MATHIJS VAN ZON

Cost Allocation in Collaborative Transportation

Cost Allocation in Collaborative Transportation

Cost Allocation in Collaborative Transportation

Kostenverdeling in coöperatief transport

Thesis

to obtain the degree of Doctor from the
Erasmus University Rotterdam
by command of the
Rector Magnificus

Prof.dr. A.L. Bredenoord

and in accordance with the decision of the Doctorate Board.

The public defence shall be held on

Friday 12 November 2021 at 10:30 hours

by

Mathijs Antonius van Zon
born in Delft, The Netherlands

Erasmus University Rotterdam

Doctoral committee

Promotor:	Prof.dr. A.P.M. Wagelmans
Other members:	Prof.dr. D. Huisman
	Prof.dr. C. Archetti
	Prof.dr. P. Borm
Copromotors:	Dr. R. Spliet
	Dr. W. van den Heuvel

Erasmus Research Institute of Management - ERIM
The joint research institute of the Rotterdam School of Management (RSM)
and the Erasmus School of Economics (ESE) at the Erasmus University Rotterdam
Internet: www.erim.eur.nl

ERIM Electronic Series Portal: repub.eur.nl

ERIM PhD Series in Research in Management, 530
ERIM reference number: EPS-2021-530-LIS
ISBN 978-90-5892-612-8

Cover image: Rutger Wijnhoven
Cover design: PanArt, www.panart.nl
Print: OBT bv, www.obt.eu

This publication (cover and interior) is printed by on FSC® paper Magno Satin MC.

Acknowledgements

First, I want to thank my promotor Albert Wagelmans and my supervisors Remy Spliet and Wilco van den Heuvel for the opportunity to develop myself as a researcher. Albert, I want to thank you for enabling me to go on a research visit and to present my work at many conferences. Remy and Wilco, your support was invaluable. I want to thank you for guiding and assisting me throughout the past four years. Together, you taught me a lot and shaped me as a researcher. Your differing expertises and viewpoints complemented each other and pushed our research beyond our ever growing expectations. I am really proud of what we accomplished together! Remy, I also want to thank you for mentoring me personally. You helped me enormously in transitioning from studying round the clock towards a sustainable work-life balance.

Next, I want to thank Guy Desaulniers for inviting me on a research visit to Montréal. During my time in Montréal I learned a lot, both academically and personally. Guy, it was inspiring to experience your unwavering passion for tackling optimisation problems in new and innovative ways. I really appreciate that you invited Jørgen Skålness, Anda and myself to visit Mont Tremblant together. Also, I would like to thank those that were at GERAD back then, for giving me warm welcome and for teaching me the card game *jeu de tarot*, and later utterly destroying me at it.

Moreover, I want to thank Prof.dr. Dennis Huisman, Prof.dr. Claudia Archetti and Prof.dr. Peter Borm for taking the time to review my thesis and being part of my inner doctoral committee. Furthermore, I want to thank Prof.dr. Margaretha Gansterer and Prof.dr. Rob Zuidwijk for completing the opposition. I want to thank the entire doctoral committee for their time and effort, I greatly appreciate it!

During my time as a PhD candidate, I had a lot of support from my fellow vehicle routers: Kevin Dalmeijer, Thomas Visser and Ymro Hoogendoorn. You all inspired me with your dedication and countless insights. Kevin, your support during the early stages of my PhD was invaluable. You taught me everything I need to know about life as a PhD candidate. Also, you introduced me to boardgames, which to this day I still

enjoy very much. Thomas, I want to thank you for our many discussions on a wide range of topics. Especially our talks about heuristic approaches for vehicle routing problems turned out to be very fruitful. Ymro, I really enjoyed our brainstorming sessions and your many facts on seemingly random topics. I also want to thank my other colleagues for the great time we shared in the past four years, both at the campus and at conferences. Thank you Rolf van Lieshout, Naut Bulten, Rowan Hoogervorst, Rutger Kerkkamp, Thomas Breugem, Weina Ma, Nemanja Milovanovic, Lisanne van Rijn, Utku Karaca, Rommert Dekker, Dennis Huisman, Twan Dollevoet, Paul Bouman, Evelot Westerink-Duijzer, Marieke Musegaas and Noa den Hertog. A special thanks is in place for my paranymphs Joey Visser and Ymro Hoogendoorn for their support, Rutger Wijnhoven for creating the cover of the thesis, and Rutger Kerkkamp for creating and sharing the template that was used for this thesis.

I would also like to thank Arjan Cornelissen. He personally renewed my passion for vehicle routing by challenging me to participate in the *Travelling Santa* challenge by *Reaktor*. This invitation resulted in a month long programming frenzy where we continuously exchanged the top spot by sharing our ideas and improving our algorithms. Building on this experience, Arjan and I formed a team with Anda and Erik Meulman to compete in the *Google Hashcode* competition. Together, we had a great time preparing for this competition, and to our surprise, managed to qualify for the world finals. I am looking forward to compete again with all of you in the foreseeable future!

The past four years have not all been about academic challenges and optimisation. I want to thank my friends for distracting me with many other activities. In particular, I want to thank Luke Prananta for introducing me to bouldering and, Joey Visser and Erik Buijing for our many long gaming sessions.

I also want to thank both my family and the families of Anda for their amazing support! A special thanks goes to my father Kees, my mother Esther, my sister Manon, and my grandparents Koos, Ria, André and Jeanne. For as long as I can remember they have shown unwavering support towards my goals and ambitions. They are the reason that I have been able to get to where I am today. I am eternally grateful to have (had) them in my life. Last but not least, I want to thank Anda Knol. Her support and affection has kept me going during my time as a PhD candidate, and will keep me going in the future. Thank you Anda!

Table of contents

Chapter 1

Introduction

In this thesis, we consider horizontal collaborations between logistic service providers. For example, consider three neighbouring stores that are supplied by three different logistic service providers. In this case, each store is visited by a different vehicle. The logistic service providers can collaborate by consolidating the deliveries at the depot of one of the logistic providers, and combining their delivery routes. If the demand permits, only a single vehicle is required to supply the three stores. Such interactions are at the core of the collaborations considered in this thesis.

The remainder of this chapter is structured as follows. First, we provide a general overview of collaborative transportation and the corresponding cost allocation problem. This part of the introduction is an adaptation of van Zon et al. (2021a) and van Zon and Desaulniers (2021). Thereafter, we present an outline of this thesis and clarify the content and contributions of Chapters 2-4.

Collaborative transportation among logistic service providers has the potential to greatly increase the efficiency of their transportation networks. By collaborating, the logistic service providers can significantly reduce their cost (Guajardo and Rönnqvist, 2016), increase their service levels, and improve their market share (Gansterer and Hartl, 2018). Moreover, such collaborations are known to provide numerous other benefits including reductions in noise pollution, road congestion, and greenhouse gas emissions as shown by Ballot and Fontane (2010) and Pérez-Bernabeu et al. (2014).

In a horizontal collaboration, it is assumed that the logistic service providers exchange some or all of their customers. Gansterer and Hartl (2018) identify three major literature streams in horizontal collaborations: centralised collaborative planning, decentralised collaborative planning without auctions, and auction-based decentral-

ised planning. In a centralised collaborative planning, the logistic serive providers offer all their requests up for exchange. As a consequence, the joint profit can be maximised. On the other hand, in decentralised collaborative plannings, each logistic service provider can offer only a subset of their customers to the other logistic service providers, typically, those which are expensive to serve. Then, the customers are assigned to other logistics service providers either by an allocation mechanism (see, e.g., Özener et al., 2011; Wang and Kopfer, 2014) or a combinatorial auction (see, e.g., Krajewska and Kopfer, 2006; Gansterer and Hartl, 2016). In this thesis, we study centralised horizontal collaborations.

1.1 Cost allocations in collaborative transportation

We assume that the logistic service providers pool all of their requests and vehicles and, thereafter, collectively plan their operations. The benefit of such a centralised collaboration is that the overall cost can be minimised, or equivalently, the overall profit can be maximised. However, it is not clear how the overall cost or profit should be allocated among the participating logistic service providers. Cost or profit allocation is often studied using cooperative game theory, see Guajardo and Rönnqvist (2016) for a review on cost allocation in collaborative transportation.

In cooperative game theory, it is common to refer to the logistic service providers as players, and to refer to a group of players as a coalition. A cooperative game is characterised by a set of players, known as the grand-coalition, and a cost function or profit function which maps each coalition to a specific cost or profit, respectively. The goal is to allocate the grand-coalition cost or profit to the players. In particular, we look for a so-called core allocation, as is common in the literature. A core allocation ensures that each coalition is not better off if they decide not to cooperate in the grand-coalition, and that the overall cost or profit is precisely allocated to the players. The set of all such allocations is known as the core.

Among the first studied collaborative transportation games is the travelling salesman game (TSG), in which each player corresponds to a single delivery location rather than a logistic service provider. Moreover, all deliveries are made on a single distribution route. The cost of a coalition is defined as the optimal objective value of a travelling salesman problem (TSP). See Applegate et al. (2006) for more on the TSP. The goal of the TSG is to allocate the cost of the distribution route among the customers that correspond to the delivery locations. The TSG was originally proposed by Fishburn and Pollak (1983), and the core of the TSG has been studied

by Dror (1990) and Potters et al. (1992).

The vehicle routing game (VRG) is a generalisation of the TSG, where each player is assumed to have a certain demand which has to be satisfied by a fleet of vehicles with finite capacity. For the VRG, the cost of a coalition is given by the optimal objective value of a capacitated vehicle routing problem (CVRP). See Toth and Vigo (2014) for more on the CVRP. Similar to the TSG, the goal of the VRG is to allocate the cost of all routes over the customers served on these routes. The problem of determining a core allocation for the VRG has been studied by Göthe-Lundgren et al. (1996) and Engevall et al. (2004). All applied cooperative games studied in this thesis are generalisations of the VRG.

1.2 Outline and contributions

Chapters 2-4 of this thesis can be read individually as the required concepts and notation are (re)introduced. Finally, in Chapter 5, we give an overview of the main results of this thesis. In the following, we present an outline and clarify the content and contributions of the remaining chapters.

Chapter 2: The Joint Network Vehicle Routing Game

Chapter 2 is based on van Zon et al. (2021a). In this chapter, we propose a generalisation of the VRG, which we call the joint network vehicle routing game (JNVRG). In the JNVRG, we consider a collaboration among logistic service providers, each with multiple customers. In contrast to the VRG, we aim to allocate the cost among groups of customers, i.e., a logistic service provider. We model the corresponding cost allocation problem as a cooperative game, the JNRVG, for which we try to determine a core allocation.

In order to determine a core allocation, an exponential number of constraints in the number of players have to be considered. Here, each constraint requires a vehicle routing problem to be solved, which is NP-hard and requires exponential running time in general. For the VRG, among other games, row generation approaches have been proposed to determine core allocations for games with a large number of players (see, e.g. Göthe-Lundgren et al., 1996; Engevall et al., 2004; Drechsel and Kimms, 2010). We also propose a row generation algorithm to determine a core allocation for the JNVRG. Here, we encounter a row generation subproblem which we model as a new variant of a vehicle routing problem with profits. We propose a branch-price-and-cut algorithm to determine solutions for the row generation subproblem. Moreover, we

propose two main acceleration strategies for the row generation algorithm. First, we generate rows by relaxing the row generation subproblem, exploiting the tight LP bound for our formulation of the row generation subproblem. Secondly, we propose to also solve the row generation subproblem heuristically and to only solve it to optimality when the heuristic fails.

We demonstrate the effectiveness of the proposed row generation algorithm and the acceleration strategies by means of numerical experiments for both the VRG and the JNVRG. While, Göthe-Lundgren et al. (1996) and Engevall et al. (2004) were able to determine core allocations for VRG instances with up to 25 players, we are able to determine core allocations, or show that none exist, for VRG instances with up to 53 players. Moreover, we can determine core allocations, or show that none exist, for JNVRG instances with up to 79 customers divided over 15 players.

Chapter 3: The Joint Network Vehicle Routing Game with Optional Customers

Chapter 3 is based on van Zon and Desaulniers (2021). In this chapter, we propose a generalisation of the JNVRG. We observed that in the JNVRG, the logistic service providers are likely to have residual capacity as a consequence of consolidating demands and combining delivery routes. This residual capacity could be used by the logistic service providers to generate additional profits by collectively serving sets of new customers, which are initially not served by any of the logistic service providers due to capacity restrictions. The additional profit generated by serving these new customers gives an additional incentive for the logistic service providers to collaborate. For example, a company with multiple branches may require to be supplied by the same logistic service provider. Therefore, to acquire the branches of this company as customers (all together), a logistic service provider must have sufficient capacity or must collaborate with one or several other logistic service providers.

We extend the setting of Chapter 2 by assuming that there exist multiple clusters of optional customers (each composed, e.g., of branches of a company) which are not assigned to one of the players. A coalition obtains a reward for visiting all customers of a cluster. In this case, we cannot simply define the profit of a coalition, as it is not trivial to determine how the clusters of optional customers are allocated between the coalition and the remaining players. We consider both a best-case profit as well as a worst-case profit for each coalition. In the best case, we assume that a coalition can freely select the clusters of optional customers it is willing to serve, thereby obtaining the maximum profit. On the other hand, in the worst case, we assume that

the remaining players form a coalition and select the clusters of optional customers which benefit them the most. In this case, the original coalition can only select the clusters of optional customers which have not been selected by the remaining players. In this manner, we aim to provide an accurate representation of the profit which the coalition may attain.

Our goal is to try to determine an allocation in the core as defined by the best-case profit. However, if this core is empty, we try to determine an allocation in the core as defined by the worst-case profit, or show that the core as defined by the worst-case profit is empty. To this end, we propose a heuristic row generation algorithm to determine allocations in the cores as defined by the best-case and the worst-case profit, or to show that both cores are empty. The corresponding row generation subproblem is a generalisation of a vehicle routing problem with profits. We propose a branch-price-and-cut algorithm for the subproblem. To this end, we adapt several well-known acceleration strategies for the problem. We show that our heuristic algorithm scales well with respect to the number of logistic service providers considered and provides allocations similar to those obtained by enumerating all best-case and worst-case profits of each coalition.

Chapter 4: The Effect of Algorithmic Capabilities on Cooperative Games

In Chapter 4, we study centralised horizontal collaborations from a more general perspective. Specifically, we assume that the operations of each player can be described by the same optimisation problem. The players can collaborate by merging their smaller problem instances into larger problem instances. In the previous chapters, we assumed that a general algorithm was available to determine solution to these problem instances. The assumption of the general availability of an exact or heuristic algorithm is also often (implicitly) made in the literature (see, e.g., Göthe-Lundgren et al., 1996; Engevall et al., 2004; Krajewska et al., 2008; Drechsel and Kimms, 2010; Zakharov and Shchegryaev, 2015; Dai and Chen, 2015). In this chapter, we remove the assumption of a common algorithm.

We propose a framework for cooperative games in which we allow each player to have their own solution algorithm. Using these algorithms, the players can collectively determine the cost of a coalition, i.e., by solving problem instances individually and executing the best found solution or by combining their algorithms into a shared algorithm. We refer to a game in which the cost of a coalition is determined by the algorithms of the players of the coalition as a algorithm quality induced game (AQI

game). In a AQI game, the algorithmic capabilities of each player is explicitly taken into account when allocating the grand-coalition cost to the players. We believe that this is a more accurate representation of collaborations in practice. Moreover, this framework also allows for the inclusion of players with a good algorithm but without any other contributions to the shared operations, i.e., consultants.

We study cost allocation for AQI games when considering several well-known allocation concepts and methods. We show analytically how the profit allocated to a player in a core allocation is influenced by improving an individual solution algorithm. Moreover, we prove tight bounds for the profit each consulting player may be allocated in a core allocation, as well as a bound for the number of consulting players that can be allocated a profit in a core allocation. Finally, we present numerical results based on 580 800 instances of the AQI game based on the pickup-and-delivery problem, a variant of the vehicle routing problem. By considering all pairs of AQI games which only differ in the algorithm of a single player, we observe that for all allocation methods, players are allocated less on average after they improved their algorithm. Moreover, we find that players in general benefit from the inclusion of a consultant except for players which have a high quality algorithm where the allocated cost may increase.

Chapter 2

The Joint Network Vehicle Routing Game

Abstract

Collaborative transportation can significantly reduce transportation costs as well as greenhouse gas emissions. However, allocating the cost to the collaborating companies remains difficult. We consider the cost-allocation problem which arises when companies, each with multiple delivery locations, collaborate by consolidating demand and combining delivery routes. We model the corresponding cost-allocation problem as a cooperative game: the joint network vehicle routing game (JNVRG). We propose a row generation algorithm to either determine a core allocation for the JNVRG, or show that no such allocation exists. In this approach, we encounter a row generation subproblem which we model as a new variant of a vehicle routing problem with profits. Moreover, we propose two main acceleration strategies for the row generation algorithm. First, we generate rows by relaxing the row generation subproblem, exploiting the tight LP bounds for our formulation of the row generation subproblem. Secondly, we initially solve the row generation subproblem heuristically and only solve it to optimality when the heuristic fails. We demonstrate the effectiveness of the proposed row generation algorithm and the acceleration strategies by means of numerical experiments for both the JNVRG as well as the traditional vehicle routing game, which is a special case of the JNVRG. We create instances based on benchmark instances of the capacitated vehicle routing problem from the literature. We are able to either determine a core allocation, or show that no core allocation exists, for instances ranging from 5 companies with a total of 79 delivery

locations to 53 companies with a total of 53 delivery locations.

This chapter is based on van Zon et al. (2021a).

2.1 Introduction

Consider a company which operates a distribution network with multiple delivery locations. The delivery locations are served by a fleet of vehicles from a central depot. Often, multiple of these distribution networks are located within the same area. For example, consider several retailers operating within the same city centres. Close proximity of their delivery locations provides companies with the opportunity to collaborate. Such collaborations can consist of consolidating demands and combining delivery routes, effectively joining their distribution networks. In over ten case studies with real life data (Guajardo and Rönnqvist, 2016), collaborative transportation provides potential cost savings ranging from 4% (Lehoux et al., 2011) to 46% (Engevall et al., 2004). Moreover, Ballot and Fontane (2010) report a reduction in CO_2 emissions by at least 25% as a consequence of collaborative transportation.

Despite the potential benefits, many collaboration opportunities are left unused in practice. One reason is that the remaining cost after collaboration still needs to be divided among the companies. Reaching an agreement can be difficult as the concerned distribution networks are likely to have different characteristics and objectives. For example, each company may have a different number of delivery locations and the demand volumes may differ as well. Nonetheless, all routing costs need to be allocated to the companies. In this chapter, we study the cost allocation problem which arises from collaborative transportation of multiple companies, each having multiple delivery locations. Here, we assume that each player has the objective to minimise its cost. We focus on the computational aspect of allocating the total cost to each of the players.

In various settings, cost allocation problems are modelled as cooperative games, see Guajardo and Rönnqvist (2016) for an overview of cooperative games in transportation. In cooperative game terminology, it is common to refer to the companies as players, and to refer to a group of players as a coalition. A cooperative game is characterised by a set of players, known as the grand-coalition, and a cost function which maps each coalition to a specific cost. Our goal is to allocate the grand-coalition cost to the players. In particular, we look for a so-called core allocation, as is common in the literature. A core allocation ensures that each coalition is worse off

if they decide not to cooperate in the grand-coalition. The set of all such allocations is known as the core.

Among the first studied collaborative transportation games is the travelling salesman game (TSG), in which each player corresponds to a single delivery location and all deliveries are made on a single distribution route. The cost of a coalition is defined as the optimal objective value of a travelling salesman problem (TSP). See Applegate et al. (2006) for more on the TSP. The TSG was originally proposed by Fishburn and Pollak (1983), and the core of the TSG has been studied by Dror (1990) and Potters et al. (1992). For a case study on the TSG, we refer the reader to Engevall et al. (1998), who determine core allocations for a single instance consisting of 5 players.

The vehicle routing game (VRG) is a generalisation of the TSG, where each player is assumed to have a certain demand which has to be satisfied by a fleet of vehicles with finite capacity. For the VRG, the cost of a coalition is given by the optimal objective value of a capacitated vehicle routing problem (CVRP). See Toth and Vigo (2014) for more on the CVRP. The problem of determining a core allocation for the VRG has been studied by Göthe-Lundgren et al. (1996), who present numerical experiments for instances with at most 25 players. Furthermore, Engevall et al. (2004) consider the VRG with a heterogeneous fleet and determine the Nucleolus (Schmeidler, 1969) for a single instance consisting of 21 players.

In order to determine a core allocation, the core is often modelled as a linear programming problem consisting of an exponential number of constraints, one for each coalition. For a small number of players, we might include all constraints and determine the cost for each coalition. Clearly, if the number of players is large or if computing the cost of a coalition is computationally expensive, this approach is no longer tractable. For example, in the TSG, a TSP has to be considered for each coalition, while the TSP is an NP-hard problem. Similarly, in the VRG, exponentially many CVRPs are considered. Note that computationally the CVRP seems more difficult than the TSP, since benchmark instances for the CVRP with up to 360 customers can be solved to optimality by state-of-the-art algorithms (Pecin et al., 2017b), whereas benchmark instances with up to 85900 customers have been solved to optimality for the TSP (Applegate et al., 2006).

Interestingly, Göthe-Lundgren et al. (1996) show that in order to describe the core of the VRG, only coalitions which can be served on a single distribution route have to be considered. As a result, the cost of any relevant coalition is computed by solving a TSP, and no coalitions need to be considered for which the cost is computed by solving a CVRP. However, as the number of coalitions remains large,

Göthe-Lundgren et al. (1996) propose a row generation algorithm to determine a core allocation. See Drechsel and Kimms (2010) for more information on row generation in the context of cooperative games. In their approach, Göthe-Lundgren et al. (1996) assume that the core is non-empty. As a consequence, the row generation subproblem they encounter simplifies, and can be modelled as a travelling salesman problem with profits. Engevall et al. (2004) propose a row generation algorithm to determine the Nucleolus of a VRG with heterogeneous fleet, even if the core is empty. They show that the row generation subproblem in this case can be modelled as a vehicle routing problem with profits. However, the authors do not propose a method to solve their row generation subproblem. Instead, they illustrate the effectiveness of the row generation technique on an instance with 21 players by enumerating the CVRP cost of over 2 million coalitions, for which they report a runtime of roughly 12 weeks.

In contrast to the TSG and VRG where each player corresponds to a single delivery location, in many real-life applications, each company has multiple delivery locations. In the following, we consider cooperative transportation games in which each player corresponds to multiple delivery locations. Krajewska et al. (2008) and Dai and Chen (2015) consider a cooperative game based on a pickup and delivery problem. Here, each player has a set of pickup and delivery requests. For each request, an order has to be collected from a pickup location before it is delivered. Krajewska et al. (2008) define the cost as the objective value of applying a heuristic to the pickup-and-delivery problem. The authors enumerate the cost for each coalition, and determine the Shapley value (Shapley, 1953) for instances consisting of 5 players, each with up to 100 delivery requests. For each allocation they also determine whether it is in the core of their game. As opposed to Krajewska et al. (2008), Dai and Chen (2015) define the cost as the optimal objective value to the pickup-and-delivery problem. They propose a row generation algorithm to determine a core allocation for the game or show that the core is empty. To this end, they formulate both the pickup-and-delivery problem and the row generation subproblem as MIPs, and use a general purpose MIP solver to solve these problems. Dai and Chen (2015) consider instances consisting of at most 5 players and 25 requests. For half of the instances consisting of 5 players, it took them over 12 hours to compute a core allocation.

Also Dahlberg et al. (2018) study a cooperative game in which the costs are defined as the optimal objective value of an underlying routing problem. They consider a collaboration among multiple freight forwarders and a single municipality, all considered as players. In their case, the cost of a coalition depends on whether

the municipality is included, in which case the freight forwarders are subject to additional costs, but gain access to an extra depot. The authors consider instances with 4 players, including the municipality, and a total of 15 delivery locations. They determine several allocations, including core allocations, by enumerating the cost of each coalition.

To the best of our knowledge, the setting of Zakharov and Shchegryaev (2015) is closest to our work. The authors consider a setting in which each player is responsible for multiple delivery locations. Given a coalition, they determine the cost by heuristically solving a CVRP consisting of the delivery locations of the players in the coalition. Based on these heuristic costs, they determine a core allocation for an instance consisting of 4 players and 200 delivery locations by enumerating all coalitions.

Contrary to the heuristically determined cost of Zakharov and Shchegryaev (2015), we define the cost of a coalition as the optimal objective value to a CVRP over the delivery locations of the players in the coalition. We call the corresponding cooperative game the joint network vehicle routing game (JNVRG). Observe that although the underlying optimisation problem of the JNVRG is the same as in Zakharov and Shchegryaev (2015), the games are not the same as the structure of the characteristic function is different. First, a core allocation of a game for which the underlying optimisation problem is solved heuristically, instead of to optimally, may not be stable in practice. A coalition might be able to improve on the heuristic solution which could correspond to a cost that is lower than what is allocated. Secondly, when the cost of a coalition is determined heuristically, it is not obvious how to find a core allocation in a cooperative game without enumerating all coalitions. Clearly, enumeration is not tractable for cooperative games with a large number of players. Alternatively, for a game in which the cost function is defined as the optimal objective value of an underlying optimisation problem, row generation can be applied. The trade-off is that more players can be considered while the number of delivery locations will be limited.

Our main contribution is as follows. We propose a row generation algorithm to either determine a core allocation for instances of the JNVRG, or show that the core is empty. In our solution approach, we encounter a row generation subproblem which we model as a generalisation of a vehicle routing problem with profits, which to our best knowledge has not been studied before. Furthermore, we propose two main acceleration strategies for the row generation algorithm. First, we introduce what we call coarse row generation, a technique where we exploit the tight bounds of the

set-partitioning formulation of the CVRP. In particular, we relax the row generation subproblem to accelerate the identification of violated core constraints. Secondly, we solve the row generation subproblem heuristically before using an exact algorithm in order to limit the number of times the row generation subproblem has to be solved to optimality. Furthermore, we demonstrate the effectiveness of the proposed row generation algorithm and acceleration strategies by means of numerical experiments. We are able to determine a core allocation or show that the core is empty for instances ranging from 5 players with a total of 79 delivery locations to 53 players with a total of 53 delivery locations within 4 hours of computation time. Note that the latter instance corresponds to a VRG, demonstrating that the proposed acceleration strategies are also effective for this game.

The remainder of this chapter is structured as follows. In Section 2.2 we introduce the JNVRG. Next, in Section 2.3 we propose a row generation algorithm to determine a core allocation in an efficient manner. We present several existing acceleration strategies for solving the row generation subproblem in Section 2.4. In Section 2.5 we propose new acceleration strategies for the row generation algorithm. Our computational results are presented and discussed in Section 2.6. Finally, we provide some concluding remarks and suggestions for future research in Section 2.7.

2.2 The Joint Network Vehicle Routing Game

In order to formally introduce the JNVRG, we first define the CVRP. Consider a complete directed graph $G = (V, A)$. The vertex set V is defined as $\{0\} \cup V'$, where 0 represents the depot and V' represents the set of all delivery locations, also referred to as customers. With each arc $(v, w) \in A$, a travel cost $c_{vw} \geq 0$ is associated, which we assume to adhere to the triangle inequality. Furthermore, each customer $v \in V'$ has a demand $d_v > 0$. An unlimited number of vehicles with capacity Q is available at the depot to satisfy the demand by visiting customers along routes. A route is defined as a simple cycle, starting and ending at the depot, such that the total demand of the customers covered by the route does not exceed the capacity. We assume that $d_v \leq Q$ for all $v \in V'$. The CVRP is the problem of constructing routes in such a way that the total cost is minimised and every customer is visited exactly once.

Next, we define the JNVRG. Let $N = \{1, 2, \ldots, n\}$ be the set of players. The set V' is the combined set of the customers of these players. In particular, $V'(i) \subset V'$ represents the customers of player i where $\bigcup_{i \in N} V'(i) = V'$ and $V'(i) \cap V'(j) = \emptyset$

for $i, j \in N$ with $i \neq j$. The cost $C(S)$ of coalition $S \subseteq N$ is the minimal cost of the CVRP on the subgraph $G(S)$ induced by $V'(S) \cup \{0\}$, where $V'(S) = \bigcup_{i \in S} V'(i)$ represents all customers of the players in coalition S. Let $C(\emptyset) = 0$. We define the JNVRG as an n-person cooperative game $\langle N, C \rangle$, where the aim is to allocate the cost $C(N)$ to the players.

2.2.1 Characterisation of the core

As is common in cooperative game theory, we search for an allocation in the core. In order to define the core, we denote the cost allocated to player i as y_i and the cost allocated to the coalition $S \subseteq N$ as $y(S) = \sum_{i \in S} y_i$. A cost allocation $y \in \mathbb{R}^n$ is said to be efficient if $y(N) = C(N)$. Furthermore, a cost allocation is said to be rational if it satisfies the rationality constraints $y(S) \leq C(S)$ for all coalitions $S \subset N$. The rationality constraints ensure that no coalition has an incentive not to cooperate. The set of rational and efficient cost allocations is known as the core (Gillies, 1959) and is described as

$$\text{Core}(\langle N, C \rangle) = \{y \in \mathbb{R}^n : y(S) \leq C(S) \; \forall S \subset N, y(N) = C(N)\}.$$

If no rational and efficient cost allocation exists, the core is said to be empty. The core of the JNVRG is related to the core of the VRG, since the VRG is a special case of the JNVRG in which each player has exactly one customer. Göthe-Lundgren et al. (1996) show that for the VRG there exist instances with an empty core. Because the VRG is a special case of the JNVRG, we conclude that there also exist instances of the JNVRG for which the core is empty.

One way to determine whether the core of a specific JNVRG instance is non-empty, is by applying the Bondareva-Shapley theorem of Bondareva (1963) and Shapley (1967). This theorem states that the core of a game is non-empty if and only if the game is balanced. A game is said to be balanced if and only if there exists no mapping $x_S : 2^{|N|} \rightarrow \mathbb{R}_+^{|N|}$ such that $\sum_{S \subseteq N : i \in S} x_S = 1$ for all $i \in N$ and $\sum_{S \subseteq N} x_S C(S) < C(N)$. For convenience, we define balancedness by means of an optimisation problem, similar to the optimisation problem used in the analysis of Göthe-Lundgren et al. (1996). A game is said to be balanced if and only if $C(N)$ is smaller than or equal to the optimal objective value C^* of the following minimisation

problem:

$$C^* = \min \quad \sum_{S \subset N} x_S C(S) \tag{2.1}$$

$$\text{s.t.} \quad \sum_{S \subset N : i \in S} x_S = 1 \qquad \forall\, i \in N, \tag{2.2}$$

$$x_S \geq 0 \qquad \forall\, S \subset N, \tag{2.3}$$

where x_S is a non-negative decision variable representing the fraction of coalition S in the mapping. Moreover, note that Constraints (2.2) and (2.3) ensure that x_S is a valid mapping from the set of all coalitions to $\mathbb{R}_+^{|N|}$ such that each player is covered exactly once.

Göthe-Lundgren et al. (1996) implicitly use the Bondareva-Shapley theorem to determine whether the core of a VRG is non-empty. To do so, they make use of so-called feasible coalitions, which they define as coalitions of which the delivery locations can be visited on a single distribution route. They show that the cost of a non-feasible coalition is precisely the sum of the costs of the feasible coalitions corresponding to each route which is part of an optimal solution for the CVRP of the non-feasible coalition. As a result, only feasible coalitions have to be considered to describe the core, and all other coalitions can be disregarded. In this case, the computation of the cost of a coalition reduces to solving a TSP instead of a CVRP.

When only considering feasible coalitions, (2.1)-(2.3) reduces to the LP-relaxation of the set-partitioning formulation of the CVRP of the grand-coalition, in which the variable x_S corresponds to a single route. Moreover, it follows that $C^* = C(N)$ if and only if the LP-relaxation of the set-partitioning formulation of the CVRP of the grand-coalition has an integer optimal solution. Interestingly, it follows that by solving the LP-relaxation of the set-partitioning formulation of a CVRP, one can determine whether the core of the VRG is non-empty.

A similar result does not hold for the JNVRG, as coalitions which are served by multiple vehicles cannot be disregarded. This is because in contrast to the VRG, the customers of a single player might be visited on multiple routes. Therefore, we cannot determine a core allocation by only considering TSPs for a limited set of coalitions, instead we consider a CVRP for all coalitions.

These results also provide insight into how often one can expect a VRG core to be empty. It is well known that many CVRP instances do not have an integer solution to the LP-relaxation of the set-partitioning formulation. Hence, we can expect the core of the VRG to be empty quite often. This expectation does not directly apply

to the JNVRG. To see this, construct an instance of the JNVRG by combining multiple players of the VRG into single players of the JNVRG. This means that the delivery locations of a single player in the JNVRG are in fact the combined delivery locations of multiple players of the VRG. Observe that the rationality constraints of this instance of the JNVRG are a subset of the rationality constraints of the VRG. Hence, the JNVRG is a restricted game with respect to the VRG, see Faigle (1989) for more on restricted games. As a result, the core of a VRG is contained in the core of a corresponding JNVRG. Put differently, if the core of the VRG is empty, this does not imply that the core of a corresponding JNVRG is empty, although the reverse implication does hold.

2.2.2 Cost allocation

Well-known allocations are the Nucleolus, the Lorenz allocation (Arin, 2003) and the allocation given by the Equal Profit Method (EPM) of Frisk et al. (2010). All of these allocations can be found by solving linear programming problems that include all rationality constraints. In this chapter, we use the EPM allocation for illustrative purposes, although our proposed approach could easily be applied to find other LP-based core allocations as well. In particular, it can be applied to allocations such as the Nucleolus and the Lorenz allocation, but not to allocations such as the Shapley value.

An EPM allocation is a core allocation which minimises the maximum difference in allocated cost to each player, relative to their individual cost. The EPM allocation y can be determined by solving the linear programming problem:

$$\min \quad \theta \tag{2.4}$$

$$\text{s.t.} \quad \frac{y_i}{C(\{i\})} - \frac{y_j}{C(\{j\})} \leq \theta \qquad \forall\, i, j \in N, \tag{2.5}$$

$$y(S) \leq C(S) \qquad \forall S \subset N, \tag{2.6}$$

$$y(N) = C(N), \tag{2.7}$$

$$y_i \geq 0 \qquad i \in V', \tag{2.8}$$

$$\theta \in \mathbb{R}. \tag{2.9}$$

In an optimal solution to (2.4)-(2.9), the maximum difference in relative costs is given by the decision variable θ as enforced by Constraints (2.5). The rationality constraints (2.6) and efficiency constraint (2.7) ensure that the allocation is in the core. Like Frisk et al. (2010), we include Constraints (2.8). However, note that due

to the monotonicity of the costs of each coalition, these constraints are implied by the rationality constraints. Finally, Constraint (2.9) specifies the domain of the decision variable θ.

2.3 A row generation algorithm

We determine an EPM allocation for the JNRVG by solving (2.4)-(2.9). The inclusion of the rationality constraints (2.6) poses two challenges. First, there are exponentially many of these constraints. Second, to determine the cost of a coalition, a CVRP has to be solved, which is known to be NP-hard (Lenstra and Rinnooy Kan, 1981). To alleviate both challenges, we propose a row generation method in which these constraints are first omitted and then iteratively generated if they are violated. This approach is similar to the approach presented by Göthe-Lundgren et al. (1996) and Engevall et al. (2004), but differs in the row generation subproblem.

We first present the general idea of the row generation algorithm which we use to determine an EPM allocation. An initial cost allocation y is obtained by solving (2.4)-(2.9) with (2.6) only for the singleton coalitions. Note that the non-negativity constraints (2.8) are no longer implied in the absence of some rationality constraints, which is why we explicitly include them. Next, a row generation subproblem is solved to determine whether there exists a coalition for which the rationality constraint is violated by the current allocation y. If a violated rationality constraint is identified, the constraint is added to the relaxed problem. This procedure is repeated until no more violated constraints can be identified, indicating that the solution is optimal, or the problem becomes infeasible, indicating that the core is empty.

In the remainder of this section we present our row generation algorithm in more detail. A description of the row generation subproblem is given in Section 2.3.1. In Section 2.3.2, we propose a branch-and-price algorithm to solve the row generation subproblem.

2.3.1 The row generation subproblem

Consider a cost allocation y. A violated rationality constraint is characterised by a coalition $S \subset N$ such that $y(S) > C(S)$. This gives rise to the following row generation subproblem:

$$\mathrm{SP}(y) = \max_{S \subset N} \{y(S) - C(S)\}. \tag{2.10}$$

If we find that the optimal objective value $\mathrm{SP}(y) > 0$ a violated rationality constraint has been identified, otherwise the current allocation is an EPM allocation. Note that the routing costs $C(S)$ of the potentially violating coalitions have not been determined, and are also part of the row generation subproblem.

The row generation subproblem can be seen as a generalisation of a capacitated profitable tour problem (CPTP) as studied by Archetti et al. (2013). In the CPTP not all customers need to be visited, whereas a prize is collected for customers which are visited. The goal of the CPTP is to maximise the collected prizes minus the routing costs. Our row generation subproblem is a generalisation of the CPTP such that a prize y_i is only collected if all customers $V'(i)$ of player i are visited. We call this modified problem the grouped capacitated profitable tour problem (GCPTP).

Next, we introduce an integer programming formulation for the GCPTP. Let R denote the set of routes in G. Define the coefficient a_r^v as the number of times customer v is included in route r, hence $a_r^v = 0$ if customer v is not included in r and $a_r^v = 1$ if customer v is included exactly once in r. We define the binary decision variable z_i such that $z_i = 1$ if player i is included in the selected coalition, and 0 otherwise, where we denote the selected coalition as $S(z) = \{i \in N : z_i = 1\}$. Furthermore, let $i(v)$ denote the player $i \in N$ such that $v \in V(i)$. Define the binary decision variable x_r such that $x_r = 1$ if route r is selected, and 0 otherwise. Let c_r be the cost of route r which is given by the sum of the arc costs on route r. We can formulate the row generation subproblem as follows:

$$\max \quad \sum_{i \in N} y_i z_i - \sum_{r \in R} c_r x_r \tag{2.11}$$

$$\text{s.t.} \quad \sum_{r \in R} a_r^v x_r \geq z_{i(v)} \qquad \forall\, v \in V', \tag{2.12}$$

$$x_r \in \{0, 1\} \qquad \forall\, r \in R, \tag{2.13}$$

$$z_i \in \{0, 1\} \qquad \forall\, i \in N. \tag{2.14}$$

Objective (2.11) represents the amount by which the rationality constraint of the selected coalition $S(z)$ is violated. Constraints (2.12) enforce that all delivery customers of a player are visited at least once if the player is included in the selected coalition. Finally, Constraints (2.13) and (2.14) specify the domains of the decision variables.

Consider an optimal solution (x^*, z^*). In this case, x^* is an optimal solution to the CVRP of coalition $S(z^*)$. Hence, after solving the row generation subproblem, the optimal routing cost of the most violated coalition has also been determined, and

the corresponding rationality constraint $y(S(z^*)) \leq C(S(z^*))$ can directly be added to the relaxed problem without having to solve an additional CVRP.

2.3.2 Solving the row generation subproblem

We propose a branch-and-price algorithm in order to determine an optimal solution to the row generation subproblem. In each node of the branch-and-bound tree we determine an upper bound for the row generation subproblem by solving the linear relaxation of (2.11)-(2.14), referred to as the master problem (MP). Since the number of route variables x_r is high, we apply column generation to solve the MP in each node. To this end, we define the restricted master problem (RMP) as the MP in which we restrict the set of variables. Then, we solve the RMP and search for any positive reduced cost variables by solving a pricing problem, which we introduce next. If such variables are found, they are added to the RMP and the process is repeated. If no more positive reduced cost variables exist, the solution to the RMP is also optimal for the MP and gives a valid upper bound for the row generation subproblem. Finally, if a solution is fractional, we branch on either fractional player variables or fractional arc flows.

2.3.2.1 The pricing problem

The pricing problem is the problem of identifying a positive reduced cost variable. In most CVRP literature employing column generation, the pricing problem is a minimisation problem to identify a negative reduced cost variable. Therefore, in an attempt to avoid confusion, we reverse the sign of the reduced cost and model the pricing problem as a minimisation problem as well, and search for negative reduced cost variables.

In order to guarantee optimality of the RMP, we must show that no routes with negative reduced cost exist. The reduced cost $\bar{c}_r$ of a route r can be expressed as the sum over the modified arc costs $\bar{c}_{vw}$:

$$\bar{c}_{vw} = c_{vw} - \frac{\mu_v + \mu_w}{2}, \quad v, w \in V' \cup \{0\} \tag{2.15}$$

where $\mu_v \geq 0$ is a dual variable associated with Constraints (2.12) and $\mu_0 = 0$. We view the pricing problem as an Elementary Shortest Path Problem with Capacity Constraints (ESPPCC) in the graph G with the modified arc cost. To efficiently solve the ESPPCC, Label-setting algorithms are often used. In the following we present a label-setting algorithm, which is similar to that of Martinelli et al. (2014).

