Extending MaxSAT to Solve the Coalition Structure Generation Problem with Externalities Based on Agent Relations

Xiaojuan LIAO(a), Nonmember, Miyuki KOSHIMURA(b), Hiroshi FUJITA(c), and Ryuzo HASEGAWA(d), Members

SUMMARY
Coalition Structure Generation (CSG) means partitioning agents into exhaustive and disjoint coalitions so that the sum of values of all the coalitions is maximized. Solving this problem could be facilitated by employing some compact representation schemes, such as marginal contribution network (MC-net). In MC-net, the CSG problem is represented by a set of rules where each rule is associated with a real-valued weights, and the goal is to maximize the sum of weights of rules under some constraints. This naturally leads to a combinatorial optimization problem that could be solved with weighted partial MaxSAT (WPM). In general, WPM deals with only positive weights while the weights involved in a CSG problem could be either positive or negative. With this in mind, in this paper, we propose an extension of WPM to handle negative weights and take advantage of the extended WPM to solve the MC-net-based CSG problem. Specifically, we encode the relations between each pair of agents and reform the MC-net as a set of Boolean formulas. Thus, the CSG problem is encoded as an optimization problem for WPM solvers. Furthermore, we apply this agent relation-based WPM with minor revision to solve the extended CSG problem where the value of a coalition is affected by the formation of other coalitions, a coalition known as externality. Experiments demonstrate that, compared to the previous encoding, our proposed method speeds up the process of solving the CSG problem significantly, as it generates fewer number of Boolean variables and clauses that need to be examined by WPM solver.

key words: weighted partial MaxSAT, coalition structure generation, externality, cooperative games

1. Introduction

1.1 Background

Coalition formation is an important capacity in multi-agent systems. It is a fundamental type of interaction that involves the creation of coherent groupings of distinct, autonomous agents in order to efficiently achieve their individual or collective goals. The area of coalition formation has been proved to be useful in a number of real-world scenarios. For example, in e-commerce, customers can form coalitions to purchase a product in bulk and take advantage of price discount [3]. In distributed sensor networks, coalitions of sensors can work together to track targets of interest [4]. In distributed vehicle routing, coalitions of delivery companies can be formed to reduce the transportation costs by sharing deliveries [5]. Coalition formation can also be used for information gathering, where several information servers form coalitions to answer queries [6]. In general, coalition formation is composed of three main activities [2]:

1. coalition value calculation,
2. coalition structure generation,
3. payoff distribution,

among which, Coalition Structure Generation (CSG) is the main research issue in the use of coalitional games in multi-agent systems. It involves partitioning a set of agents into exhaustive and disjoint coalitions, where each coalition is assigned a real-valued payoff. This partition is called a coalition structure. Solving the CSG problem amounts to finding a coalition structure such that the total value of all the coalitions is maximized.

The general assumption made in solving the CSG problem is that given any coalition structure, the value of a coalition is not affected by the way non-members are partitioned. Such settings are known as Characteristic Function Games (CFGs), where the value of a coalition is given by a characteristic function. Recently, there has been interest in more realistic partition function form of coalition values, where the value of a coalition is affected by the formation of other coalitions. This class of coalitional games is called Partition Function Games (PFGs). The effect is known as externality from coalition formation. Examples of games with externalities include collusion in oligopolies, congestion games, as well as various forms of international policy coordination between countries [7], [8].

The space of possible solutions to the CSG problem is the same for both CFGs and PFGs, with O (n^m) for n agents. However, it should be noted that the input size is O (2^n) for CFGs, and O (n^m) for PFGs [9]. Therefore, solving the CSG problem for PFGs is more challenging than the CFG case, which is already NP-complete [10].

1.2 Related Works

There have been a number of attempts to solve the CSG problem for CFGs; they are broadly classified as dynamic programming [11], [12], anytime optimal algorithms [2],...
[13], and the combination of the two approaches [14], [15]. By contrast, to date, few works have attempted to solve the CSG problem in PFGs. The initial work assumed that the externalities are either always positive or always negative [9], [16]. These assumptions were eliminated in [17], which incorporates mixed externalities in the agent type-based framework, i.e., positive and negative externalities could co-exist in a problem instance. All of the above works represent characteristic functions or partition functions by listing the values of all possible coalitions, thus requiring space exponential in the number of agents in the game [18]. When the number of agents becomes large, solving the problem with the naive representation is prohibitive. That is why these works could only solve problem instances with only a few dozen agents.

In order to reduce the representation size for the CSG problem, a variety of concise representations have been developed [19]–[22], where a characteristic function or a partition function is represented by a set of rules. In this context, Ohta et al. [20] initially formalized the CSG problem as a problem of finding the subset of rules that maximizes the sum of rule values under certain constraints. Their work takes an initial step towards developing efficient constraint optimization algorithms for solving the CSG problem in CFGs, showing that, with concise representations such as MC-net [18], the CSG problem could be solved within significantly less time than other works. In the case of PFGs, applying compact representation schemes to the CSG problem is more desirable, as PFGs require greater space than CFGs if we use the naive representation. In this context, embedded MC-net [23], a compact representation designed for coalitional games with externalities, has been developed. By using the embedded MC-net, Ueda et al. [21] extended the formalization of CSG in [20] and developed a direct encoding algorithm to handle the externalities among the CSG problem. Later on, Liao et al. [24] encoded Ohta and Ueda’s algorithms into Boolean propositional logic and showed that weighted partial MaxSAT solvers can contribute as a more efficient constraint optimization tool to solving the CSG problem.

