The Resource Assignment Problem

Luuk Gerrits

The Resource Assignment Problem

Luuk Gerrits MSc (323320)

Erasmus School of Economics

Master's programme:
Econometrics & Management Science

Specialization:
Operations Research & Quantitative Logistics

Supervisor:	*Co-reader:*	*Company supervisor:*
Dr. D. Huisman	Dr. W. van den Heuvel	Drs. A. Rietveld

16 May 2011

Abstract

This Master's thesis deals with the Resource Assignment Problem (RAP). This problem is not yet treated in literature. It is encountered in ORTEC TD, a software package for transportation planning. The RAP involves the allocation of a set of transportation trips to a number of available resources, mainly trucks, drivers and trailers. The objective is to minimize total costs, while complying with several constraints, such as driving time legislation and time windows.

For this problem, the trip assignment method is developed by ORTEC, based on column generation. Furthermore, we developed a more extensive version of this method, which is capable of assigning the individual sections of which the trips consists. We therefore refer to this method as the section assignment method.

The implementation of the trip assignment method in ORTEC TD is tested using a case from practice, proving its capability to produce usable results within acceptable time. To compare this method with the section assignment method, both methods are also implemented in MATLAB and applied to instances of different size. This reveals that in most cases, the more complicated section assignment method can be used to obtain a better solution value, especially in small- and medium-sized instances. However, computation times for this method are significantly higher. It is therefore concluded that the section assignment method is the most useful when the problem size is not too large, or available time is ample. For very large instances as well as for situations in which a rough estimate is needed within a limited amount of time, the less complicated trip assignment method is the better choice.

Acknowledgements

Several people are acknowledged for their contribution to this research. First, I am thankful to Dennis Huisman for his advice and supervision. Also, I would like to thank Arjen Rietveld from ORTEC, for providing support as well as good company during the past months. Also from ORTEC, Joaquim Gromicho and Leendert Kok are acknowledged for their help and valuable input. Finally, I would like to express my gratitude to all others who have contributed, directly or indirectly, to the realization of this thesis.

Luuk Gerrits
Amsterdam, 16 May 2011

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This thesis deals with the Resource Assignment Problem. It is conducted for ORTEC, which will be described briefly in the next section. After that, the problem statement is introduced and the relevance of this problem is discussed. Finally, the outline of this thesis is given.

1.1 Introduction to ORTEC

The research presented in this thesis is performed under the authority of ORTEC. ORTEC is a Dutch company specialized in advanced planning and optimization software solutions. It was founded in 1981 and has grown into an international organization with over 700 employees. It delivers software products as well as consulting services in various fields, such as vehicle routing, workforce scheduling, and production planning. This thesis is primarily written for the ORTEC Product Management (OPM) department, which is responsible for all tasks related to the various standard products in ORTEC's portfolio.

1.2 Introduction to ORTEC TD

ORTEC Transport & Distribution (ORTEC TD) is a software package that supports organizations in planning and fulfilling transport and distribution orders. ORTEC TD is used by transportation companies, as well as other companies that deal with large and complex distribution processes. Among ORTEC's customers for this package are Simon Loos, TNT, Nabuurs and DHL.

The essence of ORTEC TD is coping with the Vehicle Routing Problem (VRP), which is a combinatorial optimization and integer programming problem seeking to service a number of customers with a given fleet of vehicles. Therefore, ORTEC TD mainly supports the operational part of the logistical process, as it is used to determine when orders are fulfilled, and by which resources (e.g. driver, truck and trailer). Furthermore, the system can be used for real-time purposes as well, using functionalities such as GPS-tracking and communication with on-board devices.

Besides supporting the manual planning process, ORTEC TD also includes an optimization tool, which is able to automatically construct a transportation plan. Often, the package is integrated with the client's Enterprise Resource Planning package and Transport Management System.

1.3 Introduction to the research problem

In ORTEC TD, a number of transportation orders are combined into several transportation trips. Such a trip is basically a route along several locations where loads are picked up or dropped off. Each of these trips needs to be carried out by a number of resources, in most cases a driver, a truck and a trailer. The combination of these trips and resources can be a complex problem in practice. The aim of the research presented in this thesis is to provide a procedure to deal with this problem. The research is split up into two parts, to match the two-phase development process at ORTEC. The contribution of this research to the development process is different, as will be explained in chapter 2. This chapter will also provide a far more elaborate explanation of the research problem.

1.3.1 Problem statement

From the research problem described above, the following problem statement is deducted:

What is the best way to assign a given set of resources (drivers, trucks, trailers) to a set of planned transportation trips, in terms of costs and calculation time?

Research questions

In order to deal with the above problem statement, the following research questions will be answered:

- In literature on combinatorial optimization, how is dealt with problems comparable to the problem studied here?
- How can ORTEC's method for the first development phase be described?
- How can the approach applied by ORTEC be extended in order to deal with the problem faced in the second development phase?
- How do both methods compare with each other in terms of solution quality and computation time?
- What recommendations towards ORTEC can be extracted from the results?

1.3.2 Relevance

The relevance of this research is in the first place found in practice, as several (potential) ORTEC TD customers have already asked for functionality to automatically assign trips to available resources. There are several reasons for ORTEC TD users to demand such a functionality. First, it is very likely that an automatic assignment procedure will be able to construct a plan much faster than a human planner. Therefore, the desired functionality allows the user to cut labor costs. Second, it is expected that the functionality will result in a better solution compared to a planner, where better usually means less expensive. Furthermore, it allows the user to easily compare different planning scenarios.

Because of the practical relevance, this research mainly aims at providing a solution procedure for the problem exactly as encountered within ORTEC TD. However, it is concluded in chapter 3 that this problem has not yet been dealt with in literature. Therefore, there is a theoretical relevance as well.

1.3.3 Approach

This thesis will continue with an extensive description of the research problem in chapter 2. In this chapter the used terminology is explained and a number of scenarios is described to exemplify the problem. Furthermore, the optimization criteria and relevant restrictions are discussed.

Chapter 3 is dedicated to literature on problems related to the resource assignment problem. It mainly discusses articles on (different versions of) some well-known problems in the field of combinatorial optimization, as well as their applicability to the problem faced in this thesis.

Chapter 4 describes the method applied by ORTEC to solve the first version of the problem. In the subsequent chapter, a small improvement to this method is explained. Chapter 6 describes how another method is developed in order to deal with the second, more complex version of the resource assignment problem.

Chapter 7 describes the application of the first method to real-life cases in OTD. In chapter 8, this method is compared to the second method by applying them both to randomly generated cases.

Finally, chapter 9 elaborates on the conclusions that are drawn from the research conducted in this thesis.

Chapter 2

Problem Description

2.1 The resource assignment problem

The purpose of a transportation or distribution plan is to determine the routing of resources, such that a given set of orders can be fulfilled. The optimization tool within ORTEC TD is capable of constructing a batch plan out of a given set of resources and orders. However, it is experienced that this does not entirely match the actual planning process as carried out by some of ORTEC's clients, mostly because the available resources are not yet known to the planner at the time the routes are planned. For these clients, the planning process is divided into two stages. First, the transport orders are combined into what we will refer to as *execution trips*. This can be done either manually or using an optimization tool. In this stage, for each trip either a trailer type or the actual trailer to be used is determined. Each execution trip consists of one or multiple *sections*. In the second stage, the planner assigns each section to the required resources, which in most cases are a driver, a truck and, if not already assigned, a trailer. In ORTEC TD, this part of the planning process cannot be done automatically yet. Therefore, this research addresses the problem of assigning planned sections to resources.

From the above, the following description is deducted: The resource assignment problem (RAP) is the problem of matching a set of sections, which are part of a number of already planned execution trips, with a given set of resources, such that total costs are minimized. In this assignment, driving and working time legislation, compatibility between sections and resources, driving times and time windows have to be respected. Furthermore, all given sections have to be assigned. In the remainder of this chapter, the concepts and restrictions mentioned here will be explained.

2.2 Terminology

2.2.1 Resource shifts

Obviously, not every resource is continuously available. For instance, it can be agreed on by the driver and his employer that the driver and his truck are available for work every week from Monday morning to Friday evening, and that he will start from and end at his home base. To incorporate these agreements, the concept of *resource shifts* will be used. A resource shift is a period in which a resource or a combination of resources is available, together with a given start location, and possibly a fixed end location as well. Furthermore, when driving time legislation is taken into account, the driver status at the start of the resource shift should be given. This is explained in section 2.5.1.

2.2.2 Execution trips

An execution trip is basically a combination of orders, and therefore consists of a series of tasks, such as pickups and deliveries. Only one trailer can be used for the entire trip. This means that when different trailers need to be assigned to parts (sections) of the execution trip, the trip needs to be split by the planner. Two important properties of an execution trip are its start and end location, which can both either be at a depot or at another address. Furthermore, from the time windows of the orders in the execution trip, time windows for the start and end of the trip can be derived. However, the time windows of the tasks within a trip have to be maintained as well, as the length of an execution trip can change as a result of the scheduling of breaks.

2.2.3 Sections

As explained earlier, the planned execution trips have to be assigned to resource shifts. However, for some trips it might not be required that the used resources remain the same during the entire trip. For instance, often the driver and truck do not have to be present during the loading of the trailer. Also, it might for some reason be advantageous to switch both the driver and truck halfway the trip. To allow for these situations, the execution trips will be divided into sections. In this respect, a section can be defined as a part of an execution trip, starting and ending at a point at which the resources assigned to that part can be changed. This point could be the begin or end of the execution trip. The planning process is visualized in Figure 2.1. In this figure, the step from level three to level four is the resource assignment problem.

The reason for using both execution trips and sections is that an execution trip can maintain the dependency between two consecutive sections. In some cases, a particular section cannot start before the preceding section, such as a preloading part, is completed. In such a case, both sections have to be part of the same execution trip. This dependency is visualized in Figure A.1, which can be found in Appendix A.

Another important remark to be made is that the duration of a section is not necessarily fixed. It may vary because of driving and working time legislation, as explained in section 2.5.1. Whether rest periods have to be scheduled in a trip, and if so, where, depends on the amount of driving time and rest the driver had in previous sections. An example of this dependency is given in Figure A.2.

2.2.4 Trailer types and specific trailers

An execution trip and its underlying sections are planned based on several trailer characteristics, such as capacities in terms of space and weight, and the type of goods it can transport (cooled products, fluids, etc.). However, prior to the resource optimization it might not always be known exactly which trailers will be used for the planned execution trip. Therefore, the planner can either use a specific trailer, which refers to an actual trailer, or a trailer type, which refers to a particular kind of trailer. In the latter case, the trailer type can be replaced by a specific trailer of that type, either during or after the resource optimization.

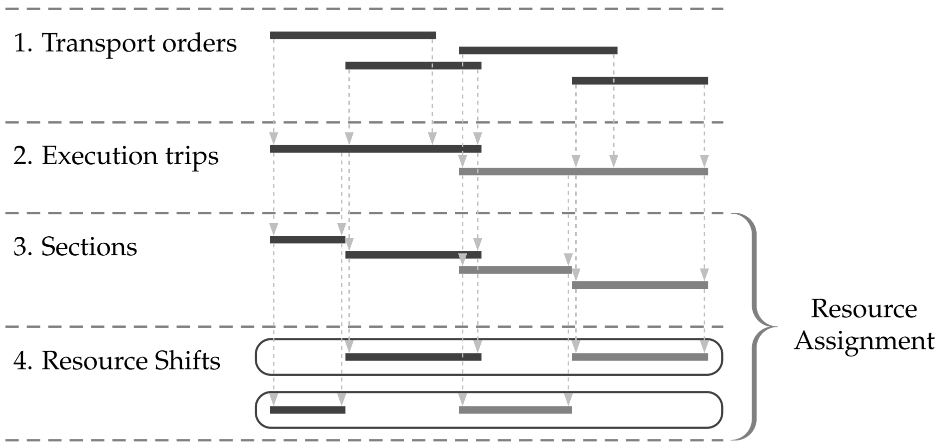


Figure 2.1: The planning process

2.3 Planning scenarios

In general, the following scenarios for the application of the resource assignment functionality can be distinguished:

- Basic functionality
- Intermodal transportation
- Preloading of trailers

These scenarios are explained below.

2.3.1 Basic functionality

The main functionality needed the enable resource shift optimization, is the possibility to assign sections to resource shifts. For this purpose, three subscenarios can be defined:

- The assignment of a section with a specific trailer to a resource shift with driver and truck.
- The assignment of a section with trailer type to a resource shift with driver, truck and trailer. The trailer type is replaced by the specific trailer in the resource shift.
- The assignment of a section with trailer type to a resource shift with driver and truck. The trailer type is maintained during the resource assignment, and only afterwards manually replaced by a specific trailer.

