Nucleolus, Happy Nucleolus, and Vehicle Routing

Daniel Ebert Antonia Ellerbrock

{ebert,ellerbrock}@dm.uni-bonn.de

Research Institute for Discrete Mathematics, University of Bonn

Abstract

We study the recently introduced fair division concept of the *happy nucleolus* for cost allocation among players in a cooperative game, with special focus on its computation. The happy nucleolus applies the same fairness criterion as the well-established nucleolus but with reduced total value. Still, we show that the relation between the two concepts is quite involved, and intuitive properties do not hold – e.g., the entry of a player in the happy nucleolus can be larger than the entry of the same player in the nucleolus, even for monotone and subadditive games. This refutes conjectures of Meir et al. [27].

Further, we study the separation problem of the linear programs appearing in the MPS scheme for computing the (happy) nucleolus. It includes linear subspace avoidance constraints, which can be handled efficiently for problems with a certain dynamic programming formulation due to Köhnemann and Toth [23]. We show how to get rid of these constraints for *all* monotone games if we allow for an arbitrarily small error of ε, thus conserving known approximation guarantees for the same problem without subspace avoidance.

Finally, we focus on practical results at the example of vehicle routing games by designing an efficient heuristic based on our previous insights and past work, and demonstrate its power.

1 Introduction

We study *cost allocations* for cooperative games.

Definition 1 (Cooperative game)**.** A *cooperative game* is a pair (P, c) of a finite set P of *players* and a cost function $c : 2^P \to \mathbb{R}$ on *coalitions* $S \subseteq P$ with $c(\emptyset) = 0$. (P, c) is called *monotone* if $c(S) \leq c(T)$ for all $S \subseteq T \subseteq P$, and *subadditive* if $c(S \cup T) \leq c(S) + c(T)$ for all $S, T \subseteq P$ with $S \cap T = \emptyset$.

Usually, c is not given explicitly, but by some combinatorial optimization problem, e.g., as the size of a maximum matching in the subgraph $G[S]$ of some graph G for $S \subseteq P$ [13].

Given a cooperative game (P, c), a *cost allocation* is a vector $y \in \mathbb{R}^P$. We will write $y(S) \coloneqq \sum_{p \in S} y_p$ for $S \subseteq P$. There are multiple concepts defining 'fair' cost allocations, such as the *Shapley value* [36], the τ-value [39] or the *nucleolus* [34]. In this work, we will focus on the latter, introduced by Schmeidler in 1969 [34], and a variant, the *happy nucleolus*, introduced by Blauth et al. in 2024 [6]. For a formal definiton, let $\text{excess}(S, y) \coloneqq c(S) - y(S)$ for $S \subseteq P$, $y \in \mathbb{R}^P$. The *excess vector* $\theta(y)$ contains all values $(\text{excess}(S, y))_{S \subseteq P}$ in non-decreasing order.

Definition 2 (Nucleolus [34])**.** For a cooperative game (P, c), the *nucleolus* $\mathcal{N}$ is the unique cost allocation that lexicographically maximizes the excess vector $\theta(y)$ among all $y \in \mathbb{R}^P$ with $y(P) = \mathcal{V} \coloneqq c(P)$.

Not all coalitions need to be happy with the nucleolus: $y \in \mathbb{R}^P$ satisfies *happiness*, also referred to as *group rationality*, iff every coalition is *happy* with it, i.e., $y(S) \leq c(S)$ for all $S \subseteq P$.

Definition 3 (Happy nucleolus [6])**.** For a cooperative game (P, c), the *happy nucleolus* $\mathcal{N}_h$ is the unique cost allocation that lexicographically maximizes the excess vector $\theta(y)$ among all $y \in \mathbb{R}^P$ with

$$y(P) \;=\; \mathcal{V}_h \;:=\; \max\left\{z(P) : z \in \mathbb{R}^P \text{ satisfies happiness}\right\} \quad .$$

Nucleolus and happy nucleolus coincide iff $\mathcal{V} = \mathcal{V}_h$, or equivalently, iff the *core* of the game is non-empty: The core C consists of all cost allocations with total value $\mathcal{V}$ that satisfy happiness.

1.1 Related Work and Our Results

In this paper, we are interested in (efficient) computation of the happy nucleolus, which we believe makes most sense when applied to games that incentivize cooperation. Thus, our focus will be on monotone and subadditive games.

For certain classes of games, the nucleolus can be computed in (pseudo-)polynomial time [14, 8, 10, 38, 22, 23, 31], whereas for others, it is NP-hard [16, 19]. For so-called *balanced* games with non-empty core (e.g. [18, 29, 12]), happy nucleolus and nucleolus coincide, and the results transfer. Otherwise, NP-hardness of computing the happy nucleolus might be implied by NP-hardness of checking core emptiness or computing the so-called *cost of stability* $\mathcal{V} - \mathcal{V}_h$ [11, 8, 25, 9, 5, 4, 33, 3, 9]. For *set covering games*, Blauth et al. [6] gave a polynomial-time algorithm, while computing the nucleolus remains NP-hard, thus proving that the complexities of the two concepts can actually differ.

In Section 2, we will discuss that the relation between nucleolus and happy nucleolus is quite involved. We will give negative answers to conjectures by Meir et al. [27]. Informally speaking and simplified, we will find that

- the happy nucleolus is not dominated player-wise by the nucleolus,
- the happy nucleolus does not equal the nucleolus of the corresponding fractional game.

Moving on to computational aspects, our focus lies on the *MPS scheme* that was found by Kopelowitz, Maschler, Peleg and Shapley for nucleolus computation early on [24, 26], and later applied in various ways [28, 15]. It does not run in polynomial time in general, but needs to solve a linear number of linear programs with exponentially many constraints. In Section 3, we will apply the MPS scheme to the happy nucleolus, and discuss the separation problem over its linear programs in detail, thereby showing how to get rid of certain linear subspace avoidance constraints for all monotone games at the expense of an arbitrarily small additive error. Similarly, Könemann and Toth showed how to handle these constraints for all games that allow for a polynomial-sized dynamic program that solves the corresponding Prize-Collecting Coalition Problem [23]. Further, they proved that linear subspace avoidance constraints increase the complexity of some optimization problems from polynomial time solvability to NP-hardness. However, for the Prize-Collecting Coalition Problem, it is still open whether subspace avoidance can make the problem harder (i.e., whether our additive error is necessary).

