Concise Characteristic Function Representations in Coalitional Games Based on Agent Types

Suguru Ueda, Makoto Kitaki, Atsushi Iwasaki, and Makoto Yokoo

Kyushu University,
744, Motooka, Nishi-ku,
Fukuoka 819-0395, Japan,
{ueda, kitaki}@agent.is.kyushu-u.ac.jp, {iwasaki, yokoo}@is.kyushu-u.ac.jp

Abstract. Forming effective coalitions is a major research challenge in AI and multi-agent systems (MAS). Thus, coalitional games, including Coalition Structure Generation (CSG), have been attracting considerable attention from the AI research community. Traditionally, the input of a coalitional game is a black-box function called a characteristic function. A range of previous studies have found that many problems in coalitional games tend to be computationally intractable when the input is a black-box function. Recently, several concise representation schemes for a characteristic function have been proposed. Although these schemes are effective for reducing the representation size, most problems remain computationally intractable.

In this paper, we develop a new concise representation scheme based on the idea of *agent types*. Intuitively, a type represents a set of agents, which are recognized as having the same contribution. This representation can be exponentially more concise than existing concise representation schemes. Furthermore, this idea can be used in conjunction with existing schemes to further reduce the representation size. Moreover, we show that most of the problems in coalitional games, including CSG, can be solved in polynomial time in the number of agents, assuming the number of possible types is fixed.

1 Introduction

Forming effective coalitions is a major research challenge in AI and multi-agent systems (MAS). A coalition of agents can sometimes accomplish things that individual agents cannot or can do things more efficiently. There are two major research topics in coalitional games. The first topic involves partitioning a set of agents into coalitions so that the sum of the rewards of all coalitions is maximized. This problem is called the Coalition Structure Generation problem (CSG) [9, 10]. The second topic involves how to divide the value of the coalition among agents. The theory of coalitional games provides a number of solution concepts, such as the core, the Shapley value, and the nucleolus.

A range of previous studies have found that many problems in coalitional games, including CSG, tend to be computationally intractable. Traditionally,

Table 1. Computational complexities of coalition formation problems and CSG using conventional representations

	Representation schemes		
	Characteristic function	SCG	MC-nets
Core non-empty	exponential	NP-complete [4]	co-NP-hard [6]
Core membership	exponential	linear[1] [4]	co-NP-complete [6]
The Shapley value	exponential	$O(2^{2n})$[2]	linear[1] [6]
CSG	$O(3^n)$ [9]	NP-hard [8]	NP-hard [8]

Table 2. Computational complexities of coalition formation problems and CSG using type-based representations

	Type-based representation schemes		
	Characteristic function	SCG	MC-nets
Core non-empty	polynomial (Thm. 2)	polynomial (Thm. 7)	polynomial (Thm. 12)
Core membership	$O(n^t)$ (Thm. 3)	$O(n^{2t})$ (Thm. 8)	$O(n^{2t})$ (Thm. 12)
The Shapley value	$O(n^t)$ (Thm. 4)	$O(n^{2t})$ (Thm. 9)	$O(\|R\| \cdot n^{2t})$ (Thm. 13)
CSG	$O(n^{2t})$ (Thm. 5)	$O(n^{2t})$ (Thm. 10)	$O(n^{2t})$ (Thm. 14)

the input of a coalitional game is a black-box function called a characteristic function, which takes a coalition as an input and returns the value of the coalition (or a coalition structure as a whole). Recently, several concise representation schemes for a characteristic function have been proposed, e.g., synergy coalition group (SCG) [4] and marginal contribution nets (MC-nets) [6]. These schemes represent a characteristic function as a set of rules rather than as a single black-box function and can effectively reduce the representation size. However, most problems are still computationally intractable (Table 1).

In this paper, we develop a new concise representation scheme for a characteristic function, which is based on the idea of *agent types*. Intuitively, a type represents a set of agents, which are recognized as having the same contribution. Most of the hardness results in Table 1 are obtained by assuming that all agents are different types. In practice, however, in many MAS application problems, while the number of agents grows, the number of different types of agents remains small. This type-based representation can be exponentially more concise than existing concise representation schemes. Furthermore, this idea can be used in conjunction with existing schemes, i.e., SCG and MC-nets, for further reducing the representation size. We show that most of the problems in coalitional games, including CSG, can be solved in polynomial time in the number of participating agents, assuming the number of possible types t is fixed (Table 2).

[1] These problems can be solved in linear time in the input size.
[2] This bound is not tight. Examining a tight bound is an open problem.

Our idea of using agent types is inspired by the recent innovative work of Shrot *et al.* [11]. They assume that a game is already represented in some concise representation, e.g., SCG. The goal of their work is first to identify agent types and then to efficiently solve problems in coalitional games by utilizing the knowledge of agent types. This approach becomes infeasible when a standard characteristic function representation is used, since there exists no efficient way for identifying agent types.

In contrast to their study, we assume that agent types are explicitly used for describing a characteristic function in the first place. Also, we consider a wider range of problems including CSG. As a result, the overlap between our work and [11] is very small. In Table 2, only two entries, i.e., Core non-empty and the Shapley value for SCG, might be considered as somewhat overlapping, while other topics are not discussed in [11].

