The Effect of Algorithmic Capabilities on Cooperative Games

Mathijs van Zon[1], Remy Spliet, Wilco van den Heuvel

Econometric Institute, Erasmus School of Economics, Erasmus University Rotterdam

PO Box 1738, Rotterdam 3000 DR, The Netherlands

Econometric Institute Report Series - EI2021-02

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 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 paper we describe the core of AQI games and analyse the effects of improving the algorithm of a single player and of adding a consultant to 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.

[1]Corresponding author. E-mail: vanzon@ese.eur.nl

Keywords: Collaborative transportation, Cooperative game theory, Vehicle Routing

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 paper, 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. (2021) (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. (2021). 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 paper, 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 paper, 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 based 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 paper is structured as follows. First, in Section 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 3 we introduce the different allocation concepts and methods which are considered in this paper. In Section 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 5 and the inclusion of consultants in Section 6. In Section 7, we present and discuss our numerical experiments. Finally, in Section 8 we provide a summary and some concluding remarks.

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 2.1 and 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 2.3. For ease of writing, in the remainder of this paper, 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 paper straightforwardly apply also to the case where the underlying optimisation problem is a maximisation problem.

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

$$C^B(S) = \min_{i \in S} \{C_i(S)\}. \tag{2.1}$$

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 usually is no such guarantee.

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

$$C^P(S) = \min_{\mathcal{P} \in \mathrm{Part}(S)} \left\{ \sum_{P \in \mathcal{P}} \min_{i \in P} \{C_i(P)\} \right\}, \tag{2.2}$$

where $\mathrm{Part}(S)$ 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.

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 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 $L^- = \bigcup_{z \in Z}$ 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

$$C(S) = \sum_{p=1}^{P} \min_{i \in S} \{C_i(Z^p, K^p)\}. \tag{2.3}$$

When using cost function (2.3), the players consider the best achievable cost given a decomposition of the instance associated with coalition $S \subseteq N$.

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 paper.

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

$$y_i = \frac{C(\{i\})}{\sum_{j \in N} C(\{j\})} C(N). \tag{3.1}$$

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

$$y_i = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(n - |S| - 1)!}{n!} (c(S \cup \{i\}) - c(S)). \tag{3.2}$$

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

$$\text{Core}(\langle N, C \rangle) = \{y \in \mathbb{R}^n : y(S) \leq C(S), y(N) = C(N)\}. \tag{3.3}$$

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 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 4.1. *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 4.1 applies, is redundant in describing the core of the BAQI game. This result is summarised in Corollary 4.1.1.

Corollary 4.1.1. *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 4.1 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 4.1 and Corollary 4.1.1 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 4.1 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 4.1. 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 4.2. *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 4.1 and Corollary 4.1.1 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 4.2, 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 4.1.

Moreover, Theorem 4.2 also leads directly to Corollaries 4.2.1 and 4.2.2 which describe relations between the cores of the BAQI game and the PAQI game.

Corollary 4.2.1. *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 4.2 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 4.2.1 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 4.2.2.

Corollary 4.2.2. *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 4.2 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 4.3. *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 4.3 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.

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 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 5.1 we consider an improvement solely in the grand-coalition cost. Thereafter, in Section 5.2, we consider improvements of any coalition.

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

$$y_i^+ = \min_{y\in\mathbb{R}^n}\{y_i : y(S) \leq C(S), y(N) = C(N)\}, \tag{5.1}$$

$$y_i'^+ = \min_{y\in\mathbb{R}^n}\{y_i : y(S) \leq C'(S), y(N) = C'(N)\}. \tag{5.2}$$

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

$$y_i^- = \max_{y\in\mathbb{R}^n}\{y_i : y(S) \leq C(S), y(N) = C(N)\}, \tag{5.3}$$

$$y_i'^- = \max_{y\in\mathbb{R}^n}\{y_i : y(S) \leq C'(S), y(N) = C'(N)\}. \tag{5.4}$$

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

Theorem 5.1. *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

$$\varepsilon = \min\left(C(\{j\}) - y_j^-, \frac{C(N) - C'(N)}{n-2}\right) \text{ and } \delta = \frac{C(N) - C'(N) + \varepsilon}{n-1} \tag{5.5}$$

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 5.1 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.

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 5.2. *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 A. □

From Theorem 5.2, 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.

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.

$$C_i(\{i\}) = 0, \tag{6.1}$$

$$C_j(S \cup \{i\}) = C_j(S) \quad \forall\, S \subseteq N \backslash \{i\},\ j \in N : j \neq i. \tag{6.2}$$

Equality (6.1) 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 (6.2) 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 (6.2) 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 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 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 6.1.*

Table 6.1: 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 6.1. *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 6.2. *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 6.1, 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 6.2, to show that players may benefit from including a copy of the consultant.

Example 6.3. *In this example, we add a copy to the PAQI game as proposed in Example 6.2. The resulting cost functions are given in Table 6.3.*

Table 6.2: 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 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 6.3.*

Table 6.3: 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.