The gained insights of Section 3 will be one building block of an efficient heuristic for computing the happy nucleolus in Section 4 on *vehicle routing games*. Nucleolus computation has been studied on practical vehicle routing instances by [8, 17, 15], to name only a few. Guajardo and Rönnqvist [21], as well as Schulte et al. [35] wrote surveys on cost allocation for these games. We are not aware of any working heuristics for computing the (happy) nucleolus on large instances

with hundreds of shipments – a size needed for practical application – and will close this gap. We will provide practical results demonstrating the quality of our heuristic by comparing to the exact happy nucleolus on random instances with 50 players, and by showing convergence on large random instances, where we are not aware of an efficient method for computing the exact happy nucleolus to compare against.

We believe that our work naturally implies future research, see Section 5.

2 Relation Between Nucleolus and Happy Nucleolus

Nucleolus and happy nucleolus are quite similar cost allocation concepts. They use the same measure of fairness, but differ in total value. We would like to understand their relation, possibly for deriving computational results. Surprisingly, even simple and intuitive properties do not hold, and we use this section to present two of them in detail.

2.1 Player-Wise Domination and the Extended Happy Core

As the happy nucleolus guarantees non-negative excess values everywhere, in particular for the grand coalition P, its total value $\mathcal{V}_h$ is never larger than the total value $\mathcal{V} = c(P)$ of the nucleolus. It was open whether this domination holds player-wise, i.e., whether $\mathcal{N}_h(p) \leq \mathcal{N}(p)$ for all $p \in P$. Intuitively, one might expect that when charging a reduced price total and applying the same fairness criterion, every single player should benefit. In [27], Meir, Rosenschein and Malizia conjectured this and related statements. We will give negative answers, demonstrating that the relation between $\mathcal{N}$ and $\mathcal{N}_h$ is quite involved.

Definition 4. For a cooperative game (P, c), the *least core* C_l contains all cost allocations y with $y(P) = \mathcal{V}$ that violate happiness by at most ε, i.e., $y(S) \leq c(S) + \varepsilon$ for all $S \subseteq P$, and ε is chosen as the smallest number such that there exists such a cost allocation [26]. The *happy core* C_h contains all cost allocations y with $y(P) = \mathcal{V}_h$ that satisfy happiness. The *extended happy core* $\mathrm{C}_h^{\uparrow}$ contains all cost allocations y with $y(P) = \mathcal{V}$ that dominate a point in the happy core player-wise, i.e., $y'_p \leq y_p$ for all $p \in P$ and some $y' \in \mathrm{C}_h$.

It is clear that the nucleolus is always part of the least core, and the happy nucleolus is always part of the happy core: $\mathcal{N} \in \mathrm{C}_l$ and $\mathcal{N}_h \in \mathrm{C}_h$. Meir et al. conjectured the inclusion $\mathrm{C}_l \subseteq \mathrm{C}_h^{\uparrow}$ (cf. Conjecture 1 in [27]), and $\mathcal{N}(p) \geq \mathcal{N}_h(p)$ for all $p \in P$ (cf. Section 5.1 in [27]). We will discuss an instance where we do not even have $\mathcal{N} \in \mathrm{C}_h^{\uparrow}$. Note that the properties of these core sets, nucleolus and happy nucleolus are equivalent for cost and profit sharing games under the transformation described in the following remark:

Remark 5 (Value functions). Cooperative games can alternatively be defined over *value functions* $\nu : 2^P \to \mathbb{R}$ that are interpreted as gains, i.e., $\mathrm{excess}(S, y) = y(S) - \nu(S)$ for a coalition S and a cost allocation y. We can transform a cost function c into a value function $\nu := -c_s$ with $c_s := c(S) + \sum_{p \in P} s_p$ and player shifts $s_p \in \mathbb{R}$ that are small enough such that $\nu \geq 0$. Then, $\mathrm{excess}_c(S, y) = \mathrm{excess}_{c_s}(S, y + s)$, and the (happy) nucleolus of (P, ν) results from the (happy) nucleolus of (P, c) by adding the player shifts s, and vice versa.

Recall that for set covering games, the cost of a coalition is determined by its minimum (not necessarily exact) covering cost. In the classic triangle instance with unit cost, happy nucleolus and nucleolus differ: $\mathcal{N}_h \equiv \frac{1}{2}$ and $\mathcal{N} \equiv \frac{2}{3}$. Fig. 1 uses three copies of this triangle, and proves Theorem 6:

Theorem 6 (Negative answers to conjectures in [27]). *There is a monotone and subadditive game (P, c), where $\mathcal{N} \notin \mathrm{C}_h^{\uparrow}$, so in particular $\mathrm{C}_l \not\subseteq \mathrm{C}_h^{\uparrow}$ and $\mathcal{N}_h(p) > \mathcal{N}(p)$ for some player $p \in P$.*

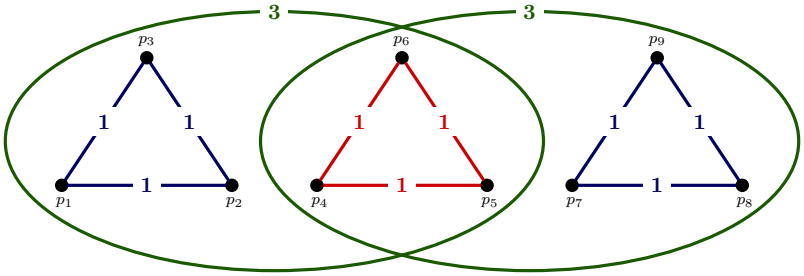


Figure 1: Set covering instance proving Theorem 6. Edges illustrate sets of size 2. We have $\mathcal{N}_h \equiv \frac{1}{2}$, $\mathcal{N}(p_i) = \frac{7}{15}$ for $i \in \{4, 5, 6\}$, and $\mathcal{N}(p_i) = \frac{3}{5}$ otherwise. Especially, $\mathcal{N}_h(p_4) > \mathcal{N}(p_4)$.

