Artificial Intelligence 222 (2015) 49–66

Contents lists available at ScienceDirect

Artificial Intelligence

www.elsevier.com/locate/artint

Finding core for coalition structure utilizing dual solution [☆]

Atsushi Iwasaki [a,*], Suguru Ueda [c], Naoyuki Hashimoto [b], Makoto Yokoo [b]

[a] *Graduate School of Information Systems, University of Electro-Communications, Tokyo, Japan*
[b] *Graduate School of Information Science and Electrical Engineering, Kyushu University, Fukuoka, Japan*
[c] *National Institute of Informatics, Tokyo, Japan*

ARTICLE INFO

Article history:
Received 21 December 2013
Received in revised form 13 January 2015
Accepted 19 January 2015
Available online 24 January 2015

Keywords:
Game theory
Cooperative games
Core
Coalition structure

ABSTRACT

When forming the grand coalition is not possible or optimal, agents need to create a coalition structure. The idea of the core can be extended to such a case. In this paper, we propose an innovative exact algorithm called *CoreD* to check core-non-emptiness for coalition structures. A more straightforward exact algorithm based on existing techniques, which we call *CoreP*, first obtains the value of optimal coalition structure by solving an integer programming problem. Then, it checks whether that value can be divided without making a blocking (dissatisfied) coalition. In contrast, *CoreD* first finds a minimal value of the optimal coalition structure so that there exists no blocking coalition. Next, it checks whether the optimal value equals the minimal value We empirically show that when the core is empty, *CoreD* is by far superior to *CoreP*. Also, to find a second-best payoff vector when the core is empty, we propose a new solution concept called the weak ε-core$^+$, which can utilize the approximate value of the optimal coalition structure. Based on the idea of *CoreD*, we further develop an algorithm for checking the non-emptiness of the weak ε-core$^+$.

1. Introduction

Coalition formation is an important capability in automated negotiation among self-interested agents. As a result, coalitional game theory has attracted much attention from AI and multi-agent systems (MAS) researchers [7,12]. In a traditional model of coalitional game theory, it is assumed that all coalitions are possible and that the characteristic function is super-additive; when two coalitions are merged, the merged coalition can obtain at least the sum of the values of the two original coalitions. However, organizing a large coalition can be costly, e.g., such coordination overhead as communication costs. If time is limited, the agents may not have time to carry out the communications and the computations required to coordinate effectively within the composite coalition, and so component coalitions may be more advantageous.

Furthermore, in many real-world applications, there can be inherent constraints on possible coalitions. For example, in many countries, antitrust laws prohibit the formation of certain coalitions of companies (cartels) that can dominate an entire market. Constraints may be placed on coalition sizes to permit or prohibit particular sizes. There can be some underlying graphical structure that determines the possible communication patterns among agents. Therefore, it is natural to assume

[☆] This paper is an extended version of a conference paper that appeared as [17].
[*] Corresponding author.
E-mail addresses: iwasaki@is.uec.ac.jp (A. Iwasaki), s-ueda@nii.ac.jp (S. Ueda), hashimoto@agent.inf.kyushu-u.ac.jp (N. Hashimoto), yokoo@inf.kyushu-u.ac.jp (M. Yokoo).

http://dx.doi.org/10.1016/j.artint.2015.01.001
0004-3702/© 2015 Elsevier B.V. All rights reserved.

that making a coalition is possible only when its members can communicate with each other. There exist several works that have considered such constraints on possible coalitions [41,11,29,35,31,21].

When the grand coalition, i.e., the coalition of all agents, is not possible or optimal, the agents should be divided into smaller coalitions; agents need to create a coalition structure to maximize the reward they can obtain [37]. Furthermore, to make a coalition structure stable, agents need to agree how to divide among themselves the reward obtained by the coalition structure. The core [13], which is a prominent solution concept in the traditional model of coalitional game theory, can be extended to such a case when the grand coalition is not possible or optimal and agents form a coalition structure [2].

For instance, consider a coalitional form game with three agents: $A = \{a, b, c\}$. The characteristic function v represents a mapping from a coalition S (a subset of agents) to the worth or the value earned by the coalition: $v(\{a\}) = v(\{b\}) = 0$, $v(\{c\}) = 3$, $v(\{a, b\}) = 12$, $v(\{b, c\}) = v(\{c, a\}) = 8$, and $v(\{a, b, c\}) = 15$. A payoff vector $y = (y_a, y_b, y_c)$ that belongs to the core can be computed by linear programming in such a way that for every coalition S, $v(S)$ does not exceed the total payoff of the agents in S and such that the total payoff equals $v(A)$. In this example, the core payoff vector is $y = (p, 12 - p, 3)$, where $p \in [5, 7]$. Let us turn to a case where the grand coalition is prohibited in the above example. The concept of the core can be extended to this situation. The agents can create coalition structure $CS = \{\{a, b\}, \{c\}\}$ and obtain 15 reward and share it efficiently. The coalition structure $\{\{a, b\}, \{c\}\}$ is optimal in the sense that the total payoff $V(CS)$ is maximized. The payoff vector in the core for the coalition structure is the same as in the first example.

Checking whether the core is non-empty or not in itself is done in polynomial time in the number of allowed coalitions because it is formulated as a linear programming problem. However, Conitzer and Sandholm [8] pointed out that it requires many constraints for all the subcoalitions and the size of the representation (input) is exponential in the number of agents. Any algorithm for computing the core payoff vector requires time exponential as long as it reads all the input. If a coalition is prohibited or if its value is not explicitly specified, the algorithm may need to compute every value of such coalitions. Computing a value of a coalition is not necessarily straightforward because the agents must solve a complex collaborative planning problem.

In general, as we noted below, computing an optimal coalition structure is known to be NP-hard and checking whether there exists a core for the (optimal) coalition structure or not is NP-complete unless its value is explicitly provided. Thus, our research goal is to develop an exact algorithm whose average runtime is much faster than traditional methods, although the worst-case complexity is doomed to be exponential in the number of explicitly given coalitions. Based on existing techniques, we can construct an algorithm to check core-non-emptiness for coalition structures, which we call *CoreP*. In this algorithm, the value of optimal coalition structure $V(CS^*)$ is obtained by solving an integer programming (IP) problem [25]. The algorithm checks whether $V(CS^*)$ can be divided without making a blocking (dissatisfied) coalition by solving a linear programming (LP) problem [8].

In this paper, we propose an exact algorithm called *CoreD*, which utilizes the dual problem of the linear relaxation of the above IP problem. Experimental evaluations show that *CoreD* is by far superior to *CoreP* when the core is empty. To find a second-best payoff vector when the core is empty, we introduce a new approximate solution concept called weak ε-core$^+$ for the weak ε-core for the optimal coalition structure. The weak ε-core is defined for a particular coalition structure that may or may not be not optimal, relaxes only the non-blocking condition with parameter ε, and efficiently distributes the rewards that the coalition structure earns. On the other hand, the weak ε-core$^+$ does *not* specify a particular coalition structure beforehand. It then relaxes the efficiency condition of the weak ε-core (for the optimal coalition structure), in addition to the non-blocking condition, and efficiently distributes the reward that the dual solution provides. Thus, the sum of the elements in the payoff vector, i.e., the sum of the rewards distributed to agents, can be less than the value of the optimal coalition structure, but the difference must be at most $\varepsilon \cdot n$, where n is the number of agents. Based on the idea of *CoreD*, we also develop an algorithm for checking the non-emptiness of the weak ε-core$^+$ called *ECore*$^+(\varepsilon)$.

This paper is organized as follows. Section 2 briefly describes the basic terms and notations and Section 3 introduces an existing technique for checking core-non-emptiness, proposes our proposed technique, and derives an associated theorem. Based on the our technique, Section 4 develops a novel solution concept, which we call weak ε-core$^+$. Section 5 empirically examines our proposed algorithm from some criteria. Section 6 describes four issues to our contributions, and Section 7 concludes this paper.

1.1. Related works

This subsection briefly explores related works. In traditional models of coalitional game theory, it is assumed that all coalitions are possible and that the characteristic function is super-additive. Forming the grand coalition is guaranteed to be optimal and the main research topic is how to divide the gain of the grand coalition among agents. The traditional theory of coalitional games provides a number of solution concepts, such as the core [13], the Shapley value [39], and the nucleolus [38].

More recently, AI and MAS researchers have been considering the case where forming the grand coalition is not possible or not optimal. In such cases, agents should form a coalition structure to maximize the reward they can obtain. This problem is called the coalition structure generation (CSG) problem and has been an active research topic in AI and MAS.

Sandholm et al. [37] show that the worst-case complexity of CSG problems is $O(n^n)$ in general. Rahwan et al. [30] develop a state-of-the-art algorithm based on dynamic programming, whose complexity is $O(3^n)$. Furthermore, in several class of games [6,3,15,42,5], the complexity of CSG problems has been examined and several tractable classes have been identified. For instance, Bachrach et al. [6] identify *coalitional skill games* where the complexity is polynomial for the games with a constant number of skills and a constant length of tree width. Many faster and scalable algorithms have also been designed [30,34,25,22,32,33].

CSG is deeply related to the winner determination (WD) problem that has been scrutinized in the literature of combinatorial auctions, see Part III [9] for an extensive survey. In combinatorial auctions, agents (bidders) are interested in buying a bundle of objects (goods) that may have some level of complementarities, i.e., an agent wants a set of objects simultaneously and submit higher bid to it than to a single object. The WD problem finds the allocation (combination of bids) that maximizes the sum of the value of the objects. Both the CSG and WD problems are essentially the *set packing* problem, which divides distinct objects into several non-overlapping subgroups such that the sum of the payoffs/values of the objects in each subgroup is maximized (see [23] for example). Since the seminal paper by Sandholm [36] reveals that the complexity of the WD problem is NP-hard in standard assumptions, it has been analyzed in a variety of settings of combinatorial auctions. Several tractable classes of problems are identified and some concise bidding languages (representations) are proposed that practically mitigate the complexity hurdle in the presence of the complementarities. In this context, the main idea of this paper is similar to Nisan [24].

