UNIVERSITY OF
MANNHEIM

Three Essays on Workforce Scheduling

Inauguraldissertation
zur Erlangung des akademischen Grades
eines Doktors der Wirtschaftswissenschaften
der Universität Mannheim

vorgelegt von

Emilio Zamorano de Acha
Mannheim

Dekan:	*Prof. Dr. Dieter Truxius*
Referent:	*Prof. Dr. Raik Stolletz*
Korreferent:	*Prof. Dr. Jens Brunner*

Tag der mündlichen Prüfung: *22. April 2016*

To Christine,

Isabel, and Sebastian

Acknowledgements

It was late summer of 2010, when I embarked on this exciting academic voyage. Since then, I have been fortunate enough to be accompanied by people who stood by me throughout this journey. Their guidance, counsel, and support allowed me to succeed in concluding the projects included in this dissertation. In the following, few words to express my gratitude.

I would like to extend my appreciation to my supervisor, Prof. Dr. Raik Stolletz, for the invaluable advice and support he provided during the last years. I would also like to thank my co-author and colleague, Annika Becker, and the rest of the colleagues from the Chair of Production Management: Sophie Weiss, Gregor Selinka, Hendrik Guhlich, Alexander Lieder, Axel Franz, Justus A. Schwarz, Semën Nikolajewski, Andrej Saweljew, Jannik Vogel, and Matteo Biondi. Thank you for your guidance, critique, support, feedback, and incredible work atmosphere. Additionally, special thanks to my esteemed colleagues from the CDSB IS & Ops Track, Manuel, Behnaz, and Ye, for a great first year in Mannheim.

To my wife, Christine, thank you for believing in me and for our time together. Your love, patience, and dedication were fundamental to make this possible. To my children, Isabel and Sebastian, thank you for all the joy and blessings you brought to our home. You are and always will be my inspiration and motivation for everything I do. To my parents, Martha and Luis Mario, and my dear sisters, Paula, Andrea, and Daniela, for their unconditional love and support. To my in-laws Marianne, Erwin, and Monika, and everybody I might have missed: I could not have made it without you.

Finally, I would like to thank the Erich Becker Foundation, a foundation of the FraPort AG, and the Julius Paul Stiegler Memorial Foundation. Their financial support that made it possible for me to take part of workshops and international scientific conferences in order to present and discuss my research.

Mannheim, April 2016,
Emilio Zamorano de Acha

Summary

For many organizations, the efficient utilization of human resources is of great importance because of their impact on operation costs and their direct relations to customer service and employee satisfaction. Hence, it may be convenient for organizations to use decision support tools in workforce planning processes, such as workforce scheduling.

This thesis presents three different problems belonging to different planning stages of the workforce scheduling process. The first article addresses a tour scheduling problem faced by a ground-handling agency at airports. In this problem, multiskilled agents are assigned to shifts and days-off within a planning horizon of one month to cover the airlines agent requirements. The second article considers a technician routing and scheduling problem form an external maintenance provider. This problem mainly involves obtaining weekly schedules such that the maintenance tasks requested by geographically distributed customers are fulfilled. The considered decisions consist of the assignment of technicians to teams, the assignment teams to tasks, and the dispatch of teams to service routes. The third article addresses a task scheduling problem for check-in counters personnel at airports. This problem involves the daily assignment of multiskilled agents to flights (check-ins and boardings) considering the change-over times between gates, such that the agent requirements from the airlines are satisfied.

On account of the combinatorial nature of the aforementioned problems, direct solutions through commercial solvers become impractical. This is due to the high computation time required for the solution realistic test instances. For this purpose, each article describes the proposed solution approaches (both heuristic and exact methods) to solve this problems in short time and with good solution quality. Numerical studies are conducted in order to compare the performance of the proposed algorithms to the performance commercial solvers, and to provide managerial insights into the general decision process.

Contents

List of Figures

List of Tables

1. Introduction

The effective utilization of human resources is important for the majority of organizations, not only because of their direct relation to overall productivity, but also because of their impact on costs. In 2012, the compensation of employees amounted to 5.6% of the overall expenses of German employers, and 22.3% of employers worldwide (The World Bank, 2016). As the service industry is more labor intensive, this figure can significantly increase; for example, for ground-handling agencies labor cost can represent 66 - 75% of the operation costs (Steer Davies Gleave, 2010). Effective workforce scheduling can help reduce such costs as well as improve customer service, and employee satisfaction (Alfares, 2004).

Due to its relevance and multiple areas of application, workforce scheduling is a topic frequently addressed in the literature (*e.g.*, see Ernst et al. (2004b,a); Alfares (2004) for thorough reviews). Nonetheless, workforce scheduling is a complex planning process comprised of different sub-stages: (i) demand modeling, (ii) days-off scheduling, (iii) shift scheduling, (iv) line of work construction, (v) task assignment, and (vi) staff assignment (Ernst et al., 2004b). Each stage constitutes a decision problem of its own, since it is characterized by different decisions, parameters, and planning horizon length.

This dissertation addresses a different stage of the workforce scheduling process per chapter. As such, each chapter provides a problem definition, an overview of related literature, as well as a description of the proposed solution approaches. Furthermore, each chapter presents numerical studies conducted using artificial and real-world data to evaluate the performance of the algorithms and to provide managerial insights for the respective area of application.

The first article of this dissertation (Chapter 2) addresses a tour scheduling problem for check-in counters personnel at airports. This article was co-authored by Raik Stolletz [1]. The problem is encountered by ground-handling agencies at airports and consists of assigning shifts and days-off to employees to cover the flights' agent requirements for a planning horizon of one month. As airlines work with different computer systems for check-ins, agents with multiple qualifications are required. Furthermore, the agent contracts allow a high schedule flexibility (*e.g.*, variable start times, non-consecutive off-days, etc.). The objective is to obtain schedules that minimize the number of hours paid while satisfying the airlines' agent requirements, contracts conditions, and labor regulations.

Because of the high schedule flexibility and the multiskilled workforce assumption, this problem is not trivial to solve. The direct solution of a mixed-integer programming formulation of this problem via commercial solvers proves to be impractical due to its high computation time. Accordingly, this article proposes a rolling planning horizon-based heuristic as an alternative solution approach. Using real-world data from two German ground-handling agencies, numerical tests are conducted to test the performance of this algorithm. Finally, this study provides insights into the impact of the skill distribution of the agents on the resulting schedules.

[1] Stolletz, R., & Zamorano, E. (2014). A rolling planning horizon heuristic for scheduling agents with different qualifications. Transportation Research Part E: Logistics and Transportation Review, 68, 39-52.

In the second article (Chapter 3) a multiperiod technician scheduling and routing problem from an external maintenance provider is presented. This article was co-authored by Raik Stolletz [2]. In the considered problem, geographically distributed customers request maintenance tasks to the provider within a planning horizon of one week. These maintenance tasks are associated with a time interval on which customers prefer the provision of the service, *i.e.*, time windows, which can span multiple working days. In addition, the type of maintenance and its duration are known which in turn also provides the information on the qualifications, proficiency, and service time required for each task. The maintenance provider decisions consist of obtaining weekly schedules comprised of: (i) the daily assignment of technician into teams, (ii) the assignment of tasks to teams according to their skill requirements, and (iii) the dispatch of teams into service routes to serve their respective tasks. The objective is to minimize the weighted sum of travel time, customer waiting time, and technician overtime. To the best of our knowledge, this problem has not been previously addressed in literature.

The overall problem structure shares many similarities with classical vehicle routing problems, thus, the proposed solution approaches are based on existing ideas from this literature stream. Consequently, this article presents two branch-and-price algorithms that solve two different decompositions of the original problem. The first decomposition aims to (i) generate daily schedules comprised of routes for all teams in the subproblems, (ii) and select the best schedules in the master problem. The second decomposition aims to (i) generate daily routes for each team in the subproblems, and

[2]Zamorano, E. & Stolletz, R. (2016). Branch-and-price approaches for the Multiperiod Technician Routing and Scheduling Problem. European Journal of Operational Research, Available online 1 July 2016, ISSN 0377-2217, http://dx.doi.org/10.1016/j.ejor.2016.06.058.

(ii) select the best team-day routes in the master problem. The numerical tests in this article evaluate the performance of the proposed algorithms and compare it to the performance of commercial solvers. Artificial data based on known benchmark data sets and real-world data from a forklift maintenance provider are used for this purpose. Accordingly, insights into the impact of time window length on computation time and objective function value are given.

The third article (Chapter 4) presents a task scheduling problem for check-in personnel at airports. This article is a joined work with Annika Becker and Raik Stolletz [3]. Similar to Chapter 2, this problem is encountered by ground-handling agencies at airports. Nonetheless, it belongs to an operational planning stage which consists of determining a daily assignment of multiskilled agents to flights (check-in and boarding) such that the agent requirements defined by the airlines are fulfilled. As check-in counters can be located in different gates at an airport, change-over times need to be considered in the assignment process. In contrast to classical task assignment models, in the current problem the end time of each task is fixed, and the processing time depends on the arrival time of the agent to the counter. The objective is to minimize the weighted sum of agent traveling time, overtime, tardiness, and outsourcing.

The solution approach proposed in this article resembles the one presented in Chapter 3, as this task assignment problem also involves routing decisions. Accordingly, a branch-and-price algorithm that solves a decomposition of the task assignment problem is presented. In this case, however, the original problem is decomposed into (i) subproblems that generate

[3]Zamorano, E., Becker, A. & Stolletz, R. (2016). Task assignment with start time-dependent processing times for personnel at check-in counters. Working paper, under consideration for publication.

flight assignments per agent, and (ii) a master problem that selects the best assignments. In a numerical study, real-world data from a German ground-handling agency are used to compare the performance of the proposed algorithm to the performance of the direct solution via commercial solvers. Additional tests are conducted to provide insights into the impact of the tardiness limit on the overall schedules.

2. A Rolling Planning Horizon Heuristic for Scheduling Agents with Different Qualifications

Co-authors:

Raik Stolletz
Chair of Production Management, Business School, University of Mannheim, Germany

Published in:

Transportation Research Part E: (TRE),
Volume 68 (2014), pages 39-52, DOI: 10.1016/j.tre.2014.05.002

Abstract:

At airports, the workforce for check-in services is managed by ground-handling companies. Although the time-dependent agent requirements are known, the scheduling process is complex because agents operate different check-in systems and contracts allow scheduling flexibility. We propose a Mixed Integer Programming model based on the Reduced Set-Covering formulation for the monthly tour scheduling problem for a workforce with multiple non-hierarchical qualifications. We present a rolling planning

horizon-based heuristic to solve this problem. Our heuristic provides near-optimal schedules within reasonable computation time for real-world cases. In addition, we provide insights into the impact of the skill distribution on the scheduling costs.

2.1. Introduction

Third-party ground-handling agencies provide passenger handling services, such as check-in of passengers at airport terminals, to airlines. After the release in 1996 of Directive 96/67/EC on the liberalization of the ground-handling market at European airports, the number of third-party ground handlers increased by 90% in five years (SH&E, 2002), leading to higher competition and price reductions (Airport Research Center, 2009). Because staff costs represent 66% to 75% of ground handlers' operating costs, optimized workforce schedules significantly increase their profitability (Steer Davies Gleave, 2010).

A typical workforce planning process undergone by ground-handling agencies is divided into four sub-stages, each with different planning horizons, objectives, and constraints: (i) head count planning, (ii) tour scheduling, (iii) task assignment, and (iv) replanning (Stolletz, 2010). Our work focuses on the second phase of this process: monthly tour scheduling. This decision problem consists of assigning individual employees to tours (*e.g.*, daily shifts and days off) to satisfy the time-dependent employee requirements for check-in of individual flights. The requirements are driven by the flight schedule and depend on the contract with the airline. Airlines work with different computer systems for their check-in processes, and only qualified agents are allowed to operate them. Agents can be cross-trained, and it is possible for them to switch between different systems during a working shift. Agent assignment also must satisfy rules based on the contracts and qualifications of the employees. The objective of this planning task is to minimize operative workforce costs with respect to the given pool of agents and their individual contracts while ensuring that enough qualified employees are assigned to cover the demand over the course of each day.

This paper presents a Mixed Integer Programming (MIP) model for tour scheduling with multiple skills. The consideration of multiple skills prevents a solution for the MIP in short CPU times with standard software. Therefore, this study presents a new heuristic based on a rolling planning horizon approach. The main idea is to decompose the main tour scheduling problem for multiple weeks into smaller but well-connected subproblems that cover only part of the entire planning horizon. To the best of our knowledge, no previous efforts have been directed toward applying a rolling planning horizon to the problem at hand.

This paper is organized as follows: Section 2.2 gives an overview of related work. Section 2.3 presents a description of the problem and the formulation as an MIP. The Rolling Planning Horizon heuristic (RPH) is described in Section 2.4. The numerical studies described in Section 2.5 demonstrate the reliability of the solution of the RPH compared to that of the MIP, obtained for data from different ground-handling agencies. A sensitivity analysis shows the impact of the skill distribution on the overall scheduling costs. Concluding remarks and managerial insights are presented in Section 2.6.

2.2. Previous research

Reviews of decision models and solution approaches in workforce scheduling and in particular tour scheduling are given in Alfares (2004), Ernst et al.

(2004b), Ernst et al. (2004a), and Van den Bergh et al. (2013). Models for tour scheduling with multiple qualified agents often assume hierarchical skill sets; see, for example, Rong and Grunow (2009). In this case, employees with higher skill levels are allowed to be assigned to jobs that require lower skill levels. The general non-hierarchical model can solve this as a special case.

Eitzen et al. (2004) provide a generalized set-covering formulation for the fortnightly scheduling of multiskilled full- and part-time employees. They propose three different solution techniques (column expansion, column subset, and branch and price) and conduct numerical tests for problem instances with different numbers of skill levels, workforce sizes and demand patterns. Compared to our model, their shift and tour building rules are less flexible (*i.e.*, a smaller number of tour combinations are possible), and skill switching during a shift is forbidden.

Love Jr. and Hoey (1990), Loucks and Jacobs (1991), and Hojati and Patil (2011) address tour scheduling problems in the fast food industry. They consider a workforce with limited availability (*i.e.*, agents are eligible to work only during specific time windows) and shifts and tours of variable length. As with our application, agents are cross-trained and allowed to change the tasks they are assigned to during a shift. Love Jr. and Hoey (1990)'s work differs from ours in that the surplus of working hours, skills, availability of employees and number of working days are modeled as coefficients of the shifts to be assigned in the objective function. In addition, their solution technique differs from ours in that it consists of decomposing the tour scheduling problem into two network flow subproblems: construction of tours and allocation of tours to employees with the objective of minimizing the surplus of manpower.

Loucks and Jacobs (1991), in contrast to our application, propose a multi-criteria optimization model to minimize the total man-hours of overstaffing and minimize the deviation from target working hours. Their solution approach mainly differs from ours in that shifts are determined by concatenating segments (each segment corresponding to a different task), while the shift and tour building rules (minimum shift length, maximum shift length, maximum number of working days, etc.) are modeled implicitly as constraints. Their approach also differs from ours in that they propose a two-phase procedure: (i) a construction phase in which rules are used to assign task-segments to employees while relaxing the maximum number of working days constraint and (ii) an improvement phase in which violations to this constraint are eliminated and other measures (overstaffing, deviation from target hours, etc.) are improved.

Hojati and Patil (2011) revisit the model proposed by Loucks and Jacobs (1991) and propose another solution approach. Their approach differs from ours in that they decompose the main problem into a two-phase algorithm: (i) determining good shifts via a linear program that maximizes the total number of eligible and available employees and (ii) assigning shifts to employees through the use of an integer linear programming-based heuristic that determines all the shifts to be assigned to an employee, one employee at a time.

To address the complexity of tour scheduling problems, the aforementioned papers propose different decomposition approaches. As Bartholdi (1981) shows, standard tour scheduling problems are already NP-complete. With the combination of flexible schedules and multiple non-hierarchical skills, additional complexity is expected. The basis of our heuristic is the rolling planning horizon approach, an idea commonly used in production scheduling to provide partial production schedules (e.g., on a weekly basis) for longer planning periods; see Modigliani (1955), Baker (1977) and Baker and Peterson (1979). This approach can also be used to decrease the size of a planning problem by solving a series of multi-period subproblems that, in the end, cover the entire planning horizon of the original problem.

Aside from production scheduling, few efforts have been directed at applying a rolling planning horizon approach to workforce scheduling. Examples of general workforce scheduling problems using a rolling horizon approach are the allocation of flexible employees to workstations in a production line (Gronalt, 2003) and reactive nurse scheduling (Bard and Purnomo, 2005). In the tour scheduling literature we find that Day and Ryan (1997) address a fortnightly tour scheduling problem for flight attendants for short-haul flights. In their solution method, they divide the main problem into a days-off allocation and lines of work construction subproblems. For the latter, they apply a two-phase procedure by which they (i) construct lines of work based on a days-off template and (ii) improve the lines of work using a branch-and-price algorithm. They make use of a rolling horizon based-procedure to link these two steps recursively. Nevertheless, to the best of our knowledge, no previous efforts have been directed toward the development of a rolling horizon-based procedure for the tour scheduling problem as a whole.

2.3. Problem Description and Model Formulation

This study considers the tour scheduling problem for check-in counters with discontinuous schedules, *i.e.*, with no overlap between shifts on different days. The planning horizon spans D days ($d = 1, ..., D$), and each day is divided into T periods ($t = 1, ..., T$) of equal length. Based on the flight schedules and the contracts between the ground handler and the airlines, the agent requirements are known. These requirements r_{qdt} specify the number of employees with a specific skill q needed in check-in counters on day d in period t. The goal is to assign each employee e to a shift of type j on day d such that the agent requirements are covered and the overall scheduling costs are minimized.

With respect to multiple qualifications, each employee e can be differently qualified and cross-trained. The set of skills of each employee is reflected in the parameter g_{eq}, with a value of 1 if employee e has skill q and 0 otherwise.

The working conditions regarding schedules are established by the ground-handling agency in the employee contracts. The following regulations are considered:

- Shifts can vary in length between 3 and 10 hours.
- Shifts can start at any period t on day d and at different times on different days.
- A minimum resting time of R hours between shifts needs to be respected.
- A maximum of w consecutive workdays without a day off is allowed. Days off can be non-consecutive.
- There is no limit on the total working hours per week. The overall number of workdays must remain grater than d_e^{min} and less than d_e^{max} days during the planning horizon of D days.
- At most h_e^{max} working hours per month are allowed per employee.

The operational costs consist of two elements:

- Paid hours, *i.e.*, total working hours.
- Overtime costs, *i.e.*, if h_e^{max} is exceeded, the respective overtime periods ot_e have an additional cost of c^{ot} per period.

A commonly used formulation for tour scheduling problems is the set-covering formulation proposed by Dantzig (1954); see also Ernst et al. (2004b). This MIP formulation uses a matrix with all possible combinations of tours (daily shifts and days off) as input that are to be assigned to employees to meet the requirements per period. However, in our case, the flexibility of the schedules and the heterogeneity of the workforce produces a tour matrix with a large amount of data, thus making the solution intractable. Stolletz (2010) proposes a Reduced Set-Covering (RSC) formulation for the tour scheduling problem with single skills. This approach requires a matrix of all possible daily shifts only. The tour-building rules (maximum and minimum overall working days, maximum consecutive working days, etc.) are then modeled implicitly. This study extends the RSC formulation to a Multiskilled Workforce Scheduling (MWS) model. Table 2.1 summarizes the notation used.

Four main decision variables are used. First, the binary variable p_{ejd} assigns employees to shifts on a day.

$$p_{ejd} = \begin{cases} 1, & \text{if employee } e \text{ is assigned to shift } j \text{ on day } d \text{ and} \\ 0, & \text{otherwise.} \end{cases} \tag{2.1}$$

Second, the skill that an employee applies to work in each period t on day d is selected through binary variable z_{eqtd}:

$$z_{eqtd} = \begin{cases} 1, & \text{if employee } e \text{ uses skill } q \text{ in period } t \text{ of day } d \text{ and} \\ 0, & \text{otherwise.} \end{cases} \tag{2.2}$$

Third, we use an auxiliary variable y_{ed} to indicate whether an employee works on a particular day:

$$y_{ed} = \begin{cases} 1, & \text{if employee } e \text{ is assigned to work on day } d \text{ and} \\ 0, & \text{otherwise.} \end{cases} \tag{2.3}$$

Last, the number of exceeding periods over the monthly hour limit h_e^{max} per employee are determined by the variable ot_e.

In case the requirements are not met in a particular period within a day, outsourcing additional resources is allowed and decided upon using integer variable o_{qtd}. This ensures feasibility independent of the problem instance. This variable is also paired with an outsourcing cost c_q^{out}, which could be interpreted as the penalty for unmet requirements.

The staffing costs c_{ej} for a certain shift j depend on the length of shift j and the number of skills employee e has (see Equation (2.4)). The number of working periods is obtained from the difference between the last working period (f_j) and the first working period (s_j) of shift j. In case the hourly wage per period depends on an agent's number of qualifications, the cost depends on the number of qualifications per agent multiplied by a differentiation factor α:

$$c_{ej} = \overbrace{(f_j - s_j + 1)}^{\text{working periods}} * \overbrace{\left(1 + \alpha \sum_q g_{eq}\right)}^{\text{skill costs}} \tag{2.4}$$

In this way, $\alpha = 0$ when all agents are paid the same per hour regardless of the qualifications they have.

The objective function (2.5) minimizes the overall costs for all assigned shifts, for outsourced resources and for overtime:

$$\text{Minimize } F = \sum_{e=1}^{E} \sum_{j=1}^{J} \sum_{d=1}^{D} c_{ej}\, p_{ejd} + \sum_{q=1}^{Q} \sum_{d=1}^{D} \sum_{t=1}^{T} c_q^{out}\, o_{qtd} + \sum_{e=1}^{E} c^{ot}\, ot_e \tag{2.5}$$

Subject to the following constraints:

$$\sum_{j=1}^{J} p_{ejd} = y_{ed} \qquad \forall e;\ \forall d \tag{2.6}$$

$$\sum_{e=1}^{E} z_{eqtd} + o_{qtd} \geq r_{qtd} \qquad \forall q;\ \forall t;\ \forall d \tag{2.7}$$

$$z_{eqtd} \leq g_{eq} \qquad \forall e;\ \forall q;\ \forall t;\ \forall d \tag{2.8}$$

$$\sum_{q=1}^{Q} z_{eqtd} \leq \sum_{j=1}^{J} a_{tj}\, p_{ejd} \qquad \forall e;\ \forall t;\ \forall d \tag{2.9}$$

$$\sum_{j=1}^{J} \sum_{d=1}^{D} (f_j - s_j + 1) p_{ejd} \leq l \cdot h_e^{max} + ot_e \qquad \forall e \tag{2.10}$$

$$\sum_{d=1}^{D} y_{ed} \geq d_e^{min} \qquad \forall e \tag{2.11}$$

$$\sum_{d=1}^{D} y_{ed} \leq d_e^{max} \qquad \forall e \tag{2.12}$$

Table 2.1.: Notation

Indices:	
$t = 1, \ldots, T$	periods to be scheduled over a day
$d = 1, \ldots, D$	days of the planning horizon
$j = 1, \ldots, J$	shift types for a day
$e = 1, \ldots, E$	employees to be assigned to shifts
$q = 1, \ldots, Q$	skills
Parameters:	
r_{qtd}	employee requirements of skill q for period t of day d
a_{tj}	1, if period t of shift j is a working period, 0 otherwise
s_j	first working period of shift j
f_j	last working period of shift j
d_e^{min}	minimum number of workdays for employee e
d_e^{max}	maximum number of workdays for employee e
R	minimum rest periods between two shifts
w	maximum number of consecutive workdays
h_e^{max}	maximum number of working hours for employee e
g_{eq}	1, if employee e has skill q, 0 otherwise
c_{ej}	cost of shift type j assigned to worker e
c_q^{out}	cost of outsourcing skill q for one period
c^{ot}	cost of an overtime period
α	wage differentiation factor
l	number of periods in an hour
Decision variables:	
p_{ejd}	1, if employee e is assigned to shift j on day d, 0 otherwise
y_{ed}	1, if employee e is assigned to a shift on day d, 0 otherwise
z_{eqtd}	1, if employee e uses skill q in period t on day d, 0 otherwise
o_{qtd}	number of outsourced agents with skill q in period t on day d
ot_e	overtime periods of employee e

$$\sum_{\overline{d}=d}^{d+w} y_{e\overline{d}} \leq w \qquad \forall e;\ \forall d \leq D-w \qquad (2.13)$$

$$T - \sum_{j=1}^{J} f_j\, p_{ejd} + \sum_{j=1}^{J} s_j\, p_{ejd+1} - y_{ed+1} \geq y_{ed+1}\, R \quad \forall e;\ \forall d \leq D-1 \qquad (2.14)$$

$$p_{ejd};\ y_{ed};\ z_{eqtd} \in \{0,1\} \qquad \forall e;\ \forall j;\ \forall q;\ \forall t;\ \forall d \qquad (2.15)$$

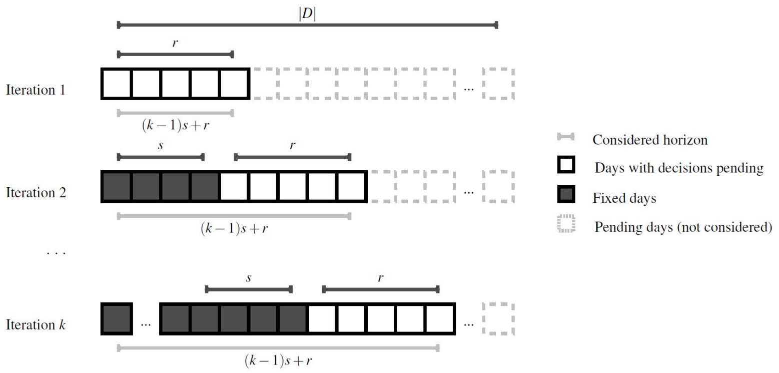
$$ot_e \geq 0 \qquad \forall e \qquad (2.16)$$

Constraints (2.6) and (2.11) to (2.15) are equivalent to those in the formulation proposed by Stolletz (2010), while the objective function (2.5) and constraints (2.7) to (2.10) and (2.16) extend the formulation to a multi-skilled workforce. Constraint (2.6) ensures that employee e is assigned to at most one shift per day and sets the variable y_{ed}. Equation (2.7) ensures that agent requirements are satisfied for each skill q, either by assigning agents or by outsourcing. Equation (2.8) ensures that employees are assigned to tasks for which they are qualified. An agent can be assigned to at most one task if the respective period is a working period, as stated in Equation (2.9). Equation (2.10) ensures that employees are assigned to at most h_e^{max} overall working hours, otherwise incurring an overtime of ot_e periods. Constraints (2.11) and (2.12) represent the lower and upper bounds for the overall working days within a tour, while Equation (2.13) allows employees to work a maximum of w consecutive days. Equation (2.14) ensures that the sum of the off periods after the end of a shift on day d plus those before starting a shift on day $d+1$ is at least R periods long.

2.4. Rolling Planning Horizon Heuristic

This section describes our proposed solution procedure based on a rolling horizon approach. The main idea of the Rolling Planning Horizon heuristic (RPH) is iteratively solving tour scheduling subproblems for shorter planning horizons and freezing the variables for the first days in each iteration (see Fig. 2.1). In iteration $k = 1$, the RPH starts with solving the subproblem for the first r days of the planning horizon. The decision variables for the first $s \leq r$ days are fixed to their solution values. The next iteration $k = 2$ solves the problem up to day $s + r$, with the decision variables fixed up to day s. While this subproblem is being solved, the tour-building constraints respect the already fixed schedule. For a certain iteration k, the subproblem is based on an already fixed schedule for $(k - 1)s$ days and covers $(k - 1)s + r$ days. After solving the MIP, the decision variables for day $(k - 1)s + 1$ up to day ks are also fixed. In the last iteration, it is ensured that the remainder of the original planning horizon is covered, especially when D is not a multiple of s.

Figure 2.1.: RPH example



To avoid infeasibility, the bounds on the overall working days and overall working hours have to be changed. Specifically, the bounds set in constraints (2.10), (2.11), and (2.12) need to be replaced, because they consider the full length of the planning horizon. The idea is to set different bounds in each iteration $k < K$ that are related to the proportion $\frac{(k-1)s+r}{D}$ of the considered planning horizon:

$$\sum_{j=1}^{J}\sum_{d=1}^{(k-1)s+r}(f_j - s_j + 1)p_{ejd} \leq \left\lceil \frac{[(k-1)s+r]\cdot l \cdot h_e^{max}}{D} \right\rceil + ot_e \quad \forall e \qquad (2.17)$$

$$\sum_{d=1}^{(k-1)s+r} y_{ed} \geq \left\lfloor \frac{[(k-1)s+r]d_e^{min}}{D} \right\rfloor \quad \forall e, \qquad (2.18)$$

$$\sum_{d=1}^{(k-1)s+r} y_{ed} \leq \left\lceil \frac{[(k-1)s+r]d_e^{max}}{D} \right\rceil \quad \forall e, \qquad (2.19)$$

Constraints (2.17)-(2.19) bind the sum of the overall assigned working hours and days up to $(k-1)s+r$ to the ratio of the minimum and maximum hours and days, respectively. In this way, the bounds are updated in every iteration. Nevertheless, these new constraints still constitute hard constraints that strictly bind the length of each tour to a fraction of the working-days limit defined in the problem setting. The last iteration K runs with the original bounds and applies the constraints (2.17)-(2.19) in place of (2.10)-(2.12), which cover the entire planning horizon. Algorithm 1 shows the pseudocode for the RPH procedure.

Algorithm 1 RPH Pseudocode

```
for $k = 1$ to $\left\lceil \frac{D}{s} \right\rceil - 1$ do
    Solve MWS subject to (2.6)-(2.9), (2.13)-(2.16) and (2.17)-(2.19) for $d \leq (k-1)s + r$ with fixed decision variables up to $d = (k-1)s$
    for $d = (k-1)s + 1$ to $ks$ do
        Fix $y_{ed}$, $o_{qdt}$, $z_{eqtd}$ and $p_{ejd}$
    end for
end for
Solve MWS subject to (2.6)-(2.16) for $d \leq D$ with fixed decision variables up to $d = (K-1)s$
```

2.5. Numerical Experiments

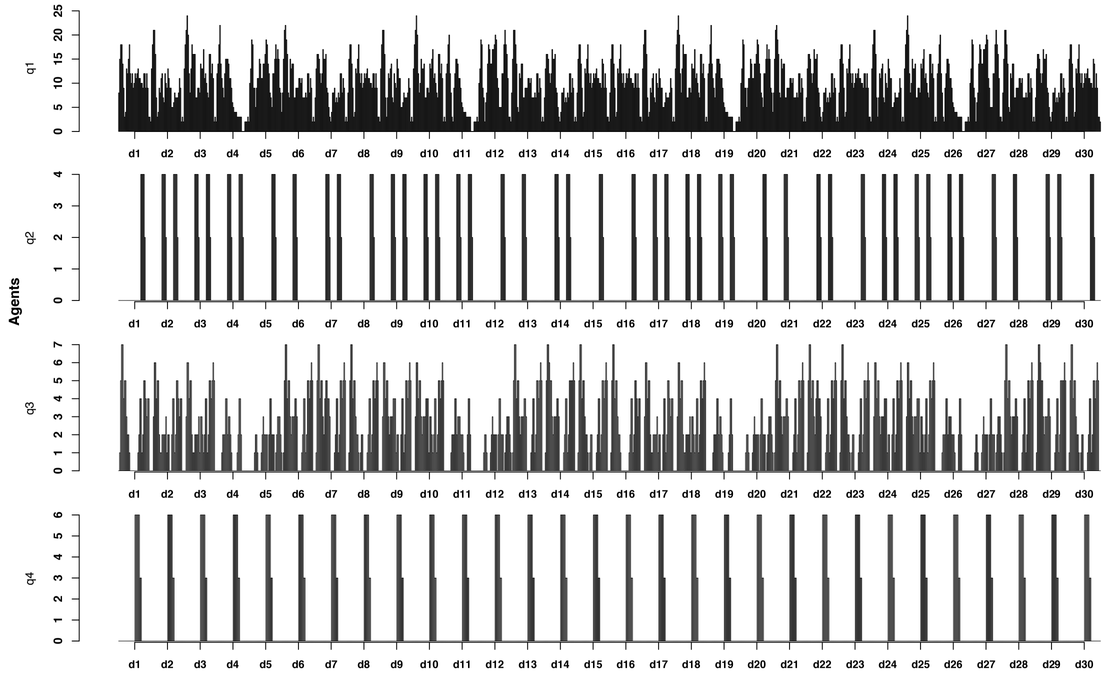
For our numerical tests, we analyze cases of two German ground-handling agencies. In the first case, we use the demand data from one ground handler to obtain good parameters r and s for the RPH. To focus on this goal, we assume fully qualified agents in Section 2.5.1. In Section 2.5.2, the same demand data are used to analyze the impact of different skill distributions on the quality of the schedules. In the second case, we test our proposed heuristic with data from a second ground-handling agency, which contain more information regarding the workforce configuration (Section 2.5.3). The MWS model and the subproblems of the RPH heuristic were solved using the Gurobi 4.6.1 solver on a 2.5-GHz Intel Core i7 machine with 8 GB of RAM.

2.5.1. Sensitivity of the RPH Parameters

For the first numerical tests, we study the impact of the choice of the parameters r and s on the performance of the RPH. The data from the first ground-handling agency contain the aggregated agent skill requirements for 30-minute intervals during the opening hours ($T = 34$) for different qualifications ($Q = 4$) within a 30-day planning horizon ($D = 30$) (see Fig. 2.2). The workforce consists of $E = 65$ employees. Because no real information on the skill distribution was given, we consider agents with all skills (*i.e.*, generalists) and no wage differentiation ($\alpha = 0$). The employee contracts establish a maximum of $w = 6$ consecutive working days without a day off and a minimum of $d_e^{min} = 19$ and maximum of $d_e^{max} = 23$ overall working days per employee. Shifts can be from 3 to 10 hours in length, with a minimum of $R = 10$ resting hours between consecutive working days. We assume there are no limits on the maximum overall working hours h_e^{max}. The costs consist of scheduled hours and outsourced hours (with $c_q^{out} = 1\times10^7$), and no overtime costs ($c_{ot} = 0$) are considered. We test different values of the RPH's parameters $r = 3, 4, 5, ..., 15$ and $s = 2, 3, 4$.