Proof. Consider the set covering instance depicted in Fig. 1, where all input sets are drawn with their costs. We have $\mathcal{N}_h \equiv \frac{1}{2}$, and this is the unique solution that satisfies happiness with a total value of $\mathcal{V}_h = \frac{9}{2}$. On the other hand, the nucleolus attains a total value of $c(P) = 5$, and balances the excess of the coalitions $P \setminus \{p_1\}, \{p_1, p_2, p_7, p_8\}$ and symmetric ones at $-\frac{2}{5}$. This determines $\mathcal{N}(p_i) = \frac{3}{5}$ for $i \in \{1, 2, 3, 7, 8, 9\}$ and, by symmetry, fixes the remaining values to $\mathcal{N}(p_i) = \frac{7}{15}$ for $i \in \{4, 5, 6\}$. In particular, $\mathcal{N}(p_4) = \frac{7}{15} < \frac{1}{2} = \mathcal{N}_h(p_4)$. Further, $\mathrm{C}_h = \{\mathcal{N}_h\}$. By $\mathcal{N}(p_4) < \mathcal{N}_h(p_4)$, $\mathcal{N} \notin \mathrm{C}_h^{\uparrow}$. With $\mathcal{N} \in \mathrm{C}_l$, this contradicts $\mathrm{C}_l \subseteq \mathrm{C}_h^{\uparrow}$. □

2.2 Fractional Costs

We will see that the happy nucleolus of a game is not always equal to the nucleolus of the corresponding fractional game, again by using set covering games.

Definition 7. For a subadditive cooperative game (P, c), its *fractional game* (P, c_f) is defined by *fractional costs* c_f for all coalitions $S \subseteq P$:

$$c_f(S) \;:=\; \min \left\{ \sum_{T \subseteq P} \alpha_T c(T) \;:\; \alpha_T \geq 0 \;\; \forall T \subseteq P \,, \sum_{p \in T \subseteq P} \alpha_T = 1 \;\; \forall p \in S \right\} .$$

We call a set $S \subseteq P$ *fractionally dominated* if $c_f(S) < c(S)$. One might think that fractionally dominated sets are irrelevant for determining the happy nucleolus. This would imply that the happy nucleolus of any game is equal to the nucleolus of the corresponding fractional game since $\mathcal{V}_h(P, c) = c_f(P)$. The example in Fig. 2 shows that this is not true in general.

Theorem 8. *There is a monotone and subadditive game (P, c) such that*

$$\mathcal{N}_h(P, c) \;\neq\; \mathcal{N}(P, c_f) \;=\; \mathcal{N}_h(P, c_f) \;.$$

Additionally, there is a set $S \subseteq P$ with $c_f(S) < c(S)$ such that increasing or decreasing $c(S)$ changes $\mathcal{N}_h(P, c)$.

Proof. Consider the set covering instance depicted in Fig. 2. Both (P, c) and (P, c_f) have a non-empty core, as shown by the cost allocation $y \equiv 5$ which satisfies happiness and has a total value of $30 = c(P) = c_f(P)$. Additionally, any cost allocation that is in the core of one of the two games has excess 0 on all sets with cost 10 by LP duality: Each of these sets appears in

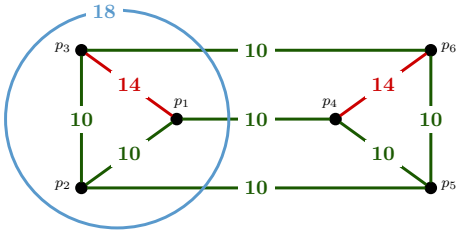


Figure 2: A set covering instance with a fractionally dominated set (blue), proving Theorem 8. Edges illustrate sets of size 2. Sets with the same cost are drawn in the same color.

some optimum fractional set covering of the instance. In particular, this is true for the (happy) nucleolus of both games.

The (happy) nucleolus of (P, c) then balances the excess values of the set with cost 18 and the set with cost 14 on the right. This results in

$$\mathcal{N}_h(P,c)_p \;=\; \mathcal{N}(P,c)_p \;=\; \begin{cases} 5+\frac{1}{3} & \text{for } p \in \{p_2, p_4, p_6\} \;, \\ 5-\frac{1}{3} & \text{for } p \in \{p_1, p_3, p_5\} \;, \end{cases}$$

and both sets having an excess value of $3+\frac{1}{3}$. It follows from the same calculation that increasing or decreasing the cost of the set with cost 18 changes the (happy) nucleolus of the instance.

For the fractional cost function c_f, the instance is symmetric because the set with cost 18 is fractionally dominated. Therefore, the (happy) nucleolus is given by $\mathcal{N}_h(P, c_f) = \mathcal{N}(P, c_f) \equiv 5$. This is determined by balancing the excess values of the coalitions $\{p_1, p_2, p_3\}$ and $\{p_4, p_5, p_6\}$, both with fractional cost 17. □

3 Separating over the Polytopes in the MPS Scheme

In this section, we study the Maschler-Peleg-Shapley (MPS) scheme which can be used to compute the nucleolus and the happy nucleolus. We show how the main subproblem that has to be solved can be reduced to an easier special case. For technical reasons, we assume that $\max_{S \subseteq P} \text{size}(c(S))$ is polynomially bounded, where $\text{size}(\cdot)$ denotes the bit complexity of the given number. This assumption is trivially true for all interesting problems we are aware of.

3.1 The MPS Scheme

As shown by Maschler, Peleg and Shapley [26], the nucleolus can be computed by solving a sequence of linear programs. An improved variant which needs at most a linear number of iterations was proposed by Solymosi [37]. It trivially extends to the happy nucleolus, see Fig. 3.

Theorem 9 (Maschler, Peleg and Shapley [26], Solymosi [37])**.** *The scheme described in Fig. 3 correctly computes the (happy) nucleolus in at most $|P|$ iterations.* □

In order to apply this scheme, one needs to be able to compute the total value V and solve the linear program (1). All other steps are trivial. For the happy nucleolus (in contrast to the nucleolus), computing $V = \mathcal{V}_h$ can also be reduced to solving (1) because $\mathcal{V}_h$ is the maximum value for which the optimum value of (1) with $\mathcal{S}_{\text{fixed}} = \emptyset$ is nonnegative (and in fact 0): $\text{size}(\mathcal{V}_h)$ is bounded polynomially[1], so the exact value of $\mathcal{V}_h$ can be computed by binary search [30].

[1]The bound on $\text{size}(\mathcal{V}_h)$ comes from the fact that $\mathcal{V}_h$ is the optimum solution value of a linear program with $|P|$ variables and coefficients with bit complexity $\max_{S \subseteq P} \text{size}(S)$.

Step 1: Let $\mathcal{S}_{\text{fixed}} \leftarrow \emptyset$ and $V \leftarrow \mathcal{V}$ (for $\mathcal{N}$) or $V \leftarrow \mathcal{V}_h$ (for $\mathcal{N}_h$).

