WORKFORCE SCHEDULING WITH LARGE-SCALE MIXED INTEGER PROGRAMMING USING COLUMN GENERATION AND 2D GENETIC ALGORITHMS: AN APPLICATION TO AIRPORT GROUND STAFF SCHEDULING

A Dissertation
Presented to
The Academic Faculty

by

Shu Wang

In Partial Fulfillment
of the Requirements for the Degree
Master's Degree in the
Electrical and Computer Engineering

Georgia Institute of Technology
December 2019

WORKFORCE SCHEDULING WITH LARGE-SCALE MIXED INTEGER PROGRAMMING USING COLUMN GENERATION AND 2D GENETIC ALGORITHMS: AN APPLICATION TO AIRPORT GROUND STAFF SCHEDULING

Approved by:

Dr. Roger Jiao, Advisor
School of Mechanical Engineering
Georgia Institute of Technology

Dr. A. P. Meliopoulos, Co-Advisor
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Seung-Kyum Choi
School of Mechanical Engineering
Georgia Institute of Technology

Dr. Ying Zhang
School of Electrical and Computer Engineering
Georgia Institute of Technology

Date Approved: 12 06, 2019

WORKFORCE SCHEDULING WITH LARGE-SCALE MIXED INTEGER PROGRAMMING USING COLUMN GENERATION AND 2D GENETIC ALGORITHMS: AN APPLICATION TO AIRPORT GROUND STAFF SCHEDULING

Approved by:

Dr. Roger Jiao, Advisor
School of Mechanical Engineering
Georgia Institute of Technology

Dr. Ying Zhang
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. A.P. Meliopoulos, Co-Advisor
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Seung-Kyum Choi
School of Mechanical Engineering
Georgia Institute of Technology

Date Approved: 12 06, 2019

[To the students of the Georgia Institute of Technology]

ACKNOWLEDGEMENTS

This thesis would not have been finished without the help of lots of people, and here I greatly thank them for their contributions to this thesis.

Firstly, I would like to thank Inno Industry Technologies, who generously sponsored this project. The project topic is airport ground staff scheduling using optimization techniques. In China, there are many airlines that still use manual workforce scheduling, which not only is a time-wasting task but also requires professional experience. Thanks to the communication between Inno and these airlines and the data they offered, we had the chance to study this project.

Secondly, I am very grateful to my tutor, Dr. Jiao, who offered a lot of technical help when obstacles came up. Initially, a lot time was spent on the technical scheme, and of course, it had many defects. Dr. Jiao gave us valuable suggestions based on his experience in optimization, which made us avoid many unnecessary mistakes. Also, Dr. Jiao also provided help on technical issues, like the crossover method for genetic algorithms and how to formulate the dual problem of the original one.

Thirdly, I would greatly thank Xuejian Gong, who worked with me on this project all the way to the end. I cannot remember how many nights we spent in our Lab studying the technical issues. And what surprised us is that we proposed the 2D GA ourselves, greatly reducing the computation time compared to the normal 1D GA, which made me feel the joy of researching for the first time. We also spent two months ourselves to figure

out the rationale of column generation, which was a great tool to solve rostering and task dispatching problems.

Finally, my parents gave their support unconditionally and made me confident when I felt upset about the project progress. Admittedly, there were a lot of obstacles during our research, and they kept giving me their support so that I could have the confidence to face these problems.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

N	The number of days in a planning horizon
T	The number of equally sized time slots in a planning horizon
a_g	Time granularity of each time slot
φ	The total amount of the requirement category
Φ	The total amount of the requirement category
s	One shift
S	A shift Set
s'	Shift format function
S'	A shift format set
D	The workforce demand
p_s	Shift start time
q_s	Shift end time
l_s	The length of shift s
α_c	Shift operation cost
α_u	Understaffing cost
α_o	Overstaffing cost
c_s	Cost of a specific shift
$u_{\varphi t}$	The number of workforce deficiency of one requirement at a time
$o_{\varphi t}$	The number of workforce excess of one requirement at a time
a_{st}	Operational variable describing the active status of a shift at some time
$m_{s\varphi}$	Operational variable describing the relationship between a shift and a requirement
v_{is}	Variable describing the category of a specified shift

V_i	The maximum number of a particular shift category
$\mathbb{C}$	The chromosome matrix in genetic algorithms
$\mathbb{S}$	The shift matrix in genetic algorithms
$\mathbb{S}'$	The shift format matrix in genetic algorithms
$\mathbb{W}$	The working matrix in genetic algorithms
M	The number of shift formats in a day
r	One roster line
R	A roster
c_r	The labor cost of a roster line
a_{sr}	Operational variable describing the relationship between a shift and a roster line
q_s	The requirement of the number of a shift
a_{gr}	Variable describing a roster line and the pattern group it belongs to
g	A pattern group
G	A pattern group set
m_g	The maximum number of a pattern group
π_s	The dual variable for shift number constraints
μ_g	The dual variable for pattern group constraints
δ	The dual variable for employee number constraints
f_r^*	The fractional part of the number of a roster line minus itself that was rounded down
τ	The threshold for the bound scheme in branch and price
γ	A task
Γ	A task set
$v_{\gamma a}$	The place of departure

$v_{\gamma b}$	The place of arrival
a_γ	The in-position time of a task
b_γ	The end time of a task
t_γ	The active time of a task
h_k	A dispatching line in task dispatching
Ω	A dispatching line set in task dispatching
c_k	The cost of a dispatching line
c_u	The cost for one missing task
u_γ	The number of missing tasks
$a_{\gamma k}$	A variable indicating whether a dispatching line contains a task
z_{sk}	A variable indicating whether a dispatching line is based on a shift
ω_γ	The dual variable for task demand constraints
t_d	The disruption management start time
t_p	The dispatching horizon of disruption management
t_r	The frequency of disruption management
Γ'	Task sets in disruption management
h'_k	A dispatching line in disruption management
Ω'	A dispatching line set in disruption management

SUMMARY

For many service industries, workforce scheduling is an important technique that helps the company to manage employees and tasks, especially for modern airports which transport lots of passengers by flights. In this thesis, workforce scheduling for airport ground staff is studied to facilitate the employee utilization rate and improve the service satisfaction when passengers take a flight.

When a flight arrives or departs the airport, it will generate lots of tasks like cleaning, catering, maintenance, check-in counter services and so on. Usually, these employees do not have all qualifications and thus they are divided into different groups. To manage these groups efficiently in the field where competence is more and more serious, workforce scheduling is implemented, which contains three optimization problems: shift planning that design shifts to cover the workforce demands, rostering that designs work-off lines and the corresponding shift assignment, and task scheduling that assigns tasks to shifts that on-shift employees work on. These problems take place during different time on the planning timeline.

In this thesis, model formulation and corresponding solution for these optimization problems. Shift planning usually has a large scale and is formulated as a set partition model. A 2D GA is proposed to solve the problem. Rostering is also formulated as a set covering model, and it is solved by a column-generation-based method to cope with the challenge of finding feasible roster lines in numerous ones. Task scheduling contains two problems: task dispatching and disruption management. A column-generation-based algorithm is presented to solve the former, while a decision-making system is proposed to convert the

latest dispatching details into a MIP that can be solved using the integer programming solver for disruption management.

CHAPTER 1. INTRODUCTION

1.1 Airport Workforce Scheduling

Modern airports work as a major transportation system and would transport a large number of passengers by flights every day. It becomes increasingly important as international interactions get more and more common. For example, the Atlanta Airport is the busiest airport and has the most people that transfer to the other parts of the world. According to the statistics, the Atlanta Airport served more than one billion and seven million passengers. This is just one example, and there are a lot of airports having a similar scale.

How to manage such a large transportation system becomes a serious problem, and to have efficient workforce scheduling might be the key. On the one hand, it will help improve the employee utilization rate and reduce the labor cost and the operation cost for airlines. On the other hand, it can improve the service satisfaction for service-oriented tasks if well-organized task schedules can be made. In this regard, it is of great significance to have a highly efficient workforce scheduling system for large modern airports.

1.2 Mathematical Modeling and Optimization for Real-Life Problems

In this research, workforce scheduling for airport ground staff using optimization techniques is studied. And this regards to the mathematically modeling and optimization for real-life problems.

There are still many airports that use the manual planning method. Compared to that, the math-based method requires precise descriptions for the problem formulation and one or multiple effective criteria as the optimization objective. On the one hand, precise descriptions require a clear expression of the relationship between each event. Also, it needs to list all the constraints that the model must satisfy and that are coherent to real-life situations. On the other hand, effective criteria will help the optimization system find a more satisfying solution that agrees to the real-world situation.

It is beneficial to use the math-based method to do workforce scheduling. Normally, manual planning takes time and needs professional experience, while the optimization system does not need you to have field knowledge after the system is built. Moreover, it takes less time. The computer nowadays has strong computing abilities, and to solve the optimization problems does not take much time if a proper algorithm is applied. The output of the optimization system can serve as a support to those manual practitioners.

1.3 Research Goals and Scopes

The research is to study the workforce scheduling problem for airport ground staff. Shift planning, rostering, and task scheduling are studied in this research. A 2D GA is proposed to solve large-scale shift planning problems. Rostering and task scheduling are solved with column-generation-based branching and price approach and other integer programming techniques. Cases from a real-life airline are studied to analyze and validate the proposed approaches.

1.4 Organization of the Thesis

This thesis is structured in seven chapters. Chapter 1 presents the basic introduction to airport workforce scheduling and relevant optimization techniques. The background review is in Chapter 2. Chapter 3 describes the planning timeline with corresponding planning objectives and the planning model for ground staff scheduling. Discussion of shift planning problems is in Chapter 4, and a 2D GA is proposed to solve the large-scale shift planning problems considering daily-wise shift formats. Chapter 5 introduces the model formulation for rostering problems with work patterns and the column-generation-based approach. The modeling and the corresponding method for task scheduling (including task dispatching and disruption management) are presented in Chapter 6. Computational experiments are conducted using cases from a real-life airline to validate the proposed approaches. Finally, the conclusions of this research and future work are discussed in Chapter 7.

CHAPTER 2. BACKGROUND REVIEW

2.1 Workforce Scheduling Problems

Workforce scheduling is a challenging problem faced by various service industries like airports, hospitals, and toll collection [1]. It has received large attention since the integer programming techniques are applied to real-world problems. The study on mathematically modeling and the corresponding algorithms for workforce scheduling can reduce the trouble of planning it manually and the solution can also serve as a reference for practitioners.

Multiple optimization problems are included in this field: shift planning, rostering, task dispatching, and disruption management. Much research is done on these individual problems.

Shift planning is the initial step of workforce scheduling and it aims to design a set of shifts with different start time and duration to cover the demands within the planning horizon. Most recent studies emphasize how to construct effective shifts, planning day-offs, and assigning the employees to the day-off schedules. Variations of these problems are found to be NP-hard and NP-complete [2]. The shift planning problem is mathematically formulated as a set covering problem, solved by summing a sequence of shifts from the shift set based on the shift design requirement [3]. Because of the advance in information management software systems, it requires smaller time granularity, larger personnel scale, and longer planning horizon [4]. Meanwhile, from the modern management perspective, designed shifts are desired to be length and placement adjustable with specific break rules [5]. Thus, more efficient shift planning algorithms need to be developed to cope with the increasing computational complexity.

In accordance with Danzig's set covering formulations, the mixed integer programming (MIP) methods have been widely applied to solve the shift planning problems [1, 6]. Metaheuristics are suggested to be critical methods for coping with combinatorial optimization problems by incorporating modular rules to implement shift planning efficiently [7, 8]. Specifically, GAs have been highlighted as an efficient heuristic to solve optimal shift planning problems. A GA encodes the shifts in the form of chromosomes and mimics the natural revolution to search for optimal solutions [9]. Popular generic encoding methods applied to shift planning include binary strings [10], the real value [11], permutation [12], among many others.

The research on rostering problems is widely studied for its critical meaning for the solution quality of workforce scheduling. The detailed review of recent research on personnel scheduling and rostering is presented [2, 13]. Furthermore, among various service industries, the nurse rostering problem (NRP) is a typical problem to study for the multi-qualification and multi-time-period shift type characteristics. Overviews on the models and methodologies are presented [14, 15]. Rostering with work patterns is also a commonly studied problem. A two-phase algorithm is proposed to address the rostering with a 5&2 work pattern [16]. A rostering problem with the 14&7 work pattern considering cyclic weekly demand is studied [17].

Task scheduling is the final step of workforce scheduling. It contains two problems: task dispatching and disruption management. Task dispatching is to assign tasks to each shift while considering the between-task travel time. What makes task dispatching challenging is how to make efficient and feasible assignments from the numerous combinations. This problem is essentially a vehicle routing problem with time windows (VRPTW) and can be solved using column generation, where employees can be regarded as vehicles and each task is a node with task start time and end time as its time window. Much research is done regarding VRPTW. The detailed reviews on VRP and its variations with the latest advances can be found in [18, 19]. Common approaches include heuristics and analytical approaches. Many metaheuristics prove to be useful: tabu search, ant colony, and hill climbing algorithms [20−22]. The column-generation-based approach, as an efficient approach to such combinatorial optimization problem, is initially proposed by Desrochers et al. [23], where the pricing problem is formulated as the shortest path problem with time windows (SPPTW). Further improvements on both speed and quality are made based on Desrochers's approach [24-26].

Disruption management, on the other hand, is less of a mathematical problem than a decision-making process. Because of the existence of uncontrollable factors (extreme weather, security problems, emergencies on airplanes, and so on), flight delays, inserts, and cancellations happen from time to time. Thus, the corresponding adjustment is needed on those affected tasks, and rearrangement operations should be taken. The overview of disruption management in the airline industry and the techniques then is initially presented by Clarke et al. [27], and it is further studied by Kohl at al [28]. Both heuristics and analytical methods prove useful in solving the disruption management problem. Clausen et al. [29] present the set partitioning problem formulation with metaheuristic approaches.

Peterson et al. [30] propose simultaneous row and column generation to deal with large-scale disruption management.

2.2 Airport Ground Handling

Airport ground handling concerns with tasks performed when the aircraft is on the ground, or tasks relating to the arrival or departure, which can be classified into technical-oriented and service-oriented tasks based on the characteristics [31]. In this section, a detailed introduction to these tasks is presented.

From the management perspective, the ground handling work can be divided into ramp tasks that target at the aircraft and terminal tasks that mainly serve the passengers, which is shown in Figure 2.1. In the figure, the terminal tasks are presented in the horizontal direction, and the ramp tasks are given in the vertical direction. Further description of the process for ground handling management is presented by Kazda et al. [32].