Solving the MWS MIP model with this setting to optimality yields a total of 7,941.5 scheduled hours with an overcapacity of 543.5 hours and no outsourcing. This optimal solution was found in 16,742 seconds. Table 2.2 shows the overall scheduled hours, outsourced hours, the overcapacity (in hours), and the computation time (in seconds) obtained for each test instance of the RPH. The absolute gap to the optimal scheduled hours of the RPH, compared to the optimal solution, is shown in the last column. The absolute gap is only reported for instances without outsourcing (50%

Figure 2.2.: Monthly agent requirements per qualification



of the cases) because the optimal solution does not include any outsourced periods. For the instance considered, there is no combination of r and s that clearly outperforms the others. Nevertheless, the configuration $r = 7$ and $s = 4$ had the lowest computation times and an acceptable optimality gap among the cases with no outsourcing.

To show that this configuration results in good solutions for other problem instances, we scale the workforce and the requirements. For this purpose, we multiply the demand and the workforce size by a scaling factor σ to simulate scenarios with lower demand and a smaller workforce ($\sigma < 1$) or higher demand and a larger workforce ($\sigma > 1$) than the original setting. We solve the RPH with $r = 7$ and $s = 4$ for several values of the scaling factor $\sigma = 0.2, 0.4, ..., 1.4$ and compare the solutions to the those from the MIP formulation. Table 2.3 shows, for each σ, the workforce size, the overall requirements (in hours), the overall scheduled hours, the sum of the outsourced hours, the overcapacity (in hours) and the computation time (in seconds). The last column shows the ratio of outsourced hours (as a percentage of the required hours) for each problem instance. Although for most of the instances, outsourcing is required according to the RPH results, the ratio of outsourced hours does not exceed 1.15%. The computation time of the RPH is considerably lower than that of the exact solution in all cases. In the case of $\sigma = 0.2$ an exact solution could not be found even after 33 hours of computation. This confirms that the configuration selected can be applied to different problem sizes, although testing different configurations of the RPH parameters for different problem instances is recommended.

Table 2.2.: Performance tests results for different RPH parameters

r	s	K	**Scheduled (h)**	**Outsourced (h)**	**Overcapacity (h)**	**CPU (s)**	**Opt. hours gap (%)**
3	2	15	7836.0	71.50	509.50	613	-
	3	10	7940.5	10.50	553.00	381	-
4	2	15	7944.5	9.00	555.50	13576	-
	3	10	7916.5	24.00	542.50	2407	-
5	2	15	7936.5	10.50	549.00	12185	-
	3	10	7930.0	14.00	546.00	28689	-
	4	8	7810.5	90.50	503.00	861	-
6	2	15	7962.0	1.00	565.00	97054	-
	3	10	7964.5	0.00	566.50	3243	4.23%
	4	8	7955.5	0.00	557.50	2304	2.58%
7	2	15	7955.0	13.50	570.50	39808	-
	3	10	7962.0	0.50	564.50	10017	-
	4	8	7957.5	0.00	559.50	1729	2.94%
8	2	15	7957.0	0.00	559.00	4302	2.85%
	3	10	7952.0	2.50	556.50	42260	-
	4	8	7941.0	5.00	548.00	38550	-
9	2	15	7960.0	0.00	562.00	13367	3.40%
	3	10	7959.0	0.00	561.00	5121	3.22%
	4	8	7930.0	6.00	538.00	6246	-
10	2	15	7959.5	0.00	561.50	18448	3.31%
	3	10	7956.5	0.00	558.50	13793	2.76%
	4	8	7946.5	0.00	548.50	12574	0.92%
11	2	15	7960.5	0.00	562.50	20722	3.50%
	3	10	7964.0	0.00	566.00	11474	4.14%
	4	8	7952.5	0.00	554.50	14673	2.02%
12	2	15	7957.5	0.00	559.50	18740	2.94%
	3	10	7961.5	0.00	563.50	7682	3.68%
	4	8	7959.0	0.00	561.00	17982	3.22%
13	2	15	7956.0	0.00	558.00	31461	2.67%
	3	10	7954.0	0.00	556.00	21035	2.30%
	4	8	7962.0	0.00	564.00	26940	3.77%
14	2	15	7952.0	0.00	554.00	35577	1.93%
	3	10	7956.0	0.00	558.00	22479	2.67%
	4	8	7944.5	0.00	546.50	33921	0.55%
15	2	15	7955.5	0.00	557.50	44787	2.58%
	3	10	7951.0	0.00	553.00	11687	1.75%
	4	8	7951.0	0.00	553.00	47946	1.75%

Table 2.3.: Performance tests for different problem sizes

	σ	**E**	**Required (h)**	**Scheduled (h)**	**Outsourced (h)**	**Overcapacity (h)**	**CPU (s)**	**Outsourced hours ratio**
MIP	0.2*	13	1479.6	2099	15.5	660.4	33 h	1.05%
	0.4	26	2959.2	3711	0	751.8	5672	0.00%
	0.6	39	4438.8	5324	0	885.2	8890	0.00%
	0.8	52	5918.4	6904	0	985.6	8002	0.00%
	1	65	7398.0	7941.5	0	543.5	16742	0.00%
	1.2	78	8877.6	10065	0	1187.4	66876	0.00%
	1.4	91	10357.2	12126	0	1768.8	28236	0.00%
RPH	0.2	13	1479.6	2064.5	17	601.9	474	1.15%
	0.4	26	2959.2	3685	6	731.8	396	0.20%
	0.6	39	4438.8	5301.5	4.5	867.2	1175	0.10%
	0.8	52	5918.4	6900.5	2	984.1	3507	0.03%
	1	65	7398.0	7957.5	0	559.5	1729	0.00%
	1.2	78	8877.6	10068.5	3	1193.9	38576	0.03%
	1.4	91	10357.2	12051.5	17	1711.3	22798	0.16%

* Best solution found after 33 hours of computation

2.5.2. Multiple Skills Analysis

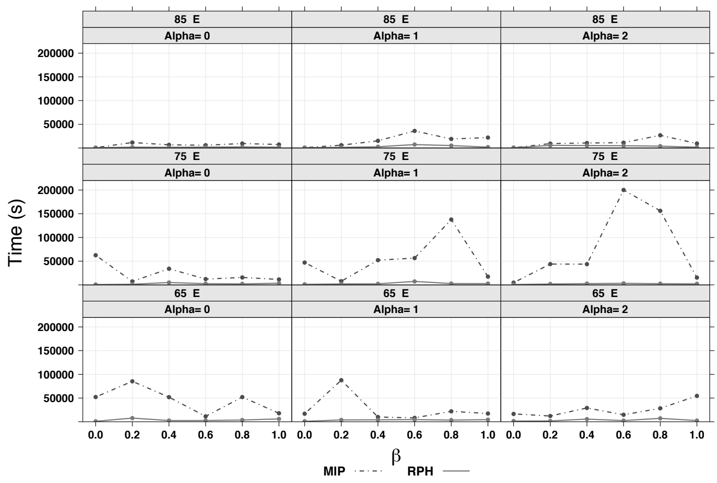
In the following tests, we address the impact that different skill distributions have on the solutions. For this purpose, we use the same requirement data as in Section 2.5.1, although with different skill distributions and workforce sizes. We consider a combination of $\beta \cdot E$ generalist agents and $(1-\beta) \cdot E$ agents with a single skill profile (specialists). The number of specialists of a specific skill is calculated by multiplying the proportion each skill is present in the requirements with $(1-\beta) \cdot E$ (rounding up for the two most required skills and rounding down for the two least required skills).

Different workforce sizes $E \in \{65, 75, 85\}$ and skill distributions ($\beta = 0.0, 0.2, ..., 1.0$) are considered, with the original agent requirements held unchanged for all cases. Furthermore, we look at different cost differentiation factors ($\alpha = 0, 1, 2$), such that for every additional skill an employee has, the hourly cost increases by α units. A total of 54 different problem instances were tested.

Figures 2.3 and 2.4 show a comparison of the results obtained with the MIP formulation and the RPH heuristic for these tests. Figure 2.3 presents a comparison of the computation time (in seconds) as a function of β for each combination of α and E. The computation time for the RPH is less than 2.14 hours in all cases, while the computation time for the MIP for the entire planning horizon is up to 55.60 hours. Figure 2.4 shows the absolute gap (in percentage) of the solutions of the RPH with respect to the optimal scheduled hours. The cases with $\beta = 0$ are not reported because the results for both the MIP and RPH require outsourcing. For all remaining cases, the gap remains below 0.34% from the optimum. To summarize, the RPH solves all cases considered very quickly with an acceptably small optimality gap. Due to the short CPU times, it is possible to evaluate the value of flexibility for different skill-mix scenarios.

Figure 2.5 shows a comparison of the overcapacity hours obtained for the solutions for both methods and all combinations of β, α, and E. Additional flexibility is acquired with larger workforce sizes, and thus, overcapacity hours are reduced in most cases. This is because more agents can be assigned to shorter shifts and fewer idle periods are covered. On the other hand, we observe different behavior in the overcapacity level when varying the values of β and α. If all agents are paid the same ($\alpha = 0$), the full benefits of adding additional fully qualified agents (generalists) have less impact when 20% or more generalists are present in the workforce. For $\alpha = 1$ and $\alpha = 2$, the the overcapacity hours are not strictly decreasing with increasing values of β, because more fully qualified agents provide more flexibility, albeit at with a higher cost.

Figure 2.3.: Computation time comparison

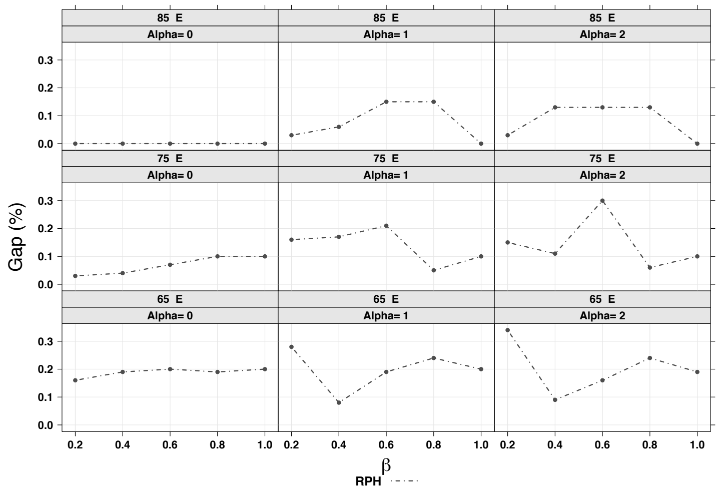


2.5.3. Real-world Workforce Case

The tests described in the previous sections were conducted using real-world demand data with artificial workforce configurations from a ground-handling agency . In this section, in contrast, we use real-world information on both the demand and the workforce from a second ground-handling agency to test the RPH heuristic under a more realistic setting.

In this new case, the agent requirements r_{qtd} are known and given in 5-minute periods ($T = 254$) for a planning horizon of $D = 30$ days. There are 12 different check-in systems, *i.e.*, $Q = 12$ skills. The requirements for each skill are highly time-dependent and are not correlated to the requirements of other skills.

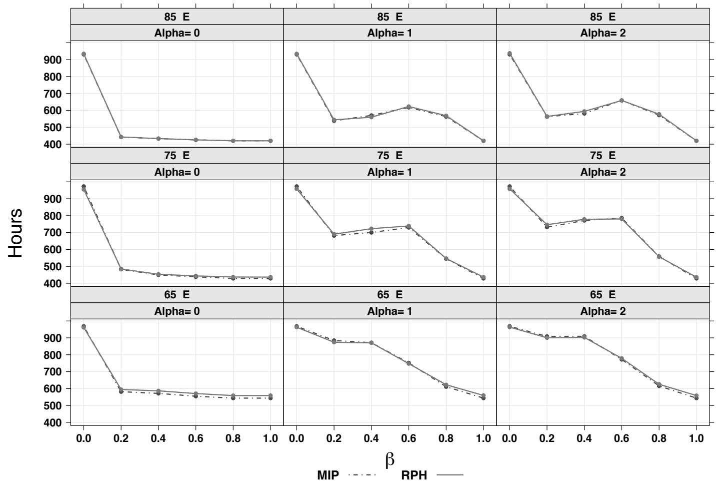
Figure 2.4.: Optimal scheduled hours gap (%)



The check-in personnel consist of by $E_{full} = 4$ full-time agents, $E_{part} = 3$ part-time agents and $E_{flexi} = 47$ flexible agents. Each month, full-time agents are allowed to work at most $h^{max}_{full} = 165$ hours and part-time agents $h^{max}_{part} = 40$ hours. Flexible agents have no monthly h^{max}_{flexi} limit. Working hours that exceed these limits are considered overtime and are paid 25% more than normal working hours. In case the agent requirements cannot be met with the available workforce, additional non-check-in personnel can be assigned to cover them, although this is not desirable because it removes them from their original job positions (*i.e.*, equivalent to outsourcing).

Shifts can have a duration of 3 to 10 hours per day and can start in 15-minute intervals, with a minimum of $R = 10$ rest hours between the end and the start of the shifts of consecutive working days. A maximum of

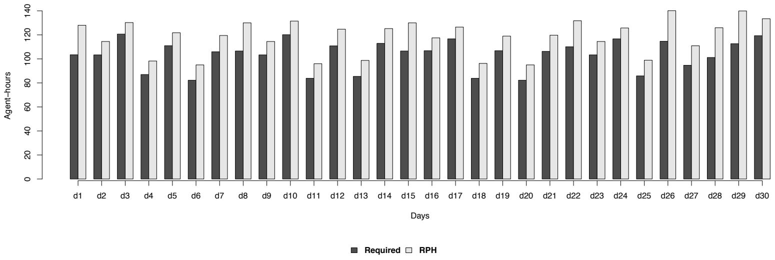
Figure 2.5.: Overcapacity



$w = 6$ consecutive working days is allowed without a day off, and days off can be non-consecutive. With respect to the overall working days, the corresponding contract rules do not take into account vacation and sick days in a specific month. Therefore, to more faithfully represent the real setting, we fix the assigned working days to the actual number of working days per employee from the data set.

An exact solution could not be obtained by solving the MIP formulation because of out-of-memory errors. The RPH approach with $r = 7$ and $s = 4$ obtained a final solution in 5.81 hours. Figure 2.6 shows a comparison of the overall scheduled hours from the RPH solution with respect to the overall requirements. As Figure 2.6 shows, overcapacity is inevitably present each day due to the heterogeneity of the demand. This is still acceptable

Figure 2.6.: Comparison of overall scheduled and required hours



because the average utilization of the workforce is 87%. Additionally, this solution incurred 14.50 overall overtime hours and only 1 hour during which non-check-in personnel were required.

This case study shows that the RPH is able to solve complex problem instances as they appear in real applications.

2.6. Conclusions and Further Research

This paper addresses the tour scheduling problem for a multiskilled workforce at check-in counters at airports. To the best of our knowledge, the combination of the non-hierarchical nature of the skills and the high degree of scheduling flexibility analyzed in this study has not been considered in previous research. We present a Mixed Integer Programming model that extends the Reduced Set-Covering formulation to multiple skills and a Rolling Planning Horizon heuristic to solve the problem.

The results of a numerical study conducted using real-world data from a ground-handling agency are presented. These results show that the proposed heuristic provides good solutions in an acceptable amount of time for different problem sizes and skill distributions. However, a proper selection of the parameters for the RPH is crucial to the performance of the heuristic, and additional testing is recommended for different problem settings. Furthermore, the results show that additional flexibility can be gained by increasing the proportion of generalist agents in the workforce. On the other hand, this additional flexibility can be acquired at a higher cost if salaries depend on the number of qualifications of the agents.

The results of additional tests conducted using data from a second ground-handling agency show that with our proposed heuristic, it is possible to solve realistic problems that are otherwise not solvable with an MIP formulation, and to obtain good-quality solutions.

Further research needs to be conducted to extend the proposed model to consider agent preferences and fairness measures in the planning process. Another research direction is to implement subsequent phases of the workforce planning process (*e.g.*, task assignment and replanning) with the present model to serve as an integrative robust planning tool for ground-handling agencies.

3. Branch-and-price approaches for the Multiperiod Technician Routing and Scheduling Problem

Co-author:

Raik Stolletz
Chair of Production Management, Business School, University of Mannheim, Germany

Published in: European Journal of Operational Research, Available online 1 July 2016, ISSN 0377-2217, DOI: doi:10.1016/j.ejor.2016.06.058

Abstract:

This paper addresses a technician routing and scheduling problem motivated by the case of an external maintenance provider. Technicians are proficient in different skills and paired into teams to perform maintenance tasks. Tasks are skill constrained and have time windows that may span multiple days. The objective is to determine the daily assignment of technicians into teams, of teams to tasks, and of teams to daily routes such that the operation costs are minimized. We propose a mixed integer program and a branch-and-price algorithm to solve this problem. Exploiting the structure

of the problem, alternative formulations are used for the column generation-phase of the algorithm. Using real-world data from an external maintenance provider, we conduct numerical studies to evaluate the performance of our proposed solution approaches.

3.1. Introduction

This paper addresses the Multiperiod Technician Routing and Scheduling Problem (MPTRSP) based on the case of an external maintenance provider (EMP) specialized in electric forklifts. Services offered by the considered EMP include preventive and corrective maintenance, failure diagnoses, and the delivery of spare parts and supplements to different geographically distributed customers. As these services are offered at the customers' locations, they require the visit of a team of technicians based on customers' service requests. Such requests may be either known in advance (*e.g.,* in the case of preventive maintenance based on yearly contracts and scheduled repairs), or requested on demand (*e.g.,* emergency repairs when breakdowns occur). In this research we consider programmed maintenance tasks, *i.e.,* maintenance demand is deterministic and known in advance, because the current problem constitutes one part of a hierarchical planning process for the EMP.

These maintenance tasks have the following features:

- First, based on maintenance contracts, a time frame for the provision of each task has been agreed upon with the respective customer. In this way, customers can specify the allowed time(s) (and day(s)) on which a task can be performed, *i.e.,* time windows are defined. These time windows can span several hours and possibly several days depending on the type of service requested. For example, a task can be allowed between Monday at 9:00 am and Friday at 1:00 pm, a corrective maintenance task between Tuesday at 4:00 pm and Wednesday at 12:00 pm, or a delivery between Thursday at 9:00 am and 4:00 pm. Due to the EMP's working day duration limit, these time windows can be formulated as multiple alternative time windows on consecutive days. Maintenance contracts incur a penalty fee if a customer is visited after the latest starting time. The contracts also define the maximum waiting time per customer.

- Second, a single customer can request more than one service; therefore, multiple tasks might need to be performed at the same location. Due to safety requirements, however, tasks are not allowed to be left unfinished or split, *i.e.,* tasks are non-preemptive, and at most one team per task is allowed. Third, tasks' service time and travel time are known, as well as the minimum skill proficiency required for each task.

As for the workforce, first, each day, technicians are paired into teams and then dispatched to visit customers. Team compositions remain fixed for the duration day (*i.e.*, a technician is assigned to at most one team per day), although different team compositions on different days is not forbidden. The number of teams and size of a team are defined based on the company's safety policies, e.g. since some maintenance tasks for forklifts require lifting heavy equipment. Each technician is qualified in certain skill domains (*e.g.,* hydraulics, mechanic, electric, etc.) and has a proficiency level in each domain (*e.g.,* basic proficiency, medium proficiency, or expert). Thus, team qualifications depend on the combined qualifications of the team members. Second, on any given day, teams are dispatched from the EMP's location and need to return to it before closing time. If a team returns after this time, overtime is incurred at a cost. Other workforce scheduling decisions, such as shift scheduling, days-off scheduling, and meal-breaks placement are outside of the scope of this paper.

The planning problem faced by this EMP consists of obtaining weekly schedules comprised by: (i) daily pairing of technicians into teams, (ii) assignment of teams to tasks, and (iii) dispatching teams into routes to perform their respective tasks. The goal is to obtain weekly schedules such that the operational costs are minimized. That is, the said schedules should minimize travel costs, customer waiting time and technician overtime. To the best of our knowledge, this problem with all its features has not been previously addressed in literature.

The contribution of this paper is three-fold:

- The multiperiod technician routing and scheduling problem is defined and its relation to existing research is presented.
- Branch-and-price algorithms to solve this problem to optimality are proposed.
- Numerical experiments using real-world data are conducted to test the performance of the proposed solution approaches.

The remainder of this paper is organized as follows. In Section 3.2, an overview of the related literature is shown, and in Section 3.3, a problem description and a model formulation are provided. A description of the proposed solution approaches and details on their implementation are shown in Section 3.4. The numerical studies conducted in Section 3.5 compare the performance of the different solution approaches using real-world data. Concluding remarks and directions for future research are presented in Section 3.6.

3.2. Related literature

Personnel scheduling has been addressed frequently by many researchers due to its presence in many application areas (see Ernst et al. (2004b), Ernst et al. (2004a), and Van den Bergh et al. (2013) for more thorough reviews). As it is a complex process, several sub-stages need to be carried on for its completion: (i) demand modeling, (ii) days-off scheduling, (iii) shift scheduling, (iv) line of work construction, (v) task assignment, and (vi) staff assignment (Ernst et al., 2004b). The present problem belongs to task assignment and staff assignment stage of this classification, although it also involves additional decisions that need to be considered, *e.g.*, the assignment of technicians into teams, the assignment of teams to tasks, the construction of routes, and the selection of the day on which a service is provided.

On the other hand, the Multiperiod Technician Routing and Scheduling Problem (MTRSP) can also be classified as a generalization of the Workforce Scheduling and Routing Problem (WSRP), as it combines aspects from personnel scheduling and vehicle routing problems (see Castillo-Salazar et al. (2014) for a review on recent WSRP literature). However, it also incorporates additional features: tasks have multiple alternative time windows on multiple days (*i.e.,* multiple periods are considered), and teams of technicians need to be formed on a daily basis (*i.e.,* team building).

In the following we review related literature that addresses similar problems. First, we analyze existing literature and classify it according to the extent to which they incorporate the special features from our problem. Second, we provide an overview of the proposed solution approaches used in these works.

3.2.1. Related problem formulations

This section reviews problem formulations similar to ours found in literature. This analysis focus on contrasting the features considered in these formulations, in comparison to the features of our proposed problem. Accordingly, the related works are then classified into the following categories: (i) a single period and no team building, (ii) multiple periods and no team building, and (iii) a single period with team building.

(i) Single period, no team building Most literature on WSRP-related problems differs to our problem in the fact that they consider single-day routes and the tasks to be assigned involve single agents (*i.e.*, no team building decisions are made).

In Xu and Chiu (2001) a staff-scheduling problem for field technicians of a telecommunication company is considered. Skill levels for each technician are given as a percentage of proficiency on each task. In contrast to our model, tasks do not require proficiency levels but rather the objective function maximizes the assignment of technicians to tasks by weighting their proficiency level, so that highly skilled technicians are more likely to be assigned. Dohn et al. (2009b) propose a manpower allocation problem in which teams of technicians are assigned into maintenance tasks constrained by time windows. Tasks have different skill requirements, which constrain their assignment to teams, and the collaboration of multiple teams in one task is allowed. Pillac et al. (2012) consider the Technician Routing and Scheduling Problem (TRSP) where the technician-task compatibility includes spare parts and tools as well. All these models, however, differ from ours in that all tasks observe a single-day planning horizon and no team building decisions are considered. Lim et al. (2004) deal with a different version of a manpower allocation problem for service personnel at the port of Singapore. They formulate this problem as a multi-objective problem where, in contrast to our model, minimize the number of servicemen used as a primary objective and minimize the routing costs as a secondary objective.

Additional WRSP literature addresses the home health care problem. In this problem staff members from a health care provider are dispatched to visit geographically dispersed clients (Akjiratikarl et al. (2007), Cheng and Rich (1998)). In other examples of home health care related literature the tasks or clients require the visit of multiple agents. In Bertels and Fahle (2006) and Eveborn et al. (2006b) a staff planning problem for home care is addressed where some visits require the coordination of multiple staff due to, *e.g.*, safety regulations. However, in contrast to our problem, these groups of staff do not remain together for the entirety of the working day. Instead, the authors model additional constraints to force the synchronization on the arrival and departure of the agents.

(ii) Multiple periods, no team building In Blakeley et al. (2003), technicians are assigned to customers and dispatched on routes in a multiperiod planning horizon. Technicians have different qualifications, and compatibility of customers and technicians is taken into consideration. Similar to the Periodic Vehicle Routing Problem (PVRP) (Francis et al., 2006, 2008), technicians are assigned to routes according to predetermined visit frequencies, and customers are visited within their preferred visit days. In contrast to our model, the visit frequency is predetermined, the validity periods do not consider specific time windows, and no team building decisions are made.

Similarly, Tang et al. (2007) consider the routing of technicians to maintenance tasks for geographically distributed customers on multiple days. The authors formulate this problem as a Multiple Tour Maximum Collection Problem with Time-Dependent rewards (MTMCPTD). In this model, single-day routes are obtained for technicians such that the reward obtained for visiting customers is maximized. In contrast to the MPTRSP, instead of time windows on multiple days, the authors consider time-dependent rewards based on the urgency of the task. This problem considers single technicians with homogeneous skills; thus, no team building is required.

Bostel et al. (2008) address a similar problem in their multiperiod planning and routing for a generic field force problem. The authors consider tasks that can be executed by single technicians on multiple days. In contrast to our problem, only part of the tasks are subject to time windows, and only a homogeneous workforce is considered. Additionally, the authors do not consider team-building decisions.

In Barrera et al. (2012) a combination of a timetabling and crew scheduling problems of a public healthcare provider is addressed. In this problem health care agents serve a set of geographically dispersed customers. Costumers can request multiple services from a portfolio of services on multiple days within a given planning horizon. The purpose of this problem is to construct routes that satisfy all services using a minimum number of health care agents and balancing their workload. The formulation of Barrera et al. (2012) differs mainly from ours on the fact that an homogeneous workforce is considered, routes do not start and end at the depot, and no team-building decisions nor task assignments are involved as its focus is on staffing decisions.

(iii) Single period with team building The previous articles address technician scheduling without assignment of technicians into teams. Approaches in which team building decisions are considered are: Cordeau et al. (2010a) and Kovacs et al. (2011).

Cordeau et al. (2010a) address the Technician and Task Scheduling Problem (TTSP) in a telecommunications company. In this problem, tasks differ in difficulty and require more than one technician with specific qualifications. Similar to our approach, technicians have different proficiency levels in several skill domains, and they are assigned into teams to serve maintenance tasks. The objective is to minimize the overall makespan while satisfying the tasks' skill requirements, precedence constraints, technician availability, and working day limitations. This problem differs from the MPTRSP in that routing decisions are not considered, outsourcing tasks is allowed, and only single-day routes are obtained.

Kovacs et al. (2011) combine elements from Cordeau et al. (2010a) with routing decisions and define the Service Technician Routing and Scheduling Problem (STRSP). Similar to the TTSP, the authors consider technicians with a number of skills on different levels that can be grouped into teams to perform maintenance tasks. In contrast to Cordeau et al. (2010a), the STRSP incorporates traveling costs to determine service routes for the teams. The objective is to obtain routes for each team such that the travel costs are minimized while satisfying the tasks' skill requirements and time windows. This problem differs from ours in the following: only single days are considered, teams can have different sizes, which depend on tasks requirements, and tasks can be left unassigned to the existing workforce and be outsourced at a different (higher) cost.

In conclusion and to the best of our knowledge, a WRSP where both multiple periods and team building are considered simultaneously has not been previously addressed in the literature.

3.2.2. Solution approaches

Among the solution approaches in the presented related literature several types of methods are used: (i) heuristics and meta-heuristics, (ii) mathematical programming approaches, and (iii) hybrid methods.

The most commonly used solution approaches are heuristics and meta-heuristics due to their versatility and short computation time. In the presented literature the heuristic methods used are: 2-step heuristics (Barrera et al., 2012), Local Search (Souffriau et al., 2013), Adaptive Large Neighborhood Search (Cordeau et al., 2010a; Kovacs et al., 2011; Pillac et al., 2012), Tabu Search (Tang et al., 2007), Particle Swarm Optimization (Akjiratikarl et al., 2007), Greedy heuristics (Xu and Chiu, 2001), and Simulated Annealing (Lim et al., 2004).

Among the mathematical programming approaches, the direct solution of a mixed integer program through a commercial solver is a viable choice (Barrera et al., 2012), although several author successfully use branch-and-price and similar algorithms (*e.g.,* Boussier et al. (2006); Dohn et al.

(2009b); Bostel et al. (2008)) to solve related problems to optimality and in short computation time.

Finally, hybrid approaches can also be applied for the solution of similar problems. For example, the combination of linear programming, constraint programming and metaheuristics (Bertels and Fahle, 2006), or the combination of linear programming and a repeating matching heuristics (Eveborn et al., 2006b).

We propose two branch-and-price algorithms for the solution of the presented problem. To conclude, the reason for the selection of this solution method is twofold. First, the MTRSP shares many elements with the classical Vehicle Routing Problem (VRP), on which branch-and-price algorithms are commonly used (see, *e.g.*, Desrochers et al. (1992), Kohl and Desrosiers (1999), and Liberatore et al. (2010)). Second, to the best of our knowledge, there is no exact solution approach developed to solve the MTRSP due to its novelty, although closely related problems can be solved successfully with branch-and-price algorithms.

3.3. Problem description and model formulation

The Multiperiod Technician Routing and Scheduling Problem (MPTRSP) can be defined as follows: a graph $\mathcal{G}(I, \mathcal{A})$ is given where the vertex set I is made of a set of I' geographically dispersed maintenance tasks and two dummy nodes (o and $\bar{o}$) representing the depot, and $\mathcal{A} = \{(i, j)|i \in I, j \in I, i \neq j\}$ corresponds to the arc set. Pairs of technicians $m, n \in M$ are assigned to a team $k \in K$ and dispatched to serve these maintenance tasks. Each day $d \in D$, each team k departs and arrives at the depot within the opening hours $[e, f]$. A transportation time t_{ij} and a traveling cost c_{ij}

(including the service time p_i associated to each task $i \in I'$) are associated to each arc $(i, j) \in \mathcal{A}$. We assume that the triangle inequality is satisfied including the case where tasks i and j are in the same location because the service times are non-negative. Furthermore, we define $l \in L$ as the proficiency level in skill domain $q \in Q$. Then, a solution for the considered problem consists of obtaining a service schedule satisfying all tasks within the planning horizon. This schedule is composed of the daily assignment of technicians into teams and the assignment of daily routes to teams. Each route (*i.e.*, a cycle from o to $\bar{o}$ in $\mathcal{G}$) is a sequence of tasks performed by a team within a working day.

Each task is associated with a time window(s), which indicates the preferred visit times and days. As daily operation hours are limited, these time windows can be represented as multiple single-day time windows. Let $[a_{id}, b_{id}]$ be the earliest and latest starting time of task i in day d, respectively, and $\bar{D}_i \subset D$ be the set of days on which performing task i is allowed. Thus, we let the set $\mathcal{A}_d \subset \mathcal{A}$ be the subset of allowed arcs for day d. Skill requirements are represented by v_{iql}, a binary parameter equal to 1 if the task i requires a technician with the skill $q \in Q$ with at least a level $l \in L$ of proficiency.

Similar to Cordeau et al. (2010a) and Kovacs et al. (2011), technician qualifications are represented by g_{mql}, which is a binary parameter equal to 1 if technician m has at least a level l of proficiency on skill domain q. Each team is composed of exactly τ technicians (in our case $\tau = 2$, although our formulation allows for different values for τ). The combined qualifications of the members of a team must satisfy the skill requirements of each task. Assigning "overqualified" teams is allowed at no cost. Team configurations remain fixed for the duration day (*i.e.*, a technician is assigned to at most one team per day), although different team configurations on different days is not forbidden.

Six main decision variables are used. First, the binary variable z_{mkd} assigns technicians to teams on a day:

$$z_{mkd} = \begin{cases} 1, & \text{if technician } m \text{ is assigned to team } k \text{ on day } d, \\ 0, & \text{otherwise.} \end{cases}, \quad \forall m \in M,\ \forall k \in K,\ \forall d \in D$$

Second, binary variable x_{ijkd} assigns a sequence of tasks to a team on a day:

$$x_{ijkd} = \begin{cases} 1, & \text{if team } k \text{ uses arc } (i,j) \text{ on day } d \\ 0, & \text{otherwise.} \end{cases}, \quad \forall (i,j) \in \mathcal{A},\ \forall k \in K,\ \forall d \in D$$

Note that for teams with no assignments, empty routes ($x_{o\bar{o}kd} = 1$) are allowed.

Third, a binary variable y_{ikd} assigns tasks to teams on a day to simplify the notation:

$$y_{ikd} = \begin{cases} 1, & \text{if task } i \text{ is assigned to team } k \text{ on day } d \\ 0, & \text{otherwise.} \end{cases}, \quad \forall i \in I,\ \forall k \in K,\ \forall d \in D$$

Fourth, the starting time of task i of team k on day d is defined by the positive variable s_{ikd} $\forall i \in I',\ \forall k \in K,\ \forall d \in D$. Fifth, a task can be started only after its earliest starting time, *i.e.*, if a team arrives earlier, it waits at no cost, whereas starting after the latest starting time, *i.e.*, the customer waits at a cost. We penalize customer waiting time for task i by the use of the positive auxiliary variable w_i, defined as

$$w_i = \begin{cases} \max(0, s_{ikd} - b_{id}), & \text{if } y_{ikd} = 1 \\ 0, & \text{otherwise,} \end{cases}, \ \forall i \in I'$$

with a cost of w^{cost} per time unit and with a maximum waiting time of w^{max}.

Last, in case a team arrives at depot at the end of its route after closing time, overtime is incurred. We model this by the use of the positive auxiliary variable ot_{kd} defined as

$$ot_{kd} = \max(0, s_{\bar{o}kd} - f), \ \forall k \in K, \ \forall d \in D,$$

with a cost of ot^{cost} per time unit and with a maximum overtime ot^{max}. Table 3.1 shows the notation used for this formulation.