In most of the existing literature on CSG, there exists no restriction on possible coalitions, that is, all coalitions are allowed. There exist several works that have considered constraints on possible coalitions. From the perspective of game theory Demange [11] characterizes the core on games where arbitrary coalitions are prohibited. From the perspective of computer science, Shehory and Klaus [41] and Rahwan and Jennings [29] examine a situation where the size of possible coalitions is limited. Ramchurn et al. [35] consider spatial and temporal constraints for the coalition formation problem of emergency responders and robots. Meir et al. [21] restrict a coalition where agents are not connected on a graph. Rahwan et al. [31] propose a very general framework for constrained coalition formation and develop an algorithm for the CSG problems. This paper considers a special case called locally constrained games in their framework.

To make a coalition structure stable, agents need to agree how to divide among themselves the reward obtained by the coalition structure. Aumann and Dreze [2] extend the core [13] when agents form a coalition structure. Assuming the number of possible coalitions is a constant, and the value of the grand coalition (or the value of an optimal coalition structure) is known, checking core-non-emptiness is easy, since the problem is reduced to a linear programming problem (see e.g., Osborne and Rubinstein [27]). However, when the value of the grand coalition (or the value of an optimal coalition structure) is not explicitly given, the problem is NP-complete[1] even if we assume the number of possible coalitions is a constant [8].

Since the core can be empty, in traditional models of coalitional game theory, several relaxed solution concepts have been proposed, e.g., strong or weak ε-core [40]. Based on the idea of the weak ε-core, we develop a new solution concept called weak ε-core$^+$, in which the efficiency condition is also relaxed by parameter ε. This idea is similar to the *cost of stability*, which is the minimal amount of money provided by an outside source to make the coalition structure stable [4]. We will discuss this point in Section 6.3.

Another line of popular research topic in AI and MAS is compact representation schemes for a characteristic function [8,16]. By naively representing a characteristic function as a table, we need to describe 2^n entries. A compact representation scheme tries to reduce the representation size of a characteristic function. In particular, a representation scheme called Synergy Coalition Group (SCG) [8] is closely related to the constrained coalition formation we consider. We will discuss the connection with SCG in Section 6.1.

2. Model

We model a coalitional form game with constraints on possible coalitions according to *locally constrained* coalition form games [31]. Let A be a set of agents, where $|A| = n$. We assume a characteristic function game, where the value of coalition $S \subseteq A$ is given by characteristic function $v : 2^A \to \mathbb{R}$. Without loss of generality, we assume $\forall S \subseteq A$, $v(S) \geq 0$ and $v(\emptyset) = 0$. Assume a set of allowed coalitions is given as $AC \subseteq 2^A$. If coalition S is not in AC, we assume the value of $v(S)$ is $-\infty$ (or if all values are positive, setting $v(S)$ to zero works as well).

One commonly used assumption about a characteristic function is *super-additivity*. We say characteristic function v is super-additive if for any two disjoint coalitions S_1 and S_2, $v(S_1 \cup S_2) \geq v(S_1) + v(S_2)$ holds. However, we assume v is not necessarily super-additive. Even if we assume the original v is super-additive, considering AC, it is no longer super-additive.

Although Rahwan et al. [31] provide more elaborated compact representations, we use this simple representation of the locally constrained game since we require that all possible allowed coalitions, i.e., AC, are explicitly represented in order to develop an efficient algorithm for checking core-non-emptiness. A similar model is used [8] to compactly represent a standard super-additive characteristic function. We assume the number of these allowed coalitions is relatively small.

Definition 1 *(Coalition structure).* Coalition structure CS^S of agents S is a partition of S into disjoint and exhaustive coalitions, where $CS^S = \{S_1, S_2, \ldots\}$ satisfies the following conditions: $\forall i, S_i \in AC, \forall i, j\ (i \neq j),\ \ S_i \cap S_j = \emptyset$, and $\bigcup_{S_i \in CS^S} S_i = S$.

[1] Greco et al. [14] show that checking core-non-emptiness is co-NP-complete in a different model, in which obtaining the value of the grand coalition can be done in polynomial time but the number of possible coalitions is not a constant.

We also assume every singleton coalition is allowed so that at least one CS^S, i.e., the singleton coalition structure, exists. Otherwise, no coalition structure CS^S may exist. The value of coalition structure CS^S, denoted as $V(CS^S)$, is given by: $V(CS^S) = \sum_{S_i \in CS^S} v(S_i)$. The optimal coalition structure CS^* is a coalition structure of all agents A satisfying the following condition: $\forall CS^A, V(CS^*) \geq V(CS^A)$.

Example 1. Consider a coalitional game with four agents $A = \{a, b, c, d\}$. Assume that all possible coalitions are allowed. The characteristic function is given by

$$\begin{array}{llllll} v(\{a\}) = 3, & v(\{b\}) = 3, & v(\{c\}) = 2, & v(\{d\}) = 2, & & \\ v(\{a,b\}) = 6, & v(\{a,c\}) = 5, & v(\{a,d\}) = 5, & v(\{b,c\}) = 5, & v(\{b,d\}) = 5, & v(\{c,d\}) = 2, \\ v(\{a,b,c\}) = 8, & v(\{a,b,d\}) = 8, & v(\{a,c,d\}) = 5, & v(\{b,c,d\}) = 5, & v(\{a,b,c,d\}) = 5. & \end{array}$$

In this case, the optimal coalition structures CS^* are $\{\{a, b, c\}, \{d\}\}$, $\{\{a, b, d\}, \{c\}\}$, and so on. The payoff $V(CS^*)$ is 10.

When agents create a coalition structure, we need to consider its stability. The concept of the core can be extended to cases where agents create an optimal coalition structure [2]. Let $y = (y_1, y_2, \ldots, y_n)$, where $y_i \geq 0$, be a payoff vector; each y_i represents the reward given to agent i. We say payoff vector y is feasible if $\sum_{i \in A} y_i \leq V(CS^A)$ holds. We also say payoff vector y is efficient if $\sum_{i \in A} y_i = V(CS^A)$ holds. If payoff vector y is efficient, then y is feasible, but not vice versa.

The core [13] is a prominent solution concept in coalitional game theory. Aumann and Dreze [2] extend the concept for considering coalition structures. In this paper, we extend their definition for our coalitional form game.

Definition 2 *(Core for coalition structure).* The core for coalition structure CS^A of all agents A is the set of payoff vectors, where each element (denoted as y) satisfies the following two conditions:

- $\forall S \in AC, \sum_{i \in S} y_i \geq v(S)$ (non-blocking condition),
- $\sum_{i \in A} y_i = V(CS^A)$ (efficiency condition).

The core is non-empty only for the optimal coalition structure CS^* for standard coalitional games [2], i.e., AC contains all possible coalitions. Also, from the definition, there should be no monetary transfers (side payments) across different coalitions, even if such transfers are not prohibited. Thus, we concentrate on finding the core for optimal coalition structure CS^*.

Example 2. Let us consider the same example in Example 1. The core for the optimal coalition structure is non-empty. For instance, a payoff vector $y = (3, 3, 2, 2)$ is in the core.

Definition 2 requires that an efficient payoff vector should not be blocked by any coalition in AC. However, one might think that a set of agents S, which is not in AC, creates its own coalition structure CS^S. Such S can be a *blocking coalition structure* if $V(CS^S)$ exceeds the sum of their payoffs. Nevertheless, Conitzer and Sandholm [8] show that this is not possible.[2]

Proposition 1 *(Lemma 2 in [8]). For payoff vector y, if for all $S \in AC$, $\sum_{i \in S} y_i \geq v(S)$ holds, then for all $S' \subseteq A$, where $S' \notin AC$, $\sum_{i \in S'} y_i \geq V(CS^{S'})$ also holds, where $CS^{S'}$ is any coalition structure of S'.*

3. Core-non-emptiness

3.1. Traditional method

We are going to present *CoreP*, which is based on existing techniques [8,25]. First, it finds the value of optimal coalition structure $V(CS^*)$ by solving the following IP problem. Ohta et al. [25] showed that this simple IP-based method is competitive against the state-of-the-art coalition structure generation techniques. Also, when we deal with locally constrained games, we cannot directly apply the elaborated techniques [31], since their algorithm utilizes their own complicated representation structure.

[2] To be more precise, they do not deal with coalition formation. However, as described in Section 6.1, by a transformation technique called super-additive cover, we can directly apply their results to the problem setting in this paper.

Definition 3 *(IP-PRIMAL: IP for finding the CS* value).*

$$\begin{aligned} \max \quad & \sum_{S_j \in AC} x_j \cdot v(S_j), \\ \text{subject to} \quad & \sum_{j|i \in S_j \in AC} x_j \leq 1, \quad \forall i, \\ & x_j \in \{0,1\}, \quad \forall j. \end{aligned}$$

Here, x_j is a decision variable for coalition S_j where $j \in [1, |AC|]$. If S_j is included in CS^*, x_j is 1, otherwise, x_j is 0. Strictly speaking, there is a chance that the obtained coalition structure does not contain all of the agents. However, since we assume the value of a coalition is non-negative, if agent i is not included in the obtained coalition structure, we can add a coalition $\{i\}$ without reducing the value of the coalition structure. When the size of AC is large, we cannot find an optimal solution of the above IP in a reasonable amount of time even with the state-of-the-art optimization package, e.g., CPLEX.

Second, *CoreP* gives a payoff vector in the core by the following LP problem [8].

Definition 4 *(LP-CNE: LP for checking core-non-emptiness).*

$$\begin{aligned} \text{find} \quad & y, \\ \text{subject to} \quad & \sum_{i \in A} y_i = V\left(CS^*\right), \\ & \sum_{i \in S_j} y_i \geq v(S_j), \quad \forall S_j \in AC, \\ & y_i \geq 0, \quad \forall i. \end{aligned}$$

In summary, *CoreP*, which returns one payoff vector in the core if it is non-empty, is described as follows.

Algorithm *CoreP*

1. Solve IP-PRIMAL to obtain the value of optimal coalition structure $V(CS^*)$.
2. Check whether there exists payoff vector y that satisfies the efficiency and non-blocking conditions by solving LP-CNE with $V(CS^*)$.
3. If such a y exists, return it as an element of the core. Otherwise, report that the core is empty.

3.2. Our method

Let us consider the linear relaxation of IP-PRIMAL and its dual problem.