1.3 Our Contribution

Boolean Satisfiability (SAT) solver is used as a tool for solving the combinatorial problem and MaxSAT, an optimization version of SAT, is for solving the combinatorial optimization problem. Inspired by the previous works [20], [21], which generalized the CSG problem as an optimization problem under certain constraints, one interesting question is whether it is possible for a MaxSAT solver to further speed up the process of solving the CSG problem. Moreover, as a variation of MaxSAT, weighted partial MaxSAT (WPM) deals with both hard and soft formulas with only positive weights, while the CSG problem may contain both positive and negative weights. In order to apply WPM solvers to the CSG problem, an inevitable problem to be solved is how to enable WPM solvers to handle formulas with negative weights.

We address the above problems by making the following contributions:

- Weighted partial MaxSAT (WPM) requires all weights associated with soft formulas must be positive. In this paper, we extend the WPM to deal with negative weights, referred to as extended WPM (EWPM), thus any propositional formulas with non-zero weights could be handled by an off-the-shelf MaxSAT solver. Moreover, in order to solve EWPM instances with WPM solvers, we present transformation from EWPM to WPM and examine the relationship between EWPM and WPM solutions. In particular, when an EWPM instance has only negative weights, we can regard it as a MinSAT instance. Thus, we can say that EWPM is an extension of both MaxSAT and MinSAT [25].

- We seek for a more efficient WPM encoding to solve the CSG problem that is represented by a set of rules. Instead of encoding rule relations into propositional logic [24], we capture a more fine-grained relation, i.e., agent relation, and show that this agent relation-based encoding manages to solve the CSG problem with higher efficiency. More specifically, problem instances for PFGs with 120 agents (rules) could be solved within 4 seconds on average, while the rule relations-based WPM, which was directly encoded from [20], [21], requires averagely 9 seconds to solve the same set of instances.

This paper is organized as follows. Preliminaries are given in Sect. 2. The extended WPM, designed for dealing with negative weights is introduced in Sect. 3, which paves the way for the agent relation-based encoding described in Sect. 4. Section 5 details the evaluation results and Sect. 6 concludes this paper.

2. Preliminary

Given a set of agents \( A \), a coalition, denoted by \( C \), is a non-empty subset of \( A \), i.e., \( C \subseteq 2^A \setminus \emptyset \). A coalition structure, \( CS \), is an exhaustive set of mutually disjoint coalitions over \( A \), i.e., \( CS \) is subject to the constraints: \( \forall i, j (i \neq j), C_i \cap C_j = \emptyset, \bigcup_{C_i \in CS} C_i = A \). We denote by \( \Pi(A) \) the set of all coalition structures over \( A \).

2.1 Characteristic Function Game

In a setting without externalities, the value of a coalition \( C \) is given by a characteristic function \( v : 2^A \to \mathbb{R} \), assigning a real-valued payoff to each coalition \( C \subseteq A \). The value of a coalition structure \( CS \) is called social welfare, denoted as \( V(CS) \), given by \( V(CS) = \sum_{C \in CS} v(C) \). The objective of solving the CSG problem is to find an optimal coalition structure that makes the social welfare maximized, i.e., given \( A \), find \( CS^* \) such that \( V(CS^*) \geq V(CS) \).
Ieng and Shoham [18] develop a concise representation called marginal contribution network (MC-net), which largely reduces the space necessary for representation.

**Definition 1**: (MC-nets). An MC-net consists of a set of rules \( R \). Each rule \( r_i \in R \) is expressed in a syntactic form \( l_i \rightarrow w_i \), where \( w_i \in \mathbb{R} \) and \( l_i \) is the condition of rule \( r_i \), denoted by a conjunction of literals over \( A \), i.e., \( \{ a_1 \wedge a_2 \wedge \cdots \wedge a_k \} \). For each rule \( r_i \), we call \( P_i \) positive literals where \( P_i = \{ a_j \}_{j=1}^{k} \) and \( N_i \) negative literals where \( N_i = \{ a_j \}_{j=1}^{m} \). A rule \( r_i \) is said to apply to a coalition \( C \) if \( P_i \subseteq C \) and \( N_i \cap C = \emptyset \), i.e., all agents in \( P_i \) are in \( C \), and none of agents in \( N_i \) are in \( C \). For a coalition \( C \), \( v(C) = \sum_{r_i \in R^C} w_i \), where \( R^C \) is the set of rules that apply to \( C \). Thus, the value of a coalition structure \( CS \) is given as \( \sum_{C \in \mathcal{CS}} v(C) \). Without loss of generality, we assume each rule has at least one positive literal.

### 2.2 Partition Function Game

In a setting with externalities, an embedded coalition is a pair \((C, CS)\). Let \( M \) denote the set of all embedded coalitions, i.e., \( M := \{(C, CS) : CS \in \Pi(A), C \in CS\} \). The value of a coalition depends on the formation of other co-existing coalitions in the coalition structure, and is specified by a partition function, which is a mapping \( w : M \rightarrow \mathbb{R} \). The problem of solving CSG seeks a coalition structure \( CS \), such that \( V(CS^*) = \sum_{C \in \mathcal{CS}} w(C, CS^*) \) is optimized. Michalak et al. [23] extend MC-nets to a partition function, called embedded MC-nets.