A partial path in G starting at the depot and ending at vertex $v \in V' \cup \{0\}$ is represented by a label λ which is defined as follows:

$$\lambda = (v, C, D, U)\,. \tag{2.16}$$

Here, C is the cost of the partial path, the load D is the demand satisfied by the partial path and U is a binary vector consisting of entries U_w which indicate whether the partial path may be extended to vertex $w \in V \cup \{0\}$. As proposed by Feillet et al. (2004), the entry U_w is set to 1 if vertex w has already been visited by the partial path or if the capacity constraint does not allow vertex w to be visited. The extension λ' of a partial path $\lambda = (v, C, D, U)$ from vertex v to vertex w is determined as

$$\lambda' = (w, C + \bar{c}_{vw}, D + d_w, f(U))\,, \tag{2.17}$$

where the resources U are updated by a function $f(U) : \mathbb{B}^{|V'|} \to \mathbb{B}^{|V'|}$. For $w' \in V(N) \cup \{0\}$ such that $w' \neq w$, $f_{w'}(U)$ equals 1 if and only if $U_{w'} = 1$ or $D + d_w + d_{w'} > Q$. Furthermore, $f_w(U)$ equals 1.

Next, we present a dominance rule to limit the number of partial paths which have to be considered. Consider two distinct labels $\lambda = (v, C, D, U)$ and $\lambda' = (v, C', D', U')$, both with end vertex v. If every completion of the partial path corresponding to λ' leads to a worse solution than the same completion of λ we say that λ dominates λ', in which case label λ' can be disregarded. Label λ dominates λ' if the following conditions hold:

$$C \leq C', \tag{2.18}$$

$$D \leq D', \tag{2.19}$$

$$U \leq U'. \tag{2.20}$$

For efficient implementation, we make use of the fact that demand is integer in the benchmark instances that we use and keep labels in buckets $B(v, q)$ based on the end vertex $v \in V'$ and load $q \in \{0, \ldots, Q\}$. We store the labels sorted on reduced cost to reduce the number of times we have to verify criterion (2.18). Initially, label $(0, 0, 0, \mathbf{0})$ is added to bucket $B(0, 0)$, the remaining buckets are initialised as empty. Then, in increasing order of load all partial paths are extended to neighbouring nodes. After the extension of a label, the dominance rule is utilised to verify if the extended label dominates any of the existing labels with equal load q and end vertex v, if this is the case, the label is stored in bucket $B(v, q)$. Any existing labels which are

dominated in the process are removed from the bucket. If no labels are dominated by the extended label, we verify if any of the existing labels dominate the extended label. If none of the existing labels dominate the extended label, it is added to the corresponding bucket $B(v, q)$.

Each negative reduced cost label included in a bucket associated to the depot corresponds to a negative reduced cost route. After the algorithm has been executed, the best ρ_E found negative reduced cost routes are added to the RMP.

2.3.2.2 Branching strategy

The optimal solution to the MP may be fractional. We propose a branching scheme to obtain an optimal integer solution. First, we branch on the sum of the variables z_i. Then, we branch on the most fractional z_i, i.e., the z_i which value is closest to $\frac{1}{2}$. If all z_i are integer, we branch on the number of vehicles used. Finally, we branch on the most fractional arc flow. Here we use the well-known result that integer arc flows also correspond to integer routes. When choosing a node to branch on, we consider the node with the highest upper bound.

2.4 Acceleration strategies for solving the subproblem

In the following, we discuss several acceleration strategies from the CVRP literature and apply them in our algorithm for the row generation subproblem which we model as a GCPTP. First, in Section 2.4.1 we describe ng-route relaxation (Baldacci et al., 2011) for the row generation subproblem in order to reduce the computational effort required to solve the pricing problem. Second, in Section 2.4.2 we describe decremental state space relaxation (Martinelli et al., 2014) to further accelerate the label-setting algorithm. In Section 2.4.3, we present a heuristic for the pricing problem. Finally, in Section 2.4.4 we describe the limited memory subset-row cuts (Pecin et al., 2017a) which we use to improve the LP bound of the MP.

2.4.1 ng-Routes

The formulation of the row generation subproblem is relaxed by allowing non-elementary routes, originally applied by Desrochers et al. (1992). Note that this does not alter the optimal objective value as it is never better to visit a customer multiple times. The benefit of allowing non-elementary routes is that it reduces the computational

effort required to solve the pricing problem. However, this comes at the expense of weaker LP bounds. We consider a subset of non-elementary routes known as ng-routes, which were introduced by Baldacci et al. (2011) and typically provide a good trade-off between weaker LP bounds and reduced computation times for the pricing problem. An ng-route is defined as a route which is allowed to visit a vertex v multiple times, if and only if, between the visits, at least one vertex v' is visited such that $v \notin \Pi_{v'}$, where $\Pi_{v'} \subseteq V'$ represents the ng-neighbourhood of vertex v'.

We implement ng-routes by altering the label-setting algorithm. We modify the definition of a label such that at vertex v we only keep track of whether the label can be extended towards the vertices in the ng-neighbourhood Π_v. It is assumed that a label can be extended to all vertices not in its neighbourhood, if this does not violate the capacity constraints. When extending a label from vertex v to vertex w we update the label according to (2.17) except for the binary vector U, which is updated according to $f'(U)$. For $w' \in V'$ such that $w' \neq w$, $f'_{w'}(U)$ equals 1 if and only if $D + d_w + d_{w'} > Q$ or $w' \in \Pi_w$ and $U_{w'} = 1$. Moreover, $f'_w(U)$ equals 1. Finally, we alter the dominance rule as follows. We say that a label $\lambda = (v, C, D, U)$ dominates another label $\lambda' = (v, C', D', U')$ if the following condition

$$U_w \leq U'_w \text{ for } w \in \Pi_v \tag{2.21}$$

holds along with conditions (2.18)-(2.19).

2.4.2 Decremental state space relaxation

To reduce the size of the ng-neighbourhoods, we apply decremental state space relaxation as proposed by Martinelli et al. (2014), which is demonstrated to be effective at improving the computational performance of label-setting algorithms. To this end, we execute the label-setting algorithm with auxiliary ng-neighbourhoods Γ_v instead of the real ng-neighbourhoods Π_v for each vertex $v \in V'$. Each time we solve a MP, the neighbourhood Γ_v is initialised as $\{v\}$ for all $v \in V'$. If the algorithm identifies a negative reduced cost route r, which is not an ng-route with respect to the neighbourhoods $\Pi_v = V'$, the neighbourhoods Γ_v are updated as follows. For each cycle in r which is not allowed, vertex v is added to the neighbourhoods of all vertices included in the cycle. In this manner, the cycle can no longer be generated in any subsequent iterations of the pricing problem. Then, the label-setting algorithm is restarted with the updated neighbourhoods.

2.4.3 Heuristic labelling

Before calling the label-setting algorithm, we solve the pricing problem heuristically. If one or multiple routes with negative reduced costs are found by the heuristic, the best ρ_H are added to the RMP. Otherwise, the exact label-setting algorithm is used. As a heuristic, we use a modified version of the label-setting algorithm, in which we only keep the label with the lowest reduced cost in each bucket not belonging to the depot in order to reduce the number of potential labels.

2.4.4 Limited memory subset row cuts

The bound of the RMP can be improved by including valid inequalities. An effective family of valid inequalities for the CVRP, which are also valid for our MP, are the limited memory subset-row cuts (LM-SRCs) as introduced by Pecin et al. (2017a). These cuts provide a good trade-off between computation time and strength of the cuts (Pecin et al., 2017b). They are an extension of the subset-row cuts (SRCs ; Jepsen et al., 2008) which we briefly explain first. Consider a fractional routing solution as visualised in Figure 2.1.

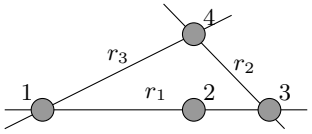

Figure 2.1: Example of a fractional routing solution.

Suppose routes $r_1 = (0, 1, 2, 3, 0), r_2 = (0, 3, 4, 0)$ and $r_3 = (0, 1, 4, 0)$ are all used $\frac{1}{2}$ times in an optimal solution to the RMP, where 0 denotes the depot. It is clear that these routes cannot be included together in any integral optimal solution to the MP, which is prevented by including SRCs. As is common, we limit ourselves to SRCs of size 3 with coefficient $\frac{1}{2}$. Given $W \subset V'$ of cardinality 3 and coefficient $\frac{1}{2}$, a SRC is formulated as

$$\sum_{r \in R} \left\lfloor \frac{1}{2} \sum_{v \in W} a_r^v \right\rfloor x_r \leq 1. \tag{2.22}$$

The labelling algorithm has to be modified to deal with additional dual variables from the inclusion of the SRC. An additional resource is added to a label for each included SRC and only a weaker form of dominance can be applied (Pecin et al., 2017b). To reduce the computational impact on the pricing algorithm, we consider the LM-SRC. Given $W \subset V'$ such that $|W| = 3$, a LM-SRC with coefficient $\frac{1}{2}$ can

be formulated as follows:

$$\sum_{r \in R} \alpha(W, M, r) x_r \leq 1, \tag{2.23}$$

where $M \subseteq V'$ is the memory set and the coefficient α is determined in the following iterative manner. Set $\alpha = 0$, consider the first customer v on route r, if $v \in W$ we increase α by $\frac{1}{2}$, else if $v \in V' \backslash M$ we set $\alpha = \lfloor \alpha \rfloor$, otherwise nothing happens. We repeat this process for each customer in r in order of appearance. Consider the previous example again. Suppose that $W = \{1, 3, 4\}$ and $M = \{1, 3, 4\}$, it holds that $\alpha(W, M, r_1) = 0$ and $\alpha(W, M, r_2) = \alpha(W, M, r_3) = 1$. In this case, the cut does not prohibit the fractional solution. However, setting $M = \{1, 2, 3, 4\}$ yields $\alpha(W, M, r_1) = \alpha(W, M, r_2) = \alpha(W, M, r_3) = 1$, which does cut off the fractional solution. By increasing the size of the memory, the LM-SRCs become stronger at the cost of increased computational effort.

To account for the inclusion of LM-SRCs, the label-setting algorithm is modified. Each label is extended to hold an additional list of resources S consisting of resources S_ω for each LM-SRC $\omega \in \Omega$, where Ω consists of all LM-SRC identified so far for which the current associated dual σ_ω is non-zero and the current customer of the label is included in the memory set of ω. Furthermore, when extending a label λ from vertex v to vertex w, we iterate over the resources in Ω. If w is included in the customer set of the corresponding LM-SRC, the resource S_ω is increased by $\frac{1}{2}$. Then, if the resource is equal to 1, the resource value is reduced to 0 and the dual is subtracted from the reduced cost of the label. Otherwise, if w is not included in the memory set of the corresponding LM-SRC, the resource is also set to 0. Finally, the dominance rule is modified (Pecin et al., 2017a). Consider two distinct labels $\lambda = (C, D, U, S)$ and $\lambda' = (C', D', U', S')$. Label λ dominates λ' if the following condition holds:

$$C \quad \leq \quad C' + \sum_{\omega \in \Omega : S_\omega > S'_\omega} \sigma_\omega, \tag{2.24}$$

along with conditions (2.19) and (2.21). We separate the LM-SRC by enumeration, once the MP has been solved to optimality. To this end, we consider all subsets $W \subset V'$ where $|W| = 3$, $\forall\ v, w \in W$, $v \in \Pi_w$ and $w \in \Pi_v$. Given a subset W, we generate a LM-SRC if the corresponding SRC is violated by at least 0.1. Then $M \subseteq V'$ is determined as follows. For each route included in the current optimal solution, all customers between the first and second visit to a customer in W are included in the memory. The same is done for any visited customer between the

third and fourth visit to a customer in W, and so on. For example, when considering Figure 2.1 once more, one can see that route r_1 requires customer 2 to be included in the memory.

2.5 Acceleration strategies for the row generation algorithm

In this section, we propose several acceleration strategies for the row generation algorithm. Note that these acceleration strategies can also be applied to determine a core allocation for the VRG. First, we present a relatively straightforward acceleration strategy after which we introduce the two main acceleration strategies.

We store all columns which are generated by the row generation subproblem algorithm in a column pool. Then, in each iteration of the row generation algorithm, the row generation subproblem is initialised with all columns in the column pool. In this manner the row generation subproblem is provided with a better initial solution at no additional computational cost.

Secondly, we propose an alternative row generation procedure. By relaxing the row generation subproblem to allow fractional route variables, a potentially violated rationality constraint is identified. Then, the cost of the identified coalition is determined by solving a CVRP. The aim of this method is to separate the identification of a violated constraint and the computation of the right-hand side value, such that both procedures can be accelerated separately. This procedure, which we call coarse row generation, is explained in detail in Section 2.5.1.

Thirdly, we propose to first generate rows heuristically to potentially reduce the number of calls to the exact algorithm for the row generation subproblem. This procedure is elaborated upon in Section 2.5.2.

Finally, in Section 2.5.3 we present an overview of the different variants of row generation algorithms we consider, which follow from combining coarse and heuristic row generation.

2.5.1 Coarse row generation

Separating rationality constraints by solving the row generation subproblem to integer optimality might be computationally intensive. Rather than modelling the vehicle routing cost exactly in the row generation subproblem, we propose to use a lower bound on the routing cost. We truncate the branching procedure, by no longer

branching on the arc flow, to obtain this lower bound. Effectively, we replace the routing cost $C(S)$ of the row generation subproblem by the lower bound $\underline{C(S)}$, where $\underline{C(S)}$ is given by the optimal objective value to the LP relaxation of the set-covering formulation for the CVRP including LM-SRCs. We call this row generation subproblem the coarse row generation subproblem, which can be formulated as follows:

$$\max_{S \subset N} \left(y(S) - \underline{C(S)} \right). \tag{2.25}$$

Note that $\max_{S \subset N}(y(S) - C(S)) \leq \max_{S \subset N} \left(y(S) - \underline{C(S)} \right)$. Hence, the coarse row generation subproblem yields an upper bound for the row generation subproblem. Given a cost allocation y, we call a coalition $S \subset N$ potentially violated if

$$\max_{S \subset N} \left(y(S) - \underline{C(S)} \right) > 0.$$

Observe that a potentially violated coalition $S \subset N$ does not necessarily correspond to a violated rationality constraint. In order to asses this we need to determine the cost $C(S)$ by solving a CVRP.

This approach provides the opportunity to accelerate the procedure of solving CVRPs by adding valid inequalities which we cannot add when solving the row generation subproblem. We add the well-known rounded capacity cuts. For a given subset of customers $W \subseteq V'$ such that $|W| \geq 2$, the rounded capacity cut is as follows:

$$\sum_{(v,w) \in A(N): v \in W, w \notin W} \left[\sum_{r \in R} b_r^{vw} x_r \right] \geq \left\lceil \frac{\sum_{w \in W} d_w}{Q} \right\rceil, \tag{2.26}$$

where b_r^{vw} denotes whether route r uses arc $(v, w) \in A$. This valid inequality puts a lower bound on the number of vehicles required to satisfy the demand of the customers in W. Note that, in contrast to the LM-SRCs, the structure of the pricing problem is not altered by the inclusion of the rounded capacity cuts.

We separate the rounded capacity cuts before separating the LM-SRC using the heuristic separation algorithms by Lysgaard et al. (2004). After adding cuts, we solve the RMP again to optimality. Furthermore, we initialise the algorithm for the CVRP with the routes used in the optimal solution to the relaxed row generation subproblem. As such, the root node has already been solved and cuts can immediately be separated.

When branching to solve the coarse row generation subproblem, we may come across a solution where the player variables z are integer with a non-negative objective

value, before finding the optimal solution to the coarse row generation subproblem. Rather than continuing to branch, we solve the corresponding CVRP and add the corresponding rationality constraint. If the constraint is not violated by the current solution, we continue branching. Otherwise, we solve the relaxed EPM problem to obtain a new allocation or show that the core is empty.

After we add a rationality constraint, we also add an anti-cycling constraint to the coarse row generation subproblem to prevent the algorithm from identifying the same constraint multiple times. This can occur due to the use of the lower bound for the routing cost. Given a coalition $S \subseteq N$, the anti-cycling constraint is as follows:

$$\sum_{i \in S} z_i - \sum_{i \in S \setminus N} z_i \leq |S| - 1. \tag{2.27}$$

Note that the anti-cycling constraints are also added for coalitions $S \subseteq N$ with $|S| = 1$ and the grand-coalition, which are included at initialisation.

2.5.2 Heuristic row generation

In order to heuristically solve the (coarse) row generation subproblem, we solve the pricing problem by only using the heuristic labelling algorithm as presented in Section 2.4.3. This leads to an upper bound on the routing cost, and in turn a lower bound on the (coarse) row generation subproblem. Hence, if a (potentially) violated rationality constraint is identified heuristically for the (coarse) row generation subproblem, it is guaranteed to also be (potentially) violated for the (coarse) row generation subproblem. However, as the heuristic gives a lower bound we cannot guarantee that no violated coalition exists if none can be identified by the heuristic. Hence, when the heuristic fails to find a violated coalition, we continue to solve the (coarse) row generation subproblem to optimality.

2.5.3 Variants of the row generation algorithm

The strategies described in Sections 2.5.1 and 2.5.2 can be applied and combined in multiple ways. First, we consider the basic row generation procedure as described in Sections 2.3 and 2.4, denoted by GEN_B. Note that this procedure does not include the acceleration methods proposed in Sections 2.5.1 and 2.5.2 but does include the column pool as mentioned at the start of Section 2.5. From now on, we refer to the row generation subproblem as the basic subproblem to be able to clearly distinguish it from the coarse row generation subproblem. We consider the row

generation procedure GEN_C, which is similar to GEN_B but only performs coarse row generation. Next, we consider the inclusion of heuristics as described in Section 2.5.2. $\text{GEN}_{CH\to BH\to B}$ denotes the procedure in which we first solve the coarse subproblem heuristically. If no violated rationality is identified, we solve the basic subproblem heuristically. If by doing so we cannot identify a violated rationality constraint, we solve the basic subproblem to optimality. Note that these steps are repeated at each iteration of the row generation algorithm. Finally, we consider the procedure denoted by $\text{GEN}_{CH\to BH\to C}$ which is similar to $\text{GEN}_{CH\to BH\to B}$ but where the coarse subproblem is solved to optimality instead of the basic subproblem.

We observed from preliminary experiments with a limited set of instances that out of all row generation algorithms including heuristics, $\text{GEN}_{CH\to BH\to B}$ and $\text{GEN}_{CH\to BH\to C}$ performed the best on average. Therefore, we only include these configurations. A summary of the methods used in our experiments can be found in Table 2.1. Note that all algorithms are guaranteed to give an optimal solution to (2.4)-(2.9) if the core is non-empty.

Table 2.1: Solution methods for the row generation subproblem.

Abbreviation	Description
GEN_B	The basic subproblem is solved to optimality.
GEN_C	The coarse subproblem is solved to optimality.
$\text{GEN}_{CH\to BH\to B}$	First, the coarse subproblem is solved heuristically, then the basic subproblem is solved heuristically, and finally the basic subproblem is solved to optimality.
$\text{GEN}_{CH\to BH\to C}$	First, the coarse subproblem is solved heuristically, then the basic subproblem is solved heuristically, and finally the coarse subproblem is solved to optimality.

2.6 Numerical results

The algorithms were implemented in C++. We used IBM ILOG CPLEX Optimiser 12.7.1 to solve all linear programming problems. The computations were conducted on a single core of an AMD Ryzen 7 2700X at 3.7 GHz with 16 GB of RAM.

The arc costs for the instances we consider are symmetric, so we branch on edge flows rather than on arc flows. After each iteration of the (heuristic) pricing problem the best $\rho_E = 10$ ($\rho_H = 20$) negative reduced cost routes are added to the RMP. Furthermore, whenever we solve a CVRP for coalition $S \subseteq N$ we use the same implementation as used for the basic subproblem by solving a GCPTP with $y_i = 0$ for every $i \in N$ and $z_i = 1$ for every $i \in S$ and $z_i = 0$ for $N \backslash S$. Moreover, we

now also include the rounded capacity cuts. For every instance and algorithm, a computation time limit of 4 hours is used. Moreover, for the sake of brevity, we say that an instance is settled if we have determined a core allocation or have shown that the core is empty.

2.6.1 Problem instances

We generate instances of the JNVRG based on CVRP benchmark instances from the literature. To this end, we consider the A-set of Augerat (1995) and the Solomon instances C101-100 and C201-100 as introduced by Solomon (1987). We select these instances to consider both non-clustered and clustered instances, which could influence the difficulty to determine a core allocation. We modify the instances as follows. We consider 5, 10 and 15 players for each of the instances. Given the number of players n we assign customer i to player $(i \bmod n) + 1$. For the Solomon instances, this creates clusters in which customers of different players are included.

From the Augerat A-set we create JNVRG instances including all customers. However, from each Solomon instance we generate multiple JNVRG instances that contain a subset of customers. For each Solomon instance, we create 5 instances of 20 customers, 3 instances of 33 customers and 2 instances of 50 customers, where we split the customers of the original instance in equal sets starting with the first customer. For example, when considering 33 customers, the first set consists of customers 1 through 33, whereas the second set consists of customers 34 through 66. Additionally, we create instances of the VRG. They are the same as the JNVRG instances, but modified such that each individual customer corresponds to a separate player.

2.6.2 Numerical results for the JNVRG

We introduce an additional algorithm ENUM to serve as a benchmark. This algorithm enumerates all coalitions, solves the corresponding CVRPs and then determines an EPM allocation by solving (2.4)-(2.9). Obviously, this method is not tractable when the number of players is high.

The results of our numerical experiments on the JNVRG instances are given in Tables 2.2 and 2.3 for the Augerat instances and Solomon instances, respectively. The first column provides the name of the instance. The column n provides the number of players and the column $|V'|$ provides the number of customers. The column C indicates whether the core is empty (E), non-empty (NE) or whether it is unknown

(?) because the instance was not settled by any of the algorithms within the time limit of 4 hours. For each of the algorithms the number of generated constraints is reported in the column m, where the number in between brackets represents the number of generated constraints which were actually violated, excluding the grand-coalition and singleton rationality constraints used at initialisation. The column T provides the total time spent in seconds to determine an EPM allocation, and the column T_{RG} represents the percentage of time which was spent on solving the row generation subproblem. For each instance, the fastest solution time has been made bold. Note that the Augerat instances A-n60-k09 to A-n80-k10 have been omitted from Table 2.2 as not many of these instances could be settled within a time limit of 4 hours. The results on these instances are presented in Table 2.6 in the Appendix and are not considered in the remainder of this section. Note that among these instances, the largest we could settle has 5 players and 79 customers.

Using the enumerative algorithm ENUM, 27 out of the 51 Augerat instances and 30 out of the 60 Solomon instances for the JNVRG were settled. By using this algorithm, most instances consisting of 5 players could be settled. However, only 21 out of 37 instances consisting of 10 players could be settled and only 2 instances consisting of 15 players could be settled within the time limit. This clearly demonstrates that enumerating all coalitions is not tractable even for a moderate number of players. With the different variants of the row generation algorithms we are able to settle significantly more instances. Using the basic row generation algorithm GEN_B we could settle a total of 82 instances, 25 more than with the enumerative algorithm, which demonstrates the effectiveness of the row generation procedure. The other variants of the algorithms GEN_C, $\text{GEN}_{CH\to BH\to B}$ and $\text{GEN}_{CH\to BH\to C}$ could settle 91, 98 and 95 instances out of the 111 instances, respectively. Note that the row generation algorithms are able to solve significantly more of both the 10 player and 15 player instances.

When comparing run times between two algorithms, we compare the average total time taken to settle all instances which have been settled using both algorithms. When considering the 5 player instances, the basic row generation algorithm was 18% slower than ENUM. On the other hand, the row generation algorithms GEN_C, $\text{GEN}_{CH\to BH\to B}$ and $\text{GEN}_{CH\to BH\to C}$ are 84%, 71% and 58% faster than ENUM, respectively. On the 10 player instances, the row generation algorithms are 40% to 810% faster than the enumerative algorithm. For all Augerat instances, GEN_C was 120% faster than GEN_B, which shows the effectiveness of the coarse row generation. Furthermore, $\text{GEN}_{CH\to BH\to B}$ was 221% faster whereas $\text{GEN}_{CH\to BH\to C}$ was

Table 2.2: Numerical results on the JNVRG instances of the Augerat A-set.

				ENUM	GEN_B			GEN_C			$\text{GEN}_{CH\to BH\to B}$			$\text{GEN}_{CH\to BH\to C}$		
Instance	n	$\lvert V\rvert$	C	T	m	T	T_{RG}	m	T	T_{RG}	m	T	T_{RG}	m	T	T_{RG}
A-n32-k05	5	31	NE	32	2 (2)	4469	100	2 (2)	35	74	2 (2)	25	62	2 (2)	**25**	62
A-n32-k05	10	31	NE	1309	5 (5)	1679	100	20 (5)	683	87	16 (5)	**226**	68	20 (5)	327	68
A-n32-k05	15	31	NE	-	15 (15)	-	100	99 (25)	10093	95	105 (20)	**2641**	81	109 (23)	3126	77
A-n33-k06	5	32	NE	58	0 (0)	29	59	1 (0)	27	55	1 (0)	23	47	1 (0)	**22**	46
A-n33-k06	10	32	NE	1758	5 (5)	1619	99	20 (8)	749	71	17 (8)	491	55	17 (8)	**465**	53
A-n33-k06	15	32	NE	-	13 (13)	2233	100	46 (16)	2414	90	49 (21)	**655**	70	56 (21)	870	79
A-n34-k05	5	33	NE	351	0 (0)	48	83	2 (0)	46	70	2 (0)	27	50	2 (0)	**26**	50
A-n34-k05	10	33	E	2382	11 (11)	14056	100	12 (7)	709	29	14 (9)	83	6	14 (9)	**82**	6
A-n34-k05	15	33	E	-	15 (15)	2550	100	23 (21)	1185	88	29 (26)	140	15	29 (26)	**139**	15
A-n36-k05	5	35	NE	153	2 (2)	7049	100	2 (2)	**86**	39	2 (2)	125	26	2 (2)	126	25
A-n36-k05	10	35	NE	4970	7 (7)	2271	100	17 (8)	869	79	17 (7)	740	78	17 (7)	**617**	72
A-n36-k05	15	35	?	-	14 (14)	-	100	8 (6)	-	2	30 (18)	-	0	30 (18)	-	0
A-n37-k05	5	36	NE	87	0 (0)	41	71	0 (0)	40	71	0 (0)	41	70	0 (0)	**38**	70
A-n37-k05	10	36	NE	6435	1 (1)	8901	100	5 (2)	366	86	5 (2)	207	72	5 (2)	**193**	73
A-n37-k05	15	36	NE	-	11 (11)	-	100	52 (16)	10245	83	54 (15)	**3754**	32	55 (15)	4602	38
A-n37-k06	5	36	NE	651	0 (0)	**614**	2	0 (0)	615	2	0 (0)	618	2	0 (0)	617	2
A-n37-k06	10	36	NE	11410	5 (5)	2335	74	15 (5)	**1698**	30	14 (5)	2078	18	14 (5)	2622	14
A-n37-k06	15	36	NE	-	15 (15)	-	96	93 (19)	-	65	107 (24)	**7537**	37	111 (26)	8002	40
A-n38-k05	5	37	NE	225	0 (0)	190	20	1 (0)	187	16	0 (0)	208	26	1 (0)	**184**	15
A-n38-k05	10	37	NE	2406	3 (3)	8499	98	15 (3)	1011	67	11 (4)	**561**	39	15 (4)	574	37
A-n38-k05	15	37	NE	-	8 (8)	-	99	123 (15)	-	79	256 (25)	**12501**	46	297 (28)	-	52
A-n39-k05	5	38	NE	474	0 (0)	292	13	1 (0)	304	12	0 (0)	**290**	12	1 (0)	298	12
A-n39-k05	10	38	NE	-	4 (4)	-	99	14 (8)	-	2	8 (8)	**894**	41	10 (9)	-	0
A-n39-k05	15	38	?	-	10 (10)	-	99	58 (24)	-	35	69 (30)	-	6	76 (30)	-	4
A-n39-k06	5	38	NE	258	0 (0)	**141**	28	2 (0)	186	26	0 (0)	143	28	2 (0)	182	26
A-n39-k06	10	38	NE	2966	7 (7)	2853	96	18 (7)	2095	27	8 (7)	**712**	54	18 (7)	932	30
A-n39-k06	15	38	NE	-	18 (18)	-	99	99 (33)	-	71	133 (35)	**9544**	52	157 (36)	10740	58
A-n44-k06	5	43	NE	287	0 (0)	**37**	63	0 (0)	38	64	0 (0)	39	64	0 (0)	38	63
A-n44-k06	10	43	NE	-	4 (4)	3378	100	30 (7)	4514	61	25 (4)	**2647**	33	30 (8)	3527	30
A-n44-k06	15	43	NE	-	14 (14)	-	100	40 (14)	-	87	53 (16)	8615	29	51 (14)	**6841**	56
A-n45-k06	5	44	NE	247	0 (0)	**109**	30	1 (0)	124	33	0 (0)	113	30	1 (0)	127	33
A-n45-k06	10	44	NE	5814	2 (2)	3633	98	15 (2)	1776	73	3 (2)	**840**	88	15 (2)	1022	52
A-n45-k06	15	44	NE	-	5 (5)	-	100	72 (15)	-	64	91 (13)	**9740**	45	105 (14)	10260	37
A-n45-k07	5	44	NE	425	1 (1)	1040	86	1 (1)	**217**	26	1 (1)	1068	86	1 (1)	219	27
A-n45-k07	10	44	NE	-	5 (5)	-	99	43 (11)	**11374**	41	44 (12)	14097	36	45 (12)	11433	13
A-n45-k07	15	44	?	-	9 (9)	-	99	26 (19)	-	8	21 (17)	-	4	25 (21)	-	0
A-n46-k07	5	45	NE	176	1 (1)	-	100	1 (1)	92	75	1 (1)	**78**	66	1 (1)	80	66
A-n46-k07	10	45	NE	13014	4 (4)	-	100	20 (7)	2832	92	17 (6)	**1096**	78	18 (6)	1172	76
A-n46-k07	15	45	?	-	2 (2)	-	100	65 (18)	-	73	91 (14)	-	25	91 (14)	-	29
A-n48-k07	5	47	NE	992	0 (0)	386	91	2 (0)	212	45	0 (0)	397	92	2 (0)	**194**	39
A-n48-k07	10	47	NE	-	2 (2)	12490	100	22 (3)	14319	19	19 (3)	9941	30	22 (3)	**5666**	19
A-n48-k07	15	47	?	-	5 (5)	-	100	18 (4)	-	19	45 (9)	-	5	52 (14)	-	4
A-n53-k07	5	52	NE	2041	0 (0)	1670	76	3 (0)	1530	14	0 (0)	**1399**	71	3 (0)	3177	5
A-n53-k07	10	52	?	-	0 (0)	-	97	7 (1)	-	4	5 (1)	-	0	5 (1)	-	0
A-n53-k07	15	52	?	-	1 (1)	-	97	14 (8)	-	21	8 (5)	-	0	8 (5)	-	0
A-n54-k07	5	53	NE	3981	0 (0)	**1928**	6	1 (0)	1968	6	0 (0)	1942	6	1 (0)	1998	6
A-n54-k07	10	53	NE	-	1 (1)	-	89	15 (6)	-	15	17 (2)	**13811**	18	19 (6)	-	5
A-n54-k07	15	53	?	-	0 (0)	-	89	17 (9)	-	25	9 (9)	-	84	32 (14)	-	3
A-n55-k09	5	54	NE	1087	0 (0)	785	14	1 (0)	**779**	11	0 (0)	805	14	1 (0)	805	11
A-n55-k09	10	54	NE	-	3 (3)	-	95	14 (5)	5272	65	13 (5)	4805	38	14 (5)	**3475**	38
A-n55-k09	15	54	?	-	1 (1)	-	97	27 (7)	-	26	8 (4)	-	91	52 (11)	-	10

240% faster than GEN_B on all Augerat instances, which demonstrates the effectiveness of solving both the subproblem and the coarse subproblem heuristically. However, on the Solomon instances, GEN_B was 20% and 11% faster than GEN_C and $\text{GEN}_{CH\to BH\to C}$, respectively, and only 5% slower than $\text{GEN}_{CH\to BH\to B}$. This result can be explained by the low number of rationality constraints which are generated in the Solomon instances in comparison to the Augerat instances, which can be attributed to the structure of the Solomon instances. These instances contain clusters consisting of customers of different players, making a collaboration highly profitable, and as a result, any efficient and individually rational allocation is likely to lie in the core. In such a case, the (coarse) subproblem only has to be solved to optimality once.

In general, the row generation algorithms $\text{GEN}_{CH\to BH\to B}$ and $\text{GEN}_{CH\to BH\to C}$

Table 2.3: Results on the JNVRG Solomon instances.

				ENUM	GEN_B			GEN_C			$\text{GEN}_{CH\to BH\to B}$			$\text{GEN}_{CH\to BH\to C}$		
Instance	n	$\lvert V\rvert$	C	T	m	T	T_{RG}	m	T	T_{RG}	m	T	T_{RG}	m	T	T_{RG}
C101-0	5	20	NE	21	0 (0)	17	65	0 (0)	18	64	0 (0)	**17**	64	0 (0)	18	64
C101-1	5	20	NE	10	1 (1)	12	82	2 (1)	13	75	1 (1)	**8**	71	2 (1)	11	72
C101-2	5	20	NE	**32**	0 (0)	34	59	0 (0)	35	59	0 (0)	33	59	0 (0)	34	59
C101-3	5	20	NE	24	0 (0)	13	44	0 (0)	13	43	0 (0)	**12**	43	0 (0)	13	43
C101-4	5	20	NE	10	0 (0)	5	63	0 (0)	5	62	0 (0)	**5**	62	0 (0)	5	62
C101-0	10	20	NE	478	1 (1)	38	84	1 (1)	46	75	1 (1)	**37**	84	1 (1)	48	76
C101-1	10	20	E	299	8 (8)	82	98	10 (10)	43	79	14 (14)	**13**	10	14 (14)	13	10
C101-2	10	20	NE	678	0 (0)	22	36	0 (0)	22	36	0 (0)	**22**	36	0 (0)	22	36
C101-3	10	20	NE	620	2 (2)	71	90	12 (3)	175	74	7 (2)	**60**	67	12 (4)	116	52
C101-4	10	20	NE	253	1 (1)	11	82	1 (1)	11	77	1 (1)	**10**	73	1 (1)	10	74
C101-0	15	20	NE	-	1 (1)	161	96	3 (1)	266	92	1 (1)	**159**	96	3 (1)	202	87
C101-1	15	20	E	13871	9 (9)	123	98	19 (18)	105	90	19 (19)	19	16	19 (19)	**16**	15
C101-2	15	20	NE	-	6 (6)	2060	99	13 (9)	3015	98	14 (10)	**993**	95	11 (8)	1236	97
C101-3	15	20	NE	-	2 (2)	**198**	96	67 (3)	1493	87	18 (2)	198	69	67 (3)	557	62
C101-4	15	20	NE	10548	1 (1)	**26**	93	1 (1)	27	90	1 (1)	26	90	1 (1)	27	90
C101-0	5	33	NE	**122**	0 (0)	169	70	1 (0)	186	65	0 (0)	168	69	1 (0)	180	64
C101-1	5	33	NE	122	0 (0)	22	31	0 (0)	23	31	0 (0)	**22**	31	0 (0)	22	31
C101-2	5	33	NE	68	0 (0)	23	45	0 (0)	24	45	0 (0)	**23**	45	0 (0)	23	46
C101-0	10	33	E	-	3 (3)	-	100	11 (10)	**1760**	40	11 (10)	3246	18	11 (10)	3302	8
C101-1	10	33	NE	-	0 (0)	106	86	6 (0)	205	73	0 (0)	**105**	85	6 (0)	173	68
C101-2	10	33	NE	1328	1 (1)	95	87	2 (1)	109	75	1 (1)	**93**	85	2 (1)	107	74
C101-0	15	33	E	-	4 (4)	-	100	23 (22)	1428	71	12 (12)	**725**	21	12 (12)	843	15
C101-1	15	33	NE	-	3 (3)	410	96	19 (3)	781	78	14 (3)	**311**	61	19 (3)	423	57
C101-2	15	33	NE	-	3 (3)	**310**	96	6 (4)	433	89	9 (6)	401	89	9 (6)	413	85
C101-0	5	50	NE	544	0 (0)	**267**	37	0 (0)	270	37	0 (0)	268	37	0 (0)	272	37
C101-1	5	50	NE	348	0 (0)	237	26	0 (0)	236	26	0 (0)	**234**	26	0 (0)	237	26
C101-0	10	50	NE	-	0 (0)	**265**	37	0 (0)	269	37	0 (0)	266	37	0 (0)	272	37
C101-1	10	50	NE	7340	2 (2)	1261	86	10 (2)	1556	68	10 (2)	1225	66	10 (2)	**1195**	59
C101-0	15	50	NE	-	1 (1)	2863	94	1 (1)	**2652**	93	1 (1)	2843	94	1 (1)	2711	93
C101-1	15	50	NE	-	6 (6)	-	99	37 (7)	9994	88	31 (6)	**3318**	69	40 (8)	4234	70
C201-0	5	20	NE	98	0 (0)	**59**	33	0 (0)	60	34	0 (0)	60	33	0 (0)	61	34
C201-1	5	20	NE	73	1 (1)	52	42	1 (1)	67	33	1 (1)	**50**	39	1 (1)	68	33
C201-2	5	20	NE	111	0 (0)	77	53	0 (0)	78	54	0 (0)	**76**	54	0 (0)	79	54
C201-3	5	20	NE	423	0 (0)	46	33	0 (0)	47	32	0 (0)	**46**	32	0 (0)	47	33
C201-4	5	20	NE	105	0 (0)	98	50	0 (0)	100	50	0 (0)	**98**	50	0 (0)	101	50
C201-0	10	20	NE	1279	2 (2)	**76**	49	2 (2)	460	7	2 (2)	77	50	2 (2)	190	14
C201-1	10	20	NE	1243	1 (1)	**47**	35	1 (1)	98	17	1 (1)	76	26	1 (1)	78	26
C201-2	10	20	NE	-	0 (0)	64	44	0 (0)	64	44	0 (0)	**63**	45	0 (0)	65	44
C201-3	10	20	NE	11348	1 (1)	51	38	1 (1)	104	20	1 (1)	**50**	38	1 (1)	104	19
C201-4	10	20	NE	2394	0 (0)	**65**	24	0 (0)	65	24	0 (0)	65	24	0 (0)	66	24
C201-0	15	20	NE	-	3 (3)	**132**	70	3 (3)	247	43	3 (3)	135	59	3 (3)	223	31
C201-1	15	20	NE	-	1 (1)	**83**	63	1 (1)	121	48	1 (1)	107	44	1 (1)	110	44
C201-2	15	20	NE	-	0 (0)	49	26	0 (0)	49	26	0 (0)	**48**	26	0 (0)	49	26
C201-3	15	20	NE	-	1 (1)	59	47	1 (1)	142	19	1 (1)	**58**	46	1 (1)	148	20
C201-4	15	20	NE	-	1 (1)	**137**	64	1 (1)	144	63	2 (2)	146	66	2 (2)	161	66
C201-0	5	33	NE	2180	0 (0)	620	40	0 (0)	646	40	0 (0)	**620**	40	0 (0)	621	40
C201-1	5	33	NE	-	0 (0)	**2719**	42	0 (0)	2723	42	0 (0)	2723	42	0 (0)	2740	42
C201-2	5	33	NE	2907	0 (0)	1917	50	0 (0)	1879	49	0 (0)	**1867**	49	0 (0)	1881	49
C201-0	10	33	NE	-	0 (0)	558	31	0 (0)	542	31	0 (0)	537	31	0 (0)	**535**	31
C201-1	10	33	NE	-	0 (0)	3645	55	0 (0)	3574	55	0 (0)	**3535**	55	0 (0)	3570	55
C201-2	10	33	NE	-	0 (0)	1553	38	0 (0)	1534	37	0 (0)	**1520**	38	0 (0)	1527	38
C201-0	15	33	NE	-	2 (2)	475	22	2 (2)	2543	4	2 (2)	**468**	21	2 (2)	2575	4
C201-1	15	33	NE	-	1 (1)	2672	39	1 (1)	5613	18	1 (1)	**2614**	39	1 (1)	5707	18
C201-2	15	33	NE	-	0 (0)	1236	23	0 (0)	**1222**	23	0 (0)	1241	23	0 (0)	1240	23
C201-0	5	50	?	-	0 (0)	-	0	0 (0)	-	0	0 (0)	-	0	0 (0)	-	0
C201-1	5	50	NE	-	0 (0)	4219	43	0 (0)	4087	43	0 (0)	**4074**	44	0 (0)	4102	43
C201-0	10	50	?	-	0 (0)	-	0	0 (0)	-	0	0 (0)	-	0	0 (0)	-	0
C201-1	10	50	NE	-	0 (0)	10929	77	0 (0)	10426	78	0 (0)	**10409**	78	0 (0)	10434	78
C201-0	15	50	?	-	0 (0)	-	0	0 (0)	-	0	0 (0)	-	0	0 (0)	-	0
C201-1	15	50	?	-	0 (0)	-	86	2 (1)	-	56	0 (0)	-	86	2 (1)	-	56

settled the most instances in the least amount of time. Comparing both algorithms on the 5 player instances settled by both algorithms, $\text{GEN}_{CH\to BH\to C}$ was on average 5% slower. Moreover, on the 10 players instances settled by both algorithms, $\text{GEN}_{CH\to BH\to C}$ was on average 11% faster. Finally, on the 15 player instances, $\text{GEN}_{CH\to BH\to B}$ was on average 16% faster.