Definition 5 *(LP-PRIMAL: LP relaxation of IP-PRIMAL).*

$$\begin{aligned} \max \quad & \sum_{S_j \in AC} x_j \cdot v(S_j), \\ \text{subject to} \quad & \sum_{j|i \in S_j \in AC} x_j \leq 1, \quad \forall i, \\ & 0 \leq x_j, \quad \forall j. \end{aligned}$$

Definition 6 *(LP-DUAL: Dual problem).*

$$\begin{aligned} \min \quad & \sum_{i \in A} y_i, \\ \text{subject to} \quad & \sum_{i \in S_j} y_i \geq v(S_j), \quad \forall S_j \in AC, \\ & y_i \geq 0, \quad \forall i. \end{aligned}$$

Before constructing our algorithm, let us define one more decision problem called IP-EFF. Our algorithm effectively bridges solution $\sum_i y_i$ denoted as V^* to a decision problem that checks whether the CS^* value $V(CS^*)$ is equal to V^*.

Definition 7 *(IP-EFF: IP for checking the efficiency condition).*

$$\begin{aligned}
&\text{find} && x, \\
&\text{subject to} && \sum_{S_j \in AC} x_j \cdot v(S_j) = V^*, \\
& && \sum_{j | i \in S_j \in AC} x_j \leq 1, \quad \forall i, \\
& && x_j \in \{0, 1\}, \quad \forall j.
\end{aligned}$$

Now, we are ready to construct *CoreD*, our newly developed algorithm, that utilizes LP-DUAL and IP-EFF. Note that a similar idea is applied for finding an approximate solution of the winner determination problem in combinatorial auctions [24].

Algorithm *CoreD*

1. Solve LP-DUAL to obtain dual solution y^*.
2. Check whether y^* satisfies the efficiency condition by solving IP-EFF with $V^* = \sum_{i \in A} y_i^*$.
3. If y^* satisfies it, return y^* as an element of the core. Otherwise, report that the core is empty.

When $V(CS^*)$ is equal to the lower bound V^* obtained from LP-DUAL, there exists an integer solution for LP-PRIMAL, that is, the optimal value of the linear relaxation happens to be identical to the original IP problem. When $V(CS^*)$ is smaller than V^*, the core is empty. Then the difference between V^* and $V(CS^*)$ is called the *cost of stability* [4], which will be discussed in Section 6.3.

Theorem 1. *Algorithm CoreD is correct; if it returns y^*, y^* is in the core, and if it reports that the core is empty, the core is empty.*

Proof. If *CoreD* returns y^*, from Definition 6, y^* satisfies the non-blocking condition for all $S \in AC$. Also, from Definition 7, y^* satisfies the efficiency condition. Thus, y^* is in the core.

If *CoreD* reports that the core is empty, IP-EFF does not have a solution. Let $x^* = (x_1^*, x_2^*, \ldots)$ be the optimal solution of LP-PRIMAL and $y^* = (y_1^*, \ldots, y_n^*)$ be the optimal solution of LP-DUAL. From the strong duality principle, the solutions of these problems must be identical, i.e., $\sum_{S_j \in AC} x_j^* \cdot v(S_j) = \sum_{i \in A} y_i^* = V^*$ holds. Since LP-PRIMAL is a linear relaxation of IP-PRIMAL, its optimal value cannot exceed V^*.

Here, y^* is a payoff vector, which satisfies the non-blocking condition, but it might not satisfy the efficiency condition, and the sum of its elements is minimized. If IP-EFF does not have a solution, then IP-PRIMAL does not have a solution whose value is better than or equal to V^*. Thus, there exists no payoff vector that simultaneously satisfies the efficiency and non-blocking conditions. Thus, the core is empty. □

CoreD first solves LP-DUAL (the *optimization* problem) and then solves IP-EFF (the *decision* problem). On the other hand, *CoreP* first solves IP-PRIMAL (the *optimization* problem) and then solves LP-CNE (the *decision* problem). When the core is non-empty, we expect that the performances of these algorithms are almost equivalent. However, when the core is empty, especially when $V(CS^*)$ is much smaller than V^*, we expect that IP-PRIMAL (which finds the optimal value) becomes much harder than IP-EFF (which simply checks whether $V(CS^*)$ can be equal to V^*). We empirically confirm these expectations in Section 5. Furthermore, for a class of games where the optimal coalition structure can be found in polynomial time [6,3,42,5], we expect that the performance of *CoreP* and *CoreD* would improve as much as IP-PRIMAL and IP-EFF, which become done in polynomial time, save the runtime. Thus, the average difference of the performance does not change so much, though both algorithms improve much.

4. Weak ε-core$^+$

Since the core can be empty, in the traditional model of coalitional game theory, several solution concepts that relax the non-blocking condition are proposed based on the core. Representative solution concepts based on the core are the strong and the weak ε-core [40]. The former minimizes the total excess of each coalition, while the latter minimizes the average one. Although we can extend both concepts for a coalition structure in general,[3] with our model of locally constrained games, we cannot guarantee the property similar to Proposition 1 with respect to the strong ε-core. Accordingly, we concentrate on the weak ε-core to utilize the idea of *CoreD*.

[3] We can, of course, extend other solution concepts as Airiau and Sen [1] extended the kernel [10] for a coalition structure.

Let us remark another important issue before introducing the weak ε-core and proposing our new solution concept. To define a feasible or allowed payoff vector with a coalition structure, we need to consider whether monetary transfers (side payments) across different coalitions are possible or not. We did not need to consider this issue for the core, since, from the non-blocking condition, no monetary transfers among coalitions are possible. However, if we relax the non-blocking condition, allowing monetary transfers among coalitions extends the space of the feasible payoff vectors. We thus concentrate on the case where transfers among coalitions are possible.[4]

This assumption can be considered fairer in some case. Consider the problem of assigning managers to departments and a very talented manager, who is as successful as the managers in any other department. Also, assume she is the only person who can reduce the deficit of one particular department. Then, assigning her to that department will maximize the total profit of the company. In this case, it is clearly unfair to withhold her bonus just because her department shows a deficit. That being said, the case where such transfers are not possible is more computationally challenging. We will revisit this topic in Section 6.2.

Now we are ready to introduce average excess and the weak ε-core.

Definition 8 *(Average excess).* For coalition S and payoff vector y, let $d(y, S)$ be the average excess of coalition S defined as follows:

$$d(y,S) = \frac{v(S) - \sum_{i\in S} y_i}{|S|}.$$

Definition 9 *(Weak ε-core).* The weak ε-core for CS^A is the set of payoff vectors, where each element (denoted as y) satisfies the following two conditions:

- $\forall S \in AC, d(y,S) \le \varepsilon$ (relaxed non-blocking condition),
- $\sum_{i\in A} y_i = V(CS^A)$ (efficiency condition).

From this definition, it is clear that if the weak ε-core is non-empty for one coalition structure CS^A, it is also non-empty for the optimal coalition structure CS^*. Similar to the core, we can show that there will be no blocking coalition structure, and the following theorem holds:

Theorem 2. *If payoff vector y satisfies $d(y,S) \le \varepsilon$ for all $S \in AC$, then $\frac{V(CS^{S'}) - \sum_{i\in S'} y_i}{|S'|} \le \varepsilon$ holds, for all $S' \notin AC$ and its arbitrary coalition structure $CS^{S'}$.*

Proof. From the definition of a coalition structure, $CS^{S'} = \{S_1, \ldots, S_k\}$, where each $S_j \in AC$ is non-overlapping, $\bigcup_{S_i \in CS^{S'}} S_j = S'$, and $V(CS^{S'}) = v(S_1) + v(S_2) + \ldots + v(S_k)$ holds. Also, from the assumption,

$$d(y,S_j) = \frac{v(S_j) - \sum_{i\in S_j} y_i}{|S_j|} \le \varepsilon$$

holds for all S_j. This means $v(S_j) - \sum_{i\in S_j} y_i \le \varepsilon \cdot |S_j|$. Then,

$$\begin{aligned} \frac{V(CS^{S'}) - \sum_{i\in S'} y_i}{|S'|} &= \frac{v(S_1) + \ldots + v(S_k) - \sum_{i\in S_1} y_i - \ldots - \sum_{i\in S_k} y_i}{|S'|} \\ &\le \frac{\varepsilon \cdot |S_1| + \ldots + \varepsilon \cdot |S_k|}{|S'|} = \varepsilon. \qquad \square \end{aligned}$$

Intuitively, this theorem holds in conjunction with Proposition 1 because the worth of any coalition is scaled proportionally to its size by a factor ε.

The weak ε-core for the optimal coalition structure CS^* requires that the payoff vector be efficient. To find an efficient payoff vector, we need to know the exact value of $V(CS^*)$. Since finding its exact value is NP-hard, it is natural to utilize an approximate, semi-optimal solution against the weak ε-core for CS^*. We introduce a new solution concept called the weak ε-core$^+$, which can be obtained by an approximate value of $V(CS^*)$, and define it as follows.

Definition 10 *(Weak ε-core$^+$).* The weak ε-core$^+$ is a set of payoff vectors, where each element (denoted as y) satisfies the following two conditions:

[4] Although this assumption is non-standard, it is examined by Kamien and Zang [18] and Perez-Castrillo [28] in addition to Aumann and Dreze [2].

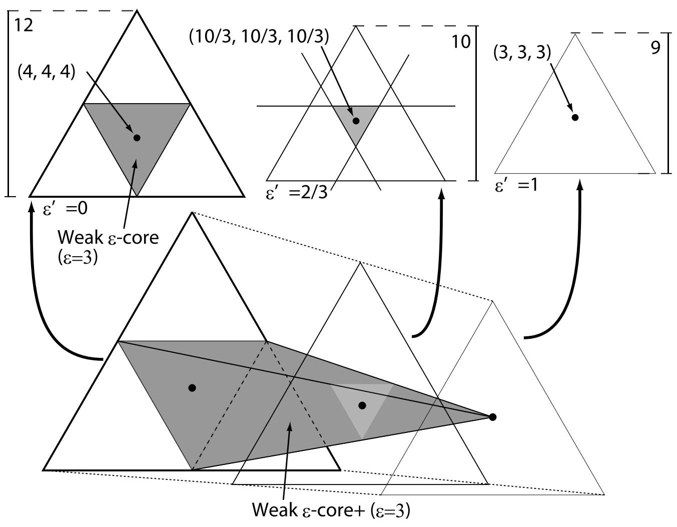

Fig. 1. Example of weak ε-core$^+$ ($\varepsilon = 3$).

- $\forall S \in AC, d(y, S) \le \varepsilon$ (relaxed non-blocking condition),
- $\sum_{i\in A} y_i \le V(CS^*)$ (relaxed efficiency condition).