**Definition 2**: (Embedded MC-nets). An embedded MC-net is given by a set of embedded rules \( ER \). Each rule \( er \in ER \) is expressed in a syntactic form \( l_0 | l_1 | \cdots | l_k \rightarrow w_{er} \), where each \( l_i \) is a conjunction of literals over \( A \). A rule \( er \) is said to apply to an embedded coalition \((C, CS)\) if \((1)\) \( l_0 \) applies to \( C \), and \((2)\) each \( l_i \) is a coalition in \( CS \). For an embedded coalition \((C, CS)\), \( w(C, CS) = \sum_{er \in ER^{C}} w_{er} \), where \( ER^{C} \) is the set of embedded rules that apply to \((C, CS)\).

**Example 1**. For \( A = \{a_1, a_2, a_3, a_4\} \), assume there are four rules: \( r_1 : a_1 \land a_2 \rightarrow 2, r_2 : a_3 \land \neg a_4 \rightarrow 1, r_3 : a_3 \land \neg a_1 \rightarrow 1, r_4 : a_1 \land a_2 \land a_4 \land \neg a_3 \rightarrow 1 \). If \( CS = \{\{a_1, a_2\}, \{a_3\}, \{a_4\}\} \), then \( r_1, r_2, r_3 \) apply to \( \{a_1, a_2\} \), \( \{a_3\} \), \( \{a_4\} \), respectively, and \( r_4 \) applies to \( \{a_1, a_2\}, \{a_3\}, \{a_4\}\). Thus, \( V(CS) = 2 + 1 + 1 + 1 = 5 \). This is also the optimized social welfare, i.e., \( V(CS^*) = V(\{a_1, a_2\}, \{a_3\}, \{a_4\}) = 5 \).

### 2.3 Weighted Partial MaxSAT

A weighted clause is a pair \((C, w)\), where \( C \) is a clause and \( w \), its weight, is a positive integer or infinity. A clause is called hard if its weight is infinity, otherwise it is soft. A weighted partial MaxSAT (WPM) instance is a multiset of weighted clauses \( \phi = [(C_1, w_1), \ldots, (C_m, w_m), (C_m+1, \infty), \ldots, (C_{m+m'}, \infty)] \), where the first \( m \) clauses are soft and the last \( m' \) clauses are hard. A truth assignment of \( \phi \) is a mapping that assigns to each variable in \( \phi \) either 0 or 1.

Given a WPM instance \( \phi \) and a truth assignment \( I \), the benefit of \( I \) on \( \phi \), noted \( \text{benefit}(\phi, I) \), is the sum of the weights of the soft clauses satisfied by \( I \) if \( I \) satisfies all the hard clauses, otherwise, \( \text{benefit}(\phi, I) = -\infty \). \( \phi \) is satisfiable if \( \phi \) has a truth assignment which satisfies all the hard clauses, otherwise it is unsatisfiable. The WPM problem for a WPM instance \( \phi \) is the problem of finding an optimal assignment \( I \) of \( \phi \) that maximizes \( \text{benefit}(\phi, I) \), that is, \( \text{benefit}(\phi, I) \geq \text{benefit}(\phi, I') \) for an arbitrary assignment \( I' \).

### 3. Extended Weighted Partial MaxSAT (EWPM)

In WPM, it is natural to limit the weights of soft clauses in the positive domain because the original intention of MaxSAT is maximizing the number of satisfied clauses. However, there are problems which we express as formulas with both positive and negative weights naturally, such as the CSG problem mentioned earlier in this paper. Moreover, current WPM solvers can usually only deal with a formula in conjunction normal form (CNF): a conjunction of clauses, while many combinatorial optimization problems that arise in the real world do not happen to be CNF formulas. This section presents the standard extension of WPM, which is able to handle propositional Boolean formulas with both positive and negative weights, and examines the relationship between solutions of WPM and its extension.

#### 3.1 EWPM-to-WPM Transformation

As an extension of WPM, an extended weighted partial MaxSAT (EWPM) instance is a multiset of weighted formulas \( \phi^E = [(F_1, w_1), \ldots, (F_m, w_m), (F_{m+1}, \infty), \ldots, (F_{m+m'}, \infty)] \), where the first \( m \) formulas are soft and the last \( m' \) formulas are hard. A weighted formula is a pair \((F, w)\), where \( F \) is a propositional Boolean formula and \( w \) is a non-zero integer or infinity. A soft formula is called positive if its weight is positive while it is called negative if its weight is negative.

Given an EWPM instance \( \phi^E \) and a truth assignment \( I^E \), the benefit of \( I^E \) on \( \phi^E \), noted \( \text{benefit}(\phi^E, I^E) \), is the sum of the weights of the soft formulas satisfied by \( I^E \) if \( I^E \) satisfies all the hard formulas, otherwise, \( \text{benefit}(\phi^E, I^E) = -\infty \). \( \phi^E \) is satisfiable if \( \phi^E \) has a truth assignment which satisfies all the hard formulas, otherwise it is unsatisfiable. The EWPM problem for an EWPM instance \( \phi^E \) is the problem of finding an optimal assignment \( I^E \) of \( \phi^E \) that maximizes \( \text{benefit}(\phi^E, I^E) \), that is, \( \text{benefit}(\phi^E, I^E) \geq \text{benefit}(\phi^E, I'^E) \) for an arbitrary assignment \( I'^E \).