For visualizations of the above subscenarios, see parts a, b and c of Figure B.1 respectively, available in Appendix B.

2.3.2 Intermodal transportation

In intermodal transportation, multiple modes of transportation are used within an execution trip. In the case of ORTEC's clients, this involves transportation by train or by ferry, both preceded and followed by road transport. For this purpose, the train or ferry part will be in a separate section, such that the resources used can differ from the road sections in the execution trip. With respect to intermodal transportation, four subscenarios can be distinguished:

- Accompanied intermodal section with trailer type: The section is assigned to a resource shift with driver, truck and trailer. The trailer type is replaced by the specific trailer in the resources shift.
- Accompanied intermodal section with specific trailer: The specific trailer is added to a resource shift with driver and truck.
- Unaccompanied intermodal section with trailer type: The section is not assigned to any resource shift during optimization. Afterwards, the trailer type is replaced manually by a specific trailer.
- Unaccompanied intermodal section with specific trailer: The section is not assigned to any resource shift.

The four above subscenarios are visualized in parts a, b, c and d of Figure B.2 respectively.

2.3.3 Preloading of trailers

Another important scenario that should be supported is the preloading of trailers. Especially in distribution environments, loading a trailer at the warehouse might take a considerable amount of time. In most cases, the driver and truck need not be present until the trailer is loaded and the actual distribution round can be started. Therefore, this should be accounted for in the planning, by accommodating the preloading part in an individual section. It should be possible to assign either a trailer type or a specific trailer to the preloading section, prior to the optimization. In this manner, it is supported that the planner decides on the actual trailer to use and communicates this to the warehouse, or that the warehouse decides this while executing the planning and then communicates it to the planner. Both subscenarios are displayed in Figure B.3 in Appendix B.

The opposite of preloading should be facilitated in a similar way. This means that a trailer can be unloaded without the presence of a driver and a truck.

2.4 Optimization

2.4.1 Objective function

The most obvious objective of the optimization would be minimizing overall costs. These costs involve the driving cost per kilometer, probably depending on the resource used, and a fixed cost for each resource shift used. However, the optimization should also be capable of dealing with situations in which a mismatch exists between the number of sections that can be covered by the available resources and the actual number of sections.

The current optimization tool present in ORTEC TD is capable of handling a certain order of optimization objectives. This can, for instance, be used to minimize costs first, then minimize the total distance traveled, and finally minimize the number of resources used. It would be desirable to have such an objective function for the resource shift optimization as well. However, whether this is possible will depend on the method used.

2.4.2 Usage

The planner working with ORTEC TD will select a set of resources and a set of sections, which will be used as input for the optimization run. It is estimated that the number of sections covered by a particular resource shift is not likely to exceed three. The use of the optimization tool will be in two ways, roughly. First, the optimization can be used overnight, to solve an instance with many resources and sections available. In this case, much computational time is available. The second application is during the execution of a planning, in which the tool will be used to optimize the resource assignment for a small number of sections. In this case, a solution should be found rather quickly.

2.5 Restrictions

2.5.1 Driving time and working time legislation

In transportation planning, driving time and working time legislation play an important role. Transportation companies violating these regulations risk very high fines. In this research, the European legislation will be used, as it applies to most of ORTEC TD's clients. Two regulations have to be taken into account. Regulation (EC) No 561/2006 (European Union, 2006) establishes the maximum amount of time a driver is allowed to be on the road; directive 2002/15/EC (European Union, 2002) restricts the weekly working time. Next, the relevant restrictions from these regulations are stated.

Weekly rest

Within 144 hours from the end of the previous weekly rest period, a new weekly rest period of at least 45 hours should be started. The duration of this period may be reduced to at least 24 hours, provided that before the end of the third calendar week after this rest, a rest period equal to the reduction is added to another rest period of at least nine hours. Any two consecutive calendar weeks should contain at least two weekly rest periods, one of which should be of at least 45 hours. If a weekly rest period falls in two calendar weeks, it can be counted in only one of them. In these restrictions, a calendar week starts on Monday 0:00 hours.

Daily rest

Within 24 hours after the end of each weekly or daily rest period, a new rest period of at least 11 hours should be completed. Between two weekly rest periods, this daily rest period may be reduced to nine hours for at most three times. Furthermore, a daily rest period of at least 11 hours may be replaced by two consecutive parts of at least three and nine hours, respectively.

Accumulated driving and working time

The accumulated weekly driving time is limited by 56 hours. However, the average weekly driving time in any two consecutive weeks should not exceed 45 hours. The maximum daily driving time is nine hours, which may be extended to ten hours twice a week.

Working time is defined as the total time devoted to all road transport activities (European Union, 2002). This means that besides driving time, also activities such as loading, unloading, cleaning, etc. count as working time. Total working time during any calendar week should not exceed 60 hours. Over a period of four months, average weekly working time should not be more than 48 hours.

Breaks

After driving continuously for at most four and a half hours, a driver must take either a rest period or a break. The duration of a break between two driving periods should be at least 45 minutes, or 30 minutes if the preceding driving period is interrupted by a break of at least 15 minutes. During a break, the driver cannot drive or carry out other work. If the total amount of working hours on a day is between six and nine hours, working time should be interrupted by a break of at least 30 minutes. If total working time exceeds nine hours, a break of at least 45 minutes should be maintained. These breaks may be subdivided into periods of at least 15 minutes.

2.5.2 Compatibility

In the optimization run, compatibility between resources must be taken into account. In this case, this is relevant when a section with a specific trailer or trailer type is assigned to a resource shift with a truck and a driver. Such an assignment should only be implemented when the truck is allowed to pull that particular trailer or trailer type. Furthermore, replacement of a trailer type by a specific trailer should only be allowed if the specific trailer is of the specified type.

2.5.3 Driving between sections

If multiple sections are assigned to a particular resource shift, it must be taken into account that the start location of a particular section might be different from the end location of the preceding section. Furthermore, the resources might have to drive from the start location of the resource shift to the start location of the first section. The same holds for the end location of the last section and the end location of the resource shift. Therefore, the time between each pair of consecutive locations in a resource shift must be equal to or larger than the time needed to drive from the one location to the other.

2.5.4 Time windows

The pickup and delivery times of most orders are restricted by time windows. This means that a particular action cannot take place before a certain point in time, and should also be completed before a given time.

2.5.5 Coverage of sections

In order to assure that all orders are fulfilled, all sections that are part of the input for the optimization run should be assigned to the required resources. In practice however, the optimization method may also be applied to situations in which not all sections can be covered. In such a situation, it might be desired to replace this restriction by an addition to the objective function, for instance maximizing the number of sections covered or minimizing the number of subcontractors needed.

Chapter 3

Literature Review

This thesis deals with the problem of assigning planned trips to a set of available resources. This problem has several similarities with a number of other problems known in the literature on combinatorial optimization. These are (i) the vehicle routing problem, (ii) the vehicle scheduling problem and (iii) the crew scheduling problem. In this chapter, various approaches for these problems are discussed based on the literature. Further, as driving time legislation plays an important role in the problem considered in this thesis, literature on such a regulation is discussed as well. Finally, the relevance of the various problems and approaches for the resource assignment problem are discussed.

3.1 The vehicle routing problem

The vehicle routing problem (VRP) addresses the problem of assigning a number of locations to be visited to vehicles, such that the demand for each location is satisfied and the total distance traveled is minimized (Dantzig & Ramser, 1959).

Li and Lim (2003) propose a tabu-embedded simulated annealing heuristic for solving the vehicle routing problem with time windows (VRPTW). The authors stress that this method could also be applied to variants of this problem and to other combinatorial problems, meaning it is likely to be relevant to the problem considered in this thesis as well. The computational results show that the solutions are comparable with other approaches, and computational time is reasonable.

Xu, Chen, Rajagopal, and Arunapuram (2003) consider a vehicle routing problem with pickups and deliveries. Their approach is practice-oriented, and therefore they take into account a number of real-world restrictions, namely time windows, compatibility between orders and carrier and vehicle types, the loading and unloading sequence and the driving time rules as applying in the United States. Because of these realistic restrictions, the problem has similarities to the resource assignment problem. The set partitioning formulation of the problem is solved by column generation, using fast heuristics for solving the subproblems. The computational results show that instances with up to 500 orders can be solved within acceptable time. Furthermore, it is shown that for instances with up to 10 orders, the obtained solution value is never more than four percent above the lower bound obtained by dynamical programming.

Zäpfel and Bögl (2008) consider an application of the VRP in a case-study, in which pickup an delivery tours have to be scheduled for a postal company. Adding to the complexity of the problem are the involvement of time windows, vehicle capacities and personal planning issues. They stress that because of the large number of variables and constraints, only heuristic approaches are relevant in practice, and conclude that Tabu Search proves to be the most efficient approach for the considered problem.

Gromicho, Van Hoorn, Kok, and Schutten (2009) use restricted dynamic programming to solve the VRP, embedded within a framework flexible enough to deal with a variety of restrictions. This approach is worth mentioning here, as it is used within ORTEC TD to combine a set of transportation orders into vehicle routes.

3.2 The vehicle scheduling problem

In the vehicle scheduling problem (VSP), trips have to be assigned to vehicles. The main difference with the VRP as discussed above is the fact that instead of visiting a location, a trip has to be executed, involving a certain duration and change in location. According to Bodin, Golden, Assad, and Ball (1983), the essence of the VSP is 'the sequencing of vehicle activities in both space and time'. Occurrences of this problem can be found in freight transportation and public transport. In the literature, often two versions are distinguished: the single-depot vehicle scheduling problem (SDVSP) and the multi-depot vehicle version (MDVSP).

Bunte and Kliewer (2010) provide an overview of various modeling approaches presented in the literature on the VSP. They consider both the single-depot and the multi-depot case, as well as the most common practical extensions, i.e. multiple vehicle types, time windows and route constraints.

3.2.1 The single-depot VSP

The SDVSP can be defined as the problem in which a given set of trips with fixed start an end times has to be scheduled such that every trip is assigned to a vehicle, while minimizing costs. Each vehicle schedule should contain one or more blocks, in which the vehicle departs from the depot, executes a sequence of trips, and then returns to the depot. In this version of the problem, only one depot exists, and all vehicles are identical (Freling, Wagelmans, & Paixão, 2001). This problem can be solved in polynomial time, and several efficient algorithms are available (Carraresi & Gallo, 1984). However, when the assumption of identical vehicles is dropped, the problem is known as the vehicle scheduling problem with multiple vehicle types (VSPMVT), which is NP-hard (Bodin et al., 1983).

Baita, Pesenti, Ukovich, and Favaretto (2000) state that although efficient algorithms exist for the theoretical VSP, dealing with the complex, real-life problems is much harder. They compare the effectiveness of a genetic algorithm, a logic programming based heuristic and an algorithm based on the assignment problem formulation. The latter proves to be efficient in practice, as it is capable of dealing with very large instances. This also applies to the the logic programming based heuristic. However, this method is expected to be less applicable in most real-life cases, as it requires very high programming skills.

3.2.2 The multi-depot VSP

The VSP with multiple depots is proven to be NP-hard (Bertossi, Carraresi, & Gallo, 1987). Several approaches are suggested in literature. Ribeiro and Soumis (1994) formulate the MDVSP as a set partitioning problem with side constraints, whose linear relaxation is then solved by column generation. They show that this approach produces better lower bounds compared to the previously applied additive bound procedure, and that it is capable of dealing with large problems.

A relevant addition to the MDVSP is the existence of time windows for the trips to be executed. This problem, the MDVSPTW, is addressed by Desaulniers, Lavigne, and Soumis (1998), using a column generation approach within a branch-and-bound framework. Optimal solutions are found for small and medium instances of the problem, and heuristic solutions for larger ones, all within acceptable computational time.

Oukil, Amor, Desrosiers, and Gueddari (2007) emphasize the effectiveness of column generation for solving the linear relaxations of large MDVSP instances. However, they also stress potential problems caused by degeneracy (a phenomenon that may for example cause the simplex method to stop), and present a stabilized column generation approach that is capable of dealing with degenerate instances, and outperforms standard column generation in terms of computational time. Furthermore, their research differs from previous work in the sense that it considers exact waiting costs between time windows instead of minimal waiting time.

In addition, Qureshi, Taniguchi, and Yamada (2009) address a version of the vehicle routing and scheduling problem in which semi soft time windows are considered, meaning that a penalty is accounted for arrivals later than the time window, the height of which depends on the lateness. Early arrivals are not penalized. This problem is solved exactly using column generation. However, the approach requires a considerable amount of time when large instances are considered, and is therefore only recommended for smaller instances.