We relax the efficiency condition so that $\sum_{i\in A} y_i$ can be less than $V(CS^*)$. The condition implies that the weak ε-core$^+$ is not directly linked with a particular coalition structure. The existing solution concepts, along with the model of Aumann and Dreze [2], require the agents to agree with a coalition structure that they form beforehand. However, the weak ε-core$^+$ does not explicitly specify a coalition structure from the fact that computing the optimal coalition structure is NP-hard. Furthermore, this definition indicates that the difference between $V(CS^*)$ and $\sum_{i\in A} y_i$ must be $n\cdot\varepsilon$ at maximum by the relaxed non-blocking condition. We will discuss more when we introduce an algorithm for this approximate solution concept. It is clear that for any $\varepsilon' \le \varepsilon$, the weak ε-core$^+$ is a superset of the weak ε'-core$^+$ and is a superset of the weak ε-core as long as ε is non-negative (the weak ε-core$^+$ becomes empty for $\varepsilon < 0$). Thus, if the weak ε-core is non-empty, the weak ε-core$^+$ is also non-empty. The converse is also true:

Theorem 3. *If the weak ε-core$^+$ is non-empty, there exists a coalition structure CS^A where the weak ε-core for CS^A is non-empty.*

Proof. It is sufficient to show that if the weak ε-core$^+$ is non-empty, the weak ε-core for the optimal coalition structure CS^* is non-empty. Assume the weak ε-core$^+$ is non-empty. Thus, we can choose one element y in it. If $\sum_{i\in A} y_i = V(CS^*)$, then y is also an element of the weak ε-core for CS^*. Otherwise, choose σ as $V(CS^*) - \sum_{i\in A} y_i$. Here, $\sigma > 0$ holds. From y and σ, create another payoff vector y', such that $y'_i = y_i + \sigma/n$. Then, $\sum_{i\in A} y'_i = V(CS^*)$ holds. Also, for each $S \in AC$, the following condition holds.

$$d(y', S) = \frac{v(S) - \sum_{i\in S} y'_i}{|S|} < \frac{v(S) - \sum_{i\in S} y_i}{|S|} \le \varepsilon.$$

Thus, y' is in the weak ε-core for CS^*; it is non-empty. $\square$

Notice that this theorem implies that there may exist a non-optimal coalition structure CS^A with $\sum_{i\in A} y_i \le V(CS^A)$ and the weak ε-core for such a coalition structure CS^A is non-empty.

Example 3. Let there be three agents, a, b, and c, and let

$$AC = \{\{a\}, \{b\}, \{c\}, \{a, b\}, \{b, c\}, \{a, c\}\},$$

where $v(\{a\}) = v(\{b\}) = v(\{c\}) = 0$, $v(\{a, b\}) = v(\{b, c\}) = v(\{a, c\}) = 12$. Optimal coalition structure CS^* is $\{\{a, b\}, \{c\}\}$, $\{\{a\}, \{b, c\}\}$, or $\{\{a, c\}, \{b\}\}$ and value $V(CS^*)$ is 12. The core for the coalition structure of this game is empty because the coalition structure can not sufficiently provide enough value to make the payoff vector stable. At least one agent always has some excess as long as the coalition of the remaining two agents receives enough payoff so that it has no excess.

Fig. 1 illustrates the weak 3-core$^+$ of this game. For the given $\varepsilon = 3$, choose $\varepsilon' \le \varepsilon$ and relax the efficiency condition: $\sum_{i\in A} y_i + n\cdot\varepsilon' = V(CS^*)$. When $\varepsilon' = 0$, the weak 3-core$^+$ becomes identical to the weak 3-core for CS^*. When coalition structure value $V(CS^*) = 12$, the linear programming problem for the weak 3-core($^+$) is given by

$$\begin{aligned}
&\text{find} && y\\
&\text{subject to} && y_{S_j} + 3 \geq 0, \quad \forall S_j \in \big\{\{a\}, \{b\}, \{c\}\big\},\\
& && y_{S_j} + 2 \cdot 3 \geq 0, \quad \forall S_j \in \big\{\{a, b\}, \{b, c\}, \{c, a\}\big\},\\
& && y_a + y_b + y_c = 12.
\end{aligned}$$

By solving this problem, each agent obtains a payoff of at most 6, and the sum does not exceed 12: $y = (6, 6, 0)$, $y = (6, 0, 6)$, or $y = (0, 6, 6)$. Some coalitions with two agents have an average excess of at most 3. Alternatively, $y = (4, 4, 4)$ is in the weak 3-core$^+$ and the weak 3-core for CS^*. The region of the payoff vector is represented by the smaller shaded triangle in the larger triangle in the leftmost of Fig. 1. Here, the height of the larger triangle is equal to $V(CS^*)$. Each point therein represents an efficient payoff vector.

When $\varepsilon' = 2/3$, the coalition structure value is reduced to 10. The weak 3-core$^+$ is equivalent to the weak 3-core with total payoffs of 10. We compute the payoff vector with $\sum_{i \in A} y_i = V(CS^*) - n \cdot \varepsilon' = 10$, which is illustrated by the height of the larger triangle in the middle of Fig. 1. Each agent obtains a payoff of at most 6, and the sum does not exceed 10. Total payoff 10 is exactly shared. The region is represented by the smaller shaded triangle in the larger triangle. $y = (10/3, 10/3, 10/3)$ is in the weak 3-core$^+$. Here if we equally distribute the sacrificed amount of efficiency to the agents, we obtain $(4, 4, 4)$, which is in the weak 3-core.

Finally, let us further relax the efficiency condition up to $\varepsilon' = 1$. Now we sacrifice the efficiency of 3 and the height of the larger triangle in the rightmost of Fig. 1 is 9. In this case, the payoff vector in the weak 3-core$^+$ is uniquely determined as $(3, 3, 3)$.

In summary, the weak 3-core$^+$ is represented as a shaded triangular prism in Fig. 1. The payoff vector belongs to each cross section of the triangular prism, which corresponds to the case with $0 < \varepsilon' < 3$.

To show an algorithm that checks the weak ε-core$^+$-non-emptiness for any given ε, let us introduce two mathematical programming algorithms.

Definition 11 *(Dual problem with the relaxed non-blocking condition).*

$$\begin{aligned}
&\text{min} && \sum_{i \in A} y_i,\\
&\text{subject to} && \sum_{i \in S_j} y_i + \varepsilon \cdot |S_j| \geq v(S_j), \quad \forall S_j \in AC,\\
& && y_i \geq 0, \quad \forall i.
\end{aligned}$$

Definition 12 *(IP for checking the relaxed efficiency condition).*

$$\begin{aligned}
&\text{find} && x,\\
&\text{subject to} && \sum_{S_j \in AC} x_j \cdot v(S_j) \geq V^*,\\
& && \sum_{j | i \in S_j \in AC} x_j \leq 1, \quad \forall i,\\
& && x_j \in \{0, 1\}, \quad \forall j.
\end{aligned}$$

Definitions 11 and 12 are constructed by slightly modifying Definitions 6 and 7. With them, we introduce the following algorithm.

Algorithm *ECore*$^+(\varepsilon)$

1. Solve the dual LP problem in Definition 11 for ε to obtain dual solution y^* and $V^* = \sum_{i \in A} y_i^*$.
2. Check whether y^* satisfies the feasibility condition ($V^* \leq V(CS^*)$) by solving the IP problem in Definition 12 for ε.
3. If y^* satisfies the relaxed efficiency condition, return y^* as an element of the weak ε-core$^+$. Otherwise, report that the weak ε-core$^+$ is empty.

If the algorithm reports a payoff vector y^*, the IP problem in Definition 12 also returns a particular coalition structure CS^A. Notice here that $\sum_{i \in A} y_i^*$ is not always equal to the reward that CS^A provides. $V(CS^A)$ is just guaranteed to be larger than or equal to $\sum_{i \in A} y_i^*$. Thus, the weak ε-core$^+$ does not always choose the payoff vector which is equivalent to the weak ε-core for CS^A. A slight modification can improve the efficiency of the weak ε-core$^+$. More precisely, by uniformly distributing the difference between the resulting coalition structure value $V(CS^A)$ and $\sum_{i \in A} y_i^*$ to the agents, we can obtain the weak ε-core for CS^A.

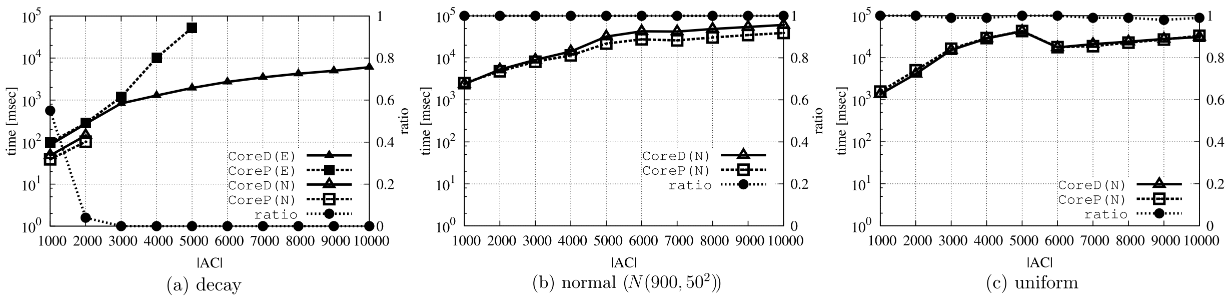

Fig. 2. Runtime for checking core-non-emptiness.

Theorem 4. *Algorithm ECore$^+(\varepsilon)$ is correct; if it returns y^*, y^* is in the weak ε-core$^+$, and if it reports the weak ε-core$^+$ is empty, the weak ε-core$^+$ is empty.*

Proof. The proof is basically similar to that of Theorem 1. The only additional issue is to show that $V(CS^*) - n \cdot \varepsilon \leq \sum_{i \in A} y_i$ holds. If the set of agents A has a unique optimal coalition structure, it implies that $A \in AC$. Thus, this fact is derived from the constraint in Definition 11. Otherwise, it is done from Theorem 2. □

Notice that we can use a simple modification of *CoreP* for checking the non-emptiness of the weak ε-core$^+$. In this case, the algorithm first finds $V(CS^*)$ and then payoff vector y that satisfies the relaxed non-blocking and efficiency conditions. However, regardless of the value of ε, it needs to find $V(CS^*)$, and thus its runtime would become too long when the number of elements in AC becomes large.