Several other works than [11] have also examined the concept of agent types to represent agent capabilities. Bachrach and Rosenschein [2] introduce *coalitional skill games*, where the capability of an agent is characterized by its skills. We can consider such skills correspond to agent types. However, they do not assume that the possible types/skills of an agent are fixed (even if the number of skills is fixed, the combinations of skills are exponential). Thus, their algorithms and complexity results are quite different from ours. Chalkiadakis *et al.* [3] consider another representation scheme that can represent any characteristic function, as well as ours. However, they examine the complexity of coalition formation problems only in *simple games*, where the value of a characteristic function is either 0 or 1, while we consider general games, where the value can be arbitrary determined.

Furthermore, the literature of *weighted voting games* [1, 5] assumes that the possible types of an agent are bounded. In this game, each agents has a weight and the value of a coalition is determined by the sum of their weights. The value of a coalition is 1 if the total weights exceeds a certain quota, and 0 otherwise. Their works are very similar to our works if we regard each agent's weight as the agent type. They showed that the core-related coalition formation problems and computing the Shapley value and the nucleolus become more tractable when the number of weights is bounded. However, as well as [3], they concentrate on only simple games and their type-based representation by using a weight is a subclass of our proposed one where all types are classified by a cardinal utility. Thus, we believe that this paper has significant new contributions since it can handle general characteristic functions.

2 Model

2.1 Coalitional Games and Coalition Structure Generation

Let $A = \{1, 2, \ldots, n\}$ be a set of all agents. The value of a coalition S is given by a characteristic function v. A characteristic function $v : 2^A \to \mathbb{R}$ assigns a value to each set of agents (coalition) $S \subseteq A$. We assume that each coalition's value is non-negative.

Let $x = (x_1, x_2, \ldots, x_n)$ be a payoff vector. A *solution concept* assigns to each coalitional game a set of reasonable payoff vectors. Two of the best-known ones are the core and the Shapley value.

Definition 1. *The core is the set of all payoff vectors x, which satisfy the feasibility condition: $\sum_{i \in A} x_i = v(A)$, and the non-blocking condition: $\forall S \subseteq A, \sum_{i \in S} x_i \geq v(S)$.*

If there exists a blocking coalition S such that $\sum_{i \in S} x_i < v(S)$ holds, then the agents in S have an incentive to collectively deviate from the grand coalition and divide $v(S)$ themselves. In general, the core can be empty or contain a large set of payoff vectors.

Definition 2. *The Shapley value of agent i, ϕ_i, is defined as:*

$$\phi_i = \sum_{S \subseteq A \setminus \{i\}} \frac{|S|!(n - |S| - 1)!}{n!} (v(S \cup \{i\}) - v(S)).$$

One intuitive interpretation of the Shapley value is that it averages an agent's marginal contribution over all possible orders in which the agent may join the coalition.

A coalition structure CS is a partition of A, into disjoint, exhaustive coalitions. More precisely, $CS = \{S_1, S_2, \ldots\}$ satisfies the following conditions:

$$\forall i, j \ (i \neq j), \ S_i \cap S_j = \phi, \quad \bigcup_{S_i \in CS} S_i = A.$$

In other words, in CS, each agent belongs to exactly one coalition, and some agents may be alone in their coalitions.

For example, if there exist three agents a, b, and c, then there are seven possible coalitions: $\{a\}$, $\{b\}$, $\{c\}$, $\{a, b\}$, $\{b, c\}$, $\{a, c\}$, $\{a, b, c\}$, and five possible coalition structures: $\{\{a\}, \{b\}, \{c\}\}$, $\{\{a, b\}, \{c\}\}$, $\{\{a\}, \{b, c\}\}$, $\{\{b\}, \{a, c\}\}$, $\{\{a, b, c\}\}$.

The value of a coalition structure CS, denoted as $V(CS)$, is given by:

$$V(CS) = \sum_{S_i \in CS} v(S_i).$$

An optimal coalition structure CS^* is a coalition structure that satisfies the following condition:

$$\forall CS, V(CS^*) \geq V(CS).$$

We say a characteristic function is super-additive, if for any disjoint sets S_i, S_j, $v(S_i \cup S_j) \geq v(S_i) + v(S_j)$ holds. If the characteristic function is super-additive, solving CSG becomes trivial, i.e., the grand coalition is optimal. In this paper, we assume a characteristic function can be non-super-additive.

Example 1. Let there be four agents a, b, c, and d. The characteristic function is given as follows:

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

In this case, there exist multiple optimal CSs. For example, $\{\{a,b,c\},\{d\}\}$ and $\{\{a,b,d\},\{c\}\}$ are optimal CSs, and the value of these CSs is 10.

2.2 Agent Types

Shrot *et al.* [11] introduced the idea of using *agent types* to reduce the computational complexity of coalition formation problems. If two agents have the same type, their marginal contributions are the same. They introduced two different notions of agent types, i.e., *strategic types* and *representational types*. The former defines types based on the strategic power of the agents, and the latter defines them based on the representation of the game.