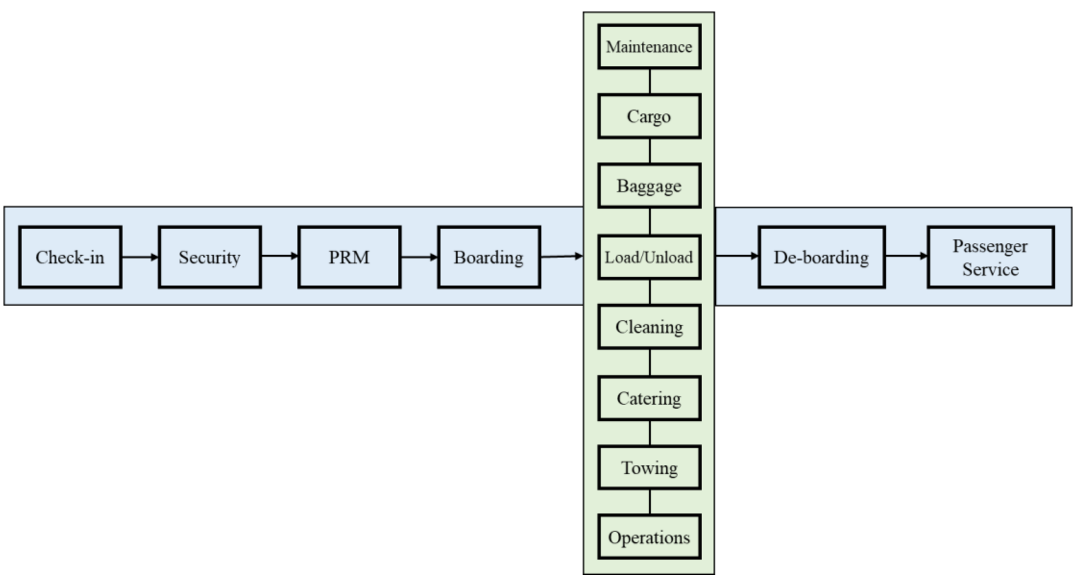

Figure 2.1 Operations from the passenger perspective (horizontal) and employee perspective (vertical).

The terminal tasks take place within the terminals and aim to handle passenger flow at the airport. Such tasks include check-in counter service, security service, and so on. Employees responsible for terminal tasks normally have short travel time between tasks and only work in a small area. The performance measures for such tasks are similar to a queueing model, which includes the waiting time and the waiting number in queue and reflects the passenger satisfaction to the services. Further details on terminal tasks are presented by Brusco et al. [33] and Stolletz et al. [34].

On the other hand, the ramp tasks are usually conducted on aprons or in front of the terminal gate, which can be further classified into above-wing and below-wing. Such tasks include catering, cleaning, maintenance, and so on. Ramp tasks tend to have longer travel time and shorter work time windows between tasks than terminal tasks because such tasks are determined by the flight departure and arrival and have a wide working area. Such characteristics make it challenging to its scheduling problems and much research is done on the optimization regarding ramp tasks [35]. More details on ramp task management can be found in [36].

Normally these airport ground tasks are taken on by service companies instead of airlines [31]. Each service company would be responsible for some part of the ground handling. Thus, the ground tasks are finished by multiple companies, which makes the employment more flexible and makes the workforce scheduling meaningful by reducing the operation cost as well as the labor cost.

2.3 Column Generation for Mixed Integer Programming

Column generation is often applied in the large-scale optimization problem where enumeration is impractical or feasible variables have many constraints [37]. It is applied under the branch and bound framework, and the combined algorithm is called branch and price [38].The pricing problem is usually formulated as the shortest path problem with resource constraints and much research has been published on this topic. Desrosiers et al. propose the algorithm considering only time resources at the beginning [39] and is later generalized by Desrochers [40]. The algorithm is later modified by discarding labels that cannot result in negative reduced cost [41]. Several accelerated algorithms are developed to improve the label correcting process, such as the state-space augmenting approach [42] and bidirectional search [24].

CHAPTER 3. AIRPORT WORKFORCE SCHEDULING: PROBLEM DEFINITION WITH AN EXAMPLE OF GROUND STAFF SCHEDULING

Modern airports work as service industries that provide a fast transportation method for passengers all over the world every day. They usually have a large-scale of employees and need to perform plenty of tasks for the flights they have. For example, statistics show that Hartsfield–Jackson Atlanta International Airport (which has the largest passenger flow every day) serves more than 100 million passengers and has more than 950,000 flights per year. To manage this large volume of transportation involves lots of logistical knowledge and decision making and need to solve corresponding operation problems. This chapter reviews the planning process and the planning problems for airport ground staff scheduling.

3.1 The Planning Horizon and Planning Timeline

Airlines usually need large-scale of employees to perform various tasks for different flights every day. To manage such a large volume of transportation well, planning should be done on employees' shifts, and task assignments in advance, which usually takes place after the flight plan is determined. Such workforce scheduling usually has a long planning horizon (which refers to the planning period) and a planning timeline. It is not practical to plan everything at an early stage, because more details on resources and tasks can only be obtained as the day of operation draws near. Thus, it is of significance to divide the planning process into different periods in the planning timeline and to set different planning objectives in each period.

The planning process can be roughly divided into four periods, as shown in Figure 3.1: Preparation, mobilizing, real-time management, and afterwards analysis. Such classification is based on the date of occurrence from the day of operation. Figure 3.2 presents the planning timeline from the management perspective.

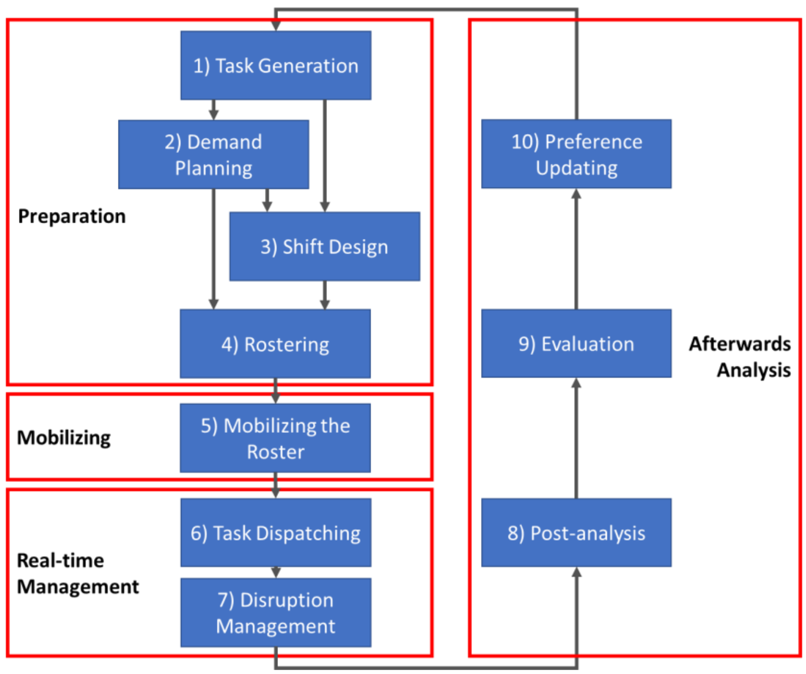

Figure 3.1 Planning process from the algorithm perspective.

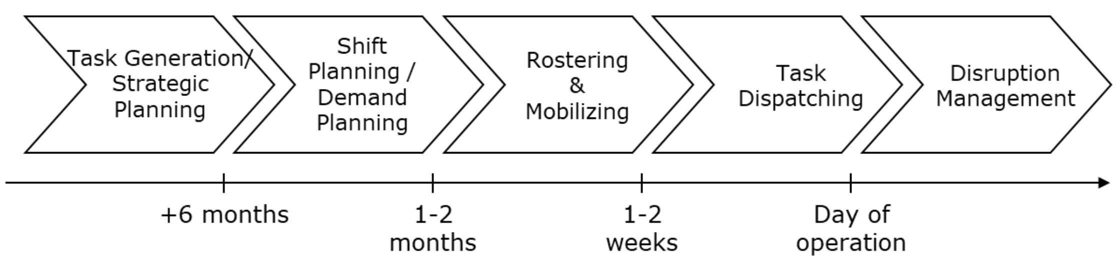

Figure 3.2 The planning timeline.

The preparation period includes long-term and mid-term decision making. In this period, resource allocation (including the number of needed staff and equipment), as well as work-off lines with specified shifts within the planning period should be determined.

Normally, the planning horizon is six months, and the preparation also starts six months ahead of the operation day. Firstly, different types of tasks are generated and organized based on the flight plan within the planning horizon. Workforce demands for each task can be evaluated and then workforce demands for every minimal time granularity within the planning horizon can be obtained. Shift planning is implemented to determine the shifts required to perform generated tasks. This can be solved either by the workforce demands or tasks directly. In this research, only demand-based shift planning is studied. Then the work-off line design and the shift assignment are implemented by rostering.

Mobilizing takes place before one or two weeks from the day of operation. This is for manually adjusting the rostering result to fit the latest resource information. For example, some employees may ask for temporary leave and their absence should be filled.

Real-time management is implemented one day before the day of operation and during that day. Based on the generated tasks and the roster lines, task dispatching is solved, so that each employee will know their task schedule on the day of operation. During that day, disruption management is performed for every fixed time to make a new arrangement when irregularities happen, like extreme weather and emergencies on airplanes.

Afterwards analysis is performed after the day of operation. Things may not go as planned during the operation, so the planning performance should be evaluated to understand the defects existing in the current workforce scheduling system, and corresponding measures should be taken to improve the system.

3.2 A Planning Model for GSS

In this research, a planning model is constructed to provide the basis for the solution of GSS. The model contains three major modules: shift planning, rostering, and task scheduling. Shift planning designs shifts with different start time and duration to cover the workforce demand generated from the flight plan. After determining the type and number of shifts, rostering combines these shifts as different roster lines based on the labor laws and employment rules. Finally, task scheduling deals with assigning specific tasks to employees' shifts before the day of operation and making quick rearrangements if unexpected events happen at the airport.

3.2.1 Shift Planning

Shift planning is the initial step during the workforce scheduling process, and it aims to design a set of shifts with different start time and duration to cover the demands within the planning horizon. The shift planning problem is mathematically formulated as a set covering problem, solved by summing a sequence of shifts from the shift set based on the shift design requirement. The mathematical model for shift planning is presented in Section 4.2.

3.2.2 Rostering

Rostering uses the shift planning result as inputs and combines these shifts as different day-off lines with specified shifts to cover the shift demand while satisfying the labor regulations and employment rules. The detailed problem formulation is in Section 5.1.

3.2.3 *Task Scheduling*

Task scheduling concerns with assigning tasks to shifts before or during the day of operation. Two separate problems of task scheduling are task dispatching and disruption management. The detailed discussion for these two problems is represented in Section 6.2 and 6.3.

3.3 Chapter Summary

On the one hand, workforce scheduling contains a sequence of planning activities. And airport workforce scheduling usually has a long planning horizon and should begin quite early from the day of operation. On the other hand, due to the large volume of transportation that an airline needs to manage, the resource scale is also large. Considering these, it is of significance to divide the scheduling process into different steps and to solve different planning problems during each stage. This chapter presents how the airport workforce scheduling process is divided into four stages and introduces the planning activities during each planning stage. A planning model is constructed with three modules (shift planning, rostering, and task scheduling), based on which airport workforce scheduling can be performed. These three modules are further studied separately as three optimization problems in this research.

CHAPTER 4. A 2D GA FOR LARGE-SCALE SHIFT PLANNING CONSIDERING DAILY-WISE SHIFT FORMATS

Owing to the computational efficiency in dealing with combinatorial optimization problems, genetic algorithms (GAs) have been widely applied to human resource planning and workforce scheduling. Shift planning is of particular importance for personnel scheduling when practical concerns must be taken into account. Daily-wise shift formats are often introduced in practical operations to facilitate execution of the planned tasks and accommodate certain managerial convenience. However, the highly repetitive nature of running daily-wise shift formats entails an extreme imbalance of set covering between the tasks and staff availability, which leads to tremendous computational challenges in solving the combinatorial optimization problem that is subject to large redundancy of zero elements. In line with the inherent two-dimensions of shift planning in terms of shift formats and days, this chapter proposes a two-dimensional (2D) encoding scheme to implement the GA for efficient shift planning. An application to a real-life airport Ground Staff Scheduling (GSS) problem is presented to illustrate the feasibility and potential of the proposed 2D GA for efficient handling of daily-wise shift formats.

4.1 Shift Planning

Shift planning is the initial step during workforce scheduling, and it aims to design a set of shifts with different start time and duration to cover the demands within the planning horizon. It has been widely studied to improve the workforce utilization ever since Dantzig proposed the first mixed integer programming model for shift planning [43]. The challenge of shift planning problems lies in the mathematical modeling for the shift design requirements and the large scale of the optimization problem.

Metaheuristics are suggested to be critical methods for coping with such combinatorial optimization problems, especially GAs. Most of the existing encoding methods such as binary encoding and real-value encoding are one- dimensional (1D), whereby the highly repetitive daily-wise shift formats will result in a large redundancy in shift matrix after instantiation in the context of shift planning. Several 2D encoding methods have been applied to represent the intrinsic 2D problems include spin grids in Ising model [44], job-machine scheduling [45], orthogonal packing problem [46], to name but a few. The high repetitiveness of the daily-wise shifts implies an intrinsic 2D structure of the shift matrix, which in terms of days and daily timeline. This 2D structure can effectively reduce the redundancy of the shift matrix, by squeezing out the zero elements significantly. However, little research has shed light on the 2D encoding and calculation for 2D GA in the context of shift planning.

The remainder of the chapter is organized as follows. Section 4.2 formally introduces the variable definition and formulates the mathematical model of the problem. Section 4.3 proposes a 2D GA method for large-scale shift planning, including chromosome encoding and the corresponding matrix operations. Section 4.4 and 4.5 give a detailed description of crossover and mutation operators for 2D chromosome. Computational experiments are shown in Section 4.6 for the validation of the proposed 2D GA. The result analysis is elaborated in Section 4.7 and the summary of this research topic is given in Section 4.8.

4.2 Set Partitioning Problem Formulation and Methodology

The solution to the shift planning problem is to explore the combination of shifts and to cover the demands based on trade-off rules while avoiding the violation of constraints from the regulatory and operational perspectives [43]. The objective should be set to minimize the operation cost while avoiding insufficient workforce. However, the set covering formulation cannot satisfy the prevailing shift planning requirements well. As the personnel management level goes higher, on the one hand, a small number of workforce deficiencies during peak hours and midnight is acceptable in order to reduce the degree of excess in workforce; on the other hand, workforce excess for some critical tasks is necessary as a backup. Considering above factors, the research of the balanced relationship among workload, understaffing, and overstaff in shift planning is widely studied [7].