Laurent and Hao (2009) test the application of an iterated local search algorithm to the MDVSP. They conclude that this approach outperforms other metaheuristics in terms of both computation time and solution quality, and that it can be a good alternative for column generation. Furthermore, they present an auction algorithm capable of quickly constructing an initial solution of good quality, which might also be useful in other approaches.

Haghani and Banihashemi (2002) present one exact and two heuristic solutions for solving the MDVSP with route time constraints. These constraints reduce the size of the problem with around 40%. However, their methods prove only capable of solving small and medium-sized instances within acceptable time, and therefore two techniques for decreasing the size of large problems are proposed as well.

In most of the literature discussed in this section, column generation is used to solve the MDVSP. Pepin, Desaulniers, Hertz, and Huisman (2009) compare five heuristic approaches for this problem, and confirm that column generation produces the best solutions and has the highest stability. However, the available computational time needs to be sufficient. Otherwise, a large neighborhood search is the best alternative in terms of solution quality and computational time.

3.3 The crew scheduling problem

The crew scheduling problem (CSP) essentially involves the assignment of a given set of tasks to duties, which can then be carried out by the available crew (Freling, Wagelmans, & Paixão, 1999). In its most basic form, the CSP can be formulated as either a set partitioning problem, in which each task is assigned to exactly one crew, or as a set covering problem, in which each task is covered by at least one crew (Huisman, 2004). An important issue in crew scheduling are various kinds of labor rules, such as the driving time legislation discussed in Sections 2.5.1 and 3.5. Fischetti, Martello, and Toth (1987) proved that if the crew is only available for a limited period of time, i.e. the CSP with spread time constraints, the problem is NP-hard. The same applies to the CSP with limitations on the total working time of each crew (Fischetti, Martello, & Toth, 1989).

One of the most relevant application fields for the CSP is the airline industry, as crew costs can have significant impact on an airliner's profitability. Borndörfer, Schelten, Schlechte, and Weider (2005) propose a column generation approach for an airline crew scheduling problem, based on a set partitioning model. This method proves capable of dealing with large and complex instances, but in its application, it was given an overall time limit of two days.

3.4 The vehicle and crew scheduling problem

The problem dealt with in this thesis is in many ways similar to the vehicle and crew scheduling problem (VCSP). This problem can be considered a combination of two subproblems, viz. the assignment of a given set of trips to vehicles and the assignment of these trips to the available crew. Methods for solving the VCSP can be classified as either a sequential approach, in which one of the subproblems is solved before the other, or an integrated approach.

3.4.1 The sequential approach

In the traditional, sequential approach to the VCSP, the vehicle scheduling problem is solved first, independent of crew scheduling issues (Freling, Huisman, & Wagelmans, 2003). Next, the crew scheduling problem is solved, considering the duties resulting from the solution to the VSP as fixed. In fact, the sequential approach to the VSP involves solving two separate problems, without any interaction.

3.4.2 The integrated approach

In recent literature on vehicle routing and crew scheduling, most authors strive towards an integrated approach (Freling et al., 2003; De Groot & Huisman, 2008; Mercier, Cordeau, & Soumisa, 2005; Papadakos, 2009). In such an approach, the routing of vehicles and the assignment of crew to these vehicles is regarded as one problem, often referred to as the (simultaneous) vehicle and crew scheduling problem. The main idea behind the integration of both problems is that solving them separately is likely to result in a suboptimal overall solution. Huisman, Freling, and Wagelmans (2005) propose two algorithms, both based on column generation and Lagrangian relaxation, for an integrated approach to multiple-depot vehicle and crew scheduling in public transport. Their results confirm that integrating both problems can significantly improve the overall solution. De Groot and Huisman (2008) explore various methods for splitting large, real-world vehicle and crew scheduling problems into smaller problems that can be solved by an integrated approach. Besides proving the effectiveness of these methods, they reconfirm the superiority of the integrated approach over the sequential approach in terms of the overall objective function. In addition, Desaulniers (2007) propose a method for dealing with a bi-level objective function for a VCSP that is solved by column generation.

Laurent, Guihaire, and Hao (2005) study a problem more or less comparable to the one discussed in this thesis. In this problem, limousines and limousine drivers have to be assigned to a given set of trip demands. Their first objective is maximization of the number of trips covered. Furthermore, running costs and waiting time are minimized. As opposed to most articles concerning crew scheduling problems, driving time between a certain end location and the start location of the following trip is taken into account. The other restrictions assign vehicle capacity, driver skill and the duration of a driver's shift. Two local search methods, hill climbing and simulated annealing, are used to improve the initial solution obtained by a greedy algorithm.

The article by Laurent et al. (2005) has a number of similarities with the resource assignment problem studied in this thesis. The main similarities with the problem studied in this thesis are the fact that a given set of trips has to be covered by vehicles and drivers, and that driving time between a certain end location and the start location of the following trip is taken into account. However, the other restrictions only assign vehicle capacity, driver skill and the duration of a driver's shift.

3.5 Driving time legislation

Archetti and Savelsbergh (2009) present an algorithm capable of finding within polynomial time a feasible driver schedule for a set of full truckload transportation request, if such a schedule exists. In this schedule, the Hours of Service regulations as applicable in the United States are taken into account, as well as time windows for the concerned pickups and deliveries. However, a schedule found by this algorithm is not guaranteed to be optimal in terms of any objective function, such as costs.

Goel (2009) states that, despite their importance in practice, regulations concerning drivers' working hours have not received much attention in literature on vehicle scheduling. Moreover, he is likely to be the first author addressing the current regulations for drivers' working hours in the European Union. He describes two methods for scheduling driving periods, breaks, rest periods and other activities, according two these regulations. The main difference between these methods is that only one of them is capable of scheduling breaks earlier than actually needed. Next, it is shown how either one of the methods can be incorporated into the vehicle routing problem with time windows (VRPTW), which is then solved by applying a large neighborhood search method. Based on computational experiments, it is concluded that the possibility to schedule rest periods earlier than needed can significantly improve the resulting schedules in terms of the number of vehicles scheduled and distance to be traveled. However, the described solution method is not guaranteed to give an optimal solution, nor will it always find a feasible schedule when such a schedule exists. Furthermore, all optional exceptions in the regulations on driving time are ignored, and all European regulations regarding working time are not taken into account.

A solution method incorporating all regulations is presented by Kok, Meyer, Kopfer, and Schutten (2009). They describe a break scheduling method that can be incorporated in a dynamic programming algorithm, such that the relevant legislation is taken into account. The results show that this approach significantly outperforms any heuristics used for the VRPTW, with less computational effort. Furthermore, it is shown that considering the optional rules in the regulations adds to the quality of the resulting schedules.

Goel (2010) proposes several variations of a breadth-first search (BFS) algorithm for scheduling driving and working time in compliance with all EC legislation. These methods can be integrated with local search methods for combined vehicle routing and truck driver scheduling. If one or more feasible schedules for a certain instance exist, the presented methods are guaranteed to find one. However, an optimal solution might not be found. Furthermore, the BFS algorithms require considerably more computational effort than other methods for this problem, such as the ones discussed earlier in this section.

3.6 Discussion

3.6.1 Relevance of general problems to the RAP

In the above sections, a number of general problems related to the resource assignment problem is discussed. To start with, the VRP differs from the problem in this thesis in the fact that the VRP assigns vehicles to fixed locations, rather than planned trips. However, many issues of the resource assignment problem, such as driving between tasks, time windows, and multiple vehicle types, are or could be relevant in the VRP as well.

The VSP seems most similar to the resource assignment problem. In the literature, the VSP is mostly related to public transport, in particular transport by bus. The main similarity with the resource assignment problem is the input: a set of resources and planned trips, which have to be matched. A difference however is the use of depots. In public transport, buses have to start and end their route at the same depot, and only a limited number of depots is present. In the RAP however, the resource shifts of a truck and driver, combined before the start of the optimization, may have any start and end location, for instance at the driver's home address. For distribution instances, trips might also start or end at a depot, but still the travel from and to the driver's home might have to be planned. Another difference is the frequency of optimization. Public transport schedules usually remain the same over several months at least (except for disturbances), meaning that optimization will mostly be applied once in a while to establish a fixed schedule. This means much computational time is available. In a transportation environment however, a different schedule is needed every day, based on the particular orders received for that particular day. This means optimization is used much more frequently, and as a result, computational time needed becomes more of an issue.

The CSP is less similar to the RAP, but is nevertheless considered in this chapter because of the issues on personal planning. The scheduling of crews usually has more restrictions to it compared to the scheduling of vehicles, such as total working time, breaks, etc. These kind of restrictions are often not accounted for in approaches to the VSP. An important difference with the RAP is that a crew cannot move from the end location of a trip to the start location of the next by itself, whereas a combination of truck and driver can.

In contrast with the VSP and CSP, the VCSP considers both vehicle and crew scheduling issues. The main difference with the problem in this thesis is the assignment of multiple resources to a planned trip. Nevertheless, it is assumed that a method capable of dealing with the VCSP should also be applicable to the RAP.

From the literature, it can be concluded that incorporating driving time legislation may add severe complexity to the scheduling problem. However, not much literature is present on this issue, especially not with respect to the current legislation in the European Union. More research on how driving time legislation can be incorporated in the various approaches to scheduling problems would be desired.

3.6.2 Methods

From the literature on the VRP, VSP and CSP, it can be concluded that two approaches are the most common. First, column generation is applied in many cases to solve the linear relaxation of the main problem concerned. The subproblems are then solved either optimally or heuristically, often depending on complexity and size. In most cases, this approach seems capable of dealing with large problems within acceptable time. The second common approach is the application of local search algorithms. As column generation, this approach seems capable of dealing with large instances as well as additional restrictions on the problem.

It appears from the literature that the exact same problem as addressed by this thesis has not yet been dealt with. However, the approaches and results for comparable problems seem promising for solving the resource assignment problem.

Chapter 4

Trip assignment method

In chapter 2, the resource assignment problem, as faced by ORTEC, was described. In practice, ORTEC will not offer the eventual solution to its clients at once. Because of the complexity of the problem, it was decided to split the development into two phases, such that already part of the new functionality can be delivered before the entire development process is finished.

The next section of this chapter will discuss which parts of the development process are actually assigned to the first and second development phase. After that, ORTEC's solution for the problem dealt with in the first phase will be explained. This solution is partly based on the literature discussed in the previous chapter.

4.1 Development phases

As explained in section 2.2, an execution trip consists of a number of consecutive sections. In each section, a different set of resources may be used, which remains the same during the section. The use of sections allows for a more flexible planning and specific scenarios such as preloading, since resources can be changed within an execution trip. However, this also creates a dependency between resource shifts, as different resource shifts may contain sections that belong to the same execution trip. This dependency was already discussed in chapter 2 and is visualized in Figure A.1 in Appendix A.

Because of this dependency as well as the extensive efforts required to facilitate the assignment of individual sections in ORTEC TD, it was decided that in the first development phase, only entire execution trips can be assigned. This means that when a particular execution trip consists of multiple sections, all of these sections will be assigned to the same resource shift, which is done simultaneously. The method developed in this phase will from now on be referred to as the *trip assignment method*. The method we developed in the second phase will be referred to as the *section assignment method*. Figure 4.1 shows the difference between the functionality of both methods.

The contribution of this research is different in both development phases. Regarding the trip assignment method, this thesis provided a collection and discussion of relevant literature to base the method on. Furthermore, the remaining of this chapter extensively describes the resulting method. Finally, in chapter 5 the effects of a small improvement to this method are examined, and in chapter 7 the application of this method within ORTEC is tested. Regarding the second development phase, we

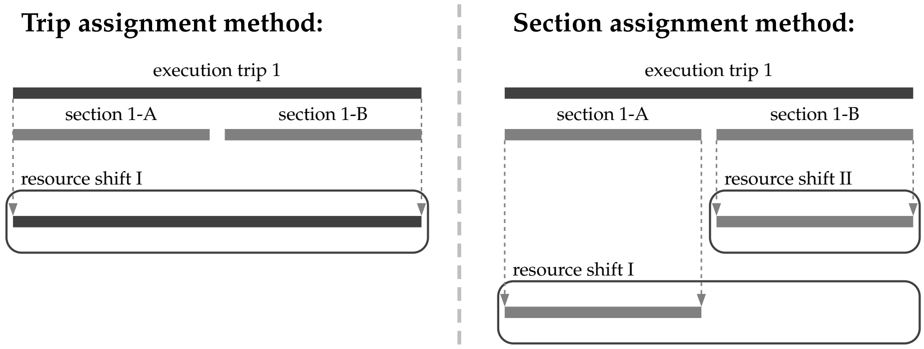


Figure 4.1: Assignment methods

will develop a new method, based on the previous method, to deal with the assignment of individual sections. It is elaborated in chapter 6. This method can be used by ORTEC and implemented within ORTEC TD.

