiple flight task delays, where the delays may be attributed to more than just weather delays. In each delay case, if there is sufficient observation data to have a distribution from which delays can be applied and propagated through the system, there may be better solutions that airlines can use that are refined approximations for the expected value problem in terms of variable and fixed costs to be anticipated.

The integrated airline planning problem would incorporate the crew pairing, rostering, air-fleet maintenance, scheduling and runway problems solved for and optimised simultaneously. Although this has been looked at for the deterministic instance with two or more of the listed components, this would be interesting to consider with delays incorporated into the system.

The feasible duties created by the algorithms can be thought of as bins into which objects or tasks need to be optimally packed into. Various bin packing techniques can be investigated for solving the problem, with dynamic objects being the tasks whose duration changes given the various stochastic events.

Algorithm MCP-1 could have run faster with additional optimisation of the various recursions running on different cores in parallel environments. The optimal computer implementation of the algorithm may be investigated to improve the run time for the algorithm especially when the problem starts to get larger.

A promising meta-heuristic to apply to the problem is the genetic algorithm. Once feasible duty classes have been identified, where each of the duty classes has attributes and path definitions of sequential duty tasks, these duty classes can represent chromosomes with genetic attributes. A potential structure of the Genetic Algorithm implementation with this problem could be as follows:

- Run MCP 1 or MCP 2 to determine initial solution;
- For Duty classes created, represent these as chromosome structures;

- Perform crossover between chromosomes per iteration;
- Select best performing crossovers for next generation/ iteration;
- Check feasibility score after crossover;
- Stopping Criteria is when feasibility score is maximized to certain threshold determined by size of problem.

The application of the genetic algorithm to the CPP has been widely researched in literature for the deterministic case, however there is still room to improve on the optimisation of pairings for the stochastic case.

5.3.1 Recommendation for computer language implementation of the air crew pairing problem

The algorithms developed were written in Java 1.7 language which lends itself to object oriented programming (OOP) and classifying various components of the crew pairing problem as objects(classes) on which rules (methods) could be applied as described in Table 3.18. Initially MATLAB was used to try develop a heuristic to generate a feasible set, however the strict matrix format and lack of object classification, meant that the algorithm was limited in being able to maintain and monitor changes to each of the objects (duties) that would occur during the code run, especially the storing of temporary duties during the enumeration stage of the Depth First Search. Mathworks have released an OOP MATLAB with their 2008 release [47] that does have very strong OOP capabilities that compete with the likes of Java and C++ or C, with the added benefit of having direct access to mathematics functions and tools that are very well defined in the MATLAB libraries. The challenge however with MathWorks was the limited support on stack exchange for the OOP MATLAB scripts, mostly due to the fact that OOP was traditionally used by developers in Java and C-suite to implement industry applications with dynamic user interface capabilities. As such, there is a wealth of stack exchange support and open source script tools available to developers using Java or C-suite for all types of applications and algorithms across a much broader spectrum than what is currently available for MATLAB OOP.

There are components of the solution methodology that rely on matrix manipulation and other mathematical functions which would have been much easier to compute in MATLAB or even Python. Open source OOP tools such as Java and the C-suite, although very dynamic, require a lot of consideration in the building blocks of any algorithm or simple mathematical formula executions. First and second generation languages require the developer to define the data structures, memory storage components and in most cases download all possible libraries that will be required to implement any mathematical or logical statement. However once defined, these can easily be imported for any main algorithm or variations thereof.

Multiple core processing and threading (also known as parallel programming) is also an important technique in developing rule based algorithms that have multiple loops or temporary data storage requirements as is the case with common meta-heuristics. Java, Matlab and C-suite have these capabilities and different techniques in which multi-core processing or data distribution can be optimised in a given program. This was a consideration for future implementation of the MCP-1 described in §3.1.4. If implemented correctly, multi-core processing can significantly reduce the CPU time for the enumeration techniques described and optimise the run time without the need for a hyper-performance computer.

Appendix A

Graph Theory

A.1 Graph representation of a flight task

Graph Theory is a field of mathematics which describes mathematical structures that can be used to model pairwise relationships between objects. The vertices and nodes are all elements of the graph and the configuration of the graph gives information about the relationship between different nodes, duties in this case.