The objective is to minimize the traveling, waiting, and overtime costs. The MPTRSP can be formulated as a mixed integer programming model as follows:

$$\text{Minimize} \ \underbrace{\sum_{(i,j)\in\mathcal{A}} \sum_{k\in K} \sum_{d\in D} c_{ij} \cdot x_{ijkd}}_{\text{Travel costs}} + \underbrace{w^{cost} \cdot \sum_{i\in I'} w_i}_{\text{Waiting costs}} + \underbrace{ot^{cost} \sum_{k\in K} \sum_{d\in D} ot_{kd}}_{\text{Overtime costs}} \tag{3.1}$$

Subject to the following constraints:

$$\sum_{k\in K} \sum_{d\in D} y_{ikd} = 1 \qquad \forall i \in I' \tag{3.2}$$

$$\sum_{j:(i,j)\in\mathcal{A}_d} x_{ijkd} = y_{ikd} \qquad \forall i \in I', \ \forall k \in K, \ \forall d \in D \tag{3.3}$$

Table 3.1.: Notation

Sets	
D	Set of days
I'	Set of tasks
I	Set of tasks $I = I' \cup \{o, \bar{o}\}$
$\bar{D}_i \subseteq D$	Set of allowed visit days of task i
$\mathcal{A}$	Set of arcs $\mathcal{A} \subseteq \{(i,j) \| i,j \in I\}$
$\mathcal{A}_d \subset \mathcal{A}$	Set of arcs $\mathcal{A}_d \subseteq \{(i,j) \| i,j \in I$ and $d \in \bar{D}_i \cap \bar{D}_j\}$
K	Set of teams
M	Set of technicians
Q	Set of skills
L	Set of proficiency levels
Parameters	
c_{ij}	Traveling cost from task i to task j
t_{ij}	Traveling time from task i to task j including service time
$[a_{id}, b_{id}]$	Earliest and latest starting time window for task i on day d
v_{iql}	1 if task i requires at least a proficiency level l on skill q for task i, 0 otherwise
p_i	Service time of task i
g_{mql}	1 if technician m has at least a proficiency level l on skill q, 0 otherwise
$[e, f]$	Daily work hours
τ	Nr. of technicians in a team
w^{cost}	Cost per time unit of customer waiting time
ot^{cost}	Cost per time unit of overtime
w^{max}	Maximum customer waiting time
ot^{max}	Maximum overtime
Decision variables	
z_{mkd}	1 if technician m is assigned to team k on day d, 0 otherwise
y_{ikd}	1 if team k is assigned to task i on day d, 0 otherwise
x_{ijkd}	1 if team k goes directly from task i to task j on day d, 0 otherwise
s_{ikd}	Start time of task i by team k on day d
w_i	Waiting time of task i
ot_{kd}	Overtime of team k on day d

$$\sum_{j:(o,j)\in\mathcal{A}_d} x_{ojkd} = 1 \qquad \forall k \in K,\ \forall d \in D \tag{3.4}$$

$$\sum_{i:(i,\bar{o})\in\mathcal{A}_d} x_{i\bar{o}kd} = 1 \qquad \forall k \in K,\ \forall d \in D \tag{3.5}$$

$$\sum_{i:(i,h)\in\mathcal{A}_d} x_{ihkd} - \sum_{j:(h,j)\in\mathcal{A}_d} x_{hjkd} = 0 \qquad \forall h \in I',\ \forall k \in K,\ \forall d \in D \tag{3.6}$$

$$x_{ijkd}(s_{ikd} + t_{ij} - s_{jkd}) \leq 0 \qquad \forall i,j : (i,j) \in \mathcal{A}_d,\ \forall k \in K,\ \forall d \in D \tag{3.7}$$

$$y_{ikd}(a_{id} - s_{ikd}) \leq 0 \qquad \forall i \in I',\ \forall k \in K,\ \forall d \in D \tag{3.8}$$

$$y_{ikd}(s_{ikd} - b_{id} - w_i) \leq 0 \qquad \forall i \in I',\ \forall k \in K,\ \forall d \in D \tag{3.9}$$

$$x_{ojkd}(s_{jkd} - e - t_{oj}) \geq 0 \qquad \forall j \in I',\ \forall k \in K,\ \forall d \in D \tag{3.10}$$

$$x_{i\bar{o}kd}(s_{ikd} + t_{i\bar{o}} - f - ot_{kd}) \leq 0 \qquad \forall i \in I',\ \forall k \in K,\ \forall d \in D \tag{3.11}$$

$$\sum_{k\in K} z_{mkd} \leq 1 \qquad \forall m \in M,\ \forall d \in D \tag{3.12}$$

$$\sum_{m\in M} z_{mkd} = \tau \qquad \forall k \in K,\ \forall d \in D \tag{3.13}$$

$$v_{iql}\, y_{ikd} \leq \sum_{m\in M} g_{mql} z_{mkd} \qquad \forall i \in I',\ \forall q \in Q,\ \forall l \in L,\ \forall k \in K,\ \forall d \in D \tag{3.14}$$

$$0 \leq w_i \leq w^{max} \qquad \forall i \in I' \tag{3.15}$$

$$0 \leq ot_{kd} \leq ot^{max} \qquad \forall k \in K,\ \forall d \in D \tag{3.16}$$

$$s_{ikd} \geq 0 \qquad \forall i \in I,\ \forall k \in K,\ \forall d \in D \tag{3.17}$$

$$x_{ijkd}, y_{ikd}, z_{mkd} \in \{0,1\} \qquad \forall i,j \in I,\ \forall m \in M,\ \forall k \in K,\ \forall d \in D \tag{3.18}$$

Constraints (3.2) and (3.3) ensure that all tasks are assigned exactly once to one team on any feasible day with respect to the requested visit days. Constraints (3.4) and (3.5) ensure that each team starts and ends its route at the depot. Constraint (3.6) ensures flow conservation when visiting a task site. Constraint (3.7) allows a task to be started only if its preceding task is completed, which implicitly corresponds to a sub-tour elimination constraint. Constraints (3.8) and (3.9) state that a task can only start within its time window. If a team starts a task after its latest starting time, a penalty on customer waiting is incurred. Note that constraints (3.7)-(3.9) can be linearized by using a big M formulation. The routes' duration limit and overtime are defined by constraints (3.10) and (3.11). Constraint (3.12) ensures that a technician is assigned to at most one team per day, and equation (3.13) defines the number of technicians per team. Constraint (3.14) ensures that the skill requirements of task i are fulfilled by the combination of skills from the technicians assigned to a team. Constraints (3.15) and (3.16) define the bounds for customer waiting and overtime, respectively. Lastly, constraints (3.17) and (3.18) define the positive and the binary variables, respectively.

From (3.1), constraints (3.2)-(3.11), and (3.15)-(3.18) we observe that the MPTRSP contains, as a special case, the Vehicle Routing Problem with Time Windows (VRPTW) with a fixed fleet size, which is an NP-hard problem (Solomon, 1987; Solomon and Desrosiers, 1988a). Note that even finding a feasible solution to the VRPTW is itself an NP-complete problem (Savelsbergh, 1985). The remaining team building constraints correspond to a special case of the two-to-one assignment problem (2-1-AP) (Goossens et al., 2012) with no cost-coefficients.

3.4. Branch-and-price algorithms

The choice of column generation-based algorithms to solve workforce scheduling and routing problems stems from their similarities with the vehicle routing problem (VRP), on which this method is commonly applied (see, *e.g.*, Desrochers et al. (1992), Kohl and Desrosiers (1999), and Liberatore et al. (2010)). This approach exploits the structure of the problem by reformulating is as a master problem (MP) and a pricing problem(s) using, *e.g.,* Danzig-Wolfe decomposition, and provides a fractional lower bound. To guarantee integrality, column generation can be embedded in a branch-and-bound algorithm to branch on fractional variables, *i.e.,* branch-and-price can be applied.

In the MIP formulation for the Multiperiod Technician Routing and Scheduling Problem (MTRSP), we observe that (3.2) and (3.3) are the only constraints that link the teams and days together, whereas the rest are team and day specific. This allows us to apply a similar decomposition and use a branch-and-price algorithm to solve the problem. However, this fact also allows a certain degree of flexibility as to which constraints are selected to be in the MP and pricing problem(s). In this section, we illustrate the application of a branch-and-price algorithm using two alternative decompositions of the MPTRSP and details on its implementation.

3.4.1. Day decomposition

Exploiting the fact that the time windows spanning several days can be represented as multiple alternative single-day time windows, the MPTRSP can be decomposed into a master problem that selects the best routes for all teams on each day in the planning horizon, and single-day pricing problems from which daily technician-to-team assignments and daily routes for all teams are obtained.

As for the MP, let P_d be the set of feasible schedules containing feasible routes (*i.e.*, paths) for all teams on day d and λ_{dp} be a binary variable equal to 1 on day d schedule p is used, 0 otherwise. Additionally, let θ_{idp} be a parameter equal to 1 if task i on day d is on schedule p, 0 otherwise, and φ_{ijkdp} be a parameter equal to 1 if arc (i,j) is used by team k on day d and schedule p, 0 otherwise. Then, the integer master problem for the day-decomposition (MP-D) can be defined as:

$$\text{Minimize} \sum_{d \in D} \sum_{p \in P_d} c_{dp} \lambda_{dp} \tag{3.19}$$

Subject to:

$$\sum_{d \in D} \sum_{p \in P_d} \theta_{idp} \lambda_{dp} = 1 \qquad \forall i \in I' \tag{3.20}$$

$$\sum_{p \in P_d} \lambda_{dp} = 1 \qquad \forall d \in D \tag{3.21}$$

$$\lambda_{dp} \in \{0,1\} \qquad \forall d \in D,\ \forall p \in P_d \tag{3.22}$$

Where the cost of a schedule $p \in P_d$ is defined by:

$$c_{dp} = \sum_{k \in K} \sum_{i,j:(i,j) \in p} c_{ij} \varphi_{ijkd} + w^{cost} \sum_{i \in p} w_i + ot^{cost} \sum_{k \in K} ot_{kd} \tag{3.23}$$

The objective is to minimize the overall costs. The set partitioning constraint (3.20) ensures that tasks are assigned exactly once. Convexity constraint (3.21) states that exactly one schedule is used per day. Last, constraint (3.22) defines the binary requirements for the assignment variables. The linear relaxation of this formulation (LMP-D), obtained by relaxing constraint (3.22), can be used to obtain a lower bound on the MP. However, set P_d may contain a large number of feasible schedules thus making the solution to this problem computationally intractable. Therefore, in a column generation approach, only subset $P'_d \subset P_d$ of schedules, *i.e.,* columns, are considered in a restricted master problem (RMP-D). Additional feasible columns are obtained by the iterative solution of a pricing problem, which consists of finding schedules with negative reduced cost $\bar{c}_{dp}$:

$$\begin{aligned} \bar{c}_{dp} &= c_{dp} - \sum_{i \in I'} \theta_{idp} \pi_i - \rho_d \\ &= \sum_{k \in K} \sum_{i,j:(i,j) \in p} (c_{ij} - \pi_i) \varphi_{ijkd} + w^{cost} \sum_{i \in p} w_i \\ &\quad + ot^{cost} \sum_{k \in K} ot_{kd} - \rho_d \end{aligned} \tag{3.24}$$

where π_i, and ρ_d correspond to the dual values of constraints (3.20) and (3.21) in the optimal solution of the RMP-D.

These values are then passed to the subproblems from which daily schedules for all teams are obtained. The pricing problems correspond to a single-day version of the original MPTRSP problem (*i.e.,* (3.1)-(3.18), without the d index), one for each day in the planning horizon, which can be solved, *e.g.*, by the use of commercial optimization software. For this, for each d-subproblem, we let $I'_{sub_d} \subset I'$ and $I_{sub_d} \subset I$ be the set of tasks that can be served on day d. Also, we let $\mathcal{A}_{sub_d} = \{(i,j) | i,j \in I_{sub_d}\}$ be the corresponding subnetwork for this subproblem.

This decomposition has the advantage that it can easily incorporate different team sizes. However, each pricing problem corresponds to a single-day MPTRSP (a general case of the VRPTW), which is an NP-hard problem. This may constitute a major drawback for this approach regarding computation time, as column generation involves solving the pricing problems multiple times.

3.4.2. Team-day decomposition

In addition to decomposing the MPTRSP by days, another possibility is to decompose it by teams. As technician-to-team assignments can change on a daily basis, however, some additional considerations are required. Because the workforce composition and team size are known in advance, we can assume without loss of generality that all possible team configurations are also known. We now let $\mathcal{K}$ be the set of all teams resulting from all possible combinations of all technicians $m \in M$ in teams of size τ size, hence $|\mathcal{K}| = \frac{|M|!}{\tau!(|M|-\tau)!}$. Therefore, the assignment of technicians to teams can be pre-defined by letting z_{mkd} be a parameter with a value of 1 if technician $m \in M$ belongs to team configuration $k \in \mathcal{K}$ on day $d \in D$, and 0 otherwise. Thus the technician-to-team assignments become parameters instead of decision variables. In other words, instead of building teams (constraints (3.12)-(3.13)), we now address a daily assignment of pre-configured teams to tasks. This allows us to decompose the MPTRSP into a master problem defining routes for each team on each day of the planning horizon and a pricing problem from which daily routes per team are obtained.

As for the master problem for this decomposition (MP-KD), let P_{kd} be the set of feasible routes (*i.e.*, paths) for team k on day d and λ_{kdp} be a binary variable equal to 1 only if team configuration k uses path p on day d. Additionally, let θ_{ikdp} be a parameter equal to 1 if task i is in path p of team configuration k on day d, 0 otherwise, φ_{ijkdp} be a parameter equal to 1 if arc (i, j) is used by team k on day d and path p, 0 otherwise, and η_{mkdp} be a parameter equal to 1 only if technician m is assigned on the path p of team configuration k on day d. Then, the MP can be defined as:

$$\text{Minimize} \sum_{k \in K} \sum_{d \in D} \sum_{p \in P_{kd}} c_{kdp} \lambda_{kdp} \tag{3.25}$$

Subject to:

$$\sum_{k \in \mathcal{K}} \sum_{d \in D} \sum_{p \in P_{kd}} \theta_{ikdp} \lambda_{kdp} = 1 \qquad \forall i \in I' \tag{3.26}$$

$$\sum_{k \in \mathcal{K}} \sum_{p \in P_{kd}} \eta_{mkdp} \lambda_{kdp} \leq 1 \qquad \forall m \in M,\ \forall d \in D \tag{3.27}$$

$$\sum_{p \in P_{kd}} \lambda_{kdp} = 1 \qquad \forall k \in \mathcal{K},\ \forall d \in D \tag{3.28}$$

$$\lambda_{kdp} \in \{0, 1\} \qquad \forall k \in \mathcal{K},\ \forall d \in D,\ \forall p \in P_{kd} \tag{3.29}$$

Where the cost of a path $p \in P_{kd}$ is defined by:

$$c_{kdp} = \sum_{i,j:(i,j) \in p} c_{ij} \varphi_{ijkd} + w^{cost} \sum_{i \in p} w_i + ot^{cost} ot_{kd} \tag{3.30}$$

This MP-KD formulation is fairly similar to the MP-D from Section 3.4.1 with the exception of constraint (3.27). Because multiple team configurations can include the same technician, constraint (3.27) ensures that each technician is assigned at most once per day. Column generation can be used to solve the linear relaxation of this set partitioning formulation (LMP-KD), which is obtained by relaxing constraint (3.29). However, as set P_{kd} may contain a large number of columns, only a subset $P'_{kd} \subset P_{kd}$ of paths are considered in a restricted version of the master problem (RMP-KD). Further feasible columns are obtained by the iterative solution of pricing problems (one for each team and each day), which consist of finding single-day paths for each team with negative reduced cost. For each $k - d$-subproblem we let $I'_{sub_{kd}} = \{i \in I' | v_{iql} \leq \sum_{m \in M} g_{mql} z_{mkd}\ \forall q \in Q,\ \forall l \in L; a_{id} \neq \varnothing\}$ and $I_{sub_{kd}} = I'_{sub_{kd}} \cup \{o, \bar{o}\}$ be the set of tasks that can be served on day d

and for which team k is qualified. Also, let $\mathcal{A}_{sub_{kd}} = \{(i,j)|i,j \in I_{sub_{kd}}\}$. A given $k-d$-subproblem can be formulated as a mixed integer programming model (we omit the k and d index to simplify the notation) as follows:

$$\begin{aligned} \text{Minimize } \bar{c}_p = & \sum_{i,j:(i,j)\in\mathcal{A}_{sub}} (c_{ij} - \pi_i)x_{ij} + w^{cost} \sum_{i\in I_{sub}} w_i + \\ & ot^{cost} ot - \sum_{m\in M} z_m \beta_m - \rho \end{aligned} \tag{3.31}$$

Subject to:

$$\sum_{j\in I_{sub}} x_{oj} = 1 \tag{3.32}$$

$$\sum_{i\in I_{sub}} x_{i\bar{o}} = 1 \tag{3.33}$$

$$\sum_{i\in I_{sub}} x_{ih} - \sum_{j\in I_{sub}} x_{hj} = 0 \qquad \forall h \in I'_{sub} \tag{3.34}$$

$$x_{ij}(s_i + t_{ij} - s_j) \leq 0 \qquad \forall i,j \in I_{sub} \tag{3.35}$$

$$a_i \leq s_i \leq b_i + w_i \qquad \forall i \in I'_{sub} \tag{3.36}$$

$$x_{oj}(s_j - e - t_{oj}) \geq 0 \qquad \forall j \in I'_{sub} \tag{3.37}$$

$$x_{i\bar{o}}(s_i + t_{i\bar{o}} - f - ot) \leq 0 \qquad \forall i \in I'_{sub} \tag{3.38}$$

$$0 \leq w_i \leq w^{max} \qquad \forall i \in I'_{sub} \tag{3.39}$$

$$0 \leq ot \leq ot^{max} \tag{3.40}$$

$$s_i \geq 0 \qquad \forall i \in I'_{sub} \tag{3.41}$$

$$x_{ij} \in \{0,1\} \qquad \forall i,j \in I_{sub} \tag{3.42}$$

where π_i, β_{md}, and ρ_{kd} correspond to the dual values of constraints (3.26), (3.27), and (3.28) in the optimal solution of the restricted master problem.

x_{ij} is a binary variable with a value of 1 if the arc (i, j) is used, and 0 otherwise. s_i and w_i are positive variables corresponding to the starting time and the customer waiting time of task i, while ot is a positive variable that comprises the (possible) overtime incurred in this route. The objective is to find a route with a minimum reduced cost (3.31). Constraints (3.32), (3.33), and (3.34) ensure the feasibility of the route. Constraint (3.35) defines the tasks' starting time, whereas constraint (3.36) ensures that time windows are respected. If a task starts after its latest starting time, customer waiting occurs. Constraints (3.37) and (3.38) define the working day limits, and if a team arrives at the depot after closing time, overtime is incurred. Note that constraints (3.35), (3.37), and (3.38) can be linearized using a big M formulation. Constraints (3.39) and (3.40) define the bounds for the waiting and overtime variables, respectively. Last, positive variables (3.41) and binary variables (3.42) are defined.

These subproblems correspond to an Elementary Shortest Path Problem with Resource Constraints (ESPPRC). This means, that it is necessary to explicitly enforce elementary paths as the solution of the subproblems, since the dual values π_i and ρ_d associated with a task may lead to negative cost cycles (*i.e.*, subcycles). Note that ESPPRC in graphs with negative cost cycles is strongly NP-hard (Dror, 1994). This could represent a major drawback for this decomposition, as the number of pricing problems depends directly on the number of possible team configurations. Because in our algorithm we need to solve these subproblems several times, an efficient solution approach is required.

Labeling algorithm In the VRPTW literature, a common approach to solving the pricing problems is to relax the elementary requirement for paths so that the subproblem changes to a Shortest Path Problem with Resource Constraints (SPPRC) (See, *e.g.,* Desrochers et al. (1992)). This approach can be solved efficiently using dynamic programming, although it has the disadvantage of weakening the lower bound computed by the column-generation master problem (Bode and Irnich, 2014b). However, several approaches successfully apply similar dynamic programming methods to efficiently solve the ESPPRC (Feillet et al., 2004) and the ESPPRC with soft time windows (Liberatore et al., 2010) because of its ability to provide tight lower bounds. In the following, we describe a dynamic programming approach similar to the approach proposed by Feillet et al. (2004) to solve the ESPPRC to optimality. Our approach, similar to Ioachim et al. (1998) and Liberatore et al. (2010), considers time dependent cost (*i.e.,* in the case of customer waiting and overtime) in addition to elementary paths.

Labeling algorithm overview In the following, we present an overview of our proposed labeling algorithm (see Algorithm 2). For this, we require the following notation: let $\mathcal{U}_i$ and $\mathcal{P}_i$ represent the set of unprocessed and processed labels for task i, respectively. These sets allow keeping track of the creation of labels throughout the algorithm. The process starts by setting sets $\mathcal{U}_i$ and $\mathcal{P}_i$ to the empty set, by creating the first label L_o, and by adding it to the unprocessed set. Then, iteratively, the label with the minimum cost is selected from the unprocessed set to look for possible extensions along its arcs. If feasible, new labels are created by extending each selected label along all its connecting arcs to successive tasks, and added to the unprocessed set. Once no more extensions are possible, the currently explored label is added to the processed set. To keep the size of these sets tractable, dominance rules are applied to discard labels that cannot lead to an optimal solution. Once all paths arriving at the depot $\bar{o}$ are obtained and no more unprocessed labels remain, the label with the minimum cost is selected and corresponds to the optimal solution of the respective $k-d$ subproblem. Its corresponding reduced cost is determined by $\bar{c} = C(T_{\bar{o}}) - \sum_{m \in M} z_m \beta_m - \rho$. If multiple labels (*i.e.*, columns) are found with the same minimum reduced cost, and none can be dominated, then all are added to the master problem.

Algorithm 2 Labeling algorithm overview

```
Set $\mathcal{U}_i = \mathcal{P}_i = \varnothing \ \ \forall i \in I_{sub}$
Set $L_o = (o, T_o = e, C(T_o) = 0, S_o = 0)$
Set $\mathcal{U}_o = \{L_o\}$
while $\bigcup_{i \in I_{sub}} \mathcal{U}_i \neq \varnothing$ do
   Choose the label $L_m \in \bigcup_{i \in I_{sub}} \mathcal{U}_i$ with the minimum cost and remove
   it from $\mathcal{U}_m$
   for all arcs $(m, j) \in N_{sub}$ do
      Extend $L_m$ along arc $(m, j)$ to create label $L_j$ using the REF's
      if $L_j$ is feasible then
         Add $L_j$ to $\mathcal{U}_j$
         Apply dominance rules to $\mathcal{U}_j$ and $\mathcal{P}_j$
      end if
      Add $L_m$ to $\mathcal{P}_m$
   end for
   Filter $\mathcal{P}_{\bar{o}}$ to find the shortest $o - \bar{o}$ elementary path
end while
```

Label setting In short, the objective of the algorithm is to obtain routes from o to $\bar{o}$ for a team on a single day, visiting a subset of the other tasks, such that the reduced cost is minimized. For this purpose, we need to define the following concepts. A *state* associated with task i corresponds to a partial path of visited tasks from the depot to i. Each task can be associated with more than one state because multiple feasible paths can end in said task. To represent such states, we make use of multidimensional resource vectors or *labels* represented by $L_i = (i, T_i, C(T_i), S_i)$ where:

- i is the last visited task,
- T_i is the starting time of task i,
- $C(T_i)$, similar to Liberatore et al. (2010), is a convex stepwise function describing the time dependent cost, and
- $S_i \in \mathbb{R}^{|I|}$, similar to Feillet et al. (2004), is a binary visitation vector of representing the tasks visited in the partial path $\langle o, ..., i \rangle$.

Resource Extension Functions (REFs) The process is initialized with $L_o = (o, T_o = e, C(T_o) = 0, S_o = 0)$. Through the algorithm, labels are extended to successive feasible labels of the form $L_j = (j, T_j, C'(T_j), S_j)$ by appending an additional arc (i, j) to the partial path and updating its corresponding resources. The consumption of each resource along the arc (i, j) is described by non-decreasing functions known as resource extension functions (REF)(Irnich, 2008). The proposed algorithm uses the following REF's:

- Start time

$$T_j = \max(a_j, T_i + t_{ij})$$

- Cost

$$C'(T_j) = \begin{cases} C(T_i) + c_{ij} + \max(0, w^{cost}(b_j - T_j)) & -\pi_i, \\ & \text{if } j \neq \bar{o} \\ C(T_i) + c_{ij} + \max(0, ot^{cost}(f - T_j)) & -\pi_i, \\ & \text{if } j = \bar{o} \end{cases}$$

- Visited tasks

$$S_{j_m} = \begin{cases} S_{j_m} + 1, & \text{if } m = j \\ S_{j_m} & \text{if } m \neq j \end{cases}$$

The same way as in Feillet et al. (2004) and Liberatore et al. (2010) in order to enforce elementariness in the path, a dummy resource is associated with each task $i \in I$ such that this resource is consumed when i is visited. That is, $S_{j_m} = 1$ if the path $\langle o, ..., i \rangle$ visits task m, 0 otherwise. The path $\langle o, ..., i \rangle$ corresponds to an elementary path only if $S_{j_m} \leq 1 \;\; \forall m \in I'_{sub}$.

Feasibility A label $(i, T_i, C(T_i), S_i)$ is feasible if $T_i \leq b_i + w^{max} | i \neq \bar{o}$ and $T_i \leq b_i + ot^{max} | i = \bar{o}$ and is an elementary path. Labels that do not satisfy these conditions are discarded.

Dominance As multiple labels per task are allowed, the ability to discard labels that do not lead to an optimal solution is crucial for the efficiency of the algorithm. For this purpose, dominance rules are applied. Let $L_i = (i, T_i, C(T_i), S_i)$ and $L'_i = (i, T'_i, C'(T'_i), S'_i)$ be two labels of task i. Then, L_i dominates L'_i if $L_i < L'_i$ component-wise, *i.e.,* if $T_i \leq T'_i$, $C(T_i) \leq C'(T'_i)$ and $S_i \leqq S'_i$ and at least one of the inequalities is strict. Note that $S_i \leqq S'_i$ means that each element of vector S_i is less than or equal to each corresponding element of vector S'_i. If a label is dominated then it can be discarded; when all components are equal, both labels are kept.

To strengthen these rules, one can also make use of an additional "reachability" resource, as presented in Feillet et al. (2004). Let $U_i(T_i)$ be a binary resource vector that indicates whether it is feasible to visit other tasks from task i given the respective travel and starting times, *i.e.,* if other tasks are reachable from i. This means:

$$U_{i_m}(T_i) = \begin{cases} 1, & \text{if } S_{i_m} = 1 \\ 1, & \text{if } T_i + t_{im} > b_i + w^{max} \\ 1, & \text{if } T_i + t_{im} > b_i + ot^{max}, m = \bar{o}, \\ 0, & \text{otherwise} \end{cases}$$

Then, from labels $L_i = (i, T_i, C(T_i), S_i)$ and $L'_i = (i, T'_i, C'(T'_i), S'_i)$ with $U_i(T_i)$ and $U'_i(T'_i)$, the dominance rule $S_i \leq S'_i$ can be replaced by $S_i + U_i(T_i) \leq S'_i + U'_i(T'_i)$.

Acceleration strategies The column generation algorithm introduced in the previous section presents an important drawback in that it requires that several ESPPRC subproblems (which are NP-hard) be solved multiple times for its completion. Solving an ESPPRC to optimality can be computationally expensive, even with our proposed labeling algorithm, when the number of tasks increases. It is possible, however, to solve each subproblem heuristically since only in the last iteration of the column generation algorithm an exact solution is required to guarantee optimality (*i.e.*, no additional columns with negative reduced cost can be found).

In the following, we propose one insertion heuristic and three heuristics based on the labeling algorithm previously described. The main idea is to apply these algorithms sequentially during the execution of the column generation algorithm. After an initialization phase, the first method is used to solve each subproblem and pass information to the master problem until no columns with negative reduced cost are found. Then, the next solution method is chosen, and the process is repeated until no more negative reduced cost columns are found with the last (exact) algorithm. Gendreau et al. (2016) propose a similar approach where a heuristically stopped labeling algorithm and an exact branch-and-cut algorithm are used in sequence to solve ESPPRC pricing problems.

Insertion heuristic The column generation is initialized by the use of a modified version of the insertion heuristic from Solomon (1987). As the vehicle "fleet" is not known in advance, we make use of the artificial teams $\mathcal{K}$ described at the beginning of this section. This algorithm can be summarized as follows: (i) for each team $k \in \mathcal{K}$ and day $d \in D$, we sequentially initialize routes by choosing the task with the highest transportation costs to and from the depot. (ii) For each task h available to team k on day d, we calculate the operation costs obtained from inserting it, if feasible, in each arc (i, j) of the initial route. (iii) The insertion $(i, h, j)^*$ with the minimum cost is selected, and the initial route is accordingly updated. Steps (ii) and (iii) are repeated until no further tasks can be inserted. (iv) The resulting routes are post-processed and inserted as initial columns of the reduced master problem. Note that this procedure can still lead to infeasible schedules, as tasks may remain unassigned. In this case, artificial columns are inserted in the master problem for each unassigned task.

(H1) Partial pricing Each $k - d$ subproblem focuses on the route of a specific team on a specific day, and thus, only the tasks corresponding to this subproblem are considered. Depending on the task time windows or on the skill configuration of the team, it is possible for multiple teams to be assigned to these tasks on the same day with the same route cost. In terms of the column generation procedure, it means that the marginal values β_{md} and ρ_{kd} have zero values, *i.e.,* there is no prize to earn nor penalty to pay by dispatching team k on day d before other teams on other days. We can exploit this fact to reduce the number of subproblems in each iteration by solving only those with marginal values different from zero. In case no subproblem satisfies this condition, one is selected randomly, and the column generation phase continues.

(H2) Relaxed dominance rules One way to accelerate the solution of the subproblems is to relax the dominance rules from the exact labeling algorithm. This is done by testing the dominance of only a subset of the labels' resource: for two labels $L_i = (i, T_i, C(T_i), S)$ and $L'_i = (i, T'_i, C'(T'_i), S')$, L_i dominates L'_i if $T_i \leq T'_i$, $C(T_i) \leq C'(T'_i)$. The elementariness of the routes is still required, although it is not guaranteed that an optimal solution is found.

(H3) Partial pricing and partial dominance rules This approach consists of a combination of both previously described heuristics: H1 and H2. Therefore, partial pricing is applied in the column generation phase and relaxed dominance rules are used in the solution of the subproblems.

3.4.3. General procedure

In this section, we outline the proposed branch-and-price algorithm. It mainly consists of a column generation algorithm embedded in a branch-and-bound procedure to guarantee integrality. Note that the same algorithm is used to solve both decompositions presented in Section 3.4.1 and 3.4.2. Therefore, we now use the terms "master problem" and "pricing problem" generically. Additionally, further details on the implementation of the algorithm are presented.

Column generation The column generation process can be described as follows: the algorithm is initialized by the insertion heuristic. The master problem is solved with a set of artificial columns obtained by the introduction of slack variables with a high cost to ensure primal feasibility. Due to their high cost, these variables eventually leave the basis once further feasible columns are obtained. From the solution of the master problem, the marginal values are obtained and passed to the subproblems. New routes (*i.e.,* columns) are obtained by solving the subproblems. If columns are found with negative reduced cost, the column generation process is started again. Otherwise, an optimal solution for the reduced master problem and thus to the master problem is found. If this solution is integer, then it is an optimal solution for the MPTRSP; if not, it corresponds to a valid fractional lower bound. Otherwise branching on fractional variables is required and column generation is applied on each node with its respective fractional bounds.

Preprocessing The pricing problems of either decomposition are solved with only a subset of the original set of tasks. For each subproblem from the day decomposition, only a subset $I_{sub_d} \subset I$ is considered, *i.e.,* the tasks with time windows covering day d. For solving each subproblem of the team-day-decomposition, only a subset $I_{sub_{kd}} \subset I$ of tasks is considered; that is, only tasks with a time window covering day d for which constraint (3.14) for the respective k team is fulfilled. If no task satisfies these criteria, *i.e.,* $I_{sub_d} = \varnothing$ or $I_{sub_{kd}} = \varnothing$, then the respective pricing problem is not considered throughout the column generation phase.

Branching To explore the branching tree a depth-first strategy is used. This allows us to solve each node starting from the feasible solutions found for its parent by storing all columns that comply with the current node's bounds.

Branching on arcs is the branching strategy selected, as branching on the λ variables is inefficient and troublesome to implement (Feillet, 2010). For each node on the branching tree, we define the set $\mathcal{A}'$ as the set of fractional arcs (i, j, k, d) such that $0 < \psi_{ijkd} < 1$, where $\psi_{ijkd} = \sum_{p \in P_d} \varphi_{ijkdp}\lambda_{dp}$ for the day decomposition, and $\psi_{ijkd} = \sum_{p \in P_k d} \varphi_{ijkdp}\lambda_{kdp}$ for the team-day decomposition. Then, the arc $(i, j, k, d)^*$ is selected as:

$$(i, j, k, d)^* \in \underset{(i,j,k,d) \in \mathcal{A}'}{\arg\max} \left\{ c_{ij} \cdot \min(\psi_{ijkd}, 1 - \psi_{ijkd}) \right\}$$

From this node, two successors are generated:

- One node where arc $(ijkd)^*$ is forbidden, *i.e.,* team k cannot visit task j after i on day d. This is done by removing this arc from its respective subproblem's graph, and discard any columns using it.
- Other node where arc $(ijkd)^*$ is fixed, *i.e.,* all other arcs entering j and leaving i for team k on day d, and all arcs entering i and leaving j for $k` \neq k$ or $d' \neq d$ are removed so that team k is forced to go from i to j on day d. Accordingly, columns using these arcs are then discarded from the column pool.

3.5. Numerical experiments

For our numerical experiments, we use artificial data generated from known benchmarks tests as well as real-world data from a external maintenance provider (EMP) to test the performance of the proposed solution approaches. Then, to focus on the impact of the multi-periodicity of the time windows, we conduct a sensitivity analysis varying the length of the time windows. Then, we combine selected test instances to create larger test instances to test the performance of our solution approaches with larger problem sizes.

All solution approaches are implemented in Python 2.7.5 using the Gurobi 5.6.2 solver on a 2.5 GHz Intel Core i7 machine with 8 GB RAM. Additionally, equations (3.7)-(3.11), (3.35), (3.37), and (3.38) are linearized using a big M formulation, with $\mathbf{M} = \sum_{i \in I} t_{oi} + t_{i\bar{o}}$.

3.5.1. Artificial instances

For our first experiments artificial test instances based on known VRPTW benchmarks (Solomon, 1987) were generated. Instances C101 to C109, R101 to R112, and RC101 to RC108 where selected, where C, R, and RC denote clustered customers, randomly located, and semi-clustered customers, respectively. From each instance the first 25 customer locations (*i.e.,* tasks) are selected (in addition to the depot) and their allowed visit days distributed along a 5-day planning horizon in two different variants: first, denoted by suffix *a* customers' time windows are defined for a single-day and evenly distributed through the planning horizon (*i.e.,* five tasks per day); second, denoted by suffix *b*, these customer's time windows are extended such that they span two consecutive days cutting off those which exceed the planning horizon's length. As the goal of these tests is to focus on the impact of the multiperiod time windows on the computational effort, no vehicle capacity nor skill requirements are considered, waiting and overtime are not allowed, and routes are limited only by the length of a working day (defined by the daily time windows of the depot from the original instances). The transportation costs c_{ij} correspond to the Euclidean distance between the two customer locations, and the transportation times are calculated as $t_{ij} = c_{ij} + p_i$, as in Kohl and Desrosiers (1999). Additionally, we consider a workforce of eight homogeneous technicians and no team building decisions are required. A total of 29 a-instances and 29 b-instances were solved setting a time limit of one hour using all three solutions methods presented before: the MIP formulation solved by a Gurobi's branch-and-cut algorithm, branch-and-price with a day decomposition, and branch-and-price with a team-day decomposition.