4.2 Framework

The approach that will be used for assigning trips to resource shifts is based on the method applied by Xu et al. (2003), as discussed in chapter 3. This is a column generation approach, which in the previous chapter was concluded to be a suitable method for solving the resource assignment problem.

To obtain a mathematical formulation of this problem, let T be the set of execution trips that have to be assigned to the available resource shifts, which are denoted by S. Furthermore, the set F represents all feasible assignments, where an assignment is defined as a combination of a resource shift and one or more trips. An assignment is considered feasible if it does not violate any of the relevant restrictions as explained in section 2.5, such as time windows and driving time legislation. Consequently, these restrictions do not have to be shown explicitly in the master problem formulation.

Next, the parameters c_f, $a_{t,f}$ and $b_{s,f}$ and the decision variable x_f are defined as follows:

$$
\begin{aligned}
c_f &= \text{costs of resource assignment } f \\
a_{t,f} &= \begin{cases} 1 & \text{if execution trip } t \text{ is in resource assignment } f \\ 0 & \text{otherwise} \end{cases} \\
b_{s,f} &= \begin{cases} 1 & \text{if resource shift } s \text{ is in resource assignment } f \\ 0 & \text{otherwise} \end{cases} \\
x_f &= \begin{cases} 1 & \text{if resource assignment } f \text{ is in solution} \\ 0 & \text{otherwise} \end{cases}
\end{aligned}
$$

With these parameters, the master problem (MP) can be formulated as a set partitioning problem. In this case, this means that a number of assignments has to be selected, such that each trip is covered exactly once and each resource shift is not used more than once.

$$\begin{aligned}
\min \quad & \sum_{f \in F} c_f x_f & & & (4.1)\\
& \sum_{f \in F} a_{t,f} x_f = 1 & & t \in T & (4.2)\\
& \sum_{f \in F} b_{s,f} x_f \leq 1 & & s \in S & (4.3)\\
& x_f \in \{0,1\} & & f \in F & (4.4)
\end{aligned}$$

In the above program, the objective function sums up the costs of all selected assignments, which should be minimized. Constraint (4.2) is incorporated to ensure that all trips are assigned exactly once. Constraint (4.3) ensures that each resource shift is used at most once in all selected assignments. Obviously, integrality is secured by constraint (4.4), such that each column is either selected or not.

4.3 Approach

The approach for solving the resource assignment problem consists of seven steps, which are elaborated below. The procedure is also visualized in Figure 4.2.

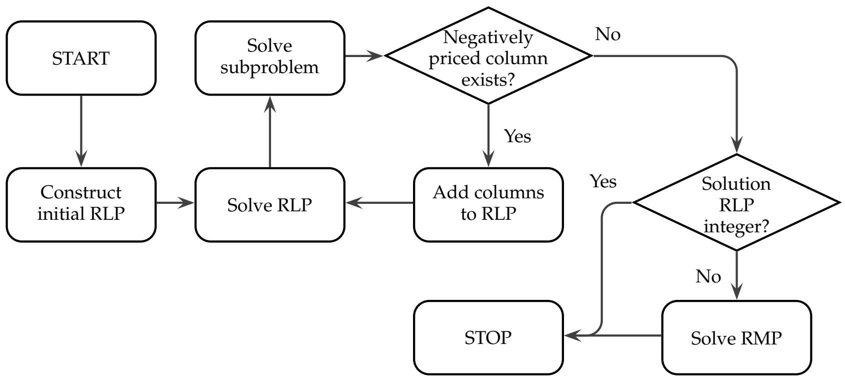


Figure 4.2: Column generation algorithm

Step 1: Linear relaxation

The linear relaxation of the above problem (LP) is obtained by replacing constraint (4.4) by $0 \leq x_f \leq 1, f \in F$. This basically means that a column can be 'partially' selected.

Step 2: Generate initial set of columns

In the column generation procedure, each assignment is stored as a column. In this step, an initial set of columns is generated by constructing dummy columns. For each trip to be assigned, a column is created in which no trips or shifts are selected except for this particular trip. These columns are given very high costs, such that any other assignment is a cheaper option. By creating these dummy columns, it is assured that later in the procedure, a solution for the master problem can always be found.

Step 3: Solve current RLP

The restricted version of LP, in which only the columns generated earlier can be selected, is referred to as RLP. In this step, a solver is used to solve the current version of RLP. The used solver is COIN-OR Linear Program solver (CLP), which is an open-source application for linear programs, written in C++.

From the solution to RLP, the corresponding values of the dual variables are obtained. These dual variables are π_t, $t \in T$ and σ_s, $s \in S$, where π_t represents the dual variables corresponding to constraint (4.2) and σ_s represents the dual variables corresponding to constraint (4.3).

Step 4: Solve subproblem

The next step involves finding new columns with negative reduced costs. These costs are calculated as follows:

$$r_f = c_f - \sum_{t \in T} \pi_t a_{t,f} - \sum_{s \in S} \sigma_s b_{s,f} \tag{4.5}$$

In order to create new columns, for each resource shift it is attempted to add each trip to this shift at least once. For this purpose, the existing Dynamic Programming (DP) algorithm in ORTEC TD is used.

This algorithm starts with the empty resource shift, and then examines for each trip whether it can be assigned to the shift, taking into account all restrictions described in section 2.5. Each of the resulting feasible assignments is represented by a node, which is defined by the set of trips that has been assigned, and the trip within that set that was assigned most recently. These nodes are then sorted on their associated reduced costs, and the algorithm proceeds with only the first H nodes, where H is a parameter that can be adjusted by the user. The use of this restrictive parameter may keep the algorithm from finding the optimal solution, but also reduces computational time. Another parameter E can be used to let the algorithm consider only the E nearest unassigned trips from each node.

Starting from the remaining nodes, a new stage is created by attempting to assign another trip to the assignments represented by the current set of nodes. For the resulting nodes, the reduced costs are calculated, and so on. A single node can be reached from multiple nodes in the preceding stage, which is the case when the set of assigned trips in these nodes is equal, as well as the trip to assign next. In this case, the lowest reduced costs are given to this node.

The procedure terminates when no additional trips can be added to the current set of nodes. Each node in the resulting graph represents a feasible assignment and thus a column that can be used in the column generation. However, only the columns with negative reduced costs are stored, as these can improve the solution to the master problem. A more elaborate description of the DP algorithm used in ORTEC TD is provided by Gromicho et al. (2009).

Figure 4.3 shows an example graph for the assignment of three trips to a resource shift. In this example, H and E are set large enough to allow all possible nodes, and all nodes are feasible. Each node represents the assignment of a number of trips to the shift. For example, {A,B},A represents a shift with trips A and B assigned to it, of which A was assigned last.

If, after termination of the DP algorithm, no columns with negative reduced costs have been found, the solution to RLP obtained in step 3 is maintained and the algorithm proceeds with step 6. If such columns are found, the procedure continues with step 5.

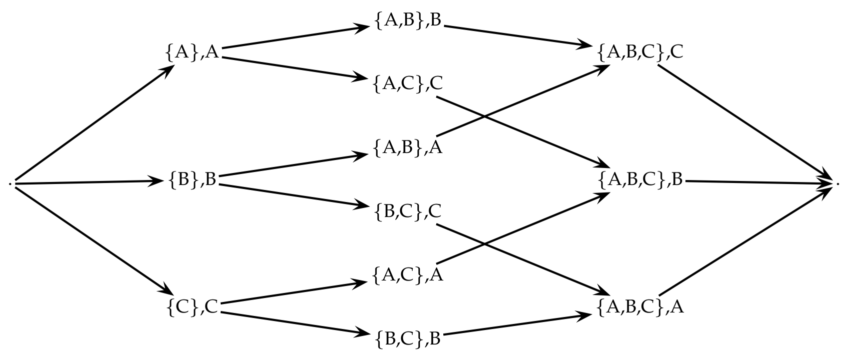


Figure 4.3: Dynamic programming example

Step 5: Add columns to RLP

The columns with negative reduced costs, found in the previous step, are added to RLP. After that, the algorithm returns to step 3.

Step 6: Integrality

When the execution of step 4 does not result in additional columns with negative reduced costs, it is checked whether the most recent solution to RLP obtained in step 3 is integral, i.e. whether constraint (4.4) from the master problem holds for all decision variables. If this is indeed the case, the solution for the linear relaxation is also feasible for the original, integer problem, and the procedure terminates. Otherwise, the procedure proceeds with step 7.

Step 7: Solve RMP

In order to obtain a solution for the original master problem, a solver called COIN-OR Branch and Cut solver (CBC) is used. The main reason for selecting this solver is its free availability. Commercial licenses for other solvers, such as the widely used CPLEX-solver, may be very costly, which would cause ORTEC TD, and the Resource Assignment Module in particular, to be considerably less attractive to potential customers.

The solver is applied to the integer version of RLP, referred to as RMP. This problem contains the dummy columns created in step 2 and all columns that were added in step 5. The obtained solution is then used as solution for the master problem.

Chapter 5

Grouping of identical columns

In the procedure described in chapter 4, feasible assignments have to be generated for each available resource shift. According to business experts at ORTEC, in practice many resource shifts may be identical to each other. This is the case when all properties (location, availability, driver status) of a number of resource shifts are the same, such that the subproblems for these shifts are identical as well. In the current procedure, this means that a lot of assignments are created more than once, requiring more computational time than strictly necessary. This could be be prevented by grouping identical resource shifts together. ORTEC was curious to know how this would affect the column generation procedure. In this section, it is shown that this does not affect the eventual solution.

The dual problem (DP) can be formulated as follows:

$$\max \quad \sum_{t \in T} \pi_t + \sum_{s \in S} \sigma_s \tag{5.1}$$

$$\sum_{t \in T} \pi_t a_{t,f} + \sum_{s \in S} \sigma_s b_{s,f} \leq c_f \qquad f \in F \tag{5.2}$$

$$\pi_t \in \mathbb{R} \qquad t \in T \tag{5.3}$$

$$\sigma_s \leq 0 \qquad s \in S \tag{5.4}$$

Now let the first two resource shifts ($s = 1$ and $s = 2$) be identical. For the first resource shift $s = 1$, let a number of n feasible assignments be available. Because the first two resource shifts are identical, this implies that the same n assignments are available for $s = 2$.

From the above dual problem, constraint (5.2) can be formulated as follows for the resource assignments in which the first resource shift is used:

$$\sum_{t \in T} \pi_t a_{t,f} + \sigma_1 \leq c_f \qquad f \in [0, 1, ..., n] \tag{5.5}$$

Similarly, constraint (5.2) looks as follows for the assignments using the second resource shift:

$$\sum_{t \in T} \pi_t a_{t,f} + \sigma_2 \leq c_f \qquad f \in [n+1, n+2, ..., 2n] \tag{5.6}$$

It is assumed that the available assignments for the first and the second resource shift are sorted in the same order. Thus, because the first two resource shifts are identical, assignment $f = i$ covers the same trips as assignment $f = i + n$ for $i = 1, 2, ..., n$. Hence, $a_{t,i} = a_{t,i+n}$ for $i = 1, 2, ..., n$. Similarly, $c_i = c_{i+n}$ for $i = 1, 2, ..., n$.

As a result, all terms in (5.5) and (5.6) except for σ_1 and σ_2, respectively, are equal. Because both σ_1 and σ_2 have a positive coefficient in the objective function of DP, it is obvious that in the optimal solution, they will have the same value.

As the sigmas for identical resource shifts are equal, it follows from the formula for the reduced costs of a column, as given by equation 4.5, that the reduced costs of two identical assignments are the same as well.

If identical resource shifts are grouped together, the formulation of the resource assignment problem changes. The set S, previously denoting all available resource shifts, now represents all unique resource shifts, which will be referred to as resource shift types. Furthermore, n_s represents the number of available resource shifts from type s.

The second version of the resource assignment master problem (MP2) can now be formulated as:

$$\min \quad \sum_{f \in F} c_f x_f \tag{5.7}$$

$$\sum_{f \in F} a_{t,f} x_f = 1 \qquad t \in T \tag{5.8}$$

$$\sum_{f \in F} b_{s,f} x_f \leq n_s \qquad s \in S \tag{5.9}$$

$$x_f \in \{0, 1\} \qquad f \in F \tag{5.10}$$

The objective function of the dual problem now becomes:

$$\max \quad \sum_{t \in T} \pi_t + \sum_{s \in S} n_s \sigma_s \tag{5.11}$$

The constraints of the dual problem do not change as a result of grouping identical resource shifts. This leads to the conclusion that if the resource assignment problem is rewritten to the formulation of MP2, the value of σ_s in MP2 for a certain resource shift type s is equal to the values of σ_s in MP of the individual resource shifts that are grouped into that resource type group.