Strategic types are defined based on the strategic power (marginal contribution) of each agent, i.e., if two agents are strategically equivalent, they belong to the same (strategic) type.

Definition 3 (Definition 2.1 in [11]). *Agents $i, j \in A$ are* strategically equivalent *if for any coalition S, such that $i, j \notin S : v(S \cup \{i\}) = v(S \cup \{j\})$.*

The notion called *representational type* is introduced to check the equivalence of agents more conveniently based on a concise representation. Agents are representationally equivalent if they only differ in their identifier. If two agents are representationally equivalent, they are also strategically equivalent, but not vice versa.

Shrot *et al.* examined the computational complexity for determining strategic or representational types for several concise representations. They further showed that if the number of types is fixed in these representations, most intractable problems in coalitional games become polynomial.

3 Type-based Characteristic Function Representation

We assume the person who is describing a game has some prior information about the equivalence of agents. Then the person will describe the game by explicitly using the information of the agent types of which he/she is aware. We need another notion of agent types. This is because (i) the information of the person can be partial and he/she is not necessarily aware of all strategic equivalence, and (ii) the equivalence that he/she is aware of is representation-independent. Therefore, we introduce another notion called *recognizable types*.

Definition 4. *Agents $i, j \in A$ are recognizably equivalent if the person who is describing the game (either by a characteristic function or by a concise representation) knows that for any coalition S, such that $i, j \notin S : v(S \cup \{i\}) = v(S \cup \{j\})$.*

From this definition, if two agents are recognizably equivalent, they are also strategically equivalent, but not vice versa. Furthermore, assuming appropriate representation is chosen, if two agents are recognizably equivalent, they are very likely to be representationally equivalent.

Let $T = \{1, 2, \ldots, t\}$ be the set of all recognizable types and n_A^i be the number of agents of type $i \in T$ in the set of all agents A. Also, $n_A = \langle n_A^1, n_A^2, \ldots, n_A^t \rangle$ denotes a vector, where each element represents the number of agents of each type in A.

We represent a characteristic function as follows:

Definition 5. *For a coalition S, the coalition type of S is a vector $n_S = \langle n_S^1, n_S^2, \ldots, n_S^t \rangle$, where each n_S^i is the number of type i agents in S. We denote the set of all possible coalition types as $A^t = \{\langle n^1, n^2, \ldots, n^t \rangle \mid 0 \leq n^i \leq n_A^i\}$. A type-based characteristic function is defined as $v_t : A^t \to \mathbb{R}$.*

From the definition of recognizable equivalence, $\forall S$ and its type n_S, $v(S) = v_t(n_S)$ holds.

Theorem 1. *A type-based characteristic function requires $O(n^t)$ space.*

Proof. It is clear from the fact that $|A^t| = (n_A^1 + 1) \times \ldots \times (n_A^t + 1) < n^t$. □

Example 2. Let agents a, b be type 1 and agents c, d be type 2 in Example 1. A type-based characteristic function representation for Example 1 is given as follows:

$$v_t(\langle 1, 0 \rangle) = 3,\ v_t(\langle 0, 1 \rangle) = 2,\ v_t(\langle 1, 1 \rangle) = 5,$$
$$v_t(\langle 2, 0 \rangle) = 6,\ v_t(\langle 0, 2 \rangle) = 2,\ v_t(\langle 2, 1 \rangle) = 8,$$
$$v_t(\langle 1, 2 \rangle) = 5,\ v_t(\langle 2, 2 \rangle) = 5.$$

For example, the type of coalition $S = \{a, b, c\}$ is $\langle 2, 1 \rangle$ because S contains two agents of type 1 and one agent of type 2. Thus, $v(S) = 8$. Here, the type-based representation defines the value for each of eight possible coalition types, while the standard representation needs to specify the value for each of fifteen possible coalitions. In general, a type-based representation is exponentially more concise than a standard representation.

We say a payoff vector is *symmetric* if all agents with the same type receive an identical amount. We can restrict our attention to *symmetric* payoff vectors without loss of generality (Lemma 3.2 in [11]). A symmetric payoff vector is represented as $\langle x_1, \ldots, x_t \rangle$, where all type i agents receive x_i. We examine the computational complexity of coalition formation problems in this restriction and show that these problems can be solved in polynomial time in the number of agents.