Step 2: Compute a pair of optimum primal and dual solutions $((\xi, y), z)$ to (1), where $\text{span}(\mathcal{S}_{\text{fixed}})$ denotes the linear subspace of $\mathbb{R}^P$ generated by the incidence vectors of coalitions in $\mathcal{S}_{\text{fixed}}$.

$$\begin{array}{lrcll} \max & \xi & & & \\ \text{s.t.} & y(S) & = & c(S) - \xi^*_S & \forall S \in \mathcal{S}_{\text{fixed}} \\ & y(S) + \xi & \leq & c(S) & \forall S \in 2^P \setminus \text{span}(\mathcal{S}_{\text{fixed}}) \\ & y(P) & = & V & \\ & \xi & \in & \mathbb{R} & \\ & y & \in & \mathbb{R}^P & \end{array} \tag{1}$$

Step 3: For each $S \in 2^P \setminus \text{span}(\mathcal{S}_{\text{fixed}})$ with $z_S \neq 0$, $\mathcal{S}_{\text{fixed}} \leftarrow \mathcal{S}_{\text{fixed}} \cup \{S\}$, $\xi^*_S \leftarrow \xi$.

Step 4: Repeat Steps 2 and 3 until $\text{span}(\mathcal{S}_{\text{fixed}}) = \mathbb{R}^P$. Return y.

Figure 3: MPS Scheme

If the constraints $y(S) + \xi \leq c(S)$ for $S \in 2^P \setminus \text{span}(\mathcal{S}_{\text{fixed}})$ can be separated efficiently, then one can also solve (1) and its dual [20], and therefore compute the happy nucleolus. For this reason, we focus on the separation problem for this family of constraints in the remainder of this section.

3.2 The Subspace-Avoiding Prize-Collecting Coalition Problem

The separation problem over the non-trivial constraints of (1) can be reformulated as follows for nonnegative cost allocations. Note that for monotone cost functions, both the nucleolus and the happy nucleolus are always nonnegative.

Problem 10 (Subspace-Avoiding Prize-Collecting Coalition)**.** Given a cooperative game (P, c), prizes $\pi \in \mathbb{R}^P_{\geq 0}$ and a linear subspace $L \subsetneq \mathbb{R}^P$, find a coalition $S \in 2^P \setminus L$ and a cost estimate $\lambda \geq c(S)$ with $\lambda + \pi(P \setminus S)$ minimum.

We will reduce the Subspace-Avoiding Prize-Collecting Coalition Problem to a simple prize-collecting problem which essentially corresponds to the special case $L = \{0\}$. Our reduction will add an arbitrarily small additive error.

Problem 11 (Prize-Collecting Coalition)**.** Given a cooperative game (P, c) and prizes $\pi \in \mathbb{R}^P_{\geq 0}$, find a coalition $S \in 2^P$ and a cost estimate $\lambda \geq c(S)$ with $\lambda + \pi(P \setminus S)$ minimum.

Lemma 12. *Let $\mathcal{F}$ be a family of monotone cooperative games and $\alpha \geq 1$ such that there is an α-approximation algorithm for the Prize-Collecting Coalition Problem for $\mathcal{F}$. Then, given $(P, c) \in \mathcal{F}$, prizes $\pi \in \mathbb{R}^P_{\geq 0}$, and disjoint subsets $A, B \subseteq P$, we can find an α-approximate solution to the Prize-Collecting Coalition Problem restricted to coalitions S with $A \subseteq S \subseteq P \setminus B$ in polynomial time.*

Proof. Let (P, c, π, A, B) be an instance of the restricted Prize-Collecting Coalition Problem with $(P, c) \in \mathcal{F}$. For $p \in P$, we define

$$\widehat{\pi}_p \coloneqq \begin{cases} \alpha \cdot U + 1 & \text{if } p \in A \ , \\ 0 & \text{if } p \in B \ , \\ \pi_p & \text{otherwise} \ , \end{cases}$$

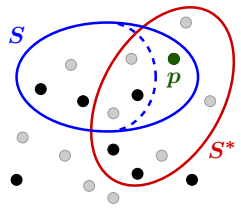
where $U \geq 0$ is an upper bound[2] on $c(P)$. Let (S', λ') be an α-approximate solution of the Prize-Collecting Coalition Problem $(P, c, \widehat{\pi})$. Then, $A \subseteq S'$ (otherwise S' cannot be an α-approximate solution by definition of $\widehat{\pi}$), so by monotonicity of c, (S'', λ') with $S'' \coloneqq S' \setminus B$ is a feasible solution to the restricted problem.

Let S^* be an optimum solution to the restricted problem for π. Then,

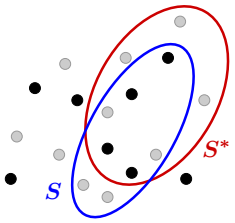
$$\begin{aligned} \lambda' \ + \ \pi\left(P \setminus S''\right) \ &= \ \lambda' \ + \ \widehat{\pi}\left(P \setminus S''\right) \ + \ \pi(B) \\ &\leq \ \alpha \cdot \left(c(S^*) \ + \ \widehat{\pi}\left(P \setminus S^*\right)\right) \ + \ \pi(B) \\ &\leq \ \alpha \cdot \left(c(S^*) \ + \ \widehat{\pi}\left(P \setminus S^*\right) \ + \ \pi(B)\right) \\ &= \ \alpha \cdot \left(c(S^*) \ + \ \pi\left(P \setminus S^*\right)\right) \ , \end{aligned}$$

so (S'', λ') is an α-approximate solution to the restricted problem for π. □

Theorem 13. *Let $\mathcal{F}$ be a family of monotone cooperative games and $\alpha \geq 1$ such that there exists an α-approximation algorithm for the Prize-Collecting Coalition Problem for $\mathcal{F}$. Then, there exists an $(\alpha+\varepsilon)$-approximation algorithm for the Subspace-Avoiding Prize-Collecting Coalition Problem for $\mathcal{F}$ for any fixed $\epsilon > 0$.*



(a) $p \in (S^* \cap P')$ satisfies (2). S and $S \setminus \{p\}$ are cheap and at least one of them is feasible.



(b) No $p \in (S^* \cap P')$ satisfies (2). S is cheap and feasible since it agrees with S^* on P'.