5. Evaluation

5.1. Settings

We experimentally evaluate our proposed algorithms. All of the tests were run on a Core i7-4770 processor with 16 GB RAM. The test machine runs Windows 7 Professional SP1 x64 Edition. We used CPLEX 12.6 for solving the linear or integer programming problem instances.

To understand the features of our algorithms, we concentrate on three simple and typical cases that is likely to be observed in practice: (i) AC tends to include small coalitions; (ii) AC tends to include large coalitions; and (iii) there is no bias on the size of each coalition in AC. Unfortunately, there exist no widely accepted standard benchmark instances for locally constrained games. We borrowed the way to generate instances [25]. To generate problem instances, we chose one of three distributions, *decay*, *normal*, and *uniform*, and determine the size of each coalition in AC according to the chosen distribution.[5] The instances made by using the decay distribution capture case (i). The normal distribution corresponds to case (ii), and the uniform distribution corresponds to case (iii). We believe that any of them are quite likely to occur in practical situations. Considering them is useful to deepen understanding of the features of our proposed technique, though we admit that this classification is slightly rough.

Let us further explain how we construct the instances for each case. For case (i), using the decay distribution we create coalitions included in AC. Create a coalition with one randomly chosen agent. Then repeatedly add a new random agent with probability α until an agent is not added or the coalition includes all the agents, where $\alpha = 0.55$. Then we choose value $v(S)$ from uniform distribution $(0, |S| \times 10]$. For case (ii), we fix the mean and the variance of the normal distribution to 900 and 50^2. The size of a coalition $|S|$ is drawn from $N(\mu, \sigma^2)$, then we randomly add agents to a coalition so that the number of agents who belong to each coalition is equals to $|S|$. For case (iii), we use the uniform distribution so that $|S|$ is consistent with the uniform distribution of $[1, n]$. Notice that for any of the distributions, the value of each coalition is drawn from uniform distribution $(0, |S| \times 10]$. We generate 100 problem instances and calculate the geometric average of these problem instances.

5.2. CoreD and CoreP

Fig. 2 compares the runtime of algorithm *CoreD* with *CoreP* for each distribution. Each data point represents the geometric average of 100 problem instances (left y-axis) and also shows the ratio of instances where the core is non-empty (right y-axis). In these experiments, we set #agents = 1000 and vary the number of elements in AC, i.e., $|AC|$ from 1000 to 10 000 (the x-axis).

[5] For convenience, we partly use the Combinatorial Auction Test Suite [19] to create coalitions with an arbitrary distribution.

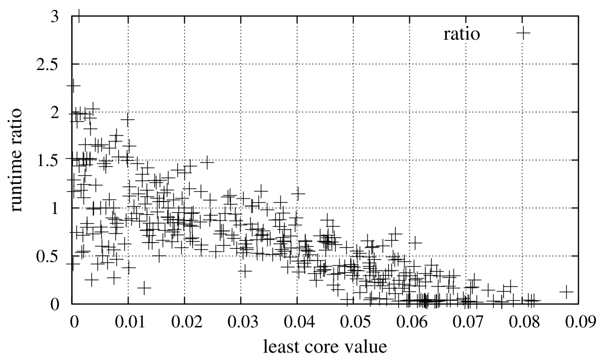

Fig. 3. Runtime ratio for the least core value.

First, in Fig. 2a the ratio decreases rapidly. In fact, when the number of elements in AC exceeds 3000, in the generated instances, the core is always empty. Let us examine the average runtimes of *CoreD* and *CoreP* for problem instances where the core is empty or non-empty for the decay distribution. When the core is empty (denoted as $CoreD(E)$ and $CoreP(E)$), *CoreD* is by far superior to *CoreP*. *CoreD* can check core-non-emptiness within 10^4 msec even when $|AC| = 10\,000$, while the runtime of *CoreP* increases very rapidly as $|AC|$ increases. As a result, *CoreP* cannot check core-non-emptiness within 10^6 msec when $|AC|$ exceeds 5000. On the other hand, when the core is non-empty (denoted as $CoreD(N)$ and $CoreP(N)$), the average runtimes of *CoreD* and *CoreP* are almost equivalent. For randomly generated problem instances, the core is always empty when $|AC| > 3000$. We artificially created problem instances so that the core is non-empty for large $|AC|$, and confirmed that the average runtimes of *CoreD* and *CoreP* are almost equivalent for these problem instances. Notice that we also investigate the normal distributions of $N(100, 50^2)$ and $N(500, 50^2)$ as similar cases to the decay distribution where the size of the coalitions in AC is relatively small. In this case, we confirm almost the same tendency as the decay distribution, except that the core is always empty and these runtimes are basically longer regardless of the number of allowed coalitions.

Next, Figs. 2b and 2c illustrate the results for normal distribution $N(900, 50^2)$ and the uniform distribution, which are substantially different from the first distribution. In the instances, the core is almost non-empty regardless of the number of allowed coalitions for both distributions and these runtimes are much larger than the decay distribution. As we see in the decay distribution when the core is non-empty, the average runtimes of *CoreD* are almost equivalent to those of *CoreP*. When $|AC| > 5000$ for the normal distribution, *CoreD* is slightly worse than *CoreP*. This may be because the LP-DUAL part requires more time as $|AC|$ increases, while IP-EFF remains as hard as IP-PRIMAL. Here we are not able to extract results when the core is empty, because we have very few instances when it is empty.

In summary, when the core is non-empty, the average runtimes of *CoreD* and *CoreP* are almost equivalent for any distribution. When the core is empty, the average runtimes of *CoreD* significantly outperform *CoreP* for the decay distribution. This suggests that when the core is empty, IP-PRIMAL, which computes the optimal solution of $V(CS^*)$ in *CoreP*, is much harder to solve than IP-EFF, which decides whether $V(CS^*)$ is equal to V^* or not in *CoreD*.

So far, we have illustrated the runtimes of *CoreD* and *CoreP* by varying the number of allowed coalitions and the probability distribution over their sizes. Let us next focus on the runtime relation between how “unstable” the game is and how much time is saved when moving from *CoreP* to *CoreD*. There are several ways to measure the stability level of a game instance. We consider the least core value, which is the minimal value of ε such that the (weak) ε-core is non-empty [20]. As we already confirmed, in almost all instances for the normal and uniform distributions, the core is non-empty. We focus on the decay distribution where the core is empty and compute the least core value. We obtained 440 of 100×10 instances such that we could compute the least value within 10^5 msec. If the size of AC exceeds 5000, we obtain no exact solution within the time limit. Otherwise, we could successfully obtain the exact least values except when $|AC|$ is 1000 or 2000, for which we could solve 45/100 and 95/100 instances, respectively. Fig. 3 scatters the runtime ratios of *CoreD* against *CoreP* in the least core value. It is clear that as the least core value is larger, *CoreD* is significantly better. If that value exceeds 0.05, *CoreD* is approximately twice as fast as *CoreP*. In contrast, when the least value is small, the runtime fluctuates within wide ranges, i.e., for some instances *CoreD* still performs well, and vice versa.

5.3. $ECore^+(\varepsilon)$

Next, Fig. 4 investigates the runtime for $ECore^+(\varepsilon)$ by varying ε and examines the ratio of instances where the weak ε-core$^+$ is non-empty. The x-axis indicates the value of ε, and the y-axis on the left and right sides shows the average runtime and the ratio of the instances, respectively. Through this experiment, we set #agents = 1000, $|AC| = 10\,000$ and vary ε from 0.01 to 10. We set the time limit to 10^5 msec; if the runtime of $ECore^+(\varepsilon)$ exceeds this time limit, we terminate the algorithm and exclude this problem instance when calculating the average runtimes and the ratios for this ε. As described below, the algorithm is terminated only when ε is slightly less than 0.1 for the decay distribution.

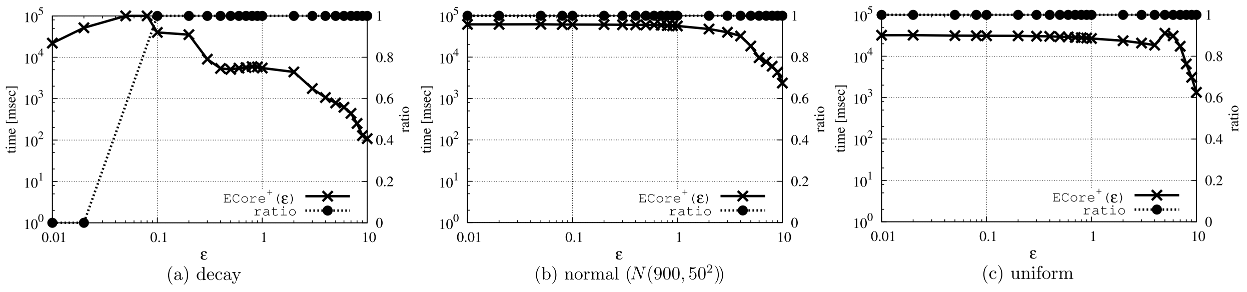

Fig. 4. Runtime for checking non-emptiness of weak ε-core$^+$.

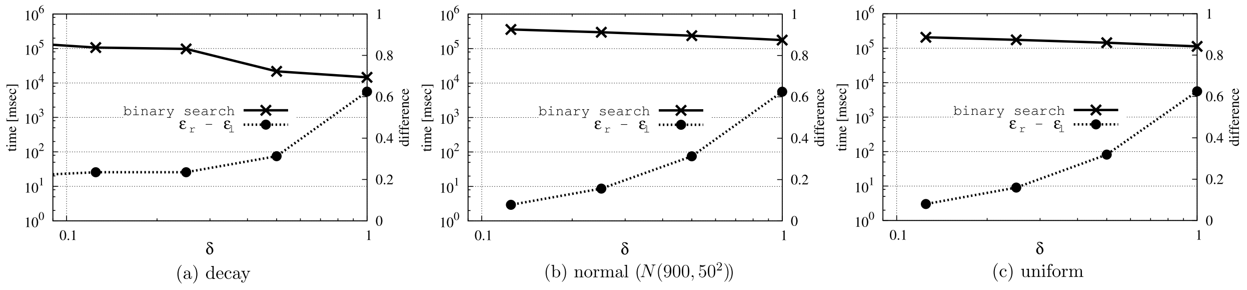

Fig. 5. Runtime for obtaining ε_l and ε_r.