Theorem 2. *If the number of agent types t is fixed, by using a type-based characteristic function, determining whether the core is non-empty can be done in polynomial time in the number of agents n.*

Proof. To check whether the core is non-empty, it is sufficient to confirm whether there exists a symmetric payoff vector that is feasible and not blocked by any coalition. To this end, we construct the following linear programming formula and check whether it has a solution. The variables of the linear program are elements of a symmetric payoff vector $x = \langle x_1, \ldots, x_t \rangle$. The program is as follows:

$$\sum_{1 \le i \le t} n_A^i \cdot x_i = v_t(n_A);\ \forall n_S \in A^t,\ \sum_{1 \le i \le t} n_S^i \cdot x_i \ge v_t(n_S)$$

From Theorem 1, if the number of agent types t is fixed, the number of constraints in this linear programming formula is polynomial in the number of agents n. Thus, this problem can be solved in polynomial time. □

Theorem 3. *By using a type-based characteristic function representation, determining whether a symmetric payoff vector x is in the core can be done in $O(n^t)$ time.*

Proof. For a given symmetric payoff vector $x = \langle x_1, \ldots, x_t \rangle$, we need to check whether x is feasible, i.e., $\sum_{1 \le i \le t} n_A^i \cdot x_i \ge v_t(n_A)$ holds, and $\forall n_S \in A^t$, x is not blocked by S, i.e., $\sum_{1 \le i \le t} n_S^i \cdot x_i \ge v_t(n_S)$ holds. Since $|A^t| = O(n^t)$, determining whether a symmetric payoff vector x is in the core can be done in $O(n^t)$ time. □

Theorem 4. *By using a type-based characteristic function representation, computing the Shapley value of any agent can be done in $O(n^t)$ time.*

Proof. The Shapley value averages an agent's marginal contributions over all possible orders in which the agents may join the coalition. Computing the Shapley value for an agent requires the agent's marginal contributions over all possible coalitions that the agent may join. However, the Shapley value of agents with the same type must be the same, since the Shapley value is symmetric. Also, if two coalitions are the same type, the marginal contribution of an agent when joining these coalitions is the same. Thus, we can obtain the Shapley value from the marginal contribution of each type for each $n_S \in A^t$, which can be done in $O(n^t)$ time. □

4 Coalition Structure Generation with Agent Types

In this section, we develop an algorithm for the CSG problem based on knapsack problems [7]. A multidimensional unbounded knapsack problem (MUKP) is the knapsack problem, where the knapsack has multidimensional constraint and multiple copies exist for each item. For each item j, we denote the profit as p_j, the weight of the i-th constraint as w_{ij}, and the number of copies packed in the knapsack as q_j. A MUKP with m items and t constraints of knapsack $c_1, \ldots, c_t$ is formalized as follows:

$$\begin{aligned} &\text{maximize } \textstyle\sum_j p_j q_j \\ &\text{subject to } \textstyle\sum_j w_{ij} q_j \le c_i,\ i = 1, \ldots, t \\ &\qquad\qquad q_j \ge 0,\ j = 1, \ldots, m \end{aligned}$$

Theorem 5. *By using a type-based characteristic function representation, finding an optimal coalition structure can be done in $O(n^{2t})$ time.*

Proof. We show that a CSG problem with $m = |A^t|$ coalition types and t possible agent types can be formalized as a MUKP with m items and t constraints. Let us assume that one possible coalition type $n_{S_j} \in A^t$ corresponds to item j, where its value p_j is equal to $v_t(n_{S_j})$ and its weight for the i-th constraint is equal to $n^i_{S_j}$. The capacity constraint of knapsack c_i is determined by n^i_A.

Let $z_j[d_1]\dots[d_t]$ be the optimal solution value for the knapsack problem (CSG) with j coalition types $\{n_{S_1}, \dots, n_{S_j}\}$ and a capacity constraint of knapsack $c_i = d_i, \forall i \in T$. If $z_{j-1}[d_1]\dots[d_t]$ is known for all capacity values $0 \leq d_i \leq n^i_A, \forall i \in T$, then we can include another coalition type n_{S_j} and compute the corresponding solutions $z_j[d_1]\dots[d_t]$ using the following recursive formula:

$$z_j[d_1][d_2]\dots[d_t] = \max \begin{cases} z_j[d_1 - n^1_{S_j}][d_2 - n^2_{S_j}]\dots[d_t - n^t_{S_j}] + v_t(n_{S_j}) \\ (\text{if } \forall i \in T, d_i \geq n^i_{S_j}) \\ z_{j-1}[d_1][d_2]\dots[d_t] \end{cases}$$

We can construct a dynamic programming based algorithm from this recursive formula, which takes $O(n^t \times |A^t|) = O(n^{2t})$ steps (see Section 9.3.2 in [7]). Thus, for any fixed t, finding an optimal coalition structure can be done in $O(n^{2t})$ time. □

A similar argument has been done for the winner determination problem in combinatorial auctions with a fixed number of types of items [12]. In fact, a CSG problem can be mapped into that problem by assuming each coalition type corresponds to a bid and that each type of agent corresponds to a type of item.

5 Combining with Concise Representation Schemes

5.1 Type-based SCG

We first show the original definition of SCG [4].