Given all hard clauses are satisfied, a natural interpretation of a soft formula \((F, w)\) is that we gain \( w \) dollars when \( F \) is true. Under the interpretation, a negative soft formula \((F, -20)\) means that we lose 20 dollars when \( F \) is true. According to the following Definition 3, we transform \((F, -20)\)
to \((-b, 20)\) where \(b\) is logically equivalent to \(F\). \((-b, 20)\) means that we gain 20 dollars when \(b\), i.e. \(F\) is false. Someone may consider this is strange because the original soft formula \((F, -20)\) implies that neither gain nor loss is obtained when \(F\) is false. This strange meaning can be resolved by the following settings: We pay a deposit of \(-w\) dollars for a negative soft formula \((F, w)\) \((w < 0)\) and have \(-w\) dollars refunded when \(F\) is false.

Take \((F, -20)\) as an example. We first pay 20 dollars as the deposit, then examine the truth value of \(F\). When \(F\) is false, we have 20 dollars refunded. Thus, the final payoff is 0 when \(F\) is false. This is in accordance with \((F, -20)\). When \(F\) is true, 20 dollars should be paid according to \((F, -20)\). Since the deposit 20 dollars has been paid beforehand, this amount of money is taken into the consideration. The total deposit is the negative sum of all negative weights. It is the negative \(W_{neg}\) in Lemmas 1, 2, and Theorem 1 below.

**Definition 3:** (EWPM-to-WPM Transformation). Let 
\[
\phi^E = \langle (F_1, w_1), \ldots, (F_m, w_m) \rangle
\]
be an EWPM instance where \(w_i\) is non-zero integer and \((F_i, w_i)\) is a soft formula \((1 \leq i \leq m)\). The EWPM-to-WPM transformation consists of two steps:

1. (For each soft formula \((F_i, w_i)\), a new Boolean variable \(b_i\) is introduced. The soft formula is then transformed into a soft clause:
   
   \[
   (b_i, w_i) \quad \text{if} \quad w_i \geq 0
   
   \quad \neg(b_i, -w_i) \quad \text{if} \quad w_i < 0
   \]
   and two hard formulas ensuring that \(F_i\) and \(b_i\) are logically equivalent:
   
   (ii) \( (F_i \rightarrow b_i, \infty) \)
   
   (iii) \( (b_i \rightarrow F_i, \infty) \)

2. Transform hard formulas into hard clauses with a satisfiability preserving CNF transformation.

Step (1) aims to introduce a new Boolean variable \(b_i\) that is logically equivalent to \(F_i\), so that the soft formula \((F_i, w_i)\) could be transformed to a soft clause \((b_i, w_i)\) or \(\neg(b_i, -w_i)\), as shown in (i). The logical equivalence between \(b_i\) and \(F_i\) is ensured by (ii) and (iii). The aim of step (2) is to encode hard formulas (ii) and (iii) into a set of clauses. In this step, any satisfiability preserving CNF transformations could be applied [26].

**Lemma 1:** Let \(\phi^E\) be a satisfiable EWPM instance, \(\phi\) be a WPM instance obtained from \(\phi^E\) by applying EWPM-to-WPM transformation, and \(W_{neg}\) be the sum of all negative weights in \(\phi^E\).

If \(I^E\) is a truth assignment of \(\phi^E\) and satisfies all the hard formulas in \(\phi^E\), then there exists a truth assignment \(I\) of \(\phi\) such that \(I\) satisfies all the hard clauses in \(\phi\) and 
\[
benefit(\phi^E, I^E) = W_{pos}^E + \sum_{i=1}^{p} w_i^E \quad \text{where} \quad W_{pos}^E \text{ denotes the sum of the weights of positive soft clauses satisfied with } I^E.
\]
We should notice that \(\sum_{i=1}^{p} w_i^E + \sum_{j=1}^{q} w_j^E = W_{neg}^E\).

Now, we make a truth assignment \(I\) of \(\phi\) by extending \(I^E\) so as to satisfy all the hard clauses in \(\phi\) as follows. If \(I^E\) satisfies a soft clause \((F_i, w_i)\), we let \(I\) satisfy \(b_i\) which is a new variable introduced by the transformation, otherwise, we let \(I\) falsify \(b_i\). It is obvious that \(I\) satisfies the hard formulas (ii) and (iii) in the transformation. Thus, we can make \(I\) satisfy all the hard clauses in \(\phi\) because we use a satisfiability-preserving CNF transformation in (2) of the transformation. From p negative soft formulas \((F_i^1, w_i^1), (F_i^2, w_i^2), \ldots, (F_i^p, w_i^p)\), \(p\) soft clauses \((-b_i^1, -w_i^1), (-b_i^2, -w_i^2), \ldots, (-b_i^p, -w_i^p)\) are obtained, and, from \(q\) negative soft formulas \((F_j^1, w_j^1), (F_j^2, w_j^2), \ldots, (F_j^q, w_j^q)\), \(q\) soft clauses \((-b_j^1, -w_j^1), (-b_j^2, -w_j^2), \ldots, (-b_j^q, -w_j^q)\) are obtained by the transformation where \(b_i^1\) and \(b_j^1\) are new variables introduced. \(I\) falsifies \(p\) soft clauses \((-b_i^1, -w_i^1), \ldots, (-b_i^p, -w_i^p)\) and satisfies \(q\) soft clauses \((-b_j^1, -w_j^1), \ldots, (-b_j^q, -w_j^q)\).