In general, when the core is empty, and if we make ε sufficiently large, the weak ε-core$^+$ will eventually become non-empty. *ECore*$^+(\varepsilon)$ exhibits an easy–hard–easy transition for the runtime by increasing ε for the decay distribution in Fig. 4a and shows that the hardest region lies between 0.05 and 0.1. For the decay distribution, the runtime of *ECore*$^+(\varepsilon)$ exceeds the time limit only when the values of ε are 0.05 (100%) and 0.08 (100%), where each number within the parentheses is the ratio of the instances that exceed the time limit. This is the only such case across this experiment. On the other hand, the simple modification of *CoreP* in the previous section cannot solve any problem instances in this setting, since we cannot obtain $V(CS^*)$ within the time limit of 10^5 msec for any distribution. Let us turn to the normal and uniform distributions depicted in Figs. 4b and 4c. The runtime of *ECore*$^+(\varepsilon)$ hardly changes varying ε up to 1. When ε exceeds 1, the runtime gradually becomes faster and faster. Note that the weak ε-core$^+$ is always non-empty for each ε.

From these results, especially for the decay distribution, we expect that there exist three regions: (i) $0 \le \varepsilon \le \varepsilon_l$, where *ECore*$^+(\varepsilon)$ can efficiently check that the ε-core$^+$ is empty, (ii) $\varepsilon_l < \varepsilon < \varepsilon_r$, where *ECore*$^+(\varepsilon)$ does not terminate within a reasonable amount of time, and (iii) $\varepsilon_r \le \varepsilon$, where *ECore*$^+(\varepsilon)$ can efficiently check that the ε-core$^+$ is non-empty. Ideally, we hope to identify value ε^*, s.t., the weak ε-core$^+$ is non-empty for any $\varepsilon \ge \varepsilon^*$, while for any $\varepsilon < \varepsilon^*$, the weak ε-core$^+$ is empty. However, such ε^* lies in region (ii). As a result, identifying ε^* is too time-consuming.

Alternatively, we can use a standard binary search procedure to identify ε_l and ε_r, when $\varepsilon_l < \varepsilon^* < \varepsilon_r$ holds. More specifically, we run *ECore*$^+(\varepsilon)$ with a predefined time limit by varying ε. We keep the largest (smallest) value of ε where we found that the weak ε-core$^+$ is empty (non-empty) as candidates of ε_l (ε_r). Also, we keep the smallest and largest values of ε (denoted as c_l and c_r), where *ECore*$^+(\varepsilon)$ did not terminate within the predefined time limit. If the difference between c_l and the candidate of ε_l (as well as the difference between c_r and the candidate of ε_r) becomes smaller than δ, we terminate the procedure and return the current candidates of ε_l and ε_r.

Fig. 5 shows the runtime of the binary search procedure and its obtained bound $\varepsilon_r - \varepsilon_l$ by varying precision δ. We set the time limit of each run of *ECore*$^+(\varepsilon)$ to half a minute, i.e., 30 000 msec. The other parameter settings are identical to Fig. 4. For the decay distribution in Fig. 5a, the runtime does not increase very rapidly by decreasing δ. The obtained bound of ε_r–ε_l, when δ is large, becomes smaller by decreasing δ, i.e., by increasing the precision, but eventually ε_r–ε_l does not change by decreasing δ due to the time limit. In contrast, for the normal and uniform distributions, as we observed so far, the runtime and its bound are almost constant in Figs. 5b and 5c.

5.4. *Efficiency*

We have empirically shown that *ECore*$^+(\varepsilon)$ effectively removed the complexity hurdle of exactly knowing $V(CS^*)$. However, it must sacrifice the efficiency (social surplus) against the weak ε-core for the optimal coalition structure up to $n \cdot \varepsilon$ in the worst case. When it is impossible to compute the exact value of the optimal coalition structure, we cannot directly investigate how far from optimal is the sum of payoffs obtained by *ECore*$^+(\varepsilon)$. Nevertheless, we can derive a non-optimal coalition structure, compute its value, and obtain the weak ε-core for it. We investigate how much efficiency is achieved in the sum of payoffs obtained by $ECore^+(\varepsilon)$ and the value of a coalition structure computed in a greedy manner.

Let us briefly describe the greedy algorithm. We sort the allowed coalitions in decreasing order by value, check whether a coalition can be utilized one by one according to that order, and obtain a coalition structure and its worth that may not be optimal. The agents form the coalition structure and share the value efficiently so that every coalition S tolerates the excess up to $|S|\varepsilon$. The greedy algorithm guarantees no lower bound with respect to efficiency, while our algorithm bounds efficiency by the difference between the optimal value and $n \cdot \varepsilon$. We fix the number of agents and the allowed coalitions to 1000 and 10 000. In this setting, exactly knowing $V(CS^*)$ is impossible. When the weak ε-core$^+$ is non-empty, if the greedy algorithm finds a coalition structure whose value exceeds $ECore^+(\varepsilon)$, we share the value of the coalition structure so that the payoff vector is in the weak ε-core for the obtained coalition structure. We observe the ratio of the coalition structure value obtained by the greedy algorithm to the sum of payoffs obtained by $ECore^+(\varepsilon)$ varying with ε.

For the decay distribution, for which the core is always empty as shown in Section 5.2. For sufficiently small ε, the weak ε-core$^+$ chosen by $ECore^+(\varepsilon)$ outperforms the weak ε-core for the coalition structure chosen by the greedy procedure in terms of the sum of payoffs (efficiency). In fact, when $\varepsilon = 0.1$, the average ratio of the greedy algorithm to $ECore^+(\varepsilon)$ is only about 0.87. As ε increases, the efficiency ratio linearly increases and when ε reaches 1.0, it is about 0.95. For 100 instances of each ε, the weak ε-core$^+$ is always non-empty. $ECore^+(\varepsilon)$ significantly outperforms the greedy algorithm. However, it is a case that the coalition structure by the greedy algorithm can earn more than that by $ECore^+(\varepsilon)$ as the number of agents or the value of ε increases. When ε exceeds one, the ratio also exceeds one. The greedy algorithm achieves better efficiency than $ECore^+(\varepsilon)$. However, each agent must face relatively high excess.

For the normal and uniform distributions, for which the core is always non-empty (except very few instances for the uniform distribution). Note that *CoreD* finds a payoff vector so that every agent has no excess. In this case, for any ε, the greedy algorithm outperforms $ECore^+(\varepsilon)$. For the uniform distribution, when $\varepsilon = 0.1$, the ratio is 1.01, the ratio linearly increase, and it is 1.11 for $\varepsilon = 1.0$. The same trend is observed for the normal distribution. This is because the size of some coalition is relatively larger than the decay distribution. the greedy algorithm can choose some large coalitions with the higher value and can find the coalition structure whose value is larger than the value that the linear relaxation problem finds, though the degree is approximately 1.0% for sufficiently small ε.

6. Discussions

6.1. Super-additive cover

Aumann and Dreze [2] presented a conceptual transformation method called super-additive cover, which transforms an original game with a non-super-additive characteristic function, into a new game with a super-additive one. More specifically, for an original characteristic function v and AC, a new super-additive one $\bar{v}$ is defined as follows.

- If $S \in AC$, $\bar{v}(S)$ is given as $v(S)$.
- Otherwise, $\bar{v}(S)$ is given as $\max_{CS^S}\{\sum_{S_i \in CS^S} v(S_i)\}$, where $CS^S = \{S_1, S_2, \ldots\}$ is a partition of S, i.e., all S_i are disjoint and $\bigcup_{S_i \in CS^S} S_i = S$, and $\forall S_i \in AC$.

Based on this conceptual transformation, the optimal coalition structure in the original game CS^* is identical to partition CS^A, such that $\bar{v}(A) = \sum_{S_i \in CS^A} v(S_i)$ holds. Thus, $V(CS^*) = \bar{v}(A)$ holds. In summary, the problem of finding an optimal coalition structure (and the core for the coalition structure) in the original game, is equivalent to that of finding the value of the grand coalition (and the standard core) in the transformed game. Actually, Conitzer and Sandholm [8] dealt with this transformed game, where the characteristic function is explicitly given only for the elements of AC, which they call the *Synergy Coalition Group*. Greco et al. [15] introduce a concept called cohesive cover, in which the transformation is applied only for the grand coalition. The result presented in this subsection holds also for cohesive cover.

6.2. Monetary transfers among coalitions

When monetary transfers among coalitions are not allowed and the core is empty, forming an optimal coalition structure is not necessarily best for minimizing ε.

Example 4. Let there be three agents, a, b, and c, and assume $v(\{a,b,c\}) = v(\{a,b\}) = v(\{b,c\}) = v(\{a,c\}) = 30$, $v(\{c\}) = 3$, and $v(\{a\}) = v(\{b\}) = 0$. Here, the optimal coalition structure is $\{\{a,b\},\{c\}\}$. Assume that transfers across coalitions are not allowed and that agents choose payoff vector $y = (15, 15, 3)$. Coalition $S = \{a,c\}$ will be unsatisfied, since they can obtain 30 if they work together, i.e., $v(S) = 30$, but they obtain only 18. The average excess $d(y,S) = \frac{v(S)-\sum_{i \in S} y_i}{|S|}$ is 6. On the other hand, if the grand coalition is formed (which is not optimal), agents can choose payoff vector $y = (10, 10, 10)$. Then, any average excess is at most 5. Thus, this sub-optimal coalition structure is better for minimizing ε.

When monetary transfers among coalitions are allowed, we can separately solve the following problems (1) finding $V(CS^*)$, and (2) finding y that satisfies $\sum_{i \in A} y_i = V(CS^*)$ and optimizes some criteria, e.g., minimizing an average excess. On the other hand, when monetary transfers among coalitions are not allowed, we need to find a desirable pair of a coalition structure and a payoff vector.

In fact, the following is the mixed integer programming (MIP) formulation:

Definition 13 *(MIP where monetary transfers are not allowed).*

$$
\begin{aligned}
\text{find} \quad & y, \\
\text{subject to} \quad & \sum_{j | i \in S_j \in AC} x_j \le 1, \quad \forall i, \\
& \sum_{i \in S_j} y_i + \varepsilon \cdot |S_j| \ge v(S_j), \quad \forall S_j \in AC, \\
& \sum_{i \in A} y_i = \sum_{S_j \in AC} x_j \cdot v(S_j), \\
& \sum_{i \in S_j} y_i = x_j \cdot v(S_j), \quad \forall S_j \in AC, \\
& x_i \in \{0, 1\}, \quad \forall S_j \in AC, \\
& y_i \ge 0, \quad \forall i.
\end{aligned}
$$

Given ε, this MIP finds a pair of a coalition structure CS^A and a payoff vector y that is in the weak ε-core for CS^A. The notations and constraints here are basically borrows from Definitions 3 and 4. The fourth constraint highlights the assumption that monetary transfers among coalition are not allowed; all the value that a coalition S_j earns must be distributed among its members.