Definition 6. *An SCG consists of a set of pairs of the form: $(S, v(S))$. For any coalition S, the value of the characteristic function is: $v(S) = \max\{\sum_{S_i \in p_S} v(S_i)\}$, where p_S is a partition of S, i.e., all S_i are disjoint and $\cup_{S_i \in p_S} S_i = S$, and for all the S_i, $(S_i, v(S_i)) \in SCG$. To avoid senseless cases that have no feasible partitions, we require that $(\{a\}, 0) \in SCG$ whenever $\{a\}$ does not receive a value elsewhere in SCG.*

Using this original definition, we can represent only super-additive characteristic functions. To allow for characteristic functions that are not super-additive, Ohta *et al.* [8] slightly modify the definition, i.e., they add the following requirement for partition p_S: $\forall p'_S \subseteq p_S$, where $|p'_S| \geq 2, (\cup_{S_i \in p'_S} S_i, v(\cup_{S_i \in p'_S} S_i))$ is not an element of SCG. We refer to this modified definition as a standard SCG.

Next, we introduce the definition of a type-based SCG.

Definition 7. *A type-based SCG consists of a set of pairs of the form:* $(n_S, v_t(n_S))$. *For any coalition type* n_S*, the value of the characteristic function is defined in a similar way as a standard SCG.*

Theorem 6. *A type-based SCG can represent any characteristic function represented in a standard SCG using at most the same amount of space and is exponentially more concise than a standard SCG for certain games.*

We omit the proofs due to space limitations. Intuitively, the worst case occurs when the recognizable types of all agents are different. Also, when all agents have an identical type, the value of a characteristic function is determined only by the number of agents in a coalition. Then the required size of a type-based SCG becomes $O(n)$, but the size of a standard SCG can be exponential.

Example 3. A type-based SCG for Example 1 is given as follows:

$$(\langle 1, 0\rangle, 3),\ (\langle 0, 1\rangle, 2),\ (\langle 0, 2\rangle, 2),\ (\langle 2, 2\rangle, 5).$$

In this case, $v_t(\langle 2, 1\rangle) = v_t(\langle 1, 0\rangle) + v_t(\langle 1, 0\rangle) + v_t(\langle 0, 1\rangle) = 8$. Here, the type-based SCG defines the value of four coalition types, while the type-based characteristic function representation needs to describe eight possible coalition types.

Let us examine the complexity of coalition formation problems when we use the type-based SCG representation. As discussed in [4], core-related coalition formation problems remain hard in a standard SCG. However, this is due to the fact that determining the value of the grand coalition is hard. If the value of the grand coalition is given explicitly, these problems become tractable. We first prove the following lemma.

Lemma 1. *Translating a type-based SCG representation to a type-based characteristic function representation (i.e., obtaining the values of characteristic function for all coalition types that are not explicitly described in SCG) can be done in* $O(n^{2t})$ *time.*

Proof. Let us consider obtaining $v_t(n_A)$, i.e., the value of the grand coalition. It can be obtained using a method similar to the DP-based algorithm described in Theorem 5. More precisely, we consider each coalition type described in the type-based SCG as an item of the knapsack problem, where the capacity constraint of knapsack $c_i = n_A^i$ for all $i \in T$. By running the DP-based algorithm, for each possible coalition type n_S, we obtain $v_t(n_S)$, which is represented as $z_m[n_S^1] \dots [n_S^t]$. One slight difference with the DP-based algorithm described in Theorem 5 is that, if $v_t(n_S)$ is already described in the type-based SCG explicitly, we fix the value of $z_j[n_S^1] \dots [n_S^t]$ to $v(n_S)$ and do not update. The algorithm can be done in $O(n^{2t})$ time. □

Now, since the value of the grand coalition can be obtained in polynomial time, it is straightforward to show that core-related coalition formation problems are tractable.

Theorem 7. *If the number of agent types t is fixed, by using a type-based SCG representation, determining whether the core is non-empty can be done in polynomial time in the number of agents n.*

Proof. When confirming that a symmetric payoff vector x is not blocked by any coalition type, it is sufficient to check against coalition types that are explicitly described in the type-based SCG. Therefore, we construct the following linear programming formula with the constraint of coalition types described in the type-based SCG. The program is as follows:

$$\sum_{1\leq i\leq t} n_A^i \cdot x_i = v_t(n_A);\ \forall (n_S, v_t) \in SCG,\ \sum_{1\leq i\leq t} n_S^i \cdot x_i \geq v_t(n_S)$$

After we obtain $v_t(n_A)$ (by Lemma 1, this can be done in polynomial time), the above program can be solved in polynomial time in the number of agents n. □

Theorem 8. *By using a type-based SCG representation, determining whether a symmetric payoff vector x is in the core can be done in $O(n^{2t})$ time.* □

Proof. We first obtain $v_t(n_A)$. By Lemma 1, this can be done in $O(n^{2t})$ time. Next, for a symmetric payoff vector $x = \langle x_1, \ldots, x_t \rangle$, we check whether $\sum_{i\in T} n_A^i \cdot x_i = v_t(n_A)$ holds. Then, we confirm that this symmetric payoff vector is not blocked by any coalition. It is sufficient to check against coalition types explicitly described in the type-based SCG. Thus, determining whether a symmetric payoff vector x is in the core can be done in $O(n^{2t})$ time. □

Unfortunately, as far as the authors are aware, there is no efficient way to compute Shapley values using SCG-based representations. However, we can use a naive translation approach, which can be done in polynomial time.