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 7.1. Here, we also detail the quality of the algorithms as well as the characteristics of the cores of these games. In Section 7.2 we report the effect of improving the algorithm of a single player on the cost allocated to each player. Finally, in Section 7.3 we consider the effect of including a consultant on the cost or profit allocated to each of the players.

7.1 Data generation and description

First, in Section 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 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 7.1.3 we report on the existence of the cores of the basic and partitioning AQI game instances.

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 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 mentioned 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 3 as well as the worst-case and best-case allocations for each player as introduced in Section 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.

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 7.1.

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 7.1: Performance of the different algorithms.

The improvements of the different player algorithms are gradual by about 1-2% for each improvement. 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 7.2, 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).

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

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 7.2: Pairwise comparison of the performance of the algorithms.

the number of iterations do no 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.

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 7.3, 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 7.3: 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.

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.

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 7.4. Here, both the Star allocation and Shapley value are in the core for the initial and improved game.

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 7.4: Allocations for each player of the initial and improved three-player 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 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.

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 7.3.

The results are presented in Table 7.5, 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 7.5: 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 7.1. 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 7.2. 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 7.2. 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 7.2, 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.

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.

7.3.1 Example

We consider the initial AQI game with three-players as presented in 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 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 7.6. 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 7.6: 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.

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 7.1. The results are presented in Table 7.7. 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 7.7: 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 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 7.8. 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 7.8: 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 clearly is 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.

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.

References

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

R. Baldacci, E. Bartolini, and A. Mingozzi. An exact algorithm for the pickup and delivery problem with time windows. *Operations research*, 59(2):414–426, 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.

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

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.

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.

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

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.

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, 5 2016.

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.

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, sep 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, July 2015.

S. N. Parragh, K. F. Doerner, and R. F. Hartl. A survey on pickup and delivery problems. *Journal für Betriebswirtschaft*, 58(2):81–117, may 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, nov 2006.

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

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

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

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

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

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

Appendix

A Proof of Theorem 5.2

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 5.2.

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.

$$y_j^- = \max \quad y_j, \tag{8.1}$$

$$\text{s.t.} \quad \sum_{i \in S} y_i \leq C(S) \qquad \forall S \subset N, \tag{8.2}$$

$$\sum_{i \in N} y_i \geq C(N), \tag{8.3}$$

$$y_i \in \mathbb{R} \qquad i \in N. \tag{8.4}$$

Here, Objective (8.1) ensures that the cost allocated to a player j is maximised. Moreover, Constraints (8.2)-(8.3) 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 (8.2)-(8.3) are still satisfied. Hence, y' is also optimal for (8.1)-(8.4) and $y(N) = c(N)$.

Next, we consider the dual of (8.1)-(8.4). Consider dual variables $\lambda_S \geq 0$ associated with Constraints (8.2) and dual variable $\lambda_N \leq 0$ associated with Constraint (8.3). The dual can be formulated as follows

$$y_j^- = \min \quad \sum_{S \subset N} \lambda_S C(S) + \lambda_N C(N), \tag{8.5}$$

$$\text{s.t.} \quad \sum_{S \subset N : i \in S} \lambda_S + \lambda_N = 0 \qquad \forall i \in N \backslash \{j\}, \tag{8.6}$$

$$\sum_{S \subset N : j \in S} \lambda_S + \lambda_N = 1, \tag{8.7}$$

$$\lambda_S \geq 0 \qquad S \subset N, \tag{8.8}$$

$$\lambda_N \leq 0. \tag{8.9}$$

Note that we can omit Constraint (8.9) since it is implied by Constraints (8.6). 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

$$y_j^- = C(N) + \min \quad \sum_{S \subset N} \lambda_S C(S) - C(N) \sum_{S \subset N : j \in S} \lambda_S, \tag{8.10}$$

$$\text{s.t.} \quad \sum_{S \subset N : j \in S, i \notin S} \lambda_S - \sum_{S \subset N : i \in S, j \notin S} \lambda_S = 1 \qquad \forall i \in N \backslash \{j\}, \tag{8.11}$$

$$\lambda_S \geq 0 \qquad S \subset N. \tag{8.12}$$

Next, we provide a proof for Theorem 5.2. 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 (8.10) for games $\langle N, C \rangle$ and $\langle N, C' \rangle$, respectively. The difference between the objective values is as follows

$$y(\lambda_S) - y'(\lambda_S) = C(N) - C'(N) + \sum_{S \subset N : j \in S} \lambda_S \left[C(S) - C'(S) \right] + \left[C'(N) - C(N) \right] \sum_{S \subset N : j \in S} \lambda_S \tag{8.13}$$

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

$$y(\lambda_S) - y'(\lambda_S) \geq C(N) - C'(N) + [\Delta + C'(N) - C(N)] \sum_{S \subset N: j \in S} \lambda_S. \tag{8.14}$$

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

$$y(\lambda_S) - y'(\lambda_S) \geq C(N) - C'(N), \tag{8.15}$$

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