It is clear that the MIP in Definition 13 cannot be separated in the same manner that monetary transfers among coalition are allowed. The fourth constraint connects the first constraint that explores a coalition structure with the second and third ones that find an appropriate payoff vector. Also, the MIP must search every coalition structure, including the non-optimal ones, as Example 4 describes. Therefore, when monetary transfers are not allowed, the idea of *CoreD* is no longer applied. Though developing such an algorithm is admittedly our immediate future work, we need an entirely different technique from the dual solution.

Note a case where a given instance has the core for the optimal coalition structure. For such instances, the payoff vector in the core does not change regardless whether monetary transfers among coalitions are allowed [2]. We do not need to transfer the payoffs among coalition to make the payoff vector stable, since each sub-coalition only has non-positive excess.

6.3. Cost of stability

This subsection discusses the connection between the weak ε-core$^+$ and *cost of stability* (*CoS*), which is the minimal amount of money provided by an outside source to make the coalition structure stable [4]. *CoS* is defined as the difference between the lower bound V^* obtained from LP-DUAL and the value $V(CS^*)$ with an optimal CS. If an outside source, e.g., a government, provides $CoS = V^* - V(CS^*)$ when agents form CS^*, they can divide $V(CS^*) + CoS = V^*$ so that there exists no blocking coalition.

Definition 14 *(Cost of stability).* Fix characteristic function v and the optimal coalition structure CS^* The cost of stability *CoS* is defined as the solution of the following problem:

$$
\begin{aligned}
\min \quad & \Delta, \\
\text{subject to} \quad & \Delta \ge 0, \\
& y_i \ge 0 \quad \forall i \in A, \\
& \sum_{i \in A} y_i = V(CS^*) + \Delta, \\
& \sum_{i \in S} y_i \ge v(S) \quad \forall S \in AC.
\end{aligned}
$$

When the core is non-empty, *CoS* is clearly zero. For the weak ε-core$^+$ and *CoS*, the following theorem holds:

Theorem 5. *If the weak ε-core$^+$ is non-empty, $CoS \le n \cdot \varepsilon$ holds.*

Proof. Assume the weak ε-core$^+$ is non-empty. Then we can choose one element y in the weak ε-core$^+$. Next, consider new payoff vector y' such that $y'_i = y_i + \varepsilon$. From the definition of the weak ε-core$^+$, the following conditions are satisfied.

- $v(S) - \sum_{i \in S} y' \le 0, \forall S \in AC$,
- $V(CS^*) \le \sum_{i \in A} y'_i \le V(CS^*) + n \cdot \varepsilon$.

Both conditions imply that y' is in the core in the game with v' such that $\sum_j \Delta_j = n \cdot \varepsilon$. $CoS \le n \cdot \varepsilon$ holds. □

We can obtain a lower-bound of ε where the weak ε-core$^+$ is non-empty based on *CoS*.

Theorem 6. *The weak ε-core$^+$ is non-empty for $\varepsilon = CoS/n$.*

Proof. From the definition of *CoS*, there exists y such that $\sum_{i \in A} y_i = V(CS^*) + CoS$ and $\forall S \in AC, \sum_{i \in S} y_i \ge v(S)$ hold. That is, y is in the core in the game with v'. Now, consider new payoff vector y' such that $y'_i = y_i - \varepsilon$ where $\varepsilon = CoS/n$. y' is in the weak ε-core$^+$ because $\sum_{i \in A} y'_i = \sum_{i \in A} y_i - n \cdot \varepsilon = V(CS^*)$ and $\forall S \in AC, d(y, S) \le \varepsilon$ hold. Thus, the weak ε-core$^+$ is non-empty. □

From these two theorems, we obtain the following theorem.

Theorem 7. *Choose $\underline{\varepsilon}$ as the least value of ε, where the weak ε-core$^+$ is non-empty. Then, $\underline{\varepsilon}$ is equal to CoS/n.*

6.4. Nucleolus

This subsection discusses the relation between the weak ε-core$^+$ and the nucleolus [38]. Since the nucleolus always exists and is uniquely determined, we focus on the weak least core [20] for the optimal coalition structure and the weak least core$^+$, i.e., the solution concepts with the least value ε with which they are guaranteed to exist.

Lemma 1. *The least value ε of the weak least core for the optimal coalition structure is equivalent to that of the weak least core$^+$.*

Proof. Let $\underline{\varepsilon}_1$ and $\underline{\varepsilon}_2$ be the least values of the weak least core (for the optimal coalition structure) and the weak least core$^+$. First assume $\underline{\varepsilon}_1 < \underline{\varepsilon}_2$. Consider a weak ε'-core$^+$ with $\varepsilon' < \underline{\varepsilon}_2$. Then the weak ε'-core$^+$ must be empty. From the assumption, the weak $\underline{\varepsilon}_1$-core must be empty. However, from its definition, since the weak $\underline{\varepsilon}_1$-core is a superset of the weak $\underline{\varepsilon}_1$-core$^+$, it must be non-empty. This contradicts that the weak $\underline{\varepsilon}_1$-core$^+$ with $\underline{\varepsilon}_1 < \underline{\varepsilon}_2$ is empty.

Next, assume $\underline{\varepsilon}_1 > \underline{\varepsilon}_2$. Consider a weak ε'-core (not -core$^+$) with $\varepsilon' < \underline{\varepsilon}_1$. From the assumption, the weak $\underline{\varepsilon}_2$-core must be empty. However, since the weak $\underline{\varepsilon}_2$-core$^+$ is non-empty, Theorem 3 derives that the weak $\underline{\varepsilon}_2$-core must be non-empty. This is a contradiction. Accordingly, $\underline{\varepsilon}_1 = \underline{\varepsilon}_2$. □

From this lemma, the next theorem is derived straightforwardly.

Theorem 8. *The weak least core$^+$ is identical to the weak least core for the optimal coalition structure CS^*, i.e., any payoff vector in the weak least core$^+$ is also in the weak least core for CS^* and vice versa.*

Proof. From the definitions and Lemma 1, the weak least core$^+$ is a superset of the weak least core (for the optimal coalition structure CS^*). Assume there exists a payoff vector y such that it is not in the weak least core, but in the weak least core$^+$. Since the non-blocking conditions are the same across these two solution concepts, this payoff vector y must violate the efficiency condition of the weak least core. Here, let us construct another payoff vector y' such that

$$y'_i = y_i + \frac{V(CS^*) - y(A)}{n} \quad \text{for all } i.$$

The payoff vector y' inevitably satisfies the efficiency condition because $y'(A) = \sum_{i \in A} y'_i = y(A) + \frac{V(CS^*)-y(A)}{n} \times n = V(CS^*)$.

Let ε^* be the least value of the weak least core. For any coalition S and payoff vector y,

$$\frac{v(S)-y(S)}{|S|}=\frac{v(S)-y'(S)}{|S|}+\frac{V(CS^*)-y(A)}{n}\le\varepsilon^*.$$

Then, we obtain $\frac{v(S)-y'(S)}{|S|}\le\varepsilon^*-\frac{V(CS^*)-y(A)}{n}$. y' is in the weak ε'-core with ε' which is equal to $\varepsilon^*-\frac{V(CS^*)-y(A)}{n}$. This contradicts the fact that ε^* is the least value of the weak ε-core because the weak ε'-core with $\varepsilon'<\varepsilon^*$ is non-empty. Accordingly, the weak least core$^+$ is identical to the weak least core for CS^*. □

This theorem shows that when ε reaches the least value, the weak ε-core$^+$ converges to the weak ε-core for CS^*. That is, the dual solution $\sum_{i\in A}y_i$ converges to $V(CS^*)$. As a result, the agents in the weak ε-core$^+$ must form CS^*, otherwise, the agents do not produce enough value. As such, if CS^* and its value are exactly computed, the weak ε-core$^+$ is no longer needed. However, as long as finding CS^* is hard, the least value is difficult to obtain in practice. We cannot justify how far a given ε from the (unknown) least value. Thus, the weak ε-core$^+$ is useful as an approximate solution concept for the weak ε-core for CS^*.

Next, we briefly explain the traditional nucleolus [38]. For any payoff vector and coalition, let us consider the excess of the coalition. Also, consider the vector of all the coalitions' excesses, sorted in descending order. The nucleolus chooses the payoff vector that lexicographically minimizes this vector; it first minimizes the greatest excess, then the second-greatest excess, etc. The nucleolus has some desirable properties. For a transferable utility game, it always exists, is unique (even if the core of the game is empty), is in the core if the core is non-empty. Furthermore, it is known that if the core is empty, the nucleolus is in the (strong) least core. The same property holds when we use the average excesses, i.e., if the weak core is empty, the weak nucleolus is in the weak least core (see [43] for example).

Before formally defining the weak nucleolus for our games with constraints on the possible coalitions, let us introduce several notations. Let $\theta(y)$ for each payoff vector y be the $|AC|$-vector whose components are the corresponding average excess $d(y,S)$ of each allowed coalition $S\in AC$, sorted in descending order: $\theta_i(y)\ge\theta_j(y)$ whenever $1\le i\le j\le|AC|$. i is associated with coalition S such that $\theta_i(y)=d(y,S)$. The lexicographic order between such excess vectors is defined as follows. We represent $\theta(y')\prec\theta(y)$ if y is preferred to y', which holds if and only if there exists an integer i^* such that

$$\theta_i(y')=\theta_i(y)\quad\text{for all } i<i^*\quad\text{and}\quad\theta_{i^*}(y')>\theta_{i^*}(y).$$

Also, when we write $\theta(y')\preceq\theta(y)$, it indicates that $\theta(y')\succ\theta(y)$ does not hold. Now we are ready to define the weak nucleolus.