Theorem 9. *If the number of agent types t is fixed, by using a type-based SCG representation, computing the Shapley value can be done in $O(n^{2t})$ time.*

Proof. From Lemma 1, we can compute the values of all coalition types in $O(n^{2t})$ time. Also, Theorem 4 shows that we can compute the Shapley value in $O(n^t)$ time if we know the value of all coalitions. Thus, computing the Shapley value can be solved in $O(n^{2t})$ time. □

Theorem 10. *By using a type-based SCG representation, finding an optimal coalition structure can be done in $O(n^{2t})$ time.*

Proof. Ohta *et al.* showed that there exists a coalition structure CS such that $V(CS) = V(CS^*)$ and $\forall S \in CS, (S, v(S)) \in SCG$ (see Theorem 3 in [8]). Using a similar argument, we can show that when searching CS^*, we need to consider only the coalition types that are explicitly provided in the type-based SCG. We can find an optimal coalition structure using the DP-based algorithm provided in Theorem 5, where possible coalition types (or items of the knapsack problem) are restricted to the elements appearing in the type-based SCG. This algorithm also takes $O(n^{2t})$ steps. Thus, finding an optimal coalition structure can be done in $O(n^{2t})$ time. □

5.2 Type-based MC-nets

We first show the original definition of MC-nets [6].

Definition 8. *An MC-net consists of a set of rules R. Each rule $r \in R$ is of the form: $(P_r, N_r) \to v_r$, where $P_r \subseteq A, N_r \subseteq A, P_r \cap N_r = \emptyset, v_r \in \mathbb{R}$. v_r can be either positive or negative. We say that rule r is applicable to coalition S if $P_r \subseteq S$ and $N_r \cap S = \emptyset$, i.e., S contains all agents in P_r (positive literals) but no agent in N_r (negative literals). For a coalition S, $v(S)$ is given as $\sum_{r \in R_S} v_r$, where R_S is the set of rules applicable to S.*

Next, we introduce the definition of type-based MC-nets.

Definition 9. *A type-based MC-net consists of a set of rules R. Each rule $r \in R$ is of the form: $(L_r, U_r) \to v_r$, where $L_r = \langle l_r^1, l_r^2, \ldots, l_r^t \rangle$ and $U_r = \langle u_r^1, u_r^2, \ldots, u_r^t \rangle$. Each l_r^i (and u_r^i) represents the lower (upper) bound of the number of i-th type agents in a coalition so that this rule becomes effective. We say that rule r is applicable to coalition S if $\forall i \in T, l_r^i \leq n_S^i \leq u_r^i$. For a coalition S, $v(S)$ is given as $\sum_{r \in R_S} v_r$, where R_S is the set of rules applicable to S.*

Theorem 11. *A type-based MC-net can represent any characteristic function represented in a standard MC-net using at most the same amount of space and is exponentially more concise than a standard MC-net for certain games.*

For space reasons, we omit the proof. We can prove the theorem using a similar argument described in Theorems 6.

Example 4. A type-based MC-net for Example 1 is given as follows:

$$r_1 : (\langle 1, 0 \rangle, \langle 1, 2 \rangle) \to 3,\ r_2 : (\langle 2, 0 \rangle, \langle 2, 1 \rangle) \to 6,$$
$$r_3 : (\langle 0, 1 \rangle, \langle 2, 2 \rangle) \to 2,\ r_4 : (\langle 2, 2 \rangle, \langle 2, 2 \rangle) \to 3.$$

In this case, r_2 and r_3 are applicable to coalition type $\langle 2, 1 \rangle$, but r_1 and r_4 are not. Thus, $v_t(\langle 2, 1 \rangle)$ is equal to $6 + 2 = 8$. Here, the type-based MC-net consists of four rules, while the type-based characteristic function representation needs to specify the value of eight possible coalition types.

Unfortunately, as far as the authors are aware, there is no efficient way to solve core-related coalition formation problems using MC-net-based representations. However, we can use a naive translation approach, which can be done in polynomial time. We first prove the following lemma.

Lemma 2. *Translating a type-based MC-net representation to a type-based characteristic function representation (i.e., obtaining the values of characteristic function for all coalition types) can be done in $O(n^{2t})$ time.*

Proof. We can safely assume that the number of rules of a type-based MC-net is $O(n^t)$. Otherwise, it's better to use the type-based characteristic function representation in the first place. For each $n_S \in A^t$, we initialize $v_t(n_S)$ to 0. Then, for each rule $r \in R$, and for each coalition type n_S, if the rule is applicable to n_S, we increment $v_t(n_S)$ by the value of r. Since each rule is applicable to at most n^t coalition types, this procedure can be done in $O(n^{2t})$ time. □

Theorem 12. *If the number of agent types t is fixed, by using a type-based MC-net representation, determining whether the core is non-empty can be done in polynomial time in the number of agents n. Also, determining whether a symmetric payoff vector x is in the core can be done in $O(n^{2t})$ time.*

Proof. This is clear since by Lemma 2 we can transform a type-based MC-net representation into a type-based characteristic function representation in $O(n^{2t})$ time. Then, from Theorem 2, we can determine whether the core is non-empty in polynomial time. Also, from Theorem 3, whether a symmetric payoff vector x is in the core can be checked in $O(n^t)$ time. Thus, the total required time is polynomial in the number of agents n (in particular, $O(n^{2t})$ for checking whether x is in the core). □

In contrast to core-related coalition formation problems, the standard MC-net representation is suitable for computing Shapley values. This is also true for our type-based MC-net representation, i.e., the following theorem holds.