Table 3.2 shows, for the solutions of the MIP formulation obtained by Gurobi's branch-and-cut algorithm, the objective function value, computation time (in minutes) and the relative gap reported by the solver. For each branch-and-price method, the objective function value, the computation time in minutes, the number of nodes explored, the number of columns generated (*i.e.,* number of pricing problems solved), and the absolute gap is shown.

In general, we observe that single day time windows (a-instances) are easier to solve than multiperiod time windows (b-instances), since allowing time windows to span multiple days increases the time required to solve the problem. This can be observed by the significant increase in computation time for either solution method. In addition, we observe that the branch-and-price algorithm with the team-day decomposition needs to explore more nodes in several cases than with the day decomposition. This implies

Table 3.2.: Artificial tests results

Inst.	MIP			Day Decomposition					Team-day Decomposition				
	Objective	CPU (min)	Gap (%) [a]	Objective	CPU (min)	Nodes	Nr. cols	Gap (%)	Objective	CPU (min)	Nodes	Nr. cols	Gap (%)
C101-25a	336.50	< 1	0.00	336.50	6.90	1	44	0.00	336.50	< 1	1.00	83.00	0.00
C102-25a	335.80	< 1	0.00	335.80	25.34	1	39	0.00	335.80	< 1	1.00	95.00	0.00
C103-25a	307.20	< 1	0.00	307.20	17.30	1	27	0.00	307.20	< 1	1.00	102.00	0.00
C104-25a	302.10	< 1	0.00	302.10	23.18	1	26	0.00	302.10	< 1	1.00	112.00	0.00
C105-25a	336.50	< 1	0.00	336.50	5.93	1	44	0.00	336.50	< 1	1.00	97.00	0.00
C106-25a	336.50	< 1	0.00	336.50	7.02	1	45	0.00	336.50	< 1	1.00	72.00	0.00
C107-25a	330.20	< 1	0.00	330.20	6.75	1	49	0.00	330.20	< 1	1.00	91.00	0.00
C108-25a	329.40	< 1	0.00	329.40	15.07	1	40	0.00	329.40	< 1	1.00	83.00	0.00
C109-25a	302.10	< 1	0.00	302.10	22.66	1	26	0.00	302.10	< 1	1.00	91.00	0.00
R101-25a	856.10	< 1	0.00	856.10	6.33	1	34	0.00	856.10	< 1	1.00	56.00	0.00
R102-25a	788.30	< 1	0.00	788.30	14.46	1	30	0.00	788.30	< 1	1.00	79.00	0.00
R103-25a	729.20	< 1	0.00	729.20	12.58	1	21	0.00	729.20	< 1	1.00	82.00	0.00
R104-25a	703.20	< 1	0.00	703.20	14.88	1	25	0.00	703.20	< 1	1.00	96.00	0.00
R105-25a	799.40	< 1	0.00	799.40	4.62	1	23	0.00	799.40	< 1	1.00	59.00	0.00
R106-25a	753.50	< 1	0.00	753.50	22.28	1	54	0.00	753.50	< 1	1.00	71.00	0.00
R107-25a	788.10	< 1	0.00	788.10	6.86	1	20	0.00	788.10	< 1	1.00	64.00	0.00
R108-25a	674.90	< 1	0.00	674.90	13.94	1	30	0.00	674.90	< 1	1.00	93.00	0.00
R109-25a	725.50	< 1	0.00	725.50	8.93	1	27	0.00	725.50	12.15	1.00	60.00	0.00
R110-25a	686.70	< 1	0.00	686.70	21.96	1	57	0.00	714.30	3.99	38.00	3161.00	0.04
R111-25a	715.50	< 1	0.00	715.50	12.86	1	28	0.00	715.50	1.93	23.00	1747.00	0.00
R112-25a	653.90	< 1	0.00	653.90	21.31	1	38	0.00	653.90	2.59	1.00	2501.00	0.00
RC101-25a	630.44	< 1	0.00	630.44	9.39	1	26	0.00	630.44	< 1	1.00	76.00	0.00
RC102-25a	616.01	< 1	0.00	616.01	17.07	1	24	0.00	616.01	< 1	1.00	95.00	0.00
RC103-25a	585.16	< 1	0.00	585.16	37.87	1	55	0.00	585.16	< 1	1.00	122.00	0.00
RC104-25a	567.14	< 1	0.00	567.14	48.46	1	58	0.00	567.14	< 1	1.00	113.00	0.00
RC105-25a	680.02	< 1	0.00	680.02	19.80	1	28	0.00	680.02	< 1	1.00	86.00	0.00
RC106-25a	602.57	< 1	0.00	602.57	36.35	1	56	0.00	602.57	< 1	1.00	102.00	0.00
RC107-25a	561.37	< 1	0.00	561.37	45.82	1	66	0.00	561.37	< 1	1.00	130.00	0.00
RC108-25a	535.34	< 1	0.00	535.34	30.91	1	42	0.00	535.34	< 1	1.00	236.00	0.00
Avg.		< 1	0.00		18.51	1.00	37.31	0.00		2.54	3.03	339.83	0.00
C101-25b	217.60	T	44.00	514.00	T	2	98	138.29	215.70	13.22	6.00	914.00	0.00
C102-25b	222.10	T	48.00	786.50	T	1	76	265.13	215.40	5.22	3.00	343.00	0.00
C103-25b	225.20	T	57.80	838.00	T	1	87	291.69	213.95	14.08	9.00	874.00	0.00
C104-25b	221.50	T	59.80	882.50	T	1	92	312.50	213.94	5.43	12.00	987.00	0.00
C105-25b	223.80	T	52.37	801.50	T	1	62	319.19	191.20	5.77	2.00	384.00	0.00
C106-25b	224.40	T	49.00	484.85	T	1	137	124.78	215.70	5.07	2.00	359.00	0.00
C107-25b	201.50	T	47.15	801.00	T	1	57	309.51	195.60	4.70	2.00	403.00	0.00
C108-25b	191.00	T	49.79	810.20	T	1	41	324.41	190.90	3.93	3.00	381.00	0.00
C109-25b	190.30	T	53.70	810.58	T	1	19	325.95	190.30	3.86	4.00	419.00	0.00
R101-25b	613.70	T	43.00	620.00	T	2	221	1.03	613.70	4.40	17.00	2758.00	0.00
R102-25b	559.80	T	10.65	559.80	45.13	1	213	0.00	559.80	< 1	3.00	244.00	0.00
R103-25b	537.80	T	12.72	987.80	T	1	216	83.67	537.80	< 1	3.00	216.00	0.00
R104-25b	529.40	T	11.17	529.40	42.82	1	204	0.00	529.40	3.58	10.00	2337.00	0.00
R105-25b	572.90	T	9.80	578.00	16.52	1	173	0.89	572.90	4.05	14.00	2598.00	0.00
R106-25b	549.20	T	13.27	552.90	38.72	1	211	0.67	549.20	3.78	16.00	1861.00	0.00
R107-25b	555.30	T	22.37	524.60	40.22	1	207	0.00	524.60	1.96	20.00	1151.00	0.00
R108-25b	508.60	T	15.59	512.30	34.34	1	171	1.84	503.05	3.09	27.00	2131.00	0.00
R109-25b	532.30	T	9.98	534.85	T	2	169	0.48	532.30	3.55	11.00	2061.00	0.00
R110-25b	512.30	T	11.91	512.30	26.49	1	196	0.00	512.30	< 1	4.00	190.00	0.00
R111-25b	520.90	T	12.19	987.80	T	1	240	89.63	520.90	2.91	14.00	1707.00	0.00
R112-25b	512.30	T	17.31	510.90	60.00	1	239	0.00	510.90	< 1	4.00	267.00	0.00
RC101-25b	3606.76	T	49.07	909.28	T	1	118	151.45	361.61	1.99	3.00	314.00	0.00
RC102-25b	3606.80	T	64.28	1258.96	T	1	36	253.31	356.33	1.89	4.00	295.00	0.00
RC103-25b	3606.63	T	70.18	1228.56	T	1	57	252.35	348.67	2.08	4.00	343.00	0.00
RC104-25b	3606.31	T	70.30	1063.34	T	1	31	220.06	332.24	1.47	3.00	351.00	0.00
RC105-25b	3606.60	T	62.14	1240.59	T	1	45	242.66	362.05	2.21	2.00	395.00	0.00
RC106-25b	3606.65	T	68.00	908.26	T	1	89	156.49	354.11	1.69	4.00	346.00	0.00
RC107-25b	3606.61	T	71.00	1190.69	T	1	44	258.39	332.24	1.50	3.00	382.00	0.00
RC108-25b	3606.59	T	70.36	1187.50	T	1	31	257.43	332.24	1.27	3.00	396.00	0.00
Avg.		T	40.58		38.03	1.10	123.45	151.10		3.58	7.31	876.10	0.00

[a] Gap to best known bound found by solver

that the team-day decomposition provides a worse linear lower bound thus requiring additional branching in order to close the integer optimality gap. This observation indicates that we face a trade-off by choosing a decomposition with smaller, easier-to-solve pricing problems but which may require more iterations to provide integer solutions.

We also observe that, for the *a*-instances the team-day decomposition on all its variants is clearly outperformed by the other two exact solution methods. That is because the team-day decomposition considers many subproblems which could be aggregated due to the workforce homogeneity (see *e.g.*, R109-25a to R112-25a). This, however, is not the case for the *b*-instances as the team-day decomposition shows the best performance among the solution approaches regarding computation and solution time. For instance, with the team-day decomposition all instances could be solved to optimality compared to only 7 and none with the day decomposition and the MIP formulation, respectively.

3.5.2. Real-world instances

In order to test for performance under real settings, we solve real-world instances with three solution methods: solving the MIP formulation directly Gurobi's branch-and-cut algorithm, the branch-and-price approach with a day decomposition, and the branch-and-price approach with a team-day decomposition. The data consist of 44 instances, each corresponding to the maintenance requests of a single week. For each task, information such as the customer location, service time, time windows, and skill requirements were provided by the company. A total of 66 different customers are considered, and the estimated traveling time to each of their location corresponds to the total travel time of the recommended driving route obtained from the Google Maps API (Google, 2016). Note that, since this route includes changes in elevation due to the topography of the respective area, the distance between locations is not Euclidean. Table 3.3 shows a summary of the characteristics (number of tasks, min., max., and avg. service time, min., max., and avg. time window length, max. number of time windows, and number of tasks with multiple time windows) from the considered instances. The maintenance provider's staff consists of six technicians qualified in five skill domains with three possible proficiency levels paired into teams of size $\tau = 2$. In addition, based on maintenance contracts and financial information from the company, the following cost parameters are considered: $w^{cost} = 300$, $ot^{cost} = 40$, and $c_{ij} = 60 \cdot t_{ij} + p_i$ in monetary units. Finally, regarding customer waiting and overtime limits, the maintenance provider uses $w^{max} = ot^{max} = 2$ hours.

In Table 3.4, the results obtained within a time limit of three hours for all methods are shown. For each method, the objective value (in monetary units), the computation time (in minutes), and the optimality gap is presented for each of the test instances. Note that for the case of the MIP solver solutions, this column shows the gap to the best bound found by the solver within the given time limit. For the methods involving branch-and-price, the number of explored nodes while branching and the number of columns are also shown. In case the time limit is reached (denoted with a "T"), the best integer solution found is shown. If no integer solution could be found, we report the integer upper bound of the reduced master problem with the columns found until reaching the time limit.

From these results, we can make several observations. First, the MIP solver is able to solve only 18 instances to optimality, while in 26 cases, the

Table 3.3.: Description of real-world instances

Inst.	Tasks	Service Time (h)			TW length (h)			Max.	Multi-TW
		Min.	Avg.	Max	Min.	Avg.	Max	TW's	tasks
1	15	0.5	1.5	3.0	7.0	7.9	8.0	2	5
2	20	0.5	1.4	3.0	5.0	7.4	8.0	2	2
3	18	0.5	1.1	2.0	6.0	7.1	8.0	1	0
4	25	0.5	2.3	4.0	6.0	18.1	40.0	5	8
5	24	0.5	1.9	5.0	6.0	12.9	40.0	5	4
6	22	0.5	2.1	4.0	6.0	14.8	40.0	5	5
7	22	0.5	2.0	4.0	7.0	15.2	40.0	5	12
8	23	0.5	1.5	5.0	6.0	7.0	8.0	1	0
9	18	0.5	1.5	3.0	6.0	7.1	8.0	1	0
10	22	0.5	1.6	3.0	6.0	19.3	40.0	5	8
11	29	0.5	1.7	4.0	6.0	15.4	40.0	5	9
12	17	0.5	1.4	3.0	5.0	15.0	40.0	5	4
13	26	0.5	1.7	6.0	6.0	12.6	40.0	5	4
14	19	0.5	1.7	3.0	6.0	7.4	8.0	1	0
15	21	0.5	1.5	5.0	6.0	10.3	40.0	5	3
16	31	1.0	1.7	5.0	6.0	15.5	40.0	5	8
17	25	0.5	1.6	4.0	8.0	17.0	40.0	5	2
18	18	0.5	1.8	5.0	8.0	15.1	40.0	5	4
19	26	0.5	1.5	5.0	7.0	12.7	40.0	5	4
20	18	0.5	1.5	4.0	7.0	13.0	40.0	5	3
21	21	0.5	2.0	7.5	4.0	9.1	40.0	5	1
22	12	0.5	1.4	4.0	6.0	12.5	40.0	5	2
23	22	0.5	1.3	3.0	6.0	7.0	8.0	5	6
24	20	0.5	2.4	6.0	6.0	22.3	40.0	5	9
25	14	0.5	1.6	3.5	6.0	16.6	40.0	5	4
26	17	0.5	2.5	7.5	6.0	15.2	40.0	5	4
27	15	0.5	1.6	3.0	6.0	13.8	40.0	5	3
28	10	0.5	1.8	3.0	4.0	10.3	40.0	5	1
29	22	0.5	2.1	5.0	5.0	10.6	40.0	5	2
30	14	0.5	1.8	3.0	4.0	7.4	8.0	1	0
31	27	0.5	1.6	4.0	6.0	18.4	40.0	5	9
32	15	0.5	1.4	3.0	6.0	16.3	40.0	5	5
33	22	0.5	1.6	4.0	6.0	13.5	40.0	5	4
34	20	0.5	1.5	3.0	5.0	12.5	40.0	5	3
35	12	1.0	1.8	5.0	5.0	13.1	40.0	5	5
36	21	1.0	1.7	3.0	5.0	18.2	40.0	5	7
37	24	0.5	1.3	3.0	5.0	19.8	40.0	5	9
38	18	1.0	1.5	3.0	5.0	14.5	40.0	5	4
39	19	0.5	1.2	3.0	4.0	13.7	40.0	5	4
40	17	0.5	1.1	2.0	6.0	13.1	40.0	5	3
41	17	1.0	2.1	4.0	5.0	9.5	40.0	5	1
42	14	0.5	2.1	4.0	4.0	7.2	8.0	2	1
43	8	1.0	1.8	3.5	5.0	7.4	8.0	2	1
44	11	1.0	1.3	3.0	4.0	6.8	8.0	1	0

computation time limit is reached. For these cases, the objective value shows a relatively high gap to the best bound found by the solver (average of 12%), although in some cases the solution found is the optimal solution (*e.g.,* instances 18, 32, 36, and 39). In the 18 instances with 0% gap, the branch-and-cut procedure from the solver found the optimal solution in less than one minute, with the exception of instances 22, 25, 26, and 40 where the optimal solution was found in 0.2, 4.7, 36.8, and 3.1 minutes, respectively. This indicates that the commercial solver spends ta great deal of time on proving this solution's optimality.

Second, the branch-and-price with a day decomposition is clearly outperformed by the other two methods. Within the set time limit only in 13 cases, an optimal solution is reported and no feasible solution for instance 24 is found. For all of the remaining 40 instances where the computation time limit is reached, the algorithm stopped during the column generation phase at the root node. The solutions reported correspond to the integer solution of the reduced master problem using the columns obtained up until the time of interruption, which explains the reason why the optimality gap is so high in some cases (see, *e.g.,* instance 11, 24, and 10). This low performance is due to the high computational effort required to solve the pricing problems in the day decomposition. Furthermore, the frequency with which the pricing problems are solved has an important impact on the performance. In the case of instances with tasks with none or few multiple time windows, the computation time is low since tasks can be assigned to few possible validity periods, thus, few iterations of the column generation phase are required. On the other hand, instances with high number of tasks with multiple periods require more time to solve as the number of combinations of possible validity days increases, thus greatly incrementing the number of iterations of the column generation phase. However, we observe that using this decomposition has the advantage that it provides better bounds than the team-day decomposition, as in all the cases where an optimal solution is obtained, it is found at the root node. Thus, there is a clear trade-off between the computational effort spent into solving larger pricing problems and the savings while exploring the branching tree due to tighter bounds.

Third, the team-day decomposition clearly outperforms the other solution methods, because it is able to find a proven optimal solution for all cases in an acceptable computation time. With a maximum of 16.25 minutes, the team-day-decomposition solves this weekly planning problem for all instances to optimality in short time.

3.5.3. Impact of the length of the time windows

The purpose of the following experiments is manyfold: we further test the performance of our algorithms with additional test instances, and we test the impact of the time window length on the objective function value and computation time. Note that we refer to window length as the number of days covered by a single time window, and not exclusively the number of periods covered by it. For this purpose, we generate 45 test instances by selecting different instances from Section 3.5.2, and multiplying their time window length by a scaling factor in the interval $[0, 2]$. The nine selected weeks correspond to the instances with:

- Type A: Min., max., and average time window length (Week 44, 24, and 13, respectively).

Table 3.4.: Real-world tests results

Inst.	MIP			Day Decomposition					Team-day Decomposition				
	Objective	CPU (min)	Gap (%) [a]	Objective	CPU (min)	Nodes	Nr. cols	Gap (%)	Objective	CPU (min)	Nodes	Nr. cols	Gap (%)
1	737.69	< 1	0.00	737.69	< 1	1	39	0.00	737.69	< 1	1	68	0.00
2	861.78	< 1	0.00	861.78	< 1	1	56	0.00	861.78	< 1	1	51	0.00
3	1320.78	< 1	0.00	1320.78	< 1	1	54	0.00	1320.78	< 1	1	48	0.00
4	462.22	T	35.00	1136.97	T	1	22	152.52	450.26	< 1	1	541	0.00
5	470.65	T	24.24	464.00	72.03	1	153	0.00	464.00	16.49	1	243	0.00
6	519.61	T	26.00	1211.55	T	1	43	134.96	515.63	< 1	1	165	0.00
7	519.86	T	20.50	871.01	T	1	74	68.69	516.33	1.38	27	18144	0.00
8	1506.18	< 1	0.00	1506.18	< 1	1	42	0.00	1506.18	< 1	1	80	0.00
9	461.53	< 1	0.00	461.53	< 1	1	15	0.00	461.53	< 1	1	100	0.00
10	406.05	T	28.90	1030.57	T	1	62	157.60	400.06	< 1	1	472	0.00
11	553.31	T	32.20	1472.55	T	1	22	177.52	530.61	4.54	1	2632	0.00
12	355.42	T	8.79	354.99	T	1	110	0.56	353.02	< 1	43	6765	0.00
13	472.71	T	27.50	470.04	T	1	126	3.33	454.91	1.22	1	938	0.00
14	367.49	< 1	0.00	367.49	< 1	1	26	0.00	367.49	< 1	19	287	0.00
15	368.17	T	14.63	368.17	29.20	1	145	0.39	366.74	< 1	28	16408	0.00
16	615.46	T	23.61	1495.07	T	1	24	150.06	597.88	1.93	1	275	0.00
17	412.14	T	32.57	777.19	T	1	107	91.92	404.95	< 1	1	165	0.00
18	356.63	T	9.00	356.63	2.37	1	89	0.00	356.63	< 1	1	276	0.00
19	872.63	T	14.64	1106.77	T	1	87	28.10	864.01	1.16	1	323	0.00
20	387.17	T	12.90	-	T	-	-	-	380.32	1.11	50	14750	0.00
21	775.06	T	4.00	775.06	82.15	1	95	0.24	773.17	2.30	61	18279	0.00
22	272.19	1.23	0.00	272.19	< 1	1	59	0.00	272.19	< 1	1	39	0.00
23	875.83	< 1	0.00	875.83	5.52	1	70	0.00	875.83	< 1	59	5754	0.00
24	464.78	T	31.36	1225.30	T	1	32	166.82	459.23	< 1	1	337	0.00
25	285.74	11.52	0.00	285.74	18.82	1	139	0.00	285.74	1.98	1	81	0.00
26	425.18	152.10	0.00	425.18	4.52	1	86	0.00	425.18	< 1	1	79	0.00
27	284.79	< 1	0.00	284.79	< 1	1	75	0.00	284.79	< 1	1	59	0.00
28	205.02	< 1	0.00	205.02	< 1	1	29	0.00	205.02	< 1	1	26	0.00
29	562.39	T	19.82	562.39	3.92	1	89	7.00	525.62	< 1	65	531	0.00
30	294.49	< 1	0.00	294.49	< 1	1	28	0.00	294.49	< 1	1	33	0.00
31	463.36	T	26.32	1127.17	T	1	43	147.58	455.28	< 1	1	274	0.00
32	350.07	T	8.79	350.07	< 1	1	101	0.00	350.07	< 1	1	76	0.00
33	379.75	T	25.34	538.81	T	1	83	51.17	356.42	< 1	1	126	0.00
34	394.73	T	23.46	394.73	137.54	1	109	0.79	391.63	1.09	60	3060	0.00
35	750.76	< 1	0.00	750.76	< 1	1	27	0.00	750.76	< 1	1	28	0.00
36	336.77	T	28.00	822.46	T	1	50	144.22	336.77	16.25	1	210	0.00
37	305.21	T	32.40	643.50	T	1	44	113.46	301.46	1.13	1	184	0.00
38	268.33	T	19.00	269.36	T	1	119	1.98	264.14	< 1	26	9828	0.00
39	299.98	T	9.00	299.98	3.67	1	164	0.00	299.98	1.79	1	167	0.00
40	303.00	17.46	0.00	303.00	< 1	1	96	0.00	303.00	< 1	1	133	0.00
41	396.26	T	13.57	396.26	150.05	1	84	0.65	393.69	2.07	75	126675	0.00
42	358.99	< 1	0.00	358.99	< 1	1	45	0.00	358.99	< 1	76	11856	0.00
43	278.73	< 1	0.00	278.73	< 1	1	16	0.00	278.73	< 1	1	15	0.00
44	257.99	< 1	0.00	257.99	< 1	1	16	0.00	257.99	< 1	1	28	0.00
Avg.[b]		10.24	12.53		19.07	1	70	37.20		1.41	14	5468	0.00

[a]Gap to best known bound found by solver
[b]The average CPU does not consider instances where the time limit is reached

- Type B: Min., max., and average service time (Week 40, 26, and 11, respectively).
- Type C: Min., max., and average number of tasks per week (Week 43, 26, and 14, respectively).

If the working day length is exceeded, a new time window is started on the next consecutive day. In case the length of the planning horizon is exceeded, the length of the last time window is cut out. To solve these instances, we select the team-day decomposition, as it shows the best performance in the results from previous tests.

Figures 3.1 and 3.2 show the objective function value and computation time resulting from each scaling factor for each of the selected instances. In these graphs, we observe the following effects. First, the overall costs are decreasing when increasing the time window length. The wider the time windows get, the lower their impact is on the reduction of overall costs. Second, we observe an increase in the computation time of all instances when increasing the time window length. Since these effects can also be observed in single-period time windows, the observed results show that the same behavior can be expected in the multi-period case. Nevertheless, these effects did not present the same magnitude in all instances (see *e.g.,* C avg. and C max.). This can be explained by the fact that the selected weeks show a high number of multiple requests per customer, *i.e.,* there are significantly more tasks than locations to visit, which results in high symmetry.

3.5.4. Performance with larger instances

To further test the performance of our solution methods, we generate larger artificial instances from the afore-described real-world data. Three instances were obtained by combining the demand of three different weeks and tripling the workforce. Each instance is generated by combining weeks (I) A min., A avg., and A max., (II) B min., B avg., and B max., and (III) C

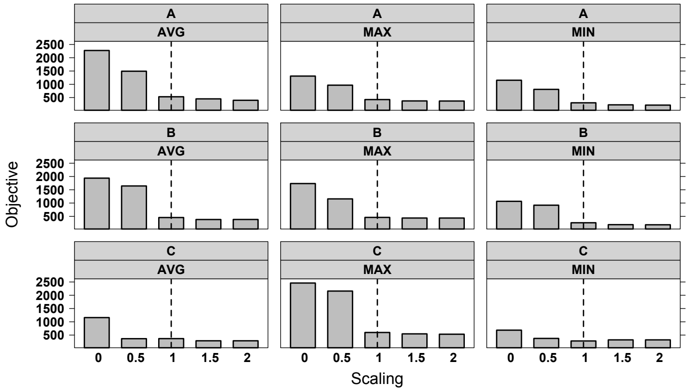


Figure 3.1.: TW length vs. costs

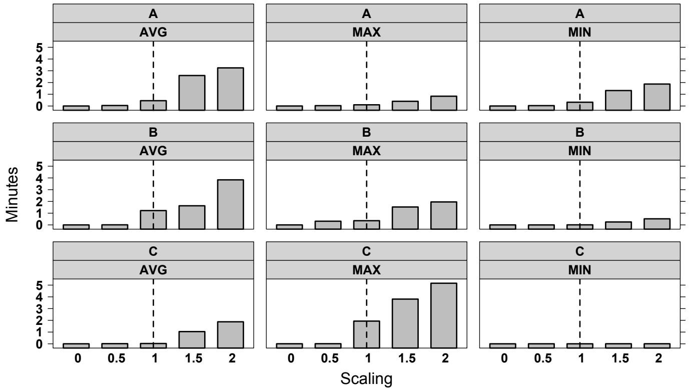


Figure 3.2.: TW length vs. CPU time

min., C avg., and C max. All three instances are solved with all proposed solution methods and within a time limit of 10 hours.

Table 3.5 shows the objective function value, computation time (in minutes), and optimality gap (in percentage) for all instances and solution methods. For the day-decomposition and team-day decomposition, in addition, the number of explored nodes and number of columns is shown. Here, we note that once again the team-day decomposition outperforms the other solution methods. In the case of the MIP, the solver reached the time limit on all three cases and was not able to find even a feasible solution for instance III. The day-decomposition also reached the computation time limit on all instances during the column generation phase. Similar to Section 3.5.2, the reported solution corresponds to the integer solution of the reduced master problem using the columns found up until the time break, which explains the large optimality gap. The team-day decomposition, however, is able to solve all instances to optimality within a reasonable amount of computation time (all cases under 45 minutes).

Table 3.5.: Results for larger instance tests

Inst.	MIP			Day Decomposition					Team-day Decomposition				
	Objective	CPU (min)	Gap (%) [a]	Objective	CPU (min)	Nodes	Nr. cols	Gap (%)	Objective	CPU (min)	Nodes	Nr. cols	Gap (%)
I	1313.00	T	70.41	2068.96	T	1	207	100.20	1033.46	42.00	18	2790	0.00
II	1795.63	T	72.15	1707.65	T	1	140	68.07	1016.04	44.91	25	5406	0.00
III	-	T	-	2107.17	T	1	48	98.51	1061.52	9.90	12	642	0.00
Avg.		T	71.28		T	1.00	218.22	88.92		32.27	18.33	2946.00	0.00

[a]Gap to best known bound found by solver

3.6. Conclusions and further research

This paper addresses a multiperiod technician routing and scheduling problem that is not previously considered in the literature. Our main contributions are the definition of this new problem, its formulation as a decision model, and the development of two different decomposition schemes implemented in a branch-and-price algorithm to solve the problem to optimality.

These solution approaches are further tested with data from an external maintenance provider (EMP).

The results of our tests show that our solution approaches are able to solve the majority of instances for this complex problem to optimality within an acceptable amount of time. However, we observe that this performance greatly depends on the selected decomposition scheme used in the branch-and-price algorithm. On the one hand, using a decomposition with fewer, harder-to-solve subproblems provides tighter fractional lower bounds, although it is computationally more expensive. On the other hand, using a decomposition with more, easier-to-solve subproblems may result in worse fractional lower bounds, although it allows us to obtain optimal solutions for all of our test instances in a shorter amount of time. This also holds for larger problem instances, as our experiments indicate. Furthermore, our sensitivity analysis shows that additional flexibility, in the form of length (and number) of time windows, has a direct impact on operations costs. This fact needs to be considered during maintenance contract negotiations.

Further research needs to be conducted to incorporate our current decision model as part of a hierarchical planning process such that, as a first step, the weekly plans obtained through our approaches are used to announce visit times to customers. As a second step, the EMP management would oversee the execution of these plans on a daily basis and adjust them in case unforeseen events occur. Additionally, other features such as technicians' preferences, planned meal breaks, balanced workloads, etc. can also be included. As for the proposed solution methods, further research is needed to improve their performance by applying ideas used in the VRP literature for this purpose, *e.g.,* relaxation of the subproblems, applying cuts in the branching trees, other branching strategies, stabilization strategies, intensification strategies, etc. Additionally, other hybrid and heuristic approaches could be adapted for this problem.

Acknowledgements

This research was supported by the Erich-Becker-Stiftung, a foundation of Fraport AG. The authors are grateful to the anonymous referees for helpful and constructive suggestions.

4. Task assignment with start time-dependent processing times for personnel at check-in counters

Co-authors:

Annika Becker and **Raik Stolletz**
Chair of Production Management, Business School, University of Mannheim, Germany

Working paper, under consideration for publication.

Abstract:

This paper addresses a task-assignment problem encountered by check-in counter personnel at airports. The problem consists of assigning multi-skilled agents to a sequence of tasks in check-in counters. Because each task's ending time is fixed to comply with the flight schedule, its processing time depends on the arrival of the assigned agents. We propose a mixed-integer program and a branch-and-price algorithm to solve this problem. We exploit the problem structure to efficiently formulate the pricing problems and improve computation time. Using real-world data from a German ground-handling agency, we conduct numerical studies to evaluate the performance of the proposed solutions.

4.1. Introduction

This paper addresses a task assignment problem encountered by ground-handling agencies at airports, where employees are assigned to process flight check-ins and boardings (*i.e.*, tasks). Because these agencies provide the manpower required for multiple airlines' flights, they need to ensure that sufficient staff is available in a timely fashion for all flights; otherwise, they may incur penalties for violating their contracts with the airlines. Furthermore, as 66% to 75% of a ground-handling agency's operation costs correspond to personnel costs (Steer Davies Gleave, 2010), inadequate workforce plans (*e.g.,* overstaffing) are highly undesirable.

The basic structure of a ground-handling agency's workforce planning process can be divided into four planning stages (Stolletz, 2010): (i) head count planning, (ii) tour scheduling, (iii) task assignment, and (iv) replanning. Our work addresses the third stage of this process, during which tasks are assigned to employees on a daily basis taking into account the agent's skills and availability, traveling time between counters, flight schedules, and contract conditions with the airlines. The specific characteristics of the present problem are described below.

The demand for agents is caused by both the flight schedules and the contracts between the airlines and the ground-handling agencies. The flight schedules provide daily flight-departure times and therefore, the times when demand occurs. They also provide information about the assignment of flights to counters, which in turn implies that the transportation time between the tasks' locations is known. The contracts with the airlines determine the agent requirements (*i.e.*, number of agents and skills required), along with the counters' opening and closing times for processing tasks. As can be observed in practice, agents are allowed to arrive at a counter after its opening time, albeit at a cost, because this tardiness is penalized in the contracts with the airlines. Regardless of an agent's tardiness, however, he or she is required to remain at the assigned counter until its closing time. This means that all tasks' end times are fixed and, thus, their duration is dependent on the assigned agent's arrival times.

The following issues are taken into consideration when managing the ground-handler's workforce. First, agent availability is derived from the shift schedules and days-off schedules obtained in the preceding planning stage, *i.e.*, tour scheduling. As explained in Stolletz (2010) and Stolletz and Zamorano (2014), shifts can have varying durations, can start at different hours of the day, and can vary from day to day. Although shift-planning decisions cannot be influenced by the task assignment, overtime is allowed if the completion of the final task fulfilled exceeds the length of the shift (which is subject to labor regulations). Second, agents are qualified to operate various airlines' check-in systems, *i.e.*, the workforce is multiskilled, and they may change the check-in system in which they are working during the course of a shift. Neither hierarchical skills nor proficiency levels are considered. If no employee is available to process a task, a limited number of fully qualified supervisors and/or management personnel can be ”outsourced” from the back office to cover the demand, although this is undesirable because it distracts the supervisors and managers from their normal responsibilities.

The goal of this planning problem is to obtain daily schedules consisting of task assignments for each agent for a planning horizon of one day. These assignments define a sequence of tasks for each agent, because the tasks' end times are fixed. Additionally, these schedules need to minimize the weighted sum of traveling time, overtime, outsourced time, and tardiness.

The contribution of this paper is manifold:

- We address a new task assignment problem that incorporates variable task duration and fixed end time;
- We propose an exact branch-and-price solution approach to solve this problem in short computation time; and
- We test our proposed approach with real-world data.

The remainder of the paper is organized as follows. In Section 4.2, related literature on similar task assignment problems is presented. The task assignment process for check-in counters is explained in detail and the algebraic notation of the mixed-integer programming model is presented in Section 4.3. In Section 4.4, the branch-and-price algorithm is described. In Section 4.5, the numerical analysis of a real-world example is conducted. Conclusions and directions for further research are presented in Section 4.6.

4.2. Literature review

This paper addresses a problem that can be classified into different streams of literature because of the different features associated with various applications of the problem. The allocation of tasks to agents corresponds to task assignment problems, although the consideration of changeover times (or traveling time) is also related to routing problems. In the following section, we present a comparison of the related literature from different application areas of task assignment, task scheduling, and routing problems, describing similarities and differences compared to the presented problem.

Task assignment problems are addressed in several workforce-planning settings. Generally, task assignment (also referred to as task allocation) consists of assigning a particular number of employees to perform a particular number of tasks (Edison and Shima, 2011). Typically, such problems involve the mere allocation of employees to tasks (see, *e.g.*, Liang and Buclatin, 1988; Miller and Franz, 1996; Campbell, 1999; Campbell and Diaby, 2002; Krishnamoorthy et al., 2012; Liu et al., 2013; Smet et al., 2014). During the last decade, research has generally not addressed task assignment problems but has focused on task scheduling problems.