Definition 15 *(Weak nucleolus).* The weak nucleolus for coalition structure CS^A of all agents A is in the set of payoff vectors where each element y satisfies

$$\{y\,|\,\theta(y')\preceq\theta(y)\text{ for all }y'\}.$$

This definition implies the efficiency condition by minimizing the largest average excess. If no coalition is constrained, it is of course equivalent to the original weak nucleolus. Furthermore, the weak nucleolus does minimize the largest excess of both the members and the non-members of AC. However, it does not lexicographically minimize the excess vector. Precisely, it lexicographically minimizes the excess vector for the members of AC, not for its non-members.[6] Therefore, this weak nucleolus may provide a different payoff vector from the conventional one if the use of some coalition is prohibited. Nevertheless, when an optimal coalition structure is given, the weak nucleolus is guaranteed to be in the weak least core for that coalition structure.

Theorem 9. *The weak nucleolus for the optimal coalition structure is in the weak least core$^+$, as well as in the weak least core for the optimal coalition structure.*

Proof. Assume there exists a payoff vector y such that it is not in the weak least core (for the optimal coalition structure), but in the weak nucleolus (for the optimal coalition structure). Let y' be a payoff vector from the weak least core. From the assumption, the largest excess $\theta_1(y)$ in the excess vector of y must exceed $\theta_1(y')$. That is, y' is preferred to y. This fact contradicts that y is in the weak nucleolus. □

Finally, let us briefly describe what happens if we relax the efficiency condition of the weak nucleolus. We can define the weak nucleolus$^+$ by imposing $\sum_{i\in A}y_i\le V(CS^*)$. However, this does not distinguish the weak nucleolus from the weak nucleolus$^+$ at all because the latter automatically achieves its efficiency due to its definition which minimizes the largest excess in y.

[6] A similar argument in the context of the synergy coalition group representation was previously discussed in [26].

7. Conclusions

When forming the grand coalition is not possible or optimal, agents need to create a coalition structure and to decide how to divide its gain among themselves. We developed an innovative algorithm called *CoreD* to check the core-non-emptiness for coalition structures. Since checking core-non-emptiness is NP-complete, we set our research goal to develop an algorithm whose average runtime is much faster than a traditional algorithm called *CoreP*. We showed that the performance of our newly developed algorithm called *CoreD* is much better than *CoreP* when the core is empty.

Furthermore, we proposed a new solution concept called the weak ε-core$^+$, which is based on the weak ε-core for the optimal coalition structure CS^*. It can utilize an approximate CS^* value. Based on the idea of *CoreD*, we developed an algorithm for checking the non-emptiness of the weak ε-core$^+$ called $ECore^+(\varepsilon)$.

Acknowledgements

This research was partially supported by KAKENHI 23500184, 24220003, and 26280081. We wish to thank Yukihiko Funaki, Takumi Kongo, Reshef Meir, and Dengji Zhao for their valuable comments. Makoto Kitaki provided excellent research assistance. We also highly appreciate the associated editor and three anonymous reviewers for their fruitful comments.

References

[1] S. Airiau, S. Sen, On the stability of an optimal coalition structure, in: Proceedings of the 19th European Conference on Artificial Intelligence (ECAI), 2010, pp. 203–208.
[2] R.J. Aumann, J.H. Dreze, Cooperative games with coalition structures, Int. J. Game Theory 3 (1974) 217–237.
[3] H. Aziz, B. de Keijzer, Complexity of coalition structure generation, in: Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS), 2011, pp. 191–198.
[4] Y. Bachrach, E. Elkind, R. Meir, D.V. Pasechnik, M. Zuckerman, J. Rothe, J.S. Rosenschein, The cost of stability in coalitional games, in: Proceedings of the 2nd International Symposium on Algorithmic Game Theory (SAGT), 2009, pp. 122–134.
[5] Y. Bachrach, P. Kohli, V. Kolmogorov, M. Zadimoghaddam, Optimal coalition structure generation in cooperative graph games, in: Proceedings of the 26th Conference on Artificial Intelligence (AAAI), 2013, pp. 81–87.
[6] Y. Bachrach, R. Meir, K. Jung, P. Kohli, Coalitional structure generation in skill games, in: Proceedings of the 24th Conference on Artificial Intelligence (AAAI), 2010, pp. 703–708.
[7] G. Chalkiadakis, E. Elkind, M. Wooldridge, Computational Aspects of Cooperative Game Theory, Morgan and Claypool Publishers, 2011.
[8] V. Conitzer, T. Sandholm, Complexity of constructing solutions in the core based on synergies among coalitions, Artif. Intell. 170 (2006) 607–619.
[9] P. Cramton, Y. Shoham, R. Steinberg, Combinatorial Auctions, MIT Press, 2006.
[10] M. Davis, M. Maschler, The kernel of a cooperative game, Nav. Res. Logist. Q. 12 (1965) 223–259.
[11] G. Demange, The strategy structure of some coalition formation games, Games Econ. Behav. 65 (2009) 83–104.
[12] E. Elkind, T. Rahwan, N.R. Jennings, Computational coalition formation, in: Multiagent Systems, 2nd edn., MIT Press, 2013, pp. 329–380.
[13] D. Gillies, Some theorems on n-person games, Ph.D. thesis, Princeton University, 1953.
[14] G. Greco, E. Malizia, L. Palopoli, F. Scarcello, On the complexity of core, kernel, and bargaining set, Artif. Intell. 175 (2011) 1877–1910.
[15] G. Greco, E. Malizia, L. Palopoli, F. Scarcello, On the complexity of the core over coalition structures, in: Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI), 2011, pp. 216–221.
[16] S. Ieong, Y. Shoham, Marginal contribution nets: a compact representation scheme for coalitional games, in: Proceedings of the 6th ACM Conference on Electronic Commerce (ACM EC), 2005, pp. 193–202.
[17] A. Iwasaki, S. Ueda, M. Yokoo, Finding the core for coalition structure utilizing dual solution, in: Proceedings of the 2013 IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT), 2013, pp. 114–121.
[18] M.I. Kamien, I. Zang, The limits of monopolization through acquisition, Q. J. Econ. 105 (1990) 465–499.
[19] K. Leyton-Brown, M. Pearson, Y. Shoham, Towards a universal test suite for combinatorial auction algorithms, in: Proceedings of the 1st ACM Conference on Electronic Commerce (ACM EC), 2000, pp. 66–76.
[20] M. Maschler, B. Peleg, L.S. Shapley, Geometric properties of the kernel, nucleolus, and related solution concepts, Math. Oper. Res. 4 (1979) 303–338.
[21] R. Meir, J.S. Rosenschein, E. Malizia, Subsidies, stability, and restricted cooperation in coalitional games, in: Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI), 2011, pp. 301–306.
[22] T.P. Michalak, J. Sroka, T. Rahwana, M. Wooldridge, P. McBurney, N.R. Jennings, A distributed algorithm for anytime coalition structure generation, in: Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS), 2010, pp. 1007–1014.
[23] T.D. Nguyen, A fast approximation algorithm for solving the complete set packing problem, Eur. J. Oper. Res. 237 (2014) 62–70.
[24] N. Nisan, Bidding and allocation in combinatorial auctions, in: Proceedings of the 1st ACM Conference on Electronic Commerce (ACM EC), 2000, pp. 1–12.
[25] N. Ohta, V. Conitzer, R. Ichimura, Y. Sakurai, A. Iwasaki, M. Yokoo, Coalition structure generation utilizing compact characteristic function representations, in: Proceedings of the 15th International Conference on Principles and Practice of Constraint Programming (CP), 2009, pp. 623–638.
[26] N. Ohta, A. Iwasaki, M. Yokoo, K. Maruono, A compact representation scheme for coalitional games in open anonymous environments, in: Proceedings of the 21st Conference on Artificial Intelligence (AAAI), 2006, pp. 697–702.
[27] M. Osborne, A. Rubinstein, A Course in Game Theory, MIT Press, 1994.
[28] D. Perez-Castrillo, Cooperative outcomes through noncooperative games, Games Econ. Behav. 7 (1994) 428–440.
[29] T. Rahwan, N.R. Jennings, An algorithm for distributing coalitional value calculations among cooperative agents, Artif. Intell. 171 (2007) 535–567.
[30] T. Rahwan, N.R. Jennings, An improved dynamic programming algorithm for coalition structure generation, in: Proceedings of the 7th International Conference on Autonomous Agents and Multiagent Systems (AAMAS), 2008, pp. 1417–1420.
[31] T. Rahwan, T.P. Michalak, E. Elkind, P. Faliszewski, J. Sroka, M. Wooldridge, N.R. Jennings, Constrained coalition formation, in: Proceedings of the 25th AAAI Conference on Artificial Intelligence (AAAI), 2011, pp. 719–725.
[32] T. Rahwan, T.P. Michalak, N.R. Jennings, Minimum search to establish worst-case guarantees in coalition structure generation, in: Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI), 2011, pp. 338–343.
[33] T. Rahwan, T.P. Michalak, N.R. Jennings, A hybrid algorithm for coalition structure generation, in: Proceedings of the 26th AAAI Conference on Artificial Intelligence (AAAI), 2012, pp. 1443–1449.

[34] T. Rahwan, S.D. Ramchurn, N.R. Jennings, A. Giovannucci, An anytime algorithm for optimal coalition structure generation, J. Artif. Intell. Res. 34 (2009) 521–567.
[35] S.D. Ramchurn, M. Polukarov, A. Farinelli, N.C. Truong, N.R. Jennings, Coalition formation with spatial and temporal constraints, in: Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS), 2010, pp. 1181–1188.
[36] T. Sandholm, Algorithm for optimal winner determination in combinatorial auctions, Artif. Intell. 135 (2002) 1–54.
[37] T. Sandholm, K. Larson, M. Andersson, O. Shehory, F. Tohmé, Coalition structure generation with worst case guarantees, Artif. Intell. 111 (1999) 209–238.
[38] D. Schmeidler, The nucleolus of a characteristic function game, SIAM J. Appl. Math. 17 (1969) 1163–1170.
[39] L.S. Shapley, A value for n-person games, in: Contributions to the Theory of Games, Princeton University Press, 1953, pp. 307–317.
[40] L.S. Shapley, M. Shubik, Quasi-cores in a monetary economy with nonconvex preferences, Economitrica 34 (1966) 805–827.
[41] O. Shehory, S. Kraus, Methods for task allocation via agent coalition formation, Artif. Intell. 101 (1998) 165–200.
[42] T. Voice, M. Polukarov, N. Jennings, Coalition structure generation over graphs, J. Artif. Intell. Res. 45 (2012) 165–196.
[43] H.P. Young, N. Okada, T. Hashimoto, Cost allocation in water resources development, Water Resour. Res. 18 (1982) 463–475.