Let V be a finite set of vertices and E a finite set of Edges, then define subsets of two distinct elements of V by the following:

$$E(V) = \{(u, v) | u, v \in V, u \neq v\}$$

A pair $G = (V, E)$ with $E \subseteq E(V)$ is called a graph (on V). The elements of V are the vertices of G , and the elements of E are the edges of G . The vertex set of a graph G is denoted V_G and the edge set by E_G . Therefore $G = (V_G, E_G)$

The following list contains common terminology used in graph theory:

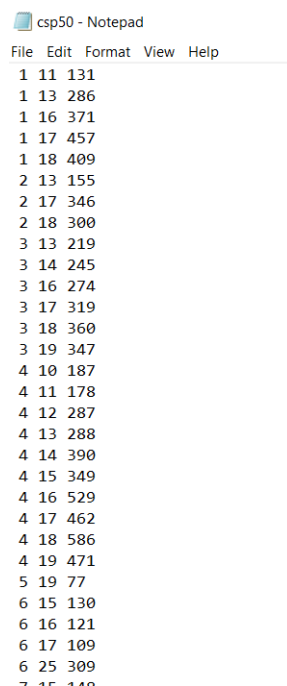
- Order - The number of vertices $v_G = |V_G|$ is the order of G
- Size - The number of edges $e_G = |E_G|$ is the size of G
- Ends - Given an edge uv , the vertices u and v are the ends
- Adjacent- Two edges $e_1 = uv$ and $e_2 = vu$, that share common ends, are said to be adjacent to each other

In addition to graphs just detailing connections between points, there may be direction associated with the edges. $D = (V, E)$ is a directed graph or digraph if the edges have direction such that for $E \subseteq V \times V$, $uv \neq vu$. Graphs and digraphs may be coloured, labelled and weighted.

Appendix B

Data Inputs from OR library

B.1 Input data for adjacency matrix



```

csp50 - Notepad
File Edit Format View Help
1 11 131
1 13 286
1 16 371
1 17 457
1 18 409
2 13 155
2 17 346
2 18 300
3 13 219
3 14 245
3 16 274
3 17 319
3 18 360
3 19 347
4 10 187
4 11 178
4 12 287
4 13 288
4 14 390
4 15 349
4 16 529
4 17 462
4 18 586
4 19 471
5 19 77
6 15 130
6 16 121
6 17 109
6 25 309
7 15 110

```

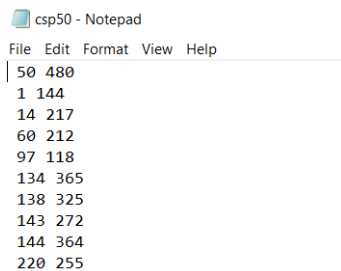
FIGURE B.1: CPP-50 raw file in .txt format where adjacencies are defined by the first column being the flight task, the second column being an adjacent flight task and the third column indicating the cost of the adjacency

B.2 Transformed adjacency matrix

	A	B	C	D	E
1	0	20	0	35	10
2	0	0	67	40	0
3	0	0	0	0	12
4	0	98	40	0	0
5	0	0	30	45	0
6					

FIGURE B.2: CPP-5 transformed adjacency matrix, where non-zero cells indicate an adjacency or connection between the row and column indices in the 5 by 5 network

B.3 Input data for start times



```

csp50 - Notepad
File Edit Format View Help
50 480
1 144
14 217
60 212
97 118
134 365
138 325
143 272
144 364
220 255

```

FIGURE B.3: The first row of the file indicates the network size, in this case $N = 50$, and the limit on duty duration, 480min. The remaining rows of the file indicate the start time and end time for each row (flight task)

B.4 Transformed Start time matrix

	A	B	C
1	1	1	30
2	2	45	100
3	3	120	150
4	4	130	180
5	5	200	240

FIGURE B.4: The corresponding start time matrix for the CPP-5 network. The first column represents the node or flight task, the second column represents the start time and the last column is the end time for the flight task.

Appendix C

Code extracts

C.1 Extracts of flight task objects

```
    * @author Thloni
    */
    public class Node {
        private int nodeIndex;
        private int nodeValue;
        //public int cost;
        private double startTime;
        private double duration;
        //private double cost;
        private double delay;
        private boolean parentDelayed = false;
        private boolean delayed = false;
        final Map<Integer, Integer> adjacency;

        public int getNodeValue() {
            return nodeValue;
        }

        public Node() {
            this.adjacency = new HashMap<>();
        }

        public int getNodeIndex() {
            return nodeIndex;
        }
    }
}
```

FIGURE C.1: This extract illustrates the object class for flight tasks with duration, start time, end time and index attributes.