Figure 4: An illustration of the proof of Theorem 13. Players in P' are drawn as black points, all others gray. S^* is an optimum solution to the subspace-avoiding problem. The solution(s) found by our algorithm are shown in blue.

Proof. Let (P, c, π, L) be an instance of the Subspace-Avoiding Prize-Col-lecting Coalition Problem with $(P, c) \in \mathcal{F}$. Let $P' \coloneqq \{p \in P : \{p\} \notin L\}$ and $K \coloneqq \left\lceil \frac{1}{\varepsilon} \right\rceil$. Note that $P' \neq \emptyset$. Let $(S^*, c(S^*))$ be an optimum solution. We proceed differently depending on whether there exists a player $p \in (S^* \cap P')$ with low prize

$$\pi_p \ \leq \ \varepsilon \cdot \pi(P \setminus S^*) \ . \tag{2}$$

If not, we want to work with a certain 'out-set' $O \subseteq (P' \setminus S^*)$ of bounded size $|O| \leq K$. Even though we have to try all possibilities for p and O (because we do not know S^*), the procedure takes only polynomial time because K is constant. Fig. 4 illustrates the following two cases.

[2]The existence of such a bound with polynomial size follows from the technical assumption that size$(c(P))$ is bounded. Note that we do not need to know the bound on size$(c(P))$ a-priori. Instead, we can simply start with an arbitrary value for U and, if this was too small, double U and restart the procedure.

(a) $p \in (S^* \cap P')$ satisfies (2).
Let (S, λ) be an α-approximate solution to the Prize-Collecting Coalition Problem restricted to coalitions containing p. This can be computed in polynomial time by Lemma 12. By $\{p\} \notin L$, we have $S \notin L$ or $(S \setminus \{p\}) \notin L$. Additionally, both solutions are sufficiently cheap, so we can simply return whichever is feasible:

$$\begin{aligned} \pi(P \setminus S) \ + \ \lambda \ &\leq \quad \alpha \quad \cdot \quad \big(\pi(P \setminus S^*) \ + \ c(S^*)\big) \ , \\ \pi(P \setminus (S \setminus \{p\})) \ + \ \lambda \ &= \ \pi_p \ + \ \pi(P \setminus S) \ + \ \lambda \\ &\leq \ (\alpha + \varepsilon) \cdot \ (\pi(P \setminus S^*) \ + \ c(S^*)) \ , \end{aligned}$$

where we used that S^* is a feasible solution to the restricted problem, λ remains valid for $S \setminus \{p\}$ by monotonicity of (P, c), and p satisfies (2).

(b) No $p \in (S^* \cap P')$ satisfies (2).
We assume to have found $O \subseteq (P' \setminus S^*)$ of size $|O| = \min\{K, |P' \setminus S^*|\}$ that maximizes the prize $\pi(O)$, and define the 'in-set'

$$I \ := \ \begin{cases} \big\{p \in (P' \setminus O) : \pi_p > \min_{q \in O} \pi_q\big\} & \text{if } \ |O| = K \ , \\ P' \setminus O & \text{if } \ |O| < K \ . \end{cases}$$

We claim that $S^* \cap P' = I$. This is trivial for $|O| < K$ by definition of O and I. Otherwise, $I \subseteq (S^* \cap P')$ because O maximizes $\pi(O)$ inside $P' \setminus S^*$. Additionally, $(S^* \cap P') \subseteq I$, because for every $p \in (S^* \cap P')$, (2) does not hold, and thus

$$\pi_p \ > \ \varepsilon \cdot \pi\left(P \setminus S^*\right) \ \geq \ \frac{1}{K} \cdot \pi\left(P' \setminus S^*\right) \ \geq \ \frac{1}{K} \cdot \pi(O) \ \geq \ \min_{q \in O} \pi_q \ .$$

Let (S, λ) be an α-approximate solution to the Prize-Collecting Coalition Problem restricted to coalitions S with $S \cap P' = S^* \cap P' = I$, again using Lemma 12. Then, $S^* \notin L$ implies $S \notin L$, so S is feasible. Additionally, $\pi(P \setminus S) + \lambda \leq \alpha \cdot (\pi(P \setminus S^*) + c(S^*))$ because S^* is a feasible solution to the restricted problem. □

With Theorem 13, we can apply known approximation algorithms for prize-collecting problems to their subspace-avoiding variants, achieving essentially the same approximation ratios:

Corollary 14 (based on [7])**.** *There is a 1.599-approximation algorithm for the Symmetric Subspace-Avoiding Prize-Collecting TSP.* □

Corollary 15 (based on [1])**.** *There is a 1.7995-approximation algorithm for the Subspace-Avoiding Prize-Collecting Steiner Tree Problem.* □

4 Heuristic Computation of the Happy Nucleolus for Vehicle Routing Games

Because of its practical relevance for postal services and certain other service providers, we will discuss heuristic computation of the happy nucleolus on vehicle routing games in this section, combining our insights from Section 3 with [6] and [2] and further speed-ups.

Definition 16. *Vehicle routing games* are defined over a complete graph $G = (P \cup \{d\}, E, w)$ with nonnegative edge weights w and certain constraints defining coalitions $T \subseteq P$ that can be served together and will be called *tours* in the following. For example, there could be a fixed

maximum number of players per tour, or a weight constraint. The cost $c(S)$ of a coalition $S \subseteq P$ is defined as the minimum total edge weight of a set of walks in G, where each walk starts and ends at the *depot* vertex d and is otherwise restricted to some tour $T \subseteq P$, and each member of S is visited by at least one walk.

In terms of their cost function, vehicle routing games are equivalent to set covering games with all tours and their cost as input sets. Thus, the happy nucleolus of vehicle routing games is fully determined by the excess values of all tours by [6]. However, vehicle routing games can usually be encoded in size polynomial in $|P|$, so the complexity results of [6] cannot be transferred. Of course, for *fixed capacity* vehicle routing games, where $T \subseteq P$ is a tour if and only if $|T| \leq k$ for some given constant k, the happy nucleolus can be computed in polynomial time as the number of tours is polynomial in $|P|$. But even in this easy case, exact computation would be too slow for practical usage.

In the following, we will work with general vehicle routing games, where happy nucleolus computation is NP-hard by reduction from TSP. Section 4.1 describes our heuristic, and Section 4.2 demonstrates practical results on randomly generated instances.