2.6.3 Numerical results for the VRG

The results of our numerical experiments for the VRG instances are given in Tables 2.4 and 2.5 for the Augerat instances and Solomon instances, respectively. Note that ENUM is not included in the results as the algorithm is no longer tractable for these high numbers of players. Moreover, we only settle instances of up to 53 players.

In the specific case of the VRG, our algorithm GEN_B reduces to the algorithm proposed, but never tested, by Engevall et al. (2004). The main difference between our algorithm and that of Göthe-Lundgren et al. (1996) is that the authors only consider coalitions which can be served on a single route, leading to improved computation times. The downside of this approach is that their procedure only works if the core of a game is non-empty, which is almost never the case for the instances presented in Tables 2.4 and 2.5.

When considering the Augerat instances, using the row generation algorithms GEN_B, GEN_C, $\text{GEN}_{CH\to BH\to B}$ and $\text{GEN}_{CH\to BH\to C}$ we could settle a total of 8, 13, 15 and 15 out of 16 instances, respectively. Moreover, $\text{GEN}_{CH\to BH\to B}$ and $\text{GEN}_{CH\to BH\to C}$ were 3123% and 3208% faster than GEN_B on the instances solved by all three methods, respectively. Again these results illustrate the effectiveness of the coarse row generation as well as heuristic row generation, also for the VRG. We find similar results for the Solomon instances, using the row generation algorithms GEN_B, GEN_C, $\text{GEN}_{CH\to BH\to B}$ and $\text{GEN}_{CH\to BH\to C}$ we could settle a total of 14 instances, independently of the algorithm used. We were not able to settle any of the instances based on the Solomon instance C201 with 33 players or more. This may be attributed to the high vehicle capacity for these instances, when compared to the customer demand. As a result, most customers can be visited on a single distribution route. Our column generation algorithm is not suitable for these instances, as the computation time of the labelling algorithm used to solve the pricing problem becomes high in the case of many customers on one route.

2.7 Concluding remarks

We presented a row generation algorithm which can be used to determine a core allocation for instances of the JNVRG, or show that no core allocation exists. Furthermore, we proposed coarse and heuristic row generation to accelerate our algorithm. The results of our numerical experiments suggest that the proposed row generation algorithm is effective for the JNVRG as well as for the VRG, as we can determine a core allocation or show that the core is empty for instances ranging from 5 players

Table 2.4: The results on given instances for the different varieties of the row generation algorithm.

				GEN_B			GEN_C			$\text{GEN}_{CH\to\underline{B}H\to B}$			$\text{GEN}_{CH\to\underline{B}H\to C}$		
Instance	n	$\lvert V\rvert$	C	m	T	T_{RG}	m	T	T_{RG}	m	T	T_{RG}	m	T	T_{RG}
A-n32-k05	31	31	E	48 (48)	1570	100	53 (53)	576	90	48 (48)	50	9	48 (48)	**49**	9
A-n33-k06	32	32	E	68 (68)	1291	99	77 (77)	506	89	80 (80)	56	7	80 (80)	**56**	7
A-n34-k05	33	33	E	41 (41)	4888	100	50 (50)	649	93	39 (39)	38	16	39 (39)	**38**	15
A-n36-k05	35	35	E	52 (52)	9020	100	64 (64)	2009	93	93 (91)	296	9	93 (91)	**265**	9
A-n37-k05	36	36	E	56 (56)	5665	100	61 (61)	2316	75	51 (51)	**110**	6	51 (51)	110	6
A-n37-k06	36	36	E	17 (17)	-	96	66 (66)	**3446**	79	118 (118)	7213	90	128 (115)	4069	14
A-n38-k05	37	37	E	45 (45)	2508	94	44 (44)	853	74	46 (46)	217	4	46 (46)	**212**	4
A-n39-k05	38	38	E	25 (25)	-	99	49 (49)	1774	82	54 (54)	409	6	54 (54)	**406**	6
A-n39-k06	38	38	E	60 (60)	5625	98	54 (54)	1750	89	61 (61)	210	3	61 (61)	**208**	3
A-n44-k06	43	43	E	60 (60)	-	100	107 (107)	10902	98	183 (183)	430	17	168 (168)	**368**	15
A-n45-k06	44	44	E	65 (65)	9916	99	71 (71)	3427	92	86 (86)	**277**	4	86 (86)	285	4
A-n45-k07	44	44	E	6 (6)	-	99	69 (69)	-	93	138 (138)	1949	81	146 (146)	**756**	47
A-n46-k07	45	45	E	43 (43)	-	100	106 (106)	11554	97	210 (210)	**481**	11	210 (210)	494	11
A-n48-k07	47	47	E	49 (49)	-	100	88 (87)	13742	95	255 (255)	765	26	267 (267)	**717**	13
A-n53-k07	52	52	?	4 (4)	-	98	41 (41)	-	95	96 (91)	-	5	96 (91)	-	5
A-n54-k07	53	53	E	24 (24)	-	88	35 (35)	-	86	183 (183)	3174	17	182 (182)	**3138**	15

Table 2.5: Numerical Results on the VRG solomon instances.

				GEN_B			GEN_C			$\text{GEN}_{CH\to\underline{B}H\to B}$			$\text{GEN}_{CH\to\underline{B}H\to C}$		
Instance	n	$\lvert V\rvert$	C	m	T	T_{RG}	m	T	T_{RG}	m	T	T_{RG}	m	T	T_{RG}
C101-0	20	20	NE	98 (98)	834	99	100 (99)	367	88	60 (60)	**50**	32	72 (72)	64	32
C101-1	20	20	E	20 (20)	60	97	24 (24)	60	84	19 (19)	**7**	6	19 (19)	7	7
C101-2	20	20	NE	15 (15)	683	98	18 (18)	646	95	16 (16)	**259**	90	16 (16)	398	93
C101-3	20	20	E	56 (56)	**65**	89	70 (69)	83	67	84 (71)	65	16	82 (69)	70	16
C101-4	20	20	NE	3 (3)	9	79	3 (3)	9	43	3 (3)	**6**	31	3 (3)	10	58
C101-0	33	33	E	14 (14)	1573	97	15 (15)	409	57	33 (33)	304	39	29 (29)	**245**	13
C101-1	33	33	E	69 (69)	6985	100	104 (97)	3038	87	66 (64)	**291**	26	88 (78)	973	71
C101-2	33	33	E	69 (69)	930	99	119 (118)	2563	74	135 (135)	749	19	123 (123)	**601**	8
C101-0	50	50	?	23 (23)	-	99	28 (28)	-	91	191 (171)	-	9	201 (193)	-	55
C101-1	50	50	E	69 (69)	14069	99	74 (74)	5806	80	93 (93)	**1145**	1	93 (93)	1166	1
C201-0	20	20	NE	7 (7)	**258**	85	7 (7)	850	33	10 (10)	404	68	11 (11)	477	47
C201-1	20	20	NE	5 (5)	141	79	6 (6)	219	30	5 (5)	**106**	40	5 (5)	135	21
C201-2	20	20	NE	1 (1)	53	33	1 (1)	83	18	1 (1)	**52**	32	1 (1)	83	17
C201-3	20	20	NE	1 (1)	**103**	70	3 (3)	278	27	1 (1)	104	70	7 (7)	1717	4
C201-4	20	20	NE	2 (2)	422	88	3 (3)	448	85	3 (3)	385	82	3 (3)	**353**	78
C201-0	33	33	?	3 (3)	-	98	9 (8)	-	40	18 (7)	-	30	21 (7)	-	10
C201-1	33	33	?	4 (4)	-	89	4 (2)	-	85	14 (6)	-	65	5 (3)	-	39
C201-2	33	33	?	1 (1)	-	94	15 (4)	-	55	1 (1)	-	94	30 (7)	-	20
C201-0	50	50	?	0 (0)	-	0	0 (0)	-	0	0 (0)	-	0	0 (0)	-	0
C201-1	50	50	?	0 (0)	-	86	0 (0)	-	86	6 (6)	-	54	6 (6)	-	22

with a total of 79 customers to instances of 53 players with a total of 53 customers, within 4 hours of computation time. Clearly, with our methods we can consider more players than previously considered for similar games in the literature. However, it is apparent that there is a trade-off between the use of heuristics and exact methods for solving the underlying optimisation problem, and the number of players and the number of delivery locations that can be dealt with using state-of-the-art methodologies.

Our contribution is mostly methodological, hence further research is required in order apply our algorithms in practice. For example, observe that our algorithm applies to settings in which the depots of the companies are the same or located near each other. However, in practice this might not be the case. Similarly, time windows or other case specific features could impact the routing cost. Further research is required to determine the effectiveness of row generation for these cases. Moreover, it would also be interesting to consider the more practical issues which arise when

implementing such a collaboration in real life. For instance, Basso et al. (2018) mention that it might be difficult for practitioners to agree on an allocation or to trust the others. Hence, it would be interesting to consider these obstacles in the form of a case study.

Appendix

2.A Remaining results on the Augerat instances

Table 2.6: Results on the remaining Augerat instances.

				ENUM	GEN_B			GEN_C			$\text{GEN}_{CH\to BH\to B}$			$\text{GEN}_{CH\to BH\to C}$		
Instance	n	$\lvert V\rvert$	C	T	m	T	T_{RG}	m	T	T_{RG}	m	T	T_{RG}	m	T	T_{RG}
A-n60-k09	5	59	NE	8070	0 (0)	8201	49	2 (0)	4566	5	0 (0)	8388	49	2 (0)	**4540**	4
A-n60-k09	10	59	?	-	0 (0)	-	77	5 (4)	-	3	1 (1)	-	77	5 (4)	-	1
A-n60-k09	15	59	?	-	0 (0)	-	77	9 (6)	-	13	4 (4)	-	77	10 (9)	-	2
A-n61-k09	5	60	NE	6493	0 (0)	6174	2	1 (0)	6172	2	0 (0)	6148	2	1 (0)	**6100**	2
A-n61-k09	10	60	?	-	2 (2)	-	74	13 (2)	11161	24	1 (1)	-	70	13 (2)	**10288**	16
A-n61-k09	15	60	?	-	1 (1)	-	68	12 (8)	-	6	15 (13)	-	21	13 (12)	-	1
A-n62-k08	5	61	NE	-	0 (0)	**1846**	8	1 (0)	-	1	0 (0)	1854	8	1 (0)	-	1
A-n62-k08	10	61	?	-	0 (0)	-	89	4 (3)	-	1	5 (4)	-	42	4 (3)	-	1
A-n62-k08	15	61	?	-	0 (0)	-	89	10 (8)	-	7	5 (5)	-	0	5 (5)	-	0
A-n63-k09	5	62	NE	-	0 (0)	859	47	1 (0)	**779**	18	0 (0)	871	47	1 (0)	785	18
A-n63-k09	10	62	?	-	0 (0)	-	97	4 (3)	-	2	0 (0)	-	97	4 (3)	-	1
A-n63-k09	15	62	?	-	1 (1)	-	98	4 (4)	-	2	5 (3)	-	18	3 (3)	-	1
A-n63-k10	5	62	?	-	0 (0)	-	0	0 (0)	-	0	0 (0)	-	0	0 (0)	-	0
A-n63-k10	10	62	?	-	0 (0)	-	0	0 (0)	-	0	0 (0)	-	0	0 (0)	-	0
A-n63-k10	15	62	?	-	0 (0)	-	0	0 (0)	-	0	0 (0)	-	0	0 (0)	-	0
A-n64-k09	5	63	?	-	0 (0)	-	0	0 (0)	-	0	0 (0)	-	0	0 (0)	-	0
A-n64-k09	10	63	?	-	0 (0)	-	0	0 (0)	-	0	0 (0)	-	0	0 (0)	-	0
A-n64-k09	15	63	?	-	0 (0)	-	0	0 (0)	-	0	0 (0)	-	0	0 (0)	-	0
A-n65-k09	5	64	NE	4774	0 (0)	-	96	3 (1)	928	30	0 (0)	-	96	3 (1)	**878**	24
A-n65-k09	10	64	NE	-	0 (0)	-	96	11 (2)	4580	40	0 (0)	-	96	11 (2)	**2884**	29
A-n65-k09	15	64	?	-	3 (3)	-	96	13 (10)	-	14	2 (2)	-	0	2 (2)	-	0
A-n69-k09	5	68	NE	-	0 (0)	13533	1	1 (0)	-	1	0 (0)	**13474**	1	1 (0)	-	0
A-n69-k09	10	68	?	-	1 (1)	-	40	2 (2)	-	2	1 (1)	-	0	1 (1)	-	0
A-n69-k09	15	68	?	-	0 (0)	-	52	3 (2)	-	2	2 (2)	-	0	2 (2)	-	0
A-n80-k10	5	79	NE	-	0 (0)	11816	4	1 (0)	-	2	0 (0)	**11807**	4	1 (0)	-	2
A-n80-k10	10	79	?	-	0 (0)	-	56	2 (1)	-	1	0 (0)	-	56	2 (1)	-	1
A-n80-k10	15	79	?	-	1 (1)	-	56	5 (4)	-	37	7 (7)	-	1	7 (7)	-	1

Chapter 3

The Joint Network Vehicle Routing Game with Optional Customers

Abstract

We consider logistic collaborations where multiple carriers collaborate by consolidating demands, combining delivery routes, and serving new customers. Logistic collaborations are known to provide numerous benefits such as an increase in profit as well as a reduction in emissions for all parties involved. One of the challenges associated with collaborations is the allocation of the additional profits. To this end, we model the corresponding profit allocation problem as a collaborative game. Here, the profit obtained by any subset of the collaborating carriers depends on the new customers served by the remaining carriers. Specifically, we try to determine an allocation in the core based on both the best-case profit and the worst-case profit that each subset of carriers can attain. To achieve this, we propose a heuristic row generation algorithm. We verify the performance of this algorithm on instances derived from benchmark instances of the capacitated vehicle routing problem. We show that our heuristic algorithm scales well with respect to the number of carriers considered and provides allocations similar to those obtained by enumerating all best-case and worst-case profits of each coalition.

This chapter is based on van Zon and Desaulniers (2021).

3.1 Introduction

Collaborative transportation among carriers has the potential to greatly increase the efficiency of their transportation networks. By collaborating, the carriers can significantly reduce their cost (Guajardo and Rönnqvist, 2016), increase their service levels, and improve their market share (Gansterer and Hartl, 2018). Moreover, such collaborations are known to provide numerous other benefits including reductions in noise pollution, road congestion, and greenhouse gas emissions as shown by Ballot and Fontane (2010) and Pérez-Bernabeu et al. (2014).

In a horizontal collaboration, carriers may collaborate by consolidating the demands of their customers and combining delivery routes. Here, it is assumed that the carriers exchange some or all of their customers. Gansterer and Hartl (2018) identify three major literature streams in horizontal collaborations: centralised collaborative planning, decentralised collaborative planning without auctions, and auction-based decentralised planning. In centralised collaborative planning, the carriers offer all their requests up for exchange. As a consequence, the joint profit can be maximised. On the other hand, in decentralised collaborative plannings, each carrier can offer only a subset of their customers to the other carriers, typically, those which are expensive to serve. Then, the customers are assigned to other carriers either by an allocation mechanism (see, e.g., Özener et al., 2011; Wang and Kopfer, 2014) or a combinatorial auction (see, e.g., Krajewska and Kopfer, 2006; Gansterer and Hartl, 2016).

In this chapter, we study a centralised collaborative planning among multiple carriers, each responsible for multiple customers with a given demand. It is assumed that each carrier serves the demand of its customers from a single depot, using a limited fleet of capacitated vehicles. Moreover, we assume that the carriers are rational and aim to maximise their profits. Because they consolidate demands and combine delivery routes, the carriers are likely to have residual capacity which can be used to generate additional profits by collectively serving new customers, which are initially not served by any of the carriers due to capacity restrictions. We believe that a larger profit increase, induced by a centralised collaboration, can be achieved when the reason for collaborating is to gather sufficient capacity to service an additional large demand. The typical example that we have in mind is the following. To benefit from an economy of scale, a company with multiple branches may require to be supplied by the same carrier. Therefore, to acquire the branches of this company as customers (all together), a carrier must have sufficient capacity or must collaborate with one or several other carriers. This is the type of collaboration that we study in

this chapter.

We focus on the profit allocation problem that arises when multiple carriers form a horizontal collaboration by consolidating demands, combining delivery routes, and collectively serving new customers. Specifically, we try to find an allocation such that each carrier has an incentive to collaborate with all the other carriers. Profit, or equivalently, cost allocation problems in collaborative transportation are often modelled as cooperative games (Guajardo and Rönnqvist, 2016). A cooperative game is defined as a pair consisting of a set of players, also known as the grand-coalition, and a characteristic function which maps each subset of players, also referred to as a coalition, to a value. We choose to interpret this value as the profit of a coalition, i.e., the value created by the collaboration. We aim to allocate the profit of the grand-coalition to the players such that each player has an incentive to cooperate. To this end, it is common to consider core allocations (Guajardo and Rönnqvist, 2016). A core allocation is an allocation of the grand-coalition profit to the players such that no coalition is better off by leaving the grand-coalition.

Originally, centralised collaborative transportation was studied from the perspective of the customers, rather than the carriers. Göthe-Lundgren et al. (1996), Engevall et al. (1998) and Engevall et al. (2004) consider a setting in which multiple customers are served by a single carrier. More specifically, Engevall et al. (1998) consider the cost allocation problem where all customers are served on a single route, whereas Göthe-Lundgren et al. (1996) and Engevall et al. (2004) consider the cost allocation problem where all customers are served on multiple routes. The aim of the allocation problem is to allocate the cost in a fair manner to each of the customers. To this end, Engevall et al. (1998) model their problem as a travelling salesman game (TSG), which was originally proposed by Fishburn and Pollak (1983). Moreover, Göthe-Lundgren et al. (1996) and Engevall et al. (2004) model their cost allocation problem as a vehicle routing game (VRG), as originally proposed by Göthe-Lundgren et al. (1996).

More recently, centralised collaborative transportation has also been studied from the perspective of carriers. For example, Krajewska et al. (2008) and Dai and Chen (2015) study a profit allocation problem which arises from a collaboration among freight carriers, each with a set of pickup and delivery requests with time windows. In a cooperative setting, the carriers, or players, collaborate by collectively serving their requests. The goal of the corresponding profit allocation problem is to allocate the profit to the carriers, rather than to the customers. Similarly, Zakharov and Shchegryaev (2015) and van Zon et al. (2021a) study a collaboration among multiple

carriers, each with multiple customers corresponding to delivery requests, which have to be satisfied from a single depot. In contrast to Krajewska et al. (2008) and Dai and Chen (2015), they formulate the problem as a cost allocation problem, with the aim to allocate the cost of the grand-coalition to the carriers.

The previously mentioned studies are mainly concerned with optimising the efficiency of the existing routing networks through collaborative transportation, but do not consider any additional gains which could be obtained by exploiting the residual capacity obtained from the collaboration to service new customers. For example, in a collaboration among multiple carriers, each carrier has a fleet at its disposal. By consolidating demands and combining delivery routes, the carriers are able to utilise their capacity more efficiently. As a result, less vehicles may be needed for the transportation and additional customers, that initially were not considered by any of the carriers, may be served with their residual capacity for a profit.

The setting of van Zon et al. (2021a) is closest to ours. The authors define the cost of a coalition as the optimal objective value to a capacitated vehicle routing problem (CVRP) of the customers belonging to the coalition. They model their cost allocation problem as the joint network vehicle routing game (JNVRG). Moreover, they propose a row generation algorithm to determine a core allocation.

In the previous multiple-player settings, a collaboration between players is established when it becomes profitable for each player to serve customers of other players or to have some of their own customers served by others. In this case, a single customer can easily foster a collaboration between two players as it, generally, does not require a large capacity to be served. On the other hand, it might yield only a small profit increase that could be exceeded by potential overhead costs associated with the collaboration. We believe that a larger profit increase can be achieved if the reason for collaborating is to gather sufficient capacity to service additional customers.

We generalise the setting of van Zon et al. (2021a) by assuming that there exist multiple clusters of optional customers (each composed, e.g., of branches of a company) which are not assigned to one of the players. A coalition obtains a reward for visiting all customers of a cluster. In this case, we cannot simply define the profit of a coalition, as it is not trivial to determine how the clusters of optional customers are allocated between the coalition and the remaining players. The allocation of the clusters of optional customers between the coalition and the remaining players can be modelled by means of combinatorial auctions (Gansterer and Hartl, 2018). However, this would implicitly assume some form of collaboration between the coalition and the remaining players. Therefore, we choose to consider both a best-case profit as

well as a worst-case profit for each coalition. In the best case, we assume that a coalition can freely select the clusters of optional customers it is willing to serve, thereby obtaining the maximum profit. On the other hand, in the worst case, we assume that the remaining players form a coalition and select the clusters of optional customers which benefit them the most. In this case, the original coalition can only select the clusters of optional customers which have not been selected by the remaining players. In this manner, we aim to provide an accurate representation of the profit which the coalition may attain. Our goal is to try to determine an allocation in the core as defined by the best-case profit. However, if this core is empty, we try to determine an allocation in the core as defined by the worst-case profit, or show that the core as defined by the worst-case profit is empty.

Our contributions are as follows. First, we define a new cooperative game which, to the best of our knowledge, has not been studied before. Here, we consider a centralised horizontal collaboration in which profits are obtained by consolidating demands, combining delivery routes, and collectively serving new customers. For each coalition, we consider both the worst-case and best-case profits that can be obtained, depending on the manner in which the clusters of optional customers are allocated. In the best case, we assume that each coalition can freely select which clusters of optional customers to serve. On the other hand, in the worst case, we assume that the players not included in the coalition get to freely select which clusters of optional customers to serve first. Thereafter, the coalition can only select among the remaining clusters which ones to serve. Secondly, we propose a heuristic row generation algorithm to determine allocations in the cores as defined by the best-case and the worst-case profit, or to show that both cores are empty. The corresponding row generation subproblem is a generalisation of a vehicle routing problem with profits. We propose a branch-and-price-and-cut algorithm to solve the subproblem. To this end, we adapt several well-known acceleration strategies for the problem.

The remainder of this chapter is structured as follows. First, in Section 3.2, we formally introduce the grouped capacitated profitable tour problem with contract customers (GCPTPCC). Moreover, we introduce the joint network vehicle routing game with optional customers (JNVRGOC), which is a cooperative game based on the GCPTPCC. Here, we also introduce the core as an allocation concept. In Section 3.3, we propose a heuristic row generation algorithm in order to determine a profit allocation. We formulate the row generation subproblem as a mixed integer programming problem in Section 3.3.1 and develop a branch-price-and-cut algorithm for solving the subproblem in Section 3.3.2. Next, we describe acceleration strategies

for the row generation algorithm in Section 3.3.3. In Section 3.4, we report numerical results obtained on instances derived from the vehicle routing instances of Augerat (1995). Finally, we present our concluding remarks in Section 3.5.

3.2 The Joint Network Vehicle Routing Game with Optional Customers

Before presenting the JNVRGOC, we introduce the GCPTPCC which is initially defined for a single carrier. Consider a complete directed graph $G = (V, A)$. The vertex set V is defined as $\{0\} \cup V^c \cup V^o$, where 0 represents the depot of the carrier, V^c the set of contract customers, and V^o the set of optional customers. The set of optional customers V^o is partitioned in m non-empty and disjoint clusters V_l^o with $l \in M = \{1, \ldots, m\}$. A reward $p_l^o > 0$ is only obtained if all customers in V_l^o are visited. In contrast to the optional customers, the contract customers in V^c all have to be visited and no reward is given for visiting these customers. With each arc $(v, w) \in A$, a travel cost $c_{vw} \geq 0$ is associated, which we assume to adhere to the triangle inequality. Furthermore, each customer $v \in V^c \cup V^o$ has a demand $d_v > 0$. A limited number of vehicles k with capacity Q is available at the depot to satisfy the demands by visiting customers along routes. A route is defined as a simple cycle, starting and ending at the depot, such that the total demand of the visited customers does not exceed vehicle capacity. Here, we assume that $d_v \leq Q$ for all $v \in V^c \cup V^o$.

The GCPTPCC is the problem of constructing routes in such a way that the profit, i.e., the total rewards minus the total costs, is maximised and every contract customer $v \in V^c$ is visited. We denote the optimal objective value of a GCPTPCC on the graph induced by the vertex sets V^o and V^c as GCPTPCC(V^o, V^c). To the best of our knowledge, the GCPTPCC has not been studied before. However, van Zon et al. (2021a) study the grouped capacitated profitable tour problem (GCPTP), which can be seen as a special case of the GCPTPCC in which no contract customers are considered.

The definition of the GCPTPCC can be generalised to consider multiple collaborating carriers as proposed below. To do so, we need to define one depot per carrier, with its associated vehicle fleet. Nevertheless, in this chapter, we assume that there is a single depot for all players which may be seen as a super-depot regrouping the players' individual depots that are assumed to be near each other. The model and algorithm presented below can, however, be easily adapted to the multiple-depot case.

We consider the profit allocation problem that arises when multiple carriers collaboratively serve both contract customers and selected optional customers. Here, we assume that the sets of optional customers as well as the routes used to service them are chosen according to an optimal solution of a GCPTPCC. The overall profit obtained by executing this solution needs to be allocated among the carriers in a fair manner. To this end, we define a cooperative game based on the GCPTPCC, which we call the JNVRGOC. A cooperative game is defined as a pair $\langle N, v\rangle$, where the grand-coalition $N = \{1, \ldots, n\}$ denotes the set of players, $v : \mathcal{P}(N) \to \mathbb{R}$ the profit of a coalition $S \subseteq N$, and $\mathcal{P}(N)$ the power set of N. Here, each player $i \in N$ is assumed to have a finite fleet of k_i vehicles with capacity Q available. Hence, a coalition $S \subseteq N$ has $\sum_{i \in S} k_i$ vehicles available. Moreover, we also assume that each player $i \in N$ has a set of contract customers V_i^c for which the demand has to be fulfilled, with $V_i^c \cap V_j^c = \emptyset$ for $i, j \in N$, $i \neq j$.

The profit of a coalition $S \subseteq N$ depends on how the clusters of optional customers are allocated among the coalition S and the remaining players $N \backslash S$. Here, we consider two specific cases. First, we consider the best-case profit $v^B(S)$ that coalition S can obtain. In the best case, coalition S can freely select clusters of optional customers to serve. We define the best-case profit $v^B(S)$ as follows

$$v^B(S) = \text{GCPTPCC}\left(V^o, \bigcup_{i \in S} V_i^c\right) + \sum_{i \in S} p_i^c, \tag{3.1}$$

where we add the stand-alone routing costs $p_i^c = -\text{GCPTPCC}(\emptyset, V_i^c)$ of the contract customers belonging to the coalition, in order to ensure that the value created by the coalition is properly represented by $v^B(S)$. Note that $v^B(S)$ is obtained not only by visiting sets of optional clusters but also by visiting more efficiently the contract customers of the players in S. Secondly, we consider the worst-case profit $v^W(S)$ of a coalition $S \subseteq N$, in which we assume that the remaining players $N \backslash S$ form a coalition and can freely select which clusters of optional customers to serve. This may significantly lower the profit that coalition S can obtain. We define the worst-case profit as follows

$$v^W(S) = \text{GCPTPCC}\left(\bigcup_{l \in M'(N \backslash S)} V_l^o, \bigcup_{i \in S} V_i^c\right) + \sum_{i \in S} p_i^c, \tag{3.2}$$

where $M'(N \backslash S) \subseteq M$ denotes the subset of indices l of the optional customer clusters V_l^o which are not supplied in an optimal solution to $v^B(N \backslash S)$. Notice that there

might be several optimal solutions, but we retain only one of them. In this chapter, we consider and compare both the best-case profit and the worst-case profit by means of the following profit function

$$v^{\alpha}(S) = \alpha v^{B}(S) + (1-\alpha)v^{W}(S), \tag{3.3}$$

where $\alpha \in [0,1]$ and $S \subseteq N$, such that $v^1(S) = v^B(S)$ and $v^0(S) = v^W(S)$. Moreover, we denote the grand-coalition profit by $v(N)$, as it is independent on the value of α, i.e., $v(N) = v^{\alpha}(N)$ for any $\alpha \in [0,1]$. We define the JNVRGOC as a cooperative game $\langle N, v^{\alpha}\rangle$.

3.2.1 Profit allocation

The aim of the JNVRGOC is to allocate the profit of the grand-coalition to the players. We denote a profit allocation by $y \in \mathbb{R}^n$, where y_i is the profit allocated to player $i \in N$. Furthermore, we define $y(S) = \sum_{i \in S} y_i$ for $S \subseteq N$. Given $\alpha \in [0,1]$, a profit allocation $y \in \mathbb{R}^n$ is said to be individually rational if and only if $y_i \geq v^{\alpha}(\{i\})$ for all $i \in N$. Moreover, y is efficient if and only if $y(N) = v(N)$. Given $\alpha \in [0,1]$, the set $I(\langle N, v^{\alpha}\rangle)$ of efficient and individually rational allocations is known as the imputation set. This set is seen as a minimum requirement for cooperation as each player benefits relative to his stand-alone profit. Note that given $\alpha_1 \geq \alpha_2$ it holds that $I(\langle N, v^{\alpha_1}\rangle) \subseteq I(\langle N, v^{\alpha_2}\rangle)$ as $v^{\alpha_1}(\{i\}) \geq v^{\alpha_2}(\{i\})$ for all $i \in N$. Moreover, it may be the case that the imputation set is empty for all $\alpha \in [0,1]$, as illustrated in the example in Appendix 3.A.

Often, profits (or costs) are allocated by means of core allocations. A core allocation is an allocation which is both efficient and group rational. Here, given $\alpha \in [0,1]$, an allocation y is said to be group rational if and only if $y(S) \geq v^{\alpha}(S)$ for all $S \subseteq N$. We denote the core, i.e., the set of all core allocations, by $\text{Core}(\langle N, v^{\alpha}\rangle)$. Note that $\text{Core}(\langle N, v^{\alpha_1}\rangle) \subseteq \text{Core}(\langle N, v^{\alpha_2}\rangle)$ if and only if $0 \leq \alpha_1 \leq \alpha_2 \leq 1$. Moreover, given $\alpha \in [0,1]$, it holds that $\text{Core}(\langle N, v^{\alpha}\rangle) \subseteq I(\langle N, v^{\alpha}\rangle)$. Next, we provide an example to illustrate the differences between the worst-case core and the best-case core.

Example 3.2.1. *Consider the 3-player instance of the JNVRGOC with $N = \{1,2,3\}$ as presented in Figure 3.1. Let $V_1^c = \{1\}, V_2^c = \{2\}$ and $V_3^c = \{3\}$ with $d_1 = d_2 = d_3 = 1$. Moreover, let $k_i = 1$ for each player $i \in N$ and assume that all vehicles have a capacity $Q = 4$. There is a single cluster of optional customers which is defined as $V_1^o = \{4,5,6\}$ with $d_4 = d_5 = d_6 = 1$, $p_1^o = 13$. Assuming that $c_{ij} = c_{ji}$ for all $(i,j) \in A$, the relevant arc costs are given in Figure 3.1.*

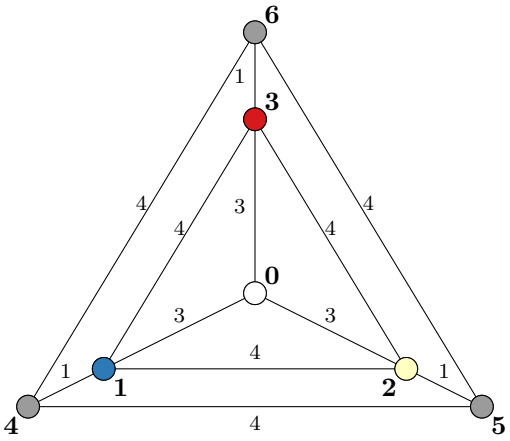

Figure 3.1: A 3-player instance of the JNVRGOC.

Note that due to the symmetry of the instance, the profits are solely dependent on the size of a coalition. For this reason, we choose a specific coalition of a certain size to determine the profit for all coalitions of this size.

First, we consider the best-case profits. Consider coalition $\{1\}$*. In the best case, it is optimal to use route* $(0, 1, 4, 5, 6, 0)$ *at cost 16. It follows that* $v^B(\{1\}) = v^B(\{2\}) = v^B(\{3\}) = 13-16+6 = 3$*. Similarly, we find that for coalitions of size 2, e.g., {1,2}, it is optimal to use routes* $(0, 1, 4, 6, 0)$ *and* $(0, 2, 5, 0)$ *at costs 12 and 8, respectively. Hence, we find that* $v^B(\{1, 2\}) = v^B(\{1, 3\}) = v^B(\{2, 3\}) = 13 - 12 - 8 + 6 + 6 = 5$*. For the grand-coalition, it is optimal to use routes* $(0, 1, 4, 6, 3, 0)$ *and* $(0, 2, 5, 0)$ *at costs 12 and 8, respectively. Hence, the profit of the grand-coalition is* $v^B(N) = 13 - 12 - 8 + 6 + 6 + 6 = 11$*.*

Secondly, we consider the worst-case profits. For the singleton coalitions, the worst-case profits are trivial ($v^W(\{i\}) = 0$*,* $\forall i \in N$*) as no optional customers can be visited. On the other hand, it is non-trivial for the coalitions of size 2. For example, consider coalition* $\{1, 2\}$*. This coalition can be visited using route* $(0, 1, 2, 0)$ *at cost 10, yielding* $v^W(\{1, 2\}) = v^W(\{1, 3\}) = v^W(\{2, 3\}) = 0 - 10 + 6 + 6 = 2$*. Hence, despite the fact that no optional customers can be visited, the coalition is still profitable. Finally, recall that the worst-case profit of the grand-coalition is equal to the best-case profit, i.e.,* $v^W(N) = 11$*.*

One can observe that, for example, $y^B = (3, 4, 4)$ *is a best-case core allocation. On the other hand,* $y^W = (9, 1, 1)$ *is clearly not a best-case core allocation. However,* y^W *still belongs to the worst-case core. Both allocations, as well as the worst-case and best-case cores are visualised in Figure 3.2. At each of the corner points, a*

single player is allocated a profit of 11 (e.g., player 1 is associated with the lower-left corner), whereas the other players are allocated zero profit. All efficient allocations can be written as convex combinations of these allocations. Hence the largest triangle is the convex hull of these allocations. Each solid line in the figure corresponds to a best-case rationality constraint, whereas each dashed line to a worst-case rationality constraint. Moreover, the arrow indicates the feasible half-space. The best-case core is represented by the dark triangle, whereas the worst-case core is represented by the grey hexagon.

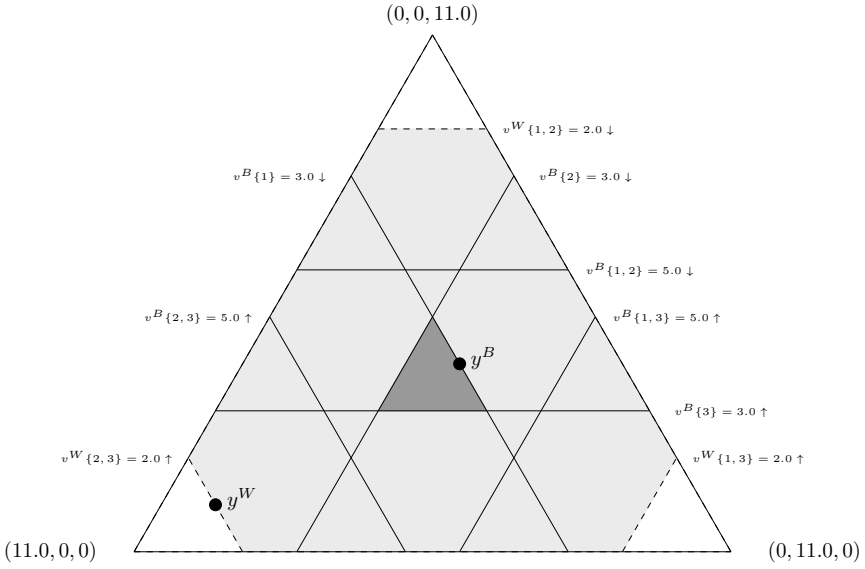

Figure 3.2: Visualisation of both worst-case and best-case cores.

In the remainder of this chapter, we aim to find an allocation for the JNVRGOC in the best-case core. However, if the best-case core is empty, we try to find an allocation in the worst-case core by *minimally* relaxing the best-case rationality constraints. To this end, we consider the following linear programming problem

$$\begin{aligned}
\max \quad & \alpha, && && (3.4)\\
\text{s.t.} \quad & y(S) \geq v^{\alpha}(S) && \forall\, S \subset N, && (3.5)\\
& y(N) = v(N), && && (3.6)\\
& 0 \leq \alpha \leq 1, && && (3.7)\\
& y_i \geq 0 && \forall\, i \in N. && (3.8)
\end{aligned}$$

To obtain an allocation which is as close as possible to the best-case core, objective function (3.4) tries to maximise α, which is considered as a decision variable. Constraints (3.5) enforce the rationality constraints whose right-hand sides are linear functions of α, whereas constraint (3.6) is the efficiency constraint. Moreover, constraints (3.7) and (3.8) define the variable domains.

Note that if there exists an optimal solution α' and y' to (3.4)–(3.8) for which $\alpha' = 1$, it holds that y' is in the best-case core. Alternatively, if (3.4)–(3.8) has no feasible solution, then the worst-case core is empty.

3.3 A heuristic row generation algorithm

Two main challenges have to be considered when solving (3.4)-(3.8) to optimality. First, there are exponentially many rationality constraints (3.5) in the number of players. Second, the optimal objective values of two GCPTPCCs have to be computed for each coalition. Here, it is important to note that the GCPTPCC is a generalisation of the NP-hard CVRP (Lenstra and Rinnooy Kan, 1981), which is well-known to be computationally expensive to solve.

In similar studies, both challenges are alleviated by applying row generation to the rationality constraints (see Göthe-Lundgren et al., 1996; Engevall et al., 2004; Drechsel and Kimms, 2010; Dai and Chen, 2015; van Zon et al., 2021a). Violated rationality constraints are identified by means of a row generation subproblem. Consider an optimal solution α' and y' to (3.4) and (3.6)-(3.8). A violated rationality constraint (3.5) can be identified by solving the following row generation subproblem

$$\max_{S \subset N} \left\{ v^{\alpha'}(S) - y'(S) \right\}. \tag{3.9}$$

However, if $\alpha' \neq 1$, it is not at all obvious how subproblem (3.9) can be solved efficiently due to the inclusion of the worst-case profit of coalition S. Alternatively, one could try to apply row generation by using lower bounds on the worst-case profits of the coalitions but it does not seem easy to determine such tight bounds. As a consequence, we were unable to devise an efficient row generation algorithm for solving (3.4)-(3.8) to optimality.

For instances with a small number of players, we can simply enumerate both the best-case profits as well as the worst-case profits. However, for instances with a larger number of players, enumeration is no longer tractable. Instead of solving (3.4)-(3.8) to optimality, we propose an iterative heuristic algorithm to determine allocations in

the best-case core or possibly in the worst-case core, if the best-case core is empty. At each iteration q, this algorithm solves a relaxation of (3.4)-(3.8) to determine an allocation. This relaxation is obtained by relaxing the constraints (3.5) for all coalitions except those in a subset $\Omega_q \subseteq \mathcal{P}(N)$. It writes as follows:

$$\max \quad \alpha_q, \tag{3.10}$$

$$\text{s.t.} \quad y(S) \geq \alpha_q v^B(S) \qquad \forall\, S \in \mathcal{P}(N) \setminus \Omega_q, \tag{3.11}$$

$$y(S) \geq \alpha_q v^B(S) + (1-\alpha_q) v^W(S) \qquad \forall\, S \in \Omega_q, \tag{3.12}$$

$$y(N) = v(N), \tag{3.13}$$

$$0 \leq \alpha_q \leq 1, \tag{3.14}$$

$$y_i \geq 0 \qquad \forall\, i \in N. \tag{3.15}$$

The objective function (3.10) still aims at maximising the value of α which is replaced by α_q at iteration q. Constraints (3.11) are the relaxed rationality constraints (the term involving $v^W(S)$ is omitted), whereas constraints (3.12) are the non-relaxed ones. The efficiency constraint is given by (3.13). Finally, the variable bounds are enforced through (3.14) and (3.15).