C.2 Extracts of duty objects

```
    | * @author Thioni  
    | */  
    | public class Duty {  
    |  
    |     public Node node;  
    |     //this should hold the paths  
    |     public List<Node> nodes = new ArrayList<Node>();  
    |     public double Duration = 0;  
    |     public double Cost = 0;  
    |     //Don't think this is necessary  
    |     public Dictionary<Node, List<Node>> paths;  
    |  
    |     public void setDuration(double duration) {  
    |         this.Duration = this.Duration + duration;  
    |     }  
    |  
    |     public void setCost(double cost) {  
    |         this.Cost = this.Cost + cost;  
    |     }  
    |  
    |     public double getCost() {  
    |         return this.Cost;  
    |     }  
    |     /*  
    |     * @return Duration  
    |     */  
    |     public double getDuration() {  
    |         return this.Duration;  
    |     }  
    |  
    |     public void addToPath(Node nodeKey, Node nodeValue) {  
    |         List<Node> adjacency = this.paths.get(nodeKey);  
    |         adjacency.add(nodeValue);  
    |     }  
    | }
```

FIGURE C.2: This extract illustrates the object class for duties with duration, start time, end time, flight task objects, path maps.

Bibliography

- [1] C A & T G, eds. *The challenge of managing airline delay costs*. Conference on Air Traffic Management Economics (University of Belgrade). German Aviation Research Society and University of Belgrade. Belgrade, Serbia, 2009.
- [2] *Airline On-Time Statistics, 2016*. (Online Forum) (Cited March 12th, 2016), Available from. URL: <https://www.transtats.bts.gov/ONTIME/FlightNumber.aspx>.
- [3] ALBERS S & MITZENMACHER M. "Average Case Analyses of List Update Algorithms with Applications to Data Compression". In: *Algorithmica* 21 (1998), pp. 312–329.
- [4] ALONOSO MT, ALVAREZ-VALDES R, PERRENO F & TAMARIT JM. "Algorithms for Pallet Building and Truck Loading in an Interdepot Transportation Problem". In: *Mathematical Problems in Engineering* 2016 (2016), pp. 11–22.
- [5] *Altitude Pairing, 2017*. (Online Forum) (Cited December 17th, 2017), Available from. URL: <https://www.ad-opt.com/crew-planning-solutions/altitude-pairing/>.
- [6] ANBIL R, GELMAN E, PATTY B & TANGA R. *Optimization in Industry: Crew Pairing Optimization at American Airline Technologies*. New York (NY): John Wiley and Sons, 1991.
- [7] O AOUN & A EL-AFIA, eds. *A robust crew pairing based on Multi-agent Markov Decision Processes*. Second World Conference on Complex Systems (WCCS). Agadir, Morocco, 2014.
- [8] ARABEYRE JP, FEARNLEY J, STEIGER FC & TEATHER W. "The Airline Crew Scheduling Problem: A Survey". In: *Transportation Science* 3 (1969), pp. 140–163.
- [9] ATANASSOV E & DIMOV IT. "What Monte Carlo models can do and cannot do efficiently". In: *Applied Mathematical Modelling* 32 (2008), pp. 1477–1500.
- [10] BEALE EML. *The use of quadratic programming in stochastic linear programming*. 2nd ed. Pitman, London, 1961.
- [11] BEASLEY JE & CHU PC. "A Genetic Algorithm for the Set Covering Problem". In: *European Journal of Operational Research* 94 (1996), pp. 392–404.
- [12] BEASLEY JE & CAO B. "A tree search algorithm for the crew scheduling problem". In: *European Journal of Operational Research* 94 (1996), pp. 517–526.
- [13] BEIN D, BEIN W & VENIGELLA S. "Cloud Storage and Online Bin Packing". In: *Intelligent Distributed Computing* 382 (2011), pp. 63–68.
- [14] BIRGE JR & YEN JW. "A stochastic programming approach to the airline crew scheduling problem". In: *Transportation Science* 40 (2006), pp. 3–14.
- [15] *Boeing Concludes Purchase of Carmen Systems, 2017*. (Online Forum) (Cited September 19th, 2017), Available from. URL: <https://boeing.mediaroom.com/2006-06-14-Boeing-Concludes-Purchase-of-Carmen-Systems>.
- [16] *Bureau of Transportation Statistics, Airline Service Quality Performance 234 and Federal Aviation Administration OPSNET, 2016*. (Online) (Cited March 7th, 2017), Available from. URL: http://www.transtats.bts.gov/ot_delay/ot_delaycause1.asp?display=data&pn=1.