4.1 Our Heuristic

In order to heuristically compute the happy nucleolus in practice, we use a constraint generation approach. We maintain a cost allocation $y \in \mathbb{R}^P$ and a set $\mathcal{T} \subseteq 2^P$ of coalitions that might be relevant for determining the happy nucleolus, motivated by the fact that there always exists a set of at most $2(|P|-1)$ coalitions that fully determines the happy nucleolus[3] [32]. Our heuristic is outlined in Fig. 5, and details for steps 2) and 3) are given below.

1) Initialization: Start with $\mathcal{T} \leftarrow \emptyset$, $\mathcal{L} \leftarrow \emptyset$ and $y_p \leftarrow \mathrm{c}(\{\mathrm{p}\})$.

2) Tour generation: Add a diverse set of coalitions to $\mathcal{T}$ that have small excess with respect to y, and avoid the linear subspaces in $\mathcal{L}$ (by heuristically separating over (1)).

3) Computing a cost allocation: Run the MPS scheme with a modified LP and restricted to $\mathcal{T}$, save the result in y, and let $\mathcal{L}$ be the set of linear subspaces $\mathrm{span}(\mathcal{S}_{\text{fixed}})$ of all iterations.

4) Repeat: Alternate steps 2) and 3). Occasionally, prune $\mathcal{T}$ by deleting coalitions that have not been relevant for some time. Stop after a fixed number of repetitions.

Figure 5: Sketched happy nucleolus heuristic for vehicle routing games

4.1.1 Tour Generation:

Given a cost allocation y, we want to find tours T that minimize the excess $c(T) - y(T)$ while avoiding the linear subspaces $L \in \mathcal{L}$ that appeared when computing y with a variant of the MPS scheme. We cannot hope to do this optimally for general L on larger instances, as this would imply identification of violated constraints of (1) (if there are any), which in general is at least as hard as prize-collecting TSP. Instead, we use a simple heuristic.

[3]For vehicle routing games, such a set is of course NP-hard to identify.

Our procedure is very similar to Case (a) of the proof of Theorem 13. The only differences are that we greedily compute minimum-excess tours instead of using an α-approximation, and that we consider player insertions next to removals. We start by computing a set of tours that covers all players, greedily minimizing the excess of each tour, and add this set to $\mathcal{T}$. Then, we consider the set of tours resulting from removing/inserting a single player from/into a tour in $\mathcal{T}$. For each subspace $L \in \mathcal{L}$, we select a minimum-excess tour that avoids L from this candidate set, and add it to $\mathcal{T}$.

4.1.2 Computing a Cost Allocation:

Given a set of tours $\mathcal{T}$ and a total value V, a cost allocation that lexicographically maximizes the excess on $\mathcal{T}$ can be computed in polynomial time by the MPS scheme. However, this includes solving up to $|P|$ linear programs which is challenging to do in reasonable time on large instances, even if $\mathcal{T}$ is comparatively small (in our case, $|\mathcal{T}| \in \mathcal{O}(|P|)$). For this reason, we modify the linear program (1) to (3) with sufficiently large γ (where $\gamma = 2$ already produces good results in practice):

$$\begin{aligned} \max\ & \gamma \cdot y(P) + \xi \\ \text{s.t.}\quad & y(S) \leq c(S) - \xi_S^* && \forall S \in \mathcal{S}_{\text{fixed}} \\ & y(S) + \xi \leq c(S) && \forall S \in \mathcal{T} \setminus \mathcal{S}_{\text{fixed}} \\ & \xi \geq 0 \\ & y \in \mathbb{R}^P_{\geq 0} \end{aligned} \tag{3}$$

Conveniently, (3) is a *packing LP*: All variables and coefficients are nonnegative, and all constraints are upper bounds. Such linear programs can be approximately solved very efficiently:

Theorem 17 (Allen-Zhu and Orecchia [2])**.** *There exists a randomized algorithm which, given a packing LP and $\varepsilon > 0$, computes with constant probability a $(1+\varepsilon)$-approximate solution in time $\tilde{\mathcal{O}}\left(\frac{N}{\varepsilon}\right)$, where N is the number of nonzero entries of the constraint matrix.* □

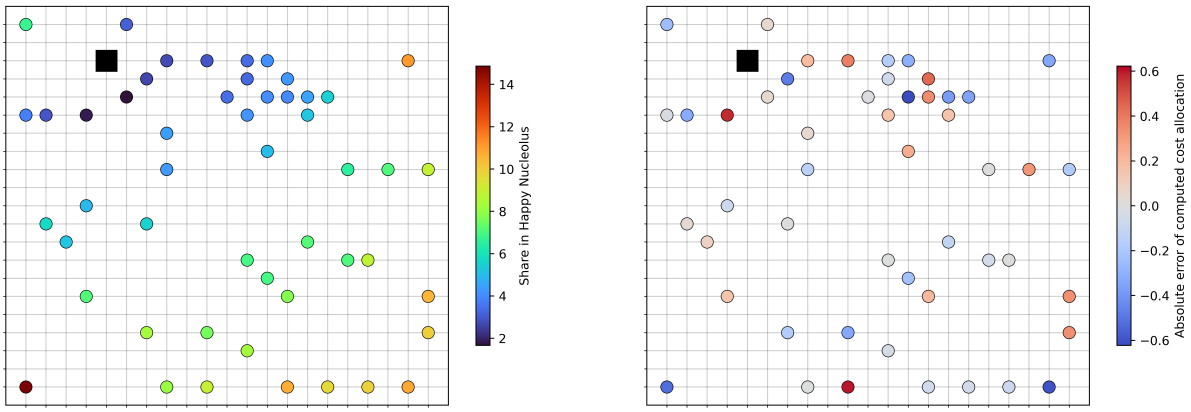
Since we modify the LP (1) to (3) and only compute approximate solutions, we run a simple post-optimization routine after the MPS scheme, where we modify at most two entries of y at a time, and allow augmenting $\mathcal{T}$ by exchanging a single player by another one for tours $T \in \mathcal{T}$.

4.2 Practical Results

We evaluate our heuristic on randomly generated vehicle routing instances with Euclidean distances. Each tour may contain at most a certain number of players. We consider instances with 50 players and a tour capacity of 5 and instances with 1000 players and a tour capacity of 50. The heuristic has also been applied successfully to real-world instances in a more complicated model, including time-dependent travel times, time windows and heterogeneous vehicle fleets. For all tests, the procedure in Fig. 5 was run for 12 iterations with the parameter $\epsilon = 0.2$ for Theorem 17, and 8 parallel threads. The post-optimization described at the end of the previous section is only active during the last 6 iterations.