In *task scheduling problems*, both the assignment of employees to tasks and the start time of a task are part of the decision. Optionally, there could be a time window during which an employee is allowed to start a task. In these problems, the processing time is given as input and it can be considered either fixed or employee-dependent. In either case, this means that the end times of the tasks become dependent variables.

Loucks and Jacobs (1991), Corominas et al. (2006), Cordeau et al. (2010b), and Lieder et al. (2015) provide examples of task-scheduling literature that assumes fixed task durations, albeit without considering routing decisions. Conversely, Li et al. (2005), Eveborn et al. (2006a), Dohn et al. (2009a), and Kovacs et al. (2012) include routing in their models. Dohn et al. (2009a) address a decision problem similar to ours regarding ground personnel at airports. In their model, employees with different skills are assigned to tasks (*e.g.*, baggage handling, check-ins, ticketing). As in our model, tasks may require more than one agent for their completion, based on the task requirements. Unlike our model, all of the agents assigned to a task are forced to start it at the same time (*i.e.*, no tardiness is allowed). The authors propose a branch-and-price approach to solve this problem and test it using real-world data from a European airport.

Other studies in the task-scheduling literature assume that the duration of a task (*i.e.*, processing time) depends on the employee fulfilling the task. The models of Corominas et al. (2010) and Olivella et al. (2013) assume that the duration of a task depends on the worker's experience and that future performance of the task can be increased. However, these two models do not include routing. With respect to similar applications that include routing, Caseau and Koppstein (1992) present an approach to solve the technician-assignment problem in the telephone industry, assuming that the duration of a task depends on the efficiency ratio based on the technician's skills. Yang (1996) and Tsang and Voudouris (1997) provide solutions to the workforce management problem at British Telecom. Engineers are allocated to jobs taking into account that job duration depends on the engineer's effectiveness rate and skill level, respectively.

Table 4.1 summarizes previous research based on the characteristics of the tasks. A task's starting time, processing time, and ending time may be inputs for the various decision models, decision variables or dependent decision variables. The assignment of an employee to a task is the main decision variable.

Existing task assignment models differ from our present problem as follows: first, they do not consider changeover times (*e.g.*, traveling time) and thus no routing decisions are made; second, starting, ending, and processing times are given, which leaves the assignment of employees to tasks as the only decision to be made. In task-scheduling problems, the end times of the tasks are dependent variables. In our problem, the task duration depends on the agent's arrival time at the task because the end time is given by the flight schedule. We are unaware of decision-problem literature that considers the start times of the tasks as decision variables when end times are given as inputs.

The proposed problem is also related to the *Vehicle Routing Problem with Time Windows* (VRPTW). The VRPTW addresses the problem of supplying a set of customers (*i.e.*, check-in counters) with a set of delivery vehicles (*i.e.*, agents) when the customers' locations, preferred visit times (*i.e.*, flight schedules), and demand quantities (*i.e.*, number of agents required) are known (Solomon and Desrosiers, 1988b; Cordeau et al., 2001). It solves two decisions simultaneously; the first decision involves determining the routes for the vehicles (*i.e.*, sequence of tasks processed by an agent) and the second decision involves assigning vehicles to customers (*i.e.*, assignments of agents to tasks). An additional variant of the VRPTW, the heterogeneous fleet VRP (h-VRP), assumes that vehicles vary with respect to their capacities and costs (Ferland and Michelon, 1988; Choi and Tcha, 2007; Pessoa et al., 2009; Jiang et al., 2014). Our problem, however, generalizes the classical VRPTW and the h-VRP as it considers a heterogeneous vehicle fleet both because the agents have different skills and because the task duration is dependent on the task's starting time.

Because of these similarities, we note that the applications mentioned before have adapted VRP solution approaches for related workforce-planning problems. They differ from our problem, however, in that the task duration is a parameter known in advance that remains fixed throughout the planning process, whereas in our problem the task duration depends on the tasks' starting time decision. Nevertheless, our problem structure allows us to apply solution approaches typical of the VRP literature (*e.g.*, Desrochers et al.

Table 4.1.: Positioning of the present paper in the literature

Category	Start time	Processing time	End time	Assignment	References
Task assignment	Input	Input	Input	Decision	Liang and Buclatin (1988), Miller and Franz (1996), Campbell (1999), Campbell and Diaby (2002), Krishnamoorthy et al. (2012), Liu et al. (2013), Smet et al. (2014)
Task scheduling	Decision	Input (fixed or variable)	Dependent	Decision	Loucks and Jacobs (1991), Caseau and Koppstein (1992), Yang (1996), Tsang and Voudouris (1997), Li et al. (2005), Corominas et al. (2006), Eveborn et al. (2006a), Dohn et al. (2009a), Cordeau et al. (2010b), Corominas et al. (2010), Kovacs et al. (2012), Olivella et al. (2013), Lieder et al. (2015)
Our model	Decision	Dependent	Input	Decision	-

(1992); Lübbecke and Desrosiers (2005); Dohn et al. (2009a); Liberatore et al. (2010)) to solve the current problem.

4.3. Problem description and model formulation

The considered problem consists of assigning each agent $k \in K$ to daily routes to perform all tasks $i \in I'$. Each task requires a number of v_i agents and is associated with an opening time a_i and a closing time b_i. Skill requirements are represented by the binary parameter r_{iq} with a value of 1 if the task requires one agent with skill q. Additionally, let the set I be the set of tasks that includes the depot (represented by o and $\bar{o}$) and N the set of arcs (i, j) such that $i, j \in I$. Finally, t_{ij} represents the traveling time from the counter of task i to the counter of task j. We assume that the triangle inequality is satisfied. This also holds for the case in which tasks i and j are in the same location because the processing times are non-negative.

Workforce qualifications are represented by the binary parameter m_{kq} with a value of 1 if agent k has skill q and 0 otherwise. Note that this formulation can also accommodate hierarchical skills and/or proficiency levels, because only the values of m_{kq} need to be changed for this purpose. Furthermore, every day, every agent departs and arrives at the depot during his/her working shift $[e_k, f_k]$ (plus overtime).

Six decision variables are used. First, the binary variable x_{ijk} assigns a sequence of tasks to an agent:

$$x_{ijk} = \begin{cases} 1, & \text{if agent } k \text{ goes directly from task } i \text{ to task } j \\ 0, & \text{otherwise.} \end{cases}$$

Note that for agents with no assignments, empty routes ($x_{o\bar{o}k} = 1$) are allowed.

Second, binary variable y_{ik} assigns tasks to agents to simplify the notation:

$$y_{ik} = \begin{cases} 1, & \text{if task } i \text{ is assigned to agent } k \\ 0, & \text{otherwise.} \end{cases}$$

Third, outsourcing agents is allowed at a cost. The variable out_i represents the number of outsourced agents for task i, with a maximum value of out_i^{max}.

Fourth, the arrival time of agent k at task i is defined by the positive variable s_{ik}. Fifth, a task can be processed only after its opening time a_i, *i.e.*, if an agent arrives earlier, he/she waits at no cost. If an agent starts the task after a_i, we penalize the delay using the positive variable w_{ik}, defined as

$$w_{ik} = \begin{cases} \max(0, s_{ik} - a_i), & \text{if } y_{ik} = 1 \\ 0, & \text{otherwise,} \end{cases}$$

with a maximum delay of w^{max}.

Finally, if an agent arrives at the depot at the end of the route after the end of his/her shift, overtime is incurred. We model this by the use of the positive, dependent variable ot_k defined as

$$ot_k = \max(0, s_{\bar{o}k} - f_k),$$

with a maximum overtime ot_k^{max}. Table 4.2 shows the notation used for this formulation.

The objective is to minimize the weighted sum of overall traveling time, tardiness, overtime, and outsourced time, *i.e.*, the amount of time each counter opened with outsourced personnel. Traveling time is a concern for the agents, who prefer working at counters that are close together. Tardiness, overtime, and outsourced agent-minutes directly affect the ground-handling agency because they implicate the agency's financial interest. The present problem can be formulated as a mixed-integer programming model as follows:

$$\text{Minimize} \quad \underbrace{\alpha_1 \cdot \sum_{(i,j)\in N} \sum_{k\in K} t_{ij} \cdot x_{ijk}}_{\text{Traveling time}} +$$

Table 4.2.: Notation

Sets	
I'	Set of tasks
I	Set of tasks $I = I' \cup \{o, \bar{o}\}$
N	Set of arcs $N \subseteq \{(i,j) \| i,j \in I\}$
K	Set of agents
Q	Set of skills
Parameters	
$[\alpha_1, ..., \alpha_4]$	Objective function weights
t_{ij}	Traveling time from task i to task j
a_i	Opening time of task i
b_i	Closing time of task i
v_i	Number of agents required for task i
r_{iq}	1 if task i requires skill q, 0 otherwise
m_{kq}	1 if agent k has skill q, 0 otherwise
$[e_k, f_k]$	Start and end of shift of agent k
w_i^{max}	Maximum allowed tardiness for task i
ot_k^{max}	Maximum overtime for agent k
out_i^{max}	Maximum outsourced agents allowed for task i
M	Sufficiently large number
Decision variables	
y_{ik}	1 if agent k is assigned to task i, 0 otherwise
x_{ijk}	1 if agent k goes directly from task i to task j, 0 otherwise
out_i	Number of outsourced agents for task i
s_{ik}	Start time of task i by agent k
w_{ik}	Tardiness of agent k on task i
ot_k	Overtime of agent k

$$\underbrace{\alpha_2 \cdot \sum_{k \in K} ot_k}_{\text{Overtime}} +$$

$$\underbrace{\alpha_3 \cdot \sum_{i \in I'} \sum_{k \in K} w_{ik}}_{\text{Tardiness}} +$$

$$\underbrace{\alpha_4 \cdot \sum_{i \in I'} (b_i - a_i) \cdot out_i}_{\text{Outsourced time}}$$

Subject to the following constraints:

$$\sum_{k \in K} y_{ik} = v_i - out_i \;\; \forall i \in I' \tag{4.1}$$

$$\sum_{j:(i,j) \in N} x_{ijk} = y_{ik} \;\; \forall i \in I',\, \forall k \in K \tag{4.2}$$

$$\sum_{j:(o,j) \in N} x_{ojk} = \sum_{i:(i,\bar{o}) \in N} x_{i\bar{o}k} = 1 \;\; \forall k \in K \tag{4.3}$$

$$\sum_{i:(i,h) \in N} x_{ihk} = \sum_{j:(h,j) \in N} x_{hjk} \;\; \forall h \in I',\, \forall k \in K \tag{4.4}$$

$$b_i + t_{ij} \leq s_{jk} + M(1 - x_{ijk}) \;\; \forall i, j : (i,j) \in N,\, \forall k \in K \tag{4.5}$$

$$s_{ik} - M(1 - y_{ik}) \leq a_i + w_i \;\; \forall i \in I',\, \forall k \in K \tag{4.6}$$

$$s_{jk} + M(1 - x_{ojk}) \geq e_k + t_{oj} \;\; \forall j \in I',\, \forall k \in K \tag{4.7}$$

$$b_i + t_{i\bar{o}} \leq f_k + ot_k + M(1 - x_{i\bar{o}k}) \;\; \forall i \in I',\, \forall k \in K \tag{4.8}$$

$$r_{iq} y_{ik} \leq m_{kq} \;\; \forall i \in I',\, \forall q \in Q,\, \forall k \in K \tag{4.9}$$

$$0 \leq w_{ik} \leq w_i^{max} \;\; \forall i \in I',\, \forall k \in K \tag{4.10}$$

$$0 \leq ot_k \leq ot_k^{max} \;\; \forall k \in K \tag{4.11}$$

$$0 \leq out_i \leq out_i^{max} \;\; \forall i \in I' \tag{4.12}$$

$$s_{ik} \geq 0 \;\; \forall i \in I,\, \forall k \in K \tag{4.13}$$

$$x_{ijk}, y_{ik} \in \{0, 1\} \;\; \forall i, j \in I,\, \forall k \in K \tag{4.14}$$

Constraints (4.1) and (4.2) ensure that the number of required agents (available or outsourced) is assigned to all tasks. Constraint (4.3) ensures that each agent starts and ends his/her route at the depot. Constraint (4.4) ensures flow conservation when visiting a counter for a task. Constraint (4.5) allows for processing a task only if its preceding task is completed. Constraint (4.6) states that an agent can arrive at a counter only before a task's opening time, otherwise, tardiness occurs. The routes' duration limit and overtime are defined by constraints (4.7) and (4.8). Constraint (4.9) ensures that the skill requirements of each task are fulfilled. Constraints (4.10), (4.11), and (4.12) define the bounds for tardiness, overtime, and outsourced agents, respectively. Finally, constraints (4.13) and (4.14) define the positive and binary variables, respectively.

4.4. Branch-and-price algorithm

In this section, we present a decomposition of the problem into a master problem and subproblems as proposed by Dantzig (1954). This decomposition can be then solved by a generic column generation method as the ones proposed by Barnhart et al. (1998), Lübbecke and Desrosiers (2005), and Liberatore et al. (2010), among others. The primary idea is to use the master problem to select agent-schedules from a pool of feasible routes generated by the subproblems.

Master Problem Let P_k be the set of feasible routes containing sequences of tasks (*i.e.*, paths) for each agent $k \in K$, and λ_{kp} be a binary variable equal to 1 only if path p of agent k is used and 0 otherwise. Additionally, let θ_{ikp} be a parameter equal to 1 if task i is in path p of agent k

and 0 otherwise. Then, the integer master problem (MP) can be defined as follows:

$$\text{Minimize} \sum_{k \in K} \sum_{p \in P_k} c_{kp} \lambda_{kp} + \alpha_4 \cdot \sum_{i \in I'} out_i \tag{4.15}$$

Subject to:

$$\sum_{k \in K} \sum_{p \in P_d} \theta_{ip} \lambda_{kp} = v_i - out_i \ \ \forall i \in I' \tag{4.16}$$

$$\sum_{p \in P_k} \lambda_{kp} = 1 \ \ \forall k \in K \tag{4.17}$$

$$out_i \leq out_i^{max} \ \ \forall i \in I' \tag{4.18}$$

$$\lambda_{kp} \in \{0, 1\} \ \ \forall k \in K, \ \forall p \in P_k \tag{4.19}$$

Where the cost of a path $p \in P_d$ is defined by:

$$\begin{aligned} c_{kp} = & \alpha_1 \cdot \sum_{(i,j) \in N} \sum_{k \in K} t_{ij} \cdot x_{ijk} + \\ & \alpha_2 \cdot \sum_{i \in I'} \sum_{k \in K} w_{ik} + \alpha_3 \cdot \sum_{k \in K} ot_k \end{aligned} \tag{4.20}$$

The objective is to minimize the overall costs (4.15). The set partitioning constraints (4.16) ensure that demand is satisfied either by available agents or outsourcing. Convexity constraints (4.17) state that exactly one route is used per agent. Constraint (4.18) ensures that the number of outsourced agents remains below the maximum allowed. Finally, constraint (4.19) defines the binary requirements for the assignment variables. The linear relaxation of this formulation (LMP), which is obtained by relaxing constraint (4.19), can be used to obtain a lower bound on the MP. However, set P_k may contain a large number of feasible routes, thus making the solution to this problem computationally intractable. Therefore, in a column generation approach, only subset $P'_k \subset P_k$ of routes are considered in a restricted master problem (RMP). Additional feasible columns are obtained using the iterative solution of each subproblem, which consists of finding routes for each agent with negative reduced cost $\bar{c}_{kp}$:

$$\begin{aligned}
\bar{c}_{kp} &= c_{kp} - \sum_{i \in I'} \theta_{ikp} \pi_i - \rho_k \\
&= \sum_{(i,j) \in N} \sum_{k \in K} (\alpha_1 \cdot t_{ij} - \pi_i) \cdot x_{ijk} \\
&\quad + \alpha_2 \cdot \sum_{i \in I'} \sum_{k \in K} w_{ik} + \alpha_3 \cdot \sum_{k \in K} ot_k - \rho_k
\end{aligned}$$

where π_i, and ρ_k correspond to the dual values of constraints (4.16) and (4.17) in the optimal solution of the RMP. These values are then passed to the subproblems from which single-agent routes are obtained.

Pricing Problems We let $I'_{sub_k} \subset I'$ and $I_{sub_k} \subset I$ be the set of tasks which can be processed by agent k. Also let $N_{sub_k} = \{(i,j) | i, j \in I_{sub_k}\}$. Then, each $k-$subproblem can be defined as follows:

$$\text{Minimize } \bar{c}_{kp} = \sum_{(i,j)\in N} (\alpha_1 \cdot t_{ij} - \pi_i) \cdot x_{ij} + \alpha_2 \cdot \sum_{i\in I'} w_i + \alpha_3 \cdot ot - \rho_k \tag{4.21}$$

$$\sum_{j:(i,j)\in N_{sub_k}} x_{ij} = y_i \quad \forall i \in I'_{sub_k} \tag{4.22}$$

$$\sum_{j:(o,j)\in N_{sub_k}} x_{oj} = \sum_{i:(i,\bar{o})\in N_{sub_k}} x_{i\bar{o}} = 1 \tag{4.23}$$

$$\sum_{i:(i,h)\in N_{sub_k}} x_{ihk} - \sum_{j:(h,j)\in N_{sub_k}} x_{hj} = 0 \quad \forall h \in I'_{sub_k} \tag{4.24}$$

$$b_i + t_{ij} - M(1 - x_{ij}) \leq s_j \quad \forall i,j : (i,j) \in N_{sub_k} \tag{4.25}$$

$$s_i - M(1 - y_i) \leq a_i + w_i \quad \forall i \in I'_{sub_k} \tag{4.26}$$

$$s_j + M(1 - x_{oj}) \geq e + t_{oj} \quad \forall j \in I'_{sub_k} \tag{4.27}$$

$$b_i + t_{i\bar{o}} - M(1 - x_{i\bar{o}}) \leq f + ot \quad \forall i \in I'_{sub_k} \tag{4.28}$$

$$0 \leq w_i \leq w_i^{max} \quad \forall i \in I'_{sub_k} \tag{4.29}$$

$$0 \leq ot \leq ot^{max} \tag{4.30}$$

$$s_i \geq 0 \quad \forall i \in I_{sub_k} \tag{4.31}$$

$$x_{ij}, y_i \in \{0,1\} \quad \forall i,j \in I_{sub_k} \tag{4.32}$$

where x_{ij} is a binary variable with a value of 1 if the arc (i,j) is used and 0 otherwise. s_i and w_i are positive variables corresponding to the starting time and the tardiness of task i, whereas ot is a positive variable that represents the overtime of the route. The objective is to find a route with a minimum reduced cost (4.21). Constraint (4.22) initializes the variable y_i. Constraints (4.23) and (4.24) ensure the feasibility of the route. Constraint (4.25) defines the task's starting time, whereas constraint (4.26) ensures that a counter's opening time is respected; otherwise, the agent is tardy. Constraints (4.27) and (4.28) define shift-length limits, and if an agent arrives at the depot after closing time, overtime is incurred. Constraints (4.29) and (4.30) define the bounds for the tardiness and overtime variables, respectively. Finally, positive variable (4.31) and binary variable (4.32) are defined.

These subproblems correspond to an Elementary Shortest Path Problem with Resource Constraints (ESPPRC). The requirement of elementary paths (*i.e.,* paths that visit each task no more than once) is required because the reduced cost associated with each task may lead to negative cost cycles. Note that ESPPRC in graphs with negative cost cycles is strongly NP-hard (Dror, 1994). This represents a major drawback because in our algorithm, we need to solve these subproblems several times. Therefore, an efficient solution approach is required.

4.4.1. Labeling algorithm

Because the ESPPRC can be computationally expensive to solve, a typical approach is to relax the elementariness requirement and then to solve it as a Shortest Path Problem with Resource Constraints (SPPRC) (*i.e.,* subcycles are allowed). The SPPRC can be solved using dynamic programming, which can be solved in pseudo-polynomial time (Desrochers et al., 1992). Nevertheless, the main drawback to this approach is that it weakens the lower bound computed by the column generation phase (Bode and Irnich, 2014a). Conversely, dynamic programming can be used to solve the ESPPRC (Feillet et al., 2004; Liberatore et al., 2010) to provide better lower bounds, albeit with a higher computational cost. It is possible, however, to solve the pricing problem described in the previous section as an SPPRC while ensuring elementary paths by exploiting the problem structure.

Below, we describe a dynamic programming algorithm we propose for the solution of the pricing problems of our branch-and-price approach, along with details on its implementation. Similar algorithms are proposed by Feillet et al. (2004) and Liberatore et al. (2010) also to solve the pricing problems embedded in a column generation algorithm to solve the classical Vehicle Routing Problem with Time Windows (VRPTW). Mainly, our algorithm differs from theirs on the fact that end times of the tasks are fixed (thus the processing time of the tasks are start time-dependent) and that elementary paths are not explicitly required.

In short, the algorithm's objective is to obtain routes from o to $\bar{o}$ for an agent such that the reduced cost is minimized. For this purpose, we define the following concepts. A *state* associated with task i corresponds to a partial path of visited counters from the depot to i. Each task can be associated with more than one state because multiple feasible paths can end in the respective task. To represent such states, we use multidimensional resource vectors or labels represented by $L_i = (i, T_i, C(T_i))$, where i is the last visited task, T_i is the starting time of task i, and $C(T_i)$ is the time-dependent cost. Additionally, let $\mathcal{U}_i$ and $\mathcal{P}_i$ represent the set of unprocessed and processed labels for task i, respectively.

The process is initialized by setting sets $\mathcal{U}_i$ and $\mathcal{P}_i$ to the empty set, creating the first label $L_o = (o, T_o = e, C(T_o) = 0)$, and appending L_o to the unprocessed set. Through the algorithm, labels are extended to successive feasible labels of the form $L_j = (j, T_j, C'(T_j))$ by appending an additional arc (i, j) to the partial path $o - i$ and updating its corresponding resources using the following resource extension functions (REF):

$$T_j = \max(a_j, b_i + t_{ij}) \tag{4.33}$$

$$C'_j = \begin{cases} C(T_i) + c_{ij} + \max(0, \alpha_2 \cdot (T_j - a_j)) - \pi_i, & \text{if } j \neq \bar{o} \\ C(T_i) + c_{ij} + \max(0, \alpha_3 \cdot (T_j - f)) - \pi_i, & \text{if } j = \bar{o} \end{cases} \tag{4.34}$$

We note that, in contrast to Feillet et al. (2004) and Liberatore et al. (2010), this algorithm takes into consideration the closing time of a counter b_i instead of recursively calculating T_j from previous states. In this way, the characteristic of fixed ending times from our problem is considered in our algorithm. Then, iteratively, the labels with the minimum time are selected from the unprocessed set to search for possible extensions along their arcs. These extensions can only be made on feasible labels. A label $(i, T_i, C(T_i))$ is feasible if $T_i \leq b_i + w^{max} | i \neq \bar{o}$ and $T_i \leq b_i + ot^{max} | i = \bar{o}$. Labels that do not satisfy these conditions are discarded. If feasible, the new extended labels are added to the unprocessed set and the currently explored labels are added to the processed set.

Because dual information may produce arcs with negative costs, sub-cycles are not strictly forbidden. Using additional vectors on each label, as proposed by Feillet et al. (2004) and Liberatore et al. (2010), the elementariness of the paths can be enforced. However, from the time extension function (4.33) we can observe that the time consumption is strictly positive, as $T_j - T_i \geq 0$ for all $(i, j) \in N_{sub_k}$ if such extension is feasible. Because the algorithm chooses the label with the least time consumption from the set of unprocessed labels, it is then guaranteed that all feasible extensions from a given label have a non-decreasing consumption of the time resource (Irnich et al., 2005), effectively forbidding sub-cycles because of the definition of feasibility. In this way, our algorithm mimics an acyclic graph without the need to explicitly enforce elementariness at the label-extension step. Therefore, in contrast to Feillet et al. (2004) and Liberatore et al. (2010), in the proposed algorithm no elementariness check is conducted while extending the labels.

To keep the size of these sets tractable, dominance rules are required to discard (multiple) labels that cannot lead to an optimal solution. For this purpose, dominance rules are applied. Let $L_i = (i, T_i, C(T_i))$ and $L'_i = (i, T'_i, C'(T'_i))$ be two labels of task i. Then, L_i dominates L'_i if $L_i < L'_i$ component-wise, *i.e.,* if $T_i \leq T'_i$, $C(T_i) \leq C'(T'_i)$, and at least one of the inequalities is strict. If a label is dominated, then it can be discarded; when all components are equal, both labels are retained.

Once all paths arriving at the depot $\bar{o}$ are obtained and no unprocessed labels remain, the label with the minimum cost is selected and corresponds to the optimal solution of the respective subproblem. Its corresponding reduced cost is determined by $\bar{c_{kp}} = C(T_{\bar{o}}) - \rho_k$. If more than one label has the same cost, and none can be dominated, then all of them are added as columns of the reduced master problem. Algorithm 3 shows an overview of the labeling process.

Algorithm 3 Labeling algorithm overview

```
Set $\mathcal{U}_i = \mathcal{P}_i = \varnothing \ \ \forall i \in I_{sub_k}$
Set $L_o = (o, T_o = e_k, C(T_o) = 0)$
Set $\mathcal{U}_o = \{L_o\}$
**while** $\bigcup_{i \in I_{sub_k}} \mathcal{U}_i \neq \varnothing$ **do**
    Choose a label $L_m \in \bigcup_{i \in I_{sub}} \mathcal{U}_i$ and remove it from $\mathcal{U}_m$
    **for** all arcs $(m, j) \in N_{sub}$ **do**
        Extend $L_m$ along arc $(m, j)$ to create label $L_j$ using the REF's
        **if** $L_j$ is feasible **then**
            Add $L_j$ to $\mathcal{U}_j$
            Apply dominance rules to $\mathcal{U}_j$ and $\mathcal{P}_j$
        **end if**
        **if** $L_m$ is not dominated **then**
            Add $L_m$ to $\mathcal{P}_m$
        **end if**
    **end for**
    Filter $\mathcal{P}_{\bar{o}}$ to find the shortest $o - \bar{o}$ path
**end while**
```

4.4.2. Acceleration strategies

To accelerate the generation of new columns with negative reduced costs, one can solve the pricing problems heuristically in the first iterations of the column generation algorithm and then solve them exactly in the last iteration(s) to guarantee optimality. For this purpose, below we propose heuristics that are applied sequentially during the execution of the algorithm. Once no new negative reduced-cost columns can be found with these heuristics methods, we solve the pricing problems exactly to ensure optimality.

Initialization To provide the initial columns for the column-generation procedure, we use two approaches. First, we sort the agents by increasing shift length and for each agent, we run the labeling algorithm, stopping when a preset number of shortest paths are found. Each path is then inserted as a column of the reduced master problem (RMP). Second, we use a modified version of the insertion heuristic from Solomon (1987). This algorithm can be summarized as follows: (i) for each agent $k \in K$, we sequentially initialize routes by choosing the task with the highest processing time; (ii) for each task h available to team k, we calculate the operation costs obtained from inserting it, if feasible, into each arc (i, j) of the initial route; (iii) the insertion $(i, h, j)^*$ with the minimum cost is selected and the initial route is updated accordingly at that point, steps (ii) and (iii) are repeated until no further tasks can be inserted; (iv) the resulting routes are inserted as columns of (RMP). If a task is not included in any path from the two methods, then it is included in the RMP as an artificial column with a high cost.

(H1) Partial pricing Each k subproblem focuses on the route of a specific agent and thus, only the tasks corresponding to this subproblem are considered. Depending on the task's characteristics, it is possible for multiple agents to be assigned to them with the same route cost. In terms of the column-generation procedure, this means that the marginal values ρ_k have zero values, *i.e.,* there is no prize to earn for dispatching agent k. We can exploit this fact to reduce the number of subproblems in each iteration by solving only those subproblems with marginal values different from zero. If no subproblem satisfies this condition, we solve all of the subproblems until three columns with reduced cost are found, and the column-generation phase continues.

(H2) Relaxed dominance rules One way to accelerate the solution of the subproblems is to relax the dominance rules from the exact labeling algorithm. This is done by testing the dominance of only a subset of the labels' resources: for two labels $L_i = (i, T_i, C(T_i))$ and $L'_i = (i, T'_i, C'(T'_i))$, L_i dominates L'_i if $C(T_i) \leq C'(T'_i)$.

(H3) Partial pricing and partial dominance rules This approach consists of a combination of both of the previously described heuristics: H1 and H2. Therefore, partial pricing is applied in the column-generation phase and relaxed dominance rules are used in the solution to the subproblems.

4.4.3. General procedure

In this section, we outline the proposed branch-and-price algorithm, which primarily consists of a column-generation algorithm embedded in a branch-and-bound procedure to guarantee integrality. Additionally, further details on the implementation of the column-generation algorithm, data preprocessing, search strategies, and branching rules are presented.

Column generation The column-generation process can be described as follows. The algorithm is initialized by the insertion heuristic. The master problem is solved using a set of artificial columns obtained by the introduction of slack variables with a high cost to ensure primal feasibility. Because of their high cost, these variables eventually leave the basis, once further feasible columns are obtained. From the solution to the master problem, the marginal values are obtained and passed to the subproblems. New routes (*i.e.,* columns) are obtained by solving the subproblems. If columns are found with a negative reduced cost, the column-generation process is started again. Otherwise, an optimal solution for the reduced master problem (and thus, for the master problem) is found. If this solution is an integer, then it is an optimal solution to the original problem; if not, then it corresponds to a valid fractional lower bound. Next, branching on fractional variables is required and column generation is applied on each node with its respective fractional bounds.

Preprocessing Each subproblem k focuses on a subset $I_{sub_k} \subset I$ of tasks because of the skill requirements. It is possible to further reduce this subset by removing tasks that cannot be operated by k agents based on the shift starting and ending times and the overtime limit ot_k^{max}. Additionally, one can reduce the number of arcs considered in the subproblem's set N_{sub_k} by redefining it as $N_{sub_k} = \{(i,j)|i,j \in I_{sub_k}, b_i + (b_i - a_i) + t_{i,j} \leq b[j] + w_i^{max}\}$. This means that infeasible arcs caused by a violation of the maximum tardiness bound are eliminated.

Branching We explore the branching tree using a depth-first strategy. In this way, the solution from the parent node is used as a warm start for the children nodes (eliminating the columns that do not comply with their respective branching bounds).

One common branching rule is to branch on the arcs (i, j), because this rule is simple to incorporate into the subproblems by removing arcs in N_{sub_k}, and deleting the columns that violate the branching rule. However, this approach does not provide strong integrality bounds for our problem, because tasks i and j might require the assignment of more than one agent. Therefore, we use the following branching rule: for each node on the branching tree, we define the set $\mathcal{A}$ as the set of fractional assignments (i,k) such that $0 < \psi_{ik} < 1$, where $\psi_{ik} = \sum_{p\in P_k} \theta_{ikp}\lambda_{kp}$; θ_{ikp} is equal to 1 if task j is assigned to agent k's path p, and 0 otherwise. Then, the assignment $(i,k)^*$ is selected as follows:

$$(i,k)^* \in \operatorname*{arg\,max}_{(i,k)\in\mathcal{A}} \{(b_i - a_i)\cdot \min(\psi_{ik}, 1-\psi_{ik})\}$$

From this node, two successors are generated:

- A left node with $\psi_{ik} = 0$, where agent k cannot be assigned to task i, *i.e.,* this task is removed from its respective subproblem's graph and columns using this assignment are discarded.
- A right node with $\psi_{ik} = 1$, where this arc is fixed, *i.e.,* agent k is assigned to task i.

Upper bound update After exploring a given number of nodes, it is possible to obtain new information on the development of the (integer) upper bound by solving the reduced master problem (RMP) of the incumbent node as an integer program. If the new bound is better than the best bound found so far, then the bound is updated and nodes with a more expensive solution are pruned. There is, however, a trade-off related to computation time, because too often, solving the integer RMP can be computationally expensive and not necessarily beneficial to the quality of the upper bound.

4.5. Numerical Experiments

For our numerical tests, we use real-world input data from a German ground-handling agency to test the performance of our proposed algorithm. First, in Section 4.5.1, we use the provided data to test the performance of the branch-and-price algorithm. We also evaluate the gains in computation times obtained by solving the pricing problems using shortest paths instead of elementary shortest paths in the column-generation phase of our algorithm. Second, in Section 4.5.2, we generate instances with different shift lengths to evaluate the performance of our algorithm under different shift profiles. Third, we generate instances of larger size based on the existing information in Section 4.5.3 to test the performance of our solution approaches in larger settings. Fourth, in Section 4.5.4, we conduct a sensitivity analysis to evaluate the impact of the maximum tardiness in the resulting schedules. All of the methods are implemented in Python 2.7.3 using the Gurobi 5.6 solver on a 2.9 GHz Intel Core i7 machine with 8 GB of RAM in OS X Yosemite 10.10.14.

4.5.1. Performance tests

The data consist of 24 real-world instances, corresponding to flight schedules on different days. In addition, we include 4 realistic but not real instances included in Appendix A and also available at http://stolletz.bwl.uni-mannheim.de/en/library.

Table 4.3 presents information regarding the tasks and the workforce for each instance: the number of tasks, the minimum, maximum, and average occupation time (in minutes), the average agent requirements, and the number of tasks requiring each skill is shown. Also, this table presents the number of agents, the minimum, maximum, and average shift length (in minutes), as well as the number of agents qualified on each skill, for each instance. The location of the check-in counters is known, and the traveling distance between the locations is provided by the ground-handling agency. Based on the contracts with the airlines, the maximum tardiness is set to one-half of the duration of each task ($w_{ik}^{max} = 0.5 \cdot (b_i - a_i) \ \ \forall i \in I, \forall k \in K$), and the maximum outsourcing is set such that at least one normal agent is present at each task ($out_i^{max} = v_i - 1 \ \ \forall i \in I$). For each instance, a shift plan is also provided that includes information about the qualifications, shift start, and shift end for each agent. The shift length per agent has a value of between 3 and 10 hours, and each agent can be qualified in one or more of the 12 available skills. To comply with labor regulations, the maximum overtime per agent is set such that the shift length plus overtime cannot exceed 10 hours. Finally, based on observations from the ground-handling agency's management, the weights for the objective function are set as $[\alpha_1, \alpha_2, \alpha_3, \alpha_4] = [0.1, 0.2, 0.3, 0.4]$.