The formulation of optimal shift design problems starts with the modeling of the timeline. The time of a planning horizon is represented by N, whose unit is a day. This planning period N is discretized as T equally sized time slots, $t = 1,2, \dots, T$. The length of each time slot is named as time granularity a_g, whose unit is minute. The workforce demand is represented in a 2D matrix $D \in \mathbb{N}^{|\Phi| \times T}$, where the row φ represents different kinds of requirements, while the column is the timeline in the planning period. The element of the demand matrix is natural numbers and describes the number of the required human resource units. The total amount of the required category is defined by $|\Phi|$. The shift is represented by s, while all s is collected in the shift set S. The shifts are distinguished by the start time p_s and end time q_s, thus a shift is in the time interval $[p_s, q_s]$. The length of the shift s is l_s, which is defined as $l_s = q_s - p_s$, $p_s, q_s, l_s \in \mathbb{N}$.

However, since the intrinsic daily-wise shift repetitiveness, the start time and end time is not varying on each day. The shift format is introduced to simplify the representation of the shifts. One shift format should represent all the shifts which start from the same time on different days in a function $s'(n)$, and all $s'(n)$ are collected in the shift format set S'. The shift is the instantiation of a shift format with an input of specific Day n:

$$s'(n) = s \tag{4.1}$$

The mixed integer programming model of the shift planning for airport ground staff can be formulated as a set partition model as following:

$$z^* = min\, \alpha_c \sum_{s \in S} c_s x_s + \alpha_u \sum_{t=1}^{T} \sum_{\varphi \in \Phi} u_{\varphi t} + \alpha_o \sum_{t=1}^{T} \sum_{\varphi \in \Phi} o_{\varphi t} \tag{4.2}$$

$$s.t. \sum_{s \in \mathrm{S}} a_{st} x_s m_{s\varphi} + \sum_{\varphi \in \Phi} u_{\varphi t} - \sum_{\varphi \in \Phi} o_{\varphi t} = D_{\varphi t} \;\; \forall \varphi, 1 \leq t \leq T \tag{4.3}$$

$$\sum_{s \in \mathrm{S}} v_{is} x_s \leq V_i, \quad \forall V_i \in V \tag{4.4}$$

$$u_{\varphi t}, o_{\varphi t}, x_s \in \mathbb{N} \tag{4.5}$$

$$a_{st} \in \{0,1\} \tag{4.6}$$

The objective function (4.2) is a weighted sum of shift operation cost, understaffing cost, and overstaffing cost, using weight factor α_c, α_u and α_o, respectively. The shift operation cost is the sum of the cost of the specific shift c_s multiply the number of this shift

x_s. The overstaffing cost and understaffing cost are calculated by summing up the number of workforce deficiency and workforce excess of all requirements along the timeline, respectively. Constraint (4.3) describes the staff equilibrium of all requirements must agree at any time along the timeline. This staff equilibrium requires the sum of available staff, and workforce deficiency must equal to demands after subtracting the workforce excess. The operational variable $a_{st} = 1$ describes shift s is active at time t and $a_{st} = 0$ describes shift s is inactive at time t. Similarly, operational variable $m_{s\varphi} = 1$ describes shift s can fulfill requirement φ, while $m_{s\varphi} = 0$ describes shift s cannot fulfill requirement φ. Constraint (4.4) is the number constraints of the specified shift category, where $v_{is} = 1$ means shift s belongs to category i. Shift category limit V_i specifies the maximum number of a particular shift category, and all V_i are collected in V. Constraint (4.5) states that the $u_{\varphi t}$, $o_{\varphi t}$, and x_s are nonnegative integers.

To solve the shift planning problem formulated above, the existing typical heuristic approach is the 1D GA. As the approach name suggests, the chromosome $[\mathbb{C}]_{1\times|S|}$ is encoded as a 1-by-$|S|$ matrix. Every gene represents the number of the corresponding shift in S. Following the similar index, the shift matrix $[\mathbb{S}]_{|S|\times T}$ is a $|S|$-by-T binary matrix, whose element is equal to a_{st} in (4.7). The number of active working resources within the planning horizon is represented in the working matrix $[\mathbb{W}]_{1\times T}$, which can be calculated by:

$$[\mathbb{C}]_{1\times|S|} \cdot [\mathbb{S}]_{|S|\times T} = [\mathbb{W}]_{1\times T} \tag{4.7}$$

4.3 Two-Dimensional Genetic Algorithm

The above GA has some drawbacks in the current problem context. Firstly, there is much redundancy in the shift matrix. The timeline of the matrix is the planning horizon, while the actual active time of a shift is less than a day. Thus, most of the elements are redundant zeros during inactive days, and their existence dramatically reduces the computational efficiency of calculating (4.7). Secondly, the prevailing 24-hour operation challenges the daily-wise shift planning problem decomposition, since the 24-hour operation implies overlapping of shifts in two adjacent days. The overnight shifts in 24-hour operation may start from the previous day but end later than the earliest potential shifts in the following day. The other daily-wise decomposition approach is separating the days before the earliest shift. However, the daily requirement start time varies from day to day. Thus, presetting a fixed time point to separate working days may tear apart the successive early morning requirement peaks. Thirdly, the 1D chromosome structure limits the efficiency of crossover and mutation operations, since the intrinsic 2D property of the shift planning problem. Because of the linear structure organizes the genes in one dimension, the simultaneously segmenting chromosomes from different dimensions is hard to implement.

To overcome the drawbacks caused by the encoding structure, the 2D GA is proposed to describe the problem in a new perspective. As is mentioned above, the shift format describes a set of shifts that have the same daily start time and end time in the shift set S. Due to the high repetitiveness of daily-wise shift formats, S can be represented by a set of shift formats with date information. Thus, the chromosome can be constructed from the perspectives of the planning horizon and shift format, encoded as $[\mathbb{C}]_{M\times N}$, a M-by-N matrix where M is the number of shift formats in a day. The gene c_{mn} represents the number of the m_{th} shift format on Day n. Correspondingly, the original shift matrix is changed into the shift format matrix $[S']_{M\times T_{24}}$, a M-by-T_{24} binary matrix where T_{24} is the number of time slots in 24 hours. Contrary to the shift matrix in the 1D GA approach, the timeline of S' is 24 hours. Every row of the matrix represents the active timeslots of a shift format for one day, which is $\frac{1}{N}$ smaller by eliminating insignificant zeros in the inactive days of that shift. On the other hand, due to the high repetitiveness, the number of shift formats is also $\frac{1}{N}$ smaller than $|S|$. Thus, the shift format matrix $[\mathbb{S}']_{M\times T_{24}}$ is $\frac{1}{N^2}$ smaller than the shift matrix $[\mathbb{S}]_{|S|\times T}$, greatly improving the computational efficiency for the calculation of working resources. The number of working resources within the planning horizon is represented in the working matrix $[W]_{N\times T_{24}}$, which can be calculated by:

$$[\mathbb{C}]_{M\times N}{}^{T} \cdot [\mathbb{S}']_{M\times T_{24}} = [\mathbb{W}]_{N\times T_{24}} \tag{4.8}$$

The chromosome and the shift format matrix should be decomposed to support the 24-hour operation. The chromosome is divided into three categories, $[\mathbb{C}_d]_{M_d\times N}$, $[\mathbb{C}_{n1}]_{M_n\times N}$, and $[\mathbb{C}_{n2}]_{M_n\times N}$, where $\mathbb{C}_d$ and $\mathbb{C}_{n1}$ represent the chromosome for shift formats that do not span two days and overnight shift formats, and M_d and M_n equal to the corresponding number of shift formats. $\mathbb{C}_{n2}$ is the combination of a column of zeros followed by the first $(N-1)$ columns of $\mathbb{C}_{n1}$. Correspondingly, the shift format matrix is divided into three categories, $[S'_d]_{M_d\times T_{24}}$, $[S'_{n1}]_{M_n\times T_{24}}$, and $[S'_{n2}]_{M_n\times T_{24}}$, where S'_d represents the shift formats that do not span two days, S'_{n1} represents the parts of overnight shift formats before midnight, and S'_{n2} represents the parts after midnight. The working matrix $[W]_{N\times T_{24}}$ can then be calculated by:

$$[\mathbb{C}_d]_{M_d \times N}{}^T \cdot [\mathbb{S}'_d]_{M_d \times T_{24}} + [\mathbb{C}_{n1}]_{M_n \times N}{}^T \cdot [\mathbb{S}'_{n1}]_{M_n \times T_{24}}$$

$$+[\mathbb{C}_{n2}]_{M_n \times N}{}^T \cdot [\mathbb{S}'_{n2}]_{M_n \times T_{24}} = [\mathbb{W}]_{N \times T_{24}}. \quad (4.9)$$

The step-by-step process of 2D GA is described in Table 4.1. The flowchart is displayed in Figure 4.1.

Table 4.1 Pseudocode of 2D Gas.

// Initialization of generation 0:
$ite \coloneqq 0$;
P_{ite} ≔initial population of p randomly generated $[\mathbb{C}]_{M \times N}$ individuals;
// **Evaluate the fitness value of the populations**
Compute $fit(i)$ for $i \in P_{ite}$;
while $ite < maximum\ generation$
do
// **Create generation** $ite + 1$:
Select a proportion of members from P_{ite};
Select another proportion of members to for crossover;
Combine the selection and offspring;
Mutate the combined set;
Compute $fit(i)$ for $i \in P_{ite+1}$;
$ite \coloneqq ite + 1$;
if the fitness of the best individual is converged
return best individual
end
end

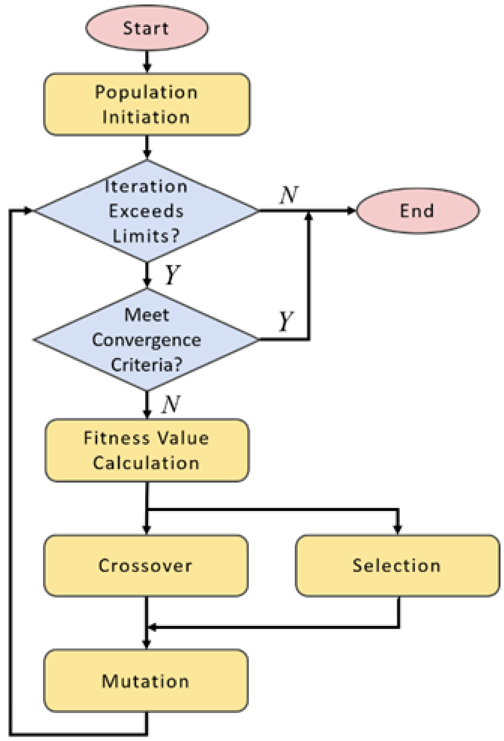

Figure 4.1 2D GA Flowchart.

4.4 Crossover Operators

The crossover operations aim to exchange part of the parents' chromosomes for producing offspring. The selected method is substring crossover, which randomly chooses the exchanging points and arbitrarily exchanges the substring. The crossover operations of 2D chromosome can divide the parent chromosome from the exchanging point horizontally and vertically. The crossover operations can be summarized in the following four steps.

The first step is selecting the parent chromosomes p_1 and p_2 based on their fitness value by applying the roulette algorithm. A fitter individual shows a superior probability of being selected. The second step is generating a random number and compare it with a preset constant, which is set as 0.5 in this research. If the random number is greater than 0.5, the crossover will be done horizontally. Otherwise, a vertical operation will be applied.

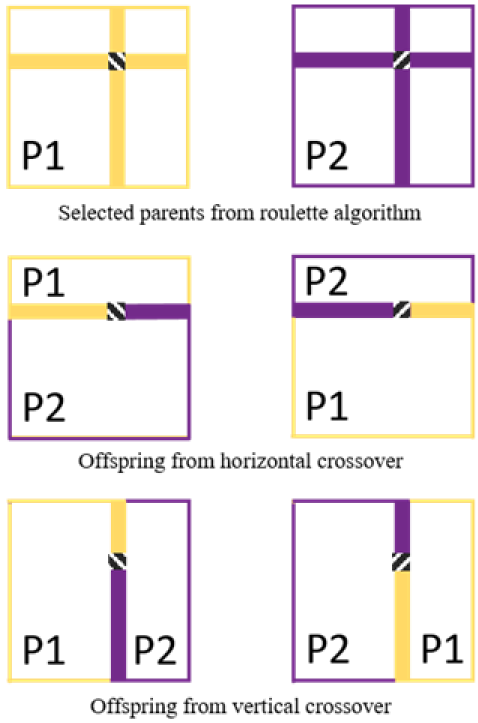

Figure 4.2 Crossover Operations.

The horizontal and vertical crossover will be based on the randomly generated exchange point indexes R_c and R_r. R_c represents the selected column index, while R_r represents the selected row index. In the horizontal crossover, all the row after the R_r will be exchanged. Meanwhile, at the row R_r, elements after R_c will be exchanged. The vertical crossover is following a similar logic. The columns after the R_c will be exchanged. At the column R_c, elements after R_r will be exchanged. The generated two offspring will have elements from both parents. Figure 4.2 shows an example of the crossover operations.

4.5 Mutation Operators

The mutation operations are applied to introduce diversities to the populations. The 1D mutation operations usually randomly change one bit from an arbitrary position. While the proposed 2D method has applied two mutation approaches, which are named as swapping and perturbation, respectively. This kind of mutation operations will be conducted when a newly generated random number is smaller than the mutation rate, which is a preset constant. Moreover, the determination of swapping or perturbation is based on the approach selection rate, which is also a preset constant. If the swapping approach is selected, two pairs of indexes (R_r, R_c) will be randomly generated, and the corresponding two elements in the chromosome will be exchanged. Otherwise, the perturbation operations will regenerate a random value on the randomly selected position (R_r, R_c). This kind of mutation operations provides an approach to balance the introduction of diversity to the population and the prevention of contamination in the late searching stage. Figure 4.3 shows an example of mutation operations.

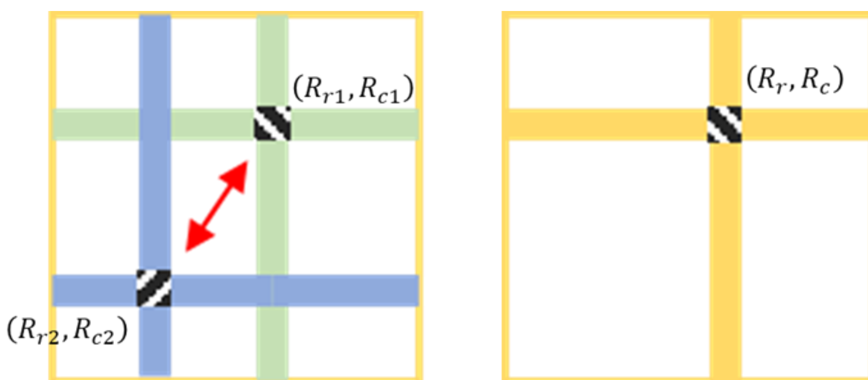

Figure 4.3 Mutation Operations.