The proposed profit allocation heuristic is described in Algorithm 1, where Υ_q denotes the set of constraints (3.11) that are initially considered in model (3.10)-(3.15) at iteration q, i.e., those whose best-case profits have already been computed. We start by setting $q = 0$, $\Omega_1 = \emptyset$, and $\Upsilon_1 = \emptyset$, and we compute $v(N)$ by solving a GCPTPCC for the grand-coalition N. Then, we perform the following steps iteratively, incrementing the iteration counter q at each iteration (Step 4). First, in Step 5, we solve (3.10)-(3.15) to optimality using row generation and store the computed optimal value α'_q. To this end, we relax the constraints (3.11) except those associated with a coalition $S \in \Upsilon_q$ and find any tight or violated best-case rationality constraints by solving a row generation subproblem (see Sections 3.3.1 and 3.3.2), which implies computing the best-case profit $v^B(S)$ for each corresponding coalition S. If $\alpha'_q = 1$, it means that an allocation in the best-case core has been found and the algorithm stops after setting $\Theta_q = \emptyset$ in Step 7 to meet the repeat loop exiting criterion in Step 20. Otherwise, after updating set Υ_q in Step 9 by appending the coalitions for which a constraint (3.11) was generated in Step 5, we identify in Step 10 the best-case rationality constraints (3.11) that are tight at optimality and define Θ_q as the set of their associated coalitions. For these coalitions, we would like to compute their worst-case profits and convert their corresponding relaxed rationality constraints (3.11) into non-relaxed constraints (3.12). To compute these worst-case

Algorithm 1 Profit allocation heuristic

1: Set $q = 0$, $\Omega_1 = \emptyset$, and $\Upsilon_1 = \emptyset$.
2: Compute $v(N)$.
3: **repeat**
4: $q = q + 1$.
5: Solve (3.10)-(3.15) to optimality using row generation and let α'_q be the optimal value.
6: **if** $\alpha'_q = 1$ **then**
7: Set $\Theta_q = \emptyset$
8: **else**
9: Update Υ_q.
10: Let $\Theta_q \subseteq \Upsilon_q$ be the set of coalitions for which constraints (3.11) are tight.
11: Let $\overline{\Theta}_q = \{S \in \mathcal{P}(N) \setminus (\Omega_q \cup \Upsilon_q) \mid N \setminus S \in \Theta_q\}$.
12: **for** $S \in \overline{\Theta}_q$ **do**
13: Determine $v^B(S)$.
14: **end for**
15: **for** $S \in \Theta_q$ **do**
16: Determine $v^W(S)$.
17: **end for**
18: Set $\Omega_{q+1} = \Omega_q \cup \Theta_q$ and $\Upsilon_{q+1} = (\Upsilon_q \setminus \Theta_q) \cup \overline{\Theta}_q$
19: **end if**
20: **until** $\Theta_q = \emptyset$

profits, we need to compute the best-case profits of their complements with respect to N (if not computed yet) and, more specifically, identify the clusters of optional customers that are not covered in the computed optimal solutions. Consequently, in Step 11, we determine the set $\overline{\Theta}_q$ of the complements for which their best-case profits have not been computed yet. Thereafter, for each coalition $S \in \overline{\Theta}_q$, we compute its best-case profit $v^B(S)$ in Step 13 and, for each coalition $S \in \Theta_q$, we find its worst-case profit in Step 16. Finally, we determine Ω_{q+1} and Υ_{q+1}. We stop iterating when $\Theta_q = \emptyset$ and output the last allocation found in Step 5.

Let (α', y') denote the solution computed by this iterative procedure. Moreover, let (α^*, y^*) denote the solution obtained by solving (3.4)-(3.8) to optimality. It holds that $\alpha' \geq \alpha^*$. Moreover, $y^* \in \text{Core}(\langle N, v^{\alpha^*} \rangle)$, whereas we can only guarantee $y' \in \text{Core}(\langle N, v^{\alpha'} \rangle)$ if $\alpha' = 1$. Note that this implies that, if $\alpha' < 1$, it is not guaranteed that the allocation is in the worst-case core, i.e., that (α', y') is feasible for (3.4)-(3.8). However, each coalition is at least allocated α' times its best-case profit.

3.3.1 The row generation subproblem

Consider an incumbent solution (α'_q, y') to (3.10)-(3.15) at iteration q. We identify tight or violated rationality constraints (3.11) at iteration q by means of the following row generation subproblem

$$SP(\alpha'_q, y') = \max_{S \in \mathcal{P}(N) \setminus \Omega_q} \left\{ \alpha'_q v^B(S) - y'(S) \right\}. \tag{3.16}$$

If $SP(\alpha'_q, y') \geq 0$, we have identified a tight or violated best-case rationality constraint of the form $y(S) \geq \alpha v^B(S)$. Otherwise, (α'_q, y') is optimal for (3.10)-(3.15) and all tight rationality constraints have been generated at iteration q. Note that the profit of a coalition has not yet been determined and is part of solving the subproblem.

As proposed by van Zon et al. (2021a), the subproblem can be modelled as a GCPTP where we consider an additional constraint with regards to the number of vehicles used. We formulate the GCPTP as the following mixed integer program. Let R denote the set of feasible routes in $G = (V, A)$. Let c_r be the cost of route r which is given by the sum of the costs of its arcs. Also, let the coefficient a_{rv} be equal to the number of times customer v is included in route r. Hence, $a_{rv} = 0$ if customer v is not included in r and $a_{rv} = 1$ if customer v is included exactly once in r. Next, for each route $r \in R$, define a binary decision variable x_r such that $x_r = 1$ if and only if route r is selected in the solution. Furthermore, for each $l \in M$, we define the binary decision variable z_l such that $z_l = 1$ if and only if all customers of V_l^o are visited exactly once in the solution. For each optional customer $v \in V_l^o$, let $l(v) = l$ and, for each contract customer $v \in V^c$ of player $i \in N$, let $i(v) = i$. Finally, let s_i be a binary variable that indicates whether or not player $i \in N$ belongs to the selected coalition S.

Given this notation, the subproblem can be formulated as follows:

$$SP(\alpha'_q, y') = \max \quad \sum_{l \in M} p^o_l z_l - \sum_{r \in R} c_r x_r + \sum_{i \in N} s_i (p^c_i - \frac{1}{\alpha'_q} y'_i) \tag{3.17}$$

$$\text{s.t.} \quad \sum_{r \in R} a_{rv} x_r = s_{i(v)} \qquad \forall\, v \in V^c, \tag{3.18}$$

$$\sum_{r \in R} a_{rv} x_r = z_{l(v)} \qquad \forall\, v \in V^o, \tag{3.19}$$

$$\sum_{r \in R} x_r \leq \sum_{i \in N} s_i k_i, \tag{3.20}$$

$$\sum_{i \in S} s_i - \sum_{i \in N \setminus S} s_i \leq |S| - 1 \qquad \forall S \in \Omega_q \tag{3.21}$$

$$x_r \in \{0, 1\} \qquad \forall\, r \in R, \tag{3.22}$$

$$z_l \in \{0, 1\} \qquad \forall\, l \in M, \tag{3.23}$$

$$s_i \in \{0, 1\} \qquad \forall\, i \in N. \tag{3.24}$$

Objective function (3.17) searches to maximise the amount by which a rationality constraint (3.11) is violated with respect to the incumbent solution (α'_q, y'). Constraints (3.18) ensure that all contract customers of the selected players are visited, while (3.19) impose that all selected optional customers are visited. Constraint (3.20) guarantees that the number of vehicles available is not exceeded. Moreover, constraints (3.21) ensure that only coalitions which are not in Ω_q are considered. When seeking all tight constraints (3.11) in Step 5 of Algorithm 1, the set of constraints (3.21) also contains constraints associated with previously identified coalitions for which (3.11) is tight. Finally, the binary requirements on the decision variables are enforced through (3.22)-(3.24).

Given a solution (x', z', s') to model (3.17)-(3.24), the selected coalition S is given by

$$S = \{i \in N \mid s'_i = 1\}. \tag{3.25}$$

If (x', z', s') is optimal, its best-case profit is derived from the solution as:

$$v^B(S) = \sum_{l \in M} p^o_l z'_l - \sum_{r \in R} c_r x'_r + \sum_{i \in N} s'_i p^c_i. \tag{3.26}$$

Note that model (3.17)-(3.24) can also be used to compute the values of $v^B(S)$ and $v^W(S)$ whenever necessary. Indeed, $v^B(S)$ with $S \subseteq N$ can be obtained by solving (3.17)-(3.24) with $s_i = 1$ for $i \in S$ and $s_i = 0$ for $i \in N \setminus S$. Moreover,

given an optimal solution (x', z') to $v^B(N \backslash S)$, $v^W(S)$ can be determined by solving (3.17)-(3.24) with $z_l = 0$ for each $l \in M$ such that $z'_l = 1$, $s_i = 1$ for $i \in S$ and $s_i = 0$ for $i \in N \backslash S$.

3.3.2 A branch-price-and-cut algorithm for the subproblem

We propose a branch-price-and-cut algorithm for solving the row generation subproblem (3.17)-(3.24) which potentially involves a large number of route variables. Branch-price-and-cut is a branch-and-bound algorithm where column generation is applied to solve the linear relaxations encountered in the search tree and cuts are added as needed to strengthen them. In this context, the linear relaxation of (3.17)-(3.24) is called the master problem. Its solution by column generation proceeds iteratively as follows. At each iteration, a so-called restricted master problem, which is defined as the master problem limited to a subset of its variables, is first solved to yield an optimal pair of primal and dual solutions. Then, to check the optimality of this solution for the complete master problem, a pricing problem that searches for positive reduced cost variables is solved. If such variables are found, they are added to the restricted master problem before starting a new iteration. Otherwise, column generation stops with an optimal solution to the master problem and a valid upper bound for the current branch-and-bound node. A detailed description of the pricing problem is given in Section 3.3.2.1. To improve the upper bound provided by the master problem solution, the adapted rounded capacity cuts described in Section 3.3.2.2 are generated. Finally, to derive integer solutions, we apply the branching strategy presented in 3.3.2.3 if required.

3.3.2.1 Pricing problem

The column generation pricing problem is used to identify route variables with a positive reduced cost or to show that no such variable exists. Let $\lambda_v \geq 0$, $v \in V^c$, $\mu_v \geq 0$, $v \in V^o$, and σ denote the dual variables associated with constraints (3.18), (3.19), and (3.20), respectively. The reduced cost $\tilde{c}_r$ of a route variable x_r, $r \in R$, can be expressed as

$$\tilde{c}_r = \sum_{(v,w) \in A_r} \tilde{c}_{vw} - \sigma, \tag{3.27}$$

where $A_r \subset A$ is the subset of arcs included in route r and the modified arc costs $\tilde{c}_{vw}$ for $(v, w) \in A$ are given by

$$\tilde{c}_{vw} = \begin{cases} -c_{vw} + \frac{\lambda_v + \lambda_w}{2} & \text{if } v, w \in V^c \\ -c_{vw} + \frac{\mu_v + \lambda_w}{2} & \text{if } v \in V^o, w \in V^c \\ -c_{vw} + \frac{\lambda_v + \mu_w}{2} & \text{if } v \in V^c, w \in V^o \\ -c_{vw} + \frac{\mu_v + \mu_w}{2} & \text{if } v, w \in V^o \end{cases} . \tag{3.28}$$

As proposed for many vehicle routing problems (see Costa et al., 2019), we model the pricing problem as an elementary shortest path problem with resource constraints (ESPPRC, see Irnich and Desaulniers, 2005), namely, a load resource to account for vehicle capacity and customer resources for imposing route elementarity. Moreover, we solve the pricing problem using a labelling algorithm in which the labels are stored in buckets corresponding to demand and customer pairs, similar to the algorithm proposed by Martinelli et al. (2014). Here, we enforce route elementarity by means of the unreachable resources introduced by Feillet et al. (2004).

Given that the computational complexity of the ESPPRC is NP-hard in the strong sense, we apply several well-known acceleration strategies with regard to the pricing problem. First of all, we relax the master problem by allowing non-elementary routes. More specifically, we consider the *ng*-route relaxation of Baldacci et al. (2011), which has been shown to provide a good trade-off between weaker upper bounds for the master problem and reduced computational effort to solve the pricing problem. Given a predefined neighbourhood $U_v \subset V^c \cup V^o$ for each customer $v \in V^c \cup V^o$ (say, its 16 closest customers), a non-elementary route that contains a cycle starting with customer w can be generated if and only if this cycle contains a node v such that $w \notin U_v$. With such a relaxation, the coefficients a_{rv} may take values larger than one.

We also apply decremental state space relaxation as proposed by Righini and Salani (2008). Here, we initially allow non *ng*-routes to be generated by only considering a subset of the unreachable resources. However, if a non *ng*-route is generated, then the subset of unreachable resources is enlarged such that this route can be no longer generated, after which the pricing problem is solved again.

Finally, we first solve the pricing problem in a heuristic manner to find any positive reduced cost variables. Only if no routes have been generated by solving the problem heuristically, we solve the pricing problem to optimality. Moreover, we also terminate the labelling algorithm early if the number of routes with a positive reduced cost which have been identified exceeds a certain threshold. Appendix 3.B describes in detail the column generation process.

3.3.2.2 Valid inequalities

To limit the number of nodes in the search and usually the overall computational time, it is common to improve the upper bound obtained from the master problem by adding valid inequalities. Before branching, we first consider an adaptation of the well-known rounded capacity cuts (Laporte et al., 1985). Consider a subset of customers $W \subseteq V^c \cup V^o$ and let $k(W) = \lceil \sum_{w \in W} d_w/Q \rceil$ be a lower bound on the number of vehicles required to service the customers in W. The traditional rounded capacity cuts would write as follows:

$$\sum_{(v,w) \in A: v \in W, w \notin W} \left(\sum_{r \in R} b_r^{vw} x_r \right) \geq k(W), \tag{3.29}$$

where b_r^{vw} indicates how often route $r \in R$ uses arc $(v, w) \in A$. It stipulates that at least $k(W)$ vehicles must traverse an arc entering in W. This inequality would be valid only if all customers in W must be visited. This is not the case for the row generation subproblem. Therefore, we develop a modified version of the rounded capacity cuts that reduce the right-hand side according to the selected players and selected optional customer clusters.

Let $k^o(l) = \lceil \sum_{v \in V_l^o} d_v/Q \rceil$ (resp., $k^c(i) = \lceil \sum_{v \in V_i^c} d_v/Q \rceil$) be a lower bound on the number of vehicles required to service the optional (resp. contract) customers in cluster V_l^o for $l \in M$ (resp., set V_i^c for $i \in N$). For a subset of customers $W \subseteq V^c \cup V^o$, we formulate the corresponding modified rounded capacity cut as follows:

$$\sum_{(v,w) \in A: v \in W, w \notin W} \left(\sum_{r \in R} b_r^{vw} x_r \right) \geq k(W) - \sum_{l \in M(W)} (1 - z_l) k^o(l) - \sum_{i \in N(W)} (1 - s_i) k^c(i), \tag{3.30}$$

where $M(W)$ denotes the subset of optional customer cluster indices $l \in M$ for which $W \cap V_l^o \neq \emptyset$. Similarly, $N(W)$ is the set of players $i \in N$ for which $W \cap V_i^c \neq \emptyset$. Here, we reduce the total number of vehicles required to serve the customers in W if either an optional customer cluster in $M(W)$ or the contracted customer set of a player in $N(W)$ is not covered exactly once in a solution. In that case, the number of vehicles required for serving W is reduced by a lower bound on the number of vehicles required to serve the customers in the corresponding cluster or set. Note that if $z_l = 1$ for all subsets in $M(W)$ and $s_i = 1$ for all players in $N(W)$, then (3.30) is equivalent to (3.29).

Proposition 1. *Let χ be the set of feasible solutions to model* (3.17)-(3.24)*. Inequal-*

ity (3.30) *is valid for conv*(χ)*, the convex hull of* χ*.*

Proof: To show that inequality (3.30) is valid for $conv(\chi)$, it is sufficient to show that any solution $S \in \chi$ satisfies (3.30). Let $\hat{S} = (\hat{x}, \hat{s}, \hat{z}) \in \chi$. Let $W(\hat{S}) \subseteq W$ be the subset of customers in W that are visited in solution $\hat{S}$ and $\bar{W}(\hat{S}) = W \setminus W(\hat{S})$ the subset of those that are not visited. First, let us deduce an upper bound on $k(W)$:

$$\begin{aligned}
k(W) &= \left\lceil \frac{\sum_{w \in W} d_w}{Q} \right\rceil \\
&= \left\lceil \frac{\sum_{w \in W(\hat{S})} d_w}{Q} + \sum_{l \in M(W)} \left(\frac{\sum_{w \in \bar{W}(\hat{S}) \cap V_l^o} d_w}{Q} \right) + \sum_{i \in N(W)} \left(\frac{\sum_{w \in \bar{W}(\hat{S}) \cap V_i^c} d_w}{Q} \right) \right\rceil \\
&\leq \left\lceil \frac{\sum_{w \in W(\hat{S})} d_w}{Q} \right\rceil + \sum_{l \in M(W)} \left\lceil \frac{\sum_{w \in \bar{W}(\hat{S}) \cap V_l^o} d_w}{Q} \right\rceil + \sum_{i \in N(W)} \left\lceil \frac{\sum_{w \in \bar{W}(\hat{S}) \cap V_i^c} d_w}{Q} \right\rceil \\
&\leq \left\lceil \frac{\sum_{w \in W(\hat{S})} d_w}{Q} \right\rceil + \sum_{l \in M(W)} \left\lceil \frac{\sum_{v \in V_l^o} d_v}{Q} \right\rceil + \sum_{i \in N(W)} \left\lceil \frac{\sum_{v \in V_i^c} d_v}{Q} \right\rceil \\
&= k(W(\hat{S})) + \sum_{l \in M(W)} (1 - \hat{z}_l) k^o(l) + \sum_{i \in N(W)} (1 - \hat{s}_i) k^c(i). \qquad (3.31)
\end{aligned}$$

The first equality comes from the definition of $k(W)$, whereas the second arises from a partition of the customers in W into the customers in $W(\hat{S})$, the uncovered optional customers, and the uncovered contract customers. The first inequality stems from the properties of the ceiling function. The second one ensues from the relations $(\bar{W}(\hat{S}) \cap V_l^o) \subseteq V_l^o$, $l \in M(W)$, and $(\bar{W}(\hat{S}) \cap V_i^c) \subseteq V_i^c$, $i \in N(W)$. Finally, the last equality is obtained from the definitions of $k(W(\hat{S}))$, $k^o(l)$, and $k^c(i)$ combined with the facts that $\hat{z}_l = 0$ for all $l \in M(W)$ and $\hat{s}_i = 0$ for all $i \in N(W)$.

Second, using the upper bound (3.31), we find the following lower bound on $k(W(\hat{S}))$:

$$k(W(\hat{S})) \geq k(W) - \sum_{l \in M(W)} (1 - \hat{z}_l) k^o(l) - \sum_{i \in N(W)} (1 - \hat{s}_i) k^c(i). \qquad (3.32)$$

Third, because only the customers in $W(\hat{S})$ are visited in $\hat{S}$, we can replace $k(W)$

by $k(W(\hat{S}))$ in the traditional inequality (3.29) for W:

$$\sum_{(v,w)\in A: v\in W, w\notin W} \left(\sum_{r\in R} b_r^{vw} x_r \right) \geq k(W(\hat{S})), \tag{3.33}$$

and use the lower bound (3.32) to deduce inequality (3.30). □

Next, we provide an example to illustrate that the modified capacity inequalities (3.30) can also cut off part of the feasible region if some of the player variables s_i or the cluster variables z_l are non-integral.

Example 3.3.1. *Consider the 2-player instance of the JNVRGOC with $N = \{1, 2\}$ as presented in Figure 3.3. Let $V_1^c = \{1\}$ and $V_2^c = \{2\}$ with $d_1 = d_2 = 1$. Moreover, let $k_i = 1$ for each player $i \in N$ and assume that all vehicles have a capacity $Q = 2$. There is a single cluster of optional customers $V_1^o = \{3\}$ with $d_3 = 1$.*

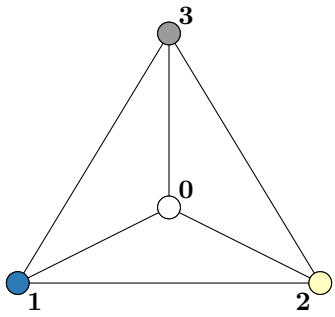

Figure 3.3: A 2-player instance of the JNVRGOC.

A solution to the linear relaxation of (3.17)-(3.24) *could have the following non-zero route variables $x_{(0,1,2,0)} = \frac{2}{3}, x_{(0,1,3,0)} = \frac{1}{3}$ and $x_{(0,2,3,0)} = \frac{1}{3}$. Let $W = \{1, 2, 3\}$ be such that $s_1 = s_2 = 1$ and $z_1 = \frac{2}{3}$. It follows that the right-hand side of inequality* (3.30) *equals $\frac{5}{3}$ while its left-hand side equals $\frac{4}{3}$. Hence, this modified rounded capacity inequality is violated.*

To separate the modified rounded capacity cuts, we first run the heuristics provided by the package of Lysgaard et al. (2004) to separate the regular rounded capacity cuts and, then, check if each subset of customers returned yields a violated modified rounded capacity cut. A maximum of 100 cuts is added at a time.

To conclude this section, we would like to mention that we also implemented the limited-memory subset row cuts (lm-SRCs) of size 3, originally proposed by

Pecin et al. (2017a). In general, the lm-SRCs have been shown to provide a good trade-off between an increase in computational time and a decrease in integrality gap (Pecin et al., 2017b). However, we observed that they did not improve the computational performance of our algorithm. We believe that this is caused by the relatively small number of customers that can be included in a feasible route for our instances. Therefore, these cuts are not present in the final version of our algorithm.

3.3.2.3 Branching strategy

To derive integer solutions, we use the following hierarchical branching strategy. When an optimal solution to the master problem is fractional and branching is required, this strategy selects the first branching entity taking a fractional value according to the following order:

1. The sum of the player variables $\sum_{i \in N} s_i$;
2. A player variable s_i;
3. The sum of the cluster variables $\sum_{l \in M} z_l$;
4. A cluster variable z_l;
5. The total number of vehicles used $\sum_{r \in R} x_r$;
6. An arc-flow variable $\sum_{r \in R} b_r^{vw} x_r$.

When multiple entities are available, the one taking the closest fractional value to 0.5 is chosen. Moreover, when choosing a node to branch on, we choose the node with the largest upper bound.

3.3.3 Acceleration strategies

In this section, we describe two acceleration strategies that can be applied to the basic row generation algorithm described in Section 3.3. First, throughout the algorithm, we store all generated columns in a pool which is shared among all branch-price-and-cut procedures. Hence, this pool is not only used when solving row generation subproblems, but also when determining best-case profits and worst-case profits. In this manner, we can generally provide a good initial set of columns for any call to a branch-price-and-cut algorithm.

Secondly, for the row generation subproblem, we terminate the branch-price-and-cut procedure as soon as a coalition corresponding to a violated or tight rationality

constraint is identified, when an integral solution with a non-negative value is found. Here, it should be noted that the profit which has been determined for this coalition is not necessarily optimal. Hence, we still need to determine the best-case profit of the identified coalition, after which we add the corresponding best-case rationality constraint.

3.4 Computational results

In this section, we first describe the instances used for our computational experiments and present our experimental plan. Then, we report the computational results obtained and discuss the quality of the computed solutions.

3.4.1 Instances and experimental setup

We constructed instances of the JNVRGOC based on the CVRP instances of Augerat (1995) as follows. Specifically, we considered the instances of the A-set which involve between 31 and 79 customers in total. For each of these instances, we defined nine JNVRGOC instances by considering 3, 5 and 7 players as well as 3, 5 and 7 clusters of optional customers. Recall that the number of customers is given by $|V| - 1$. We assigned the first $n \left\lfloor \frac{0.7(|V|-1)}{n} \right\rfloor$ customers to the players, such that $i(v) = (v \mod n) + 1$ with $v \in \left\{1, \ldots, n \left\lfloor \frac{0.7(|V|-1)}{n} \right\rfloor\right\}$. The remaining customers $\left\{n \left\lfloor \frac{0.7(|V|-1)}{n} \right\rfloor + 1, \ldots, |V|\right\}$ were divided among the clusters of optional customers such that $l(v) = \left[\left(v - \left(n \left\lfloor \frac{0.7(|V|-1)}{n} \right\rfloor + 1\right)\right) \mod m\right] + 1$ with $v \in \left\{n \left\lfloor \frac{0.7(|V|-1)}{n} \right\rfloor + 1, \ldots, |V|\right\}$.

Consider an instance of the JNVRGOC with players $\{1, \ldots, n\}$, contract customers V^c and optional customers V^o. For each instance, we assign each player $i \in N$ precisely enough vehicles to cover its own customers: $k_i = \left\lceil \frac{\sum_{v \in V_i^c} d_v}{Q} \right\rceil$. Moreover, given a cluster of optional customers V_l^o with $l \in M$, we define the reward p_l^o as $-0.8 \cdot \text{GCPTPCC}(\emptyset, V_l^o)$, where $-\text{GCPTPCC}(\emptyset, V_l^o)$ denotes the optimal value of a CVRP defined over the customers in V_l^o.

We solved these instances using four algorithms. The first algorithm, denoted by RG, is the row generation algorithm described in Section 3.3.2. To determine the performance of the modified capacity cuts (3.30), we also consider an algorithm, denoted by RG NO MCC, which is the row generation algorithm RG but excluding the modified rounded capacity cuts. Moreover, we also execute two enumeration

algorithms ENUM RG and ENUM ALL. The former enumerates a priori the best-case profit for each coalition, and only determine the worst-case profit for coalitions associated with tight rationality constraints at each iteration of Algorithm 1. In the latter, we enumerate the best-case profit and the worst-case profit for each coalition, which allows us to solve (3.4)-(3.8) to optimality. Algorithms ENUM RG and ENUM ALL are used as benchmarks with respect to computation times and optimal solution values, respectively.

All algorithms were implemented in C++, using IBM ILOG Optimiser 12.8.0 to solve any linear programming problems. The computations were conducted on a single thread of an Intel Xeon Gold 6130 @ 2.1 GHz with a turbo of 3.7 GHz and 16 GB of RAM. Unless otherwise specified, a time limit of 7,200 seconds (2 hours) is enforced for solving each instance.

3.4.2 Computational performance

To compare the performance of the four tested algorithms, we only consider 10 of the smallest Augerat instances, with only 3 and 7 players, and 3 and 7 clusters of optional customers. Specifically, we consider Augerat instances A-n32-k05, A-n33-k06, A-n34-k05, A-n36-k05, A-n37-k06, A-n38-k05, A-n39-k05, A-n39-k06, A-n44-k06 and A-n45-k06, leading to a total of 40 JNVRGOC instances.

Summarised results obtained for these small instances are shown in Table 3.1. They are grouped by the number of players n and the number of optional customer clusters m. For each algorithm, two columns are presented to report the number of instances solved to optimality within the time limit (OPT), and the average time taken to solve these instances (T).

		RG NO MCC		RG		ENUM RG		ENUM ALL	
n	m	OPT	T(s)	OPT	T(s)	OPT	T(s)	OPT	T(s)
3	3	8	2783	10	403	10	279	10	299
3	7	8	1757	10	388	10	249	10	278
7	3	2	2007	8	3124	1	3457	1	6339
7	7	2	1648	9	4040	1	3297	1	6054

Table 3.1: Comparative results on small JNVRGOC instances.

For each (n, m) combination, we were able to solve significantly more instances with algorithm RG than with algorithm RG NO MCC. For example, we solved 8 of the 10 instances with 7 players and 3 clusters of optional customers with algorithm RG, compared to only 2 instances with algorithm RG NO MCC. This clearly demonstrates the effectiveness of the modified capacity cuts. Moreover, the effectiveness of the row

generation algorithm is illustrated by comparing algorithms RG NO MCC and ENUM RG. Here, we can see that even without the capacity cuts, algorithm RG NO MCC is able to solve more instances than ENUM RG for the 7 player instances. For these instances, the difference is much more striking when comparing algorithms RG and ENUM RG

Furthermore, it can be seen that algorithm RG does not outperform the enumeration algorithms ENUM RG and ENUM ALL for instances with 3 players. For these instances, only 6 coalitions have to be considered, which causes most, if not all, coalitions to be generated by the row generation algorithm with the additional overhead of solving the row generation subproblem multiple times. However, for the 7-player instances, we are able to solve significantly more instances with algorithm RG in significantly less time when compared to ENUM RG and ENUM ALL. Here, it should be noted that ENUM ALL is an exact algorithm whereas algorithms RG and RG ENUM may yield a solution that is not feasible to (3.4)-(3.8).

Next, we discuss the performance of algorithm RG in more detail. A summary with respect to the performance of this algorithm for each combination (n, m) is provided in Table 3.2. We consider the following summary statistics: the number of instances solved (OPT), the average run time in seconds (T), and the average number of rationality constraints of the form (3.11) that have been generated (I). Moreover, we also consider the proportions of the time spent solving the row generation subproblems (T_i) and all other routing problems (T_g), as well as the average number of best-case (BP) and worst-case (WP) problems solved. Finally, we also list the average number of branch-and-bound nodes explored (BBN) and the modified capacity cuts generated (MCC). It should be noted that the averages are computed only over the instances which were solved before reaching the time limit. The complete computational results for each individual instance can be found in Appendix 3.C.

n	m	OPT	$T(S)$	I	$T_i(\%)$	$T_g(\%)$	BP	WP	BBN	MCC
3	3	21	1014	2.1	59	41	5.1	2.0	666	485
3	5	19	1274	2.6	53	47	5.5	2.3	843	511
3	7	19	1277	2.7	57	43	5.5	2.6	642	634
5	3	12	1569	7.7	55	44	12.8	4.4	1395	689
5	5	13	1910	8.2	56	44	13.3	5.0	1577	838
5	7	12	2193	11.4	57	43	15.7	4.9	1489	870
7	3	9	3098	17.3	64	36	24.8	7.0	2720	1280
7	5	8	3363	20.1	59	41	27.4	7.1	2628	1117
7	7	9	4040	21.0	56	44	27.1	7.1	2694	1151

Table 3.2: Summary results of the algorithm RG on the JNVRGOC instances.

On average 42%, 30% and 15% of all possible coalitions were generated by al-

gorithm RG on the 3, 5 and 7-player instances that have been solved, respectively. Here, it should be noted that the maximum number of coalitions which could have been generated is equal to 6, 30 and 126 for the 3, 5 and 7-player instances, respectively. Clearly, the percentage of coalitions generated decreases as the number of players increases, which illustrates the effectiveness of a row generation procedure for instances with a relatively high number of players. Moreover, the average size of coalitions generated is 1.7, 3.0 and 3.4 for the 3, 5 and 7-player instances, respectively.

The time spent on the heuristic is almost split equally between solving the row generation subproblems and determining best-case and worst-case profits. However, we can observe that less subproblems are solved than the number of profits computed, which shows that computing profits is on average less time-consuming. This is not surprising given that, to do so, some player and cluster variables are fixed. Moreover, it is interesting to note that, for 3-player instances, the best-case profits have been determined for almost every coalition. This is clearly not the case for the 5 and 7-player instances. Finally, we can observe that the total numbers of branch-and-bound nodes explored and cuts generated increase with the number of players but not with the number of optional customer clusters.

To conclude this section, let us remark that we also considered instances in which each player was assigned an additional vehicle compared to the minimum number of vehicles required. However, these instances did not provide any additional insights and, therefore, their results have not been included in this chapter.

3.4.3 Solution quality

In this section, we assess the quality of the solutions produced by algorithm RG. To do so, we compare them with the optimal solutions obtained for model (3.4)-(3.8), namely, those computed by algorithm ENUM RG. To find more optimal solutions with this algorithm and, thus, make a comparison on a larger number of instances, we have increased its time limit to 36,000 seconds. The time limit imposed for algorithm RG has, however, remained equal to 7,200 seconds.

The results are summarised in Table 3.3. For each combination of n and m, 26 instances were considered. For each algorithm, we report the number of instances solved to optimality within their respective time limit (OPT) and the average objective value of the computed solution, denoted α' for algorithm RG and α for algorithm ENUM ALL. Moreover, when algorithm ENUM ALL was able to compute the best-case and worst-case profits of all coalitions, it becomes possible to determine if the allocation obtained by algorithm RG is feasible for model (3.4)-(3.8) and, if so, what is the

objective value of this allocation according to this model. The columns SH and INF indicate the number of instances for which both algorithms terminated before their respective time limit and, among these instances, the number of allocations computed by algorithm RG that are infeasible for model (3.4)-(3.8), respectively. Indeed, such an allocation y can be infeasible if there exists no α value such that all constraints (3.5) hold, i.e., if there exists a coalition S for which $y(S) < v^W(S)$. Finally, for both algorithms, column α_{sh} reports the average value of the computed coalitions according to model (3.4)-(3.8).

Because algorithm ENUM ALL computes an optimal solution, the value of α_{sh} obtained by algorithm RG is always less than or equal to that derived by algorithm ENUM ALL. The gap between these two values indicates the quality of the solution computed by algorithm RG. Again, the complete results for each instance are provided in Appendix 3.C.

		RG					ENUM ALL		
n	m	OPT	α'	SH	INF	α_{sh}	OPT	α	α_{sh}
3	3	21	1.00	21	0	1.00	24	1.00	1.00
3	5	19	0.96	19	1	0.96	21	0.96	0.97
3	7	19	0.91	19	1	0.90	23	0.92	0.91
5	3	12	0.91	12	3	0.95	13	0.89	0.96
5	5	13	0.81	13	5	0.79	14	0.73	0.86
5	7	12	0.77	11	4	0.64	12	0.71	0.80
7	3	9	0.71	3	2	0.56	3	0.61	0.63
7	5	8	0.65	4	1	0.38	4	0.59	0.56
7	7	9	0.63	4	1	0.46	4	0.56	0.53

Table 3.3: Summarised results on all JNVRGOC instances.

From these results, we make the following observations. First, if we consider the objective value α', we see that players are on average allocated at least 96%, 83% and 66% of their best-case profit for the 3, 5 and 7-player instances, respectively. Here, it is interesting to note that players are on average allocated less profit relative to their best-case profit if the size of the grand-coalition increases. A similar pattern occurs with respect to the optimal value α, which is on average 96%, 78% and 58% for the 3, 5 and 7-player instances, respectively. We believe that the decrease in relative allocated profit is partially accounted for by the structure of the instances. For example, consider a given instance with a fixed number of optional customer clusters. Recall that each player has precisely enough vehicles to cover its contract customers. Hence, the overall number of vehicles is likely to be larger for a larger number of players as the customers are served less efficiently. The larger number of vehicles may cause smaller coalitions to already be able to serve most clusters

of optional customers, thereby reducing the value created by having many players collaborating. As a consequence, each coalition is allocated less when compared to their best-case profit.

Moreover, we observe that the worst-case profit core is non-empty for all instances solved by the algorithm ENUM ALL. This suggests that the allocation is suitable for use in practice as there is always an incentive to cooperate. Here, it should be noted that the allocation computed by algorithm RG may not be in the worst-case profit core, even if it is non-empty. Specifically, in 3%, 33% and 36% of the cases, this allocation is not in the worst-case profit core for the 3, 5 and 7-player instance, respectively, as indicated by the values reported in the columns SH and INF. However, for the allocations that are in the worst-case core, the objective value is comparable to that of the optimal allocation computed by algorithm ENUM ALL (see both columns α_{sh}). The average deviations between these values are 1%, 7% and 13% on average for the 3, 5 and 7-player instances, respectively.

For each computed allocation, the carriers are given an incentive to collaborate as they are allocated at least as much profit when compared to their worst-case profit. However, in theory, the collaboration between multiple carriers may be hindered if an allocation is proposed which is not in the worst-case core. If this is the case, a subset of carriers may want to leave the collaboration. Nonetheless, in practice, it is very difficult for a subset of carriers to verify whether an allocation is in the worst-case core. Moreover, if a subset of carriers identifies that a given allocation is not in the worst-case core, the corresponding violated rationality constraints can be incorporated after which a new allocation can be easily determined.

3.5 Concluding remarks

We proposed a heuristic row generation algorithm to determine core allocations for the JNVRGOC, based on both the worst-case profit and the best-case profit. Our computational experiments have shown that an enumeration algorithm which requires computing the worst-case and best-case profits for all possible coalitions becomes rapidly inefficient as the number of players increases. At the opposite, the performance of our row generation approach is less impacted by such an increase. Moreover, our results suggest that the modified capacity cuts do significantly reduce the computation time for the row generation algorithm. As for the quality of the solutions produced by the proposed heuristic, we find that the average deviation of their values with respect to those of the optimal allocations is relatively small. On the other

hand, our approach might sometimes fail to find an allocation in the worst-case core when it is not empty.

In order to improve the performance of our algorithm, further research are required to determine good lower bounds for the worst-case profits. We believe that such developments would help to solve larger problem instances in terms of the number of players and the number of customers. Finally, as already mentioned, our model and algorithm can theoretically be adapted to the multiple-depot case. However, the computational performance is yet to be observed.

Appendix

3.A Emptiness of the imputation set

Consider the 3-player instance of the JNVRGOC with $N = \{1, 2, 3\}$ as presented in Figure 3.4. Let $V_1^c = \{1\}, V_2^c = \{2\}$ and $V_3^c = \{3\}$ with $d_1 = d_2 = d_3 = 1$. Moreover, let $k_i = 1$ for each player and assume all vehicles to have a capacity of 2. The set of optional customers is defined such that $V_1^o = \{4\}$ and $V_2^o = \{5\}$ with $d_4 = 1, d_5 = 2$, $p_1^o = 5$ and $p_2^o = 8$, respectively. Assuming that $c_{ij} = c_{ji}$ for all arcs $(i, j) \in A$, the arc costs are given in Figure 3.4 as edge costs (those not specified are equal to 2).

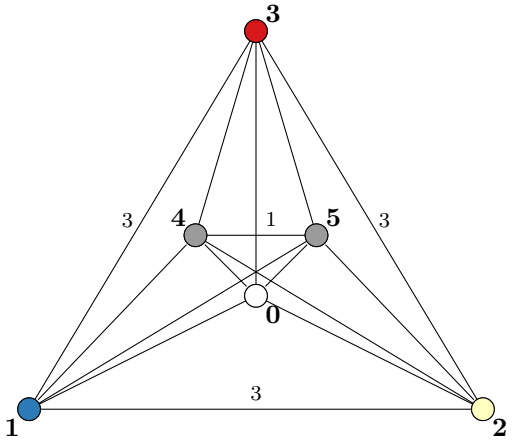

Figure 3.4: A 3-player instance of the JNVRGOC.

We will show that the imputation set is empty with respect to the worst-case profit, i.e, $\alpha = 0$. In order to determine the worst-case profit $v^W(\{1\}) = v^W(\{2\}) = v^W(\{3\})$, we first have to determine the best-case profit of the coalitions of size

two, i.e., we have to determine $v^B(\{1,2\}) = v^B(\{1,3\}) = v^B(\{2,3\})$. Given a two-player coalition, it holds that it is optimal to visit optional customer 5. For example, consider the coalition $\{1,2\}$. Using routes $(0,1,2,0)$ and $(0,5,0)$, a reward of 8 is collected, while the routes cost 7 and 4, respectively. Moreover, note that $p_1^c = p_2^c = p_3^c = 4$. Hence, $v^B(\{1,2\}) = v^B(\{1,3\}) = v^B(\{2,3\}) = 8 - 7 - 4 + 4 + 4 = 5$. It follows that the singleton coalitions $\{1\}, \{2\}$ and $\{3\}$ are only allowed to visit optional customer 4. In the worst-case solution, a singleton coalition visits its contract customers as well as optional customer 4 on the same trip for a reward of 5. Hence, $v^W(\{1\}) = v^W(\{2\}) = v^W(\{3\}) = 5 - 6 + 4 = 3$. The optimal profit of the grand-coalition is given by visiting every customer using the routes $(0,1,4,0)$, $(0,2,3,0)$ and $(0,5,0)$ at a routing cost of $6 + 7 + 4$. This results in a profit $v^B(N) = v^W(N)$ of $5 + 8 - 6 - 7 - 4 + 4 + 4 + 4 = 8$. Hence, it follows that $v^W(\{1\}) + v^W(\{2\}) + v^W(\{3\}) = 9 > 8 = v^W(N)$, which implies that the imputation set $I(\langle N, v^0 \rangle)$ is empty, and thus $I(\langle N, v^\alpha \rangle) = \emptyset$ for all $\alpha \in [0,1]$.

3.B Details on the column generation algorithm

The column generation algorithm proceeds as follows. At each iteration, the restricted master problem is solved. If the number of columns it contains exceeds 5000, then all positive reduced cost columns are removed and stored in a pool of columns. Then we seek negative reduced cost columns using different procedures. As soon as one procedure finds negative reduced cost columns, these columns are added to the restricted master problem which is then re-optimised to start a new iteration. The procedures, which are called in the following order, are as follows:

1. Check if any of the columns in the pool of previously removed columns have a negative reduced cost;
2. Apply heuristic labelling algorithm HLA1;
3. Apply heuristic labelling algorithm HLA2;
4. Apply heuristic labelling algorithm HLA3;
5. Apply heuristic labelling algorithm HLA4;
6. Apply heuristic labelling algorithm HLA5;
7. Apply heuristic labelling algorithm HLA6;

8. Apply the exact labelling algorithm.

All labelling algorithms solve the *ng*-route relaxed pricing problem, where the neighbourhoods consist of the 16 closest customers. In heuristics HLA1 to HLA4, we do not apply decremental state-space relaxation because dominance is performed only on the reduced cost of a route and a network of reduced size is considered. In HLA1 to HLA4, we restrict the arcs to the arcs between each customer and its 2, 4, 8 and 12 closest neighbours, respectively. In HLA5, HLA6 and the exact labelling algorithm, decremental state-space relaxation is applied as full dominance is considered. However, in HLA5 and HLA6, a reduced network is considered, namely, we restrict the arcs to the arcs between each customer and its 2 and 4 closest neighbours, respectively.