Thus, 
\[
benefit(\phi, I) = W_{pos} + \sum_{i=1}^{p} w_i^E - \sum_{j=1}^{q} w_j^E = \sum_{i=1}^{p} w_i^E + \sum_{j=1}^{q} w_j^E = W_{neg}.
\]

**Lemma 2:** Let \(\phi^E, \phi,\) and \(W_{neg}\) be the same as those in Lemma 1. If \(I\) is a truth assignment of \(\phi\) and satisfies all the hard clauses in \(\phi\), then there exists a truth assignment \(I^E\) of \(\phi^E\) such that \(I^E\) satisfies all the hard formulas in \(\phi^E\) and 
\[
benefit(\phi^E, I^E) = benefit(\phi, I) + W_{neg}.
\]
Proof 2: We make a truth assignment \(I^E\) by restricting \(I\) to variables in \(\phi^E\). By a similar way in the proof of Lemma 1, we conclude \(I^E\) satisfies the above conditions.

By Lemmas 1 and 2, we conclude the following theorem.

**Theorem 1:** Let \(\phi^E\) be a satisfiable EWPM instance and \(\phi\) be a WPM instance obtained from \(\phi^E\) by applying EWPM-to-WPM transformation. Then, \(\phi\) is satisfiable. Furthermore, if \(I^E_{sol}\) is an EWPM solution of \(\phi^E\) and \(I_{sol}\) is a WPM solution of \(\phi\), then 
\[
benefit(\phi^E, I^E_{sol}) = benefit(\phi, I_{sol}) + W_{neg}
\]

where \(W_{neg}\) is the sum of all negative weights in \(\phi^E\).

**Proof 3:** According to Lemma 1, there exists a truth assignment \(I'\) of \(\phi\) such that 
\[
benefit(\phi, I') = benefit(\phi, I) + W_{neg}
\]

According to Lemma 2, there exists a truth assignment \(I^E\) of \(\phi^E\) such that \(benefit(\phi^E, I^E) = benefit(\phi, I_{sol}) + W_{neg}\)

Here, \(benefit(\phi^E, I^E_{sol}) \geq benefit(\phi^E, I^E)\) and \(benefit(\phi, I_{sol}) \geq benefit(\phi, I')\) because \(benefit(\phi^E, I^E_{sol})\) and \(benefit(\phi, I_{sol})\) are maximal benefits. These inequalities and the above two equalities imply 
\[
benefit(\phi, I_{sol}) = benefit(\phi, I') + benefit(\phi^E, I^E_{sol}) = benefit(\phi, I_{sol}) + W_{neg}.
\]
In essential, the EWPM-to-WPM transformation encodes a formula $F_i$ into a logically equivalent Boolean variable $b_i$ by introducing a set of hard clauses ensuring the logical equivalence, so that a soft formula $(F_i, w_i)$ could be transformed to a soft clause $(b_i, w_i)$ or $(\lnot b_i, \lnot w_i)$. During this process, the quality of assignment remains unchanged except that $F_i$ is replaced with $b_i$. Since $F_i$ and $b_i$ are logically equivalent, it is obvious that the truth assignment $I^-_\text{sol}$ is the same as $I^-_\text{sol}$.

Under the assumption of Theorem 1, the total deposit is $-W_{\text{neg}}$ dollars. Thus, the expression $(\text{benefit}(\phi^E, I^-_\text{sol}) = \text{benefit}(\phi, I_\text{sol}) + W_{\text{neg}})$ in Theorem 1 means that the maximal gain from $\phi^E$ equals the difference between maximal gain from $\phi$ and the total deposit.

3.2 Redundancy in Transformation
The hard formula (ii) for a positive soft formula and the hard formula (iii) for a negative soft formula in Definition 3 are redundant for solving an EWPM problem. That is, without these hard formulas, Theorem 1 holds.

Proposition 1: Let $\phi^E$ be a satisfiable EWPM instance and $\phi$ be a WPM instance obtained from $\phi^E$ by applying EWPM-to-WPM transformation. Let $\phi^-$ denote a multiset of weighted clauses shown as follows:

(1) soft clauses in $\phi$, generated from step (1)-(i),
(2) hard clauses in $\phi$, generated from step (1)-(i)-(iii) if $(F_i, w_i)$ is a positive formula,
(3) hard clauses in $\phi$, generated from step (1)-(ii) if $(F_i, w_i)$ is a negative formula,
(4) hard clauses in $\phi$, generated from step (2).

In other words, $\phi^-$ is obtained from $\phi$ by eliminating the hard clauses from (ii) for positive soft formulas and (iii) for negative soft formulas.

If $I_\text{sol}$ is a WPM solution of $\phi$ and $I^-_\text{sol}$ is a WPM solution of $\phi^-$, then $\text{benefit}(\phi, I_\text{sol}) = \text{benefit}(\phi^-, I^-_\text{sol})$.

Proof: $\phi$ is a superset of $\phi^-$, and the soft clauses in $\phi$ and those in $\phi^-$ are the same, so $I_\text{sol}$ is a truth assignment of $\phi^-$ and satisfies all the hard clauses in $\phi^-$. Therefore, $\text{benefit}(\phi, I_\text{sol}) \leq \text{benefit}(\phi^-, I^-_\text{sol})$ because $I^-_\text{sol}$ gives the maximal benefit of $\phi^-$. Next, we prove $\text{benefit}(\phi, I_\text{sol}) \geq \text{benefit}(\phi^-, I^-_\text{sol})$ by showing that $I^-_\text{sol}$ satisfies all the hard clauses in $\phi$.