The results for all of the test instances are obtained from the solution of the MIP formulation and two versions of the branch-and-price algorithm: one on which the subproblems are modeled as shortest path problems (SPPRC) and the other on which the subproblems are modeled as elementary shortest path problems (ESPPRC). The solutions reported are those obtained within a time limit of 1 hour. We use the exact solutions from the BP to calculate ex-post the absolute gap of the MIP results for the cases where optimality is not proven within the time limit. For the MIP results, Table 4.4 presents the objective function value, the computation time (in minutes), and the absolute gap to optimality (in percentage). For both versions of the branch-and-price (BP) results, the table shows the objective function

Table 4.3.: Test instances summary

		Occupation (m)			Avg.	Skills (Nr. of tasks)											
Instance	Tasks	Min.	Max.	Avg.	Demand	q1	q2	q3	q4	q5	q6	q7	q8	q9	q10	q11	q12
1	64	30	120	68.95	1.23	5	10	0	1	0	0	12	0	0	0	0	3
2	64	30	140	67.74	1.23	10	4	0	1	0	0	13	0	0	0	0	3
3	70	30	140	71.03	1.33	18	4	0	1	0	1	7	0	2	0	0	3
4	54	30	140	68.65	1.26	8	4	0	1	0	0	9	0	0	0	1	3
5	68	30	140	68.18	1.25	9	6	0	1	0	1	12	0	0	0	1	3
6	52	30	120	68.10	1.19	5	5	0	1	0	0	10	2	0	0	1	3
7	64	30	140	69.44	1.20	8	4	0	1	0	1	13	0	0	0	1	3
8	64	30	120	68.95	1.23	5	10	1	1	0	0	12	0	0	0	0	3
9	64	30	140	67.74	1.23	10	4	0	1	0	0	13	0	0	0	0	3
10	70	30	140	71.03	1.33	18	4	0	1	0	1	7	0	0	0	0	3
11	54	30	140	68.65	1.26	8	4	0	1	0	0	9	0	0	0	1	3
12	68	30	140	68.18	1.25	9	6	0	1	0	1	12	0	0	0	1	3
13	52	30	120	68.10	1.19	5	5	0	1	0	0	10	0	0	0	1	3
14	64	30	140	69.44	1.20	8	4	0	1	0	1	13	0	0	0	1	3
15	64	30	120	68.95	1.23	5	10	0	1	0	0	12	0	0	0	0	3
16	64	30	140	67.74	1.23	10	4	0	1	0	0	13	0	0	0	0	3
17	52	30	120	68.10	1.19	5	5	0	1	0	0	10	0	0	0	1	3
18	64	30	140	69.44	1.20	8	4	0	1	0	1	13	0	3	1	1	3
19	64	30	120	68.95	1.23	5	10	0	1	0	0	12	0	0	0	0	3
20	70	30	140	71.03	1.33	18	4	0	1	0	1	7	0	0	0	0	3
21	52	30	120	68.10	1.19	5	5	0	1	0	0	10	0	0	0	1	3
22	64	30	140	69.44	1.20	8	4	0	1	0	1	13	0	0	0	1	3
23	64	30	120	68.95	1.23	5	10	1	1	0	0	12	0	0	0	0	3
24	64	30	140	67.74	1.23	10	4	0	1	0	0	13	0	0	0	0	3
A1	62	30	140	69.44	1.24	12	1	1	13	0	4	31	-	-	-	-	-
A2	66	30	140	68.18	1.28	15	1	1	12	0	4	33	-	-	-	-	-
A3	62	30	140	69.44	1.24	12	1	1	13	0	4	31	-	-	-	-	-
A4	50	30	120	68.10	1.24	10	1	0	10	0	4	25	-	-	-	-	-

		Shift length (m)			Skills distribution (Nr. of agents)											
Instance	Agents	Min.	Max.	Avg.	q1	q2	q3	q4	q5	q6	q7	q8	q9	q10	q11	q12
1	45	180	585	314.67	42	37	11	25	18	7	39	1	0	0	36	25
2	37	180	555	353.92	34	30	9	19	14	5	32	2	0	0	30	18
3	41	180	585	376.83	38	35	13	24	17	9	35	3	1	0	31	22
4	31	180	570	371.61	27	29	10	17	11	7	27	1	1	1	25	15
5	38	180	570	384.47	34	35	11	22	18	9	33	2	1	1	30	22
6	35	180	525	324.00	33	33	10	19	12	8	31	1	1	1	30	21
7	42	180	555	336.07	41	35	10	22	15	8	35	1	0	1	34	22
8	40	180	570	344.25	37	35	7	19	15	6	32	3	0	1	32	19
9	35	180	555	375.86	33	30	11	16	14	8	30	3	1	1	29	18
10	38	180	585	417.63	35	32	12	19	14	9	33	4	1	1	30	19
11	30	180	585	375.00	26	25	7	15	11	6	27	2	1	1	23	14
12	37	180	570	394.46	33	32	9	19	17	5	30	1	0	1	30	19
13	35	180	525	349.29	31	31	6	15	14	5	28	2	1	0	28	16
14	48	180	585	375.94	42	42	11	25	19	8	40	4	1	0	39	25
15	49	180	570	332.14	45	42	12	25	21	10	40	3	1	1	41	26
16	38	180	555	396.71	37	32	6	19	15	5	33	2	1	1	30	18
17	29	180	585	385.86	27	26	8	18	13	4	25	1	0	0	25	18
18	36	180	585	375.00	32	31	8	17	14	6	30	2	1	1	30	18
19	40	180	585	336.38	35	36	7	22	16	6	36	3	1	0	32	21
20	35	180	585	435.86	32	30	10	20	11	7	32	1	1	1	26	19
21	34	180	585	407.65	31	27	7	17	12	4	29	1	0	0	25	16
22	37	180	585	372.16	35	31	9	19	15	6	33	1	0	0	28	20
23	43	180	600	343.26	38	37	11	21	18	9	37	2	1	1	36	23
24	37	180	585	442.30	35	31	9	20	15	5	33	0	0	1	28	19
A1	42	180	555	336.07	42	23	19	35	1	34	42	-	-	-	-	-
A2	38	180	570	384.47	38	23	21	33	2	30	38	-	-	-	-	-
A3	48	180	585	375.94	48	26	23	40	1	39	48	-	-	-	-	-
A4	29	180	585	385.86	29	18	15	25	0	25	29		-	-	-	-

value, the computation time (in minutes), the number of explored nodes, and the overall number of columns generated. No optimality gap is reported because an exact solution is found for all instances for both versions of the BP. Detailed results including final schedules, assignments, and routes for instances A1-A4 are shown in Appendix B.

Table 4.4.: Performance tests results

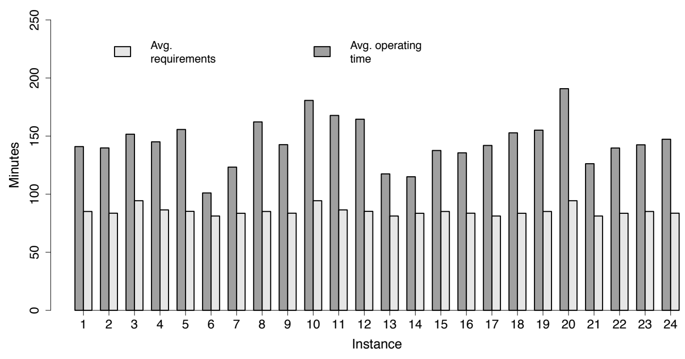
Inst.	MIP			BP						
	Objective	Gap (%)	CPU (m)	Objective	SPPRC			ESPPRC		
					CPU (m)	Nodes	Columns	CPU (m)	Nodes	Columns
1	30.80	0.00	TL	30.80	6.18	16	16445	18.68	20	17825
2	NA	NA	TL	25.00	7.73	22	25335	14.95	24	27687
3	NA	NA	TL	27.00	21.40	20	36410	45.78	25	34549
4	20.80	0.00	10.53	20.80	1.23	2	1439	6.53	2	1350
5	25.60	0.00	58.87	25.60	5.00	7	8675	15.93	7	8401
6	18.20	0.00	14.31	18.20	3.47	19	14383	8.51	11	14888
7	24.60	0.00	59.02	24.60	6.73	17	17551	13.76	13	17301
8	30.80	0.00	25.73	30.80	8.40	29	26780	23.27	32	29212
9	23.60	0.00	15.06	23.60	7.98	21	22658	12.89	18	23730
10	76.10	66.16	TL	45.80	22.22	21	32765	35.25	24	32557
11	38.80	0.00	6.15	38.80	2.40	9	6566	4.53	8	6760
12	40.00	0.00	21.81	40.00	7.23	17	19029	11.61	21	18480
13	24.80	0.00	2.54	24.80	1.10	1	719	2.66	1	672
14	29.40	0.00	46.74	29.40	9.75	21	27366	19.6	25	27038
15	53.80	0.00	TL	53.80	3.85	6	6127	6.35	7	6482
16	22.20	0.00	21.16	22.20	7.29	16	16909	11.85	15	15796
17	31.40	0.00	5.86	31.40	1.05	2	1275	3.8	4	1326
18	72.40	0.00	22.67	72.40	6.84	20	20761	11.74	21	20916
19	31.20	0.00	37.59	31.20	10.45	30	30848	12.73	33	29230
20	46.60	0.00	34.44	46.60	16.23	20	30075	21.84	21	29692
21	31.80	0.00	4.39	31.80	1.91	5	4240	3.8	10	4044
22	24.60	0.00	24.29	24.60	2.46	1	943	7.11	1	949
23	30.80	0.00	30.32	30.80	2.53	1	873	7.32	1	877
24	21.80	0.00	19.61	21.80	9.47	18	21010	16.96	17	20457
	Average	2.76	31.71		7.20	14.21	16215.92	14.06	15.04	16259.13
A1	115.00	0.00	13.64	115.00	4.33	2	2459	4.33	4	4918
A2	158.50	0.00	13.05	158.50	8.15	14	14938	8.15	18	19206
A3	135.50	0.00	TL	135.50	5.19	4	4878	5.19	5	6482
A4	122.50	0.00	TL	122.50	3.44	19	15228	3.44	19	15796
	Average	0.00	56.68		5.28	9.75	9375.75	5.28	11.50	11600.50

From these results, we make the following observations: On the one hand, the MIP solver is unable to obtain a feasible solution for instances 2 and 3 within the time limit. Also, on instances 1, 10, 15, A3, and A4 optimality is not proven within the time limit. Although the solution for instances 1, 15, A3, and A4 corresponds to the actual optimal solution, instance 10 shows a low solution quality (45% away from the best solution found). Without the time limit, instances 1, 2, 3, 10, 15, A3, and A4 are solved to optimality, although with a significant increase in the computation time, reaching a maximum of 195.96 minutes. All other 19 instances are solved to optimality within the time limit, with an average computation time of 44.19 minutes. On the other hand, the branch-and-price approach clearly outperforms the MIP formulation: it is able to solve all instances to optimality in less than 23 minutes (with an average computation time of 6.24 minutes).

Additionally, these results show that explicitly requiring elementary paths when solving the pricing problems is computationally expensive. In addition, we observe that the columns generated by the shortest path subproblems are equivalent to those generated by the elementary shortest path subproblems in terms of the sequence of tasks on each path. This fact is shown in the number of columns and the number of nodes explored by the algorithm, because they do not significantly vary among methods. This indicates that solving the pricing problems as shortest path problems (exploiting the problem structure) provides equivalent solutions without the computational effort of requiring elementariness on each subproblem's path.

In addition, further comments regarding the schedule quality for the real instances can be made. For this purpose, Figure 4.1 shows a comparison of the average required counter time per agent and the average operating time per agent, defined as the amount of time an agent spends at the assigned tasks or waiting for a counter to be opened, plus travel time (*i.e.*, route length). In this figure, we observe that the requirements for all instances are significantly overestimated by the operating time. In the best case (instance 6), 80% of the operating time is destined to serving tasks, whereas in the worst case (instance 20), this value is only 49%. Overall, in the current schedules, agents spend an average of 60% of their operating time in check-in tasks, 4.5% traveling between counters, and the remaining time they idle. This effect, however, is a product of the relation between flight schedules (*e.g.*, long gaps between flights), shift plans (*e.g.*, long shifts covering periods with no flights), and employee qualifications (*e.g.*, employees with single qualifications) for which no specific behavior could be observed in the tested instances.

Figure 4.1.: Average operating time per agent



4.5.2. Impact of shift length

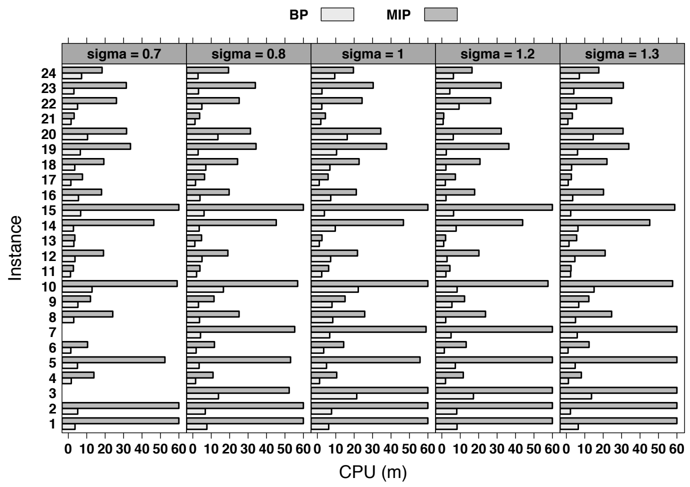
For the following tests, we concentrate on the shift length's impact on the resulting schedules. For this purpose, we artificially create different shift plans for each real-world test instance by multiplying the shift length for each agent by a scaling factor $\sigma = [0.7, 0.8, 1.2, 1.3]$. We keep the start time e_k unchanged, and f_k depends on the new values for the shift length. If the new shift length does not comply with the shift requirements (minimum 3 and maximum 10 hours), then f_k is adjusted accordingly. Additionally, if the new shift end time exceeds the length of the planning horizon, then f_k is truncated up to that time point. We then solve all of these new 96 instances both with the MIP solver and with the branch-and-price algorithm with the shortest path subproblems given a time limit of one hour.

Table 4.5 and Figure 4.2 present a comparison of the results obtained for all of the test instances with both solution methods. The table shows the average values of the computation time (in minutes) for the solutions of both methods, along with the number of nodes explored and the number of columns generated by the branch-and-price algorithm and the absolute gap (in percentage) for the MIP. In the figure, we plot the computation time (in minutes) for each instance and σ. Detailed results can be found in Appendix C in Tables C.1 through C.4.

Table 4.5.: Summary results for different values of σ

σ	BP			MIP	
	CPU (m)	Nodes	Cols	CPU (m)	Abs. Gap (%)
0.7	4.52	65.64	4588.91	27.88	0.25
0.8	5.06	70.13	4299.21	30.34	0.74
1.2	5.05	65.92	9853.42	30.82	0.18
1.3	5.19	63.92	10154.42	30.60	0.00
Average	4.95	66.40	7223.99	29.91	0.29

Figure 4.2.: Computation time for different values of σ



From these results, we make the following observations. Instances 3 and 7 with $\sigma = 0.7$ are shown to be infeasible for both methods, because the reduction of the shifts to 70% of the original setting is too restrictive. The MIP solver is unable to solve 10 out of the 94 remaining cases within the one-hour computation time limit. From the cases in which the time limit is reached but a solution is obtained, the MIP formulation shows an average optimality gap of 0.29%, with a maximum value of 11.95%. The branch-and-price algorithm, however, is able to solve all of the test instances to optimality in less than 17 minutes (with an average of 4.95 minutes). This indicates, then, that our proposed algorithm is not only robust enough to address instances with different shift profiles but also capable of solving them in a short computation time.

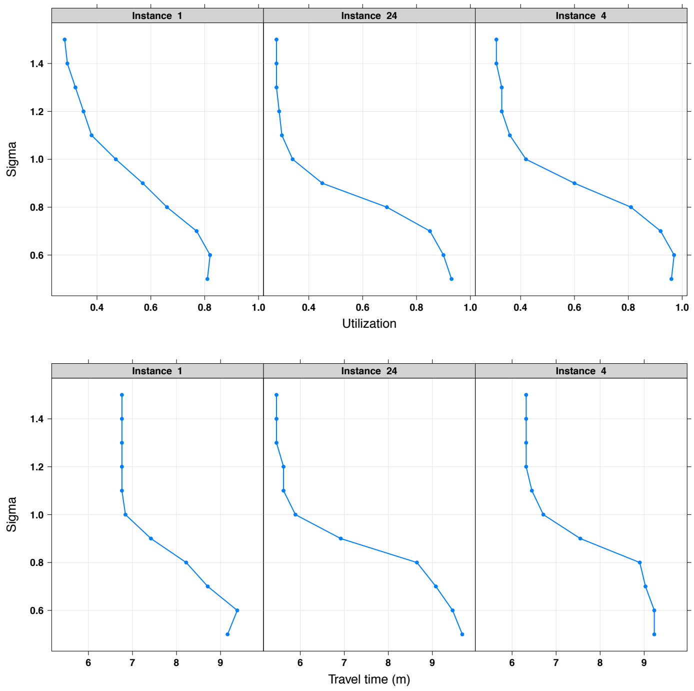
Moreover, the shift length has a direct impact on the operation of the agents. In Figure 4.3, the average utilization per agent (average operation time divided by the average shift length per agent) and the average travel time per agent are depicted. For clarity, only the results for a subset of instances are shown. Instances 1, 24, and 4 are selected as these are the instances with the minimum, maximum, and average shift length. From this figure, we can make the following observations: First, as expected, utilization decreases when the shift length increases. However, this shows a lower impact once the shift length exceeded the 100% of the original setting. Second, the average travel time per agent as well decreases when the shift length is increased. This is due to the fact that longer shifts allow agents to be assigned to additional tasks within the same gate, and thus lower the overall traveling time. This reduction is, however, not that significant once the average shift length exceeds the length of the original setting.

4.5.3. Performance of instances of larger size

For the next tests, we select the three real-world instances with maximum, minimum, and average demand (instances 20, 6, and 22, respectively) and generate instances of larger size by doubling and tripling the demand and the number of agents, resulting in 6 new test instances.

Table 4.6 shows the objective function value, the computation time (in minutes), the optimality gaps (in percentage), the number of explored nodes, and the number of columns generated for each solved instance. The MIP formulation is able to solve all instances with an average of

Figure 4.3.: Avg. Utilization and Avg. Travel time for different values of σ



84.25 minutes of computation time. Consequently, we conclude that the branch-and-price algorithm outperforms the MIP formulation by solving all instances to optimality in less than 25 minutes.

In Figure 4.4, a comparison of the computation time (in minutes) of the MIP solver and the BP approach for these instances and the original ones is

Table 4.6.: Results for large size instances

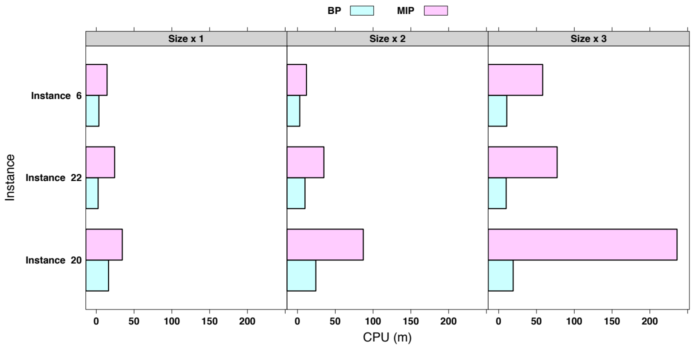
Instance	MIP			BP			
	Objective	CPU (m)	Abs. gap (%)	Objective	CPU (m)	Nodes	Columns
6x2	36.40	11.91	0.00	36.40	3.02	5	6183
6x3	54.40	58.20	0.00	54.40	10.71	13	22983
20x2	93.20	87.09	0.00	93.20	24.27	11	30030
20x3	140.40	236.00	0.42	139.80	19.17	4	6582
22x2	49.20	34.93	0.00	49.20	9.95	6	9324
22x3	73.80	77.35	0.00	73.80	9.92	18	25373
Average	74.57	84.25	0.07	74.47	12.84	9.50	16745.83

shown. We first note that the computation time increases when increasing the size of the instances. This was the case for all instances with the exception of instance 6x2 as the BP approach is 2.4 minutes faster than the original instance. Second, we observe that increasing the instance size has a different impact on both methods on all instances. More specifically, the BP approach shows a lower increase in computation time in comparison to the MIP solver. This can be explained by the more-efficient exploration of branching nodes of the BP approach, as the pricing problems are solved with a pseudo-polynomial algorithm less sensitive to increases in the solution space.

4.5.4. Impact of tardiness limit

The final tests focus on the impact of the maximum tardiness limit in the daily schedules of our planning problem. For this purpose, we selected the real-world instance that shows the maximum overall tardiness from Section 4.5.1 (*i.e.*, instance 15) and modified the original maximum tardiness bound (w_i^{max}). In the original setting, this bound corresponds to half of the

Figure 4.4.: Computation time comparison

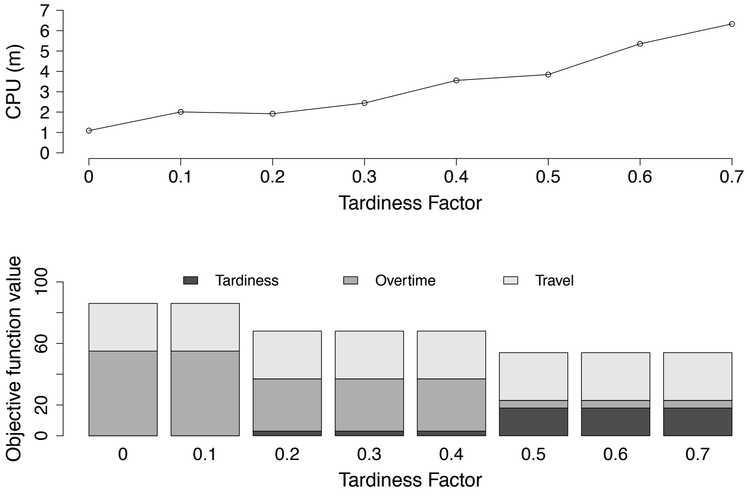


maximum duration of a task. In these tests, we set $w_i^{max} = \beta \cdot (b_i - a_i)$, and set the values for the tardiness factor $\beta = [0.0, 0.1, ..., 0.7]$.

Figure 4.5 shows two graphs with results from the aforementioned tests. The first graph shows the computation time for each tardiness factor. The second graph shows the objective function value of the solution for each tardiness factor, along with the weighted value of the components of the objective function: overall tardiness, overall overtime, and overall traveling time. Outsourcing is not displayed in these results because its value is 0 for all cases.

As can be observed, the computation time monotonically increases with the tardiness factor. Conversely, we observe a reduction of the objective function value with every increment of the tardiness factor. Although the weighted tardiness cost increases, because a larger w_i^{max} allows it, the weighted overtime can be decreased by means of this added flexibility, thus lowering the overall costs. The result is only advantageous from the

Figure 4.5.: Computation time and objective function values for different tardiness factors



ground-handling agency's point of view. For the customers (*i.e.*, airlines), the agents' tardiness may decrease service availability.

4.6. Conclusions and further research

This paper presents a task assignment problem encountered by ground-handling agencies at airports. This problem assumes a variable task-processing time dependent on the agent's arrival time. To the best of our knowledge, this feature has not previously been addressed in the literature.

We propose a branch-and-price algorithm to solve this problem to optimality in a short computation time. Taking advantage of the inherent structure of the problem, the dynamic programming algorithm used to solve the pricing problems can be modeled as a shortest path problem with resource constraints (SPPRC).

We test the performance of the proposed algorithm using real-world data from a German ground-handling agency. The results show that our algorithm outperforms the standard MIP formulation not only in real-world instances but also in semi-artificial instances with different shift schedules and instances of larger size. Additionally, we test two modeling options for our dynamic program formulation (*i.e.*, SPPRC and ESPPRC) and observe that the SPPRC formulation obtains equivalent solutions in lower computation time than an ESPPRC formulation. Furthermore, we conduct a sensitivity analysis to gain insights into the impact of tardiness on solving the problem. We observe that increasing the maximum allowed tardiness increases the computational effort required to obtain the optimal schedule; however, the added flexibility contributes to a reduction in the objective function value.

Further research needs to be conducted to integrate the current planning problem into other planning stages of the operational workforce planning process (*e.g.*, replanning, tour scheduling, etc.). Furthermore, additional features could be considered such as employee preferences, synchronization of agents' arrival, or the possibility that agents might leave their counters earlier.

Acknowledgements

This research was supported by the Erich-Becker-Stiftung, a foundation of Fraport AG. The authors are grateful to the anonymous referees for helpful and constructive suggestions.

5. Conclusions and outlook

5.1. Conclusions

This dissertation presents three articles on workforce scheduling dealing with decision problems from different areas of application. For each of these scheduling problems, a formal definition, an overview of related literature, and a mixed- integer programming formulation are given. The inherent combinatorial nature of the problems under consideration, however, makes their direct solution impractical due to the high computational effort required for it. To this end, all three articles propose exact and/or heuristic solution approaches that derive optimal and/or near-optimal solutions in short computation time for artificial and real-world test instances.

The first article presents a tour scheduling problem for check-in agents at airports and proposes a rolling planning horizon-based heuristic to solve it. The results from the numerical study show that the proposed algorithm is able to produce near-optimal solutions in short computation time, although the selection of the heuristic parameters is crucial to achieve this. Lastly, the results from the sensitivity analysis show there is a trade-off between the overall costs and the scheduling flexibility provided by more qualified employees.

The second article addresses a multiperiod technician routing and scheduling problem from an external maintenance provider and proposes two branch-and-price algorithms to solve it. The numerical tests present a comparison between different decomposition schemes for the column generation phase of the algorithms. The results show that a decomposition with smaller and easier-to-solve subproblems is preferable for a good overall performance. In addition, results also show that time windows can span over multiple period has an significant impact on scheduling costs and computation time.

The third article presents a task scheduling problem for check-in counters personnel at airports and proposes a branch-and-price algorithms to solve it. By exploiting the structure of the problem, the algorithms used to solve the subproblems can be simplified in order to improve its performance. The results from the numerical tests corroborate this. Finally, a sensitivity analysis on the tardiness limit indicates that a higher tardiness limit contributes to lower operation costs, although it can have a negative impact on customer satisfaction.

Despite the fact that all decision models presented in this thesis share many similarities, they also have many differences. Table 5.1 shows a comparison of main elements of the decision models on each chapter chapter, such as the decisions considered, the planning horizon, the characteristics of the workforce, and their application areas. In summary, several distinctions can be made. First, the decision model in Chapter 2 considers shift scheduling and days-off scheduling decision, whereas the models in Chapter 3 and 4 consider fixed schedules. Second, the decision model from Chapter 3 differs from that on Chapter 4 on the fact that teams of agents are built on a daily basis and that tasks can be served on multiple days. Third, the planning horizon of all articles is one month, one week, and one day for the decision models of Chapter 2, 3, and 4, respectively. Last, all decision models consider employees with multiple non-hierarchical qualifications, although in Chapter 3 the decision model considers additionally proficiency levels for each qualification.

Table 5.1.: Comparison of decision models per chapter

Chapter	Decisions	Planning Horizon	Workforce	Application
2	Shift Scheduling, Days-off scheduling	1 month	Multiple, non-hierarchical skills	Airport
3	Task assignment, Task Scheduling, Routing, Team building	1 week	Multiple skills with proficiency levels	Maintenance provider
4	Task assignment, Routing	1 day	Multiple, non-hierarchical skills	Airport

Moreover, in addition to a MIP formulation, all chapters propose alternative solution approaches for the presented decision models. These solution approaches share common elements although several distinctions can be made. Table 5.2 presents a comparison of the key elements of the solution methods of each chapter including the name of the approach and information on the subproblems (*i.e.*, type of problem, solution method, and how are the subproblems connected for the overall solution). As shown in this table, the main distinction among all solution methods is that the approach proposed in Chapter 2 is an heuristic, whereas the ones in Chapter 3 and 4 are exact methods where optimality of the solutions is guaranteed. Second, all solution methods, in short, attempt to solve the original problem by decomposing it into smaller subproblems. An overall solution is then obtained through a connection of the solution of the subproblems. In the case of the approach of Chapter 2 this is achieved by a modification of the original formulation, whereas in the approaches presented in Chapter 3 and 4 apply a Dantzig-Wolfe decompositions for this purpose. Last, in Chapter 3 alternative Dantzig-Wolfe decompositions are proposed, one of which considers subproblems that belong to the Elementary Shortest Path Problem with Resource Constraints (ESPPRC) class. The approach presented in Chapter 4, however, exploits certain characteristics of the problem at hand to formulate the subproblems as Shortest Path Problems with Resource Constraints (SPPRC), which require less computational effort for their solution.

Table 5.2.: Comparison of solution methods per chapter

Chapter	Approach	Subproblems		
		Type	Solution	Connection
2	Rolling Planning Horizon heuristic	Reduced version of original problem	MIP solver	Modified tour building constraints
3	Branch-and-price	1) Reduced version of original problem	1) MIP solver	1) Master problem solved with Column Generation
		2) Elementary Shortest Path with Resource Constaints (ESPPRC)	2) Labeling algorithm	2) Master problem solved with Column Generation
4	Branch-and-price	Shortest Path with Resource Constraints (SPPRC)	Labeling algorithm	Master problem solved with Column Generation

5.2. Further research directions

Notwithstanding that the presented scheduling problems correspond to different stages of the workforce scheduling process, future research can be oriented towards their integration into a single decision support system. One option can be to integrate them in a hierarchical planning framework, where that problems are solved sequentially with information links to connect them. Likewise, another integration option can be to incorporate several decisions of different problems into the formulation of a single decision model.

As all presented decision models are workforce related, further research can be oriented to common directions such as incorporating employees preferences, fairness, workload balance, meal-break placements, etc. On the other hand, all models from this thesis assume information is deterministic and known in advance. Hence, robust optimization approaches need to be developed in order to accommodate for the stochastic effects of the information.

In relation to the solution approaches proposed in this thesis, future research is required for an improvement in their performance. In the case of Chapter 2, alternative solution approaches (*e.g.*, tabu search, simulated annealing, etc.) can be developed for solving the subproblems of the rolling horizon-based heuristic. For Chapter 3 and 4, alternative ideas from existing solution approaches for similar routing problems can be implemented. Such ideas include, for example, the implementation acceleration and stabilization strategies in the column generation phase of the algorithms, or the implementation of ng-paths (Baldacci et al., 2011), bi-directional labeling (Liberatore et al., 2010) or hybrid methods (Gendreau et al., 2016) for the solution of the subproblems.

Since scheduling problems can be found in more areas of application than those mentioned in this thesis, further research involves transferring the proposed solution approaches to scheduling problems in other applications (*e.g.*, job-shop scheduling, production scheduling with sequence-dependent setup times, etc.).

Bibliography

Airport Research Center (2009). *Study on the Impact of Directive 96/67/EC on Ground Handling Services 1996-2007.* Aachen.

Akjiratikarl, C., P. Yenradee, and P. R. Drake (2007). Pso-based algorithm for home care worker scheduling in the uk. *Computers & Industrial Engineering 53*(4), 559–583.

Alfares, H. K. (2004). Survey, Categorization, and Comparison of Recent Tour Scheduling Literature. *Annals of Operations Research 127*(1-4), 145–175.

Baker, K. (1977). An experimental study of the effectiveness of rolling schedules in production planning. *Decision Sciences 8*(1), 19–27.

Baker, K. and D. Peterson (1979). An analytic framework for evaluating rolling schedules. *Management Science 25*(4), 341–351.

Baldacci, R., A. Mingozzi, and R. Roberti (2011). New route relaxation and pricing strategies for the vehicle routing problem. *Operations research 59*(5), 1269–1283.

Bard, J. F. and H. W. Purnomo (2005). Hospital-wide reactive scheduling of nurses with preference considerations. *IIE Transactions 37*(7), 589–608.

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

Barrera, D., N. Velasco, and C.-A. Amaya (2012). A network-based approach to the multi-activity combined timetabling and crew scheduling problem: Workforce scheduling for public health policy implementation. *Computers & Industrial Engineering 63*(4), 802–812.

Bartholdi, J. J. I. (1981). A guaranteed-accuracy round-off algorithm for cyclic scheduling and set covering. *Operations Research 29*(3), 501–510.

Bertels, S. and T. Fahle (2006). A hybrid setup for a hybrid scenario: combining heuristics for the home health care problem. *Computers & Operations Research 33*(10), 2866–2890.

Blakeley, F., B. Arguello, B. Cao, W. Hall, and J. Knolmajer (2003). Optimizing Periodic Maintenance Operations for Schindler Elevator Corporation. *Interfaces 33*(1), 67–79.

Bode, C. and S. Irnich (2014a). The shortest-path problem with resource constraints with (k,2)-loop elimination and its application to the capacitated arc-routing problem. *European Journal of Operational Research 238*(2), 415–426.

Bode, C. and S. Irnich (2014b). The shortest-path problem with resource constraints with -loop elimination and its application to the capacitated arc-routing problem. *European Journal of Operational Research 238*(2), 415–426.

Bostel, N., P. Dejax, P. Guez, and F. Tricoire (2008). Multiperiod Planning and Routing on a Rolling Horizon for Field Force Optimization Logistics.

In B. Golden, S. Raghavan, and E. Wasil (Eds.), *The Vehicle Routing Problem: Latest Advances and New Challenges*, Volume 43 of *Operations Research/Computer Science Interfaces*, Chapter 3, pp. 503–525. Boston, MA: Springer US.

Boussier, S., D. Feillet, and M. Gendreau (2006). An exact algorithm for team orienteering problems. *4OR 5*(3), 211–230.

Campbell, G. M. (1999). Cross-utilization of workers whose capabilities differ. *Management Science 45*(5), 722–732.

Campbell, G. M. and M. Diaby (2002). Development and evaluation of an assignment heuristic for allocating cross-trained workers. *European Journal of Operational Research 138*(1), 9–20.

Caseau, Y. and P. Koppstein (1992). A cooperative-architecture expert system for solving large time/travel assignment problems. In *Database and Expert Systems Applications*, pp. 197–202. Springer.

Castillo-Salazar, J. A., D. Landa-Silva, and R. Qu (2014). Workforce scheduling and routing problems: literature survey and computational study. *Annals of Operations Research*, 1–29.

Cheng, E. and J. L. Rich (1998, 04). A home health care routing and scheduling problem. Technical report, Rice University.

Choi, E. and D.-W. Tcha (2007). A column generation approach to the heterogeneous fleet vehicle routing problem. *Computers & Operations Research 34*(7), 2080–2095.

Cordeau, J.-F., G. Desaulniers, J. Desrosiers, M. M. Solomon, and F. Soumis (2001). The VRP with time windows. In P. Toth and D. Vigo

(Eds.), *The vehicle routing problem*, pp. 157–193. Society for Industrial and Applied Mathematics.

Cordeau, J.-F., G. Laporte, F. Pasin, and S. Ropke (2010a). Scheduling technicians and tasks in a telecommunications company. *Journal of Scheduling 13*(4), 393–409.

Cordeau, J.-F., G. Laporte, F. Pasin, and S. Ropke (2010b). Scheduling technicians and tasks in a telecommunications company. *Journal of Scheduling 13*(4), 393–409.