- [17] CACCHIANI V, HUISMAN D, KIDD M, KROON L, TOTTH P, VEELTENTURF V & WAGENAAR J. "An overview of recovery models and algorithms for real-time railway scheduling". In: *Transportation Research Part B* 63 (2014), pp. 15–37.
- [18] CAPRARA A, TOTTH P & FICSHETTI M. "Algorithms for the Set Covering Problem". In: *Annals of Operations Research* 98 (2000), pp. 353–371.
- [19] CAPRARA A, GALLI L, STILLER S & TOTTH P. "Delay-Robust Event Scheduling". In: *Operations Research* 62.2 (2014), pp. 30–36.
- [20] *Carmen Systems, 2017*. (Online Forum) (Cited September 19th, 2017), Available from. URL: <https://www.crunchbase.com/organization/carmen-systems>.
- [21] *Causes of air traffic delay in the National Airspace System, 2013*. (Online Forum) (Cited June 18th, 2016), Available from. URL: <https://www.faa.gov/nextgen/programs/weather/faq/>.
- [22] CHAN V. *Theory and Applications of Monte Carlo Simulations*. Intech, 2013.
- [23] CHARNES A & COOPER WW. "Goal Programming and Multiple Objective Optimizations". In: *European Journal of Operations Research* 1 (1977), pp. 39–54.
- [24] CHARNES A & COOPER WW. *Management Models and Industrial Applications of Linear programming*. John Wiley and Sons, New York (NY), 1961.
- [25] CHARNES A & COOPER WW. "Programming with linear fractional functionals". In: *Naval Research Logistics* 9 (1962), pp. 181–186.
- [26] CHEN CH, CHOU TY, LIU TK, CHOU JH & LEE CN. "Optimization of short-haul airline crew pairing problems using a multi-objective genetic algorithm". In: *International Journal of Innovative Computing, Information and Control* 6 (2010), pp. 3943–3959.
- [27] *Column generation with a rule modelling language for airline crew pairing, 2017*. (Online Forum) (Cited April 28th, 2017), Available from. URL: <https://www.orsnz.org.nz/PDFs/Hjorring>.
- [28] CORDEAU JF, STOJKOVIC G, SOUMIS F & DESROSIERS J. "Benders decomposition for simultaneous aircraft routing and crew scheduling". In: *Transportation Science* 35 (2001), pp. 375–388.
- [29] CORMEN TH, LEISERSON CE, RIVEST RL & STEIN C. *Introduction to Algorithms*. 2nd ed. MIT Press, Massachusetts, 2001.
- [30] *Crew Pairing Optimization, 2011*. (Online Serial) (Cited April 28, 2017), Available from. URL: <https://www.researchgate.net/publication/226927493>.
- [31] *Crew Scheduling, 2017*. (Online Serial) (Cited October 18th, 2017), Available from. URL: <https://i11www.iti.kit.edu/amore/housos>.
- [32] DANTZIG GB. "Linear Programming under uncertainty". In: *Management Science* 1 (1955), pp. 197–206.
- [33] DUFEK J. *Monte Carlo Methods and Simulations in Nuclear Technology, SH2704, Introduction*. (Lecture Notes). Nuclear Reactor Technology Division, Department of Physics, School of Engineering Sciences, 2017.
- [34] EHRGOTT M & RYAN D. "Constructing robust crew schedules with bi-criteria optimization". In: *Journal of Multi-Criteria Decision Analysis* 11 (2002), pp. 139–150.
- [35] ELLER RAG & MOREIRA M. "The main cost-related factors in airlines management". In: *Journal of Transport Literature* 8 (2014), pp. 8–23.
- [36] EMDEN-WEINERT T & PROKSCH M. "Best practice simulated annealing for the airline crew scheduling problem". In: *Journal of Heuristics* 5 (1999), pp. 419–436.