Consider a positive soft formula $(F_i, w_i)$ ($w_i > 0$). For this soft formula, the soft clause $(b_i, w_i)$ is generated. Assume that $I_\text{sol}$ falsifies the hard formula (ii) $(F_i \rightarrow b_i, \infty)$, then $I^-_\text{sol}$ satisfies $F_i$ and falsifies $b_i$. Suppose that a truth assignment $I^-_\text{sol}$ is almost the same as $I^-_\text{sol}$ except for the assignment to $b_i$. In other words, the only difference between $I^-_\text{sol}$ and $I^-_\text{sol}$ is that $b_i$ is satisfied in $I^-_\text{sol}$ but unsatisfied in $I^-_\text{sol}$. In this condition, the hard clause $(b_i, w_i)$ is satisfied under the assignment $I^-_\text{sol}$, thus the benefit given by $I^-_\text{sol}$ is $w_i$ greater than $\text{benefit}(\phi^-, I^-_\text{sol})$. This contradicts the maximality of $\text{benefit}(\phi^-, I^-_\text{sol})$. Consequently, $I^-_\text{sol}$ has to satisfy the hard formula (ii). Thus, $I^-_\text{sol}$ satisfies the hard clauses from (ii) while these clauses are not contained in $\phi^-$.

By the similar way, we can show that $I^-_\text{sol}$ satisfies the hard clauses obtained from the hard formula (iii) for a negative soft formula. Consequently, $I^-_\text{sol}$ satisfies all the hard clauses in $\phi$.

Proposition 1 relieves the EWPM-to-WPM transformation of redundant clauses. If the formula to be handled is positive, (ii) is skipped, otherwise, (iii) can be omitted safely. In the experiments in Sect. 5, we also omit these redundant clauses.

3.3 Considerations
Let us consider MinSAT that is an alternative to MaxSAT. MinSAT is the problem of finding a truth assignment that satisfies all the hard clauses and minimizes the sum of weights of satisfied soft clause. We can say that EWPM “minimizes” the sum of absolute values of weights of satisfied negative soft formulas. From this point of view, we can solve MinSAT with EWPM.

Let $\phi^\text{Min} = \{(C_1, w_1), \ldots, (C_m, w_m), (C_{m+1}, \infty), \ldots, (C_{m+n'}, \infty)\}$ be a satisfiable weighted partial MinSAT instance. We make an EWPM instance $\phi^\text{Max}$ by negating the weights in $\phi^\text{Min}$:

$\phi^\text{Max} = \{(C_1, -w_1), \ldots, (C_m, -w_m), (C_{m+1}, \infty), \ldots, (C_{m+n'}, \infty)\}$

If a truth assignment satisfies all the hard clauses in $\phi^\text{Min}$, then it satisfies all the hard clauses in $\phi^\text{Max}$, and vice versa. It must be noted that $\text{benefit}(\phi^\text{Max}, I) = -\text{benefit}(\phi^\text{Min}, I)$ where $I$ is a truth assignment. This implies that $\text{benefit}(\phi^\text{Max}, I)$ becomes larger as $\text{benefit}(\phi^\text{Min}, I)$ becomes smaller. Consequently, a MinSAT solution of $\phi^\text{Min}$, which gives the minimum benefit of $\phi^\text{Min}$, is a MaxSAT solution of $\phi^\text{Max}$, which gives the maximum benefit of $\phi^\text{Max}$. Reversely, a MaxSAT solution of $\phi^\text{Max}$ is a MinSAT solution of $\phi^\text{Min}$.

In short, $\phi^\text{Min}$ and $\phi^\text{Max}$ have the same solution; accordingly, we may say that EWPM is not only an extension of MaxSAT but also that of MinSAT, or EWPM is an integration of MaxSAT and MinSAT.

4. Encoding Agent Relations into WPM
We apply EWPM described in the previous section to the CSG problem. In MC-nets or embedded MC-nets, each (embedded) rule contains at least two agents, one of which contains only one agent. In this case, the rule is always selected because no matter how a $C_S$ is structured, a rule with a single agent $a$ always applies to a coalition that contains $a$. Without loss of generality, in the rest of the paper, we assume each rule contains at least two agents, one of which must be positive literal.
4.1 Agent Relation

Let \( A_r \) be the set of agents in rule \( r \), \( A_r^+ \) be the set of negative literals in \( r \), and \( A_r^- \) be the set of positive literals in \( r \). We define a Boolean variable \( C_{i,j} \) if \( i < j \) for two agents \( a_i \) and \( a_j \). \( C_{i,j} = 1 \) if \( a_i \) and \( a_j \) are in the same coalition, otherwise \( C_{i,j} = 0 \).

Given a rule \( r \) of \( m \) agents, we call \( a_p \) a standard agent if \( p \) is the minimum index of all positive literals in \( r \), formally, \( a_p \in A_r^+, \forall a_l \in A_r^- \setminus \{a_p\}, p < l \). With \( a_p \), the agent relation between each pair of agents in \( r \) can be captured by the relations between \( a_p \) and \( a_i, a_k \in A_r \setminus \{a_p\} \). The agent relation between \( a_p \) and \( a_i, a_k \) is denoted by the Boolean variable \( C_{p,k} \) if \( p < k \) or \( C_{k,p} \) (if \( p > k \)). Intuitively, the value of the Boolean variable is equal to 1 if \( a_k \in A_r^+ \), and equal to 0 if \( a_k \in A_r^- \).