Corominas, A., J. Olivella, and R. Pastor (2010). A model for the assignment of a set of tasks when work performance depends on experience of all tasks involved. *International Journal of Production Economics 126*(2), 335–340.

Corominas, A., R. Pastor, and E. Rodríguez (2006). Rotational allocation of tasks to multifunctional workers in a service industry. *International Journal of Production Economics 103*(1), 3–9.

Dantzig, G. (1954). A Comment on Edie's "Traffic Delays at Toll Booths". *Journal of the Operations Research Society of America 2*(3), 339–341.

Day, P. R. and D. M. Ryan (1997). Flight attendant rostering for short-haul airline operations. *Operations Research 45*(5), 649–661.

Desrochers, M., J. Desrosiers, and M. M. Solomon (1992). A new optimization algorithm for the vehicle routing problem with time windows. *Operations Research 40*(2), 342–354.

Dohn, A., E. Kolind, and J. Clausen (2009a). The manpower allocation problem with time windows and job-teaming constraints: A branch-and-price approach. *Computers & Operations Research 36*(4), 1145–1157.

Dohn, A., E. Kolind, and J. Clausen (2009b). The manpower allocation problem with time windows and job-teaming constraints: A branch-and-price approach. *Computers & Operations Research 36*(4), 1145–1157.

Dror, M. (1994). Note on the complexity of the shortest path models for column generation in VRPTW. *Operations Research 42*(5), 977–978.

Edison, E. and T. Shima (2011). Integrated task assignment and path optimization for cooperating uninhabited aerial vehicles using genetic algorithms. *Computers & Operations Research 38*(1), 340–356.

Eitzen, G., D. Panton, and G. Mills (2004). Multi-Skilled Workforce Optimisation. *Annals of Operations Research 127*(1-4), 359–372.

Ernst, A., H. Jiang, and M. Krishnamoorthy (2004a). An annotated bibliography of personnel scheduling and rostering. *Annals of Operations Research 127*(1-4), 21–144.

Ernst, A., H. Jiang, and M. Krishnamoorthy (2004b). Staff scheduling and rostering: A review of applications, methods and models. *European Journal of Operational Research 153*(1), 3–27.

Eveborn, P., P. Flisberg, and M. Rönnqvist (2006a). LAPS CARE - An operational system for staff planning of home care. *European Journal of Operational Research 171*(3), 962–976.

Eveborn, P., P. Flisberg, and M. Rönnqvist (2006b). Laps care- an operational system for staff planning of home care. *European Journal of Operational Research 171*(3), 962–976.

Feillet, D. (2010). A tutorial on column generation and branch-and-price for vehicle routing problems. *4OR 8*, 407–424.

Feillet, D., P. Dejax, M. Gendreau, and C. Gueguen (2004). An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems. *Networks 44*(3), 216–229.

Ferland, J. A. and P. Michelon (1988). The vehicle scheduling problem with multiple vehicle types. *Journal of the Operational Research Society 39*(6), 577–583.

Francis, P., K. Smilowitz, and M. Tzur (2006). The period vehicle routing problem with service choice. *Transportation Science 40*(4), 439–454.

Francis, P. M., K. R. Smilowitz, and M. Tzur (2008). The Period Vehicle Routing Problem and its Extensions. In B. Golden, S. Raghavan, and E. Wasil (Eds.), *The Vehicle Routing Problem: Latest Advances and New Challenges*, Volume 43 of *Operations Research/Computer Science Interfaces*, Chapter 1, pp. 73–102. Boston, MA: Springer US.

Gendreau, M., D. Manerba, and R. Mansini (2016). The multi-vehicle traveling purchaser problem with pairwise incompatibility constraints and unitary demands: A branch-and-price approach. *European Journal of Operational Research 248*(1), 59–71.

Google (2016). The Google Maps Distance Matrix API. https://developers.google.com/maps/documentation/distance-matrix/intro.

Goossens, D., S. Polyakovskiy, F. C. R. Spieksma, and G. J. Woeginger (2012). Between a rock and a hard place: the two-to-one assignment problem. *Mathematical Methods of Operations Research 76*(2), 223–237.

Gronalt, M. (2003). Workforce planning and allocation for mid-volume truck manufacturing: a case study. *International journal of production research 4*(3).

Hojati, M. and A. S. Patil (2011). An integer linear programming-based heuristic for scheduling heterogeneous, part-time service employees. *European Journal of Operational Research 209*(1), 37–50.

Ioachim, I., S. Gélinas, J. Desrosiers, and F. Soumis (1998). A Dynamic Programming Algorithm for the Shortest Path Problem with Time Windows. *Networks 31*(3), 193–204.

Irnich, S. (2008). Resource extension functions: Properties, inversion, and generalization to segments. *OR Spectrum 30*(1), 113–148.

Irnich, S., G. Desaulniers, and M. M. Solomon (2005). Shortest path problems with resource constraints. *Column generation 6730*, 33–65.

Jiang, J., K. M. Ng, K. L. Poh, and K. M. Teo (2014). Vehicle routing problem with a heterogeneous fleet and time windows. *Expert Systems with Applications 41*(8), 3748–3760.

Kohl, N. and J. Desrosiers (1999). 2-path cuts for the vehicle routing problem with time windows. *Transportation Science* (1), 101–116.

Kovacs, A. a., S. N. Parragh, K. F. Doerner, and R. F. Hartl (2011). Adaptive large neighborhood search for service technician routing and scheduling problems. *Journal of Scheduling 15*(5), 579–600.

Kovacs, A. A., S. N. Parragh, K. F. Doerner, and R. F. Hartl (2012). Adaptive large neighborhood search for service technician routing and scheduling problems. *Journal of Scheduling 15*(5), 579–600.

Krishnamoorthy, M., A. T. Ernst, and D. Baatar (2012). Algorithms for large scale shift minimisation personnel task scheduling problems. *European Journal of Operational Research 219*(1), 34–48.

Li, Y., A. Lim, and B. Rodrigues (2005). Manpower allocation with time windows and job-teaming constraints. *Naval Research Logistics (NRL) 52*(4), 302–311.

Liang, T. T. and B. B. Buclatin (1988). Improving the utilization of training resources through optimal personnel assignment in the U.S. Navy. *European Journal of Operational Research 33*(2), 183–190.

Liberatore, F., G. Righini, and M. Salani (2010). A column generation algorithm for the vehicle routing problem with soft time windows. *4OR 9*(1), 49–82.

Lieder, A., D. Moeke, G. Koole, and R. Stolletz (2015). Task scheduling in long-term care facilities: A client-centered approach. *Operations Research for Health Care 6*, 11 – 17.

Lim, A., B. Rodrigues, and L. Song (2004). Manpower allocation with time windows. *Journal of the Operational Research Society*, 1178–1186.

Liu, C., N. Yang, W. Li, J. Lian, S. Evans, and Y. Yin (2013). Training and assignment of multi-skilled workers for implementing seru production systems. *The International Journal of Advanced Manufacturing Technology 69*(5-8), 937–959.

Loucks, J. S. and F. R. Jacobs (1991). Tour scheduling and task assignment of a heterogeneous work force: A heuristic approach. *Decision Sciences 22*(4), 719–738.

Love Jr., R. and J. Hoey (1990). Management science improves fast-food operations. *Interfaces 20*(2), 21–29.

Lübbecke, M. E. and J. Desrosiers (2005). Selected topics in column generation. *Operations Research 53*(6), 1007–1023.

Miller, J. L. and L. S. Franz (1996). A binary-rounding heuristic for multi-period variable-task-duration assignment problems. *Computers & Operations Research 23*(8), 819–828.

Modigliani, F. (1955). Production planning over time and the nature of the expectation and planning horizon. *Econometrica, Journal of the Econometric Society 23*(1), 46–66.

Olivella, J., A. Corominas, and R. Pastor (2013). Task assignment considering cross-training goals and due dates. *International Journal of Production Research 51*(3), 952–962.

Pessoa, A., E. Uchoa, and M. Poggi de Aragão (2009). A robust branch-cut-and-price algorithm for the heterogeneous fleet vehicle routing problem. *Networks 54*(4), 167–177.

Pillac, V., C. Guéret, and a. L. Medaglia (2012). A parallel matheuristic for the technician routing and scheduling problem. *Optimization Letters 7*(7), 1525–1535.

Rong, A. and M. Grunow (2009). Shift designs for freight handling personnel at air cargo terminals. *Transportation Research Part E: Logistics and Transportation Review 45*(5), 725–739.

Savelsbergh, M. W. P. (1985). Local search in routing problems with time windows. *Annals of Operations Research 4*(1), 285–305.

S&HE International Air Transport Consultancy (2002). *Study on the quality and efficiency of ground handling services at EU airports as a result of the implementation of Council Directive 9667/EC*. London.

Smet, P., T. Wauters, M. Mihaylov, and G. V. Berghe (2014). The shift minimisation personnel task scheduling problem: A new hybrid approach and computational insights. *Omega 46*, 64–73.

Solomon, M. and J. Desrosiers (1988a). Survey Paper-Time Window Constrained Routing and Scheduling Problems. *Transportation Science 22*(1), 1–13.

Solomon, M. M. (1987). Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research 35*(2), 254–265.

Solomon, M. M. and J. Desrosiers (1988b). Survey paper - Time window constrained routing and scheduling problems. *Transportation science 22*(1), 1–13.

Souffriau, W., P. Vansteenwegen, G. Vanden Berghe, and D. Van Oudheusden (2013). The multiconstraint team orienteering problem with multiple time windows. *Transportation Science 47*(1), 53–63.

Steer Davies Gleave (2010). *Possible revision of Directive 96/67/EC on access to the groundhandling market at Community airports*. London.

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

Stolletz, R. and E. Zamorano (2014). A rolling planning horizon heuristic for scheduling agents with different qualifications. *Transportation Research Part E: Logistics and Transportation Review 68*, 39 – 52.

Tang, H., E. Millerhooks, and R. Tomastik (2007). Scheduling technicians for planned maintenance of geographically distributed equipment. *Transportation Research Part E: Logistics and Transportation Review 43*(5), 591–609.

The World Bank (2016). Compensation of employees (% of expense). World Data Bank: http://databank.worldbank.org/.

Tsang, E. and C. Voudouris (1997). Fast local search and guided local search and their application to British Telecom's workforce scheduling problem. *Operations Research Letters 20*(3), 119–127.

Van den Bergh, J., J. Beliën, P. De Bruecker, E. Demeulemeester, and L. De Boeck (2013). Personnel scheduling: A literature review. *European Journal of Operational Research 226*(3), 367 – 385.

Xu, J. and S. Chiu (2001). Effective heuristic procedures for a field technician scheduling problem. *Journal of Heuristics 7*(5), 495–509.

Yang, R. (1996). Solving a workforce management problem with constraint programming. In *The 2nd International Conference on the Practical Application of Constraint Technology*, pp. 373–387. The Practical Application Company Ltd.

A. Realistic instances from Chapter 4

In the following section detailed informations on realistic instances A1-A4 is presented. Tables A.1 to A.4 show the task index, the counter opening time, the occupation time (in minutes), the agent requirements, and the skills required. Tables A.5 to A.8 present information regarding the workforce, including: agent index, qualifications, shift start time, and shift end time. Last, tables A.9 to A.12 display the travel time (in minutes) from and to each task (including depot). This information is available at http://stolletz.bwl.uni-mannheim.de/en/library.

We also use the following values for the remaining parameters:

- $w_{ik}^{max} = 0.5 \cdot (b_i - a_i) \;\; \forall i \in I, \forall k \in K$
- $out_i^{max} = v_i - 1 \;\; \forall i \in I$
- The maximum overtime per agent is set such that the shift length plus overtime cannot exceed 10 hours.
- $[\alpha_1, \alpha_2, \alpha_3, \alpha_4] = [0.1, 0.2, 0.3, 0.4]$
- 1 hour computation time limit

Table A.1.: A1- Tasks

Task	Opening	Occ. (m)	Reqs.	q1	q2	q3	q4	q5	q6	q7
i1	75	45	2	0	0	0	0	0	0	1
i2	170	30	1	0	0	0	0	0	0	1
i3	180	105	2	0	0	0	0	0	1	0
i4	180	40	1	0	0	0	0	0	0	1
i5	200	30	1	0	0	0	0	0	0	1
i6	210	30	1	0	0	0	0	0	0	1
i7	220	30	1	0	0	0	0	0	0	1
i8	230	40	1	0	0	0	0	0	0	1
i9	240	45	1	0	0	0	0	0	0	1
i10	250	40	2	0	0	0	0	0	0	1
i11	290	90	1	0	0	0	1	0	0	0
i12	290	120	2	1	0	0	0	0	0	0
i13	290	110	1	1	0	0	0	0	0	0
i14	300	90	1	0	0	0	1	0	0	0
i15	310	80	1	0	0	0	0	0	1	0
i16	310	90	1	0	0	0	1	0	0	0
i17	310	30	1	0	0	0	0	0	0	1
i18	360	120	2	1	0	0	0	0	0	0
i19	390	140	2	1	0	0	0	0	0	0
i20	400	90	1	0	0	0	1	0	0	0
i21	405	30	1	0	0	0	0	0	0	1
i22	415	30	1	0	0	0	0	0	0	1
i23	415	30	1	0	0	0	0	0	0	1
i24	495	30	1	0	0	0	0	0	0	1
i25	495	90	1	0	0	0	1	0	0	0
i26	505	90	1	0	0	0	1	0	0	0
i27	505	90	1	0	0	0	1	0	0	0
i28	525	40	1	0	0	0	0	0	0	1
i29	575	40	1	0	0	0	0	0	0	1
i30	585	90	1	0	0	0	1	0	0	0
i31	585	40	1	0	0	0	0	0	0	1
i32	630	40	1	0	0	0	0	0	0	1
i33	635	110	1	0	0	1	0	0	0	0
i34	655	80	1	0	1	0	0	0	0	0
i35	655	45	2	0	0	0	0	0	0	1
i36	660	30	1	0	0	0	0	0	0	1
i37	680	30	1	0	0	0	0	0	0	1
i38	695	110	1	1	0	0	0	0	0	0
i39	725	30	1	0	0	0	0	0	0	1
i40	735	30	1	0	0	0	0	0	0	1
i41	760	45	1	0	0	0	0	0	0	1
i42	760	105	2	0	0	0	0	0	1	0
i43	765	45	1	0	0	0	0	0	0	1
i44	770	140	1	1	0	0	0	0	0	0
i45	780	120	2	1	0	0	0	0	0	0
i46	800	120	2	1	0	0	0	0	0	0
i47	815	90	1	0	0	0	1	0	0	0
i48	825	90	1	0	0	0	1	0	0	0
i49	845	45	2	0	0	0	0	0	0	1
i50	880	120	2	1	0	0	0	0	0	0
i51	880	40	1	0	0	0	0	0	0	1
i52	885	120	2	1	0	0	0	0	0	0
i53	915	30	1	0	0	0	0	0	0	1
i54	925	30	1	0	0	0	0	0	0	1
i55	930	30	1	0	0	0	0	0	0	1
i56	950	105	2	0	0	0	0	0	1	0
i57	990	110	1	1	0	0	0	0	0	0
i58	1005	90	1	0	0	0	1	0	0	0
i59	1015	90	1	0	0	0	1	0	0	0
i60	1020	90	1	0	0	0	1	0	0	0
i61	1080	30	1	0	0	0	0	0	0	1
i62	1200	120	2	1	0	0	0	0	0	0

Table A.2.: A2- Tasks

Task	Opening	Occ. (m)	Reqs.	q1	q2	q3	q4	q5	q6	q7
i1	75	45	2	0	0	0	0	0	0	1
i2	170	30	1	0	0	0	0	0	0	1
i3	180	105	2	0	0	0	0	0	1	0
i4	180	40	1	0	0	0	0	0	0	1
i5	200	30	1	0	0	0	0	0	0	1
i6	200	40	1	0	0	0	0	0	0	1
i7	210	30	1	0	0	0	0	0	0	1
i8	220	30	1	0	0	0	0	0	0	1
i9	240	45	1	0	0	0	0	0	0	1
i10	275	30	1	0	0	0	0	0	0	1
i11	280	80	1	0	0	0	0	0	1	0
i12	290	90	1	0	0	0	1	0	0	0
i13	290	120	2	1	0	0	0	0	0	0
i14	290	110	1	1	0	0	0	0	0	0
i15	300	90	1	0	0	0	1	0	0	0
i16	310	90	1	0	0	0	1	0	0	0
i17	310	30	1	0	0	0	0	0	0	1
i18	360	120	2	1	0	0	0	0	0	0
i19	395	120	2	1	0	0	0	0	0	0
i20	400	90	1	0	0	0	1	0	0	0
i21	405	30	1	0	0	0	0	0	0	1
i22	415	30	1	0	0	0	0	0	0	1
i23	415	30	1	0	0	0	0	0	0	1
i24	490	40	1	0	0	0	0	0	0	1
i25	495	90	1	0	0	0	1	0	0	0
i26	505	90	1	0	0	0	1	0	0	0
i27	505	90	1	0	0	0	1	0	0	0
i28	575	40	1	0	0	0	0	0	0	1
i29	585	40	1	0	0	0	0	0	0	1
i30	595	40	1	0	0	0	0	0	0	1
i31	600	110	1	1	0	0	0	0	0	0
i32	605	30	1	0	0	0	0	0	0	1
i33	630	40	1	0	0	0	0	0	0	1
i34	655	80	1	0	1	0	0	0	0	0
i35	655	45	2	0	0	0	0	0	0	1
i36	695	90	2	1	0	0	0	0	0	0
i37	695	110	1	1	0	0	0	0	0	0
i38	705	110	1	0	0	1	0	0	0	0
i39	725	30	1	0	0	0	0	0	0	1
i40	730	40	2	0	0	0	0	0	0	1
i41	735	30	1	0	0	0	0	0	0	1
i42	760	105	2	0	0	0	0	0	1	0
i43	765	45	1	0	0	0	0	0	0	1
i44	770	140	1	1	0	0	0	0	0	0
i45	810	80	2	1	0	0	0	0	0	0
i46	815	90	1	0	0	0	1	0	0	0
i47	825	90	1	0	0	0	1	0	0	0
i48	830	45	2	0	0	0	0	0	0	1
i49	835	30	1	0	0	0	0	0	0	1
i50	885	120	2	1	0	0	0	0	0	0
i51	915	30	1	0	0	0	0	0	0	1
i52	915	30	1	0	0	0	0	0	0	1
i53	925	30	1	0	0	0	0	0	0	1
i54	930	30	1	0	0	0	0	0	0	1
i55	935	105	2	0	0	0	0	0	1	0
i56	955	120	2	1	0	0	0	0	0	0
i57	985	30	1	0	0	0	0	0	0	1
i58	1005	90	3	1	0	0	0	0	0	0
i59	1005	90	1	0	0	0	1	0	0	0
i60	1015	90	1	0	0	0	1	0	0	0
i61	1020	90	1	0	0	0	1	0	0	0
i62	1050	30	1	0	0	0	0	0	0	1
i63	1070	30	1	0	0	0	0	0	0	1
i64	1105	120	2	1	0	0	0	0	0	0
i65	1170	120	2	1	0	0	0	0	0	0
i66	1190	120	2	1	0	0	0	0	0	0

Table A.3.: A3- Tasks

Task	Opening	Occ. (m)	Reqs.	q1	q2	q3	q4	q5	q6	q7
i1	75	45	2	0	0	0	0	0	0	1
i2	170	30	1	0	0	0	0	0	0	1
i3	180	105	2	0	0	0	0	0	1	0
i4	180	40	1	0	0	0	0	0	0	1
i5	200	30	1	0	0	0	0	0	0	1
i6	210	30	1	0	0	0	0	0	0	1
i7	220	30	1	0	0	0	0	0	0	1
i8	230	40	1	0	0	0	0	0	0	1
i9	240	45	1	0	0	0	0	0	0	1
i10	250	40	2	0	0	0	0	0	0	1
i11	290	90	1	0	0	0	1	0	0	0
i12	290	120	2	1	0	0	0	0	0	0
i13	290	110	1	1	0	0	0	0	0	0
i14	300	90	1	0	0	0	1	0	0	0
i15	310	80	1	0	0	0	0	0	1	0
i16	310	90	1	0	0	0	1	0	0	0
i17	310	30	1	0	0	0	0	0	0	1
i18	360	120	2	1	0	0	0	0	0	0
i19	390	140	2	1	0	0	0	0	0	0
i20	400	90	1	0	0	0	1	0	0	0
i21	405	30	1	0	0	0	0	0	0	1
i22	415	30	1	0	0	0	0	0	0	1
i23	415	30	1	0	0	0	0	0	0	1
i24	495	30	1	0	0	0	0	0	0	1
i25	495	90	1	0	0	0	1	0	0	0
i26	505	90	1	0	0	0	1	0	0	0
i27	505	90	1	0	0	0	1	0	0	0
i28	525	40	1	0	0	0	0	0	0	1
i29	575	40	1	0	0	0	0	0	0	1
i30	585	90	1	0	0	0	1	0	0	0
i31	585	40	1	0	0	0	0	0	0	1
i32	630	40	1	0	0	0	0	0	0	1
i33	635	110	1	0	0	1	0	0	0	0
i34	655	80	1	0	1	0	0	0	0	0
i35	655	45	2	0	0	0	0	0	0	1
i36	660	30	1	0	0	0	0	0	0	1
i37	680	30	1	0	0	0	0	0	0	1
i38	695	110	1	1	0	0	0	0	0	0
i39	725	30	1	0	0	0	0	0	0	1
i40	735	30	1	0	0	0	0	0	0	1
i41	760	45	1	0	0	0	0	0	0	1
i42	760	105	2	0	0	0	0	0	1	0
i43	765	45	1	0	0	0	0	0	0	1
i44	770	140	1	1	0	0	0	0	0	0
i45	780	120	2	1	0	0	0	0	0	0
i46	800	120	2	1	0	0	0	0	0	0
i47	815	90	1	0	0	0	1	0	0	0
i48	825	90	1	0	0	0	1	0	0	0
i49	845	45	2	0	0	0	0	0	0	1
i50	880	120	2	1	0	0	0	0	0	0
i51	880	40	1	0	0	0	0	0	0	1
i52	885	120	2	1	0	0	0	0	0	0
i53	915	30	1	0	0	0	0	0	0	1
i54	925	30	1	0	0	0	0	0	0	1
i55	930	30	1	0	0	0	0	0	0	1
i56	950	105	2	0	0	0	0	0	1	0
i57	990	110	1	1	0	0	0	0	0	0
i58	1005	90	1	0	0	0	1	0	0	0
i59	1015	90	1	0	0	0	1	0	0	0
i60	1020	90	1	0	0	0	1	0	0	0
i61	1080	30	1	0	0	0	0	0	0	1
i62	1200	120	2	1	0	0	0	0	0	0

Table A.4.: A4- Tasks

Task	Opening	Occ. (m)	Reqs.	q1	q2	q3	q4	q5	q6	q7
i1	75	45	2	0	0	0	0	0	0	1
i2	170	30	1	0	0	0	0	0	0	1
i3	180	105	2	0	0	0	0	0	1	0
i4	180	40	1	0	0	0	0	0	0	1
i5	200	30	1	0	0	0	0	0	0	1
i6	210	30	1	0	0	0	0	0	0	1
i7	220	30	1	0	0	0	0	0	0	1
i8	240	45	1	0	0	0	0	0	0	1
i9	290	90	1	0	0	0	1	0	0	0
i10	290	120	2	1	0	0	0	0	0	0
i11	290	110	1	1	0	0	0	0	0	0
i12	300	90	1	0	0	0	1	0	0	0
i13	310	90	1	0	0	0	1	0	0	0
i14	350	30	1	0	0	0	0	0	0	1
i15	360	120	2	1	0	0	0	0	0	0
i16	405	30	1	0	0	0	0	0	0	1
i17	415	30	1	0	0	0	0	0	0	1
i18	470	120	2	1	0	0	0	0	0	0
i19	490	40	1	0	0	0	0	0	0	1
i20	495	90	1	0	0	0	1	0	0	0
i21	505	90	1	0	0	0	1	0	0	0
i22	575	40	1	0	0	0	0	0	0	1
i23	575	40	1	0	0	0	0	0	0	1
i24	600	110	1	1	0	0	0	0	0	0
i25	655	80	1	0	1	0	0	0	0	0
i26	655	45	2	0	0	0	0	0	0	1
i27	685	110	1	1	0	0	0	0	0	0
i28	700	30	1	0	0	0	0	0	0	1
i29	725	30	1	0	0	0	0	0	0	1
i30	735	30	1	0	0	0	0	0	0	1
i31	760	105	2	0	0	0	0	0	1	0
i32	815	90	1	0	0	0	1	0	0	0
i33	820	30	1	0	0	0	0	0	0	1
i34	820	120	2	1	0	0	0	0	0	0
i35	825	90	1	0	0	0	1	0	0	0
i36	845	45	2	0	0	0	0	0	0	1
i37	895	40	1	0	0	0	0	0	0	1
i38	915	30	1	0	0	0	0	0	0	1
i39	925	30	1	0	0	0	0	0	0	1
i40	930	30	1	0	0	0	0	0	0	1
i41	940	120	2	1	0	0	0	0	0	0
i42	950	105	2	0	0	0	0	0	1	0
i43	970	30	1	0	0	0	0	0	0	1
i44	975	40	1	0	0	0	0	0	0	1
i45	1005	90	1	0	0	0	1	0	0	0
i46	1005	110	1	1	0	0	0	0	0	0
i47	1015	90	1	0	0	0	1	0	0	0
i48	1020	90	1	0	0	0	1	0	0	0
i49	1055	80	1	0	0	0	0	0	1	0
i50	1090	120	2	1	0	0	0	0	0	0

Table A.5.: A1- Agents

Agent	q1	q2	q3	q4	q5	q6	q7	Shift start	Shift end
k1	1	1	1	1	0	1	1	20	410
k2	1	1	1	1	0	1	1	20	560
k3	1	1	0	1	0	1	1	35	380
k4	1	1	1	1	0	1	1	95	515
k5	1	1	1	1	0	1	1	110	665
k6	1	0	0	1	0	0	1	140	320
k7	1	0	0	0	0	0	1	140	335
k8	1	0	1	0	0	0	1	140	335
k9	1	0	0	1	0	0	1	140	320
k10	1	0	0	0	0	1	1	140	335
k11	1	1	1	1	0	1	1	155	665
k12	1	0	0	0	0	1	1	155	410
k13	1	0	0	1	0	1	1	155	590
k14	1	1	1	1	0	1	1	185	710
k15	1	0	0	1	0	1	1	185	365
k16	1	0	1	1	0	1	1	215	515
k17	1	0	0	1	0	1	1	215	500
k18	1	0	0	0	0	0	1	230	410
k19	1	0	0	1	0	1	1	230	410
k20	1	0	0	1	0	1	1	275	590
k21	1	1	1	1	0	1	1	410	920
k22	1	1	1	1	0	1	1	455	785
k23	1	1	1	1	0	1	1	470	995
k24	1	0	0	1	0	1	1	470	815
k25	1	1	1	1	0	1	1	530	1040
k26	1	1	0	0	0	1	1	530	770
k27	1	1	0	1	0	0	1	560	995
k28	1	0	0	1	0	1	1	590	1025
k29	1	1	1	1	1	1	1	605	950
k30	1	1	1	1	0	1	1	605	950
k31	1	1	1	1	0	1	1	620	890
k32	1	1	1	1	0	1	1	620	1025
k33	1	1	0	1	0	1	1	620	800
k34	1	1	0	1	0	0	1	620	935
k35	1	0	0	0	0	1	1	635	815
k36	1	0	0	1	0	1	1	635	815
k37	1	1	1	1	0	1	1	650	1205
k38	1	0	0	1	0	1	1	680	935
k39	1	0	0	1	0	0	1	680	1010
k40	1	1	1	1	0	1	1	710	1205
k41	1	1	1	1	0	1	1	755	1010
k42	1	1	0	1	0	1	1	890	1205

Table A.6.: A2- Agents

Agent	q1	q2	q3	q4	q5	q6	q7	Shift start	Shift end
k1	1	1	1	1	0	1	1	20	365
k2	1	1	0	1	0	1	1	20	350
k3	1	0	1	1	0	1	1	35	500
k4	1	0	0	1	0	0	1	95	590
k5	1	1	1	1	0	1	1	110	665
k6	1	0	1	0	0	0	1	140	335
k7	1	0	0	1	0	0	1	140	320
k8	1	0	0	1	0	0	1	140	335
k9	1	0	0	1	0	1	1	140	335
k10	1	1	1	1	0	1	1	140	320
k11	1	1	1	1	0	1	1	155	725
k12	1	0	0	1	0	1	1	155	605
k13	1	0	0	1	0	1	1	185	515
k14	1	0	0	0	0	0	1	185	365
k15	1	0	0	0	0	1	1	185	515
k16	1	1	1	1	0	1	1	230	590
k17	1	1	1	1	0	1	1	275	665
k18	1	0	1	1	0	0	1	275	710
k19	1	1	1	1	0	1	1	410	920
k20	1	0	0	1	0	1	1	470	965
k21	1	1	1	1	0	1	1	530	1040
k22	1	1	0	0	0	1	1	530	770
k23	1	0	0	1	0	0	1	530	1025
k24	1	1	1	1	0	1	1	545	890
k25	1	1	0	1	0	1	1	560	1025
k26	1	1	1	1	0	1	1	560	830
k27	1	1	0	1	0	1	1	575	965
k28	1	0	0	1	0	1	1	590	1025
k29	1	1	1	1	1	1	1	605	1010
k30	1	1	1	1	0	1	1	605	1115
k31	1	1	1	1	0	1	1	620	1175
k32	1	1	1	1	0	1	1	620	920
k33	1	1	1	1	0	1	1	650	1205
k34	1	0	0	0	0	1	1	680	1205
k35	1	1	1	1	0	1	1	680	1115
k36	1	1	1	1	0	1	1	740	1115
k37	1	1	0	1	0	0	1	875	1175
k38	1	1	1	1	1	1	1	890	1205

Table A.7.: A3- Agents

Agent	q1	q2	q3	q4	q5	q6	q7	Shift start	Shift end
k1	1	0	0	0	0	1	1	20	410
k2	1	1	1	1	0	1	1	20	410
k3	1	0	0	1	0	0	1	35	500
k4	1	1	1	1	0	1	1	95	410
k5	1	1	0	1	0	1	1	110	665
k6	1	1	1	1	0	1	1	140	335
k7	1	1	1	1	0	1	1	140	320
k8	1	0	0	1	0	1	1	140	455
k9	1	0	1	1	0	1	1	140	335
k10	1	1	1	1	0	1	1	155	590
k11	1	1	1	1	0	1	1	170	740
k12	1	1	1	1	0	1	1	170	515
k13	1	0	0	0	0	0	1	170	755
k14	1	0	0	1	0	0	1	185	515
k15	1	1	0	1	0	1	1	200	380
k16	1	0	0	1	0	1	1	200	410
k17	1	1	0	0	0	1	1	200	380
k18	1	0	0	1	0	1	1	230	800
k19	1	1	1	1	0	1	1	230	410
k20	1	0	0	0	0	1	1	275	800
k21	1	0	0	1	0	0	1	380	860
k22	1	0	0	1	0	1	1	380	860
k23	1	1	1	1	0	1	1	380	860
k24	1	0	1	0	0	0	1	380	860
k25	1	1	1	1	0	1	1	380	860
k26	1	0	0	1	0	1	1	380	860
k27	1	0	0	1	0	1	1	380	860
k28	1	1	1	1	0	1	1	410	920
k29	1	0	0	1	0	1	1	455	725
k30	1	0	0	1	0	0	1	530	860
k31	1	0	0	0	0	0	1	530	1040
k32	1	0	0	0	0	1	1	530	770
k33	1	1	1	1	0	1	1	530	935
k34	1	1	1	1	0	1	1	560	995
k35	1	1	1	1	0	1	1	575	725
k36	1	1	0	1	0	0	1	575	785
k37	1	0	0	0	0	0	1	590	1025
k38	1	1	1	1	0	1	1	605	950
k39	1	1	1	1	0	1	1	605	950
k40	1	1	1	1	1	1	1	620	935
k41	1	1	1	1	0	1	1	635	995
k42	1	0	0	1	0	1	1	635	815
k43	1	0	0	1	0	1	1	650	1205
k44	1	1	0	1	0	1	1	650	1205
k45	1	1	1	1	0	1	1	680	1010
k46	1	1	1	1	0	1	1	695	1025
k47	1	1	1	1	0	1	1	710	1205
k48	1	0	0	1	0	1	1	755	1025

Table A.8.: A4- Agents

Agent	q1	q2	q3	q4	q5	q6	q7	Shift start	Shift end
k1	1	0	0	0	0	1	1	20	515
k2	1	1	1	1	0	1	1	20	605
k3	1	0	0	1	0	1	1	35	605
k4	1	1	1	1	0	0	1	110	545
k5	1	1	1	1	0	1	1	140	335
k6	1	1	1	1	0	1	1	140	335
k7	1	1	1	1	0	1	1	140	500
k8	1	1	0	0	0	1	1	140	320
k9	1	0	0	1	0	0	1	155	665
k10	1	0	0	1	0	1	1	155	590
k11	1	0	0	0	0	0	1	185	695
k12	1	1	1	1	0	1	1	410	920
k13	1	1	0	1	0	1	1	425	605
k14	1	1	1	1	0	1	1	440	1010
k15	1	0	0	1	0	1	1	530	1100
k16	1	1	1	1	0	1	1	530	1010
k17	1	0	0	0	0	1	1	530	770
k18	1	1	1	1	0	1	1	560	1025
k19	1	1	1	1	0	1	1	590	1025
k20	1	0	0	1	0	1	1	605	1070
k21	1	1	1	1	0	1	1	620	1100
k22	1	0	0	1	0	1	1	665	1025
k23	1	1	0	1	0	1	1	680	1100
k24	1	1	1	1	0	1	1	695	1010
k25	1	0	0	1	0	1	1	695	1070
k26	1	0	1	1	0	1	1	785	1100
k27	1	1	1	1	0	1	1	830	1010
k28	1	1	1	1	0	1	1	845	1025
k29	1	1	0	1	0	0	1	845	1025