One example is shown in Fig. 6. Fig. 6b shows the absolute error of the cost allocation computed by the heuristic in 2.2 seconds, with an average relative error of 4.4 %. The exact happy nucleolus (Fig. 6a) was computed in 413 seconds by enumerating all tours. The distribution of relative errors across all players of 100 instances of the same size is shown in Fig. 7.

Turning to larger instances with 1000 players, our heuristic took roughly 2.5 hours per instance, while computing the exact happy nucleolus is intractable with current methods. Therefore, we cannot compute the error of the heuristic on those instances. However, Fig. 8 shows that the



(a) The exact happy nucleolus of the instance.

(b) The difference between the heuristically computed cost allocation and the happy nucleolus.

Figure 6: Results for a randomly generated Euclidean vehicle routing instance with depot (black square), 50 players (circles) and a tour capacity of 5.

cost allocation seems to converge to the final result across the constraint generation iterations and does not change too much in the last few iterations. While the final result could in theory be far away from the actual happy nucleolus, one would expect the cost allocation to be less stable in that case. We observed this kind of convergence across all our tests.

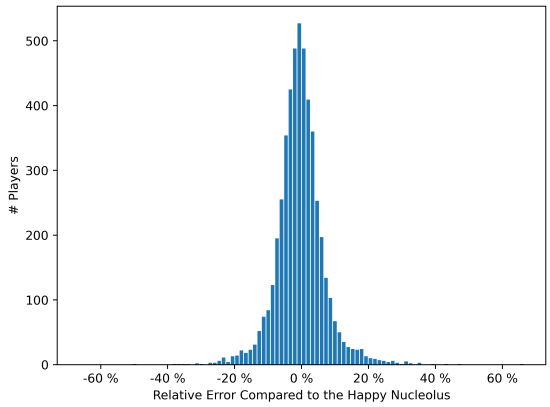


Figure 7: Histogram of the relative errors across 100 vehicle routing instances, each with 50 players and a tour capacity of 5.

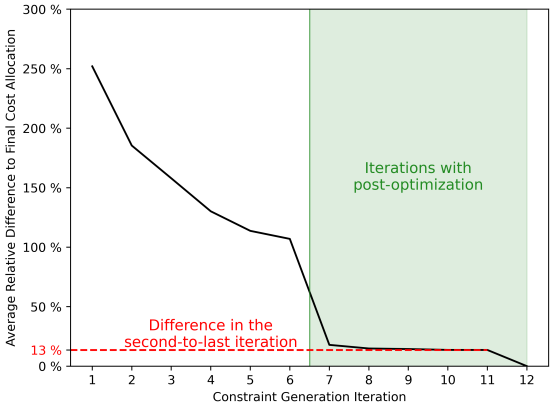


Figure 8: Relative difference of the cost allocation in each iteration compared to the final iteration for an instance with 1000 players and a tour capacity of 50.

5 Future Research Directions

We have discussed aspects of the relation between nucleolus and happy nucleolus in Section 2, and disproved general player-wise domination, even for monotone and subadditive games. As a natural next step, we consider it interesting to study a (multiplicative or additive) bound on the player-wise difference between the two concepts, which cannot exist (instance-independent) for general games, but might exist for monotone and subadditive games.

The discussion in Section 3.1 might suggest that nucleolus computation is at least as hard as happy nucleolus computation. This is true when restricting to the MPS scheme: If we are able to solve (1), we can compute the happy nucleolus, but not necessarily the nucleolus. Beyond the MPS scheme, we find a different situation. For one, there are monotone games for which computing $\mathcal{V}_h$ is NP-hard, whereas $\mathcal{V}$ is trivially given (e.g., for *weighted voting games* by [4]). Even more, there are games for which computing the nucleolus takes polynomial time, but happy nucleolus computation is NP-hard (e.g., for *graph games* by [14]). It would be interesting to see such an example for a subadditive game, or prove the structural result that nucleolus computation is at least as hard as happy nucleolus computation for the class of monotone and subadditive games.

Further, we are interested in approximability of the happy nucleolus. For certain classes of games, we have seen how to approximately solve the separation problem of the MPS scheme for happy nucleolus computation in Section 3. This does not trivially lead to approximation of the happy nucleolus yet, as errors might accumulate. Is constant-factor approximation even possible? For vehicle routing games, the trivial choice of singleton costs does not yield any guarantee.

Acknowledgements

We want to thank Jens Vygen for providing the initial draft of Fig. 2.

References

[1] Ali Ahmadi, Iman Gholami, Mohammad Taghi Hajiaghayi, Peyman Jabbarzade, and Mohammad Mahdavi. "Prize-Collecting Steiner Tree: A 1.79 Approximation". In: *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*. 2024, pp. 1641–1652.

[2] Zeyuan Allen-Zhu and Lorenzo Orecchia. "Nearly Linear-Time Packing and Covering LP Solvers". In: *Mathematical Programming* 175 (2019), pp. 307–353.

[3] Haris Aziz, Felix Brandt, and Paul Harrenstein. "Monotone Cooperative Games and their Threshold Versions". In: *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*. 2010, pp. 1017–1024.

[4] Yoram Bachrach et al. "The Cost of Stability in Coalitional Games". In: *International Symposium on Algorithmic Game Theory*. Springer. 2009, pp. 122–134.

[5] Péter Biró, Dömötör Pálvölgyi, Walter Kern, and Daniel Paulusma. "Generalized Matching Games for International Kidney Exchange". In: *18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*. 2019, pp. 413–421.

[6] Jannis Blauth, Antonia Ellerbrock, Vera Traub, and Jens Vygen. "Cost Allocation for Set Covering: The Happy Nucleolus". In: *Operations Research Letters* 57.107158 (2024).

[7] Jannis Blauth, Nathan Klein, and Martin Nägele. "A Better-Than-1.6-Approximation for Prize-Collecting TSP". In: *Mathematical Programming* (2025), pp. 1–23.

[8] Alberto Caprara and Adam N Letchford. "New Techniques for Cost Sharing in Combinatorial Optimization Games". In: *Mathematical Programming* 124.1 (2010), pp. 93–118.

[9] Georgios Chalkiadakis, Gianluigi Greco, and Evangelos Markakis. "Characteristic Function Games with Restricted Agent Interactions: Core-Stability and Coalition Structures". In: *Artificial Intelligence* 232 (2016), pp. 76–113.