4.6 Computational Experiments

The case is based on a real-world shift planning problem at a large airport. The case is implemented by MATLAB 2019a, under Win 10 pro operation software. The CPU is Intel Xeon CPU E3-1505M v5 @ 2.90GHz. The planned shifts and the coverage of the demands of a 7-day shift planning problem are shown in Figure 4.4. The time granularity is 1 minute. In everyday demand, there are two peaks, which are morning and evening rush hours, respectively. Figure 4.5 shows the coverage details on the sixth day. The result has shown that most of the demands have been covered. The fitness function value of the fittest child in every generation is plotted in Figure 4.6. This plot is to show the convergence of the algorithm. Figure 4.6 suggests it takes around 200 iterations to converge.

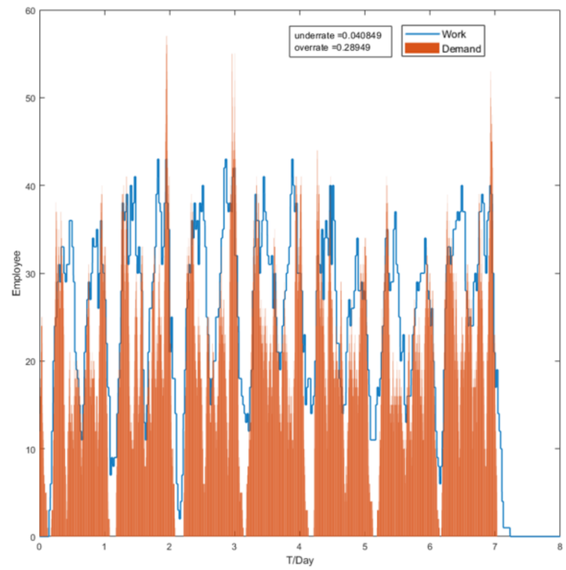

Figure 4.4 Shift coverage result of a 7-day shift planning problem.

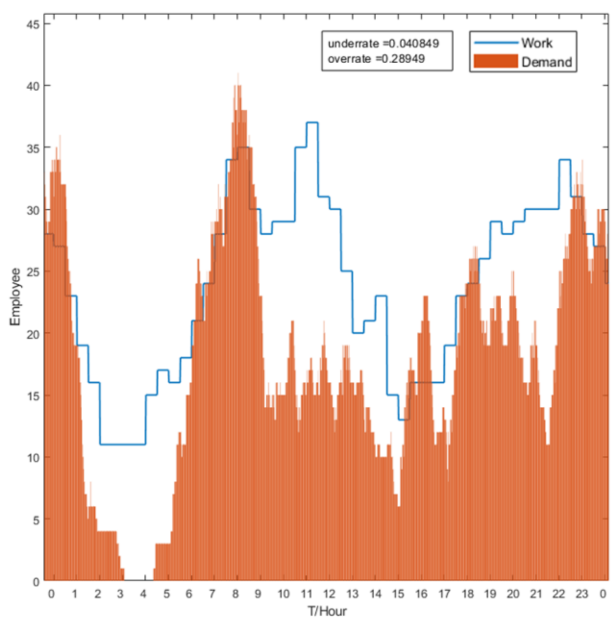

Figure 4.5 Shift coverage in one day span.

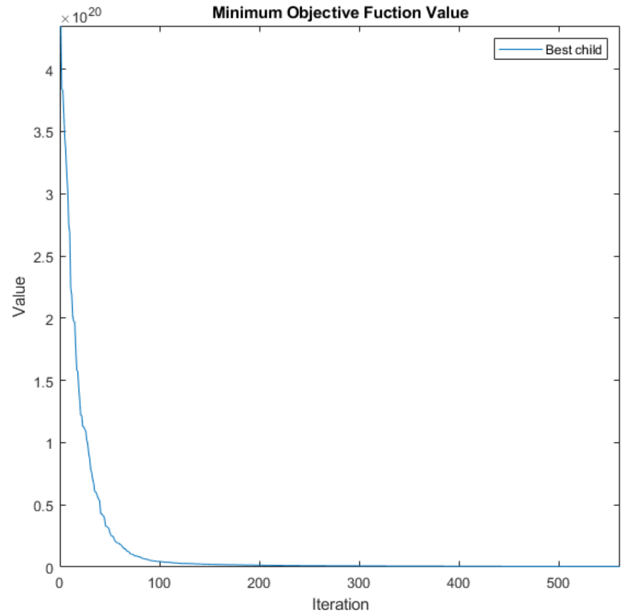

Figure 4.6 Objective function value and iterations.

4.7 Results and Analysis

In the last section, experiments of 7-day shift planning are conducted to validate the proposed 2D GA. Three experiment responses are used as performance measure: the understaffing rate (the ratio of uncovered demands over the sum of total demands), the overstaffing rate (the ratio of excess workforce over the sum of total demands), and the computing time. As is seen in Figure 4.4, the combined shifts cover most of the demands: the understaffing rate is 4.09%, with some peaks uncovered during morning and evening rush hours, while the overstaffing rate is 28.94%. The computing time is 65.5 seconds, which is much smaller when compared to that of 1D GA (1220.4 seconds). The results are acceptable according to the company that sponsors this research. On the one hand, the overstaffing rate is below 30%, which is often applied as a criterion in most airlines. On the other hand, the results cover most of the demands except some peaks, and the coverage of those peaks will bring much workforce excess and makes the overstaffing rate much larger.

4.8 Chapter Summary

Applying GAs on shift planning has become increasingly critical for practitioners and researchers since they can provide a nearly optimal result in an acceptable time. The 1D GA encoding schemes have limited the application to the large-scale shift planning problems, which leads to a high redundant shift representation and inefficient chromosome operations. In this regard, this chapter proposes a 2D encoding scheme as well as the corresponding computing approach and operations to serve the introduction of shift formats. This scheme has improved the algorithm computational efficiency by squeezing out the redundant zero. The transform of the chromosome from 1D to 2D allows easier crossover operations by segmenting the parent chromosomes in two dimensions. The corresponding computation approach can support efficient matrix operations and a 24-hour operation paradigm. Finally, an application case of a 24-hour airport GSS scheduling problem from the real world has been demonstrated to examine the efficiency and potential of proposed 2D GA in coping with large-scale shift planning problems with shit formats.

CHAPTER 5. ROSTERING WITH SPECIFIC WORK PATTERNS USING COLUMN GENERATION

Different from shift planning problems, the rostering problem is to combine different types of shifts and obey a certain work pattern to form a roster line. Rostering algorithms need to face the challenge of the combinatorial explosion and have a large impact on the staff utilization rate in airlines. This chapter addresses the rostering problem with work patterns. A set covering model is formulated to model the relationship between formed roster lines and required shifts. A branch and price algorithm is developed, using the label correcting algorithm for column generation to solve the shortest path problem with restricted constraints. Cases from real-life airlines are studied to validate the developed algorithm. Computational results are analyzed to verify the feasibility.

5.1 Rostering

Workforce scheduling aims to assign workers with a line of specified shifts with day-off information and is often regarded as a multi-stage planning process [47]. Rostering operations are conducted during the mid-term of the planning timeline and happen after shift planning is finished [31]. It concerns with minimizing the operation cost while satisfying the requirements of different shifts needed during the planning horizon. Because rostering is to form a roster line with different combinations of shifts, even one more shift than needed can bring large workforce excess. Thus, the quality of rostering operations has a great impact on workforce scheduling results.

The rostering problem can be formulated as a set covering problem, which is solved by forming rostering lines using the shifts existing in the shift planning result to cover the shift demands while satisfying specific work patterns and avoiding violating the labor law and the employment rules [48]. Some definitions are given in Table 5.1 to better understand the rostering problem. Examples of shift demands, a roster line, and a roster are given in Table 5.2, Table 5.3, and Table 5.4.

Table 5.1 Rostering term explanation.

Term	Explanation
Shift demand	Number of different required shifts from the shift planning results
Work pattern	The minimum period that specifies the sequence of the days of work and required days of rest, sometimes certain shift types are specified on some workday, which is often seen in nurse scheduling problem [49]. (For example, M-A-O means working on a morning shift on the first day, an afternoon shift on the second, and off on the third day.)
Roster line	A line of specified shifts on the planning horizon [38]
Roster	A set of roster lines for needed employees on the planning horizon
Rostering	The process of selecting roster lines to cover the shift demands based on the labor law and employment rules

Table 5.2 An example of shift demands.

Shifts	A_1	A_2	P_1	A_3	P_2	P_3	A_4	A_5	P_4	A_6	A_7	P_5	A_8	P_6	P_7	A_9	A_{10}	P_8
Days	1	1	1	2	2	2	3	3	3	4	4	4	5	5	5	6	6	6
Number	5	7	13	15	8	6	7	9	14	5	8	16	19	9	10	7	4	12

Table 5.3 An example of a 6-day roster line.

Day 1	Day 2	Day 3	Day 4	Day 5	Day 6
A_2	P_3	O	A_6	P_6	O

Table 5.4 An example of a 6-day roster.

Employee type	Number	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6
1	3	A_2	P_3	O	A_6	P_6	O
2	5	P_1	O	A_4	P_5	O	A_9
3	4	O	A_3	P_4	O	A_8	P_8

The challenge of the rostering problem lies in two aspects. On the one hand, to find feasible roster lines with specified work patterns while satisfying constraints from the labor law and employment can be formulated as a shortest path problem with resource constraints, which is NP-hard in the strong sense [50]. On the other hand, as the planning horizon grows linearly, the number of feasible roster lines grows exponentially, making it impractical to enumerate every combination to solve it directly, especially when the number of everyday shifts is large. To solve this problem, the branch and price algorithm using column generation is often applied [2].

Column generation is the key idea due to the characteristics of the rostering problem. Because the feasible roster lines are numerous and enumeration is not possible, also it is not easy to find sets of feasible roster lines, column generation can be used to generate high-quality feasible lines by computing the reduced cost, and the combinatorial problem is left with the master problem. This greatly improves the algorithm efficiency by only using a set of effective lines.

In this regard, this chapter is organized as follows. Section 5.2 presents formal definitions of basic variables and the MIP model for the rostering with work patterns problem. Section 5.3 proposes the column-generation-based approach and it is divided into two parts, corresponding to the two optimization problems decomposed from the original one. The first part introduces the master problem, and the second part presents the subproblem with one dynamic programming solution. Section 5.4 provides the branching scheme to guarantee the feasibility of the solution. Section 5.5 presents the computational experiments using a rostering case from a real-life airline. Results are discussed and analyzed in Section 5.6. Finally, the summary of the research on rostering with work patterns is given in Section 5.7.

5.2 Set Covering Problem Formulation

The rostering problem entails an exploration of selecting a set of roster lines to cover the shift demands while avoiding the violation of constraints from the labor law and the employment rules. The goal is to minimize the operation cost while satisfying the requirements on the number of different shifts. Because lacking one shift results in large workforce deficiency, the number of an arranged shift in this research is set to be no smaller than the corresponding required number.

The inputs of the problem include the work pattern, employment rules, and the shift planning result. The work pattern describes the required day-off sequence from the management perspective. Sometimes the work pattern specifies the time period of the shift on some day. For example, shifts in a day can be classified as three kinds: morning shifts with starting time from 04:00 to 12:00, afternoon shifts starting from 12:00 to 18:00, and night shifts starting from 18:00 to 04:00. Thus, a work pattern M-M-A-N-O-O represents the sequence of two morning shifts, one afternoon shift, one night shift and two offs. Employment rules define the feasibility of a roster line with constraints on some parameters: the minimal time between two shifts, the planning horizon, minimal working time and maximal working time in a week. The shift planning result contains a set of shifts with corresponding properties: the work date, start time, end time, and the required number.

The challenge for the mathematical model is typically how to generate a feasible roster line with high qualities. Because the rotation of the work pattern (like $M - A - O$, A-O-M, and O-M-A) and various constraints on the shift arrangement, it is nearly impossible to enumerate all the feasible roster lines for practical use. In the meantime, compared to the number of total possible combinations, only a very small number will be used in the solution. Thus, column generation is considered to flexibly generate effective roster lines to the problem.

The integer programming model for generalized rostering problems can be formulated as a set covering model described in the following:

$$min \sum_{r \in R} c_r x_r \tag{5.1}$$

$$s.t. \sum_{r \in R} a_{sr} x_r \geq q_s, \forall s \in S \tag{5.2}$$

$$\sum_{r \in R} a_{gr} x_r \le m_g, \forall g \in G \tag{5.3}$$

$$\sum_{r \in R} x_r \le n \tag{5.4}$$

$$x_r \in \mathbb{N} \tag{5.5}$$

$$a_{sr}, a_{gr} \in \{0,1\} \tag{5.6}$$

Since the rostering process is after the shift planning and airport staff has a high understaffing cost while a low labor cost, the objective function (5.1) is set to be the sum of labor cost. The shift planning result is identified by the shift set S containing every shift used s. The roster line set is represented by R, and each roster line r in the set is a $|S| \times 1$ binary matrix, in which each element suggests whether the corresponding shift is chosen in that roster line. The labor cost of a specific roster line is represented by c_r, which is calculated by the total working hours of that roster line. The shift demand requirement is described in Constraint (5.2). a_{sr} indicates whether a shift s is chosen in a roster line r, with $a_{sr} = 1$ meaning the shift belongs to this roster line and 0 meaning the shift is not chosen. x_r refers to the number of a roster line in the solution. The corresponding requirement for the number of a shift is $q_s, \forall q_s \in Q_s$. The limitation on the work pattern groups is represented in Constraint (5.3). A pattern group g is one variation of the specified work pattern, and $\forall g \in G$. a_{gr} indicates the pattern group that the roster line belongs to. The maximum number of a pattern group is represented by m_g. This constraint is often used to better manage employees working on the same work pattern. For example, a roster line M -O-M belongs to a pattern group W-O-W (work-off-work) and a_{gr} equals to the number that represents the pattern group and corresponding m_g is the maximum number of roster lines on W-O-W. Constraint (5.4) indicates the limitation on the number of employees, and Constraint (5.5) guarantees the integrality of the solution.

5.3 Column Generation for Rostering with Work Patterns

For some combinatorial optimization problems, it is nearly impossible to explicitly enumerate all feasible combinations. Such problems are often solved by the column generation approach, which is embedded into the branch and bound algorithm. The rationale of the column generation technique is similar to the simplex method. Due to the fact that the non-basic variables of the solution are in the majority, only the variables that have the potential to make the solution performance better should be considered. Thus, this approach decomposes the problem into two related optimization problems: the master problem that measures the performance of the current solution and the pricing problem (also referred to as the subproblem) that continuously generates high-quality and feasible combinations for the master problem. Figure 5.1 represents the flowchart of the approach.