Theorem 13. *If the number of agent types t is fixed, by using a type-based MC-net representation, computing the Shapley value of any agent can be done in $O(|R| \cdot n^{2t})$ time.*

Proof. To compute the Shapley value of an agent, we can compute its Shapley value for each rule and use the summation of these values (see Proposition 5 in [6]). Furthermore, we can decompose a rule into multiple rules, where each decomposed rule has a form: $(\langle y^1, \ldots, y^t \rangle, \langle y^1, \ldots, y^t \rangle) \to v$, i.e., the rule is applicable to exactly one coalition type. The Shapley value of type i agent for rule r (denoted as $\phi_{i,r}$) is computed by the following procedure:

$$\phi_{i,r} = \frac{v}{n!}(f_i^+(y_1, \ldots, y_t) - f_i^-(y_1, \ldots, y_t)),$$

$$f_i^+(y_1, \ldots, y_t) =$$

$$\begin{cases} 0 & \text{if } y_i = 0, \\ \Pi_{j \neq i} \, {}_{n_A^j}C_{y_j} \cdot {}_{n_A^i - 1}C_{y_i - 1} \cdot (s_y - 1)!(n - s_y)! & \text{otherwise.} \end{cases}$$

$$f_i^-(y_1, \ldots, y_t) =$$

$$\begin{cases} 0 & \text{if } y_i = n_A^i, \\ \Pi_{j \neq i} \, {}_{n_A^j}C_{y_j} \cdot {}_{n_A^i - 1}C_{y_i} \cdot (s_y)!(n - s_y - 1)! & \text{otherwise.} \end{cases}$$

$$\text{where } s_y = \sum_{j \in T} y_j, \; n = \sum_{j \in T} n_A^j.$$

Here, $f_i^+(y_1, \ldots, y_t)$ represents the number of orderings where the marginal contribution of one type i agent is v, and $f_i^-(y_1, \ldots, y_t)$ represents the number of orderings where the marginal contribution of one type i agent is $-v$. Thus, $\frac{v}{n!}(f_i^+(y_1, \ldots, y_t) - f_i^-(y_1, \ldots, y_t))$ represents the Shapley value of type i agent for this rule. Using this procedure, the required time for computing the Shapley value of an agent becomes $O(|R| \cdot n^{2t})$. □

Although Ohta *et al.* proposed an efficient method for solving CSG problems based on the standard MC-net [8], we cannot apply this method straightforwardly. Nevertheless, we can still rely on a naive approach that translates a type-based MC-net representation into the corresponding type-based characteristic function representation.

Theorem 14. *If the number of agent types t is fixed, by using a type-based MC-net representation, finding an optimal coalition structure can be done in $O(n^{2t})$ time.*

Proof. This is clear since by Lemma 2 we can transform a type-based MC-net representation into a type-based characteristic function representation in $O(n^{2t})$ time. Then, from Theorem 5, we can find an optimal coalition structure in $O(n^{2t})$ time. Thus, the total required time is $O(n^{2t})$. □

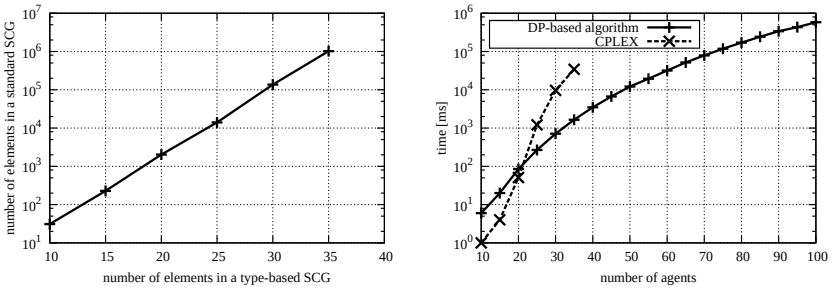
6 Experimental Evaluations

In this section, we experimentally evaluate the performance of our proposed methods. We concentrate on methods for type-based SCG, since we can control the input size of a problem instance. In addition, the DP-based algorithm is also used in other representations. All tests were run on a Core 2 Quad Q9650 3GHz processor with 16GB RAM. The test machine runs Windows 7 Enterprise x64 Edition.

Let us consider a type-based SCG problem instance, where n agents have one of five different types ($t = 5$). We vary n from 10 to 100 and set the number of elements in a type-based SCG to n, (i.e., equal to the number of agents). We generate each element using a decay distribution as follows. Initially, the required number of agents in each type is set to zero. First, we randomly choose one type and increment the required number of agents in the type by one. Then, we repeatedly choose a type randomly and increment its required number of agents with probability α, until a type is not chosen or the required number of agents exceeds the limit. We choose the value of that coalition between 1 and $10 \times n$ uniformly at random and use $\alpha = 0.55$. In this way, we generated 50 problem instances for each n.