3.C Detailed results for the algorithms *RG* and *ENUM ALL*

In Tables 3.4-3.12, we present detailed results for each instance solved with algorithms RG and ENUM ALL. For each instance, we report the following statistics for algorithm RG:

α': The objective value returned by algorithm RG;

α: The objective value of the computed allocation with respect to (3.4)-(3.8);

T: The overall computation time in seconds;

I: The number of rationality constraints of the form (3.11) generated;

T_i: The percentage of time taken to solve the row generation subproblem;

T_g: The percentage of time taken to solve all routing problems excluding the row generation subproblem;

BP: The number of best-case profits computed, including that for the grand-coalition;

WP: The number of worst-case profits computed;

BBN: The overall number of nodes explored in all branch-price-and-cut procedures;

MCC: The overall number of modified capacity cuts generated.

For algorithm ENUM ALL, we only report the α value and the time T in seconds.

	RG										ENUM ALL	
Instance	α'	α	T(s)	I	T_i(%)	T_g(%)	BP	WP	BBN	MCC	α	T(s)
A-n32-k05	1.00	1.00	72	3	85	14	6	2	101	112	1.00	26
A-n33-k06	1.00	1.00	32	2	43	57	5	2	222	166	1.00	23
A-n34-k05	1.00	1.00	37	2	52	47	5	2	83	95	1.00	50
A-n36-k05	1.00	1.00	295	2	83	17	5	2	199	106	1.00	92
A-n37-k05	1.00	1.00	108	2	74	26	5	2	116	165	1.00	46
A-n37-k06	1.00	1.00	1207	2	26	74	5	2	2336	749	1.00	1083
A-n38-k05	1.00	1.00	54	2	75	25	5	2	51	58	1.00	20
A-n39-k05	1.00	1.00	1351	2	28	71	5	2	1240	678	1.00	1158
A-n39-k06	1.00	1.00	116	3	71	29	6	2	121	177	1.00	38
A-n44-k06	1.00	1.00	397	2	78	22	5	2	400	465	1.00	239
A-n45-k06	1.00	1.00	464	2	61	39	5	2	503	408	1.00	263
A-n45-k07	1.00	1.00	2436	2	20	80	5	2	1910	1072	1.00	3294
A-n46-k07	1.00	1.00	877	2	51	49	5	2	391	965	1.00	795
A-n48-k07	1.00	1.00	271	2	80	20	5	2	143	269	1.00	105
A-n53-k07	—	—	—	0	0	100	1	0	3217	609	—	—
A-n54-k07	1.00	1.00	1377	2	73	27	5	2	134	785	1.00	326
A-n55-k09	1.00	1.00	199	2	78	21	5	2	98	457	1.00	64
A-n60-k09	1.00	1.00	1110	2	72	28	5	2	242	591	1.00	425
A-n61-k09	1.00	1.00	1926	2	69	31	5	2	1660	996	1.00	1381
A-n62-k08	—	—	—	2	33	67	5	1	1110	1537	1.00	4528
A-n63-k09	—	—	—	1	5	95	2	0	2349	1340	1.00	8006
A-n63-k10	1.00	1.00	1484	2	19	81	5	2	697	1029	1.00	1349
A-n64-k09	1.00	1.00	6758	2	8	92	5	2	2993	498	1.00	10347
A-n65-k09	1.00	1.00	726	2	88	12	5	2	354	340	1.00	180
A-n69-k09	—	—	—	2	12	88	3	0	2956	441	1.00	6987
A-n80-k10	—	—	—	0	0	100	1	0	1203	1445	—	—

Table 3.4: Results on JNVRGOC instances with 3 players and 3 clusters of optional customers.

	RG										ENUM ALL	
Instance	α'	α	T(s)	I	T_i(%)	T_g(%)	BP	WP	BBN	MCC	α	T(s)
A-n32-k05	1.00	1.00	53	2	75	25	5	2	89	111	1.00	24
A-n33-k06	0.81	0.81	84	3	31	67	7	3	470	197	0.81	53
A-n34-k05	1.00	1.00	33	2	58	42	5	2	64	190	1.00	24
A-n36-k05	1.00	1.00	91	2	84	16	5	2	72	82	1.00	29
A-n37-k05	0.98	0.98	150	3	79	21	5	3	121	100	0.98	64
A-n37-k06	0.84	0.72	1807	5	19	81	6	3	3358	835	0.83	1850
A-n38-k05	1.00	1.00	146	2	37	63	5	2	187	110	1.00	107
A-n39-k05	0.86	x	620	4	56	43	6	3	528	600	0.86	456
A-n39-k06	0.91	0.91	368	3	52	46	7	3	276	139	0.91	191
A-n44-k06	1.00	1.00	256	2	85	15	5	2	126	499	1.00	58
A-n45-k06	1.00	1.00	213	2	36	63	5	2	239	407	1.00	193
A-n45-k07	1.00	1.00	886	2	36	64	5	2	1094	911	1.00	733
A-n46-k07	0.87	0.87	500	3	63	35	7	3	318	398	0.87	285
A-n48-k07	1.00	1.00	3567	2	17	83	5	2	3060	1044	1.00	3960
A-n53-k07	—	—	—	0	0	100	1	0	4553	710	—	—
A-n54-k07	—	—	—	3	24	76	6	1	2207	1162	1.00	15106
A-n55-k09	1.00	1.00	257	2	68	31	5	2	198	325	1.00	131
A-n60-k09	1.00	1.00	6215	2	52	48	5	2	1201	1169	1.00	2992
A-n61-k09	1.00	1.00	434	2	69	30	5	2	165	580	1.00	1488
A-n62-k08	—	—	—	0	11	89	2	0	1691	972	—	—
A-n63-k09	—	—	—	0	0	100	1	0	2249	1754	—	—
A-n63-k10	1.00	1.00	6770	3	19	81	5	2	4066	1305	1.00	11710
A-n64-k09	—	—	—	2	5	93	3	0	2677	593	1.00	24825
A-n65-k09	—	—	—	0	0	100	1	0	3017	588	—	—
A-n69-k09	1.00	1.00	1749	3	69	31	6	2	383	711	1.00	885
A-n80-k10	—	—	—	0	94	6	1	0	868	1107	—	—

Table 3.5: Results on JNVRGOC instances with 3 players and 5 clusters of optional customers.

	RG										ENUM ALL	
Instance	α'	α	T(s)	I	T_i(%)	T_g(%)	BP	WP	BBN	MCC	α	T(s)
A-n32-k05	0.92	0.88	69	3	63	36	5	3	142	157	0.92	35
A-n33-k06	0.70	0.70	21	3	57	39	7	3	94	151	0.70	10
A-n34-k05	0.96	0.96	36	3	61	38	5	3	82	85	0.96	25
A-n36-k05	1.00	1.00	147	2	81	18	5	2	122	190	1.00	73
A-n37-k05	0.94	x	199	3	86	14	5	3	132	145	0.94	60
A-n37-k06	0.81	0.81	1148	3	23	75	7	3	1864	1516	0.81	1055
A-n38-k05	0.92	0.92	181	3	45	54	5	3	169	84	0.92	133
A-n39-k05	0.72	0.72	1460	4	52	48	6	3	858	780	0.72	1080
A-n39-k06	1.00	1.00	138	2	80	20	5	2	157	184	1.00	111
A-n44-k06	0.81	0.81	428	3	61	35	7	3	418	555	0.81	151
A-n45-k06	0.83	0.69	249	4	69	31	6	3	153	477	0.79	105
A-n45-k07	—	—	—	0	0	100	1	0	5161	1146	—	—
A-n46-k07	0.81	0.70	618	4	79	21	6	3	254	725	0.81	285
A-n48-k07	0.96	0.96	848	3	85	15	5	3	385	608	0.96	1214
A-n53-k07	1.00	1.00	2404	2	39	61	5	2	949	1132	1.00	4604
A-n54-k07	—	—	—	0	13	87	2	0	3094	1392	0.93	17040
A-n55-k09	1.00	1.00	1267	2	33	67	5	2	1295	584	1.00	1064
A-n60-k09	1.00	1.00	1343	2	72	28	5	2	417	715	1.00	851
A-n61-k09	—	—	—	0	0	100	1	0	3855	916	1.00	25897
A-n62-k08	—	—	—	2	17	83	4	1	1430	1227	1.00	20997
A-n63-k09	1.00	1.00	2418	2	50	50	5	2	641	1862	1.00	7742
A-n63-k10	1.00	1.00	6898	2	14	86	5	2	2539	1391	1.00	6811
A-n64-k09	—	—	—	1	34	66	2	0	2364	837	1.00	7550
A-n65-k09	—	—	—	2	81	19	4	0	5531	774	—	—
A-n69-k09	1.00	1.00	4388	2	24	76	5	2	1536	704	1.00	4180
A-n80-k10	—	—	—	0	0	100	1	0	1455	545	—	—

Table 3.6: Results on JNVRGOC instances with 3 players and 7 clusters of optional customers.

	RG										ENUM ALL	
Instance	α'	α	T(s)	I	T_i(%)	T_g(%)	BP	WP	BBN	MCC	α	T(s)
A-n32-k05	0.81	0.65	263	10	89	11	14	5	342	254	0.74	183
A-n33-k06	1.00	1.00	280	4	34	66	9	4	1581	371	1.00	496
A-n34-k05	1.00	1.00	1381	9	83	17	14	4	2200	772	1.00	4269
A-n36-k05	0.91	x	942	10	48	52	15	5	786	606	0.89	32152
A-n37-k05	0.93	0.93	3659	11	39	61	17	5	2143	880	0.93	5297
A-n37-k06	0.65	x	336	7	69	31	13	5	571	468	0.44	588
A-n38-k05	1.00	1.00	898	5	24	76	10	4	1590	480	1.00	2263
A-n39-k05	1.00	1.00	2193	6	47	53	11	4	995	1171	1.00	11949
A-n39-k06	0.68	x	1009	9	70	30	15	5	693	762	0.54	1320
A-n44-k06	1.00	1.00	1190	7	56	44	12	4	507	580	1.00	2892
A-n45-k06	1.00	1.00	586	10	85	15	14	4	406	476	1.00	514
A-n45-k07	1.00	1.00	6088	4	21	79	9	4	4922	1449	1.00	11912
A-n46-k07	—	—	—	6	92	8	7	0	1615	2040	1.00	33196
A-n48-k07	—	—	—	7	59	41	8	0	2420	2149	—	—
A-n53-k07	—	—	—	4	36	64	8	3	2081	1733	—	—
A-n54-k07	—	—	—	0	0	100	1	0	3350	956	—	—
A-n55-k09	—	—	—	1	18	82	3	0	6057	616	—	—
A-n60-k09	—	—	—	5	28	68	6	0	3613	1495	—	—
A-n61-k09	—	—	—	4	18	82	8	2	3521	1640	—	—
A-n62-k08	—	—	—	4	62	36	5	0	339	2366	—	—
A-n63-k09	—	—	—	0	0	100	1	0	2837	1414	—	—
A-n63-k10	—	—	—	4	26	74	6	0	4403	1331	—	—
A-n64-k09	—	—	—	0	0	100	1	0	2853	896	—	—
A-n65-k09	—	—	—	5	73	27	7	1	1225	2828	—	—
A-n69-k09	—	—	—	0	0	100	1	0	4014	1088	—	—
A-n80-k10	—	—	—	0	0	100	1	0	2257	590	—	—

Table 3.7: Results on JNVRGOC instances with 5 players and 3 clusters of optional customers.

	RG										ENUM ALL	
Instance	α'	α	T(s)	I	$T_i(\%)$	$T_g(\%)$	BP	WP	BBN	MCC	α	T(s)
A-n32-k05	0.73	0.22	152	13	90	10	16	5	278	226	0.63	71
A-n33-k06	0.80	0.80	495	5	28	72	11	5	2096	519	0.80	800
A-n34-k05	0.71	x	3550	10	44	56	16	5	3598	1264	0.34	7697
A-n36-k05	—	—	—	3	11	89	5	0	9472	573	0.69	26227
A-n37-k05	0.76	0.62	1567	11	66	34	14	6	861	492	0.76	2526
A-n37-k06	0.84	0.78	604	11	64	36	17	6	646	547	0.81	1150
A-n38-k05	0.86	x	1444	9	53	47	15	5	1942	470	0.81	2464
A-n39-k05	0.76	x	2255	9	57	43	13	5	863	953	0.69	9228
A-n39-k06	0.76	x	1751	11	80	19	17	6	1123	755	0.62	1283
A-n44-k06	0.88	0.88	800	6	49	51	12	5	911	714	0.88	4346
A-n45-k06	0.38	x	2366	6	42	58	12	5	2197	716	0.18	2296
A-n45-k07	1.00	1.00	1050	5	47	53	10	4	1030	827	1.00	7271
A-n46-k07	—	—	—	2	63	37	4	0	3978	997	—	—
A-n48-k07	—	—	—	9	97	3	10	0	1685	1537	—	—
A-n53-k07	—	—	—	8	76	24	9	0	1435	1526	—	—
A-n54-k07	—	—	—	0	0	100	1	0	3431	875	—	—
A-n55-k09	1.00	1.00	7044	6	55	45	11	4	4480	1784	1.00	27511
A-n60-k09	—	—	—	5	76	24	6	0	1318	2634	—	—
A-n61-k09	1.00	1.00	1757	4	60	40	9	4	482	1629	1.00	11056
A-n62-k08	—	—	—	0	0	100	1	0	2133	1133	—	—
A-n63-k09	—	—	—	0	0	100	1	0	3046	795	—	—
A-n63-k10	—	—	—	0	36	64	2	0	4176	1040	—	—
A-n64-k09	—	—	—	0	0	100	1	0	3313	1037	—	—
A-n65-k09	—	—	—	8	78	22	9	0	1301	1969	—	—
A-n69-k09	—	—	—	0	0	100	1	0	3434	1136	—	—
A-n80-k10	—	—	—	0	0	100	1	0	2138	594	—	—

Table 3.8: Results on JNVRGOC instances with 5 players and 5 clusters of optional customers.

	RG										ENUM ALL	
Instance	α'	α	T(s)	I	$T_i(\%)$	$T_g(\%)$	BP	WP	BBN	MCC	α	T(s)
A-n32-k05	0.69	x	208	14	86	14	17	5	253	238	0.63	120
A-n33-k06	0.81	0.26	239	10	62	38	13	5	1009	470	0.67	395
A-n34-k05	0.80	0.27	3145	14	40	60	18	5	3986	1002	0.70	4361
A-n36-k05	—	—	—	3	22	78	5	0	9708	527	0.59	33251
A-n37-k05	0.71	x	1171	12	44	56	18	5	664	520	0.70	2487
A-n37-k06	0.44	x	2785	6	29	71	12	5	2781	1467	0.36	4486
A-n38-k05	0.86	0.79	1648	10	59	41	14	5	1805	493	0.83	2340
A-n39-k05	0.74	0.57	2641	13	72	28	16	5	937	970	0.71	11389
A-n39-k06	0.72	0.57	1003	15	62	38	19	6	1072	682	0.68	1842
A-n44-k06	1.00	1.00	1219	8	53	47	13	4	1049	752	1.00	2871
A-n45-k06	1.00	1.00	2189	9	44	56	14	4	1710	545	1.00	5342
A-n45-k07	—	—	—	0	0	100	1	0	5611	1032	—	—
A-n46-k07	0.74	x	4649	14	69	31	18	5	1458	1365	0.65	15269
A-n48-k07	0.74	—	5421	12	66	34	16	5	1143	1936	—	—
A-n53-k07	—	—	—	2	26	72	3	0	2591	1536	—	—
A-n54-k07	—	—	—	0	0	100	1	0	3323	931	—	—
A-n55-k09	—	—	—	10	66	34	14	3	2959	2090	—	—
A-n60-k09	—	—	—	8	66	34	10	0	1211	2549	—	—
A-n61-k09	—	—	—	0	0	100	1	0	3930	912	—	—
A-n62-k08	—	—	—	4	31	29	5	0	1191	1039	—	—
A-n63-k09	—	—	—	3	44	47	4	0	960	2565	—	—
A-n63-k10	—	—	—	0	0	100	1	0	4889	777	—	—
A-n64-k09	—	—	—	0	0	100	1	0	2202	1163	—	—
A-n65-k09	—	—	—	6	23	53	7	0	2155	2445	—	—
A-n69-k09	—	—	—	0	0	100	1	0	2004	975	—	—
A-n80-k10	—	—	—	0	0	100	1	0	2179	594	—	—

Table 3.9: Results on JNVRGOC instances with 5 players and 7 clusters of optional customers.

	RG										ENUM ALL	
Instance	α'	α	T(s)	I	$T_i(\%)$	$T_g(\%)$	BP	WP	BBN	MCC	α	T(s)
A-n32-k05	0.64	0.56	709	19	42	58	26	7	558	686	0.63	18812
A-n33-k06	0.80	x	772	17	95	5	24	7	1526	958	0.67	6339
A-n34-k05	0.70	—	4121	21	36	64	28	7	9561	1069	—	—
A-n36-k05	0.78	—	5316	19	61	39	26	7	2141	1578	—	—
A-n37-k05	0.73	—	2888	17	73	27	24	7	1182	898	—	—
A-n37-k06	0.56	—	5842	14	61	39	22	7	3940	2409	—	—
A-n38-k05	0.76	x	2307	17	67	33	25	7	2132	1118	0.52	25322
A-n39-k05	—	—	—	13	60	33	14	0	3144	1626	—	—
A-n39-k06	0.63	—	2135	16	70	30	24	7	1532	1212	—	—
A-n44-k06	—	—	—	12	34	60	13	0	3601	2024	—	—
A-n45-k06	0.82	—	3790	16	74	26	24	7	1911	1592	—	—
A-n45-k07	—	—	—	0	0	100	1	0	1206	649	—	—
A-n46-k07	—	—	—	17	64	30	18	0	1636	3054	—	—
A-n48-k07	—	—	—	0	16	84	2	0	5113	973	—	—
A-n53-k07	—	—	—	0	11	89	2	0	5704	983	—	—
A-n54-k07	—	—	—	0	14	86	2	0	3078	1105	—	—
A-n55-k09	—	—	—	18	74	26	20	0	4417	1889	—	—
A-n60-k09	—	—	—	1	17	83	3	0	3381	1179	—	—
A-n61-k09	—	—	—	0	0	100	1	0	2845	1094	—	—
A-n62-k08	—	—	—	0	0	100	1	0	2383	1175	—	—
A-n63-k09	—	—	—	0	0	100	1	0	2501	1116	—	—
A-n63-k10	—	—	—	8	44	56	10	0	1504	2434	—	—
A-n64-k09	—	—	—	4	38	38	5	0	1418	1988	—	—
A-n65-k09	—	—	—	1	11	89	3	0	2546	1523	—	—
A-n69-k09	—	—	—	0	0	100	1	0	3615	806	—	—
A-n80-k10	—	—	—	0	0	100	1	0	2150	448	—	—

Table 3.10: Results on JNVRGOC instances with 7 players and 3 clusters of optional customers.

	RG										ENUM ALL	
Instance	α'	α	T(s)	I	$T_i(\%)$	$T_g(\%)$	BP	WP	BBN	MCC	α	T(s)
A-n32-k05	0.64	0.53	902	19	57	43	27	7	900	715	0.57	16329
A-n33-k06	0.71	x	449	16	82	18	22	7	1028	872	0.64	6169
A-n34-k05	0.57	0.19	4039	22	39	61	30	7	7960	1184	0.47	26038
A-n36-k05	0.68	—	3857	21	67	33	28	7	2450	1297	—	—
A-n37-k05	0.55	—	5834	19	31	69	27	7	1434	765	—	—
A-n37-k06	0.71	—	5243	19	71	29	25	7	3347	1986	—	—
A-n38-k05	0.68	0.44	2317	23	59	41	30	7	1956	992	0.67	15266
A-n39-k05	—	—	—	7	55	35	8	0	3134	1894	—	—
A-n39-k06	0.67	—	4262	22	64	36	30	8	1948	1126	—	—
A-n44-k06	—	—	—	17	89	11	18	0	4049	1932	—	—
A-n45-k06	—	—	—	20	79	21	28	7	2936	2083	—	—
A-n45-k07	—	—	—	1	1	13	2	0	4757	1261	—	—
A-n46-k07	—	—	—	15	91	6	16	0	1469	2406	—	—
A-n48-k07	—	—	—	0	11	89	2	0	6077	823	—	—
A-n53-k07	—	—	—	2	26	74	4	0	2429	1706	—	—
A-n54-k07	—	—	—	0	0	100	1	0	3128	897	—	—
A-n55-k09	—	—	—	4	63	37	6	0	3287	1515	—	—
A-n60-k09	—	—	—	0	100		1	0	2040	1045	—	—
A-n61-k09	—	—	—	0	0	100	1	0	3069	924	—	—
A-n62-k08	—	—	—	0	13	87	2	0	366	1452	—	—
A-n63-k09	—	—	—	0	0	100	1	0	2505	1710	—	—
A-n63-k10	—	—	—	0	0	100	1	0	5294	733	—	—
A-n64-k09	—	—	—	0	0	100	1	0	2761	505	—	—
A-n65-k09	—	—	—	1	8	92	3	0	2555	1626	—	—
A-n69-k09	—	—	—	0	84	16	1	0	2027	690	—	—
A-n80-k10	—	—	—	0	39	61	2	0	1007	1864	—	—

Table 3.11: Results on JNVRGOC instances with 7 players and 5 clusters of optional customers.

	RG										ENUM ALL	
Instance	α'	α	T(s)	I	$T_i(\%)$	$T_g(\%)$	BP	WP	BBN	MCC	α	T(s)
A-n32-k05	0.52	0.40	750	22	52	48	26	7	637	655	0.48	13040
A-n33-k06	0.69	x	409	19	82	18	26	7	872	947	0.66	6054
A-n34-k05	0.52	0.43	3968	22	27	73	29	8	7928	967	0.50	34934
A-n36-k05	0.59	—	6850	19	51	49	27	7	3377	1353	—	—
A-n37-k05	—	—	—	24	21	79	30	5	1573	1105	—	—
A-n37-k06	0.68	—	5228	21	75	25	25	7	4236	1779	—	—
A-n38-k05	0.66	0.55	4069	22	54	46	28	7	2306	911	0.61	21586
A-n39-k05	—	—	—	6	62	31	7	0	3273	1688	—	—
A-n39-k06	0.61	—	4924	23	53	47	31	7	1659	915	—	—
A-n44-k06	0.71	—	5982	21	55	45	26	7	1851	1513	—	—
A-n45-k06	0.67	—	4183	20	53	47	26	7	1378	1316	—	—
A-n45-k07	—	—	—	1	63	37	3	0	5413	1192	—	—
A-n46-k07	—	—	—	21	91	6	22	0	805	1915	—	—
A-n48-k07	—	—	—	0	21	79	2	0	6014	812	—	—
A-n53-k07	—	—	—	4	18	75	5	0	2167	1891	—	—
A-n54-k07	—	—	—	0	0	100	1	0	3580	876	—	—
A-n55-k09	—	—	—	2	21	79	4	0	6224	1126	—	—
A-n60-k09	—	—	—	4	13	6	5	0	1694	1604	—	—
A-n61-k09	—	—	—	0	0	100	1	0	3132	1123	—	—
A-n62-k08	—	—	—	0	34	66	1	0	1092	1507	—	—
A-n63-k09	—	—	—	5	73	27	7	0	1106	2973	—	—
A-n63-k10	—	—	—	0	0	100	1	0	5233	781	—	—
A-n64-k09	—	—	—	0	0	100	1	0	2741	909	—	—
A-n65-k09	—	—	—	2	20	80	4	0	2401	1853	—	—
A-n69-k09	—	—	—	4	62	18	5	0	1675	2146	—	—
A-n80-k10	—	—	—	0	0	100	1	0	1589	1184	—	—

Table 3.12: Results on JNVRGOC instances with 7 players and 7 clusters of optional customers.

Chapter 4

The Effect of Algorithmic Capabilities on Cooperative Games

Abstract

Collaborations lead to cost reductions, both monetary and environmentally. However, it is not immediately clear how multiple companies with a shared optimisation problem should arrive at solutions to this shared problem, or at a fair allocation of the resulting cost or profit. In contrast to the literature, we assume each company, also referred to as a player, to have access to a potentially heuristic algorithm that is used to determine solutions to this shared optimisation problem. Together, the players can use these algorithms to determine solutions to shared problem instances. We call a cooperative game in which player algorithms are explicitly taken into account an algorithm quality induced game (AQI game). In an AQI game, the cost that is allocated to a player also depends on their algorithmic capabilities, that is, the quality of their algorithms. Moreover, it also allows us to model consultants, i.e., players that do have a good algorithm for the shared optimisation problem, but do not contribute in any other manner to the shared operations. In an AQI game, such players can be allocated a profit. In this chapter we describe the core of AQI games and analyse the effects of improving the algorithm of a single player and including a consultant in a collaboration. Moreover, we present numerical results for 580 800 instances of the AQI game. We quantify the effect of improving an algorithm on

the allocated cost to this player. We show that a player in general is allocated less after improving their algorithm, while in some cases the allocated cost increases. Moreover, we find that in general players with a bad algorithm benefit most from the addition of a consultant while players with a good algorithm may not benefit at all.

This chapter is based on van Zon et al. (2021b).

4.1 Introduction

We consider multiple companies whose operations are concerned with the same optimisation problem, e.g., aviation companies that collectively operate flights or logistic service providers that collectively deliver parcels. We assume that each player has a problem instance of the shared problem and an algorithm to solve such instances. By collaborating, the companies can reduce their costs as well as their carbon footprint, i.e., by transporting passengers or goods more efficiently. In order to collaborate, the companies need to determine a solution to their shared problem instance. This problem instance can be obtained by merging their individual problem instances, i.e., a single problem instance that describes all goods that need to be transported and all vehicles that can be used for this purpose. However, it is not immediately clear how this problem instance should be solved by the companies with their algorithms. We propose multiple manners in which the algorithms of the companies can be used to arrive at a solution to the shared problem instance. Because of this dependence on the algorithms, the quality of the algorithm of a company is also reflected in their ability to negotiate a larger part of the savings.

In this chapter, we consider cooperative games where the cost or profit is dependent on an underlying optimisation problem. In Operations Research (OR), optimisation problems give rise to cooperative games. For example, consider the well-known vehicle routing problem (VRP). The VRP concerns serving a set of requests using a fleet of capacitated vehicles, see Toth and Vigo (2014) for more on the VRP. In a vehicle routing game, each company, for now on referred to as player, is associated with a set of requests and a set of vehicles. The cost of a group of players, known as a coalition, is given by a (heuristic) objective value to a VRP instance represented by the combined sets of requests and vehicles of the players in the coalition. By pooling these requests and vehicles, a significant cost reduction can be achieved, both monetary (Krajewska et al., 2008) and environmentally (Ballot and Fontane, 2010). Examples of games based on the VRP are given by Krajewska et al.

(2008), Frisk et al. (2010) and van Zon et al. (2021a), see Guajardo and Rönnqvist (2016) for more on OR games in collaborative routing. Besides collaborative routing, many other problems give rise to OR games, i.e., maintenance problems, scheduling problems and production problems (Borm et al., 2001).

One could argue that for any OR game, savings can be realised by merging smaller problem instances into large problem instances. It is customary to use a single exact or heuristic algorithm to arrive at solutions to these instances, see Göthe-Lundgren et al. (1996), Engevall et al. (2004), Krajewska et al. (2008), Drechsel and Kimms (2010), Zakharov and Shchegryaev (2015), Dai and Chen (2015) and van Zon et al. (2021a). Implicitly, this approach assumes that the players have access to a common algorithm to determine such solutions. There exist cases in which this assumption holds, i.e., a logistic service provider allocating costs to its customers (e.g. Engevall et al. (2004) and Naber et al. (2015)). Here, cost of each coalition is determined by the same algorithm, that is the algorithm of the logistic service provider. On the other hand, consider a collaboration among multiple logistic service providers, where the cost has to be allocated among the logistic service providers. In this case, there is not a common algorithm which the players can use to determine the cost of each coalition, although the potentially different algorithms of each of the players might be used. In general, heuristic algorithms are used for practical problems due to the computational effort required to determine solutions to industry sized problem instances, especially those obtained by collaborating. Due to the nature of heuristic algorithms, it is reasonable to assume that each player has access to a different heuristic algorithm.

In this chapter, we remove the implicit assumption of a common algorithm. We propose a framework for OR games in which we assume each player to have access to an algorithm which is used to determine solutions for a common optimisation problem. In the case of a logistic service provider allocating the cost to its customers, the algorithm is the same for each player. On the other hand, in the case of a collaboration among multiple logistic service providers, each player may have access to their own algorithm. Using these algorithms, the players can collectively determine the cost of a coalition, i.e., by solving problem instances individually and executing the best found solution or by combining their algorithms into a shared algorithm. We refer to a game in which the cost of a coalition is determined by the algorithms of the players of the coalition as an algorithm quality induced game (AQI game). In an AQI game, the algorithmic capabilities of each player are explicitly taken into account when allocating the grand-coalition cost to the players. We believe that this

is a more accurate representation of collaborations in practice.

Interestingly, the framework which we develop in this chapter, allows for players that do have an algorithm but do not contribute to the joint operations in any other manner. These players accurately reflect the case of consultants in real-life, hence we refer to these players as consulting players. A consulting player can contribute to a coalition by having a good algorithm, and thereby reducing the cost of a coalition. In contrast to the regular interpretation of cooperative games, these players create value and can be allocated a profit in return for their contribution in AQI games, which is precisely the business case of a consultant in real-life.

Our contribution is as follows. We propose a new framework for cooperative games in which individual player algorithms are explicitly taken into account. This framework also allows for the inclusion of consulting players, i.e., players that solely contribute to the collaboration with their algorithm. We study cost allocation for AQI games when considering several well-known allocation concepts and methods. We show analytically how the profit allocated to a player is influenced by improving an individual solution algorithm. Moreover, we prove both tight bounds for the profit each consulting player may be allocated in a core allocation, as well as a bound for the number of consulting players that can be allocated a profit in a core allocation. Finally, we present numerical results on 580 800 instances of the AQI game based on the pickup-and-delivery problem, a well known optimisation problem from the field of vehicle routing. We observe that for all allocation methods, on average, players are allocated less after they improved their algorithm. However, in certain cases the cost allocated to a player increased by improving their algorithm. Moreover, we find that players in general benefit from the inclusion of a consultant except for players which have high quality algorithms.

The remainder of this chapter is structured as follows. First, in Section 4.2, we introduce the two types of AQI games, the basic and partitioning AQI games. These games differ in the manner in which the cost of each coalition is determined given the player algorithms. Next, in Section 4.3 we introduce the different allocation concepts and methods which are considered in this chapter. In Section 4.4, we describe the cores of the basic and partitioning AQI games and derive multiple properties. We analyse the effects of improving an algorithm in Section 4.5 and the inclusion of consultants in Section 4.6. In Section 4.7, we present and discuss our numerical experiments. Finally, in Section 4.8 we provide a summary and some concluding remarks.

4.2 Algorithm quality induced games

Consider a cooperative game $\langle N, C\rangle$, with grand-coalition $N = \{1, \ldots, n\}$ and characteristic function $C : 2^N \to \mathbb{R}$, where 2^N consists of all subsets of N, and $C(\emptyset) = 0$. Associated with each coalition $S \subseteq N$ is an instance of the same underlying optimisation problem. Furthermore, for two disjoint coalitions $S_1, S_2 \subseteq N$, feasible solutions to the corresponding instances with objective values z_1 and z_2, respectively, can be combined into a feasible solution for the instance corresponding to coalition $S_1 \cup S_2$ with solution value $z_1 + z_2$. Moreover, each player has a deterministic algorithm at its disposal to solve instances of this optimisation problem. Denote by $C_i(S)$ the objective value of a feasible, but not necessarily optimal, solution to the problem instance corresponding to coalition $S \subseteq N$ found by the algorithm of player $i \in N$. Note that if no feasible solution is found, we let $C_i(S) = \infty$ or $C_i(S) = -\infty$ if the underlying problem is a minimisation or a maximisation problem, respectively. We consider the case that $C(S)$ is dependent in some way on the player algorithms, and say that $C(S)$ is induced by the player algorithms. In practice, there are various forms which can be observed, and we introduce several cases in the following subsections. We first propose two problem independent functions for the characteristic value of a coalition in Sections 4.2.1 and 4.2.2. Finally, to illustrate how features of the specific underlying optimisation problem can play a role in the characteristic function, we provide an example based on the pickup and delivery problem in Section 4.2.3. For ease of writing, in the remainder of this chapter, we consider the underlying optimisation problem to be a minimisation problem and refer to the characteristic function as a cost function. The analyses throughout the chapter straightforwardly apply also to the case where the underlying optimisation problem is a maximisation problem.

4.2.1 Basic cost function

Given a non-empty coalition $S \subseteq N$, we define the basic cost function as the best cost that the players $i \in S$ can determine by comparing the objective values of their solutions on the instance of coalition S, that is

$$C^B(S) = \min_{i \in S} \{C_i(S)\} . \tag{4.1}$$

We refer to a game $\langle N, C^B\rangle$ as a basic algorithm quality induced game (BAQI game). The basic cost function can be determined by applying each of the player algorithms and selecting the best solution value. However, it should be noted that the basic

cost function is not necessarily super-additive, i.e., there may exist disjoint coalitions $S, T \subseteq N$ such that $C^B(S) + C^B(T) < C^B(S \cup T)$. In such a case, S and T do not benefit from collaborating. Note that, if for each coalition at least one of the player algorithms provides the optimal solution, the cost function is super-additive, but when heuristics are used there is usually no such guarantee.

4.2.2 Partitioning cost function

In the partitioning cost function, we define the cost of a non-empty coalition $S \subseteq N$ as the best cost that can be obtained by applying the player algorithms to all subsets of the coalition and finding the best cost partition of S. The partitioning cost function is defined as

$$C^P(S) = \min_{\mathcal{P} \in \text{Part}(S)} \left\{ \sum_{P \in \mathcal{P}} \min_{i \in P} \{C_i(P)\} \right\}, \tag{4.2}$$

where $\text{Part}(S)$ denotes the set of partitions of S. We refer to a game $\langle N, C^P \rangle$ as a partitioning algorithm quality induced game (PAQI game). Note that in contrast to the basic cost function, the partitioning cost function is super-additive by definition, meaning that collaboration leads to non-increasing costs. Moreover, the cost obtained for a coalition by applying the partitioning cost function is never larger than the cost obtained by applying the basic cost function, i.e., $C^P(S) \leq C^B(S)$ for all $S \subseteq N$ because S itself is a partition of S. Furthermore, if a player algorithm yields an optimal solution for all instances, the basic cost function and partitioning cost function are equivalent for all coalitions that include this player.

Unlike the basic cost function, if each player is able to determine the stand-alone cost, the partitioning cost function value is guaranteed to be finite for each coalition. Note that we can compute the value of the partitioning cost function for a coalition by enumerating all partitions. Since the number of partitions is exponential in the number of players, this approach is only tractable for a small number of players. For a large number of players, application specific algorithms, or even heuristics, could be used to evaluate the partition cost function.

4.2.3 Problem specific cost functions

The previous approaches are applicable independently of the underlying optimisation problem. However, for some applications it is more natural to consider a problem specific cost function. We illustrate this by means of an example where we partition the instances on problem specific features, instead of on players.

First, we present the pickup and delivery problem (PDP) based on the descriptions of Savelsbergh and Sol (1995) and Ropke and Pisinger (2006). See Parragh et al. (2008) for a survey on the PDP. We also use the PDP in the numerical experiments in Section 4.7. Let Z be a set of transportation requests. Here, each request $z \in Z$ has load size $q_z \in \mathbb{N}$, one or multiple origins L_z^+ and one or multiple destinations L_z^-. Moreover, the load is divided over the origins and destinations such that $q_z = \sum_{l \in L_z^+} q_z(l) = -\sum_{l \in L_z^-} q_z(l)$, where we use positive values for pickup loads $q_z(l)$ with $l \in L_z^+$ and negative values for delivery loads $q_z(l)$ with $l \in L_z^-$. We define the location set $L = L^+ \cup L^-$ where $L^+ = \bigcup_{z \in Z} L_z^+$ and $L^- = \bigcup_{z \in Z} L_z^-$ denote the sets of all origins and all destinations, respectively. Each location $l \in L$ has an earliest service time s_l and latest service time e_l. The pickup and delivery requests are satisfied by a fleet of vehicles K, each with capacity $Q_k \in \mathbb{N}$, maximum travel time $T_k \in \mathbb{N}$, start depot D_k^+ and end depot D_k^- with $k \in K$. Given a request $z \in Z$, a specific vehicle might be required, i.e., a vehicle with a freezing compartment to transport frozen goods. We denote by K_z the set of vehicles that are allowed to serve request z.

Let $D = D^+ \cup D^-$ denote the set of all start depots D^+ and all end depots D^-. Let $V = L \cup D$ denote the set of all locations and depots. We consider a complete directed graph (V, A) where we associate a travel time $t_{vw} \geq 0$ and a travel cost $c_{vw} \geq 0$ with each arc $(v, w) \in A$. The requests are satisfied on routes R_k with $k \in K$. Here, route R_k starts at D_k^+, visits locations $L_k \subseteq L$ exactly once and ends in D_k^- such that for each request $z \in Z$ the corresponding locations are either not visited or all visited such that all origin locations L_z^+ precede the destination locations L_z^-, and the time windows are not violated. Moreover, the load of vehicle k should not exceed Q_k at any point on the route, the vehicle should be able to serve all requests, i.e., $k \in K_z$ for all requests $z \in Z$ served, and the travel time should not exceed T_k. A feasible solution to the PDP is a set of routes $\mathcal{R} = \{R_k : k \in K\}$ such that all requests are satisfied. Furthermore, the PDP is the problem of finding the minimum cost solution $\mathcal{R}$, where the cost of each route is determined by summing over the arc costs of the arcs used in the solution.

Consider a cooperative game $\langle N, C \rangle$ with the PDP as underlying optimisation problem. Special cases of cooperative games based on the PDP are given by Krajewska et al. (2008) and Dai and Chen (2015). Each player $i \in N$ has to fulfil requests $Z(\{i\})$ using fleet $K(\{i\})$. Hence, the instance belonging to coalition $S \subseteq N$ is characterised by requests $Z(S) = \bigcup_{i \in N} Z(\{i\})$ and fleet $K(S) = \bigcup_{i \in S} K(\{i\})$. Observe that state-of-the-art algorithms for the PDP only solve instances of up to

100 requests to optimality in a reasonable computation time (Baldacci et al., 2011; Gschwind et al., 2018). Instances corresponding to large coalitions may be intractable to solve as a whole. In that case, decomposition of the instances is required. Instead of decomposing based on the players as is done in the partitioning cost function, it seems more natural to first cluster the requests and vehicles based on geographical regions to decompose the instance, after which the smaller problems associated with the decomposition are solved separately. This procedure is known as cluster-first route-second, and is often used to solve similar problems (Parragh et al., 2008). However, due to the potentially large number of requests and vehicles, optimally determining such a decomposition may not be tractable as well. In practice, it is more likely that the players use a heuristic approach to perform the clustering. Suppose that the players use such an algorithm to obtain P clusters with requests Z^p and vehicles K^p, with $p \in \{1 \dots, P\}$. Moreover, let $C_i(Z^p, K^p)$ denote the cost of the instance induced by requests Z^p and vehicles K^p as determined by the algorithm of player $i \in N$. The cost of a coalition can be determined in the following manner

$$C(S) = \sum_{p=1}^{P} \min_{i \in S} \{C_i(Z^p, K^p)\}. \tag{4.3}$$

When using cost function (4.3), the players consider the best achievable cost given a decomposition of the instance associated with coalition $S \subseteq N$.

4.3 Cost allocation

The aim of a cooperative cost game $\langle N, C \rangle$ is to allocate the cost $C(N)$ to each player $i \in N$. Let $y \in \mathbb{R}^n$ denote a cost allocation such that y_i denotes the cost allocated to player $i \in N$, and $y(S) = \sum_{i \in S} y_i$ denotes the cost allocated to coalition $S \subseteq N$. Next, we describe common allocation concepts, which are of interest in this chapter.

The Star allocation allocates the grand-coalition cost proportional to the stand-alone cost of each player. The Star allocation to a player $i \in N$ is defined as

$$y_i = \frac{C(\{i\})}{\sum_{j \in N} C(\{j\})} C(N). \tag{4.4}$$

The Shapley value (Shapley, 1953) is a well-known allocation, where the cost allocated to each player $i \in N$ is a weighted average of the marginal contributions of this player

to each coalition. The Shapley value is defined as

$$y_i = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(n - |S| - 1)!}{n!}(c(S \cup \{i\}) - c(S)). \tag{4.5}$$

However, given the Star allocation or the Shapley value, it may be the case that a specific coalition is better off by forming a collaboration without the remaining players of the grand-coalition. Therefore, in OR games, core allocations are often considered, which instead ensure that no coalition has an incentive not to cooperate with the entire grand-coalition. See Borm et al. (2001) and Guajardo and Rönnqvist (2016) for surveys on cost allocation in OR games. Next, we describe the core, which was originally introduced by Gillies (1959). An allocation y is said to be individually rational if $y(i) \leq C(i)$ for all $i \in N$, coalitionally rational if $y(S) \leq C(S)$ for all $S \subseteq N$ and efficient if $y(N) = c(N)$. The imputation set is defined as all individually rational and efficient allocations. The core is a subset of the imputation set defined as the set of all coalitionally rational and efficient allocations. We denote the core of a game $\langle N, C \rangle$ as

$$\text{Core}(\langle N, C \rangle) = \{y \in \mathbb{R}^n : y(S) \leq C(S), y(N) = C(N)\}. \tag{4.6}$$

Note that the Star allocation and Shapley value are not guaranteed to be in the core. However, the Star allocation is guaranteed to be in the imputation set if it exists. Moreover, the Shapley value is in the imputation set if the cost function is sub-additive, that is $C(S \cup T) \leq C(S) + C(T)$ for all $S, T \subseteq N$.