Chapter 6

Section assignment method

As explained in section 4.1, the method described in the previous chapter does not deal with all aspects of the resource assignment problem. It can be applied to assign entire trips to resource shifts, but ignores the fact that a trip may consist of multiple sections which, in reality, can be assigned individually. Therefore, in this chapter we will describe a method which is capable of dealing with the assignment of individual sections, the so-called section assignment method.

This chapter will start by explaining why the method described before cannot be used to assign individual sections. Next, the alternative approach will be explained by elaborating every step of the developed procedure.

6.1 Shortcomings trip assignment method

As explained earlier, trips may contain multiple sections. The reason for maintaining these sections within the same trip, is that they should be planned in the right order. For instance, the first section within a particular trip may take a trailer from point A to point B, whereas the second section takes it from point B onwards to point C. Although these two sections can be assigned to different resource shifts, the second section obviously cannot start until the first one has finished. The required order of sections creates a dependency between various assignments. For example, if two assignments both contain a section of the same trip and the first section is not completed at the time the second section starts, the two assignments cannot be selected both.

An important aspect of the trip assignment method described in chapter 4 is the division between the master problem and the subproblem. Most of the constraints, such as the rather complex driving time legislation, are dealt with in the subproblem, in which assignments are created. Next, the master problem selects assignments only based on the covered trips, the used resource shift, and the associated costs. This means that start and end times of trips or sections are ignored, and thus that the dependency between sections of the most trips is not taken into account. Using the trip assignment method to assign individual sections to resource shift may therefore result in a solution that is infeasible in practice. Therefore, we develop an alternative approach.

6.2 Section assignment method

The most straightforward approach for the problem faced in the second development phase would be to assign individual sections instead of entire trips, and adjust the algorithm such that the interdependency between sections belonging to the same trip is taken into account. This could be achieved by checking dependency with each of the newly generated columns with all of the other columns in the problem, and subsequently adding constraints to the master problem which for each pair of conflicting columns prevent that both are selected. However, the number of checks needed in this procedure might easily become very large. If a total of n columns are generated, then $n(n-1)/2$ checks have to be made.

Another disadvantage of assigning individual sections instead of trips is that it will likely result in the generation of many more columns, because of two reasons. First, multiple columns are needed to cover a trip consisting of multiple sections. Second, the number of possible assignments will be considerably larger, as more combinations can be made. This will have a negative impact on the required computational time.

We developed the procedure for the sections assignment problem in such a way that the impact of these implications on the performance of the algorithm is minimized. This is achieved by first assigning only entire trips to the available resource shifts, as is done by the trip assignment method. Only after this part of the procedure has terminated, the assignment of individual sections is considered.

This approach has a number of advantages over an approach which immediately starts with assigning sections. First, the available options in the first part of the procedure are limited and dependency between columns does not have to be taken into account. The second, more computationally intensive part is only used to improve the solution, if possible. A second advantage is that the number of checks on dependency between assignments is minimized. The set of columns generated in the assignment of full trips does not have to be taken into account when checking dependency, as the set partitioning formulation of the master problem prevents that a columns which covers all sections of a trip is selected together with a column that covers only part of the sections belonging to that trip.

The section assignment procedure is explained in more detail below, and visualized by the flow chart in Figure 6.1.

6.3 Approach

Step 1 to 6: Assignment of trips

For the first part of the approach, steps 1 to 6 from the trip assignment method are used, as described in section 4.3. In these steps, the linear relaxation of the master problem is used, and columns are added by the DP-algorithm for solving the subproblem, until no additional columns with negative reduced costs can be found.

Step 7: Conversion of problem

In order to facilitate the assignment of sections instead of trips, the problem is adjusted, such that each column now represents an assignment of sections to a resource shift. This is done by replacing each column, of which the length equalizes the number of trips to assign plus the number of resource shifts, by a column with length equal to the total number of sections plus the number of resource shifts. A '1' is then

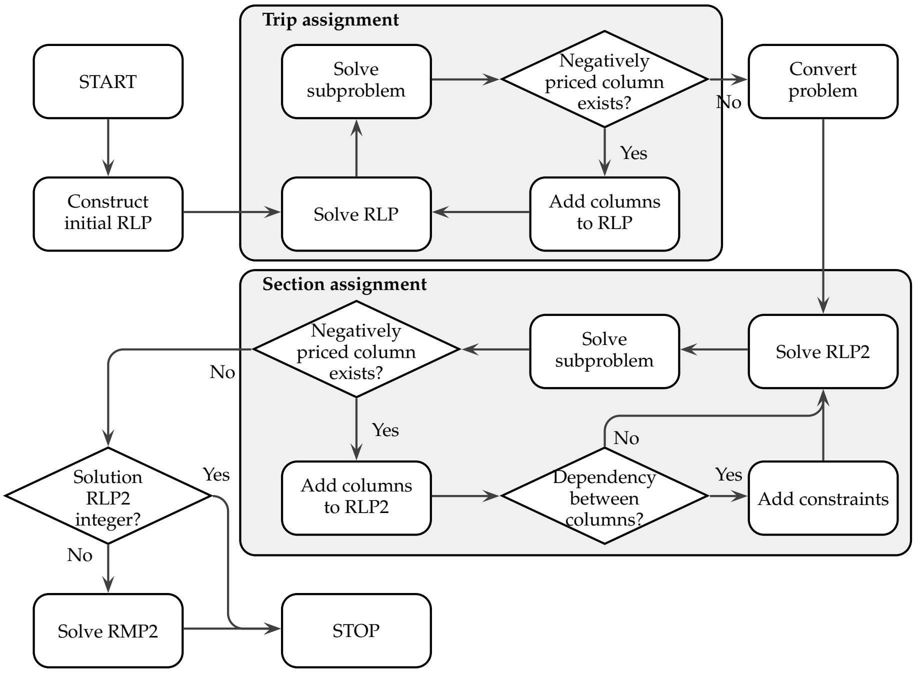


Figure 6.1: Section assignment method outline

placed at every position that represents a section from a trip that is covered by that particular assignment. If for instance a certain column covered two trips which both contain three sections, this column is then replaced by one that covers all six of these individual sections.

The resulting version of the relaxed master problem is referred to as RLP2.

Step 8: Solve current RLP2

The master problem, RLP2, is solved using the CLP solver. In the succeeding steps, extra constraints can be added to incorporate the dependency between sections. These constraints will then be taken into account in this step.

Step 9: Solve subproblem

Sections are assigned to the resource shifts in the same manner as trips earlier in the procedure. For each generated column, the start and end times of the assigned sections are stored as well. This information is needed later on, in order to determine which columns conflict with each other. If columns with negative reduced costs are found, these columns will be added to the problem in step 10. Otherwise, the algorithm jumps to step 13.

Step 10: Add columns to RLP2

All columns with negative reduced costs are added to the master problem. This is done regardless of any dependency between columns.

Step 11: Check interdependency

For each of the newly generated columns, it is checked whether there are existing columns in the master problem which cannot be selected together with the evaluated column, due to the precedence restriction. This is done by first looking for columns which cover different sections from the same trip. For each of the columns found, it is also checked whether it covers at least one section that is also covered by the newly generated columns. If this is the case, no constraint has to be added, as the master problem constraint ensuring each section to be covered exactly once (constraint (5.8)) already prevents both columns from being selected both. Otherwise, it is checked for each section in the new column, whether there is a succeeding section covered by the existing column that starts before that section has ended, or a preceding section that ends after the start of that section. In both cases, it is determined that the two columns cannot be selected at the same time.

Figure 6.2 shows three cases where the precedence restriction is checked. In this picture, '1-A' for example stands for trip 1, section A. In the first case, the consecutive sections of trip 2 are assigned to different shifts, but the second section starts after the first one has finished, so both assignments can be selected in the master problem at the same time, and no constraint has to be added. In the second case, the precedence restriction is violated, but adding a constraint is not necessary because at most one of the two assignments will be selected, since they both cover section 3-A. In the third case, an additional constraint is needed to prevent both assignments from being selected.

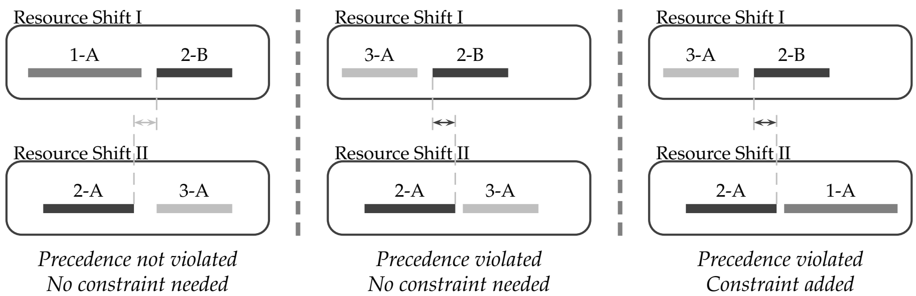


Figure 6.2: Precedence cases

Step 12: Add constraints

In order to prevent two conflicting columns from being selected both, the most straightforward procedure would be to add a constraint for each pair of conflicting columns k, l:

$$x_k + x_l \leq 1 \tag{6.1}$$

However, the number of columns in the master problem can easily become considerably large, which may in turn result in a very large number of conflicting columns and thus constraints to be added. In order to reduce the total numbers of constraints, a different approach is used. In the previous step it was checked for each new column whether it conflicts with any of the existing columns. Any conflicts with existing columns are then stored by constraints (6.2), where the parameter d_i is defined as follows:

$$d_{i,n} = \begin{cases} 1 & \text{if existing column } i \text{ conflicts with new column } n \\ 0 & \text{otherwise} \end{cases}$$

$$\sum_{i=1}^{n-1} d_{i,n} x_i + \sum_{i=1}^{n-1} d_{i,n} x_n \leq \sum_{i=1}^{n-1} d_{i,n} \tag{6.2}$$

As an example, presume that a check for the newly generated column $n = 4$ reveals that it conflict with columns 1, 2 and 3. The constraint $x_1 + x_2 + x_3 + 3x_4 \leq 3$ then ensures that the fourth column will not be selected together with any of the first three columns.

Step 13 and 14: Solve master problem

The last steps are similar to those of the trip assignment method. It is checked whether the latest solution to the RLP is integer. If so, it is used as final solution to the master problem. If not, the CBC solver is used to obtain an integer solution, using the columns from the RLP.

Chapter 7

Case study

During the execution of this research, the method for assigning trips to resource shifts, as described in chapter 4, was implemented and integrated in the ORTEC TD product. In order to test this functionality, a test case from an actual client is used. This particular client is a large retail company, which uses ORTEC TD to plan distribution rounds, starting from and ending at a central depot.

This chapter starts with a short description of the implementation of the method. Next, the details of the case are described, the test results are given and these are discussed. Finally, conclusions are drawn based on this test case.

7.1 Implementation

The column generation approach for the assignment of trips to resource shifts is implemented by ORTEC, using the Python programming language. Obviously, a number of additions to the ORTEC TD graphic user interface are made as well, such that the resource assignment functionality can be configured and utilized by the end user. The tests are performed on a PC with Intel Xeon X5550 2.66Ghz processor.

7.2 Case description

The data for this case are obtained from a large retail company, operating worldwide. Here, its transport activities in one particular country are concerned. In this country, the company operates three depots, from which a large number of shops throughout the country is supplied. Here, the planning activities at each depot are independent of each other, so each depot represents a different instance. The planning process starts with the combination of a number of distribution orders into a trip, such that each trip starts at the depot with a particular load, then visits a number of shops in a given order to unload, and finally returns at the depot empty. Each trip is given a specific trailer type. At that point in the planning process, the resource assignment functionality is used to allocate these trips to a given set of resource shifts, which contain a driver and a truck. When the resulting plan is executed, the employees loading the trailers at the depot decide which trailer to use for each order, and communicate this to the planning department such that the data in ORTEC TD can be updated accordingly. A plan is created per 24-hour period.

7.3 Results and discussion

The results are displayed in table 7.1.

Test problem				Trip assignment				
dep.	# of trips	# of shifts	# of shift types	gap (%)	# of shifts used	CPU	avg # of trips per shift	max # of trips per shift
A	21	12	12	4,71	11	57	1,91	3
B	13	8	8	0,00	8	76	1,63	2
C	64	69	3	0,00	37	169	1,73	2

Table 7.1: Test results distribution case

The fifth column of the results table shows the gap between the solution value of the final, integer-valued problem and of the linear relaxation of this problem. In the first instance, the integer solution value lies within 5% of this lower bound, whereas in the other two instances, the value of the objective function matches the lower bound. This could indicate that the solution to the LP-problem was already integer valued, although this does not necessarily have to be the case. Nevertheless, an integrality gap of zero percent means that the solution value is optimal.