Table A.9.: A1- Travel times

	o	i1	i2	i3	i4	i5	i6	i7	i8	i9	i10	i11	i12	i13	i14	i15	i16	i17	i18	i19	i20	i21	i22	i23	i24	i25	i26	i27	i28	i29	i30	i31	i32	i33	i34	i35	i36	i37	i38	i39	i40	i41	i42	i43	i44	i45	i46	i47	i48	i49	i50	i51	i52	i53	i54	i55	i56	i57	i58	i59	i60	i61	i62	$\bar{o}$
o	0	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	0
i1	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	5
i2	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i3	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i4	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i5	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i6	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i7	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i8	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i9	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i10	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i11	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	5
i12	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	5
i13	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i14	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i15	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i16	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i17	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	5
i18	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i19	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i20	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i21	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i22	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i23	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i24	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i25	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i26	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i27	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i28	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i29	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	5
i30	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i31	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i32	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	5
i33	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i34	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i35	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i36	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i37	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i38	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i39	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i40	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i41	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i42	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	5
i43	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	5
i44	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i45	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i46	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i47	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i48	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	5
i49	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i50	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i51	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i52	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i53	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i54	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i55	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i56	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i57	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i58	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i59	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i60	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	5
i61	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i62	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
$\bar{o}$	0	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	0

Table A.10.: A2- Travel times

	o	i1	i2	i3	i4	i5	i6	i7	i8	i9	i10	i11	i12	i13	i14	i15	i16	i17	i18	i19	i20	i21	i22	i23	i24	i25	i26	i27	i28	i29	i30	i31	i32	i33	i34	i35	i36	i37	i38	i39	i40	i41	i42	i43	i44	i45	i46	i47	i48	i49	i50	i51	i52	i53	i54	i55	i56	i57	i58	i59	i60	i61	i62	i63	i64	i65	i66	*ō*
o	0	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	0
i1	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	5
i2	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	5
i3	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	5
i4	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	5
i5	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	5
i6	5	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5
i7	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	5
i8	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	5
i9	5	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5
i10	5	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5
i11	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	5
i12	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	5
i13	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	5
i14	5	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5
i15	5	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5
i16	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	5
i17	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	5
i18	5	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5
i19	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	5
i20	5	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5
i21	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	5
i22	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	5
i23	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	5
i24	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	5
i25	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	5
i26	5	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5
i27	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	5
i28	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	5
i29	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	5
i30	5	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5
i31	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	5
i32	5	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5
i33	5	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5
i34	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	5
i35	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	5
i36	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	5
i37	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	5
i38	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	5
i39	5	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5
i40	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	5
i41	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	5
i42	5	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5
i43	5	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5
i44	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	5
i45	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	5
i46	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	5
i47	5	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5
i48	5	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5
i49	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	5
i50	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	5
i51	5	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5
i52	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	5
i53	5	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5
i54	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	5
i55	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	5
i56	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	5
i57	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	5
i58	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	5
i59	5	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5
i60	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	5
i61	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	5
i62	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	10	10	10	0	0	5	0	10	5	5	0	0	10	5	5	10	10	5	0	5	0	0	0	0	10	5	0	0	0	5	10	5	5	5
i63	5	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5
i64	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	0	0	0	10	10	5	10	0	5	5	10	10	0	5	5	0	0	5	10	5	10	10	10	10	0	5	10	10	10	5	0	5	5	5
i65	5	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5
i66	5	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5	5	5	5	5	0	5	5	0	0	5	5	5	0	0	5	5	0	5	0	5	5	5	5	5	0	5	5	5	0	5	0	0	5
ō	0	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	0

Table A.11.: A3- Travel times

	o	i1	i2	i3	i4	i5	i6	i7	i8	i9	i10	i11	i12	i13	i14	i15	i16	i17	i18	i19	i20	i21	i22	i23	i24	i25	i26	i27	i28	i29	i30	i31	i32	i33	i34	i35	i36	i37	i38	i39	i40	i41	i42	i43	i44	i45	i46	i47	i48	i49	i50	i51	i52	i53	i54	i55	i56	i57	i58	i59	i60	i61	i62	ō
o	0	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	0
i1	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	5
i2	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i3	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i4	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i5	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i6	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i7	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i8	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i9	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i10	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i11	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	5
i12	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	5
i13	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i14	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i15	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i16	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i17	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	5
i18	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i19	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i20	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i21	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i22	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i23	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i24	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i25	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i26	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i27	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i28	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i29	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	5
i30	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i31	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i32	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	5
i33	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i34	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i35	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i36	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i37	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i38	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i39	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i40	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i41	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i42	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	5
i43	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	5
i44	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i45	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i46	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i47	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i48	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	5
i49	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i50	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i51	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i52	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i53	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i54	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i55	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i56	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i57	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i58	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i59	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	10	0	0	0	5	5	5	5	0	0	10	10	5	5	5	0	10	0	0	0	0	0	5	5	5	0	0	0	10	5	5	5
i60	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	0	10	10	10	5	5	5	5	10	10	0	0	5	5	5	10	0	10	10	10	10	10	5	5	5	10	10	10	0	5	5	5
i61	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
i62	5	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5	5	5	5	0	0	0	0	5	5	5	5	0	0	0	5	5	5	5	5	5	5	0	0	0	5	5	5	5	0	0	5
ō	0	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	0

Table A.12.: A4- Travel times

	o	i1	i2	i3	i4	i5	i6	i7	i8	i9	i10	i11	i12	i13	i14	i15	i16	i17	i18	i19	i20	i21	i22	i23	i24	i25	i26	i27	i28	i29	i30	i31	i32	i33	i34	i35	i36	i37	i38	i39	i40	i41	i42	i43	i44	i45	i46	i47	i48	i49	i50	*ō*
o	0	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	0
i1	5	0	5	10	10	10	5	5	0	10	10	0	5	5	10	5	10	10	0	5	10	10	10	0	5	5	0	5	10	10	10	5	5	0	10	10	0	5	5	10	5	10	10	0	5	10	10	10	0	5	5	5
i2	5	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5
i3	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	5
i4	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	5
i5	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	5
i6	5	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5
i7	5	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5
i8	5	0	5	10	10	10	5	5	0	10	10	0	5	5	10	5	10	10	0	5	10	10	10	0	5	5	0	5	10	10	10	5	5	0	10	10	0	5	5	10	5	10	10	0	5	10	10	10	0	5	5	5
i9	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	5
i10	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	5
i11	5	0	5	10	10	10	5	5	0	10	10	0	5	5	10	5	10	10	0	5	10	10	10	0	5	5	0	5	10	10	10	5	5	0	10	10	0	5	5	10	5	10	10	0	5	10	10	10	0	5	5	5
i12	5	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5
i13	5	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5
i14	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	5
i15	5	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5
i16	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	5
i17	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	5
i18	5	0	5	10	10	10	5	5	0	10	10	0	5	5	10	5	10	10	0	5	10	10	10	0	5	5	0	5	10	10	10	5	5	0	10	10	0	5	5	10	5	10	10	0	5	10	10	10	0	5	5	5
i19	5	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5
i20	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	5
i21	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	5
i22	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	5
i23	5	0	5	10	10	10	5	5	0	10	10	0	5	5	10	5	10	10	0	5	10	10	10	0	5	5	0	5	10	10	10	5	5	0	10	10	0	5	5	10	5	10	10	0	5	10	10	10	0	5	5	5
i24	5	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5
i25	5	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5
i26	5	0	5	10	10	10	5	5	0	10	10	0	5	5	10	5	10	10	0	5	10	10	10	0	5	5	0	5	10	10	10	5	5	0	10	10	0	5	5	10	5	10	10	0	5	10	10	10	0	5	5	5
i27	5	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5
i28	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	5
i29	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	5
i30	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	5
i31	5	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5
i32	5	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5
i33	5	0	5	10	10	10	5	5	0	10	10	0	5	5	10	5	10	10	0	5	10	10	10	0	5	5	0	5	10	10	10	5	5	0	10	10	0	5	5	10	5	10	10	0	5	10	10	10	0	5	5	5
i34	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	5
i35	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	5
i36	5	0	5	10	10	10	5	5	0	10	10	0	5	5	10	5	10	10	0	5	10	10	10	0	5	5	0	5	10	10	10	5	5	0	10	10	0	5	5	10	5	10	10	0	5	10	10	10	0	5	5	5
i37	5	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5
i38	5	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5
i39	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	5
i40	5	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5
i41	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	5
i42	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	5
i43	5	0	5	10	10	10	5	5	0	10	10	0	5	5	10	5	10	10	0	5	10	10	10	0	5	5	0	5	10	10	10	5	5	0	10	10	0	5	5	10	5	10	10	0	5	10	10	10	0	5	5	5
i44	5	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5
i45	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	5
i46	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	5
i47	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	10	5	0	0	0	5	5	10	0	0	10	5	5	0	5	0	0	10	5	0	0	0	10	5	5	5
i48	5	0	5	10	10	10	5	5	0	10	10	0	5	5	10	5	10	10	0	5	10	10	10	0	5	5	0	5	10	10	10	5	5	0	10	10	0	5	5	10	5	10	10	0	5	10	10	10	0	5	5	5
i49	5	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5
i50	5	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5	0	5	5	5	0	0	5	5	5	5	0	0	5	0	5	5	5	0	5	5	5	5	0	0	5
ō	0	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	0

B. Results for realistic instances from Chapter 4

The following section presents detailed results for instances A1-A4. Tables B.1 to B.4 show, for each agent, the overall overtime (in minutes) and the assigned route (starting and ending at the depot). Tables B.5 to B.8 present the starting time and overall waiting time (in minutes) per agent per task in the final schedule.

Table B.1.: Assignments and routes for instance A1

Agent	Overtime (m)	Route
k1	0	[(o, i1), (i1, i11), (i11, $\bar{o}$)]
k2	0	[(o, i2), (i2, i10), (i10, i19), (i19, $\bar{o}$)]
k3	0	[(o, i1), (i1, i17), (i17, $\bar{o}$)]
k4	0	[(o, i3), (i3, i18), (i18, $\bar{o}$)]
k5	0	[(o, i6), (i6, i14), (i14, i23), (i23, i24), (i24, $\bar{o}$)]
k11	0	[(o, i3), (i3, i19), (i19, $\bar{o}$)]
k12	0	[(o, i5), (i5, i8), (i8, i13), (i13, $\bar{o}$)]
k13	0	[(o, i4), (i4, i9), (i9, i18), (i18, i28), (i28, $\bar{o}$)]
k14	0	[(o, i10), (i10, i16), (i16, i21), (i21, i27), (i27, i35), (i35, $\bar{o}$)]
k16	0	[(o, i7), (i7, i15), (i15, i20), (i20, $\bar{o}$)]
k17	0	[(o, i12), (i12, $\bar{o}$)]
k20	0	[(o, i12), (i12, $\bar{o}$)]
k21	0	[(o, i22), (i22, i26), (i26, i33), (i33, i49), (i49, $\bar{o}$)]
k23	0	[(o, i25), (i25, i30), (i30, i37), (i37, i45), (i45, i54), (i54, $\bar{o}$)]
k25	60	[(o, i29), (i29, i32), (i32, i42), (i42, i50), (i50, i58), (i58, $\bar{o}$)]
k27	0	[(o, i31), (i31, i46), (i46, i55), (i55, $\bar{o}$)]
k28	35	[(o, i46), (i46, i56), (i56, $\bar{o}$)]
k29	0	[(o, i35), (i35, i41), (i41, i49), (i49, $\bar{o}$)]
k30	0	[(o, i43), (i43, i48), (i48, $\bar{o}$)]
k32	80	[(o, i47), (i47, i53), (i53, i57), (i57, $\bar{o}$)]
k34	0	[(o, i36), (i36, i44), (i44, $\bar{o}$)]
k37	0	[(o, i34), (i34, i40), (i40, i52), (i52, i59), (i59, $\bar{o}$)]
k38	0	[(o, i38), (i38, i51), (i51, $\bar{o}$)]
k39	0	[(o, i52), (i52, $\bar{o}$)]
k40	105	[(o, i39), (i39, i45), (i45, i56), (i56, i61), (i61, i62), (i62, $\bar{o}$)]
k41	0	[(o, i42), (i42, i50), (i50, $\bar{o}$)]
k42	120	[(o, i60), (i60, i62), (i62, $\bar{o}$)]

Table B.2.: Assignments and routes for instance A2

Agent	Overtime (m)	Route
k1	0	[(o, i1), (i1, i3), (i3, i17), (i17, $\bar{o}$)]
k3	0	[(o, i1), (i1, i3), (i3, i16), (i16, $\bar{o}$)]
k4	0	[(o, i7), (i7, i12), (i12, i19), (i19, $\bar{o}$)]
k5	0	[(o, i9), (i9, i18), (i18, i30), (i30, $\bar{o}$)]
k11	0	[(o, i2), (i2, i8), (i8, i13), (i13, i25), (i25, i31), (i31, $\bar{o}$)]
k12	0	[(o, i4), (i4, i22), (i22, i27), (i27, $\bar{o}$)]
k13	0	[(o, i6), (i6, i10), (i10, i18), (i18, $\bar{o}$)]
k15	0	[(o, i5), (i5, i11), (i11, i21), (i21, $\bar{o}$)]
k16	0	[(o, i15), (i15, i19), (i19, $\bar{o}$)]
k17	0	[(o, i14), (i14, i20), (i20, i26), (i26, i32), (i32, $\bar{o}$)]
k18	0	[(o, i13), (i13, i35), (i35, $\bar{o}$)]
k19	0	[(o, i23), (i23, i24), (i24, i28), (i28, i38), (i38, i48), (i48, $\bar{o}$)]
k21	5	[(o, i29), (i29, i40), (i40, i45), (i45, i55), (i55, $\bar{o}$)]
k23	55	[(o, i35), (i35, i41), (i41, i47), (i47, i51), (i51, i56), (i56, $\bar{o}$)]
k30	0	[(o, i40), (i40, i44), (i44, i55), (i55, i62), (i62, $\bar{o}$)]
k31	45	[(o, i36), (i36, i46), (i46, i58), (i58, i64), (i64, $\bar{o}$)]
k32	0	[(o, i33), (i33, i39), (i39, i42), (i42, $\bar{o}$)]
k33	45	[(o, i34), (i34, i43), (i43, i48), (i48, i59), (i59, i65), (i65, $\bar{o}$)]
k34	75	[(o, i36), (i36, i49), (i49, i50), (i50, i58), (i58, i65), (i65, $\bar{o}$)]
k35	0	[(o, i37), (i37, i45), (i45, i54), (i54, i61), (i61, $\bar{o}$)]
k36	0	[(o, i42), (i42, i53), (i53, i57), (i57, i60), (i60, $\bar{o}$)]
k37	140	[(o, i50), (i50, i58), (i58, i64), (i64, i66), (i66, $\bar{o}$)]
k38	110	[(o, i52), (i52, i56), (i56, i63), (i63, i66), (i66, $\bar{o}$)]

Table B.3.: Assignments and routes for instance A3

Agent	Overtime (m)	Route
k1	0	[(o, i1), (i1,i17),(i17,$\bar{o}$)]
k2	0	[(o, i1), (i1,i11),(i11,$\bar{o}$)]
k3	0	[(o, i12), (i12,$\bar{o}$)]
k5	0	[(o, i3), (i3,i18),(i18,i26),(i26,$\bar{o}$)]
k8	0	[(o, i2), (i2,i10),(i10,i16),(i16,i21),(i21,$\bar{o}$)]
k10	0	[(o, i6), (i6,i14),(i14,i20),(i20,i28),(i28,$\bar{o}$)]
k11	0	[(o, i5), (i5,i8),(i8,i15),(i15,i25),(i25,i30),(i30,$\bar{o}$)]
k12	0	[(o, i3), (i3,i18),(i18,$\bar{o}$)]
k13	0	[(o, i4), (i4,i10),(i10,i19),(i19,$\bar{o}$)]
k14	0	[(o, i7), (i7,i13),(i13,i23),(i23,$\bar{o}$)]
k18	0	[(o, i9), (i9,i19),(i19,i35),(i35,$\bar{o}$)]
k20	0	[(o, i12), (i12,i24),(i24,$\bar{o}$)]
k28	0	[(o, i22), (i22,i27),(i27,i33),(i33,i49),(i49,$\bar{o}$)]
k31	65	[(o, i35), (i35,i41),(i41,i49),(i49,i53),(i53,i57),(i57,$\bar{o}$)]
k33	0	[(o, i29), (i29,i32),(i32,i42),(i42,i51),(i51,$\bar{o}$)]
k34	0	[(o, i31), (i31,i46),(i46,i54),(i54,$\bar{o}$)]
k37	0	[(o, i50), (i50,$\bar{o}$)]
k39	0	[(o, i45), (i45,$\bar{o}$)]
k40	0	[(o, i34), (i34,i40),(i40,i47),(i47,$\bar{o}$)]
k41	0	[(o, i37), (i37,i39),(i39,i44),(i44,i55),(i55,$\bar{o}$)]
k43	0	[(o, i36), (i36,i38),(i38,i52),(i52,i58),(i58,$\bar{o}$)]
k44	0	[(o, i43), (i43,i48),(i48,i60),(i60,$\bar{o}$)]
k45	0	[(o, i50), (i50,$\bar{o}$)]
k46	35	[(o, i46), (i46,i56),(i56,$\bar{o}$)]
k47	105	[(o, i45), (i45,i56),(i56,i61),(i61,i62),(i62,$\bar{o}$)]
k48	300	[(o, i42), (i42,i52),(i52,i59),(i59,i62),(i62,$\bar{o}$)]

Table B.4.: Assignments and routes for instance A4

Agent	Overtime (m)	Route
k1	0	[(o, i1), (i1,i2),(i2,i6),(i6,i15),(i15,$\bar{o}$)]
k2	0	[(o, i1), (i1,i8),(i8,i11),(i11,i18),(i18,$\bar{o}$)]
k3	0	[(o, i3), (i3,i10),(i10,i21),(i21,$\bar{o}$)]
k4	0	[(o, i7), (i7,i13),(i13,i19),(i19,$\bar{o}$)]
k7	0	[(o, i12), (i12,$\bar{o}$)]
k9	0	[(o, i4), (i4,i10),(i10,i17),(i17,i22),(i22,$\bar{o}$)]
k10	0	[(o, i3), (i3,i9),(i9,i16),(i16,i20),(i20,$\bar{o}$)]
k11	0	[(o, i5), (i5,i14),(i14,i15),(i15,$\bar{o}$)]
k12	0	[(o, i18), (i18,i26),(i26,i36),(i36,$\bar{o}$)]
k15	15	[(o, i36), (i36,i43),(i43,i48),(i48,$\bar{o}$)]
k18	35	[(o, i23), (i23,i26),(i26,i33),(i33,i38),(i38,i42),(i42,$\bar{o}$)]
k19	40	[(o, i24), (i24,i31),(i31,i37),(i37,i41),(i41,$\bar{o}$)]
k20	0	[(o, i28), (i28,i34),(i34,i41),(i41,$\bar{o}$)]
k21	20	[(o, i25), (i25,i31),(i31,i40),(i40,i46),(i46,$\bar{o}$)]
k22	0	[(o, i27), (i27,i32),(i32,i44),(i44,$\bar{o}$)]
k23	40	[(o, i29), (i29,i34),(i34,i42),(i42,i49),(i49,$\bar{o}$)]
k25	145	[(o, i30), (i30,i39),(i39,i47),(i47,i50),(i50,$\bar{o}$)]
k26	115	[(o, i35), (i35,i45),(i45,i50),(i50,$\bar{o}$)]

Table B.5.: Start times and waiting times for instance A1

Task	Agent	Start	Waiting (m)
i1	k1	75	0
i1	k3	75	0
i2	k2	170	0
i3	k11	180	0
i3	k4	180	0
i4	k13	180	0
i5	k12	200	0
i6	k5	210	0
i7	k16	220	0
i8	k12	230	0
i9	k13	240	0
i10	k14	250	0
i10	k2	250	0
i11	k1	290	0
i12	k17	290	0
i12	k20	290	0
i13	k12	290	0
i14	k5	300	0
i15	k16	310	0
i16	k14	310	0
i17	k3	310	0
i18	k13	360	0
i18	k4	360	0
i19	k11	390	0
i19	k2	390	0
i20	k16	400	0
i21	k14	405	0
i22	k21	415	0
i23	k5	415	0
i24	k5	495	0
i25	k23	495	0
i26	k21	505	0
i27	k14	505	0
i28	k13	525	0
i29	k25	575	0
i30	k23	585	0
i31	k27	585	0
i32	k25	630	0
i33	k21	635	0
i34	k37	655	0
i35	k14	655	0
i35	k29	655	0
i36	k34	660	0
i37	k23	680	0
i38	k38	695	0
i39	k40	725	0
i40	k37	735	0
i41	k29	760	0
i42	k25	760	0
i42	k41	760	0
i43	k30	765	0
i44	k34	770	0
i45	k23	780	0
i45	k40	780	0
i46	k27	800	0
i46	k28	800	0
i47	k32	815	0
i48	k30	825	0
i49	k21	845	0
i49	k29	845	0
i50	k25	880	0
i50	k41	880	0
i51	k38	880	0
i52	k37	885	0
i52	k39	885	0
i53	k32	915	0
i54	k23	925	0
i55	k27	930	0
i56	k28	950	0
i56	k40	950	0
i57	k32	990	0
i58	k25	1005	0
i59	k37	1015	0
i60	k42	1020	0
i61	k40	1080	0
i62	k40	1200	15
i62	k42	1200	0

Table B.6.: Start times and waiting times for instance A2

Task	Agent	Start	Waiting (m)
i1	k1	75	0
i1	k3	75	0
i2	k11	170	0
i3	k1	180	0
i3	k3	180	0
i4	k12	180	0
i5	k15	200	0
i6	k13	200	0
i7	k4	210	0
i8	k11	220	0
i9	k5	240	0
i10	k13	275	0
i11	k15	280	0
i12	k4	290	0
i13	k11	290	0
i13	k18	290	0
i14	k17	290	0
i15	k16	300	0
i16	k3	310	0
i17	k1	310	0
i18	k13	360	0
i18	k5	360	0
i19	k16	395	0
i19	k4	395	0
i20	k17	400	0
i21	k15	405	0
i22	k12	415	0
i23	k19	415	0
i24	k19	490	0
i25	k11	495	0
i26	k17	505	0
i27	k12	505	0
i28	k19	575	0
i29	k21	585	0
i30	k5	595	0
i31	k11	600	0
i32	k17	605	0
i33	k32	630	0
i34	k33	655	0
i35	k18	655	0
i35	k23	655	0
i36	k31	695	0
i36	k34	695	0
i37	k35	695	0
i38	k19	705	0
i39	k32	725	0
i40	k21	730	0
i40	k30	730	0
i41	k23	735	0
i42	k32	760	0
i42	k36	760	0
i43	k33	765	0
i44	k30	770	0
i45	k21	810	0
i45	k35	810	0
i46	k31	815	0
i47	k23	825	0
i48	k19	830	0
i48	k33	830	0
i49	k34	835	0
i50	k34	885	0
i50	k37	885	0
i51	k23	915	0
i52	k38	915	0
i53	k36	925	0
i54	k35	930	0
i55	k21	935	0
i55	k30	935	0
i56	k23	955	0
i56	k38	955	0
i57	k36	985	0
i58	k31	1005	0
i58	k34	1005	0
i58	k37	1005	0
i59	k33	1005	0
i60	k36	1015	0
i61	k35	1020	0
i62	k30	1050	0
i63	k38	1080	10
i64	k31	1105	10
i64	k37	1105	0
i65	k33	1170	45
i65	k34	1170	15
i66	k38	1190	0
i66	k37	1230	40

Table B.7.: Start times and waiting times for instance A3

Task	Agent	Start	Waiting (m)
i1	k1	75	0
i1	k2	75	0
i2	k8	170	0
i3	k12	180	0
i3	k5	180	0
i4	k13	180	0
i5	k11	200	0
i6	k10	210	0
i7	k14	220	0
i8	k11	230	0
i9	k18	240	0
i10	k13	250	0
i10	k8	250	0
i11	k2	290	0
i12	k20	290	0
i12	k3	290	0
i13	k14	290	0
i14	k10	300	0
i15	k11	310	0
i16	k8	310	0
i17	k1	310	0
i18	k12	360	0
i18	k5	360	0
i19	k13	390	0
i19	k18	390	0
i20	k10	400	0
i21	k8	405	0
i22	k28	415	0
i23	k14	415	0
i24	k20	495	0
i25	k11	495	0
i26	k5	505	0
i27	k28	505	0
i28	k10	525	0
i29	k33	575	0
i30	k11	585	0
i31	k34	585	0
i32	k33	630	0
i33	k28	635	0
i34	k40	655	0
i35	k18	655	0
i35	k31	655	0
i36	k43	660	0
i37	k41	680	0
i38	k43	695	0
i39	k41	725	0
i40	k40	735	0
i41	k31	760	0
i42	k33	760	0
i42	k48	760	0
i43	k44	765	0
i44	k41	770	0
i45	k39	780	0
i45	k47	780	0
i46	k34	800	0
i46	k46	800	0
i47	k40	815	0
i48	k44	825	0
i49	k28	845	0
i49	k31	845	0
i50	k37	880	0
i50	k45	880	0
i51	k33	880	0
i52	k43	885	0
i52	k48	885	0
i53	k31	915	0
i54	k34	925	0
i55	k41	930	0
i56	k46	950	0
i56	k47	950	0
i57	k31	990	0
i58	k43	1005	0
i59	k48	1015	0
i60	k44	1020	0
i61	k47	1080	0
i62	k47	1200	15
i62	k48	1200	0
i62	k30	1050	0
i63	k38	1080	10
i64	k31	1105	10
i64	k37	1105	0
i65	k33	1170	45
i65	k34	1170	15
i66	k38	1190	0
i66	k37	1230	40

L

Table B.8.: Start times and waiting times for instance A4

Task	Agent	Start	Waiting (m)
i1	k1	75	0
i1	k2	75	0
i2	k1	170	0
i3	k10	180	0
i3	k3	180	0
i4	k9	180	0
i5	k11	200	0
i6	k1	210	0
i7	k4	220	0
i8	k2	240	0
i9	k10	290	0
i10	k3	290	0
i10	k9	290	0
i11	k2	290	0
i12	k7	300	0
i13	k4	310	0
i14	k11	350	0
i15	k1	360	0
i15	k11	385	25
i16	k10	405	0
i17	k9	415	0
i18	k12	470	0
i18	k2	470	0
i19	k4	490	0
i20	k10	495	0
i21	k3	505	0
i22	k9	575	0
i23	k18	575	0
i24	k19	600	0
i25	k21	655	0
i26	k12	655	0
i26	k18	655	0
i27	k22	685	0
i28	k20	700	0
i29	k23	725	0
i30	k25	735	0
i31	k19	760	0
i31	k21	760	0
i32	k22	815	0
i33	k18	820	0
i34	k20	820	0
i34	k23	820	0
i35	k26	825	0
i36	k12	845	0
i36	k15	845	0
i37	k19	895	0
i38	k18	915	0
i39	k25	925	0
i40	k21	930	0
i41	k19	940	0
i41	k20	940	0
i42	k18	950	0
i42	k23	950	0
i43	k15	970	0
i44	k22	975	0
i45	k26	1005	0
i46	k21	1005	0
i47	k25	1015	0
i48	k15	1020	0
i49	k23	1060	5
i50	k25	1110	20
i50	k26	1100	10

C. Results for instances with different shift length from Chapter 4

Table C.1.: Results with $\sigma = 0.7$

Instance	MIP			BP			
	Objective	Abs. Gap (%)	CPU (m)	Objective	CPU (m)	Nodes	Columns
1	243.60	0.00	60.00	243.60	3.65	34	1083
2	343.50	5.56	60.00	325.40	5.11	87	6608
3	INFEASIBLE						
4	346.20	0.00	13.91	346.20	1.62	75	988
5	291.60	0.00	52.44	291.60	4.98	83	4986
6	261.90	0.00	10.46	261.90	1.38	30	937
7	INFEASIBLE						
8	381.30	0.00	24.13	381.30	2.97	40	1093
9	375.90	0.00	12.03	375.90	5.22	80	8343
10	530.90	0.00	59.08	530.90	12.89	81	2211
11	334.20	0.00	2.78	334.20	1.20	47	815
12	354.90	0.00	19.11	354.90	3.63	56	1468
13	318.20	0.00	3.56	318.20	3.04	107	10540
14	182.60	0.00	46.41	182.60	2.80	34	1227
15	423.90	0.00	60.00	423.90	6.71	109	12946
16	255.50	0.00	18.04	255.50	5.52	58	6711
17	378.40	0.00	7.69	378.40	1.41	46	907
18	350.40	0.00	19.29	350.40	3.52	48	1227
19	458.40	0.00	33.74	458.40	6.53	88	14358
20	552.40	0.00	31.62	552.40	10.45	67	2048
21	345.60	0.00	3.17	345.60	1.51	63	936
22	369.90	0.00	26.14	369.90	5.03	81	1661
23	440.00	0.00	31.49	440.00	3.06	43	2306
24	436.10	0.00	18.34	436.10	7.21	87	17557
Average	362.52	0.25	27.88	361.70	4.52	65.64	4588.91

Table C.2.: Results with $\sigma = 0.8$

Instance	MIP			BP			
	Objective	Abs. Gap (%)	CPU (m)	Objective	CPU (m)	Nodes	Columns
1	229.60	5.81	60.00	217.00	7.65	95	12935
2	312.00	11.95	60.00	278.70	6.78	107	12687
3	220.30	0.00	52.30	220.30	13.93	80	9386
4	282.50	0.00	10.98	282.50	1.51	52	850
5	196.20	0.00	53.05	196.20	3.43	57	1251
6	193.80	0.00	11.74	193.80	1.81	82	1055
7	137.50	0.00	55.30	137.50	4.18	59	1436
8	300.00	0.00	25.11	300.00	3.64	66	1107
9	308.70	0.00	11.51	308.70	3.08	60	1236
10	368.00	0.00	56.90	368.00	16.62	80	8363
11	294.30	0.00	3.88	294.30	2.05	81	2828
12	262.20	0.00	19.04	262.20	5.00	91	1689
13	254.60	0.00	4.74	254.60	1.12	40	832
14	161.20	0.00	45.29	161.20	3.46	50	1226
15	363.70	0.00	60.00	363.70	6.10	81	10071
16	189.70	0.00	19.71	189.70	3.89	77	1344
17	231.90	0.00	6.38	231.90	1.48	59	812
18	283.60	0.00	24.29	283.60	7.07	104	13845
19	399.40	0.00	34.32	399.40	2.89	86	1098
20	361.30	0.00	31.29	361.30	13.64	75	13005
21	266.20	0.00	3.71	266.20	1.20	40	806
22	270.20	0.00	25.16	270.20	4.93	78	2918
23	365.30	0.00	33.95	365.30	3.10	32	1151
24	327.40	0.00	19.39	327.40	2.84	51	1250
Average	274.15	0.74	30.34	272.24	5.06	70.13	4299.21

Table C.3.: Results with $\sigma = 1.2$

Instance	MIP			BP			
	Objective	Abs. Gap (%)	CPU (m)	Objective	CPU (m)	Nodes	Columns
1	NA	NA	60.00	30.40	8.26	100	23154
2	NA	NA	60.00	21.80	8.11	112	25370
3	NA	NA	60.00	27.00	17.16	102	24274
4	19.60	0.00	11.65	19.60	2.01	62	7656
5	NA	NA	60.00	24.20	7.33	105	20025
6	17.40	0.00	13.21	17.40	1.36	40	3538
7	NA	NA	60.00	24.20	5.01	72	6657
8	30.40	0.00	23.71	30.40	2.12	35	863
9	22.00	0.00	12.34	22.00	5.49	94	13095
10	27.00	0.00	57.62	27.00	8.29	50	2324
11	19.20	0.00	4.40	19.20	2.24	66	5503
12	24.20	0.00	20.14	24.20	2.84	53	968
13	17.40	0.00	2.04	17.40	1.03	25	1242
14	23.40	0.00	43.86	23.40	7.77	98	17196
15	31.45	3.45	60.00	30.40	6.38	72	18861
16	20.80	0.00	17.81	20.80	2.27	25	827
17	17.40	0.00	7.36	17.40	1.92	47	5989
18	23.40	0.00	20.72	23.40	2.24	54	1822
19	30.40	0.00	36.39	30.40	2.43	38	917
20	27.60	0.00	32.24	27.60	6.22	42	1131
21	17.40	0.00	1.04	17.40	0.73	18	599
22	24.00	0.00	26.47	24.00	9.35	135	30662
23	30.80	0.00	32.19	30.80	4.35	50	11384
24	20.80	0.00	16.39	20.80	6.30	87	12425
Average	23.40	0.18	30.82	23.80	5.05	65.92	9853.42

Table C.4.: Results with $\sigma = 1.3$

Instance	MIP			BP			
	Objective	Abs. Gap (%)	CPU (m)	Objective	CPU (m)	Nodes	Columns
1	NA	NA	60.00	30.40	6.47	95	14929
2	NA	NA	60.00	21.80	2.23	37	1755
3	NA	NA	60.00	27.00	13.66	90	12759
4	19.60	0.00	8.15	19.60	1.13	36	1286
5	NA	NA	60.00	24.00	4.80	63	12236
6	17.40	0.00	12.34	17.40	0.94	22	629
7	NA	NA	60.00	23.80	5.91	93	14362
8	30.40	0.00	24.58	30.40	4.98	74	13919
9	21.60	0.00	12.23	21.60	6.75	115	19587
10	27.00	0.00	57.67	27.00	15.05	77	16783
11	19.20	0.00	2.55	19.20	2.32	57	8222
12	23.60	0.00	21.06	23.60	4.65	66	9403
13	17.40	0.00	5.59	17.40	1.46	37	4038
14	23.40	0.00	45.27	23.40	6.32	81	16986
15	30.40	0.00	58.83	30.40	2.38	49	901
16	20.80	0.00	20.18	20.80	3.45	40	3557
17	17.40	0.00	2.73	17.40	1.01	29	679
18	23.40	0.00	22.05	23.40	2.85	47	3579
19	30.40	0.00	33.91	30.40	6.08	75	17411
20	27.60	0.00	30.79	27.60	14.58	83	22484
21	17.40	0.00	3.27	17.40	0.86	26	684
22	24.00	0.00	24.57	24.00	5.50	79	16757
23	30.80	0.00	30.94	30.80	4.06	51	11215
24	20.20	0.00	17.63	20.20	7.07	112	19545
Average	23.26	0.00	30.60	23.71	5.19	63.92	10154.42

Curriculum Vitae

Emilio Zamorano de Acha

Personal Information

Place of birth	Guadalajara, Mexico
Nationality	German - Mexican

Education

2010—2016	Doctoral Candidate
	Business School, University of Mannheim
2008—2009	Master of Engineering (M. I.)
	Universidad Panamericana, Mexico
2003—2007	Industrial Engineering (B.Sc.)
	Universidad Panamericana, Mexico

Professional Experience

2011—2015	Research Assistant, Chair of Production Management,
	Business School, University of Mannheim
2008—2010	Industrial Engineering Program Coordinator
	Universidad Panamericana, Guadalajara, Mexico
2007—2008	Material Planning and Logistics Trainee
	Ford Motor Co., Almussafes, Spain