Many allocation methods are based on the core. For example, the allocation given by the Lorenz method (Arin, 2003) is a core allocation which minimises the absolute difference in allocated cost to each of the players. Furthermore, the allocation given by the equal profit method (EPM; Frisk et al. (2010)) is defined as a core allocation which minimises the relative difference in allocated cost with respect to the stand-alone cost $C(\{i\})$ of each player $i \in N$. Both the Lorenz method and the EPM only yield an allocation if the core is non-empty. Moreover, these allocations are not guaranteed to be unique. The Nucleolus (Schmeidler, 1969) is a unique allocation (Driessen, 1988) which is in the core if the core is non-empty. The Nucleolus is defined as the imputation associated with the lexicographically largest excess vector, where the excess vector contains the excess $e(S) = C(S) - y(S)$ of each coalition $S \subseteq N$ in non-decreasing order. Note that the Nucleolus does not exist if the imputation set is empty.

4.4 The core of BAQI and PAQI games

Consider a BAQI game $\langle N, C^B \rangle$ and a PAQI game $\langle N, C^P \rangle$, with player algorithms C_i for each player $i \in N$. In this section, we study the cores of the BAQI game and the PAQI game. First, we show that the rationality constraints which correspond to non-trivial partitions for the partitioning cost function are redundant when describing the core of a PAQI game. Here, we say that the trivial partition for $C^P(S)$ is $\{S\}$.

Lemma 1. *If a non-trivial partition is optimal for coalition $S \subset N$ with the partitioning cost function $C^P(S)$, then the rationality constraint $y(S) \leq C^P(S)$ is redundant in the description of the core of the PAQI game.*

Proof. Consider a coalition $S \subset N$ for which there exists a non-trivial optimal partition. Assume that $\mathcal{P}$ is a maximal optimal partition of S, that is, the cost strictly increases by further splitting any of the coalitions $P \in \mathcal{P}$ in the partition. Observe that $C^P(S) = \sum_{P \in \mathcal{P}} \min_{i \in P} \{C_i(P)\} = \sum_{P \in \mathcal{P}} C^B(P)$. Note that as the partition is a maximal partition, the trivial partition is optimal for each coalition $P \in \mathcal{P}$.

Next, we show that the rationality constraint $y(S) \leq C^P(S)$ is redundant by showing that it is the sum of rationality constraints corresponding to coalitions for which the trivial partition is optimal for the partitioning cost function. To this end, note that $C^P(S) = \sum_{P \in \mathcal{P}} C^B(P) = \sum_{P \in \mathcal{P}} C^P(P)$. Hence, the rationality constraint is implied by rationality constraints $y(P) \leq C^P(P)$ for $P \in \mathcal{P}$. □

Note that each rationality constraint of a BAQI game which corresponds to a coalition $S \subset N$ to which Lemma 1 applies, is redundant in describing the core of the BAQI game. This result is summarised in Corollary 1.1.

Corollary 1.1. *If a non-trivial partition is optimal for coalition $S \subset N$ with the partitioning cost function $C^P(S)$, the rationality constraint $y(S) \leq C^B(S)$ is redundant in the description of the core of the BAQI game.*

Proof. Consider a coalition $S \subset N$ for which there exists a non-trivial optimal partition. Assume that $\mathcal{P}$ is a maximum optimal partition for S. From Lemma 1 we know that $y(S) \leq C^P(S)$ is implied by rationality constraints $y(P) \leq C^B(P)$ for $P \in \mathcal{P}$. Observe that $C^B(S) \geq C^P(S) = \sum_{P \in \mathcal{P}} C^B(P)$. Hence, rationality constraint $y(S) \leq C^B(S)$ is also implied by rationality constraints $y(P) \leq C^B(P)$ for $P \in \mathcal{P}$. □

For several applications with a specific underlying optimisation problem, one can easily determine to which coalitions Lemma 1 and Corollary 1.1 apply. For instance,

Göthe-Lundgren et al. (1996) show that the rationality constraints corresponding to coalitions of which the requests cannot be served on a single vehicle are redundant in the description of the core of the vehicle routing game. This occurs because the players have to be partitioned over multiple vehicles, that is, there exists a non-trivial partition for which the partitioning cost is optimal. Lemma 1 generalises the result of Göthe-Lundgren et al. (1996) to the setting of BAQI and PAQI games.

If redundancy can be determined in a simple manner such as for the vehicle routing game, one can simplify the computation of core allocations such as the EPM allocation and the Lorenz allocation by using Lemma 1. Moreover, this result also holds for the Nucleolus if the core is non-empty, which is discussed by (Chardaire, 2001) for cooperative games in general. Moreover, they also show that if the core is empty, coalitions corresponding to redundant rationality constraints still need to be taken into account when determining the Nucleolus.

Next, we show that the rationality constraints of the BAQI game and the rationality constraints of the PAQI game describe the same feasible region.

Theorem 2. *Consider an allocation $y \in \mathbb{R}^n$. It holds that $y(S) \leq C^B(S)$ for all $S \subseteq N$ if and only if $y(S) \leq C^P(S)$ for all $S \subseteq N$.*

Proof. From Lemma 1 and Corollary 1.1 it follows that it is sufficient to only consider coalitions $S \subset N$ for which the trivial partition is optimal for the partitioning cost, that is $C^B(S) = C^P(S)$. Hence, the non-redundant rationality constraints are equal for the BAQI game and the PAQI game. □

As a consequence of Theorem 2, the rationality constraints associated with the PAQI game can be replaced by those of the BAQI game when describing the core. This leads to a computational benefit when determining core allocations for the PAQI game because the optimal partition only has to be determined for the grand-coalition, and no optimal partition has to be determined for all other coalitions. Note that this also applies to the Nucleolus if the core is non-empty. This follows from the fact that the rationality constraints corresponding to coalitions $S \subset N$ for which $C^B(S) \neq C^P(S)$ are redundant as shown in Lemma 1.

Moreover, Theorem 2 also leads directly to Corollaries 2.1 and 2.2 which describe relations between the cores of the BAQI game and the PAQI game.

Corollary 2.1. *If the core of the PAQI game is empty, then the core of the BAQI game is also empty.*

Proof. Assume to the contrary that the core of the BAQI game is non-empty, and let $y \in \text{Core}(\langle N, C^B \rangle)$ be a core allocation. From Theorem 2 we find that $y(S) \leq C^P(S)$

for all $S \subseteq N$. Next, construct an allocation y' by reducing the cost allocated to an arbitrary player by $C^B(N) - C^P(N) \geq 0$. It follows that $y'(S) \leq C^P(S)$ for all $S \subseteq N$ and $y'(N) = C^P(N)$. Hence, $y' \in \text{Core}(\langle N, C^P(N)\rangle)$ which proves the desired result. □

The converse of 2.1 holds true if $C^P(N) = C^B(N)$. In this case, the core of the BAQI game and PAQI game are equivalent, which is shown in 2.2.

Corollary 2.2. *Core*$(\langle N, C^B\rangle) =$ *Core*$(\langle N, C^P\rangle)$ *if and only if* $C^P(N) = C^B(N)$ *and both cores are non-empty.*

Proof. The result follows directly from Theorem 2 and the definition of the core. □

Next, we adapt a well-known result in cooperative game theory for inessential games to the context of PAQI games. We show that if a non-trivial partition is optimal for the grand-coalition cost $C^P(N)$, it holds that each coalition of the partition is allocated precisely its cost in any core allocation for the PAQI game.

Theorem 3. *For any non-trivial partition* $\mathcal{P} \in$ *Part*(N) *which is optimal for the grand-coalition cost* $C^P(N)$*, it holds that* $y(S) = C^P(S)$ *for* $S \in \mathcal{P}$ *and any core allocation* $y \in \mathbb{R}^n$*.*

Proof. We provide a proof by contradiction. Consider a core allocation $y \in \mathbb{R}^n$ and let $\mathcal{P}$ be a non-trivial partition of N such that $\mathcal{P}$ is optimal for the grand-coalition cost, that is $\sum_{S\in\mathcal{P}} C^P(S) = C(N)$. It holds that $y(N) = C^P(N)$ as well as $y(S) \leq C^P(S)$ for all $S \in \mathcal{P}$. Suppose that there exists a coalition $S \in \mathcal{P}$ such that $y(S) < C^P(S)$, it follows that $y(N) = \sum_{S\in\mathcal{P}} y(S) < \sum_{S\in\mathcal{P}} C^P(S) = C(N)$, which is a contradiction. Hence, $y(P) = C^P(S)$ for all $S \in \mathcal{P}$. □

Every coalition that belongs to an optimal partition to the partitioning cost of the grand-coalition is allocated precisely its cost. Theorem 3 generalises a result for the vehicle routing game by Göthe-Lundgren et al. (1996). They show that the cost of a route of the grand-coalition is precisely allocated among the requests served on that route.

It is tempting to view the core of the cooperative game with partitioning cost function as a Cartesian product of the cores of multiple PAQI games of the coalitions belonging to an optimal partition. However, it should be noted that some rationality constraints linking the games remain. For example, consider a PAQI game $\langle\{1,2,3\}, C\rangle$ such that partition $\{\{1,2\},\{3\}\}$ is optimal with $C^P(N) = 5$. Moreover, let $C(\{1\}) = C(\{2\}) = C(\{3\}) = 2$, $C(\{1,2\}) = C(\{1,3\}) = C(\{2,3\}) = 3$.

Note that $(1,2) \in \text{Core}(\langle\{1,2\}, C\rangle)$ and $2 \in \text{Core}(\langle\{3\}, C\rangle)$, but $y = (1,2,2) \notin \text{Core}(\langle\{1,2,3\}, C\rangle)$ as $y(\{2,3\}) = 4 > 3 = C(\{2,3\})$. Hence, in general, a core allocation cannot be determined by considering these smaller cooperative games.

4.5 Value of algorithm improvement

In a AQI game with the basic cost function or the partitioning cost function, the value of an algorithm is explicitly taken into account. Therefore, players have an incentive to improve their algorithm to potentially decrease their allocated cost. In this section, we study the effect of replacing the algorithm of a single player with an improved algorithm. It should be noted that the effect on the allocation depends on the allocation method used. To this end, we compare the Shapley value, the Nucleolus, the EPM and the Lorenz method as introduced in Section 4.3. Moreover, we also consider the best-case and worst-case core allocation to a single player, which are described in detail later.

We consider the AQI games $\langle N, C\rangle$ and $\langle N, C'\rangle$, based on player algorithms C_i and C'_i for each player $i \in N$, respectively. Consider $S \subseteq N$, we let $C'_j(S) \leq C_j(S)$ for a single player $j \in N$, while $C'_i(S) = C_i(S)$ for all other players $i \in N \backslash \{j\}$. That is, we assume that only player j improves their algorithm, while the other players do not change their algorithm. Note that given a unique allocation method, the resulting allocations y and y' are only based on the values of the cost functions of $\langle N, C\rangle$ and $\langle N, C'\rangle$, respectively. Hence, we choose to characterise the improvements in the player algorithm by means of an improvement in C' when compared to C. Specifically, we assume that there exists a coalition $S \subseteq N$ such that $C'(S) < C(S)$, otherwise the cost function has not changed and the allocations will trivially remain unaffected by the improvement of the algorithm. First, in Section 4.5.1 we consider an improvement solely in the grand-coalition cost. Thereafter, in Section 4.5.2, we consider improvements of any coalition.

4.5.1 Grand-coalition improvements

Initially, we consider improvements only in the grand-coalition cost. That is, $C'(S) = C(S)$ for all $S \subset N$ and $C'(N) < C(N)$. This is closely related to the notion of aggregate monotonicity as introduced by Megiddo (1974). An allocation method is said to be monotone in the aggregate if $C'(N) \leq C(N)$ and $C'(S) = C(S)$ for all $S \subseteq N$ implies that $y'_i \leq y_i$ for all players $i \in N$. Here, it is important to note that the cost function does not in any way reflect that the reduction in the grand-coalition

cost is attributable to any individual player. Therefore, there is no reason to expect this gain to result in an improved allocation for the player that provided the improved algorithm. Hence, we do not consider a single player in particular. Clearly, the Star allocation is monotone in the aggregate. Furthermore, Megiddo (1974) shows that the Shapley value is monotone in the aggregate. Moreover, it is easy to see that if the grand-coalition cost strictly decreases, the Shapley value allocation to each player also strictly decreases. However, Megiddo (1974) shows that the Nucleolus is not monotone in the aggregate by means of a counterexample in which the grand-coalition cost strictly decreases. That is, the allocation to a single player may increase if the grand-coalition cost is decreased. This is undesirable as it poses an incentive not to improve the grand-coalition cost. The core allocations given by the EPM and the Lorenz method are not monotone in the aggregate as the allocations are not guaranteed to be unique if the core exists. Therefore, if the solution is not unique, there exists at least one other solution in which a single player is allocated strictly more. In order to still quantify the potential increase or decrease in allocated cost in a core allocation, we compare the best-case and worst-case cost allocated to each player given a core allocation. To this end, we let y_i^+ and $y_i'^+$ denote the best-case cost allocated to player $i \in N$ for the games $\langle N, C\rangle$ and $\langle N, C'\rangle$, respectively. The best-case allocated costs are defined as

$$y_i^+ = \min_{y\in\mathbb{R}^n}\{y_i : y(S) \leq C(S), y(N) = C(N)\}, \tag{4.7}$$

$$y_i'^+ = \min_{y\in\mathbb{R}^n}\{y_i : y(S) \leq C'(S), y(N) = C'(N)\}. \tag{4.8}$$

Observe that the rationality constraints remain feasible by decreasing the cost allocated to a player. Therefore, the best-case cost allocated to each player decreases by $y_i^+ - y_i'^+ = C(N) - C'(N) > 0$ if $\text{Core}(\langle N, C\rangle)$ is non-empty. Hence, the best-case core allocation is monotonic in the aggregate if the core of $\langle N, C\rangle$ is non-empty. Let y_i^- and $y_i'^-$ denote the worst-case cost allocated to player $i \in N$ for the games $\langle N, C\rangle$ and $\langle N, C'\rangle$, respectively. The worst-case allocated costs are defined as

$$y_i^- = \max_{y\in\mathbb{R}^n}\{y_i : y(S) \leq C(S), y(N) = C(N)\}, \tag{4.9}$$

$$y_i'^- = \max_{y\in\mathbb{R}^n}\{y_i : y(S) \leq C'(S), y(N) = C'(N)\}. \tag{4.10}$$

In the following we show that the worst-case cost allocation is not monotone in the aggregate.

Theorem 4. *Suppose that $C'(S) = C(S)$ for $S \subset N$ and $C'(N) < C(N)$. If*

Core$(\langle N, C\rangle)$ *and Core*$(\langle N, C'\rangle)$ *are both non-empty, then for each player* $j \in N$ *it holds that* $y_j'^- > y_j^-$ *if* $y_j^- < C(\{j\})$*, and* $y_j'^- = y_j^-$ *otherwise.*

Proof. Note that the result is immediate for $n \leq 2$ as the worst-case allocated cost is always equal to the stand-alone cost. Assume that $n > 2$, and let

$$\varepsilon = \min\left(C(\{j\}) - y_j^-, \frac{C(N) - C'(N)}{n-2}\right) \text{ and } \delta = \frac{C(N) - C'(N) + \varepsilon}{n-1} \tag{4.11}$$

Note that $\varepsilon \leq \frac{C(N)-C'(N)}{n-2}$ which implies that $\varepsilon(n-1) - \varepsilon \leq C(N) - C'(N)$ this, in turn, implies that $\varepsilon \leq \frac{C(N)-C'(N)+\varepsilon}{n-1} = \delta$. Next, we construct a core allocation y' for the game $\langle N, C'\rangle$. Let $y'_j = y_j + \varepsilon$ and $y'_i = y_i - \delta$ for $i \in N\backslash\{j\}$. It straightforwardly follows that $y'(S) \leq y(S) \leq C(S)$ if $S \neq \{j\}$ since $\varepsilon \leq \delta$, and that $y'_j = y_j + \varepsilon \leq y_j + C(\{j\}) - y_j = C(\{j\})$. Hence, y' satisfies the rationality constraints. Moreover, $y'(N) = y(N) + \varepsilon - (n-1)\delta = C(N) + \varepsilon - \varepsilon - (C(N) - C'(N)) = C'(N)$. Hence, y' is a core allocation for $\langle N, C'\rangle$. Note that $y_j'^- \geq y'_j$, which implies that $y_j'^- > y_j^-$ if and only if $\varepsilon > 0$, that is $y_j^- < C(\{j\})$. Otherwise, the worst-case core allocation to player j is bounded by the rationality constraint of coalition $\{j\}$. □

From Theorem 4 it follows that if the cost allocated to a player is not limited by the stand-alone cost, then a reduction in grand-coalition cost is guaranteed to increase the worst-case cost allocated to this player. On the other hand, the best-case allocated cost is guaranteed to improve. This follows from the fact that the entire reduction in the grand-coalition cost can be allocated to a single player, thereby reducing the best-case cost allocated to each player by precisely the reduction in grand-coalition cost.

4.5.2 Coalition improvements

Recall the assumption that there exists coalitions $S \subset N$ such that $C'(S) < C(S)$. We assume that these improvements can be attributed to a single player $j \in N$. Moreover, we assume that $C'(N) \leq C(N)$. This is closely related to the notion of coalitional monotonicity as introduced by Shubik (1962), which is used to describe how an allocation changes for a single player if only the costs of coalitions which include this player are decreased. An allocation method is coalitionally monotonic if for player $j \in N$, $C'(S) \leq C(S)$ for all $S \subseteq N$ with $j \in S$ and $C'(S) = C(S)$ otherwise, implies that $y'_j \leq y_j$. Clearly, the Star allocation and the Shapley value are coalitionally monotonic. Moreover, since there exists a coalition S with $j \in S$ such that $C'(S) < C(S)$, it holds that $y'_j < y_j$ for the Shapley value is considered. Young

(1985) shows that for more than 5 players no core allocation method, including the Nucleolus, is coalitionally monotonic. Note that the worst-case core allocation cannot increase if $C'(N) = C(N)$ since the feasible region as described by the rationality constraints does not increase. Note that this describes a type of monotononicity that is a special case of the definition of coalitional monotonicity by Shubik (1962) in which $C(N) = C'(N)$. Next, we present a bound for the difference in worst-case allocated cost.

Theorem 5. *Consider a player $j \in N$ with $C'(S) = C(S)$ if $j \notin S$, and $C'(S) \leq C(S)$ if $j \in S$ and $S \subseteq N$. Suppose that Core$(\langle N, C\rangle)$ and Core$(\langle N, C'\rangle)$ are both non-empty. Let Δ denote the minimum coalition improvement, excluding the grand-coalition. That is, $\Delta = \min_{S \subset N : j \in S} \{C(S) - C'(S)\}$. It holds that $y_j^- - y_j'^- \geq C(N) - C'(N)$ if $\Delta \geq C(N) - C'(N)$.*

Proof. See Appendix 4.A. □

From Theorem 5, it follows that the worst-case core allocated cost of a player strictly decreases if the minimum coalition improvement, excluding the grand-coalition, is equal to or exceeds the reduction in grand-coalition cost and both values are strictly positive.

4.6 Consultants

Players could add value to the grand-coalition by having a good algorithm. In particular, we consider the inclusion of players that do not face the underlying optimisation problem, i.e., the corresponding instance of the underlying optimisation problem is empty. We refer to such players as consultants. We say that a player $i \in N$ is a consultant if and only if the addition of a consultant $i \in N$ to a coalition $S \subseteq N \backslash \{i\}$ does not change the underlying instance. As a consequence, the following equalities hold for a consulting player i.

$$C_i(\{i\}) = 0, \tag{4.12}$$

$$C_j(S \cup \{i\}) = C_j(S) \quad \forall\, S \subseteq N \backslash \{i\},\ j \in N : j \neq i. \tag{4.13}$$

Equality (4.12) states that each consultant has no cost associated with the execution of the shared optimisation problem. Note that this implies that $C(\{i\}) = 0$, both for the basic and the partitioning cost function. As a result, a consulting player is never allocated a profit according to the Star allocation. Equalities (4.13) specify

that the cost of a coalition as determined by the players of that coalition does not change by adding a consultant. We believe this to be a realistic definition as the problem characteristics of a coalition do not change by adding a consultant. Note that Condition (4.13) implies that $C(S \cup \{i\}) \leq C(S)$ for the basic and the partitioning cost function.

Next, we consider the properties of core allocations to consultants. Here, we consider a cooperative game $\langle N, C \rangle$. First, we derive bounds for the allocated cost to a consultant in a core allocation. Here, note that negative costs correspond to profits. Observe that for each coalition $S \subseteq N$ it holds that $y(S) \leq C(S)$ and $y(S) \geq C(N) - C(N \backslash S)$ because $y(N \backslash S) \leq C(N \backslash S)$ and $y(N) = C(N)$. By combining these inequalities we obtain that $C(N) - C(N \backslash S) \leq y(S) \leq C(S)$. For a consultant $i \in N$ this implies that $C(N) - C(N \backslash \{i\}) \leq y_i \leq 0$. That is, the profit allocated to a consultant is non-negative and bounded by the value added to the grand-coalition. Furthermore, consultants are only allocated a profit if they contribute to the grand-coalition, and all consultants together are allocated at most the value they collectively add to the grand-coalition. Next, we show that the bounds $C(N) - C(N \backslash \{i\}) \leq y_i \leq 0$ are tight using a small example.

Example 4.6.1. *Let $N = \{1, 2\}$ with 2 a consultant. Moreover, let $C_1(\{1\}) = C_1(N) = 2$, $C_2(\{2\}) = 0$ and $C_2(N) = 1$. In the following we determine the cost of a coalition using the basic cost function. Hence, $C(N) = 1$. Note that both $y = (1, 0)$ and $y' = (2, -1)$ are core allocations. In the former case we find that $y_2 = 0$ while in the latter case we find that $y'_2 = -1 = C(N) - C(N \backslash \{2\})$.*

By means of an example, we show that there exist games in which the lower bound of 0 profit is not attained.

Example 4.6.2. *Consider a PAQI game with grand-coalition $N = \{1, 2, 3\}$ where player 3 is a consultant. The corresponding cost functions are given in Table 4.1.*

Table 4.1: Cost structure of the 3-player PAQI game

S	C_1	C_2	C_3	C
{1}	2			2
{2}		2		2
{3}			0	0
{1,2}	4	4		4
{1,3}	2		1	1
{2,3}		2	1	1
{1,2,3}	4	4	3	3

Observe that $y = (2, 2, -1)$ is the only core allocation.

We can also bound the number of consultants which are allocated a profit. First, we bound the number of consultants which can be allocated a profit for BAQI games.

Theorem 6. *In a BAQI game $\langle N, C^B \rangle$, at most one consultant can be allocated a profit in a core allocation.*

Proof. Let $\mathcal{C} \subseteq N$ denote the set of consultants. Assume that $|\mathcal{C}| \geq 2$, otherwise the result follows immediately. Suppose that two consultants $j_1, j_2 \in \mathcal{C}$ are allocated a profit in a core allocation, that is $y_{j_1} < 0$ and $y_{j_2} < 0$. Recall that the profit of a consultant $i \in \mathcal{C}$ is bounded by $C(N) - C(N \backslash \{i\}) \leq y_i \leq 0$. Hence, we find that $C(N) - C(N \backslash \{j_1\}) \leq y_{j_1} < 0$. This implies that $C(N) < C(N \backslash \{j_1\})$ and thus that $C(N)$ is only attained by the algorithm of player j_1. However, we analogously find that $C(N) < C(N \backslash \{j_2\})$ and which implies that $C(N)$ is only attained by the algorithm of player j_2. This is a contradiction. Hence, at most a single consulting player can be allocated a profit. □

Interestingly enough, while multiple consultants can contribute to different coalitions, only the consultant that provides the best solution to the grand-coalition can be allocated a profit in a core allocation.

For a PAQI game, multiple consultants can contribute to the grand-coalition if this cost is only attained by non-trivial partitions. As a consequence, multiple consultants can be allocated a profit. We provide a bound to the number of players that can be allocated a profit in a PAQI game.

Theorem 7. *In a PAQI game $\langle N, C^P \rangle$, the number of consultants which can be allocated a profit in a core allocation is equal to the minimum cardinality of all partitions that attain the grand-coalition cost.*

Proof. Consider a minimal partition $\mathcal{P} \in \text{Part}(N)$ for which the grand-coalition cost is attained, that is $\sum_{S \in \mathcal{P}} C^P(S) = C(N)$ and $C^B(S) < C^B(S_1) + C^B(S_2)$ for all disjoint $S_1, S_2 \subseteq S$ with $S_1 \cup S_2 = S$ and $S \in \mathcal{P}$. It follows that for every coalition $S \in \mathcal{P}$, the cost is equal to the cost as described by the basic cost function. Following the proof of Theorem 6, one can show that at most a single consultant in each coalition $S \in \mathcal{P}$ can be allocated a profit. However, we should still consider each possible minimal partition. Hence, the number of consultants which can be allocated a profit in a core allocation is equal to the minimum cardinality of all partitions that attain the grand-coalition cost. □

The previous result also indicates that the different consultants are only allocated a profit if they contribute to the grand-coalition in a different manner. Moreover, it also follows that two consulting players with equal algorithmic capabilities are both allocated no profit in a core allocation. Intuitively, this occurs because the consultants outcompete each other.

In some cases, a consultant may be able to provide good solutions for smaller coalitions rather than to the grand-coalition, for instance, if the cost function of the consultant is not sub-additive. In such cases, it may be desirable to let the consultant cooperate with multiple coalitions at once. To achieve this, copies of the consultant may be added. In the following we propose an extension of the game introduced in Example 4.6.2, to show that players may benefit from including a copy of the consultant.

Example 4.6.3. *In this example, we add a copy to the PAQI game as proposed in Example 4.6.2. The resulting cost functions are given in Table 4.3.*

Table 4.2: Cost structure of the 4-player PAQI game.

S	C_1	C_2	C_3	C_4	C
{1}	2				2
{2}		2			2
{3}			0		0
{4}				0	0
{1,2}	4	4			4
{1,3}	2		1		1
{1,4}	2			1	1
{2,3}		2	1		1
{2,4}		2		1	1
{3,4}			0	0	0
{1,2,3}	4	4	3		3
{1,2,4}	4	4		3	3
{1,3,4}	2		1	1	1
{2,3,4}		2	1	1	1
{1,2,3,4}	4	4	3	3	2

Observe that the grand-coalition cost is either obtained by partition $\{\{1,3\},\{2,4\}\}$ *or by partition* $\{\{1,4\},\{2,3\}\}$. *Moreover,* $C(N) - C(N\backslash\{3\}) = 1$ *and* $C(N) - C(N\backslash\{4\}) = 1$. *For this game, the only core allocation is given by* $y = \left(\frac{3}{2}, \frac{3}{2}, -\frac{1}{2}, -\frac{1}{2}\right)$.

Observe that rather than adding a copy of the consultant, one could equivalently make the cost function of the consultant sub-additive by considering all possible

partitions, similar to the procedure followed in the partitioning cost function.

Finally, we show that the the core of an AQI game can become empty by adding a consultant.

Example 4.6.4. *Consider a PAQI game with grand-coalition $N = \{1, 2, 3, 4\}$ where players 4 is a consultant. The corresponding cost functions are given in Table 4.3.*

Table 4.3: Cost structure of the 4-player PAQI game.

S	C_1	C_2	C_3	C_4	C
{1}	5				5
{2}		5			5
{3}			5		5
{4}				0	0
{1,2}	9	9			9
{1,3}	9		9		9
{1,4}	5			5	5
{2,3}		9	9		9
{2,4}		5		5	5
{3,4}			5	5	5
{1,2,3}	12	12	12		12
{1,2,4}	9	9		7	7
{1,3,4}	9		9	7	7
{2,3,4}		9	9	7	7
{1,2,3,4}	12	12	12	12	12

Recall that the consulting player is allocated at most $C(N) - C(N \backslash \{4\}) = 0$ profit. This implies that each pair of two players is allocated at most 7, which implies that $y_1 + y_2 + y_3 \leq 10\frac{1}{2}$. Together, with $y_4 = 0$, this implies that the core is empty. However, note that in a game without the consultant, allocation $y = (4, 4, 4)$ is in the core.

4.7 Numerical experiments

We generated 580 800 instances of the AQI game. These instances are constructed by applying a full combinatorial design to 12 instances of the pickup-and-delivery problem and 5 different algorithms, constructing all possible 3 and 4 player games with and without an additional consultant, who corresponds to a sixth algorithm. This procedure is detailed in Section 4.7.1. Here, we also detail the quality of the algorithms as well as the characteristics of the cores of these games. In Section

4.7.2 we report the effect of improving the algorithm of a single player on the cost allocated to each player. Finally, in Section 4.7.3 we consider the effect of including a consultant on the cost or profit allocated to each of the players.

4.7.1 Data generation and description

First, in Section 4.7.1.1, we describe in detail how we generated a total of 580 800 instances for the basic and partitioning AQI games. In Section 4.7.1.2 we describe in detail the performance of the algorithms on which the instances are based. The performance of these algorithms is crucial to understand the differences in allocated cost to the players for each of the AQI game instances. Finally, in Section 4.7.1.3 we report on the existence of the cores of the basic and partitioning AQI game instances.

4.7.1.1 Data generation

We generate instances of the basic and partitioning AQI games based on the pickup and delivery problem as introduced in Section 4.2.3. We associate both an instance of the pickup and delivery problem, given by a set of requests and a set of vehicles, and an algorithm with each player. Here, we consider the twelve instances with 50 pickup and delivery requests as introduced by Ropke and Pisinger (2006), which are referred to as *Prob50A* through *Prob50L*. These problem instances represent varying characteristics that companies might encounter in practice. Specifically, these instances represent differences in route types, request types and geographical distributions, see Ropke and Pisinger (2006). Here, we consider routes with equal or differing start and end locations, requests that can either be served by all vehicles or only by specific vehicles, and coordinates that are distributed either uniformly, clustered or semi-clustered. Moreover, each player is assigned an algorithm. As algorithms, we consider and use our implementation of the adaptive large neighbourhood search of Ropke and Pisinger (2006), where we differ the number of iterations to obtain algorithms of varying qualities. We consider the adaptive large neighbourhood search algorithm with 500, 2 500, 5 000, 7 500 and 10 000 iterations. It should be noted that the algorithm includes multiple random operations. Therefore, a seed is associated with each algorithm. Specifically, we differentiate between varying and equal seeds for the algorithms. Note that an algorithm with more iterations is only guaranteed to yield an equal or better solution than an algorithm with less iterations if equal seeds are considered.

We consider both three and four player AQI games based on the previously men-

tioned problem instances and algorithms. Each possible pair consisting of one of the twelve instances and one of the five algorithms is associated with at most one player in each game. By considering each combination of instance and algorithm, we generate $\binom{12}{3} \cdot 5 \cdot 4 \cdot 3 = 13\,200$ three player and $\binom{12}{4} \cdot 5 \cdot 4 \cdot 3 \cdot 2 = 59\,400$ four player game instances, respectively. For each of these game instances, we consider both the case in which the seed varies for each algorithm and the case in which the same seed is used for each algorithm. We do so in order to differentiate between algorithms that are strictly better and algorithms that are better on average. This is especially relevant since strict orderings may not occur in practice due to the fact that randomness is often used in state-of-the-art algorithms. Moreover, we also consider the addition of a consulting player. Here, a consulting player is represented by an empty pickup-and-delivery instance and an algorithm with 25 000 iterations. Hence, we generate a total of 52 800 three player and 237 600 four player basic and partitioning AQI games, respectively. For ease of notation we refer to the number of non-consulting players. For each of these games, we determine the cost allocations as introduced in Section 4.3 as well as the worst-case and best-case allocations for each player as introduced in Section 4.5.1.

The algorithm of Ropke and Pisinger (2006) as well as the algorithms to determine each of the allocations were implemented in Python, where the open source package Google OR-Tools was used to solve linear programming problems associated with some of the allocation methods. Moreover, the equal seed was chosen as the varying seed of the algorithm belonging to the consultant. In this manner, the solutions corresponding to the algorithm of the consultant only had to be determined once.

4.7.1.2 Quality of the player algorithms

In the following we compare the different algorithms to provide insights into the difference in solution quality. We consider 12 pickup and delivery instances with 50 requests, 66 pickup and delivery instances with 100 requests, 220 pickup and delivery instances with 150 requests, 495 pickup and delivery instances with 200 requests. In total, 793 pickup and delivery instances were solved with each of the eleven algorithms. First, we report the average cost that is obtained for the instances by each algorithm with a given number of iterations. Here, we differentiate between using equal and varying seeds for each algorithm. For each algorithm, represented by the number of iterations I, the average cost and the relative improvement in percentages when compared to the algorithm with 500 iterations are shown in Table 4.4.

I	Cost		Improvement	
	Equal	Varying	Equal	Varying
500	209 130	208 975	0.0%	0.0%
2 500	205 923	205 890	1.5%	1.5%
5 000	203 710	203 842	2.6%	2.5%
7 500	200 963	200 998	3.9%	3.8%
10 000	197 688	197 633	5.5%	5.4%
25 000	186 473	186 473	10.8%	10.8%

Table 4.4: Performance of the different algorithms.

The improvements of the different player algorithms are gradual by about 1-2% for each step. Moreover, the algorithm of the consulting player is significantly better when compared to the player algorithms.

Next, we compare how often each algorithm outperforms the other algorithms. Note that an algorithm with less iterations may only outperform an algorithm with more iterations if varying seeds are considered. Otherwise, more iterations never lead to a worse solution. The results are presented in Table 4.5, where we report the number of times that the algorithm with number of iterations as reported in column I strictly outperforms the algorithm with the number of iterations in the top-most row. Note that the cases for which the solution values are equal can be deduced from the table. Moreover, we differentiate between the use of equal seeds (E) versus varying seeds (V).

S	I	500	2 500	5 000	7 500	10 000	25 000
E	500	-	0	0	0	0	0
	2 500	676	-	0	0	0	0
	5 000	764	625	-	0	0	0
	7 500	791	777	696	-	0	0
	10 000	793	792	788	718	-	0
	25 000	793	793	793	793	791	-
V	500	-	117	24	0	0	0
	2 500	676	-	157	18	1	0
	5 000	769	636	-	88	4	1
	7 500	793	775	705	-	71	0
	10 000	793	792	789	722	-	1
	25 000	793	793	792	793	792	-

Table 4.5: Pairwise comparison of the performance of the algorithms on the 793 pickup and delivery instances.

Clearly, the random elements in the adaptive large neighbourhood search may cause worse algorithms to outperform better algorithms. This almost exclusively occurs for algorithms when the number of iterations do not differ much. The player algorithms are outperformed by worse algorithms in 10-23% of the considered instances. On the other hand, the algorithm of the consulting player is only outperformed twice by player algorithms, which corresponds to less than 1 percent of the considered instances. Moreover, note that when considering equal seeds, in approximately 4% of the comparisons it holds that an algorithm with more iterations does not outperform an algorithm with less iterations.

4.7.1.3 The cores of basic and partitioning AQI games

Next, we report statistics on the non-emptiness of the cores of the basic and partitioning AQI games. On average, the cores of basic and partitioning games were non-empty in 97.0% and 99.2% of the cases, respectively. This shows that the cores of the basic and partitioning AQI games are different. In Table 4.6, we present how often the core was non-empty (*NE*), where we differentiate based on the number of non-consulting players N, the seed type S, varying (V) or equal (E), the inclusion of a consultant C, yes (Y) or no (N), the type of AQI game G, basic (B) and partitioning (P).

S	N	C	G	*NE*
E	3	N	B	95.2%
			P	98.4%
		Y	B	99.8%
			P	100.0%
	4	N	B	93.7%
			P	97.8%
		Y	B	99.8%
			P	100.0%
V	3	N	B	95.4%
			P	99.4%
		Y	B	99.8%
			P	100.0%
	4	N	B	94.2%
			P	98.6%
		Y	B	99.8%
			P	100.0%

Table 4.6: The percentage of non-empty cores.

Observe that for most cases the core is non-empty, indicating that in most cases all players are incentivised to collaborate. Out of all 290 400 basic and partitioning AQI games, the core was only empty for 8 721 and 2 452 games, respectively. When considering partitioning games including a consulting player with equal and varying seeds, the core was empty for 0 and 12 out of the 72 600 games, respectively. Here, it should be noted that the 12 games with empty cores all correspond to the same pickup-and-delivery instances. The games only differ slightly in the algorithms assigned to each player. Moreover, we find that for 46% and 86% of the games the Star allocation and Shapley value are in the core, respectively.

4.7.2 Improving an algorithm

In the following we illustrate potential benefits that players can obtain by improving their algorithm. To this end, we consider all pairs of AQI games in which only the algorithm of a single player differs. First, we illustrate this effect of improving an algorithm on the allocation by means of a single pair of AQI games. Thereafter, we summarise the effects for all AQI games.

4.7.2.1 Example

We consider two basic AQI games with equal seeds and three players which only differ in the algorithm assigned to a single player. Specifically, player 1 is associated with instance *Prob50C*, an initial algorithm with 2 500 iterations and an improved algorithm with 5 000 iterations. Moreover, players 2 and 3 are associated with instances *Prob50K* and *Prob50L* and algorithms with 500 and 7 500 iterations, respectively. In the following, we refer to the basic AQI game in which player 1 has access to the initial algorithm as the initial game, and the basic AQI game in which player 1 has access to the improved algorithm as the improved game.

In the initial game the costs of each coalitions are as follows $C^B(1) = 65\,763$, $C^B(2) = 64\,203$, $C^B(3) = 66\,371$, $C^B(1,2) = 119\,773$, $C^B(1,3) = 122\,135$, $C^B(2,3) = 118\,948$ and $C^B(1,2,3) = 175\,560$. The only differences in the improved game are the costs obtained for coalitions $\{1\}$ and $\{1,2\}$ with costs 65 179 and 119 239, respectively. The resulting allocations for both the initial and the improved game are presented in Table 4.7. Here, both the Star allocation and Shapley value are in the core for the initial and improved game.

The allocated cost to player 1 as determined by the Star allocation, Shapley value, Nucleolus, EPM and worst-case core allocation are strictly reduced by improving the

Game	Player	Star	Shapley	Nucleolus	EPM	Lorenz	Best-case	Worst-case
Initial	1	58 804	59 347	59 857	58 804	58 520	56 612	65 763
	2	57 409	56 974	56 671	57 409	58 520	53 425	63 161
	3	59 347	59 239	59 032	59 347	58 520	55 787	65 523
Improved	1	58 456	59 064	59 679	58 456	58 520	56 612	65 179
	2	57 580	56 982	56 492	57 580	58 520	53 425	62 627
	3	59 524	59 514	59 389	59 524	58 520	56 321	65 523

Table 4.7: Allocations for each player of the initial and improved three-player game.

cost algorithm, and remain equal for the Lorenz and best-case core allocation.

Here, it should be noted that player 1 improved its own competitive position, while not improving the grand-coalition cost. As a result, in many of the allocations the reduction in allocated cost to player 1 results in an increase in allocated cost to players 2 and 3. More specifically, the allocated cost to players 2 and 3 increases in 3 and 5 out of the 7 allocations, respectively. Furthermore, the allocated cost to player 2 is decreased for the Nucleolus and the worst-case core allocation, while the cost allocated to player 3 is not decreased for any of the considered allocations. Moreover, it should be observed that the best-case core allocation of player 3 decreases. Both of which are attributed to the fact that the cost of coalition $\{1, 2\}$ improved, while the costs of the coalitions including player 3 did not. In these specific cases, one could say that player 2 also benefited from the improved algorithm of player 1.

4.7.2.2 Overall results

In the following, we consider all pairs of games which only differ in a single algorithm. We construct these pairs as follows. For each three-player game, we have two unused algorithms. We can exchange the algorithm of each player with one of the unused algorithms. Note that we do not allow the exchange of the algorithm belonging to the consultant. Therefore, there exist six games which only differ in the algorithm of a single player. Similarly, it follows that for each four-player game there exist four games which only differ in the algorithm of a single player. Note that we are only interested in the games in which a player improves their algorithm. Hence, only half of the aforementioned games have to be considered. In total, we consider 1 267 200 pairs of AQI games which only differ in the algorithm of a single player. Here, it should be noted that we also consider games which include a consulting player. However, the effect on the allocated cost of each player by including a consultant is discussed in Section 4.7.3.