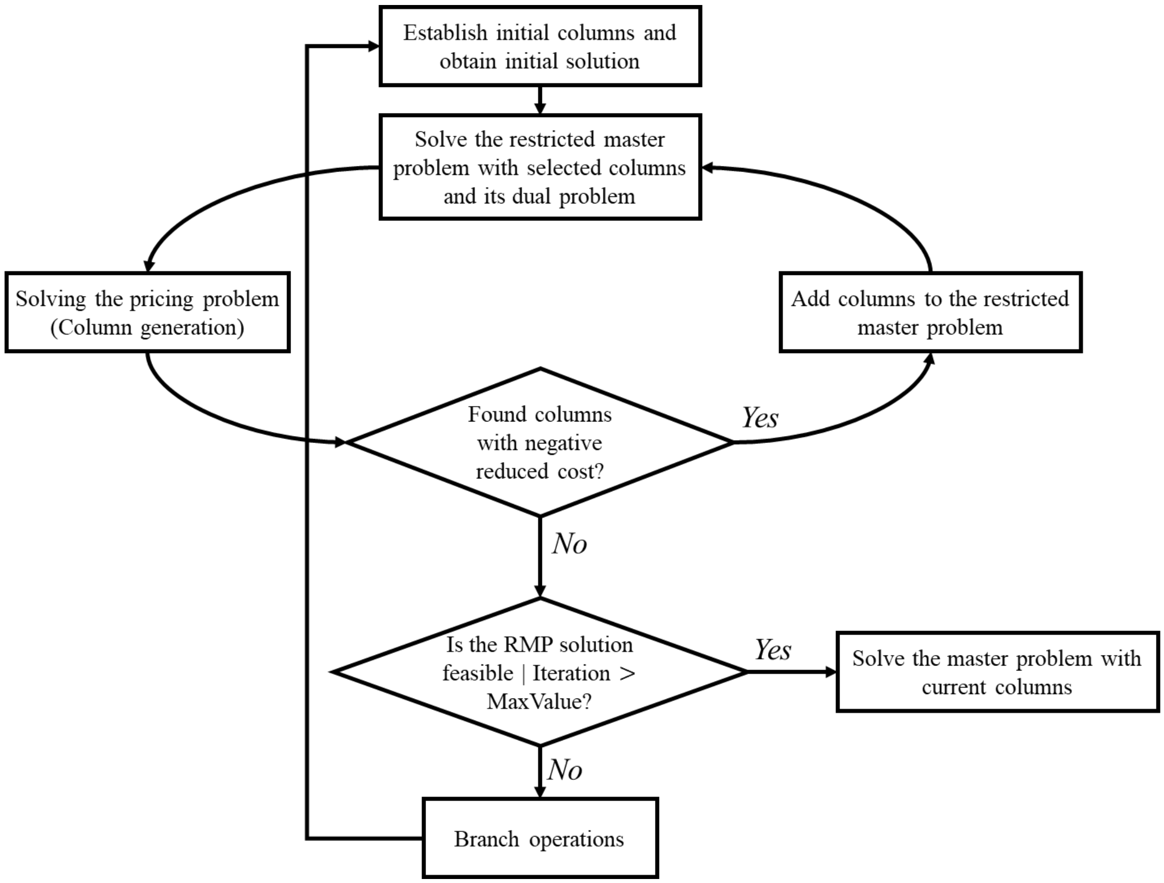

Figure 5.1 The flowchart of branch and price algorithms.

5.3.1 The Master Problem

The master problem aims to combine existing roster lines to satisfy the shift demands while avoiding the violation of the pattern group limitations. Thus, the variables used in the master problem is not the universal set but only a subset. The new variables (roster lines) will continuously generated by applying column generation until no more roster lines that have negative reduced costs can be found. However, the integrality requirement on x_r needs to be relaxed in order to embed column generation into the branch and bound [38]. Thus, Constraint (5.7) is replaced by the following formula:

$$x_r \in \mathbb{R}_+ \tag{5.7}$$

The mathematical modeling on the objective function and other requirements remain the same as in the problem formulation, and this revised problem is called the restricted master problem (RMP). RMP and its dual problem are then be used to measure the performance of current columns and generate new high-quality columns.

5.3.2 The Pricing Problem

The objective of the pricing problem is to generate high-quality variables that have the potential to improve the solution performance based on the current variables until no such variables can be found. This is done by exploring the variables with minimum negative reduced costs using the dual solution of the current restricted master problem. The pricing problem is formulated in the following:

$$min_{r\in R}\left(c_r - \sum_{s\in S} a_{sr}\pi_s - \sum_{g\in G} a_{gr}\mu_g - \delta\right) \tag{5.8}$$

In the objective function (5.8), the shift combination of a roster line is represented by the matrix of a_{sr} and the pattern group a_{gr}. A roster line must satisfy all the labor law and the employment rules, which contain the following: Employees can work no more than one shift per day; Two consecutive shifts of a roster line has a required minimum time interval; The day-off sequence of the roster line must correspond to the pattern group it belongs to. π_s, μ_g, and δ are dual variables for shift number constraints, pattern group constraints, and total employee number constrains correspondingly. Because of the commonality on the pattern group and the number of employees, the pricing problem can be simplified as follows:

$$min_{r \in R}\left(c_r - \sum_{s \in S} a_{sr}\pi_s\right) \tag{5.9}$$

This pricing problem in this context can be formulated as a shortest path problem with resource constraints (SPPRC). In SPPRC, each shift is a node with the dual variable value as its cost. A feasible path is to combine the shifts and off days based on the rules of connectivity and the total working hour constraints. The objective is to find the path with the minimum negative reduced cost. Usually this is solved by the label correcting algorithm based on the concept of dynamic programming. The basic idea is to set the labels for each shift and track them while extending them to the connectable shifts and off days through the graph. After visiting a node, labels on that node will be compared and the unpromising ones will be discarded though they are feasible. Those left labels will continuously be extended to the destination and form a feasible path. Paths with minimum negative costs are then chosen and the corresponding variables will enter the basis of the RMP.

Some basic definition in the label correcting algorithm is given in Table 5.5.

Table 5.5 Basic definition in the label correcting algorithm.

Variable/Function	Definition
s	Each shift is encoded as a node on the graph

λ	A non-dominated label
Λ	A list of non-dominated labels
$succ(s)$	The set of successor shifts of some shift
$Extend(\lambda_i, s_j)$	Function of extending labels from λ_i to s_j
F_{ij}	The set of labels extends from s_i to s_j
$EFF(\Lambda)$	Function of discarding dominated labels in Λ

Pseudocode of the label correcting algorithm is presented in Table 5.6.

Table 5.6 Pseudocode of the label correcting algorithm.

```
// Initialization
Λ_p ← ∅
For all s_i ∈ S\{p}
    do Λ_i ← ∅
End for
E ← {p}
// Label correcting
While E ≠ ∅
    pick s_i ∈ E
    for all succ(s_j)
        do F_ij ← ∅
        for all λ_i ∈ Λ_i
                F_ij ← F_ij ∪ Extend(λ_i, s_j)
        end for
        Λ_j = EFF(Λ_i, F_ij)
        if Λ_j is changed
                E ← E∪s_j
        end if
    end for
    E ← E\{s_i}
End while
```

5.4 Solution Integrality

One problem that remains to be solved is the solution feasibility. Due to the relaxation on x_r in the RMP, the solution of RMP might be fractional, which makes no sense in practical problems. Thus, the branching strategy is needed to fathom the solution with decimal part.

The idea to apply variable fixing every time the RMP is solved. The branching strategy in this research adopts a greedy method [47]. Because of the computational cost to bound variables with the upper bound, the strategy only considers doing with lower bounds.

Assume the optimal solution to one RMP is x^*. If $x_r^* \epsilon \mathbb{Z}_+, \forall r \in R,$ then x^* is the optimal solution that is feasible to the original formulation. Otherwise, the fractional part $f_r^* = x_r^* - \lfloor x_r^* \rfloor$ of every variable value is taken to bound the variables with a predefined threshold τ, where $\tau \in (0,1)$. The following strategy is taken:

$$x_r \geq \lceil x_r^* \rceil, \forall r \in R: f_r^* \geq \tau \qquad (5.10)$$

However, if $f_r^* < \tau$ for every $r \in R$, then the variable with the largest fractional part will be rounded up:

$$\text{if } r \in R: f_r^* \geq \tau = \emptyset, \text{ then } x_r \geq \lceil x_r^* \rceil, r = argmax_{x_r} f_r^* \qquad (5.11)$$

5.5 Computational Experiments

The experiment is a case based on a real-world rostering problem. The objective is to generate a 7-day rostering. The input is from a 7-day shift planning result, as shown in Table 5.7, where the shift date, start time, and duration (represented in minutes), and the required number are given for each type of shift. The work pattern is set as W-W-W-W-W-O-O, and the interval between two shifts in one roster line must be larger than 10 hours. The minimum weekly working hour is 40 and the maximum is 45. The experiment result is shown in Table 5.8, indicating the shifts with specified date, start time and duration of each roster line. Figure 5.2 represents the shift coverage of the current rostering solution on each day. The result shows that all shifts are covered and there are three superfluous shifts, two on the fifth day and one on the last day of the rostering horizon.

Table 5.7 Inputs from shift planning results

Day	Start (min)	Duration (min)	Num	Day	Start (min)	Duration (min)	Num	Day	Start (min)	Duration (min)	Num
1	210	510	1	3	450	480	2	5	900	480	2
1	240	480	2	3	810	480	3	5	930	480	1
1	330	480	1	3	840	480	1	6	240	480	2
1	360	480	3	3	870	480	2	6	330	480	1
1	390	480	1	3	930	480	2	6	360	480	3
1	450	480	1	3	1050	480	1	6	420	480	2
1	810	480	2	4	240	480	1	6	450	480	1
1	840	480	4	4	330	480	2	6	810	480	4
1	870	480	1	4	390	480	5	6	900	480	3
1	900	480	2	4	420	480	2	6	930	480	2
2	210	600	1	4	810	480	1	6	1050	480	1
2	240	600	1	4	840	480	3	7	240	510	1
2	330	480	1	4	870	480	4	7	330	510	2
2	360	480	1	4	1020	480	2	7	360	510	1
2	390	480	1	5	240	540	1	7	450	480	2
2	450	480	2	5	240	600	1	7	480	480	1
2	810	480	2	5	330	510	1	7	510	510	1

2	870	480	4	5	360	480	1	7	870	480	2
2	930	480	3	5	390	480	1	7	900	480	1
3	210	510	1	5	420	480	1	7	930	480	2
3	240	480	1	5	450	480	1	7	1050	510	3
3	330	510	2	5	810	480	3				
3	360	480	3	5	870	480	2				

Table 5.8 Results of a 7-day rostering with a 5&2 work pattern

Day	Start (min)	Duration (min)	Day	Start (min)	Duration (min)	Day	Start (min)	Duration (min)	Day	Start (min)	Duration (min)	Day	Start (min)	Duration (min)
1	240	480	4	390	480	5	390	480	6	900	480	7	1050	510
1	840	480	4	390	480	5	810	480	6	900	480	7	900	480
1	900	480	4	1020	480	5	810	480	6	810	480	7	900	480
1	330	480	2	330	480	3	330	510	4	840	480	7	1050	510
1	360	480	2	240	600	3	360	480	4	870	480	5	810	480
2	210	600	3	360	480	4	240	480	5	330	510	6	240	480
3	360	480	4	330	480	5	360	480	6	360	480	7	450	480
1	810	480	2	930	480	3	930	480	4	870	480	7	930	480
3	810	480	4	870	480	5	870	480	6	810	480	7	1050	510
1	810	480	2	870	480	3	870	480	6	240	480	7	360	510
1	360	480	4	390	480	5	930	480	6	810	480	7	510	510
2	870	480	3	840	480	4	840	480	5	900	480	6	900	480
2	810	480	3	450	480	4	810	480	5	450	480	6	450	480
2	390	480	3	330	510	4	390	480	5	420	480	6	930	480
1	210	510	2	450	480	3	450	480	6	360	480	7	330	510
1	870	480	2	930	480	3	810	480	4	1020	480	7	450	480
1	840	480	2	870	480	3	1050	480	6	420	480	7	930	480
1	840	480	4	330	480	5	450	480	6	930	480	7	870	480
1	840	480	4	420	480	5	240	600	6	420	480	7	240	510
1	360	480	2	810	480	3	810	480	6	360	480	7	480	480
1	900	480	2	870	480	3	870	480	4	840	480	5	900	480
2	930	480	3	930	480	4	870	480	5	870	480	6	1050	480
1	390	480	4	390	480	5	360	480	6	330	480	7	330	510
1	240	480	2	360	480	3	210	510	4	420	480	5	240	540
1	450	480	2	450	480	3	240	480	6	810	480	7	870	480

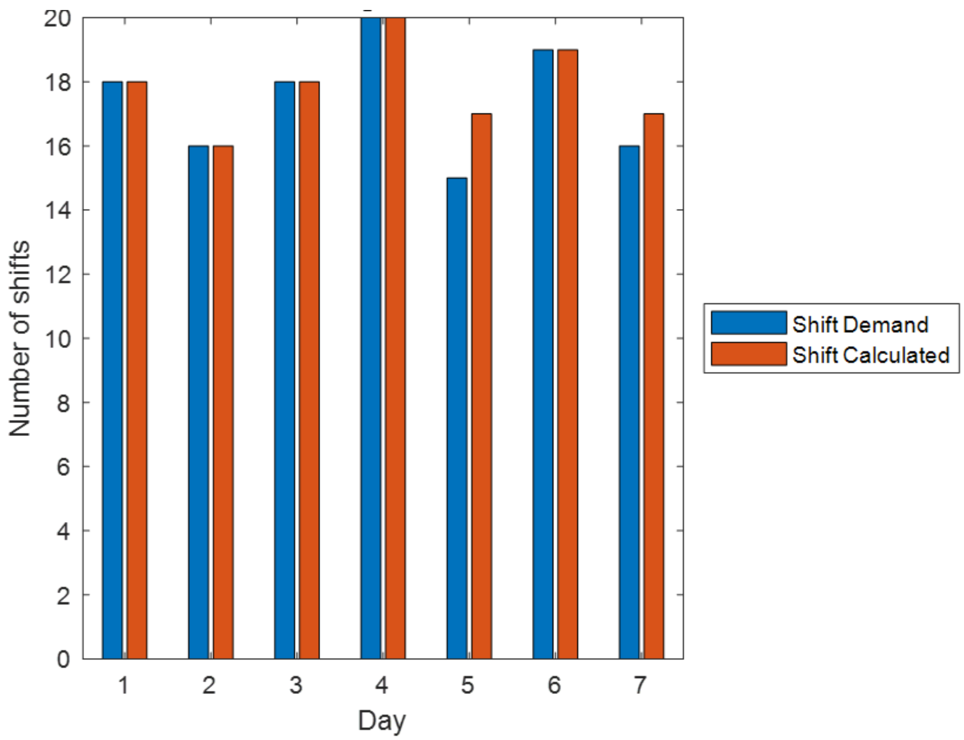

Figure 5.2 Rostering with a 7-day rostering problem with a 5&2 work pattern

5.6 Results and Analysis

Two performance measures are used to evaluate the rostering algorithm performance: the computing time and the number of superfluous shifts. In the experiment above, the computing time is 28.48 seconds with around ten shift types each day, in which most of the time is used on column generation. The required number of shifts to fully cover the demands from the solution is 125, which is 2.5% more than the actual required number. The results are acceptable for the industry application.