In Definition 1, the condition of a rule is formally expressed by \( \{ a_1 \land a_2 \land \cdots \land a_i \land \neg a_{i+1} \land \cdots \land \neg a_m \} \). In this case, the standard agent is \( a_1 \). We interpret the condition of a rule by using the agent relations between the standard agent \( a_1 \) and other agents in \( A_r \), showing in Definition 4.

**Definition 4:** (Agent relation-based MC-nets). An agent relation-based MC-net consists of a set of rules \( R \). Each rule \( r_i \in R \) is expressed in a soft formula \((S_i, w_i)\), where \( S_i \) is a conjunction of agent relations, formally expressed by \( C_{i,1} \land \cdots \land C_{i,j} \land \neg C_{i,j+1} \land \cdots \land \neg C_{i,m} \). A rule \( r_i \) is said to apply to a coalition \( C \) if \( \{C_{i,k}\}_{k=2}^{m} \) are all true, and \( \{C_{i,k}\}_{k=m+1}^{m} \) are all false.

To handle embedded rules, we reform the explicit form of \( er : I_0 \mid I_1, \ldots, I_r \rightarrow w_r \). For each \( I_i (i \in \{1, \ldots, k\}) \), we first investigate whether \( P_0 \cap N_i \neq \emptyset \) or \( P_i \cap N_0 \neq \emptyset \) holds. If at least one of the conditions holds, it means \( I_0 \) and \( I_i \) cannot apply to the same coalition by nature, then nothing would be done. Otherwise, we add one agent in \( P_i \) to \( N_0 \), so that the coalition that \( I_i \) applies to (suppose \( C \)) contains at least one agent in \( N_0 \), which prohibits \( I_0 \) from applying to \( C \). This explicit form guarantees that \( I_0 \) and \( I_i \) \((i \in \{1, \ldots, k\})\) cannot apply to the same coalition. For convenience, we assume in the rest of this paper, that each embedded rule is expressed in our reformulated explicit form.

**Example 2.** For an embedded rule \( er : a_1 \land a_2 \mid a_4 \land \neg a_3 \rightarrow 1 \), the explicit form of \( er \) is \( a_1 \land a_2 \land \neg a_4 \land a_3 \land \neg a_5 \rightarrow 1 \).

The definition of the agent relation-based MC-nets could be extended to the embedded MC-nets, as Definition 5 shows.

**Definition 5:** (Agent relation-based embedded MC-nets). An agent relation-based embedded MC-net consists of a set of embedded rules \( ER \). Each rule \( er \in ER \) is expressed in a soft formula \((S_0 \land S_1 \land \cdots \land S_r, w_0) \), where each \( S_i \) : \( i \in \{0, \ldots, k\} \) is a conjunction of agent relations, as defined in Definition 4. An embedded rule \( er \) is said to apply to an embedded coalition \((C, CS)\) if \( \{S_i\}_{i=0}^{k} \) are all true.

Apparently, the relation that two agents are in the same coalition is transitive, i.e., any three agents \( a_i, a_j, \) and \( a_k \) \((1 \leq i < j < k \leq m)\) satisfy the following transitive laws:

- \( \neg C_{i,j} \lor \neg C_{j,k} \lor C_{i,k} \)
- \( \neg C_{i,j} \lor \neg C_{i,k} \lor C_{j,k} \)
- \( \neg C_{i,k} \lor \neg C_{j,k} \lor C_{i,j} \)

where \( m \) is the number of agents. The number of hard clauses for representing the transitive laws is \( m \cdot (m - 1) \cdot (m - 2) / 2 \).

**Example 3.** In agent relation-based MC-net, four rules in Example 1 are expressed as follows: \( r_1 : (C_{1,2}, 2), r_2 : (\neg C_{3,4}, 1), r_3 : (\neg C_{1,4}, 1), r_4 : (C_{1,2} \land \neg C_{1,4} \land \neg C_{3,4}, 1) \). The four agents totally generate 12 hard clauses by the transitive laws.

The transitive laws introduced above could be refined in some cases. Take the example of a CSG problem with three rules: \( r_1 : a_1 \land a_2 \rightarrow 2, r_2 : a_3 \land a_4 \land \neg a_5 \rightarrow 1, r_3 : a_1 \land \neg a_6 \rightarrow 1 \). Agents \( a_3, a_4, \) and \( a_5 \) only appear in \( r_2 \) and have nothing to do with \( a_1, a_2, \) and \( a_6 \). Then we can combine \( a_3, a_4, \) and \( a_5 \) into a group, and make the rest three agents form the other group. Obviously, these two groups are independent of each other. Thus, the number of clauses for representing transitive laws is calculated within two groups respectively. In this example, the number of transitive laws is only 6, while the original one sums up to \( 6 \cdot 5 \cdot 4 / 2 = 60 \).

4.2 Encoding Positive Value (Embedded) Rules by Agent Relations

With EWPM-to-WPM transformation and the agent relations, rules in MC-nets and embedded MC-nets could be encoded in straightforward way. Specifically, encodings of positive value (embedded) rules are given in this subsection and those for negative value (embedded) rules are introduced in the next subsection.

**Definition 6:** (WPM encoding for positive value rules). For a positive value rule \( r_i : (S_i, w_i) (w_i > 0) \), where \( S_i = C_{1,2} \land \cdots \land C_{i,j} \land \neg C_{i,j+1} \land \cdots \land \neg C_{i,m} \), the following clauses are introduced, where \( u_i \) is a new Boolean variable.