- [37] L F, ed. *A container Bin Packing Algorithm for Multiple Objects*. International Conference on Intelligent Computing. Harbin, China, 2015.
- [38] GABALLA A. "Planning Callout Reserves for Aircraft Delays". In: *Interfaces* 9 (1979), pp. 78–86.
- [39] *Global Airline Survey: Getting clear of the clouds, 2015*. (Online Forum) (Cited March 23th, 2016), Available from. URL: <https://pwc.com/us/en/industrial-products/publications/2015-global-airline-ceo-survey.html>.
- [40] GLOVER F & KOCHENBERGER GA. *Handbook of Metaheuristics*. 1st ed. Kluwer Academic Publishers, Dordrecht, 2003.
- [41] K H & S P, eds. *Crew Pairing Optimization with Genetic Algorithms*. Hellenic Conference on Artificial Intelligence (Ioannina, Greece). 2002.
- [42] HASTONIA BJ. "Stochastic Linear Programming with recourse: A tutorial". In: *Decision Science* (1980).
- [43] HOFFMAN KI & PADBERG M. "Solving Airline Crew Scheduling Problems by Branch-and-Cut". In: *Management Science* 39 (1993), pp. 657–682.
- [44] HOFFMAN KL & PADBERG MW. *Set Covering, Packing and Partitioning Problems*. 2nd ed. 7 vols. Encyclopedea of Optimization, 2001, pp. 30–36.
- [45] HOLLAND JH. *Adaptation in Natural and Artificial Systems*. MIT Press, Massachusetts, 1975.
- [46] HOLMGREN S. "Optimization of flight deck crew assignments on Scandanavian Airlines intercontinental flights". MSc Thesis. Linkopings University, Sweden, 2006.
- [47] *Introduction to Object Oriented Programming in MATLAB, 2008*. (Online Forum) (Cited October 20th, 2017), Available from. URL: <https://www.mathworks.com/company/newsletters/artciles/introduction-to-object-oriented-programming-in-matlab.html>.
- [48] JAMES F. *Monte Carlo Theory and Practice*. Review. (Unpublished) Technical Report. Data Handling Division, CERN, 1980.
- [49] LORENA LAN & LOPES FB. "A surrogate Heuristic for set covering problems". In: *European Journal of Operations Research* 79 (4 1994), pp. 138–150.
- [50] MADANSKY A. *The use of "expected value solution" in linear programming with uncertainty*. RAND Corporation, Santa Monica CA, 1960.
- [51] MARTI R, PANOS P & RESENDE M. *Handbook of Heuristics*. Springer International Publishing, 2016.
- [52] *Maximizing productivity while achieving stability, 2016*. (Online Forum) (Cited March 22nd, 2016), Available from. URL: <https://ww1.jepesen.com/commercial/Pairing>.
- [53] MERCIERA A, CORDEAU JF & SOUMISA F. "A computational study of Benders decomposition for the integrated aircraft routing and crew scheduling problem". In: *Computers and Operations Research* 31 (2005), pp. 1451–1476.
- [54] MITCHELL M. *An introduction to genetic algorithms*. 1st ed. MIT Press, Massachusetts, 1998.
- [55] MUTER I, BIRBIL SI, BULBUL K, SAHIN G, YENIGUN H, TAS D & TUZUN D. "Solving a robust airline crew pairing problem with column generation". In: *Computers and Operations Research* 40 (2013), pp. 815–830.
- [56] NEMHAUSER GL, SCHAEFER AJ, JOHNSON EE & Kleywegt AJ. "Air crew scheduling under uncertainty". In: *Transportation Science* 39 (2005), pp. 340–348.