The results are presented in Table 4.8, where in each row we consider a specific selection of pairs of games. We characterise for both games in the selected pairs

on what type of seed is used S, equal (E) or varying (V), whether a consultant is included C, yes (Y) or no (N) and the type of game G, basic (B) or partitioning (P). In the columns *old* and *new* we specify how many iterations are used by the considered player in the initial game and the improved game, respectively. In columns six through fourteen we summarise the average relative changes in percentages of the grand-coalition cost, the stand-alone cost of the player with improved algorithm, and the allocated costs to the player with improved algorithm for each of the different allocation methods. Here, we consider savings, that is, a positive percentage represents a decrease in costs. We are not able to determine the relative savings for all pairs of games, i.e., we can not determine the Lorenz allocation if the core is empty for one of the two games. Hence, the reported averages for each allocation only consider the pairs for which we were able to determine the relative savings for the corresponding allocation. Finally, in columns *ETNE* (empty to non-empty) and *NETE* (non-empty to empty), we consider the percentage of pairs for which the core of the initial game is empty but the core of the improved game is not empty and the percentage of pairs for which the core of the improved game is empty but the core of the initial game is not empty, respectively.

S	C	G	*Old*	*New*	$\Delta C(N)$	$\Delta C(i)$	Star	Shapley	Nucleolus	Lorenz	EPM	Worst	Best	ETNE	NETE
V	N	B	500	2 500	0.0	1.7	1.2	0.6	0.1	0.0	0.9	0.6	0.0	0.1	0.8
V	N	P	500	2 500	0.0	1.7	1.2	0.6	0.1	0.1	0.9	0.6	0.0	0.2	0.1
V	Y	B	500	2 500	0.0	1.7	1.2	0.5	0.1	0.0		1.0	0.0	0.0	0.0
V	Y	P	500	2 500	0.0	1.7	1.2	0.5	0.1	0.0		1.0	0.0	0.0	0.0
V	N	B	500	5 000	0.0	3.1	2.2	1.2	0.4	0.2	1.8	1.6	0.0	0.2	1.1
V	N	P	500	5 000	0.0	3.1	2.2	1.2	0.4	0.2	1.7	1.6	0.1	0.1	0.3
V	Y	B	500	5 000	0.0	3.1	2.2	0.9	0.2	0.2		2.1	0.0	0.0	0.0
V	Y	P	500	5 000	0.0	3.1	2.2	0.9	0.2	0.2		2.1	0.0	0.0	0.0
V	N	B	500	7 500	0.2	4.0	3.1	2.5	1.9	0.9	2.6	3.8	0.5	0.7	2.6
V	N	P	500	7 500	0.2	4.0	3.1	2.5	1.9	1.0	2.6	3.8	0.6	0.2	1.0
V	Y	B	500	7 500	0.0	4.0	2.9	1.7	0.6	0.3		3.1	0.0	0.0	0.1
V	Y	P	500	7 500	0.0	4.0	2.9	1.7	0.6	0.3		3.1	0.0	0.0	0.0
V	N	B	500	10 000	1.8	5.1	5.4	6.3	6.3	4.2	5.7	5.4	6.9	4.5	5.0
V	N	P	500	10 000	1.8	5.1	5.4	6.3	6.3	4.3	5.7	5.4	6.9	1.1	1.3
V	Y	B	500	10 000	0.0	5.1	3.7	3.3	2.1	0.5		5.0	0.0	0.0	0.1
V	Y	P	500	10 000	0.0	5.1	3.7	3.3	2.1	0.5		4.9	0.0	0.0	0.0
V	N	B	2 500	5 000	0.0	1.4	1.1	0.6	0.3	0.2	0.8	1.0	0.0	0.7	1.0
V	N	P	2 500	5 000	0.0	1.4	1.1	0.6	0.3	0.2	0.8	1.0	0.0	0.2	0.4
V	Y	B	2 500	5 000	0.0	1.4	1.1	0.5	0.2	0.1		1.2	0.0	0.0	0.0
V	Y	P	2 500	5 000	0.0	1.4	1.0	0.5	0.2	0.1		1.2	0.0	0.0	0.0
V	N	B	2 500	7 500	0.2	2.4	1.9	2.0	1.7	0.9	1.6	3.0	0.5	0.7	2.0
V	N	P	2 500	7 500	0.2	2.4	1.9	2.0	1.7	0.9	1.6	3.0	0.5	0.1	1.1
V	Y	B	2 500	7 500	0.0	2.4	1.8	1.2	0.5	0.2		2.1	0.0	0.0	0.0
V	Y	P	2 500	7 500	0.0	2.4	1.8	1.2	0.5	0.2		2.1	0.0	0.0	0.0
V	N	B	2 500	10 000	1.8	3.5	4.2	5.8	6.1	4.2	4.8	4.7	6.9	4.8	4.7
V	N	P	2 500	10 000	1.8	3.5	4.2	5.8	6.1	4.2	4.9	4.7	6.9	1.1	1.4
V	Y	B	2 500	10 000	0.0	3.5	2.5	2.8	2.0	0.4		3.7	0.0	0.0	0.1
V	Y	P	2 500	10 000	0.0	3.5	2.5	2.8	2.0	0.4		3.7	0.0	0.0	0.0
V	N	B	5 000	7 500	0.2	1.0	0.9	1.4	1.3	0.7	0.8	1.7	0.5	1.3	2.3
V	N	P	5 000	7 500	0.2	1.0	0.9	1.4	1.3	0.7	0.8	1.8	0.5	0.3	1.0
V	Y	B	5 000	7 500	0.0	1.0	0.7	0.8	0.3	0.1		0.9	0.0	0.0	0.1
V	Y	P	5 000	7 500	0.0	1.0	0.7	0.8	0.3	0.1		0.9	0.0	0.0	0.0
V	N	B	5 000	10 000	1.8	2.0	3.2	5.3	5.8	3.9	4.1	3.6	6.9	4.8	4.4
V	N	P	5 000	10 000	1.8	2.0	3.2	5.2	5.8	4.0	4.2	3.7	6.9	1.1	1.2
V	Y	B	5 000	10 000	0.0	2.0	1.5	2.4	1.7	0.3		2.3	0.0	0.0	0.1
V	Y	P	5 000	10 000	0.0	2.0	1.5	2.4	1.7	0.3		2.3	0.0	0.0	0.0
V	N	B	7 500	10 000	1.6	1.1	2.4	4.0	4.7	2.8	2.7	1.7	6.6	4.9	3.4
V	N	P	7 500	10 000	1.6	1.1	2.4	4.0	4.7	2.8	2.7	1.8	6.6	1.2	0.6
V	Y	B	7 500	10 000	0.0	1.1	0.8	1.6	1.2	0.1		1.4	0.0	0.0	0.0
V	Y	P	7 500	10 000	0.0	1.1	0.8	1.6	1.2	0.1		1.4	0.0	0.0	0.0
E	N	B	500	2 500	0.0	2.0	1.5	0.6	0.2	0.1	1.1	0.9	0.0	0.0	0.2
E	N	P	500	2 500	0.0	2.0	1.5	0.6	0.2	0.1	1.1	0.9	0.0	0.1	0.0
E	Y	B	500	2 500	0.0	2.0	1.5	0.5	0.2	0.1		1.3	0.0	0.0	0.0
E	Y	P	500	2 500	0.0	2.0	1.5	0.5	0.2	0.1		1.3	0.0	0.0	0.0
E	N	B	500	5 000	0.0	3.0	2.2	1.2	0.5	0.3	1.7	1.7	0.0	0.0	0.8
E	N	P	500	5 000	0.0	3.0	2.2	1.2	0.5	0.3	1.7	1.7	0.0	0.1	0.2
E	Y	B	500	5 000	0.0	3.0	2.2	0.9	0.3	0.2		2.2	0.0	0.0	0.0
E	Y	P	500	5 000	0.0	3.0	2.2	0.9	0.3	0.2		2.2	0.0	0.0	0.0
E	N	B	500	7 500	0.1	5.1	3.9	2.7	2.1	1.1	3.1	4.5	0.4	0.3	4.9
E	N	P	500	7 500	0.1	5.1	3.9	2.8	2.1	1.2	3.1	4.5	0.4	0.3	1.3
E	Y	B	500	7 500	0.0	5.1	3.8	1.9	0.9	0.5		4.2	0.0	0.0	0.1
E	Y	P	500	7 500	0.0	5.1	3.8	1.9	0.9	0.5		4.2	0.0	0.0	0.0
E	N	B	500	10 000	1.7	5.8	5.9	6.4	6.4	4.4	6.1	5.9	6.9	4.5	6.0
E	N	P	500	10 000	1.8	5.8	5.9	6.4	6.5	4.4	6.1	5.8	6.8	2.0	1.9
E	Y	B	500	10 000	0.0	5.8	4.3	3.4	2.5	0.6		5.6	0.0	0.0	0.2
E	Y	P	500	10 000	0.0	5.8	4.3	3.4	2.5	0.6		5.6	0.0	0.0	0.0
E	N	B	2 500	5 000	0.0	1.0	0.8	0.6	0.3	0.2	0.6	0.8	0.0	0.0	0.6
E	N	P	2 500	5 000	0.0	1.0	0.8	0.6	0.3	0.2	0.6	0.8	0.0	0.1	0.2
E	Y	B	2 500	5 000	0.0	1.0	0.8	0.4	0.2	0.1		0.9	0.0	0.0	0.0
E	Y	P	2 500	5 000	0.0	1.0	0.8	0.4	0.2	0.1		0.9	0.0	0.0	0.0
E	N	B	2 500	7 500	0.1	3.2	2.5	2.1	1.8	0.9	1.9	3.4	0.4	0.2	4.7
E	N	P	2 500	7 500	0.1	3.2	2.5	2.2	1.8	1.0	1.9	3.4	0.4	0.3	1.4
E	Y	B	2 500	7 500	0.0	3.2	2.4	1.4	0.7	0.4		2.8	0.0	0.0	0.1
E	Y	P	2 500	7 500	0.0	3.2	2.4	1.4	0.7	0.4		2.8	0.0	0.0	0.0
E	N	B	2 500	10 000	1.7	3.9	4.5	5.9	6.2	4.2	5.0	4.9	6.8	4.4	5.7
E	N	P	2 500	10 000	1.8	3.9	4.6	5.9	6.2	4.3	5.1	4.9	6.8	2.0	2.0
E	Y	B	2 500	10 000	0.0	3.9	2.9	2.9	2.2	0.4		4.0	0.0	0.0	0.2
E	Y	P	2 500	10 000	0.0	3.9	2.9	2.9	2.2	0.4		4.0	0.0	0.0	0.0
E	N	B	5 000	7 500	0.1	2.2	1.7	1.6	1.4	0.7	1.3	2.4	0.4	0.3	4.1
E	N	P	5 000	7 500	0.1	2.2	1.7	1.6	1.4	0.8	1.3	2.5	0.4	0.3	1.2
E	Y	B	5 000	7 500	0.0	2.2	1.6	1.0	0.6	0.2		2.0	0.0	0.0	0.1
E	Y	P	5 000	7 500	0.0	2.2	1.6	1.0	0.6	0.2		2.0	0.0	0.0	0.0
E	N	B	5 000	10 000	1.7	2.9	3.8	5.4	5.9	4.0	4.4	4.0	6.9	4.6	5.3
E	N	P	5 000	10 000	1.7	2.9	3.8	5.4	5.9	4.0	4.5	4.0	6.8	2.0	1.9
E	Y	B	5 000	10 000	0.0	2.9	2.1	2.5	2.0	0.3		3.1	0.0	0.0	0.2
E	Y	P	5 000	10 000	0.0	2.9	2.1	2.5	2.0	0.3		3.1	0.0	0.0	0.0
E	N	B	7 500	10 000	1.6	0.7	2.1	3.9	4.8	2.7	2.6	1.5	6.6	5.1	1.9
E	N	P	7 500	10 000	1.6	0.7	2.1	3.9	4.7	2.7	2.7	1.5	6.6	1.9	0.9
E	Y	B	7 500	10 000	0.0	0.7	0.5	1.5	1.1	0.1		1.0	0.0	0.0	0.1
E	Y	P	7 500	10 000	0.0	0.7	0.5	1.5	1.1	0.1		1.0	0.0	0.0	0.0

Table 4.8: Average effects of improving an algorithm on the allocated cost and emptiness of the core.

Observe that the decreases in the stand-alone costs are similar to the average improvements as presented in Table 4.4. However, it should be noted that the stand-alone cost is increased for 52 800 of the considered 633 600 pairs with varying seeds. The increases are solely caused by randomness and similar to the results presented in Table 4.5. Furthermore, only for 3 504 of the considered pairs with varying seeds, the grand-coalition cost is increased. On average, the grand-coalition cost decreases for any of the considered selections of AQI games. Observe that the average decrease is near zero for cases in which both the old and new algorithm have a low number of iterations. In these cases, the remaining player have access to the algorithms with a high number of iterations, i.e., the algorithms with 7 500 or 10 000 iterations, which are likely to outperform the other algorithms as shown in Table 4.5. Moreover, when a consultant is included, the algorithms of the players only outperform the algorithm of the consultant in a single combined pickup-and-delivery instance as observed in Table 4.5, which explains the observed near zero changes in grand-coalition cost if a consultant is included.

Next, we consider the differences in allocated cost to the player that improves their algorithm. On average, the decrease in worst-case and best-case allocated cost is between 0.6% and 5.9%, and 0.0% and 6.9%. This indicates that algorithmic improvements on average always improves the ability for each player to negotiate their cost. Moreover, for each allocation, the average decrease in allocated cost given the old algorithm is strictly increasing in the number of iterations of the new algorithm, independently of the type of game or the inclusion of a consultant. On average the allocated cost decreases by 2.4%, 2.3%, 1.9%, 1.0%, 2.7%, 2.6% and 1.4% for the Star allocation, Shapley value, Nucleolus, Lorenz allocation, EPM, worst-case core allocation and the best-case core allocation, respectively. These figures could provide an indication to the maximum budget that a company may want to allocate to improve their algorithm. That is, up to 1-2% of the allocated cost.

However, in AQI games with a consultant, we observe significantly smaller decreases in allocated cost for most cases. Hence, players are less incentivised to improve their algorithm if a consultant is part of the game. Note that these observations also hold for the worst-case and best-case core allocations. Despite the average decreases in allocated cost, there still exist cases in which the allocated cost is increased. For example, consider all 316 800 pairs of basic AQI games with varying seeds. The allocated cost is increased in 7.6%, 3.6%, 5.0%, 2.0%, 5.1%, 9.0% and 0.5% of the pairs for the Star allocation, Shapley value, Nucleolus, Lorenz allocation, EPM, worst-case core allocation and the best-case core allocation, respectively. Hence, in over 91% of

the cases it holds that an algorithmic improvement leads to a decrease in allocated cost when considering algorithms that include randomness. It should be noted that many of these cases may be caused by an increase of the stand-alone cost, i.e, for the Star allocation. On the other hand, when considering basic AQI games with equal seeds the stand-alone cost of a player cannot increase. In this case, we find that the allocated cost is increased in 0.0%, 0.0%, 0.1%, 0.1%, 0.0%, 0.2% and 0.0% of the pairs for the Star allocation, Shapley value, Nucleolus, Lorenz allocation, EPM, worst-case core allocation and the best-case core allocation, respectively. It follows that in over 99% of the cases, strict algorithmic improvements lead to a decrease in allocated cost. Clearly, the probability of an algorithmic improvement leading to a reduction in allocated cost is impacted by the randomness included in the algorithms.

Finally, we consider the effects of improving an algorithm on the emptiness of the core. Here, it can be observed that the effects on the core are small if a consultant is included. Specifically, out of all 633 600 pairs with a consultant, 10 times an empty core became non-empty after improving an algorithm, and 242 times a non-empty core became empty after improving an algorithm. In contrast, out of all 633 600 pairs without a consultant 8 978 times an empty core became non-empty after improving an algorithm, and 12 758 times a non-empty core became empty after improving an algorithm. This shows that the probability of destabilizing the collaboration by improving an algorithm is small.

4.7.3 Including a consultant

In the following we consider the effect of including a consultant to the allocations of each player. First, we present a single example of a game to which we add a consultant to illustrate this effect. Thereafter, we summarise the effects by considering all pairs of AQI games with and without a consultant.

4.7.3.1 Example

We consider the initial AQI game with three-players as presented in 4.7.2.1. The costs of the additional coalitions after adding a consultant, referred to as player 4, to this game are as follows: $C^B(4) = 0, C^B(1,4) = 63\,313$, $C^B(2,4) = 58\,531$, $C^B(3,4) = 64\,273$, $C^B(1,2,4) = 109\,339$, $C^B(1,3,4) = 116\,549$, $C^B(2,3,4) = 111\,266$ and $C^B(1,2,3,4) = 161\,918$. Note that the grand-coalition cost is 13 642 lower than that of the initial game in Section 4.7.2.1. We say that the created value of the consultant is 13 642. Moreover, the cost of each coalition has improved by including

a consulting player. The resulting allocations for the initial game with consultant are presented in Table 4.9. Here, we also report, in between brackets, the average relative savings when compared to the allocations for the initial game. Moreover, note that the EPM allocation is not included because it is not defined if a consultant is included.

Player	Star		Shapley		Nucleolus		Lorenz		Best-case		Worst-case	
1	54 235	(7.8)	56 966	(4.0)	56 953	(4.9)	53 973	(7.8)	50 652	(10.5)	65 763	(0.0)
2	52 948	(7.8)	53 357	(6.3)	52 905	(6.6)	53 973	(7.8)	45 369	(15.1)	64 203	(-1.6)
3	54 736	(7.8)	57 833	(2.4)	58 881	(0.3)	53 973	(7.8)	52 580	(5.7)	66 371	(-1.3)
4	0		-6 237		-6 821		0		-13 642		0	

Table 4.9: Allocations for the initial three-player game with consultant.

Both the Star allocation and the Shapley value are in the core. Moreover, for the three non-consulting players, the allocated cost as prescribed by the Star allocations, Shapley value, Nucleolus and Lorenz allocation is lower when the consultant is included. For these allocations, the allocated cost is decreased by 4.0-7.8%, 6.3-7.8% and 0.3-7.8% for players 1, 2 and 3, respectively. Clearly, player 2, the player with the worst algorithm, benefits most from the algorithm of the consultant. Moreover, the best-case allocated cost also decreases by 10.5%, 15.1% and 5.7% for players 1, 2 and 3, respectively. On the other hand, the worst-case cost is increased for player 2 by 1.6% and player 3 by 1.3% while it remains equal for player 1, which coincides with the stand-alone cost. Finally, observe that the consultant is only allocated a profit according to the Shapley value, the Nucleolus and the best-case core allocation. Here, the profit is 45.7%, 50.0% and 100.0% of the created value for the Shapley value, the Nucleolus and the best-case core allocation, respectively.

4.7.3.2 Overall results

We study both the profit allocated to the consultant as well as the effect on the allocated cost of the non-consulting players by including a consultant. First, we study the profit that is assigned to the consultant by considering each pair of AQI games which only differ in the inclusion of a consultant. In total, we consider 145 200 pairs of basic and partitioning AQI games. Here, we summarise the results only based on the type of game. It should be noted that there is no significant difference between the use of varying or equal seeds. This may be explained by the fact that the consultant was mostly able to determine the best cost of each coalition as shown in Table 4.4. The results are presented in Table 4.10. In the first column we represent

the type of AQI game (G), basic (B) of partitioning (P). In columns 2 through 5 we present the profit allocated to the consultant in percentages of the decrease in grand-coalition cost obtained by adding a consultant for several allocations. Finally, in columns *ETNE* and *NETE*, we consider the percentage of pairs for which the core of the initial game is empty but the core of the improved game is not empty and the percentage of pairs for which the core of the improved game is empty but the core of the initial game is not empty, respectively.

G	Star	Shapley	Nucleolus	Lorenz	Worst	Best	ETNE	NETE
B	0.0	48.6	51.0	0.5	0.5	99.6	5.8	0.1
P	0.0	48.7	51.1	0.6	0.5	99.9	1.7	0.0

Table 4.10: Average profit allocated to a consulting player in terms of added value and the effect on the emptiness of the core by adding a consulting player.

Note that by definition of the Star allocation the consultant is not allocated a profit. Interestingly enough, about half of the value that is created by the consultant is allocated to the consultant for both the Shapley value and the Nucleolus. In contrast, only slightly more profit is allocated to the consultant in the Lorenz allocation than is in the worst-case allocated profit. This is explained by the opposing goals of the players in the cost allocation when a consultant is included. While the non-consulting players are allocated costs, the constant is allocated a profit. The Lorenz method aims to minimize differences in allocated cost among all players. Hence, it often assigns the worst-case profit to the consultant. This shows that despite the fact that the Lorenz method allocates a non-zero profit to the consultant on average, the allocation is not well suited for games which include a consultant or both costs and profits should be allocated.

While the profit allocated in the worst-case core allocation is near zero, the allocated profit according to the best-case core allocation is almost equal to the created value by the consultant. Note that these values are on average close to the bounds derived in Section 4.6. Furthermore, for 190 and 12 of the 145 200 considered pairs of basic and partitioning AQI games the core of initial game becomes empty by adding a consultant, respectively. Note that this is far less than the 8 359 and 2 440 times that the core of basic and partitioning AQI games becomes non-empty by adding a consultant, respectively. This show that in general the addition of a consultant does not destabilize a collaboration, but may in fact lead to stability.

Next, we consider the impact of the allocations for non-consulting players by

including a consultant. The results are presented in Table 4.11. Here, we separated the allocation pairs based on the type of seed and game used, as well as the number of iterations of each player as represented by the first three columns S, G and I, respectively. For each of these selections, we report the average change in grand-coalition costs in the fourth column $\Delta C(N)$. Next, in columns 5 through 9 we present the average decrease in allocated cost to the non-consulting players in percentages for each of the allocation methods.

S	G	I	$\Delta C(N)$	Star	Shapley	Nucleolus	Lorenz	Worst	Best
E	B	500	6.3	6.3	4.2	4.3	6.9	-1.5	8.4
E	P	500	6.3	6.3	4.2	4.2	6.8	-1.6	8.4
E	B	2 500	6.3	6.3	4.1	4.3	6.9	-1.0	8.4
E	P	2 500	6.3	6.3	4.1	4.3	6.9	-1.1	8.4
E	B	5 000	6.3	6.3	3.9	4.1	6.8	-0.9	8.3
E	P	5 000	6.3	6.3	3.9	4.1	6.8	-1.0	8.3
E	B	7 500	6.3	6.3	3.1	3.1	6.2	-1.1	7.0
E	P	7 500	6.3	6.3	3.1	3.1	6.2	-1.2	7.1
E	B	10 000	5.9	5.9	1.1	-0.5	4.0	-2.3	1.3
E	P	10 000	5.8	5.8	1.0	-0.6	4.0	-2.4	1.4
V	B	500	6.3	6.3	4.2	4.3	6.8	-1.6	8.3
V	P	500	6.3	6.3	4.2	4.3	6.8	-1.7	8.4
V	B	2 500	6.3	6.3	4.0	4.3	6.8	-1.2	8.3
V	P	2 500	6.3	6.3	4.0	4.3	6.8	-1.3	8.4
V	B	5 000	6.3	6.3	3.8	4.1	6.8	-1.0	8.1
V	P	5 000	6.3	6.3	3.8	4.1	6.8	-1.0	8.3
V	B	7 500	6.2	6.2	3.1	2.9	6.1	-1.6	6.9
V	P	7 500	6.2	6.2	3.0	2.9	6.1	-1.6	7.0
V	B	10 000	5.8	5.8	1.0	-0.6	3.9	-2.7	1.2
V	P	10 000	5.8	5.8	1.0	-0.6	3.9	-2.8	1.3

Table 4.11: Average effects of including a consultant on the cost allocated to the non-consulting players.

Clearly, most allocation methods differentiate among the non-consulting players based on the quality of their solution algorithm. The only exception here is the Star allocation for which the relative reduction in allocated cost is equal for all players. Note that the relative savings for the Star and Lorenz allocation are inflated as the consultant is allocated either no profit at all or a low profit for these allocations. Therefore, the relative savings as reported for the Shapley value, Nucleolus, worst-case and best-case core allocation are more representative of collaborations in practice. For these allocation methods, it can be seen that the players with lower

quality algorithm benefit the most from the addition of a consultant. Intuitively, this occurs because these players benefit the most from the solutions provided by the consultant. This is especially clear when considering the relatively small increases in the worst case allocated cost when compared to the larger decreases in the best-case allocated cost. On the other hand, players with higher quality algorithm benefit less from the addition of a consultant. For the Shapley value, the Nucleolus, the Lorenz allocation, the worst-case core allocation and the best-case core allocation, the cost reduction is clearly less if the quality of the player algorithm is higher. Note that for players with an algorithm with 10 000 iterations, the allocated cost may increase. Specifically, this happens for the Nucleolus. Intuitively, this occurs because the player no longer has the best algorithm and therefore adds relatively less value to the collaboration. The observed relation between the allocated cost and the quality of the player algorithm does hold for the best-case core allocation, but not for the worst-case core allocation. Here, it is important to note that the worst-case core allocation becomes about 1-3% worse on average. This may be explained by the high average decreases in the grand-coalition costs. In conclusion, most players benefit by including a consultant. However, players with very well performing algorithms may be against the inclusion of a consultant as it may increase their allocated cost.

4.8 Conclusion

We consider cooperative games that are based on a shared optimisation problem, for which we propose a framework in which the algorithmic capabilities of each player are taken into account. Here, our framework naturally gives rise to players that act as a consultant, i.e., players with an empty problem instance but with a good algorithm. We propose two cost functions which are used to determine the cost of each coalition, the basic cost function and the partitioning cost function. We refer to games with the basic and partitioning cost function as basic algorithm quality induced games and partitioning algorithm quality induced games, respectively. We derive that the core of a basic algorithm quality induced game is empty if the core of the corresponding partitioning algorithm quality induced game is empty. Moreover, we describe the effects of improving an algorithm on the allocated cost of the corresponding player for various allocation methods. We show that improvements only in the grand-coalition cost lead to an equal decrease in best-case allocated cost and a strict increase in worst-case allocated cost if the worst-case allocated cost was originally lower than the stand-alone cost. We also show that under certain assumptions, the worst-case

allocated cost to a player decreases. Furthermore, given a core allocation, we derive bounds for the profit that can be allocated to a consultant as well as a limit on the number of consultants that can be allocated a profit in a core allocation.

We consider numerical experiments to examine the effect on the allocated cost of improving an algorithm and including a consultant. Our numerical results are based on 580 800 instances of the algorithm quality induced game. By considering all 1 267 200 pairs of algorithm quality induced games which only differ in the algorithm of a single player, we observe that for all allocation methods, players are allocated less after they improved their algorithm. On average, players are able to save about 1-6% of their allocated cost by improving their algorithm. Next, we considered the effect of including a consultant. First, we demonstrate that, unsurprisingly, the inclusion of a consultant with a good algorithm significantly reduces the grand-coalition cost. Moreover, the consultant is allocated about half of the reduction in grand-coalition cost as profit for the allocations given by the Shapley value and the Nucleolus. We also consider the effect of including a consultant on the cost allocated to the non-consulting players. Here, we show that players in general benefit from the inclusion of a consultant except for players which correspond to high quality algorithms. Interestingly, these players were allocated more when the Nucleolus was used.

We show that our framework is well suited towards players with a good algorithm but without any problem characteristics, i.e., consultants. On the other hand, our framework also naturally allows for the inclusion of players with only a partial share in the problem of the grand-coalition and without an algorithm. For instance, consider a collaboration among logistic service providers. Specifically, consider a player who owns one or multiple vehicles with a relatively large capacity when compared to the other players, but does not have to serve any requests. This player could solely contribute to the grand-coalition by sharing these vehicles, which may allow for more efficient routing.

Appendix

4.A Proof of Theorem 5

Consider a cooperative game $\langle N, C\rangle$. First, we introduce a linear programming formulation to determine the worst-case core allocation for player $j \in N$. Thereafter, we derive the dual of this problem which we use to prove Theorem 5.

Let y denote a cost allocation such that y_i denotes the cost allocated to player $i \in N$. The worst-case core allocation to player j can be determined by solving the following linear programming problem.

$$
\begin{aligned}
y_j^- = \max \quad & y_j, && && (4.14)\\
\text{s.t.} \quad & \sum_{i \in S} y_i \leq C(S) && \forall S \subset N, && (4.15)\\
& \sum_{i \in N} y_i \geq C(N), && && (4.16)\\
& y_i \in \mathbb{R} && i \in N. && (4.17)
\end{aligned}
$$

Here, Objective (4.14) ensures that the cost allocated to a player j is maximised. Moreover, Constraints (4.15)-(4.16) ensure that the allocation is in the core. Note that we allow the overall allocated cost to be larger than the grand-coalition cost in order to reduce the feasible region of the dual problem. To see that the maximum cost allocated to player j does not change, consider an optimal allocation y^* of which the total exceeds the cost $C(N)$, that is $y^*(N) > C(N)$. Consider the allocation y' which is obtained by reducing the allocated cost to a player $i \in N \backslash \{j\}$ by $y^*(N) - C(N)$. Here, $y'_j = y^*_j$ and Constraints (4.15)-(4.16) are still satisfied. Hence, y' is also optimal for (4.14)-(4.17) and $y(N) = c(N)$.

Next, we consider the dual of (4.14)-(4.17). Consider dual variables $\lambda_S \geq 0$ associated with Constraints (4.15) and dual variable $\lambda_N \leq 0$ associated with Constraint (4.16). The dual can be formulated as follows

$$
\begin{aligned}
y_j^- = \min \quad & \sum_{S \subset N} \lambda_S C(S) + \lambda_N C(N), && && (4.18)\\
\text{s.t.} \quad & \sum_{S \subset N : i \in S} \lambda_S + \lambda_N = 0 && \forall i \in N \backslash \{j\}, && (4.19)\\
& \sum_{S \subset N : j \in S} \lambda_S + \lambda_N = 1, && && (4.20)\\
& \lambda_S \geq 0 && S \subset N, && (4.21)\\
& \lambda_N \leq 0. && && (4.22)
\end{aligned}
$$

Note that we can omit Constraint (4.22) since it is implied by Constraints (4.19). To see this, let $i \in N \backslash \{j\}$ and observe that $\sum_{S \subset N : i \in S} \lambda_S \geq 0$ which implies that

$\lambda_N \leq 0$. Moreover, by substituting λ_N for $1 - \sum_{S \subset N: j \in S} \lambda_S$ and rewriting the remaining expressions, we obtain the following alternate formulation for the dual problem

$$y_j^- = C(N) + \min \quad \sum_{S \subset N} \lambda_S C(S) - C(N) \sum_{S \subset N: j \in S} \lambda_S, \tag{4.23}$$

$$\text{s.t.} \quad \sum_{S \subset N: j \in S, i \notin S} \lambda_S - \sum_{S \subset N: i \in S, j \notin S} \lambda_S = 1 \quad \forall i \in N \backslash \{j\}, \tag{4.24}$$

$$\lambda_S \geq 0 \qquad S \subset N. \tag{4.25}$$

Next, we provide a proof for Theorem 5. Consider games $\langle N, C \rangle$ and $\langle N, C' \rangle$. Let $S \subseteq N$, we assume that $C'(S) \leq C(S)$ if $j \in S$ and $C'(S) = C(S)$ otherwise. Given a dual solution λ_S for $S \subset N$, we compare the objective values $y(\lambda_S)$ and $y'(\lambda_S)$ as given by Objective (4.23) for games $\langle N, C \rangle$ and $\langle N, C' \rangle$, respectively. The difference between the objective values is as follows

$$\begin{aligned} &y(\lambda_S) - y'(\lambda_S) \\ &\quad = C(N) - C'(N) + \sum_{S \subset N: j \in S} \lambda_S \left[C(S) - C'(S)\right] + \left[C'(N) - C(N)\right] \sum_{S \subset N: j \in S} \lambda_S \end{aligned} \tag{4.26}$$

Here, we used the fact that $S \subset N$ with $j \notin S$, it holds that $C'(S) = C(S)$. Assume that $\Delta = \min_{S \subset N: j \in S} \{C(S) - C'(S)\} \geq C(N) - C'(N)$. Next, we derive the following bound

$$y(\lambda_S) - y'(\lambda_S) \geq C(N) - C'(N) + \left[\Delta + C'(N) - C(N)\right] \sum_{S \subset N: j \in S} \lambda_S. \tag{4.27}$$

Note that $\Delta + C'(N) - C(N) \geq 0$, which implies that

$$y(\lambda_S) - y'(\lambda_S) \geq C(N) - C'(N), \tag{4.28}$$

from which it follows that $y_j^- - y_j'^- \geq C(N) - C'(N)$.

Chapter 5

Conclusion

In this thesis, we consider multiple challenges in centralised horizontal logistic collaborations. In Chapter 2, we consider the cost allocation problem that arises if logistic service providers collaborate by consolidating demands and combining delivery routes. We extend this setting in Chapter 3 by also considering the additional profits that can be obtained by collectively serving new customers with the residual capacity. In Chapters 2 and 3, we focus on efficiently determining stable cost and profit allocations, respectively. In these chapters, we implicitly assume that a shared algorithm is available to determine cost or profit allocations. In Chapter 4, we describe collaborations among parties that are concerned with the same optimisation problem, but for which no shared optimisation algorithm is available. We propose a framework, based on cooperative game theory, in which collaborative solutions are determined by combining the algorithms of the players in the collaboration. We illustrate our framework by means of an application in collaborative transportation. In the following, we discuss our main findings for each chapter.

In Chapter 2, we model the cost allocation problem that arises if logistic service providers collaborate as new cooperative game, the Joint Network Vehicle Routing Game (JNVRG). Here, the players are able to attain cost reductions by consolidating demand and combining delivery routes. We propose a row generation algorithm to determine core allocations for the JNVRG, or to show that no core allocation exists. Here, we encountered a row generation subproblem which we model as a new vehicle routing problem with profits and solve it using a branch-and-price-and-cut algorithm. Moreover, we propose two acceleration strategies for the row generation algorithm. Our numerical results show that the row generation algorithm is effective in solving

instances of both the vehicle routing game and the joint network vehicle routing game. For the vehicle routing game, we are able to solve instances with up to 53 players, which is more than double the number of players that was previously considered in the literature.

In Chapter 3, we extend the setting of the JNVRG. Here, the logistic service providers can generate additional profits by serving new customers that are not associated with any of the logistic service providers. We model the resulting profit allocation problem as a cooperative game which we call the joint network vehicle routing game with optional customers (JNVRGOC). In the JNVRGOC, we consider both the best-case and worst-case profit of each coalition. Based on these profits, we consider the worst-case and best-case core for the JNVRGOC. Moreover, we propose a heuristic row generation algorithm to determine allocations in the best-case and worst-case cores or show that both cores are empty. We show that our algorithm scales well in the number of players.

In Chapter 4, we propose a framework for horizontal collaborations in which each player is concerned with the same optimisation problem. We assume that each player has access to an algorithm to solve instances of the shared optimisation problem. In contrast to previous approaches in cooperative game theory, we assume that the cost of a coalition is determined using the algorithms of the players. Specifically, we propose two different manners in which the player algorithms can be used to determine the cost of a coalition. We call games with these specific cost functions, algorithm quality induced games (AQI games). We derive multiple results for the core of these games and describe the effects of improving an algorithm on the allocated cost for varying allocation methods. We show that improvements only in the grand-coalition, lead to an equal decrease in the best-case allocated cost and a strict increase in the worst-case allocated cost if the worst-case allocated cost was originally lower than the stand-alone cost. We also show that under certain assumptions the worst-case allocated cost to a player decreases. Furthermore, we derive bounds for the profit that can be allocated to a consulting player as well as for the number of consulting players that can be allocated a profit. Specifically, we show that only consultants that contribute to the grand-coalition can be allocated a profit in a core allocation.

Moreover, we numerically describe the effect on the allocated cost of improving an algorithm or including a consulting player. To this end, we consider a numerical study based on 580 800 instances of the AQI game. We show that players are allocated on average less by improving their algorithm, while in some cases players may be allocated more. Moreover, we show that most players benefit from the inclusion of a

consultant. Only players with high quality algorithms get allocated more on average by including a consultant.

References

D. L. Applegate, R. E. Bixby, V. Chvatal, and W. J. Cook. *The traveling salesman problem: a computational study.* Princeton university press, 2006.

C. Archetti, N. Bianchessi, and M. Speranza. Optimal solutions for routing problems with profits. *Discrete Applied Mathematics*, 161(4):547 – 557, 2013. Seventh International Conference on Graphs and Optimization 2010.

J. Arin. Egalitarian distributions in coalitional models: The lorenz criterion. IKER-LANAK 2003-02, Universidad del País Vasco - Departamento de Fundamentos del Análisis Económico I, 2003.

P. Augerat. *Approche polyédrale du problème de tournées de véhicules.* Institut national polytechnique, 1995.

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

E. Ballot and F. Fontane. Reducing transportation co_2 emissions through pooling of supply networks: perspectives from a case study in french retail chains. *Production Planning & Control*, 21(6):640–650, 2010.

F. Basso, S. D'Amours, M. Rönnqvist, and A. Weintraub. A survey on obstacles and difficulties of practical implementation of horizontal collaboration in logistics. *International Transactions in Operational Research*, 26(3):775–793, 2018.

O. N. Bondareva. Some applications of linear programming methods to the theory of cooperative games. *Problemy kibernetiki*, 10:119–139, 1963.

P. Borm, H. Hamers, and R. Hendrickx. Operations research games: A survey. *Top*, 9(2):139–199, 2001.

P. Chardaire. The core and nucleolus of games: A note on a paper by göthe-lundgren et al. *Mathematical Programming*, 90(1):147–151, 2001.

L. Costa, C. Contardo, and G. Desaulniers. Exact branch-price-and-cut algorithms for vehicle routing. *Transportation Science*, 53(4):946–985, 2019.

J. Dahlberg, S. Engevall, and M. Göthe-Lundgren. Consolidation in urban freight transportation — cost allocation models. *Asia-Pacific Journal of Operational Research*, 35(04):1850023, 2018.

B. Dai and H. Chen. Proportional egalitarian core solution for profit allocation games with an application to collaborative transportation planning. *European Journal of Industrial Engineering*, 9(1):53–76, 2015.

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

J. Drechsel and A. Kimms. Computing core allocations in cooperative games with an application to cooperative procurement. *International Journal of Production Economics*, 128(1):310 – 321, 2010. Integrating the Global Supply Chain.

T. Driessen. A game theoretic approach to the cost allocation problem by means of the τ-value, the nucleolus and the shapley value. In *Theory and Decision Library*, pages 91–110. Springer Netherlands, 1988.

M. Dror. Cost allocation: the traveling salesman, binpacking, and the knapsack. *Applied Mathematics and Computation*, 35(2):191 – 207, 1990.

S. Engevall, M. Göthe-Lundgren, and P. Värbrand. The traveling salesman game: An application of cost allocation in a gas and oil company. *Annals of Operations Research*, 82(0):203–218, 1998.

S. Engevall, M. Göthe-Lundgren, and P. Värbrand. The heterogeneous vehicle-routing game. *Transportation Science*, 38(1):71–85, 2004.

U. Faigle. Cores of games with restricted cooperation. *Zeitschrift für Operations Research*, 33(6):405–422, 1989.

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

P. C. Fishburn and H. O. Pollak. Fixed-route cost allocation. *The American Mathematical Monthly*, 90(6):366–378, 1983.

M. Frisk, M. Göthe-Lundgren, K. Jörnsten, and M. Rönnqvist. Cost allocation in collaborative forest transportation. *European Journal of Operational Research*, 205 (2):448 – 458, 2010.

M. Gansterer and R. F. Hartl. Request evaluation strategies for carriers in auction-based collaborations. *OR spectrum*, 38(1):3–23, 2016.

M. Gansterer and R. F. Hartl. Collaborative vehicle routing: A survey. *European Journal of Operational Research*, 268(1):1–12, 2018.

D. B. Gillies. *Solutions to general non-zero-sum games.* Number 40 in Annals of Mathematics Studies. Princeton University Press, Princeton, 1959.

M. Göthe-Lundgren, K. Jörnsten, and P. Värbrand. On the nucleolus of the basic vehicle routing game. *Mathematical Programming*, 72(1):83–100, 1996.

T. Gschwind, S. Irnich, A.-K. Rothenbächer, and C. Tilk. Bidirectional labeling in column-generation algorithms for pickup-and-delivery problems. *European Journal of Operational Research*, 266(2):521–530, 2018.

M. Guajardo and M. Rönnqvist. A review on cost allocation methods in collaborative transportation. *International Transactions in Operational Research*, 23(3):371–392, 2016.

S. Irnich and G. Desaulniers. Shortest path problems with resource constraints. In G. Desaulniers, J. Desrosiers, and M. M. Solomon, editors, *Column Generation*, chapter 2, pages 33–65. Springer US, Boston, MA, 2005.

M. A. Krajewska and H. Kopfer. Collaborating freight forwarding enterprises. *OR spectrum*, 28(3):301–317, 2006.

M. A. Krajewska, H. Kopfer, G. Laporte, S. Ropke, and G. Zaccour. Horizontal cooperation among freight carriers: Request allocation and profit sharing. *The Journal of the Operational Research Society*, 59(11):1483–1491, 2008.

G. Laporte, Y. Nobert, and M. Desrochers. Optimal routing under capacity and distance restrictions. *Operations Research*, 33(5):1050–1073, 1985.