Computation times for the above cases range from roughly one to three minutes, which can be considered acceptable. Despite the fact that in the first case there are more trips to assign, more resource shift types and more trips per resource shift compared to the second case, the computation time for this case is lower. Based on the available data from these cases, this cannot be fully explained. The most likely cause is the variation in computation times, which can be rather large, as will be pointed out in the next chapter as well. Furthermore, it is uncertain how high computation times will be when the instances get larger. Nevertheless, it is worth mentioning that manual assignment of trips in ORTEC TD would require at least the same amount of time just for the necessary actions to be completed. Considering the fact that the user would also need time to gather information to base his decisions on, and that he might follow a trial-and-error procedure, it is likely that the resource assignment functionality will outperform the user in terms of speed.

Although the third case is considerably larger than the first two cases in terms of trips and resource shifts, computation time is less than three times as high. This can very likely be explained by the fact that the number of resource shift types is lower. As the column generation algorithm creates assignments for each unique resource shift, less resource shift types means less DP calls and therefore a smaller master problem.

7.4 Conclusion and recommendations

From the distribution case, it can be concluded that the resource assignment module is capable of assigning trips to resource shifts within acceptable computation time, at least for instances of limited size. As could be expected, the results clearly show that the number of unique resource shifts has a significant impact on computation time, as the DP-algorithm has to be called for every unique resource shift. It is therefore strongly recommended to keep the number of unique resource shifts as low as possible. Several procedures might be used for this, such as leveling out small differences in drivers' statuses regarding driving time, or reducing the number of start locations.

Furthermore, the integrality gaps indicate that, at least in the three instances considered here, the trip assignment method produces integer-valued solutions of which the objective value approach or even match the lower bound obtained by solving the linearly relaxed problem.

Chapter 8

Method comparison

In chapter 4 of this thesis, the method used to assign trips to resource shifts was described. Furthermore, chapter 6 described an extension to this method, which we developed to allow for the assignment of the individual sections of which the trips consist. In this chapter, these two methods will be compared to each other, based on a computational study.

This chapter starts with a description of the case on which both methods are tested. Next, the application of both methods is explained, and the results are given. The chapter ends with a discussion of these results, and with a number of conclusions and recommendations regarding the use of both methods.

8.1 Implementation

Both the trip assignment method and the section assignment method are implemented in MATLAB. The tests are performed on a PC with an Intel Duo Core T2400 1.83GHz processor. In the first application of the section assignment method, it appeared that the time needed to solve the MILP problem at the end of the procedure may vary heavily, even for instances with similar specifications. For the largest instances, computation times mostly fluctuated around several minutes, but could sometimes be as high as several hours. It was therefore decided to limit the time used to solve the MILP in the section assignment method to fifteen minutes. If this maximum is reached, the best feasible solution found so far is used. If no feasible solution is available, it is concluded that the section assignment method is in this case not capable of improving the solution found by the trip assignment method. Furthermore, for some of the largest instances considered in this case, solving the MILP incidentally failed due to a lack of memory. In most of these cases, this was prevented by setting the time limit. In the few other cases where this happened before the time limit, these cases are dealt with similar to cases in which the time limit is reached before the MILP is solved.

8.2 Case description

The solution methods will be tested on randomly generated instances, such that multiple instances can be used. Furthermore, this allows for examining the impact of variations in the case parameters, such as area size and the length of a resource shift, on the performance of both methods. The generation of the test cases is however roughly based on an actual case from practice. This ensures that the results from the test case are representative for real life cases.

In the case used to base the tests on, a transportation company picks up and drops off loads throughout the Netherlands. The planners combine these orders into trips, each of which uses a particular trailer. Next, the planners might cut longer trips into multiple sections to gain more planning flexibility. Furthermore, the available drivers and trucks are joined into resource shifts. These resource shifts start at a limited number of different times and places, and return to their start location at the end of the shift. Each resource shift has a maximum duration of fourteen hours.

In order to roughly simulate the circumstances from this case, a square area of 250 by 250 kilometers is used. Trucks travel through this area with a constant speed of 60 kilometers per hour. The start and end location of trips as well as the start location of the resource shifts are randomly placed within this area, by letting both coordinates be uniformly distributed on the interval [0 250]. The fourteen-hour resource shifts are divided into early, mid-day or late shifts, starting at 0:00, 5:00 and 10:00, respectively. Since in the real-life case, time windows may vary per order, this is simulated as well. The time window is represented by a random variable, which is uniformly distributed between zero and four hours. The earliest start time and latest finish time of a particular trip therefore differ by the value of this variable plus the duration of the trip.

8.2.1 Variations

In order to gain an understanding of how well both methods perform in different cases as well as of the impact of different parameters on the performance, different instances of the case described above are generated. These instances differ on three parameters, viz., the number of trips to be assigned, the maximum number of sections within a trip, and the number of resource shift types available. These parameters are chosen because they will vary in practice as well, both within this case and among different cases.

- **Number of trips**: The number of trips that have to be assigned in the test instances will be either 25, 50 or 100, as those numbers fall within the range for which the resource assignment functionality will be used the most in practice. Note that if each trip contains multiple sections, the number of individual sections to assign is much higher.

- **Maximum number of sections per trip**: This parameter defines the maximum number of sections in which a trip can be split up. If, for example, the value of this parameter is 3, the trips are divided into three groups based on the distance to travel. The trips in the group with the longest distance are each split up into three sections, the second group is split up in two sections, and the shortest trips are not split up at all. For the test instances, the maximum number of sections per trip is set to either two, three or four, as this corresponds to practice.

- **Number of resource shift types**: As explained in section 8.2, the resource shifts start at three different times. Furthermore, resource shifts can have different start locations. Because identical resource shifts can be grouped together to reduce computation time, the number of resource shift types will be given different values. In the test instances, either three or ten resource shift types will be available. The total number of resource shifts is set equal to the number of trips, such that ample capacity is available.

By combining the different values of the above parameters, 18 different instances have to be tested. For each individual instance, 20 tests are performed. Together, these instances will provide a representative impression of the performance of both methods. However, all instances are based on the same real-life case. Other cases may differ from this case on various parameters. Two of those, area size and time window length, will be different in each particular case and, moreover, are expected to have a severe impact on the performance of the methods. Therefore, their effects are examined as well.

- **Area size**: As stated earlier, the area size used in the previous test was 250 by 250 kilometers. To examine the effect of the area size on the test results, instances with a smaller (100 x 100) and a larger (400 x 400) area will be tested. Two instances from the above tests will be used as a benchmark.
- **Time window length**: In chapter 6 it was explained that the section assignment method adds constraints to the master problem if the precedence restriction between two columns is violated. The DP part of the column generation algorithm plans trips or sections simply as early as possible, without taking this restriction into account. The larger the time window length, the more flexibility the algorithm has to vary the start time of each individual section. It is expected that because of this, the precedence restriction will be violated more often and therefore more constraints have to be added, which will in turn affect the performance of the section assignment method. Therefore, this effect is worth examining. In the above tests, time window varies between 0 and 4 hours. For this test, the time windows will be randomly chosen between 4 and 8 hours, and between 8 and 12 hours.

8.3 Results

Table 8.1 shows the results of the method comparison in MATLAB. The first thee columns show the values of the instances parameters, the remaining columns show the averages over the 20 tests performed per instance. The fourth and eighth columns show the relative gap between the solution value of the integer problem and of its linearly relaxed version. Further, two columns per method show computation times. The first column shows the time consumed by the entire execution of the method, the second shows the time that was used to solve the MILP problem, which is the last step of both methods. The rightmost column shows the average improvement the section assignment method gained over the trip assignment method.

In table 8.2, the results of the variation in area size are shown. The results of the variation in time window length are shown in table 8.3.

8.4 Discussion

The main purpose of the tests described above is to make a comparison between the two methods developed for the resource assignment problem. The first and most important finding that results from the tests is that in most cases, the assignment of individual sections and therefore the application of the section assignment method proves capable of yielding a better solution value than the trip assignment method. The improvement of total costs achieved by the section assignment method over the trip assignment method varies between zero and 8%, but averages 0.82% overall.

Test problem			Trip assignment method				Section assignment method					
# of trips	max # of sections per trip	# of shift types	Int. gap (%)	CPU (s)	CPU MILP (s)	# of columns	Int. gap (%)	CPU (s)	CPU MILP (s)	# of columns	# of constr. added	reduction (%)
25	2	3	0.07	0.42	0.04	122.8	0.31	2.63	1.40	295.8	54.0	0.55
25	2	10	0.00	1.04	0.00	273.0	0.30	6.61	3.37	668.6	130.0	2.58
25	3	3	0.00	0.33	0.00	117.2	0.26	6.70	4.66	517.2	215.2	2.50
25	3	10	0.06	1.10	0.15	355.2	0.09	18.19	11.05	1221.8	520.8	3.09
25	4	3	0.03	0.34	0.02	113.8	0.04	69.50	65.24	816.4	500.2	0.13
25	4	10	0.10	1.70	0.14	388.6	0.94	281.31	241.79	2496.4	1571.0	0.29
50	2	3	0.00	2.93	0.00	367.0	0.01	9.54	4.71	662.2	106.4	0.59
50	2	10	0.07	14.61	1.24	1287.2	0.37	104.64	83.28	2100.6	356.0	1.10
50	3	3	0.00	2.83	0.12	342.2	0.01	40.28	32.22	1029.6	358.6	0.83
50	3	10	0.01	9.99	0.31	1024.4	0.43	252.87	216.55	2737.0	1170.6	0.58
50	4	3	0.00	3.39	0.09	363.3	0.10	103.69	85.97	1904.3	1114.8	0.19
50	4	10	0.00	15.78	0.67	1107.6	0.04	826.00	607.45	4527.4	2806.4	0.18
100	2	3	0.00	37.27	0.44	836.0	0.00	50.84	8.29	1234.6	82.4	0.26
100	2	10	0.00	129.81	3.61	2364.4	0.04	313.36	166.80	3622.8	621.6	0.30
100	3	3	0.00	37.86	0.49	964.8	0.22	187.13	131.05	2453.2	956.8	0.21
100	3	10	0.01	97.92	2.32	2150.0	0.14	825.81	640.31	5110.2	2119.6	0.27
100	4	3	0.00	29.50	0.55	904.4	0.00	349.09	199.20	3605.4	2096.8	0.19
100	4	10	0.01	151.07	6.97	2653.0	0.91	1222.33	822.33	8994.0	5352.7	0.00

Table 8.1: Test results

Test problem				Trip assignment method				Section assignment method					
# of trips	max # of sections per trip	# of shift types	area size	Int. gap (%)	CPU (s)	CPU MILP (s)	# of columns	Int. gap (%)	CPU (s)	CPU MILP (s)	# of columns	# of constr. added	reduction (%)
25	3	10	250 x 250	0.06	1.10	0.15	355.2	0.09	18.19	11.05	1221.8	520.8	3.09
25	3	10	100 x 100	0.11	50.82	1.88	4324.6	0.30	1095.08	498.71	8486.8	3268.4	0.17
25	3	10	400 x 400	0.00	0.25	0.00	80.8	0.01	22.68	19.89	376.4	140.0	3.43
50	3	3	250 x 250	0.00	2.83	0.12	342.2	0.01	40.28	32.22	1029.6	358.6	0.83
50	3	3	100 x 100	0.17	146.81	4.29	4352.4	0.22	1474.18	1161.48	9136.0	4091.4	0.00
50	3	3	400 x 400	0.00	0.44	0.02	99.4	0.01	4.36	1.57	315.6	49.4	1.19

Table 8.2: Area variation

Test problem				Trip assignment method				Section assignment method					
# of trips	max # of sections per trip	# of shift types	time window	Int. gap (%)	CPU (s)	CPU MILP (s)	# of columns	Int. gap (%)	CPU (s)	CPU MILP (s)	# of columns	# of constr. added	reduction (%)
25	3	10	0 - 4	0.06	0.95	0.15	355.2	0.09	7.14	11.05	1221.8	520.8	3.09
25	3	10	4 - 8	0.01	8.84	0.26	1387.6	0.54	215.81	597.16	4695.4	2775.2	0.22
25	3	10	8 - 12	0.33	14.14	1.68	1658.8	0.48	467.01	472.55	5340.6	2989.8	0.00
50	3	3	0 - 4	0.00	2.72	0.12	342.2	0.01	8.06	32.22	1029.6	358.6	0.83
50	3	3	4 - 8	0.15	14.17	1.66	852.2	1.07	58.82	538.84	3049.6	1784.4	0.00
50	3	3	8 - 12	0.14	31.23	2.13	1456.7	0.43	250.67	603.33	4840.0	2915.0	0.00

Table 8.3: Time window variation

This improvement decreases as the instance gets larger, in terms of the number of trips and sections to assign. One explanation for this could be that if there are many trips to assign, already a very large number of combinations between trips is possible, so the possibility to assign individual sections adds less compared to a case with fewer trips. Also, in larger instances it happens more often that a solution to the integer version of the master problem cannot be found within the time allowed for this, which means the solution value obtained by the trip assignment method is not improved. The results also show that the gap between the LP solution value and the value of the solution to the integer problem is in most cases close to zero percent, and hardly ever exceeds 1%.