5.7 Chapter Summary

Rostering is to assign shifts to different work-off lines based on the specified work pattern as well as labor laws and employment rules. It is challenging to implement not only because it has high requirements for the feasibility of a roster line but also because the combinations are numerous and to enumerate is impractical. This chapter represents the column-generation-based method, which can generate high-quality feasible roster lines by decomposing the original problem into the master problem and the pricing problem. The master problem is a set covering problem, while the pricing problem is solved by using the dual variable values to find routes with minimum negative reduced costs continuously. The pricing problem is a SPPRC and applies the label correcting algorithm based on dynamic programming so that only routes that would improve the solution are chosen instead of enumerating all feasible routes. Finally, an experiment is conducted using a case from a real-world rostering problem to validate the feasibility of the presented algorithm.

CHAPTER 6. TASK SCHEDULING: TASK DISPATCHING AND DISRUPTION MANAGEMENT BY INTEGER PROGRAMMING

Task scheduling happens after roster lines are assigned to every employee, and it is the final step in workforce scheduling. The objective is to assign individual tasks to each shift. Task scheduling contains two kinds of problems: operational problems regarding task dispatching and real-time problems regarding disruption management, which are addressed in this chapter. The mathematical model for task scheduling is formulated and variable management is also introduced. A column-generation-based approach is proposed to solve the task dispatching as a vehicle routing problem with time windows. A decision-making system is developed to analyze the latest task information and to build a MIP model, which can be solved by integer programming solvers. Finally, a case of shuttle bus dispatching from a real-life airline is studied to analyze and verify the feasibility of the proposed approach.

6.1 Task Scheduling

Different from shift planning and rostering, the focus of task scheduling is not on the coverage of workload demands nor the shift assignment, but on the assignment of tasks carrying high-level details like qualification, position, start time, and duration. Task scheduling aims to have tasks scheduled and assigned to employees from the rostering result which indicates the workforce availability during a day [38]. In the task scheduling problem, three perspectives are considered: the allocation of tasks, the scheduling of tasks, and routing of personnel or vehicles within the shift on that day. In other words, an optimal scheduling question that needs to be answered is “who is to finish which task at which time and place” [38]. An efficient task scheduling algorithm can improve the utilization rate of employees and facilities, thus reducing the labor cost and operation cost for the company.

For service industries like airlines, task scheduling can be decomposed into two separate problems: task dispatching that happens before the day of operation and disruption management dealing with the changes that occur during the day of operation [31]. Thus, task dispatching is an operational optimization that puts the solution quality in the first place, while disruption management is real-time optimization, which requires quick reactions to sudden task changes and makes a new arrangement.

According to different task characteristics of airport ground staff, tasks can be decomposed into three types from the task scheduling perspective: passenger services, above wing services, and below wing services. Passenger services handle passenger flow in the airport, and such tasks have neglectable travel distance and are determined based on the passenger arriving patterns. Above wing services deal with aircraft services that have neglectable travel distance with specific limited time windows, like cleaning, catering, and boarding tasks. Below wing services are concerned with aircraft services with long travel distance. Such tasks have not only limited time windows but limited equipment, including shuttle bus driving, tow tractor driving, luggage handling, and so on. Among these tasks, below wing task scheduling is the most challenging for the consideration of limited time windows and equipment. In this regard, this research mainly studies the below wing services.

In the remainder of this chapter, further details of task scheduling are reviewed. In section 6.2, task scheduling problems are introduced. The definition of basic variables is provided, and a set partition model is formulated. A column-generation-based approach is applied. The pricing problem is formulated as SPPTW and is solved by dynamic programming. Section 6.3 presents the disruption management problem. A set partition model is formulated. A decision-making system is developed to implement real-time response by analyzing the latest task information. Section 6.4 presents the computational experiments of both task dispatching and disruption management using a shuttle bus dispatching case from a real-life airline. Section 6.5 further discusses the results and analysis. The summary of task scheduling is given in 6.6.

6.2 Task Dispatching

Task dispatching aims to assign tasks to the assigned shifts from rostering results so that each employee has a specific timetable indicating which tasks to finish with time and location information. It happens after rostering in workforce scheduling and is the last step before the day of operation on the planning timeline. Task dispatching concerns with minimizing the missing tasks and the operation cost calculated by the travel time between tasks while avoiding the violation of labor regulations and time constraints of tasks. When the time span is long and the task number is large, there exist numerous feasible routes to finish tasks within shifts. Also, depending on the travel time between tasks, the utilization rate can vary on a large scale. Thus, it is of great significance to have a high-quality solution for the assignment to improve the employee utilization rate and reduce the operation cost.

6.2.1 Set Partitioning Problem Formulation

The task dispatching problem formulation starts with the modeling of tasks. To efficiently manage needed data, each task is instantiated with five attributes: the place of departure, the place of arrival, in position time, end time, and task active time. Thus, $task_\gamma = (v_{\gamma a}, v_{\gamma b}, a_\gamma, b_\gamma, t_\gamma)$, where γ represents a certain task, while all γ are collected in the task set Γ. v indicates the place, and a, b, t represents the start time, the end time, and the active time.

The integer programming model for task dispatching can be formulated as a set partition model, and it is described as following:

$$\min \sum_{h_k \in \Omega} c_k x_k + c_u \sum_{\gamma \in \Gamma} u_\gamma \tag{6.1}$$

$$s.t. \sum_{h_k \in \Omega} a_{\gamma k} x_k + u_\gamma = D_\gamma, \forall \gamma \in \Gamma \tag{6.2}$$

$$\sum_{h_k \in \Omega} z_{sk} \le s_j, \forall s_j \in S \tag{6.3}$$

$$x_k, u_\gamma \in \mathbb{N} \tag{6.4}$$

$$a_{\gamma k}, z_{sk} \in \{0,1\} \tag{6.5}$$

The objective function (6.1) is the sum of the operation cost and the cost of missing tasks. Each dispatching line is a shift carrying tasks to be fulfilled and is defined as h_k with the index k, and the set of dispatching lines is defined as Ω. The operation cost of a dispatching line is c_k, calculated by the accumulative travel time between tasks, and the cost of a missing workforce on a task is a fixed value c_u. The number of each dispatching line is x_k and the number of missing workforces of task γ is u_γ. The task demand requirement is described in Constraint (6.2), where $a_{\gamma k}$ indicates whether the dispatching line h_k contains task γ, with $a_{\gamma k} = 1$ meaning the task is assigned to this dispatching line and 0 meaning not. The limitation on the number of shifts used is described in Constraint (6.3). z_{sk} indicates the type of shift that a dispatching line uses, with $z_{sk} = 1$ meaning the dispatching line h_k uses the shift s and 0 meaning not. Because task dispatching uses the rostering results as inputs, the type and number of shifts used to dispatch tasks should correspond with those obtained from rostering on that day. Constraint (6.4) guarantees the integrality of the solution.

6.2.2 Column Generation for Task Dispatching

The challenge to solve task dispatching is to efficiently choose a set of routes to cover the task demands from numerous combinations while satisfying the time constraints of task start time and end time. For a task dispatching problem with enough long time span, it is not practical to enumerate all feasible dispatching lines. Thus, column generation can be applied in a branch and bound framework and decomposes the problem into two: the master problem to select proper routes to cover the task demand, and the pricing problem to continuously generate high-quality and feasible routes.

6.2.2.1 The Master Problem

The master problem is a set partitioning problem, a restricted version of the original problem with only a subset of dispatching lines. It aims at selecting dispatching lines from the existing routes to cover the task demand while minimizing the sum of the operation cost and the cost of missing tasks. The problem formulation is the same as in Section 6.2.1, except that for RMP, the integrality of x_k and u_γ is relaxed, and Constraint (6.4) is replaced by Constraint (6.6).

$$x_k, u_\gamma \in \mathbb{R}_+ \tag{6.6}$$

The dual problem of RMP will then be solved, and its solution will be used in column generation to generate new routes.

6.2.2.2 The Pricing Problem

The pricing problem uses the dual variable values to generate new variables of high quality for the master problem. As it can be formulated as a shortest path problem with time windows, the exploration of new variables involves calculating the reduced cost along feasible routes and choosing those with minimum negative reduced costs. The objective function of the pricing problem is thus as following:

$$min_{h_k \in \Omega}\left(c_k - \sum_{\forall \gamma \in \Gamma} a_{\gamma k} \cdot \omega_\gamma\right) \tag{6.7}$$

In this SPPTW, each task is represented as a node with start time and end time. Only the difference of start time and end time of two tasks is smaller than the travel time between them can they be connected. A feasible dispatching line is represented by a sequence of tasks that can connect to each other. The reduced cost of a route is the difference of the operation cost and the sum of dual variable values (defined as ω_γ) that assigned tasks correspond to, where the operation cost is calculated by the sum of travel time between tasks.

The process of exploration of routes with minimum negative reduced cost is implemented by the label correcting algorithm based on dynamic programming. The idea is to set labels for tasks and to track the status while extending the labels from the first to the last task within the shift. The label of every visited task will be updated based on the travel time between tasks and the dual variable value. Unpromising labels will be discarded after visiting each task. Finally, dispatching lines with minimum negative reduced costs will be chosen to enter the basis in RMP.

6.3 Disruption Management

Disruption management is responsible to make quick reactions to sudden changes in tasks caused by unexpected factors. Causes for such irregularity can be classified into following classes: air carrier delay, extreme weather delay, NAS delay, security delay, and aircraft arriving late [30]. Disruption management aims to assign unallocated tasks and conflict tasks quickly and it happens during the day of operation on the planning timeline, thus requiring high speed on computation. It is a real-time optimization problem and concerns with minimizing the missing tasks and the operation cost. The algorithm runs by a fixed time to make new rearrangements. The difficulty of disruption management lies in two perspectives: the tracking of both tasks and employees, and the identification of tasks that need to be rearranged. Thus, a decision-making system is proposed to quickly clear out the information and constructs a mixed integer programming problem that can be solved by the integer programming solver.

6.3.1 Set Partitioning Problem Formulation

Disruption management needs to track the status of every task and employee to find dispatching lines that have conflict tasks and to implement quick rearrangement. The first thing to do is to determine the time period of the disruption management operation. In this problem, the disruption management operation starts from t_d. t_p represents the time range of the operation, and t_r is defined as the time interval between each run of disruption management. t_d, t_p, and t_r are set by users. Apart from the time period information, employee data and task data are also needed. Each employee is specified by the employee ID, the existing Gantt, the status variable (indicating whether the employee is on the shift or off work), the shift start time, and the shift duration. Each task is instantiated by the place of departure and arrival, in position time and end time, task ID, employee ID (which indicates to whom the task is assigned), and the status variable. The status of a task is classified as follows: waiting to be assigned, assigned, should have been assigned, in progress, finished, and cancelled.

The integer programming model for disruption management is nearly the same as that for task dispatching. The set partition model is formulated as below:

$$\min \sum_{h'_k \in \Omega'} c_k x_k + c_u \sum_{\gamma \in \Gamma'} u_\gamma \tag{6.8}$$

$$s.t. \sum_{h'_k \in \Omega'} a_{\gamma k} x_k + u_\gamma = D_\gamma, \forall \gamma \in \Gamma' \tag{6.9}$$

$$\sum_{h'_k \in \Omega} z_{sk} \leq s_j, \forall s_j \in S \tag{6.10}$$

$$x_k, u_\gamma \in \mathbb{N} \tag{6.11}$$

$$a_{\gamma k}, z_{sk} \in \{0,1\} \tag{6.12}$$

The difference between this model and the task dispatching model is that the tasks only reassign those start within the time period. Thus, the corresponding set of dispatching lines is represented by Ω', and the task set is defined as Γ'. Because the dispatching line h'_k only considers a small subset of tasks, feasible routes can be enumerated, which is further discussed in the next section.

6.3.2 Decision-making System for Disruption Management

By inputting related data, the decision-making system sorts out the tasks that need to be reassigned and make a quick arrangement. The process can be decomposed into the following steps: data collection, conflict check, conflict disposal, optimization problem construction, and application of the integer programming solver. For this problem, these steps are organized as a decision-making system to implement disruption management in a short time.

Data collection aims at identifying the tasks and employees that are active within a certain time period. Two kinds of tasks are considered in this problem: tasks that take place during the operation ($\left[t_d, t_d + t_p\right]$), and tasks that end within the last t_r ($[t_d - t_r, t_d]$). The reason to count in the tasks that take place before the current time period is to consider the connectivity between the first task after t_d and the last task before t_d. On the other hand, the selection of employees only considers whether their shifts overlap with the operation period.

Conflict check is to identify tasks that are not reasonable in the employee Gantt. Two scenarios are considered here: whether the task is out of the range of the employee's shift range; whether the tasks in the Gantt conflicts with each other. The Gantt here refers to a sequence of tasks that are assigned to the employee and start between $\left[t_d - t_r, t_d + t_p\right]$. Because of the flight delay and some other factors, the task starts later than scheduled and may move out of the shift or cannot be connected to the following tasks with a shorter interval or none. Thus, every task is checked whether it starts after the employee's shift ends and the ID of such task will be put into the "unprocessed task pool." On the other hand, every adjacent two tasks are checked whether they are connectable. If yes, the IDs of the conflict tasks and tasks after them will be put into the "unprocessed task pool."

The conflict disposal operation is conducted after the unprocessed task pool is obtained. For each task in the pool, the employee ID of the task is erased. Also, the Gantt of the corresponding employee needs to eliminate that task and the tasks in the following.

Till this step, the process for conflict tasks is done. The tasks that need reassignment are these plus the newly generated tasks that take place within the operation period.

Optimization problem construction aims to convert the analysed data into the mathematical problem formulated in Section 6.3.1. The basic idea is to enumerate all feasible dispatching lines because the number of tasks in this problem is not large. Firstly, the dispatching line matrix $A_{\gamma k}$ is represented by the combination of every employee's Gantt. Each dispatching line h'_k is the employee's Gantt, a $1 \times |\Gamma'|$ binary matrix where $|\Gamma'|$ is the number of tasks chosen in the data collection process. Every 1 in h'_k means the corresponding task is assigned to this employee's Gantt. Then every employee checks whether the unprocessed tasks can fit into his Gantt one by one. If the task can be scheduled, a new Gantt is generated and enters $A_{\gamma k}$. As this process is done task by task, all the feasible dispatching lines can be enumerated in the end. The cost of a line is the sum of travel time between tasks.