N. Lehoux, S. D'Amours, Y. Frein, A. Langevin, and B. Penz. Collaboration for a two-echelon supply chain in the pulp and paper industry: the use of incentives to increase profit. *Journal of the Operational Research Society*, 62(4):581–592, 2011.

J. K. Lenstra and A. H. G. Rinnooy Kan. Complexity of vehicle routing and scheduling problems. *Networks*, 11(2):221–227, 1981.

J. Lysgaard, A. N. Letchford, and R. W. Eglese. A new branch-and-cut algorithm for the capacitated vehicle routing problem. *Mathematical Programming*, 100(2): 423–445, 2004.

R. Martinelli, D. Pecin, and M. Poggi. Efficient elementary and restricted non-elementary route pricing. *European Journal of Operational Research*, 239(1):102 – 111, 2014.

N. Megiddo. On the nonmonotonicity of the bargaining set, the kernel and the nucleolus of game. *SIAM Journal on Applied Mathematics*, 27(2):355–358, 1974.

S. Naber, D. de Ree, R. Spliet, and W. van den Heuvel. Allocating co2 emission to customers on a distribution route. *Omega*, 54:191–199, 2015.

O. Ö. Özener, Ö. Ergun, and M. Savelsbergh. Lane-exchange mechanisms for truck-load carrier collaboration. *Transportation Science*, 45(1):1–17, 2011.

S. N. Parragh, K. F. Doerner, and R. F. Hartl. A survey on pickup and delivery problems. *Journal für Betriebswirtschaft*, 58(2):81–117, 2008.

D. Pecin, A. Pessoa, M. Poggi, and E. Uchoa. Improved branch-cut-and-price for capacitated vehicle routing. *Mathematical Programming Computation*, 9(1):61–100, 2017a.

D. Pecin, A. Pessoa, M. Poggi, E. Uchoa, and H. Santos. Limited memory rank-1 cuts for vehicle routing problems. *Operations Research Letters*, 45(3):206 – 209, 2017b.

J. A. M. Potters, I. J. Curiel, and S. H. Tijs. Traveling salesman games. *Mathematical Programming*, 53(1):199–211, 1992.

E. Pérez-Bernabeu, A. A. Juan, J. Faulin, and B. B. Barrios. Horizontal cooperation in road transportation: a case illustrating savings in distances and greenhouse gas emissions. *International Transactions in Operational Research*, 22(3):585–606, 2014.

G. Righini and M. Salani. New dynamic programming algorithms for the resource constrained elementary shortest path problem. *Networks*, 51(3):155–170, 2008.

S. Ropke and D. Pisinger. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation Science*, 40(4): 455–472, 2006.

M. W. P. Savelsbergh and M. Sol. The general pickup and delivery problem. *Transportation Science*, 29(1):17–29, 1995.

D. Schmeidler. The nucleolus of a characteristic function game. *SIAM Journal on Applied Mathematics*, 17(6):1163–1170, 1969.

L. S. Shapley. A value for n-person games. *Contributions to the theory of games*, 2: 307–317, 1953.

L. S. Shapley. On balanced sets and cores. *Naval Research Logistics Quarterly*, 14 (4):453–460, 1967.

M. Shubik. Incentives, decentralized control, the assignment of joint costs and internal pricing. *Management Science*, 8(3):325–343, 1962.

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

P. Toth and D. Vigo. *Vehicle Routing*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2014.

M. A. van Zon and G. Desaulniers. The joint network vehicle routing game with optional customers. *Computers & Operations Research*, 133:105375, 2021.

M. A. van Zon, R. Spliet, and W. van den Heuvel. The joint network vehicle routing game. *Transportation Science*, 55(1):179–195, 2021a.

M. A. van Zon, R. Spliet, and W. van den Heuvel. The effect of algorithmic capabilities on cooperative games. Technical Report EI2021-02, Econometric Institute Research Papers, 2021b.

X. Wang and H. Kopfer. Collaborative transportation planning of less-than-truckload freight. *OR spectrum*, 36(2):357–380, 2014.

H. P. Young. Monotonic solutions of cooperative games. *International Journal of Game Theory*, 14(2):65–72, 1985.

V. V. Zakharov and A. N. Shchegryaev. Stable cooperation in dynamic vehicle routing problems. *Automation and Remote Control*, 76(5):935–943, 2015.

Abstract

Significant cost reductions, both monetary and environmentally, can be realised by collaborative efforts in logistics. Specifically, we study centralised horizontal collaborations between logistic service providers in the field of transportation. In this particular setting, cost reductions are achieved by pooling requests, combining delivery routes and collectively serving new customers.

First, we consider a centralised horizontal collaboration between logistic service providers that is governed by a third party. Here, we assume that the collaborative operations are planned by this third party. We focus on how to determine a stable allocation of the overall cost or profit among the logistic service providers. That is, no subset of logistic service providers is better off by leaving the collaboration. For two different collaborative settings, we propose algorithms to determine allocations which ensure a stable collaboration among the logistic service providers, or to show that no such allocation exists. Moreover, we demonstrate the efficiency of these algorithms.

Second, we focus on a centralised horizontal collaboration that is not governed by a third party. That is, the logistic service providers need to plan their joint operations together. We propose a framework, based on cooperative game theory, in which the algorithms of each logistic service provider can be used to determine a planning for their joint operations. In doing so, any logistic service provider or third party with a good algorithm can add value to the collaboration, even without being a logistic service provider, i.e., a consultant. We study this setting both analytically and numerically for collaborations in transportation, and provide insights into the value of the quality of an algorithm in collaborations.

Nederlandse Samenvatting (Abstract in Dutch)

Samenwerkingen in de logistieke sector kunnen leiden tot significante reducties in financiële kosten en uitstoot van broeikasgassen. In het bijzonder focussen we ons op gecentraliseerde horizontale samenwerkingen tussen verschillende logistieke dienstverleners op het gebied van transport. In dit geval kunnen kosten bespaard worden door het samenvoegen van transportverzoeken, het combineren van distributieroutes en het aannemen van nieuwe opdrachten.

Ten eerste beschouwen we een gecentraliseerde horizontale samenwerking tussen logistieke dienstverleners die gecoördineerd wordt door een onafhankelijke partij. Hier nemen we aan dat de logistieke planning gedaan wordt door deze onafhankelijke partij. We richten ons op het bepalen van stabiele kosten- of winstverdelingen van de gehele kosten of winst over de logistieke dienstverleners. Dit betekent dat we een kosten- of winstverdeling proberen te vinden zodat geen enkele logistieke dienstverlener beter af is door niet met de andere logistieke dienstverleners samen te werken. In dit proefschrift presenteren we twee algoritmes voor twee verschillende logistieke situaties. Met behulp van deze algoritmes zijn we in staat om stabiele kosten- of winstverdelingen te bepalen, of om te laten zien dat een dergelijke kosten- of winstverdeling niet bestaat. Verder demonstreren we ook de computationele effectiviteit van onze algoritmes.

Ten tweede beschouwen we een gecentraliseerde horizontale samenwerking tussen logistieke dienstverleners die niet gecoördineerd wordt door een onafhankelijke partij. Dit betekent dat de logistieke dienstverleners gezamenlijk tot een logistieke planning moeten komen. We presenteren een theoretisch kader dat gebaseerd is op coöperatieve speltheorie, waarbij we gebruik maken van de algoritmes van de logistieke dienstverleners om tot een gezamenlijke logistieke planning te komen. Dit kader

stelt ons in staat om de waarde van een algoritme expliciet uit te drukken. Op deze manier kan dus zelfs een derde partij, eventueel zonder te beleveren klanten, waarde toevoegen aan de samenwerking door het beschikbaar maken van een goed algoritme. Een voorbeeld van een dergelijke derde partij is een consultant. We beschouwen dit probleem zowel vanuit een theoretisch als een numeriek perspectief. Op basis hiervan presenteren we meerdere inzichten over de waarde van een algoritme in logistieke samenwerkingen.

About the author

Mathijs van Zon was born on March 23, 1995 in Delft, the Netherlands. In 2016, Mathijs received his Bachelor's degrees in Applied Mathematics and Applied Physics cum laude from the Technical University Delft. In 2017, Mathijs received his Master's degree in Econometrics and Management Science, with a specialization in Operations Research and Quantitative Logistics, cum laude from the Erasmus University Rotterdam. During his studies, he worked as a student assistant at the Technical University Delft and as an author for educational publisher Malmberg.

In 2017, Mathijs started his PhD candidacy at the Erasmus Research Institute of Management at the Erasmus University Rotterdam under supervision of Prof.dr. Albert Wagelmans, Dr. Remy Spliet and Dr. Wilco van den Heuvel. During his time as a PhD candidate, Mathijs developed new concepts and methodologies for cost allocation in collaborative transportation. He presented his work at seminars and conferences in Austria, Canada, Italy and the Netherlands. Moreover, his research has been published in *Transportation Science* and *Computers & Operations Research*. In 2019, Mathijs went on a research visit to the world-renowned research centre GERAD in Montreal to collaborate with Prof.dr. Guy Desaulniers. Mathijs is currently employed as an algorithm expert at PostNL, where he strives to bring his expertise in solving optimisation problems into practice.

Portfolio

Publications in International Peer-reviewed Journals

M. A. van Zon, R. Spliet, and W. van den Heuvel. The joint network vehicle routing game. *Transportation Science*, 55(1):179–195, 2021a

M. A. van Zon and G. Desaulniers. The joint network vehicle routing game with optional customers. *Computers & Operations Research*, 133:105375, 2021

Technical Reports

M. A. van Zon, R. Spliet, and W. van den Heuvel. The effect of algorithmic capabilities on cooperative games. Technical Report EI2021-02, Econometric Institute Research Papers, 2021b

Research Visits

Multiple visits to the research centre GERAD. Collaboration with Prof. dr. Guy Desaulniers from the Polytechnique Montréal resulting in van Zon and Desaulniers (2021).

Reviewing Activities for International Peer-reviewed Journals

IEEE Access.

Transportation Science.

Networks.

Teaching activities

Teaching assistant and course design for the *Econometrics and Operations Research* bachelor course *Academic Writing*, 2018-2020.
Teaching and technical assistant for the *Econometrics and Operations Research* bachelor course *Combinatorial Optimisation*, 2020.
Teaching assistant for the *Economics and Business Economics* bachelor course *Mathematics and Game Theory*, 2020-2021.
Supervision of bachelor theses in *Econometrics and Operations Research*, 2018-2019 and 2021.
Second reader of master theses in *Operations Research and Quantitative Logistics*, 2021.

PhD courses and Certificates

Algorithms and Complexity.
Interioir Point Methods.
Markov Decision Processes.
Integer Programming Methods.
Noncooperative Games.
Algorithmic Game Theory.
Robust Optimisation.
Convex Analysis for Optimisation.
Networks and Polyhedra.
Scientific Integrity.
Publishing Strategy.
Cambridge English: Certificate in Advanced English.

Conference Presentations

Odysseus 2018, Cagliari, Italy.
TSL Workshop 2019, Vienna, Austria.
LNMB Conference 2020, Lunteren, The Netherlands.
Econometric Institute PhD Conference 2020, Rotterdam, The Netherlands.

Other Presentations

Econometrics Institute Seminar, 2019, Rotterdam, The Netherlands.
GERAD seminar, 2019, Montréal, Canada.
Research Meeting on Collaborative Transportation, 2020, Rotterdam, The Netherlands.

The ERIM PhD Series

The ERIM PhD Series contains PhD dissertations in the field of Research in Management defended at Erasmus University Rotterdam and supervised by senior researchers affiliated to the Erasmus Research Institute of Management (ERIM). All dissertations in the ERIM PhD Series are available in full text through the ERIM Electronic Series Portal: repub.eur.nl/pub. ERIM is the joint research institute of the Rotterdam School of Management (RSM) and the Erasmus School of Economics (ESE) at the Erasmus University Rotterdam (EUR).

Dissertations in the last four years

Ahmadi, S., *A motivational perspective to decision-making and behavior in organizations*, Promotors: Prof. J.J.P. Jansen & Dr. T.J.M. Mom, EPS-2019-477-S&E, https://repub.eur.nl/pub/116727

Akemu, O., *Corporate Responses to Social Issues: Essays in Social Entrepreneurship and Corporate Social Responsibility*, Promotors: Prof. G.M. Whiteman & Dr. S.P. Kennedy, EPS-2017-392-ORG, https://repub.eur.nl/pub/95768

Albuquerque de Sousa, J.A., *International stock markets: Essays on the determinants and consequences of financial market development*, Promotors: Prof. M.A. van Dijk & Prof. P.A.G. van Bergeijk, EPS-2019-465-F&A, https://repub.eur.nl/pub/115988

Alserda, G.A.G., *Choices in Pension Management*, Promotors: Prof. S.G. van der Lecq & Dr. O.W. Steenbeek, EPS-2017-432-F&A, https://repub.eur.nl/pub/103496

Anantavrasilp, S., *Essays on Ownership Structures, Corporate Finance Policies and Financial Reporting Decisions*, Promotors: Prof. A. de Jong & Prof. P.G.J. Roosenboom, EPS-2021-516-F&E, https://repub.eur.nl/pub/134947

Arampatzi, E., *Subjective Well-Being in Times of Crises: Evidence on the Wider Impact of Economic Crises and Turmoil on Subjective Well-Being*, Promotors: Prof. H.R. Commandeur, Prof. F. van Oort & Dr. M.J. Burger, EPS-2018-459-S&E, https://repub.eur.nl/pub/111830

Arslan, A.M., *Operational Strategies for On-demand Delivery Services*, Promotors: Prof. R.A. Zuidwijk & Dr. N.A.H. Agatz, EPS-2019-481-LIS, https://repub.eur.nl/pub/126463

Aydin, Z., *Mobile Consumers and Applications: Essays on Mobile Marketing*, Promotors: Prof. G.H. van Bruggen & Dr. B. Ataman, EPS-2021-519-MKT, https://repub.eur.nl/pub/135352

Azadeh, K., *Robotized Warehouses: Design and Performance Analysis*, Promotors: Prof. dr. ir M.B.M. de Koster & Prof. D. Roy, EPS-2021-515-LIS, https://repub.eur.nl/pub/135208

Avci, E., *Surveillance of Complex Auction Markets: a Market Policy Analytics Approach*, Promotors: Prof. W. Ketter, Prof. H.W.G.M. van Heck & Prof. D.W. Bunn, EPS-2018-426-LIS, https://repub.eur.nl/pub/106286

Balen, T.H. van, *Challenges of Early Stage Entrepreneurs: the Roles of Vision Communication and Team Membership Change*, Promotors: Prof. J.C.M. van den Ende & Dr. M. Tarakci, EPS-2019-468-LIS, https://repub.eur.nl/pub/115654

Bansraj, S.C., *The Principles of Private Equity: Ownership and Acquisitions*, Promotors: Prof. J.T.J Smit & Dr. V. Volosovych, EPS-2020-507-F&A, https://repub.eur.nl/pub/132329

Bavato, D., *With New Eyes: The recognition of novelty and novel ideas*, Promotors: Prof. D.A. Stam & Dr. S. Tasselli, EPS-2020-500-LIS, https://repub.eur.nl/pub/134264

Bernoster, I., *Essays at the Intersection of Psychology, Biology, and Entrepreneurship*, Promotors: Prof. A.R. Thurik, Prof. I.H.A. Franken & Prof. P.J.F. Groenen, EPS-2018-463-S&E, https://repub.eur.nl/pub/113907

Blagoeva, R.R., *The Hard Power Of Soft Power: A behavioral strategy perspective on how power, reputation, and status affect firms*, Promotors: Prof. J.J.P. Jansen & Prof. T.J.M. Mom, EPS-2020-495-S&E, https://repub.eur.nl/pub/127681

Bouman, P., *Passengers, Crowding and Complexity: Models for Passenger Oriented Public Transport*, Promotors: Prof. L.G. Kroon, Prof. A. Schöbel & Prof. P.H.M. Vervest, EPS-2017-420-LIS, https://repub.eur.nl/pub/100767

Breugem, T., *Crew Planning at Netherlands Railways: Improving Fairness, Attractiveness, and Efficiency*, Promotors: Prof. D. Huisman & Dr. T.A.B. Dollevoet, EPS-2020-494-LIS, https://repub.eur.nl/pub/124016

Bunderen, L. van, *Tug-of-War: Why and when teams get embroiled in power struggles*, Promotors: Prof. D.L. van Knippenberg & Dr. L. Greer, EPS-2018-446-ORG, https://repub.eur.nl/pub/105346

Burg, G.J.J. van den, *Algorithms for Multiclass Classification and Regularized Regression*, Promotors: Prof. P.J.F. Groenen & Dr. A. Alfons, EPS-2018-442-MKT, https://repub.eur.nl/pub/103929

Chammas, G., *Portfolio concentration*, Promotor: Prof. J. Spronk, EPS-2017-410-F&E, https://repub.eur.nl/pub/94975

Chan, H.Y., *Decoding the consumer's brain: Neural representations of consumer experience*, Promotors: Prof. A. Smidts & Dr. M.A.S. Boksem, EPS-2019-493-MKT, https://repub.eur.nl/pub/124931

Couwenberg, L., *Context dependent valuation: A neuroscientific perspective on consumer decision-making*, Promotors: Prof. A. Smit, Prof. A.G. Sanfrey & Dr. M.A.S. Boksem, EPS-2020-505-MKT, https://repub.eur.nl/pub/129601

Dalmeijer, K., *Time Window Assignment in Distribution Networks*, Promotors: Prof. A.P.M. Wagelmans & Dr. R. Spliet, EPS-2019-486-LIS, https://repub.eur.nl/pub/120773

Dennerlein, T., *Empowering Leadership and Employees' Achievement Motivations: the Role of Self-Efficacy and Goal Orientations in the Empowering Leadership Process*, Promotors: Prof. D.L. van Knippenberg & Dr. J. Dietz, EPS-2017-414-ORG, https://repub.eur.nl/pub/98438

Dolgova, E., *On Getting Along and Getting Ahead: How Personality Affects Social Network Dynamics*, Promotors: Prof. P.P.M.A.R. Heugens & Prof. M.C. Schippers, EPS-2019-455-S&E, https://repub.eur.nl/pub/119150

Duijzer, L.E., *Mathematical Optimization in Vaccine Allocation*, Promotors: Prof. R. Dekker & Dr. W.L. van Jaarsveld, EPS-2017-430-LIS, https://repub.eur.nl/pub/101487

Fasaei, H., *Changing the Narrative: The Behavioral Effects of Social Evaluations on the Decision Making of Organizations*, Promotors: Prof. J.J.P. Jansen, Prof. T.J.M. Mom & Dr. M.P. Tempelaar, EPS-2020-492-S&E, https://repub.eur.nl/pub/129598

Eijlers, E., *Emotional Experience and Advertising Effectiveness: on the use of EEG in marketing*, Promotors: Prof. A. Smidts & Prof. M.A.S. Boksem, EPS-2019-487-MKT, https://repub.eur.nl/pub/124053

El Nayal, O.S.A.N., *Firms and the State: An Examination of Corporate Political Activity and the Business-Government Interface*, Promotors: Prof. J. van Oosterhout & Dr. M. van Essen, EPS-2018-469-S&E, https://repub.eur.nl/pub/114683

Feng, Y., *The Effectiveness of Corporate Governance Mechanisms and Leadership Structure: Impacts on strategic change and firm performance*, Promotors: Prof. F.A.J. van den Bosch, Prof. H.W. Volberda & Dr. J.S. Sidhu, EPS-2017-389-S&E, https://repub.eur.nl/pub/98470

Frick, T.W., *The Implications of Advertising Personalization for Firms, Consumer, and Ad Platfroms*, Promotors: Prof. T. Li & Prof. H.W.G.M. van Heck, EPS-2018-452-LIS, https://repub.eur.nl/pub/110314

Fytraki, A.T., *Behavioral Effects in Consumer Evaluations of Recommendation Systems*, Promotors: Prof. B.G.C. Dellaert & Prof. T. Li, EPS-2018-427-MKT, https://repub.eur.nl/pub/110457

Gai, J., *Contextualized Consumers: Theories and Evidence on Consumer Ethics, Product Recommendations, and Self-Control*, Promotors: Prof. S. Puntoni & Prof. S.T.L. Sweldens, EPS-2020-498-MKT, https://repub.eur.nl/pub/127680

Ghazizadeh, P., *Empirical Studies on the Role of Financial Information in Asset and Capital Markets*, Promotors: Prof. A. de Jong & Prof. E. Peek, EPS-2019-470-F&A, https://repub.eur.nl/pub/114023

Giurge, L., *A Test of Time; A temporal and dynamic approach to power and ethics*, Promotors: Prof. M.H. van Dijke & Prof. D. De Cremer, EPS-2017-412-ORG, https://repub.eur.nl/pub/98451

Gobena, L., *Towards Integrating Antecedents of Voluntary Tax Compliance*, Promotors: Prof. M.H. van Dijke & Dr. P. Verboon, EPS-2017-436-ORG, https://repub.eur.nl/pub/103276

Groot, W.A., *Assessing Asset Pricing Anomalies*, Promotors: Prof. M.J.C.M. Verbeek & Prof. J.H. van Binsbergen, EPS-2017-437-F&A, https://repub.eur.nl/pub/103490

Hanselaar, R.M., *Raising Capital: On pricing, liquidity and incentives*, Promotors: Prof. M.A. van Dijk & Prof. P.G.J. Roosenboom, EPS-2018-429-F&A, https://repub.eur.nl/pub/113274

Harms, J.A., *Essays on the Behavioral Economics of Social Preferences and Bounded Rationality*, Promotors: Prof. H.R. Commandeur & Dr. K.E.H. Maas, EPS-2018-457-S&E, https://repub.eur.nl/pub/108831

Hendriks, G., *Multinational Enterprises and Limits to International Growth: Links between Domestic and Foreign Activities in a Firm's Portfolio*, Promotors: Prof. P.P.M.A.R. Heugens & Dr. A.H.L. Slangen, EPS-2019-464-S&E, https://repub.eur.nl/pub/114981

Hengelaar, G.A., *The Proactive Incumbent: Holy grail or hidden gem? Investigating whether the Dutch electricity sector can overcome the incumbent's curse and lead the sustainability transition*, Promotors: Prof. R.J.M. van Tulder & Dr. K. Dittrich, EPS-2018-438-ORG, https://repub.eur.nl/pub/102953

Jacobs, B.J.D., *Marketing Analytics for High-Dimensional Assortments*, Promotors: Prof. A.C.D. Donkers & Prof. D. Fok, EPS-2017-445-MKT, https://repub.eur.nl/pub/103497

Jia, F., *The Value of Happiness in Entrepreneurship*, Promotors: Prof. D.L. van Knippenberg & Dr. Y. Zhang, EPS-2019-479-ORG, https://repub.eur.nl/pub/115990

Kahlen, M.T., *Virtual Power Plants of Electric Vehicles in Sustainable Smart Electricity Markets*, Promotors: Prof. W. Ketter & Prof. A. Gupta, EPS-2017-431-LIS, https://repub.eur.nl/pub/100844

Kampen, S. van, *The Cross-sectional and Time-series Dynamics of Corporate Finance: Empirical evidence from financially constrained firms*, Promotors: Prof. L. Norden & Prof. P.G.J. Roosenboom, EPS-2018-440-F&A, https://repub.eur.nl/pub/105245

Karali, E., *Investigating Routines and Dynamic Capabilities for Change and Innovation*, Promotors: Prof. H.W. Volberda, Prof. H.R. Commandeur & Dr. J.S. Sidhu, EPS-2018-454-S&E, https://repub.eur.nl/pub/106274

Keko, E., *Essays on Innovation Generation in Incumbent Firms*, Promotors: Prof. S. Stremersch & Dr. N.M.A. Camacho, EPS-2017-419-MKT, https://repub.eur.nl/pub/100841

Kerkkamp, R.B.O., *Optimisation Models for Supply Chain Coordination under Information Asymmetry*, Promotors: Prof. A.P.M. Wagelmans & Dr. W. van den Heuvel, EPS-2018-462-LIS, https://repub.eur.nl/pub/109770

Khattab, J., *Make Minorities Great Again: a contribution to workplace equity by identifying and addressing constraints and privileges*, Promotors: Prof. D.L. van Knippenberg & Dr. A. Nederveen Pieterse, EPS-2017-421-ORG, https://repub.eur.nl/pub/99311

Kim, T.Y., *Data-driven Warehouse Management in Global Supply Chains*, Promotors: Prof. R. Dekker & Dr. C. Heij, EPS-2018-449-LIS, https://repub.eur.nl/pub/109103

Klitsie, E.J., *Strategic Renewal in Institutional Contexts: The paradox of embedded agency*, Promotors: Prof. H.W. Volberda & Dr. S. Ansari, EPS-2018-444-S&E, https://repub.eur.nl/pub/106275

Koolen, D., *Market Risks and Strategies in Power Systems Integrating Renewable Energy*, Promotors: Prof. W. Ketter & Prof. R. Huisman, EPS-2019-467-LIS, https://repub.eur.nl/pub/115655

Kong, L., *Essays on Financial Coordination*, Promotor: Prof. M.J.C.M. Verbeek, Dr. D.G.J. Bongaerts & Dr. M.A. van Achter. EPS-2019-433-F&A, https://repub.eur.nl/pub/114516

Korman, B., *Leader-Subordinate Relations: The Good, the Bad and the Paradoxical*, Promotors: S.R. Giessner & Prof. C. Tröster, EPS-2021-511-ORG, https://repub.eur.nl/pub/135365

Kyosev, G.S., *Essays on Factor Investing*, Promotors: Prof. M.J.C.M. Verbeek & Dr. J.J. Huij, EPS-2019-474-F&A, https://repub.eur.nl/pub/116463

Lamballais Tessensohn, T., *Optimizing the Performance of Robotic Mobile Fulfillment Systems*, Promotors: Prof. M.B.M de Koster, Prof. R. Dekker & Dr. D. Roy, EPS-2019-411-LIS, https://repub.eur.nl/pub/116477

Leung, W.L., *How Technology Shapes Consumption: Implications for Identity and Judgement*, Promotors: Prof. S. Puntoni & Dr. G. Paolacci, EPS-2019-485-MKT, https://repub.eur.nl/pub/117432

Li, W., *Competition in the Retail Market of Consumer Packaged Goods*, Promotors: Prof. D.Fok & Prof. Ph.H.B.F. Franses, EPS-2021-503-MKT, https://repub.eur.nl/pub/134873

Li, X., *Dynamic Decision Making under Supply Chain Competition*, Promotors: Prof. M.B.M de Koster, Prof. R. Dekker & Prof. R. Zuidwijk, EPS-2018-466-LIS, https://repub.eur.nl/pub/114028

Liu, N., *Behavioral Biases in Interpersonal Contexts*, Promotors: Prof. A. Baillon & Prof. H. Bleichrodt, EPS-2017-408-MKT, https://repub.eur.nl/pub/95487

Maas, A.J.J., *Organizations and their external context: Impressions across time and space*, Promotors: Prof. P.P.M.A.R. Heugens & Prof. T.H. Reus, EPS-2019-478-S&E, https://repub.eur.nl/pub/116480

Maira, E., *Consumers and Producers*, Promotors: Prof. S. Puntoni & Prof. C. Fuchs, EPS-2018-439-MKT, https://repub.eur.nl/pub/104387

Manouchehrabadi, B., *Information, Communication and Organizational Behavior*, Promotors: Prof. G.W.J. Hendrikse & Dr. O.H. Swank, EPS-2020-502-ORG, https://repub.eur.nl/pub/132185

Matawlie, N., *Through Mind and Behaviour to Financial Decisions*, Promotors: Prof. J.T.J. Smit & Prof. P. Verwijmeren, EPS-2020-501-F&A, https://repub.eur.nl/pub/134265

Mirzaei, M., *Advanced Storage and Retrieval Policies in Automated Warehouses*, Promotors: Prof. M.B.M. de Koster & Dr. N. Zaerpour, EPS-2020-490-LIS, https://repub.eur.nl/pub/125975

Nair, K.P., *Strengthening Corporate Leadership Research: The relevance of biological explanations*, Promotors: Prof. J. van Oosterhout & Prof. P.P.M.A.R. Heugens, EPS-2019-480-S&E, https://repub.eur.nl/pub/120023

Nullmeier, F.M.E., *Effective contracting of uncertain performance outcomes: Allocating responsibility for performance outcomes to align goals across supply chain actors*, Promotors: Prof. J.Y.F. Wynstra & Prof. E.M. van Raaij, EPS-2019-484-LIS, https://repub.eur.nl/pub/118723

Okbay, A., *Essays on Genetics and the Social Sciences*, Promotors: Prof. A.R. Thurik, Prof. Ph.D. Koellinger & Prof. P.J.F. Groenen, EPS-2017-413-S&E, https://repub.eur.nl/pub/95489

Peng, X., *Innovation, Member Sorting, and Evaluation of Agricultural Cooperatives*, Promotor: Prof. G.W.J. Hendriks, EPS-2017-409-ORG, https://repub.eur.nl/pub/94976

Petruchenya, A., *Essays on Cooperatives: Emergence, Retained Earnings, and Market Shares*, Promotors: Prof. G.W.J. Hendriks & Dr. Y. Zhang, EPS-2018-447-ORG, https://repub.eur.nl/pub/105243

Plessis, C. du, *Influencers: The Role of Social Influence in Marketing*, Promotors: Prof. S. Puntoni & Prof. S.T.L.R. Sweldens, EPS-2017-425-MKT, https://repub.eur.nl/pub/103265

Pocock, M., *Status Inequalities in Business Exchange Relations in Luxury Markets*, Promotors: Prof. C.B.M. van Riel & Dr. G.A.J.M. Berens, EPS-2017-346-ORG, https://repub.eur.nl/pub/98647

Polinder, G.J., *New Models and Applications for Railway Timetabling*, Promotors: Prof. D. Huisman & Dr. M.E. Schmidt, EPS-2020-514-LIS, https://repub.eur.nl/pub/134600

Pozharliev, R., *Social Neuromarketing: The role of social context in measuring advertising effectiveness*, Promotors: Prof. W.J.M.I. Verbeke & Prof. J.W. van Strien, EPS-2017-402-MKT, https://repub.eur.nl/pub/95528

Qian, Z., *Time-Varying Integration and Portfolio Choices in the European Capital Markets*, Promotors: Prof. W.F.C. Verschoor, Prof. R.C.J. Zwinkels & Prof. M.A. Pieterse-Bloem, EPS-2020-488-F&A, https://repub.eur.nl/pub/124984

Reh, S.G., *A Temporal Perspective on Social Comparisons in Organizations*, Promotors: Prof. S.R. Giessner, Prof. N. van Quaquebeke & Dr. C. Troster, EPS-2018-471-ORG, https://repub.eur.nl/pub/114522

Riessen, B. van, *Optimal Transportation Plans and Portfolios for Synchromodal Container Networks*, Promotors: Prof. R. Dekker & Prof. R.R. Negenborn, EPS-2018-448-LIS, https://repub.eur.nl/pub/105248

Romochkina, I.V., *When Interests Collide: Understanding and modeling interests alignment using fair pricing in the context of interorganizational information systems*, Promotors: Prof. R.A. Zuidwijk & Prof. P.J. van Baalen, EPS-2020-451-LIS, https://repub.eur.nl/pub/127244

Schie, R.J.G. van, *Planning for Retirement: Save More or Retire Later?*, Promotors: Prof. B.G.C. Dellaert & Prof. A.C.D. Donkers, EOS-2017-415-MKT, https://repub.eur.nl/pub/100846

Schneidmüller, T., *Engaging with Emerging Technologies: Socio-cognitive foundations of incumbent response*, Promotors: Prof. H. Volberda & Dr. S.M. Ansari, EPS-2020-509-S&E, https://repub.eur.nl/pub/131124

Schouten, K.I.M., *Semantics-driven Aspect-based Sentiment Analysis*, Promotors: Prof. F.M.G. de Jong, Prof. R. Dekker & Dr. F. Frasincar, EPS-2018-453-LIS, https://repub.eur.nl/pub/112161

Sihag, V., *The Effectiveness of Organizational Controls: A meta-analytic review and an investigation in NPD outsourcing*, Promotors: Prof. J.C.M. van den Ende & Dr. S.A. Rijsdijk, EPS-2019-476-LIS, https://repub.eur.nl/pub/115931

Slob, E., *Integrating Genetics into Economics*, Promotors: Prof. A.R. Thurik, Prof. P.J.F. Groenen & Dr. C.A. Rietveld, EPS-2021-517-S&E, https://repub.eur.nl/pub/135159

Smolka, K.M., *Essays on Entrepreneurial Cognition, Institution Building and Industry Emergence*, Promotors: P.P.M.A.R. Heugens & Prof. J.P. Cornelissen, EPS-2019-483-S&E, https://repub.eur.nl/pub/118760

Straeter, L.M., *Interpersonal Consumer Decision Making*, Promotors: Prof. S.M.J. van Osselaer & Dr. I.E. de Hooge, EPS-2017-423-MKT, https://repub.eur.nl/pub/100819

Stuppy, A., *Essays on Product Quality*, Promotors: Prof. S.M.J. van Osselaer & Dr. N.L. Mead, EPS-2018-461-MKT, https://repub.eur.nl/pub/111375

Subaşi, B., *Demographic Dissimilarity, Information Access and Individual Performance*, Promotors: Prof. D.L. van Knippenberg & Dr. W.P. van Ginkel, EPS-2017-422-ORG, https://repub.eur.nl/pub/103495

Suurmond, R., *In Pursuit of Supplier Knowledge: Leveraging capabilities and dividing responsibilities in product and service contexts*, Promotors: Prof. J.Y.F Wynstra & Prof. J. Dul, EPS-2018-475-LIS, https://repub.eur.nl/pub/115138

Toxopeus, H.S., *Financing sustainable innovation: From a principal-agent to a collective action perspective*, Promotors: Prof. H.R. Commandeur & Dr. K.E.H. Maas, EPS-2019-458-S&E, https://repub.eur.nl/pub/114018

Turturea, R., *Overcoming Resource Constraints: The Role of Creative Resourcing and Equity Crowdfunding in Financing Entrepreneurial Ventures*, Promotors: Prof. P.P.M.A.R. Heugens, Prof. J.J.P. Jansen & Dr. I. Verheuil, EPS-2019-472-S&E, https://repub.eur.nl/pub/112859

Valboni, R., *Building Organizational (Dis-)Abilities: The impact of learning on the performance of mergers and acquisitions*, Promotors: Prof. T.H. Reus & Dr. A.H.L. Slangen, EPS-2020-407-S&E, https://repub.eur.nl/pub/125226

Vandic, D., *Intelligent Information Systems for Web Product Search*, Promotors: Prof. U. Kaymak & Dr. Frasincar, EPS-2017-405-LIS, https://repub.eur.nl/pub/95490

Verbeek, R.W.M., *Essays on Empirical Asset Pricing*, Promotors: Prof. M.A. van Dijk & Dr. M. Szymanowska, EPS-2017-441-F&A, https://repub.eur.nl/pub/102977

Visser, T.R., *Vehicle Routing and Time Slot Management in Online Retailing*, Promotors: Prof. A.P.M. Wagelmans & Dr. R. Spliet, EPS-2019-482-LIS, https://repub.eur.nl/pub/120772

Vlaming, R. de., *Linear Mixed Models in Statistical Genetics*, Promotors: Prof. A.R. Thurik, Prof. P.J.F. Groenen & Prof. Ph.D. Koellinger, EPS-2017-416-S&E, https://repub.eur.nl/pub/100428

Vongswasdi, P., *Accelerating Leadership Development: An evidence-based perspective*, Promotors: Prof. D. van Dierendonck & Dr. H.L. Leroy, EPS-2020-512-ORG, https://repub.eur.nl/pub/134079

Vries, H. de, *Evidence-Based Optimization in Humanitarian Logistics*, Promotors: Prof. A.P.M. Wagelmans & Prof. J.J. van de Klundert, EPS-2017-435-LIS, https://repub.eur.nl/pub/102771

Wang, R., *Corporate Environmentalism in China*, Promotors: Prof. P.P.M.A.R. Heugens & Dr. F. Wijen, EPS-2017-417-S&E, https://repub.eur.nl/pub/99987

Wang, R., *Those Who Move Stock Prices*, Promotors: Prof. P. Verwijmeren & Prof. S. van Bekkum, EPS-2019-491-F&A, https://repub.eur.nl/pub/129057

Wasesa, M., *Agent-based inter-organizational systems in advanced logistics operations*, Promotors: Prof. H.W.G.M. van Heck, Prof. R.A. Zuidwijk & Dr. A.W. Stam, EPS-2017-LIS-424, https://repub.eur.nl/pub/100527

Wessels, C., *Flexible Working Practices: How Employees Can Reap the Benefits for Engagement and Performance*, Promotors: Prof. H.W.G.M. van Heck, Prof. P.J. van Baalen & Prof. M.C. Schippers, EPS-2017-418-LIS, https://repub.eur.nl/

Wiegmann, P.M., *Setting the Stage for Innovation: Balancing Diverse Interests through Standardisation*, Promotors: Prof. H.J. de Vries & Prof. K. Blind, EPS-2019-473-LIS, https://repub.eur.nl/pub/114519

Wijaya, H.R., *Praise the Lord!: Infusing Values and Emotions into Neo-Institutional Theory*, Promotors: Prof. P.P.M.A.R. Heugens & Prof. J.P. Cornelissen, EPS-2019-450-S&E, https://repub.eur.nl/pub/115973

Williams, A.N., *Make Our Planet Great Again: A Systems Perspective of Corporate Sustainability*, Promotors: Prof. G.M. Whiteman & Dr. S. Kennedy, EPS-2018-456-ORG, https://repub.eur.nl/pub/111032

Witte, C.T., *Bloody Business: Multinational investment in an increasingly conflict-afflicted world*, Promotors: Prof. H.P.G. Pennings, Prof. H.R. Commandeur & Dr. M.J. Burger, EPS-2018-443-S&E, https://repub.eur.nl/pub/104027

Wu, J., *A Configural Approach to Understanding Voice Behavior in Teams*, Promotors: Prof. D.L. van Knippenberg & Prof. S.R. Giessner, EPS-2020-510-ORG, https://repub.eur.nl/pub/132184

Ye, Q.C., *Multi-objective Optimization Methods for Allocation and Prediction*, Promotors: Prof. R. Dekker & Dr. Y. Zhang, EPS-2019-460-LIS, https://repub.eur.nl/pub/116462

Yuan, Y., *The Emergence of Team Creativity: a social network perspective*, Promotors: Prof. D.L. van Knippenberg & Dr. D.A. Stam, EPS-2017-434-ORG, https://repub.eur.nl/pub/100847

Zhang, Q., *Financing and Regulatory Frictions in Mergers and Acquisitions*, Promotors: Prof. P.G.J. Roosenboom & Prof. A. de Jong, EPS-2018-428-F&A, https://repub.eur.nl/pub/103871

Significant cost reductions, both monetary and environmentally, can be realised by collaborative efforts in logistics. Specifically, we study centralised horizontal collaborations between logistic service providers in the field of transportation. In this particular setting, cost reductions are achieved by pooling requests, combining delivery routes and collectively serving new customers.

First, we consider a centralised horizontal collaboration between logistic service providers that is governed by a third party. Here, we assume that the collaborative operations are planned by this third party. We focus on how to determine a stable allocation of the overall cost or profit among the logistic service providers. That is, no subset of logistic service providers is better off by leaving the collaboration. For two different collaborative settings, we propose algorithms to determine allocations which ensure a stable collaboration among the logistic service providers, or to show that no such allocation exists. Moreover, we demonstrate the efficiency of these algorithms.

Second, we focus on a centralised horizontal collaboration that is not governed by a third party. That is, the logistic service providers need to plan their joint operations together. We propose a framework, based on cooperative game theory, in which the algorithms of each logistic service provider can be used to determine a planning for their joint operations. In doing so, any logistic service provider or third party with a good algorithm can add value to the collaboration, even without being a logistic service provider, i.e., a consultant. We study this setting both analytically and numerically for collaborations in transportation, and provide insights into the value of the quality of an algorithm in collaborations.

ERiM

The Erasmus Research Institute of Management (ERIM) is the Research School (Onderzoekschool) in the field of management of the Erasmus University Rotterdam. The founding participants of ERIM are the Rotterdam School of Management (RSM), and the Erasmus School of Economics (ESE). ERIM was founded in 1999 and is officially accredited by the Royal Netherlands Academy of Arts and Sciences (KNAW). The research undertaken by ERIM is focused on the management of the firm in its environment, its intra- and interfirm relations, and its business processes in their interdependent connections.

The objective of ERIM is to carry out first rate research in management, and to offer an advanced doctoral programme in Research in Management. Within ERIM, over three hundred senior researchers and PhD candidates are active in the different research programmes. From a variety of academic backgrounds and expertises, the ERIM community is united in striving for excellence and working at the forefront of creating new business knowledge.

ERiM

ERIM PhD Series
Research in Management

Erasmus University Rotterdam (EUR)
Erasmus Research Institute of Management
Mandeville (T) Building
Burgemeester Oudlaan 50
3062 PA Rotterdam, The Netherlands

P.O. Box 1738
3000 DR Rotterdam, The Netherlands
T +31 10 408 1182
E info@erim.eur.nl
W www.erim.eur.nl