The application of the section assignment method evidently leads to better solutions. However, the computation time required by this approach is significantly higher compared to the trip assignment method. This can, to some extent, be explained by the fact that the trip assignment method is almost entirely incorporated into the section assignment method, as explained in chapter 6. Therefore, all computation time required to generate assignments with individual sections and to solve the RLP in between will add to the difference in computation time between both methods. However, the results show that this difference becomes considerably larger as the problem instance and therefore the number of generated columns and added constraints gets larger, up to the point were the computation time required by the section assignment method is sometimes more than twenty times higher. The major cause for this difference is the time consumed by the solver to obtain a solution for the MILP, at the very end of the procedure. The percentage of total CPU time that is consumed by the solver for obtaining an integer solution for the master problem varies between 30% for small problems and 90% for large problems. As can be seen in table 8.1, the time needed for obtaining an integer solution for the master problem is obviously very high when both the number of generated columns and the number of added constraints are high. The number of constraints that are added to the problem depends on the number of conflicting columns, which in turn depends mainly on the number of sections per trip and the number of resource shift types.

With respect to the variations in area size, the results in table 8.2 show that when the area is very small, much more columns are added to the problem. This is not surprising, as in this case the trips will be located within shorter distance of each other, meaning that more combinations will be available. As a result, the number of constraints that are added to the problem is much higher, which together with the high number of columns has a dramatic effect on computation times. Nevertheless, it appears from the results that the average reduction in costs of applying the section assignment method rather than the trip assignment method, is much lower. In the fifth instance, this is partly explained by the fact that in 60% of the runs, a final solution to the integer problem could not be found, due to a lack of memory. Besides this, the small average improvement can likely be explained by the fact that if all trips are located within a small area, more options are available for keeping distance to a minimum.

When the area is large, the opposite is the case. Computation time is lower, because less assignments can be created, and the average cost reduction by the section assignment method is higher. The latter can be explained by the fact that less distance between trips or sections have to be traveled when the area is larger. If, for example, a trip starts at the one end of the area and finished at the other side, splitting this trip into sections prevents resources from having to travel across the entire area. The larger this area is, the larger the advantage will be.

The variations in the length of the time windows show that when the time windows are larger, the reduction yielded by the application of the section assignment method gets significantly smaller. One reason for this is that in these cases, the fraction of runs in which a final solution to the MILP problem could not be found ranges between 20 and 40 percent. As time windows get larger, individual sections can be planned within a larger period of time. Therefore, it happens more often that a section in a particular assignment starts before its preceding section in another assignment has finished, meaning that a constraint has to be added to the master problem. The fact that the average number of constraints is much higher in the instances with larger time windows is in line with this explanation. The large number of constraints not only heavily increases computation time and the chance that no integer solution will be found in time, it also means that less combinations of columns are possible and therefore less improvement can be made by the section assignment method.

8.5 Conclusion and recommendations

From the test results described and discussed above, it can be concluded that both methods can be successfully applied to assign either entire trips or individual sections to resource shifts. The section assignment method often finds a better solution compared to the trip assignment method, but needs significantly more computation time for this. The bigger the problem instance is, in terms of number of trips, sections per trip and resource shift types, the smaller the difference between the solution values obtained by both methods, but the bigger the difference in computation times.

Although the section assignment method yields better solutions than the trip assignment method, its application might sometimes be impractical in practice due to the accompanying computation times. It is therefore recommended to make a well-considered choice of the method to use in a particular situation, based on the specifications of the case. If either the problem is not too large or sufficient time is available, it is advised to use the section assignment method. However, the trip assignment method might be favored if the problem is large, or if the functionality is used to obtain a rough estimate, for instance regarding the number of resource shifts needed, which might sometimes be the case in practice. The test results also indicate that assigning individual sections rather than full trips is more likely to result in significant cost improvements when the area is large, and time windows are narrow.

Furthermore, all test results prove very clearly that the number of resource shift types, or the number of unique resource shifts, has a great impact on the performance of both methods, as could be expected. This corresponds to the conclusion drawn in chapter 7. We therefore stress once again that the number of resource shifts should be kept to a minimum.

Finally, the test results show that the total number of columns generated in both procedures easily become very large. These columns all become part of the master problem, and therefore add to the computation time needed to solve this problem, which might especially get very large for the integer version. It is therefore recommended for further research to explore the possibilities of adding a procedure for the removal of columns. This has already been proven to be an effective method for improving computation time without compromising solution quality (Larsen, 2004).

Chapter 9

Conclusion

The aim of the research conducted in this thesis is to obtain a solution procedure for the Resource Assignment Problem (RAP). This problem is regularly encountered within ORTEC TD, a software package for transportation planning. It involves the assignment of a number of execution trips, which are essentially series of transportation tasks characterized by a start and end location and time window, to a number of resource shifts. A resource shift represents the availability of a certain combination of resources, which can be drivers, trucks and trailers. The assignment of trips to resource shifts is complicated by the fact that all trips have to be assigned, and that the number of resource shifts can be limited. Furthermore, the resource combination must be able to move from the end location of the one trip to the start of the next one in time, and rather complicated driving time legislation applies.

A second version of the resource assignment problem results from ORTEC's desire to be able to assign individual parts of a trip, which are referred to as sections. Assigning individual sections brings along the restriction that each section that is assigned individually should not start before the preceding section is completed. The initial problem and this variation to the problem are referred to as the trip assignment problem and the section assignment problem, respectively. The main contribution of this research to the trip assignment problem is a review of related literature to base the solution method on, a description of the developed method, and the application of this method to a representative real-life case, in order to examine its applicability and performance. Furthermore, an extension to this method is developed in order to deal with the section assignment problem. Both methods are implemented in MATLAB and applied to simulated cases, such that their performance can be compared.

The resource assignment problem as discussed in this thesis is not treated as such in combinatorial optimization literature. Nevertheless, it has several similarities with a number of well-studied problems, especially the vehicle routing problem, the vehicle scheduling problem and the crew scheduling problem. From literature on these problems, it is concluded that column generation is the most widely used approach for this kind of problems. It proves capable of dealing with large problems in acceptable time. Furthermore, it can handle complex restrictions, such as driving time legislation, by incorporating them in the subproblem.

The method for solving the trip assignment problem is a general column generation approach. The subproblem involves the creation of assignments, which in each iteration is solved for each unique resource shift. At the end of each iteration, a solver is used to solve the linear relaxation of the master problem, in which assignments are selected such that all trips are covered and total costs are minimized. At the end of the procedure, a solver is applied to the integer version of the master problem. We extended this method such that it can deal with the section assignment problem. After assignments with full trips are generated, the set of columns is extended by generating assignments containing individual sections. Since a section can only start after the preceding sections has ended, after each iteration constraints are added to prevent columns that cannot be in the solution at the same time due to this restriction from being selected both.

To test the implementation of the trip assignment method in ORTEC, a test case is deducted from one of ORTEC's client, which will be using the resource assignment functionality for planning distribution rounds. On all three instances within this case, the trip assignment method proved capable of assigning all trips to resource shifts, and within acceptable time. As expected, computational times are significantly affected by the number of resource shift types, as assignments are created for each unique resource shift. It is therefore recommended to keep the number of unique resource shifts to a minimum. In the tested instances, the method closely approached or even matched the lower bound obtained by solving the linear relaxation.

Because the section assignment method could not yet be implemented in ORTEC TD, both methods are implemented in MATLAB and applied to simulated instances. The improvement in total costs obtained by the section assignment method, compared to the trip assignment method, averages around 0.8% overall. The average improvement in smaller instances is significantly higher (around 3%), but approaches 0% in the largest cases. However, in all instances the sections assignment method requires considerably more computation time.

In the larger instances, the master problem dealt with by the section assignment method grows to considerable size, such that the time needed to solve its integer version at the end of the algorithm consumes a considerable amount of time. As a result, it occurs several times that either an integer solution is not found before the time limit of fifteen minutes is reached, or the procedure terminates because of a lack of memory. In these cases, the section assignment method fails to provide an improvement over the trip assignment method. It is advised to consider adding a column deleting procedure to the method, such that the needed computation time can be reduced.

The results show that the assignment of individual sections rather than full trips is especially advantageous in instances where less combinations can be made, for example when the number of trips is small, area size is large or time windows are narrow. In these instances, splitting trips into sections adds more flexibility. Considering both this finding as well as the higher computation times, it is concluded that the section assignment method is the most useful in cases that are not too large in terms of the number of sections and resource shift types, or when ample time is available. The trip assignment method however might be favored when available time is limited, the instance is very large or only estimates regarding for instance the number of shifts to use are desired.

Appendix A

Dependencies

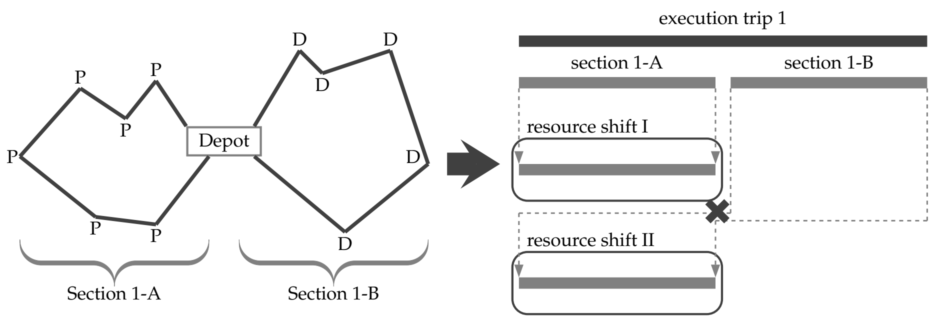


Figure A.1: Dependency between sections in the same execution trip

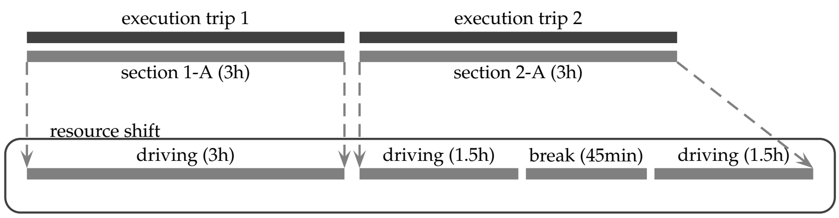
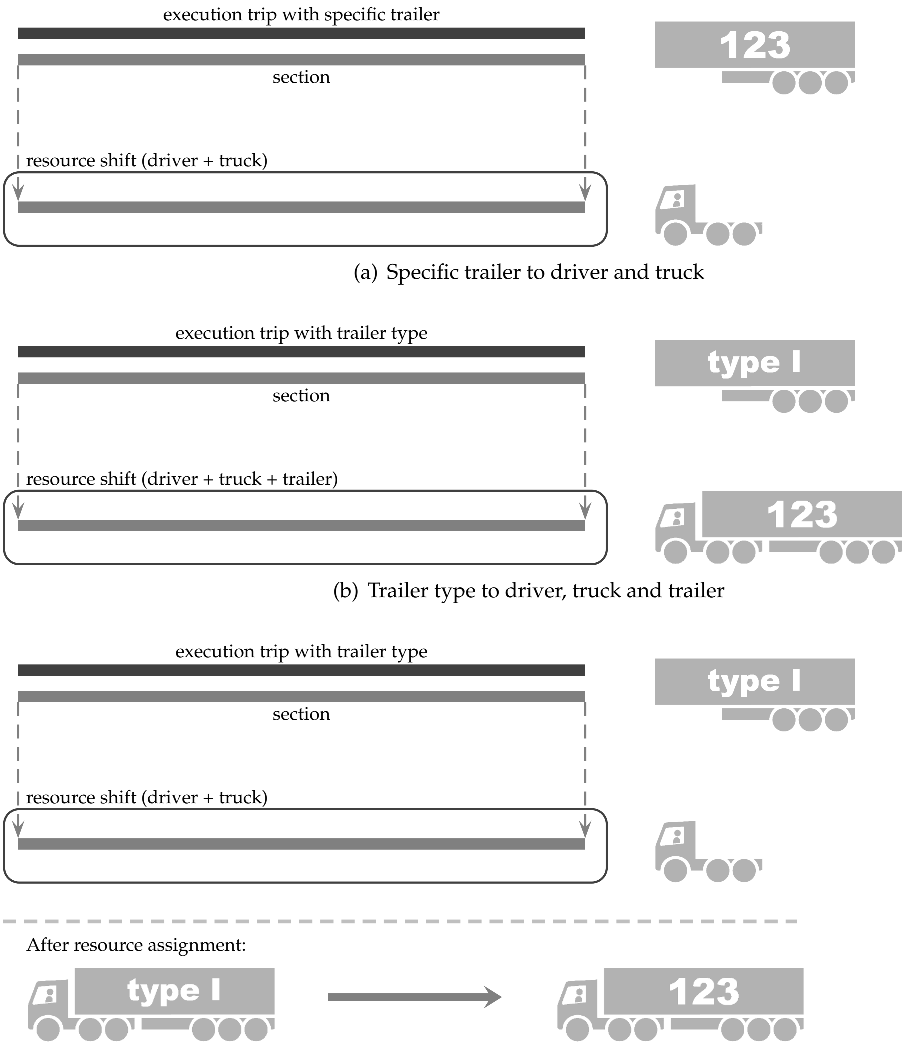