[10] Ning Chen, Pinyan Lu, and Hongyang Zhang. "Computing the Nucleolus of Matching, Cover and Clique Games". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 26. 1. 2012, pp. 1319–1325.

[11] Xiaotie Deng. "Combinatorial Optimization and Coalition Games". In: *Handbook of Combinatorial Optimization: Volume 1–3*. Ed. by Ding-Zhu Du and Panos M. Pardalos. Springer US, 1998, pp. 823–849.

[12] Xiaotie Deng, Qizhi Fang, and Xiaoxun Sun. "Finding Nucleolus of Flow Game". In: *Journal of Combinatorial Optimization* 18.1 (2009), pp. 64–86.

[13] Xiaotie Deng, Toshihide Ibaraki, and Hiroshi Nagamochi. "Algorithmic Aspects of the Core of Combinatorial Optimization Games". In: *Mathematics of Operations Research* 24.3 (1999), pp. 751–766.

[14] Xiaotie Deng and Christos H Papadimitriou. "On the Complexity of Cooperative Solution Concepts". In: *Mathematics of Operations Research* 19.2 (1994), pp. 257–266.

[15] Stefan Engevall, Maud Göthe-Lundgren, and Peter Värbrand. "The Heterogeneous Vehicle-Routing Game". In: *Transportation Science* 38.1 (2004), pp. 71–85.

[16] Ulrich Faigle, Walter Kern, and Jeroen Kuipers. "Note: Computing the Nucleolus of Min-Cost Spanning Tree Games is NP-Hard." In: *International Journal of Game Theory* 27.3 (1998).

[17] Maud Göthe-Lundgren, Kurt Jörnsten, and Peter Värbrand. "On the Nucleolus of the Basic Vehicle Routing Game". In: *Mathematical Programming* 72.1 (1996), pp. 83–100.

[18] Daniel Granot and Gur Huberman. "Minimum Cost Spanning Tree Games". In: *Mathematical Programming* 21.1 (1981), pp. 1–18.

[19] Gianluigi Greco, Enrico Malizia, Luigi Palopoli, and Francesco Scarcello. "The Complexity of the Nucleolus in Compact Games". In: *ACM Transactions on Computation Theory (TOCT)* 7.1 (2015), pp. 1–52.

[20] Martin Grötschel, László Lovász, and Alexander Schrijver. "The ellipsoid method and its consequences in combinatorial optimization". In: *Combinatorica* 1.2 (1981), pp. 169–197.
[21] Mario Guajardo and Mikael Rönnqvist. "A Review on Cost Allocation Methods in Collaborative Transportation". In: *International Transactions in Operational Research* 23.3 (2016), pp. 371–392.
[22] Jochen Könemann, Kanstantsin Pashkovich, and Justin Toth. "Computing the Nucleolus of Weighted Cooperative Matching Games in Polynomial Time". In: *Mathematical Programming* 183.1 (2020), pp. 555–581.
[23] Jochen Könemann and Justin Toth. "A General Framework for Computing the Nucleolus via Dynamic Programming". In: *International Symposium on Algorithmic Game Theory*. Springer. 2020, pp. 307–321.
[24] Alexander Kopelowitz. *Computation of the Kernels of Simple Games and the Nucleolus of n-Person Games*. Tech. rep. Dept. of Mathematics, Hebrew University of Jerusalem, 1967.
[25] Zhixin Liu. "Complexity of Core Allocation for the Bin Packing Game". In: *Operations Research Letters* 37.4 (2009), pp. 225–229.
[26] Michael Maschler, Bezalel Peleg, and Lloyd S Shapley. "Geometric Properties of the Kernel, Nucleolus, and Related Solution Concepts". In: *Mathematics of Operations Research* 4.4 (1979), pp. 303–338.
[27] Reshef Meir, Jeffrey S Rosenschein, and Enrico Malizia. "Subsidies, Stability, and Restricted Cooperation in Coalitional Games". In: *IJCAI*. Vol. 11. 2011, pp. 301–306.
[28] Tri-Dung Nguyen and Lyn Thomas. "Finding the Nucleoli of Large Cooperative Games". In: *European Journal of Operational Research* 248.3 (2016), pp. 1078–1092.
[29] Guillermo Owen. "On the Core of Linear Production Games". In: *Mathematical Programming* 9.1 (1975), pp. 358–370.
[30] Christos H Papadimitriou. "Efficient Search for Rationals". In: *Information Processing Letters* 8.1 (1979), pp. 1–4.
[31] Kanstantsin Pashkovich. "Computing the Nucleolus of Weighted Voting Games in Pseudo-Polynomial Time". In: *Mathematical Programming* 195.1 (2022), pp. 1123–1133.
[32] Hans Reijnierse and Jos A M Potters. "The B-Nucleolus of TU-Games". In: *Games and Economic Behavior* 24.1-2 (1998), pp. 77–96.
[33] Ezra Resnick, Yoram Bachrach, Reshef Meir, and Jeffrey S Rosenschein. "The Cost of Stability in Network Flow Games". In: *International Symposium on Mathematical Foundations of Computer Science*. Springer. 2009, pp. 636–650.
[34] David Schmeidler. "The Nucleolus of a Characteristic Function Game". In: *SIAM Journal on Applied Mathematics* 17.6 (1969), pp. 1163–1170.
[35] Frederik Schulte, Eduardo Lalla-Ruiz, Silvia Schwarze, Rosa G González-Ramírez, and Stefan Voß. *Scalable Core and Shapley Value Allocation Methods for Collaborative Transportation*. Tech. rep. University of Hamburg, 2019.
[36] Lloyd S Shapley. "A Value for n-Person Games". In: *Contributions to the Theory of Games II, Annals of Mathematical Studies* 28 (1953).
[37] Tamás Solymosi. "On Computing the Nucleolus of Cooperative Games". PhD thesis. University of Illinois at Chicago, 1993.
[38] Balázs Sziklai, Tamás Fleiner, and Tamás Solymosi. "On the Core and Nucleolus of Directed Acyclic Graph Games". In: *Mathematical Programming* 163.1-2 (2017), pp. 243–271.
[39] S Tijs. "Bounds for the Core of a Game and the τ-Value". In: *Game Theory and Mathematical Economics*. North-Holland Publishing Company, 1981, pp. 123–132.