We translate each generated problem instance represented by type-based SCG into an equivalent problem instance represented by standard SCG. In Figure 1(a), the x-axis shows the number of elements in the type-based SCG representation, and the y-axis shows the number of elements in the standard SCG representation. Each data point shows the average of 50 problem instances. The number of elements in the standard SCG grows exponentially compared to that in the type-based SCG When the number of elements in the type-based SCG representation exceeds 40, we cannot translate the problem instances due to insufficient memory. This result illustrates that the type-based representation is exponentially more concise than the standard representation.

We investigate the computation time of our DP-based algorithm. For comparison, we show the results of the MIP formulation in Ohta *et al.* [8], which uses



(a) Number of elements in a standard SCG

(b) Computation time

Fig. 1. Experimental Results

a standard SCG representation. To obtain this result, we used CPLEX version 12.1, a general-purpose mixed integer programming package.

Figure 1(b) illustrates the average computation times for solving the generated problem instances by our DP-based algorithm using the type-based SCG (DP) and by CPLEX in the MIP formulation using the standard SCG (CPLEX). The x-axis indicates the number of agents, and the y-axis shows the average computation times. When $n \leq 20$, CPLEX is faster than DP, while DP eventually outperforms CPLEX for $n > 20$. CPLEX can reduce the search space efficiently when the input size is relatively small. However, the input size for CPLEX grows exponentially. Thus, its computation time increases very rapidly. When $n > 40$, even generating problem instances becomes infeasible. On the other hand, the computation time for DP grows more slowly than the exponential rate. This result corresponds to the theoretical complexity presented in Theorem 10, i.e., finding an optimal coalition structure can be done in $O(n^{2t})$ time. As shown in this result, the type-based SCG enables us to solve a CSG problem instance with up to 100 agents in a reasonable amount of time.

7 Conclusion

In this paper, we developed a new concise representation scheme for a characteristic function, which is based on the idea of *agent types*. The type-based representation can be exponentially more concise than existing concise representation schemes. Furthermore, this idea can be used in conjunction with existing schemes, i.e., MC-nets and SCG, for further reducing the representation size. We showed that most problems in coalitional games, including CSG, can be solved in polynomial time in the number of agents, assuming the number of types t is fixed. We also experimentally showed that a type-based SCG enables us to

solve a CSG problem instance with up to 100 agents in a reasonable amount of time. Our idea of using agent types is inspired by the recent work of Shrot *et al.* [11]. However, in contrast to their study, our work introduced the idea of describing a characteristic function explicitly using agent types in the first place, and considered a wider range of problems in coalitional games including CSG.

Our future works include examining the complexity of solving other problems in coalitional games, e.g., finding the nucleolus, and combining the idea of agent types with other concise representation schemes such as [13].

References

1. Bachrach, Y., Elkind, E.: Divide and conquer: false-name manipulations in weighted voting games. In: AAMAS. pp. 975–982 (2008)
2. Bachrach, Y., Rosenschein, J.S.: Coalitional skill games. In: AAMAS. pp. 1023–1030 (2008)
3. Chalkiadakis, G., Elkind, E., Jennings, N.R.: Simple coalitional games with beliefs. In: IJCAI. pp. 85–90 (2009)
4. Conitzer, V., Sandholm, T.: Complexity of constructing solutions in the core based on synergies among coalitions. Artificial Intelligence 170(6), 607–619 (2006)
5. Elkind, E., Pasechnik, D.V.: Computing the nucleolus of weighted voting games. In: SODA. pp. 327–335 (2009)
6. Ieong, S., Shoham, Y.: Marginal contribution nets: a compact representation scheme for coalitional games. In: EC. pp. 193–202 (2005)
7. Kellerer, H., Pferschy, U., Pisinger, D.: Knapsack Problems. Springer (2004)
8. Ohta, N., Conitzer, V., Ichimura, R., Sakurai, Y., Iwasaki, A., Yokoo, M.: Coalition structure generation utilizing compact characteristic function representations. In: CP. pp. 623–638 (2009)
9. Rahwan, T., Jennings, N.R.: An improved dynamic programming algorithm for coalition structure generation. In: AAMAS. pp. 1417–1420 (2008)
10. Sandholm, T., Larson, K., Andersson, M., Shehory, O., Tohmé, F.: Coalition structure generation with worst case guarantees. Artificial Intelligence 111(1-2), 209–238 (1999)
11. Shrot, T., Aumann, Y., Kraus, S.: On agent types in coalition formation problems. In: AAMAS. pp. 757–764 (2010)
12. Tennenholtz, M.: Some tractable combinatorial auctions. In: AAAI. pp. 98–103 (2000)
13. Ueda, S., Iwasaki, A., Yokoo, M., Silaghi, M.C., Hirayama, K., Matsui, T.: Coalition structure generation based on distributed constraint optimization. In: AAAI. pp. 197–203 (2010)