Finally, the constructed optimization problem can be solved using the integer programming solver, and the corresponding Gantt of each employee in the new arrangement can be obtained.

6.4 Computational experiments

6.4.1 Task Dispatching

This experiment is from a case of shuttle bus dispatching from a real-world airline. Shuttle buses are major transportation that transfers passengers of arrival from the apron to the terminal or passengers of departure from the terminal to the apron. Thus, shuttle bus tasks usually have a long travel distance, which makes the pricing problem more challenging to solve. On the other hand, considering the frequency of the irregularity, the dispatching horizon is only several hours, and then the disruption management will be performed for every fixed time.

In this experiment, the dispatching horizon is set as 4 hours. Table 6.1 shows the details of tasks within the planning horizon. Seven types of data are given: task ID, status, in position time, end time, the apron of arrival, and the apron of departure. Status shows how the task is planned, including 1 for "waiting to be assigned", 2 for "assigned", 3 for "should have been assigned", 4 for "work in progress", 5 for "finished", and 6 for "cancelled". The travel time between each apron is given in Table 6.2. Information on active employees that can be dispatched is presented in Table 6.3, including the employee ID, the shift start time, and the shift end time. These data are inputs of the dispatching algorithm. The dispatching result is shown in Table 6.4, including the task ID, task status, and employee ID. If there are not enough employees to perform the task, the task status will be set as three and the employee ID be "[]".

Table 6.1 Details of tasks to be dispatched

Task ID	Status	In Position Time	End Time	Arrival	Departure
1169624838194331684	1	2019-01-16 05:25:00	2019-01-16 05:45:00	3	4
1169624838190137854	1	2019-01-16 05:30:00	2019-01-16 05:50:00	3	4
1169624838190137684	1	2019-01-16 05:40:00	2019-01-16 06:04:00	3	5
1169624838164971555	1	2019-01-16 05:45:00	2019-01-16 06:05:00	3	4
1169624838194331933	1	2019-01-16 05:45:00	2019-01-16 06:05:00	3	4
1169624838164971640	1	2019-01-16 05:50:00	2019-01-16 06:10:00	3	4
1169624838164971620	1	2019-01-16 06:00:00	2019-01-16 06:20:00	3	4
1169624838164971621	1	2019-01-16 06:05:00	2019-01-16 06:25:00	3	4
1169624838190137954	1	2019-01-16 06:05:00	2019-01-16 06:25:00	3	4
1169624838190137681	1	2019-01-16 06:10:00	2019-01-16 06:30:00	3	4
1169624838190137969	1	2019-01-16 06:15:00	2019-01-16 06:39:00	3	5
1169624838164971543	1	2019-01-16 06:25:00	2019-01-16 06:45:00	3	4
1169624838164971545	1	2019-01-16 06:25:00	2019-01-16 06:41:00	3	3
1169624838185943261	1	2019-01-16 06:30:00	2019-01-16 06:46:00	3	3
1169624838164971572	1	2019-01-16 06:35:00	2019-01-16 06:55:00	3	4

1169624838164971573	1	2019-01-16 06:40:00	2019-01-16 07:00:00	3	4
1169624838164971523	1	2019-01-16 06:45:00	2019-01-16 07:09:00	3	5
1169624838164971524	1	2019-01-16 06:50:00	2019-01-16 07:14:00	3	5
1169624838164971614	1	2019-01-16 06:50:00	2019-01-16 07:06:00	3	3
1169624838164971533	1	2019-01-16 07:10:00	2019-01-16 07:34:00	3	5
1169624838164971575	1	2019-01-16 07:10:00	2019-01-16 07:30:00	3	4
1169624838185943255	1	2019-01-16 07:10:00	2019-01-16 07:26:00	3	3
1169624838164971534	1	2019-01-16 07:15:00	2019-01-16 07:39:00	3	5
1169624838164971584	1	2019-01-16 07:15:00	2019-01-16 07:31:00	3	3
1169624838164971585	1	2019-01-16 07:20:00	2019-01-16 07:36:00	3	3
1169624838190137859	1	2019-01-16 07:20:00	2019-01-16 07:36:00	3	3
1169624838185943247	1	2019-01-16 07:30:00	2019-01-16 07:54:00	3	5
1169624838164971638	1	2019-01-16 07:35:00	2019-01-16 07:51:00	3	3
1169624838185943248	1	2019-01-16 07:35:00	2019-01-16 07:59:00	3	5
1169624838164971566	1	2019-01-16 07:45:00	2019-01-16 08:05:00	3	4
1169624838185943263	1	2019-01-16 07:45:00	2019-01-16 08:01:00	3	3
1169624838194332048	1	2019-01-16 07:45:00	2019-01-16 08:09:00	3	5
1169624838164971567	1	2019-01-16 07:50:00	2019-01-16 08:10:00	3	4
1169624838164971629	1	2019-01-16 07:55:00	2019-01-16 08:15:00	3	4
1169624838185943228	1	2019-01-16 07:55:00	2019-01-16 08:19:00	3	5
1169624838164971630	1	2019-01-16 08:00:00	2019-01-16 08:20:00	3	4
1169624838185943229	1	2019-01-16 08:00:00	2019-01-16 08:24:00	3	5
1169624838164971590	1	2019-01-16 08:05:00	2019-01-16 08:29:00	3	5
1169624838185943230	1	2019-01-16 08:05:00	2019-01-16 08:29:00	3	5
1169624838164971591	1	2019-01-16 08:10:00	2019-01-16 08:34:00	3	5
1169624838185943233	1	2019-01-16 08:25:00	2019-01-16 09:01:00	3	N2/M
1169624838194331935	1	2019-01-16 08:25:00	2019-01-16 08:49:00	3	5
1169624838190137856	1	2019-01-16 08:25:00	2019-01-16 08:49:00	3	5
1169624838185943234	1	2019-01-16 08:30:00	2019-01-16 09:06:00	3	N2/M
1169624838194331936	1	2019-01-16 08:30:00	2019-01-16 08:54:00	3	5
1169624838164971563	1	2019-01-16 08:35:00	2019-01-16 08:59:00	3	5
1169624838194332402	1	2019-01-16 08:35:00	2019-01-16 09:02:00	3	3
1169624838190137687	1	2019-01-16 08:35:00	2019-01-16 09:11:00	3	N2/M
1169624838164971564	1	2019-01-16 08:40:00	2019-01-16 09:04:00	3	5
1169624838194332403	1	2019-01-16 08:40:00	2019-01-16 09:04:00	3	3
1169624838169165941	1	2019-01-16 08:45:00	2019-01-16 09:12:00	3	3
1169624838185943240	1	2019-01-16 08:45:00	2019-01-16 09:05:00	3	4
1169624838194332404	1	2019-01-16 08:45:00	2019-01-16 09:04:00	3	3
1169624838190137929	1	2019-01-16 08:45:00	2019-01-16 09:01:00	3	3
1169624838169165942	1	2019-01-16 08:50:00	2019-01-16 09:14:00	3	3
1169624838185943266	1	2019-01-16 08:50:00	2019-01-16 09:10:00	3	4
1169624838194332405	1	2019-01-16 08:50:00	2019-01-16 09:04:00	3	3
1169624838190137966	1	2019-01-16 08:50:00	2019-01-16 09:26:00	3	N2/M
1169624838169165943	1	2019-01-16 08:55:00	2019-01-16 09:14:00	3	3
1169624838190137967	1	2019-01-16 08:55:00	2019-01-16 09:31:00	3	N2/M
1169624838164971569	1	2019-01-16 09:00:00	2019-01-16 09:24:00	3	5
1169624838169165944	1	2019-01-16 09:00:00	2019-01-16 09:14:00	3	3
1169624838164971570	1	2019-01-16 09:05:00	2019-01-16 09:29:00	3	5
1169624838164971536	1	2019-01-16 09:10:00	2019-01-16 09:34:00	3	5

Table 6.2 Travel time between aprons

Apron / Time (min) / Apron	‘3’	‘4’	‘5’	‘93’	‘95’	‘N1’	‘N2/M’
‘3’	5	7	9	11	13	19	15
‘4’	7	5	7	9	11	17	13

‘5’	9	7	5	7	9	15	11
‘93’	11	9	7	5	7	13	9
‘95’	13	11	9	7	5	11	7
‘N1'	19	17	15	13	11	5	9
'N2/M'	15	13	11	9	7	9	5

Table 6.3 Employee information

Employee ID	Start (min)	End (min)	Employee ID	Start (min)	End (min)	Employee ID	Start (min)	End (min)
1000422	300	780	1000451	480	990	1000483	480	990
1000423	420	930	1000452	480	990	1000496	480	990
1000425	480	990	1000455	480	990	1000502	300	780
1000428	360	840	1000456	480	990	1000504	480	990
1000433	480	990	1000457	480	990	1000505	480	990
1000440	480	990	1000458	240	750	1000506	360	840
1000450	480	990	1000482	480	990			

Table 6.4 Dispatching details

Task ID	Status	Employee ID	Task ID	Status	Employee ID
1169624838194331684	2	1000422	1169624838164971567	2	1000428
1169624838190137854	2	1000458	1169624838164971629	2	1000502
1169624838190137684	3	[]	1169624838185943228	3	[]
1169624838164971555	3	[]	1169624838164971630	2	1000423
1169624838194331933	2	1000502	1169624838185943229	2	1000458
1169624838164971640	3	[]	1169624838164971590	2	1000425
1169624838164971620	2	1000422	1169624838185943230	2	1000504
1169624838164971621	2	1000458	1169624838164971591	2	1000422
1169624838190137954	2	1000428	1169624838185943233	2	1000506
1169624838190137681	2	1000506	1169624838194331935	2	1000505
1169624838190137969	3	[]	1169624838190137856	2	1000455
1169624838164971543	3	[]	1169624838185943234	2	1000428
1169624838164971545	2	1000502	1169624838194331936	2	1000456
1169624838185943261	2	1000422	1169624838164971563	2	1000457
1169624838164971572	2	1000428	1169624838194332402	2	1000450
1169624838164971573	2	1000458	1169624838190137687	2	1000458
1169624838164971523	2	1000506	1169624838164971564	2	1000451
1169624838164971524	3	[]	1169624838194332403	2	1000452
1169624838164971614	2	1000502	1169624838169165941	2	1000423
1169624838164971533	3	[]	1169624838185943240	2	1000496
1169624838164971575	2	1000422	1169624838194332404	2	1000483
1169624838185943255	2	1000458	1169624838190137929	2	1000440
1169624838164971534	2	1000502	1169624838169165942	2	1000482
1169624838164971584	2	1000428	1169624838185943266	2	1000433
1169624838164971585	2	1000506	1169624838194332405	2	1000434
1169624838190137859	2	1000423	1169624838190137966	2	1000422

1169624838185943247	3	[]	1169624838169165943	2	1000435
1169624838164971638	2	1000458	1169624838190137967	2	1000502
1169624838185943248	3	[]	1169624838164971569	3	[]
1169624838164971566	2	1000506	1169624838169165944	3	[]
1169624838185943263	2	1000422	1169624838164971570	3	[]
1169624838194332048	3	[]	1169624838164971536	3	[]

6.4.2 Disruption Management

The experiment for disruption management also uses a case of shuttle bus dispatching from a real-world airline. The disruption management checks tasks from 06:00 to 07:00, and the frequency t_r is set as 20 minutes. Table 6.5 shows the active employees during the dispatching horizon. The latest dispatching details are presented in Table 6.6, where the tasks with status 1 and 2 should be checked and have a new arrangement. The result is shown in Table 6.7.

Table 6.5 Scheduling table of active employees during the dispatching horizon

Employee ID	Start (min)	End (min)	Employee ID	Start (min)	End (min)
1000427	270	840	1000465	330	900
1000435	270	840	1000467	330	900
1000477	270	840	1000470	330	900
1000424	300	870	1000487	330	900
1000434	330	900	1000490	330	900
1000441	330	900	1000496	330	960
1000442	330	900	1000497	330	900
1000443	330	900	1000432	360	900
1000446	330	900	1000448	360	930
1000454	330	900	1000481	360	930
1000459	330	990	1000464	390	960
1000461	330	960	1000499	420	990
1000463	330	900	1000493	465	1065

Table 6.6 The latest dispatching details

Task ID	Status	In Position Time	End Time	Employee ID	Arrival	Departure
1173516560808808785	5	2019-01-17 05:30:00	2019-01-17 05:55:00	1000434	4	3

1173516560808808543	4	2019-01-17 05:40:00	2019-01-17 06:05:00	1000435	4	3
1173516560808808565	2	2019-01-17 05:40:00	2019-01-17 06:01:00	1000477	3	3
1173516560808808715	2	2019-01-17 06:00:00	2019-01-17 06:25:00	1000463	4	3
1173516560808808768	2	2019-01-17 06:05:00	2019-01-17 06:26:00	1000463	3	3
1173516560800419990	2	2019-01-17 06:15:00	2019-01-17 06:40:00	1000481	4	3
1173516560808808578	2	2019-01-17 06:15:00	2019-01-17 06:40:00	1000448	4	3
1173516560808808587	2	2019-01-17 06:15:00	2019-01-17 06:40:00	1000464	4	3
1173516560808808520	2	2019-01-17 06:20:00	2019-01-17 06:41:00	1000463	3	3
1173516560808808451	2	2019-01-17 06:25:00	2019-01-17 07:06:00	1000463	N2/M	3
1173516560800419991	2	2019-01-17 06:25:00	2019-01-17 06:50:00	1000477	4	3
1173516560808808788	2	2019-01-17 06:35:00	2019-01-17 06:56:00	1000424	3	3
1173516560808808549	1	2019-01-17 06:40:00	2019-01-17 07:05:00	[]	4	3
1173516560813002752	1	2019-01-17 06:45:00	2019-01-17 07:06:00	[]	3	3
1173516560808808550	1	2019-01-17 06:50:00	2019-01-17 07:15:00	[]	4	3
1173516560808808702	1	2019-01-17 06:50:00	2019-01-17 07:19:00	[]	5	3
1173516560808808589	1	2019-01-17 06:55:00	2019-01-17 07:20:00	[]	4	3
1173516560808808713	1	2019-01-17 06:55:00	2019-01-17 07:20:00	[]	4	3
1173516560808808571	1	2019-01-17 07:00:00	2019-01-17 07:25:00	[]	4	3
1173516560808808703	1	2019-01-17 07:00:00	2019-01-17 07:29:00	[]	5	3

Table 6.7 New dispatching details

Task ID	Status	Employee ID	Task ID	Status	Employee ID
1173516560808808785	5	1000434	1173516560800419991	2	1000477
1173516560808808543	4	1000435	1173516560808808788	2	1000424
1173516560808808565	2	1000477	1173516560808808549	2	1000459
1173516560808808715	2	1000496	1173516560813002752	2	1000454
1173516560808808768	2	1000427	1173516560808808550	2	1000446
1173516560800419990	2	1000481	1173516560808808702	2	1000427
1173516560808808578	2	1000448	1173516560808808589	2	1000443
1173516560808808587	2	1000464	1173516560808808713	2	1000442
1173516560808808520	2	1000497	1173516560808808571	2	1000441
1173516560808808451	2	1000435	1173516560808808703	2	1000477

6.5 Results and Analysis

The experiment analysis comes down to task dispatching and disruption management. For the case of task dispatching, there are totally 66 tasks and 13 employees to be dispatched in the first 4 hours. The computation takes 179.20 seconds, which is not fast, but task dispatching usually happens before the day of operation and does not have a high requirement for computational efficiency. Thus, it is still acceptable for industry applications. The result shows 15 tasks cannot be scheduled, which is caused by the lack of active employees. From the scheduling result, the manager can assign employees to perform those tasks manually.