- [57] *On time performance, 2015*. (Online Forum) (Cited December 30th, 2015), Available from. URL: <http://www.airports.co.za/business/statistics/on-time-performance>.
- [58] *OR-Library, 2016*. (Online) (Cited April 26th, 2016), Available from. URL: <http://people.brunel.ac.uk/~mastjjb/jeb/info.html>.
- [59] OZDEMIR HT & MOHAN CK. "Flight Graph based Genetic Algorithm for Crew Scheduling in Airlines". In: *INFORMS 133* (2001), pp. 165–173.
- [60] OZDEMIR U. "Methodology for crew-pairing in airline crew scheduling". MSc Thesis. Bogazici University, Turkey, 2009.
- [61] *Planning for Delay: influence of flight scheduling on airline punctuality, 2008*. (Online Forum) (Cited October 20th, 2017), Available from. URL: <https://www.eurocontrol.int/publication>.
- [62] POWELL WB & TOPALOGU H. "Stochastic Programming in Transport Logistics". In: *Handbooks Operations Research and Management Sciences*. Elsevier Science, Amsterdam, 2003.
- [63] PUBLICATION A. *Economic benefits from air transport in South Africa*. Ed. by SA International Airport Assosiation. Technical Report. 2015.
- [64] *Random Number Generation, 2018*. (Online Forum) (Cited December 1st, 2017), Available from. URL: <https://reference.wolfram.com/language/tutorial.html>.
- [65] H RL, ed. *Introduction to Monte Cralo Simulation*. AIP Conference (Washington). 2011.
- [66] RYAN DM & FOSTER BA. "An Interger Programing Approach to the Vehicle Scheduling Problem". In: *Operational Research Quarterly 27* (1976), pp. 367–384.
- [67] SEN S & HILGE J. "Stochastic decomposition; and algorithm for two stage-linear programming with recourse". In: *Annals of Operations Research 16* (1991), pp. 650–669.
- [68] SHEBALOV S & KLABJAN D. "Robust Airline Crew Pairing: Move-up Crews". In: *Transportation Science 40* (2006), pp. 300–312.
- [69] SILVERWOOD MW. "Application of stochastic programming techniques to airline scheduling". MSc Thesis. University of Witwatersrand, Johannesburg, 2011.
- [70] SRINIVASAN A. "Optimal Advertisement Scheduling in Breaks of Random Lengths". MSc Thesis. Singapore Managemen University, Singapore, 2009.
- [71] *Statistical Concepts and Reasoning, 2018*. (Online Forum) (Cited December 24th, 2017), Available from. URL: <https://onlinecourses.science.psu.edu/stat100/node/18>.
- [72] STEE SV. "Combinatorial algorithms for packing and scheduling problems". MSc Thesis. Habilitationsschrift Universitat Karlsruhe, Germany, 2006.
- [73] STIGLER GJ. "The Cost of Substinence". In: *Journal of Farm Economics 27* (1945), pp. 303–314.
- [74] *Stochastic Optimization - Decisions with uncertainty, 2002*. (Online Forum) (Cited December 15th, 2018), Available from. URL: https://www.me.utexas.edu/~jensen/ORMM/supplements/units/stoch_opt/stoch_opt.pdf.
- [75] *Subsequence Generation for the Airline Crew Pairing Problem, 2011*. (Online Serial) (Cited April 17th, 2017), Available from. URL: https://www.researchgate.net/publication/265635551_Subsequence_generation_for_the_Airline_Crew_Pairing_Problem.
- [76] *Survey Sampling Methods, 2017*. (Online Forum) (Cited December 24th, 2016), Available from. URL: <https://www.statpac.com/surveys/sampling.html>.

-
- [77] TAM B, EHRGOTT M, RYAN D & ZAKERI G. "A comparison of stochastic programming and bi-objective optimisation approaches to robust airline crew scheduling". In: *OR Spectrum* 33 (2011), pp. 49–75.
- [78] PRC V, M N, F MJ & CLDA F, eds. *A heuristic approach for large scale crew scheduling problems at Rio-Sul Airlines*. United States, 2000.
- [79] VALENTE P, R RM & POOJAR CA. *A Stochastic Prpgramming Intergrated Environment*. In: *Applications of Stochastic Programming, Series in Optimization*. S.W Wallace and W. T. Ziemba editors, pp. 123–148.
- [80] WEDELIN D. "An algorithm for large scale 0-1 integer programming with application to airline crew scheduling". In: *Annals of Operations Research* 57 (1995), pp. 283–301.
- [81] WETS RJB. *Stochastic programming: solution techniques and approximation schemes, in Mathematical Programming: the state of the art*. Springer, 1982.
- [82] *What's this fuss about true randomness?*, 2018. (Online Forum) (Cited December 26th, 2017), Available from. 2018. URL: <https://www.random.org>.
- [83] ZEREN B & OZKOL I. "An Improved Genetic Algorithm for Crew Pairing Optimization". In: *Journal of Intelligent Learning Systems and Applications* 4 (2012), pp. 70–80.