Figure A.2: Dependency between sections in the same resource shift

Appendix B

Assignment diagrams

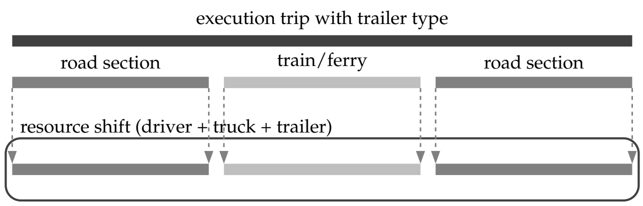


(a) Specific trailer to driver and truck

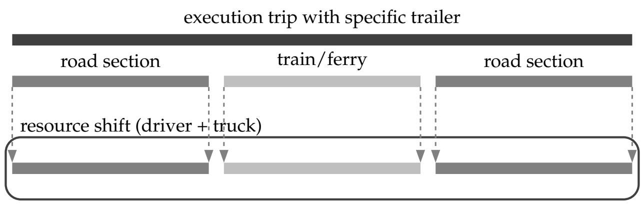
(b) Trailer type to driver, truck and trailer

(c) Trailer type to driver and truck

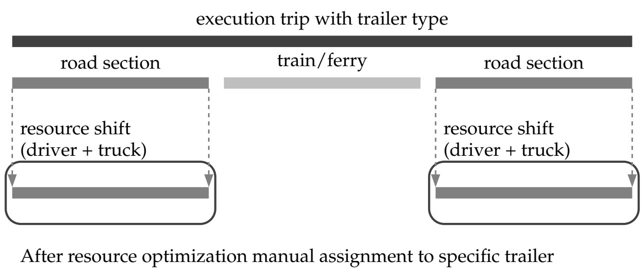
Figure B.1: Subscenarios for assignment of sections



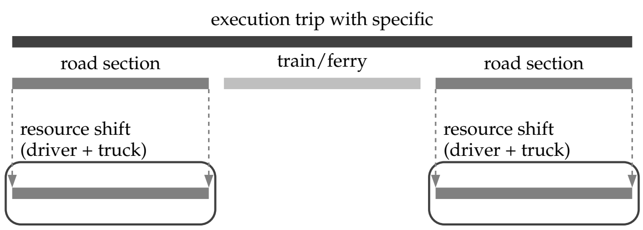
(a) Accompanied trailer type



(b) Accompanied specific trailer

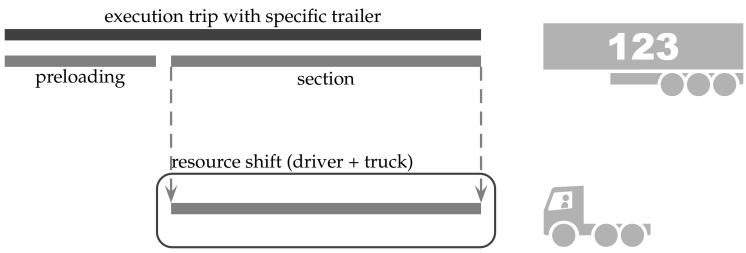


(c) Unaccompanied trailer type

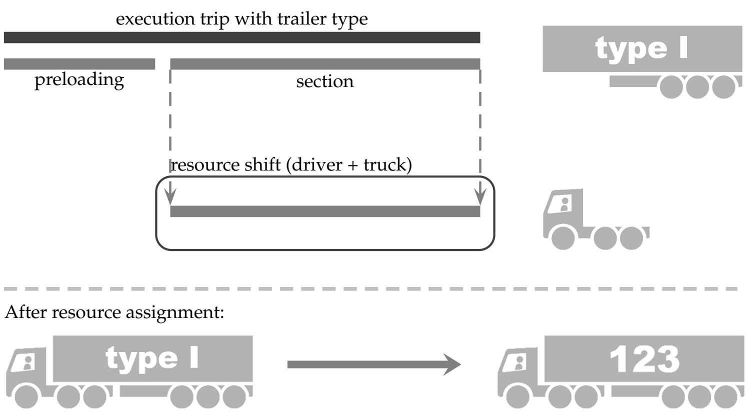


(d) Unaccompanied specific trailer

Figure B.2: Subscenarios for assignment of intermodal sections



(a) Preloading with trailer type



(b) Preloading with specific trailer

Figure B.3: Subscenarios for assignment of preloading sections

Bibliography

Archetti, C., & Savelsbergh, M. (2009). The trip scheduling problem. *Transportation Science*, *43*(4), 417-431.

Baita, F., Pesenti, R., Ukovich, W., & Favaretto, D. (2000). A comparison of different solution approaches to the vehicle scheduling problem in a practical case. *Computers & Operations Research*, *27*(13), 1249 - 1269.

Bertossi, A. A., Carraresi, P., & Gallo, G. (1987). On some matching problems arising in vehicle scheduling models. *Networks*, *17*(3), 271 - 281.

Bodin, L., Golden, B., Assad, A., & Ball, M. (1983). Routing and scheduling of vehicles and crews: The state of the art. *Computers & Operations Research*, *10*(2), 63-212.

Borndörfer, R., Schelten, U., Schlechte, T., & Weider, S. (2005). A column generation approach to airline crew scheduling. In H.-D. Haasis, H. Kopfer, & J. Schönberger (Eds.), *Operations research proceedings 2005, selected papers of the annual international conference of the german operations research society (GOR)* (p. 343-348). Berlin: Springer-Verlag.

Bunte, S., & Kliewer, N. (2010). An overview on vehicle scheduling models. *Public Transport*, *1*(4), 299-317.

Carraresi, P., & Gallo, G. (1984). Network models for vehicle and crew scheduling. *European Journal of Operational Research*, *16*(2), 139 - 151.

Dantzig, G. B., & Ramser, R. H. (1959). The Truck Dispatching Problem. *Management Science*, *6*.

De Groot, S. W., & Huisman, D. (2008). Vehicle and crew scheduling: solving large real-world instances with an integrated approach. In M. Hickman, P. Mirchandani, & S. Voß (Eds.), *Computer-aided systems in public transport, lecture notes in economics and mathematical systems* (p. 43-56). Berlin: Springer-Verlag.

Desaulniers, G. (2007). Managing large fixed costs in vehicle routing and crew scheduling problems solved by column generation. *Computers & Operations Research*, *34*(4), 1221-1239.

Desaulniers, G., Lavigne, J., & Soumis, F. (1998). Multi-depot vehicle scheduling problems with time windows and waiting costs. *European Journal of Operational Research*, *111*(3), 479 - 494.

European Union. (2002). Directive 2002/15/ec of the european parliament and of the council of 11 march 2002 on the organisation of the working time of persons performing mobile road transport activities. *Official Journal of the European Union*, *L 080*. (23.03.2002)

European Union. (2006). Regulation (ec) no 561/2006 of the european parliament and of the council of 15 march 2006 on the harmonisation of certain social legislation relating to road transport and amending council regulations (eec) no 3821/85 and (ec) no 2135/98 and repealing council regulation (eec) no 3820/85. *Official Journal of the European Union*, *L 102/1*. (11.04.2006)

Fischetti, M., Martello, S., & Toth, P. (1987). The fixed job schedule problem with spread-time constraints. *Operations Research*, *35*(6), 849–858.

Fischetti, M., Martello, S., & Toth, P. (1989). The fixed job schedule problem with working-time constraints. *Operations Research*, *37*(3), 395–403.

Freling, R., Huisman, D., & Wagelmans, A. P. M. (2003). Models and algorithms for integration of vehicle and crew scheduling. *Journal of Scheduling*, *6*(1), 63–85.

Freling, R., Wagelmans, A. P. M., & Paixão, J. M. P. (1999). An overview of models and techniques for integrating vehicle and crew scheduling. In N. H. M. Wilson (Ed.), *Computer-aided transit scheduling* (pp. 441–460). Berlin: Springer-Verlag.

Freling, R., Wagelmans, A. P. M., & Paixão, J. M. P. (2001). Models and algorithms for single-depot vehicle scheduling. *Transportation Science*, *35*(2), 165–180.

Goel, A. (2009). Vehicle scheduling and routing with drivers' working hours. *Transportation Science*, *43*(1), 17–26.

Goel, A. (2010). Truck driver scheduling in the European Union. *Transportation Science*, *44*(4), 429-441.

Gromicho, J., Van Hoorn, J. J., Kok, A. L., & Schutten, J. M. J. (2009). *Restricted dynamic programming: a flexible framework for solving realistic* VRP*s* (No. 266). Enschede: Beta Research School for Operations Management and Logistics, University of Twente.

Haghani, A., & Banihashemi, M. (2002). Heuristic approaches for solving large-scale bus transit vehicle scheduling problem with route time constraints. *Transportation Research Part A: Policy and Practice*, *36*(4), 309-333.

Huisman, D. (2004). *Integrated and dynamic vehicle and crew scheduling*. PhD Thesis, Erasmus University, Rotterdam.

Huisman, D., Freling, R., & Wagelmans, A. P. M. (2005). Multiple-Depot Integrated Vehicle and Crew Scheduling. *Transportation Science*, *39*(4), 491-502.

Kok, A. L., Meyer, C. M., Kopfer, H., & Schutten, J. M. J. (2009). *Dynamic programming algorithm for the vehicle routing problem with time windows and ec social legislation* (No. 270). Enschede: Beta Research School for Operations Management and Logistics, University of Twente.

Larsen, J. (2004). Refinements of the column generation process for the vehicle routing problem with time windows. *Journal of Systems Science and Systems Engineering*, *13*(3), 326-341.

Laurent, B., Guihaire, V., & Hao, J.-K. (2005). A heuristic solution for a driver-vehicle scheduling problem. In H.-D. Haasis, H. Kopfer, & J. Schönberger (Eds.), *Operations research proceedings 2005, selected papers of the annual international conference of the german operations research society (GOR)* (p. 703-708). Berlin: Springer-Verlag.

Laurent, B., & Hao, J.-K. (2009). Iterated local search for the multiple depot vehicle scheduling problem. *Computers & Industrial Engineering*, *57*(1), 277 - 286.

Li, H., & Lim, A. (2003). Local search with annealing-like restarts to solve the VRPTW. *European Journal of Operational Research*, *150*(1), 115-127.

Mercier, A., Cordeau, J.-F., & Soumisa, F. (2005). A computational study of Benders decomposition for the integrated aircraft routing and crew scheduling problem. *Computers & Operations Research*, *32*(6), 1451–1476.

Oukil, A., Amor, H. B., Desrosiers, J., & Gueddari, H. E. (2007). Stabilized column generation for highly degenerate multiple-depot vehicle scheduling problems. *Computers & Operations Research*, *34*(3), 817-834.

Papadakos, N. (2009). Integrated airline scheduling. *Computers & Operations Research*, *36*(1), 176–195.

Pepin, A.-S., Desaulniers, G., Hertz, A., & Huisman, D. (2009). A comparison of five heuristics for the multiple depot vehicle scheduling problem. *Journal of Scheduling*, *12*, 17-30.

Qureshi, A. G., Taniguchi, E., & Yamada, T. (2009). An exact solution approach for vehicle routing and scheduling problems with soft time windows. *Transportation Research Part E: Logistics and Transportation Review*, *45*(6), 960 - 977.

Ribeiro, C. C., & Soumis, F. (1994). A Column Generation Approach to the Multiple-Depot Vehicle Scheduling Problem. *Operations Research*, *42*(1), 41-52.

Xu, H., Chen, Z.-L., Rajagopal, S., & Arunapuram, S. (2003). Solving a practical pickup and delivery problem. *Transportation Science*, *37*(3), 347–364.

Zäpfel, G., & Bögl, M. (2008, June). Multi-period vehicle routing and crew scheduling with outsourcing options. *International Journal of Production Economics*, *113*(2), 980-996.