On the other hand, it takes 0.45 seconds for the disruption management algorithm to check 20 tasks and 26 employees. Table 6.6 suggests that eight tasks are new tasks, while the fourth, fifth, ninth, and tenth tasks cause conflict. After the rearrangement, these tasks are assigned to 16 employees, and no more conflict exists. It also suggests that there are more employees than needed, and the manager could arrange fewer employees during this time. Compared to task scheduling, disruption management implements real-time responses to prevent workforce deficiency when unexpected events happen.

6.6 Chapter Summary

Task scheduling is the last step of workforce scheduling and it aims to provide task schedules for on-shift employees. This includes task dispatching that assigns tasks before the day of operation and disruption management that makes new arrangements when irregularities happen. Task dispatching needs to dispatch on-work employees to perform existing tasks while minimizing the operation cost and the cost of missing tasks and it is essentially a combinatorial optimization problem. This chapter formulates the problem as a set partition model and presents a column-generation-based method to avoid the enumeration of all feasible dispatching. On the other hand, disruption management needs to check conflicts existing in the current task schedules and make new rearrangements in real-time. To solve this problem, this chapter proposes a decision-making system to convert the latest schedules to a MIP that can be solved with the integer programming solver. It normally returns the result in seconds and shows the optimal dispatching under the current scenario. Finally, experiments are conducted using a shuttle bus dispatching case from a real-world airline to validate the feasibility of the proposed methods of task scheduling.

CHAPTER 7. CONCLUSIONS, LIMITATIONS, AND FUTURE WORK

7.1 Contributions

This research aims at using optimization techniques to do workforce scheduling for airport ground staff to replace manual planning or serves as a reference for these technicians. Workforce scheduling is divided into three optimization problems: shift planning, rostering, and task scheduling. The contributions of this research are summarized as follows.

A comprehensive mathematical model for airport workforce scheduling is formulated and corresponding solutions are proposed to solve it by optimization techniques. This research studies the major problems in different stages of scheduling from the algorithm perspective. This means when generated tasks within the planning horizon are input into the proposed model, it finally outputs a rostering and task schedules for every employee before or during the day of operation, which is exactly what the manual scheduling aims at.

A 2D GA is proposed with corresponding genetic operators to solve the large-scale shift planning problem with daily-wise shift patterns. Compared to 1D GAs, it greatly improves the computing efficiency by effectively compress the unnecessary information. The experiment is done to validate the proposed 2D GA.

To solve the rostering problem, a column-generation-based method is presented. Compared to the shift planning problem, it has numerous feasible basic variables, and it is not easy to find a feasible variable. Thus, enumerating all feasible variables is not practical and column generation is introduced to the problem. Column generation decomposes the problem into a master problem and a pricing problem. The former uses part of variables within the total set and obtains a temporary solution. Then the dual problem of the restricted master problem is solved, and variable values are used to generate high-quality feasible variables in the pricing problem by computing the reduced cost, while the pricing problem is solved by the label correcting algorithm based on dynamic programming. The experiment is conducted to validate the proposed method.

Task scheduling contains two problems: task dispatching and disruption management. A column-generation-based method is presented to solve the task dispatching problem because it has similar characteristics with the rostering problem. On the other hand, a decision-making system is proposed to convert the latest dispatching details into a MIP that can be solved using the integer programming solver for disruption management.

7.2 Research Limitations

Admittedly, this research has some limitations. The major limitations come down to three. Firstly, there is no comparison experiment between the analytical approach and the proposed 2D GA for shift planning. Without such comparison, only the conclusion that 2D GAs have better performance than 1D GAs can be made, but whether analytical approaches perform better than 2D GAs is not known. Secondly, the computational performance of column generation is not satisfying in both rostering and task scheduling. When there are more than 80 shift types per day, the rostering can take one or two hours to obtain the solution. On the other hand, when the dispatching horizon in task dispatching is long, the column-generation-based method also has a very slow computing speed. Finally, a rigorous branch strategy should be applied to test the algorithm performance for rostering problems instead of using the greedy branch strategy, which results in a more accurate solution but takes longer to finish.

7.3 Future Work

If more time is given, improvement can be made for this research in the following aspects. For shift planning, a comparison between the analytical solution and 2D-GA-based solution should be made to compare their performance difference. For rostering problems, accelerating algorithms should be studied to improve the performance of the current algorithm, like using the bidirectional search method. Also, the rigorous branch scheme should be applied to compare the performance from the solution accuracy and the computing speed perspectives by using the greedy scheme. For the task dispatching problem, the column-generation-based algorithm should be accelerated, so that it could manage a longer dispatching horizon using shorter computing time.

REFERENCES

[1] Herbers, J. (2005). Models and algorithms for ground staff scheduling on airports (Doctoral dissertation, Bibliothek der RWTH Aachen).

[2] Ernst, A. T., Jiang, H., Krishnamoorthy, M., & Sier, D. (2004). Staff scheduling and rostering: A review of applications, methods and models. European journal of operational research, 153(1), 3-27.

[3] Musliu, N., Schaerf, A., & Slany, W. (2004). Local search for shift design. European journal of operational research, 153(1), 51-64.

[4] Van den Bergh, J., Beliën, J., De Bruecker, P., Demeulemeester, E., & De Boeck, L. (2013). Personnel scheduling: A literature review. European journal of operational research, 226(3), 367-385.

[5] Bechtold, S. E., & Jacobs, L. W. (1990). Implicit modeling of flexible break assignments in optimal shift scheduling. Management Science, 36(11), 1339-1351.

[6] Di Gaspero, L., Gärtner, J., Kortsarz, G., Musliu, N., Schaerf, A., & Slany, W. (2007). The minimum shift design problem. Annals of operations research, 155(1), 79-105.

[7] Clausen, T. (2010). A Rule-Based Local Search Algorithm for General Shift Design Problems in Airport Ground Handling.

[8] Lagodimos, A. G., & Leopoulos, V. (2000). Greedy heuristic algorithms for manpower shift planning. International Journal of Production Economics, 68(1), 95-106.

[9] Beasley, J. E., & Chu, P. C. (1996). A genetic algorithm for the set covering problem. European journal of operational research, 94(2), 392-404.

[10] Aiello, G., La Scalia, G., & Enea, M. (2012). A multi objective genetic algorithm for the facility layout problem based upon slicing structure encoding. Expert Systems with Applications, 39(12), 10352-10358.

[11] Ip, W. H., Wang, D., & Cho, V. (2012). Aircraft ground service scheduling problems and their genetic algorithm with hybrid assignment and sequence encoding scheme. IEEE Systems Journal, 7(4), 649-657.

[12] Pan, Q. K., Wang, L., Gao, L., & Li, W. D. (2011). An effective hybrid discrete differential evolution algorithm for the flow shop scheduling with intermediate buffers. Information Sciences, 181(3), 668-685.

[13] Ernst, A. T., Jiang, H., Krishnamoorthy, M., Owens, B., & Sier, D. (2004). An annotated bibliography of personnel scheduling and rostering. Annals of Operations Research, 127(1-4), 21-144.

[14] Cheang, B., Li, H., Lim, A., & Rodrigues, B. (2003). Nurse rostering problems—a bibliographic survey. European Journal of Operational Research, 151(3), 447-460.

[15] Burke, E. K., De Causmaecker, P., Berghe, G. V., & Van Landeghem, H. (2004). The state of the art of nurse rostering. Journal of scheduling, 7(6), 441-499.

[16] Alfares, H. K. (1998). An efficient two-phase algorithm for cyclic days-off scheduling. Computers & Operations Research, 25(11), 913-923.

[17] Alfares, H. K. (2002). Optimum workforce scheduling under the (14, 21) days-off timetable. Advances in Decision Sciences, 6(3), 191-199.

[18] Toth, P., & Vigo, D. (Eds.). (2002). The vehicle routing problem. Society for Industrial and Applied Mathematics.

[19] Golden, B. L., Raghavan, S., & Wasil, E. A. (Eds.). (2008). The vehicle routing problem: latest advances and new challenges (Vol. 43). Springer Science & Business Media.

[20] Lau, H. C., Sim, M., & Teo, K. M. (2003). Vehicle routing problem with time windows and a limited number of vehicles. European journal of operational research, 148(3), 559-569.

[21] Pureza, V., Morabito, R., & Reimann, M. (2012). Vehicle routing with multiple deliverymen: Modeling and heuristic approaches for the VRPTW. European Journal of Operational Research, 218(3), 636-647.

[22] Lim, A., & Zhang, X. (2005, January). A two-stage heuristic for the vehicle routing problem with time windows and a limited number of vehicles. In Proceedings of the 38th Annual Hawaii International Conference on System Sciences (pp. 82c-82c). IEEE.

[23] Desrochers, M., Desrosiers, J., & Solomon, M. (1992). A new optimization algorithm for the vehicle routing problem with time windows. Operations research, 40(2), 342-354.

[24] Righini, G., & Salani, M. (2006). Symmetry helps: Bounded bi-directional dynamic programming for the elementary shortest path problem with resource constraints. Discrete Optimization, 3(3), 255-273.

[25] Cook, W., & Rich, J. L. (1999). A parallel cutting-plane algorithm for the vehicle routing problem with time windows.

[26] Feillet, D., Dejax, P., Gendreau, M., & Gueguen, C. (2004). An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems. Networks: An International Journal, 44(3), 216-229.

[27] Clarke, M. D. D. (1998). Irregular airline operations: a review of the state-of-the-practice in airline operations control centers. Journal of Air Transport Management, 4(2), 67-76.

[28] Kohl, N., Larsen, A., Larsen, J., Ross, A., & Tiourine, S. (2007). Airline disruption management—perspectives, experiences and outlook. Journal of Air Transport Management, 13(3), 149-162.

[29] Clausen, J., Larsen, A., Larsen, J., & Rezanova, N. J. (2010). Disruption management in the airline industry—Concepts, models and methods. Computers & Operations Research, 37(5), 809-821.

[30] Petersen, J. D. (2012). Large-scale mixed integer optimization approaches for scheduling airline operations under irregularity (Doctoral dissertation, Georgia Institute of Technology).

[31] Clausen, T., & Pisinger, D. (2010). Airport ground staff scheduling. Technical University of Denmark (DTU).

[32] Kazda, T. and Caves, B. (2010), "Aircraft Ground Handling", Kazda, A. and Caves, R. (Ed.) Airport Design and Operation, Emerald Group Publishing Limited, pp. 173-196.

[33] Brusco, M. J., Jacobs, L. W., Bongiorno, R. J., Lyons, D. V., & Tang, B. (1995). Improving personnel scheduling at airline stations. Operations Research, 43(5), 741-751.

[34] Stolletz, R. (2010). Operational workforce planning for check-in counters at airports. Transportation Research Part E: Logistics and Transportation Review, 46(3), 414-425.

[35] Chew, K. L. (1991). Cyclic schedule for apron services. Journal of the Operational Research Society, 42(12), 1061-1069.

[36] Bonilla, J., Mohammed, H., Norris, J., Phung, T., & Zegada-Frias, I. (2005, April). Ramp Operations Monitoring System. In 2005 IEEE Design Symposium, Systems and Information Engineering (pp. 127-133). IEEE.

[37] Barnhart, C., Johnson, E. L., Nemhauser, G. L., Savelsbergh, M. W., & Vance, P. H. (1998). Branch-and-price: Column generation for solving huge integer programs. Operations research, 46(3), 316-329.

[38] Dohn, A. H., Larsen, J., & Clausen, J. (2010). Rostering and task scheduling: Applications in manpower planning. DTU Management.

[39] Desrosiers, J., Soumis, F., & Desrochers, M. (1984). Routing with time windows by column generation. Networks, 14(4), 545-565.

[40] Desrochers, M. (1988). An algorithm for the shortest path problem with resource constraints. École des hautes études commerciales, Groupe d'études et de recherche en analyse des décisions.

[41] Lübbecke, M. E. (2005). Dual variable based fathoming in dynamic programs for column generation. European Journal of Operational Research, 162(1), 122-125.

[42] Boland, N., Dethridge, J., & Dumitrescu, I. (2006). Accelerated label setting algorithms for the elementary resource constrained shortest path problem. Operations Research Letters, 34(1), 58-68.

[43] Dantzig, G. B. (1954). Letter to the editor—A comment on Edie's “Traffic delays at toll booths”. Journal of the Operations Research Society of America, 2(3), 339-341.

[44] Anderson, C. A., Jones, K. F., & Ryan, J. (1991). A two-dimensional genetic algorithm for the Ising problem. Compleex Systems, 5(3), 327-334.

[45] Tsai, M. W., Hong, T. P., & Lin, W. T. (2015). A two-dimensional genetic algorithm and its application to aircraft scheduling problem. Mathematical Problems in Engineering, 2015.

[46] Gonçalves, J. F. (2007). A hybrid genetic algorithm-heuristic for a two-dimensional orthogonal packing problem. European Journal of Operational Research, 183(3), 1212-1229.

[47] Lusby, R. M., Hansen, A. D., Range, T. M., & Larsen, J. (2010). An integrated approach to the ground crew rostering problem with work patterns.

[48] Lin, D., Xin, Z., & Huang, Y. (2015, December). Ground crew rostering for the airport check-in counter. In 2015 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM) (pp. 1462-1466). IEEE.

[49] Jaumard, B., Semet, F., & Vovor, T. (1998). A generalized linear programming model for nurse scheduling. European journal of operational research, 107(1), 1-18.

[50] Boland, N., Dethridge, J., & Dumitrescu, I. (2006). Accelerated label setting algorithms for the elementary resource constrained shortest path problem. Operations Research Letters, 34(1), 58-68.