(i) a weighted soft clause: \( (u_i, w_i) \), and
(ii) \((m - 1)\) hard clauses: \( \neg u_i \lor C_{1,2}, \ldots, \neg u_i \lor C_{1,j}, \neg u_i \lor \neg C_{1,j+1}, \ldots, \neg u_i \lor \neg C_{1,m} \).

Definition 6 is in accordance with EWPM-to-WPM transformation, which leads the following theorem to hold.

**Theorem 2:** The encoding given by Definition 6 with the transitive laws leads a MaxSAT solver to output the correct results of the CSG problem, which consists of a set of positive value rules.

**Proof 5:** If there is a coalition structure \( CS \), then we can make an assignment \( A \) which satisfies all hard clauses for the transitive laws so as to agree with \( CS \). Reversely, if there is an assignment \( A \) which satisfies all hard clauses for the transitive laws, then we obtain a coalition structure \( CS \).
which agrees with $A$. Thus, there is a one-to-one correspondence between a set of coalition structures and a set of assignments which satisfy all hard clauses for the transitive laws.

Definition 6. (i) and (ii) correspond to Definition 3. (i) and (iii), respectively, where $(u_i, w_i)$ is transformed from the positive formula $(S_i, w_i)$, and the $(m - 1)$ hard clauses are transformed from the hard formula $(u_i \rightarrow S_i, \infty)$. According to Theorem 1 and Proposition 1, Definition 3. (i) and (iii) guarantee the correctness of EWPM-to-WPM transformation for positive soft formulas. Therefore, MaxSAT solvers calculate the correct social welfare.

Definition 7: (WPM encoding for positive value embedded rules). For a positive value embedded rule $er : (S_0 \land S_1 \land \cdots \land S_k, w_{er}) (w_{er} > 0)$, the following clauses are introduced, where $u_{er}$ is a new Boolean variable.

(i) a weighted soft clause: $(u_{er}, w_{er})$, and
(ii) $(k + 1)$ propositional Boolean formulas that could be expanded to several hard clauses: $\neg u_{er} \lor S_0, \ldots, \neg u_{er} \lor S_1, \ldots, \neg u_{er} \lor S_k$.

Theorem 3: The encoding given by Definition 7 with the transitive laws leads a MaxSAT solver to output the correct results of the CSG problem, which consists of a set of positive value embedded rules.

We omit the proof for saving space, since it is basically identical to Theorem 2.

Example 4. In Example 1, a positive value rule $r_3 : a_4 \land \neg a_1 \rightarrow 1$ is encoded into a soft clause $(u_3, 1)$ and a hard clause $\neg u_3 \lor \neg C_{1,4}$. A positive value embedded rule $r_4 : a_1 \land a_2 \land \neg a_4 \land \neg a_3 \rightarrow 1$ is encoded into a soft clause $(u_4, 1)$ and three hard clauses: $\neg u_4 \lor C_{1,2}, \neg u_4 \lor \neg C_{1,4}, \neg u_4 \lor \neg C_{3,4}$. Other rules are treated in the similar way, as the agent relation-based encoding is independent of the relations between multiple rules. To solve the problem instance in Example 1, a total of 10 variables and 22 clauses are generated. These clauses include: 4 weighted soft clauses representing rules, 6 hard clauses encoded from conjunction formulas, and 12 hard clauses representing transitive laws.

4.3 Encoding Negative Value (Embedded) Rules by Agent Relations

If a rule is a negative value rule $r_1 : (S_i, w_i) (w_i < 0)$, as explained in previous section, the EWPM-to-WPM transformation referred to Definition 3. (i) and (ii), which derive the following encodings.

Definition 8: (WPM encoding for negative value rules). For each negative value rule $r_1 : (S_i, w_i) (w_i < 0)$, the following clauses are introduced, where $u_i$ is a new Boolean variable.

(i) a weighted soft clause: $(\neg u_i, w_i)$, and
(ii) a hard clause: $\neg S_i \lor u_i$.

The social welfare after encoding is $(-W_{neg})$ larger than the original one, where $W_{neg}$ is the sum of negative weights.

Theorem 4: The encoding given by Definition 8 with the transitive laws leads a MaxSAT solver to output the correct results of the CSG problem, which consists of a set of negative value rules.

Proof 6: Let $CS$ be a coalition structure. We can make an assignment $A$ which satisfies all hard clauses for the transitive laws so as to agree with $CS$.

Definition 8. (i) and (ii) correspond to Definition 3. (i) and (ii), respectively, where $(\neg u_i, -w_i)$ is transformed from the negative formula $(S_i, w_i)$, and the hard clause is transformed from the hard formula $(S_i \rightarrow w_i, \infty)$. According to Theorem 1 and Proposition 1, Definition 3. (i) and (ii) guarantee the correctness of EWPM-to-WPM transformation for negative soft formulas. Therefore, MaxSAT solvers calculate the correct social welfare.

Definition 9: (WPM encoding for negative value embedded rules). For each negative value embedded rule $er : (S_0 \land S_1 \land \cdots \land S_k, w_{er}) (w_{er} < 0)$, the following clauses are introduced, where $u_i$ is a new Boolean variable.

(i) a weighted soft clause: $(\neg u_{er}, -w_{er})$, and
(ii) a hard clause: $\neg S_0 \lor \neg S_1 \lor \cdots \lor \neg S_k \lor u_i$.

The social welfare after encoding is $(-W_{neg})$ larger than the original one, where $W_{neg}$ is the sum of negative weights.

Theorem 5: The encoding given by Definition 9 with the transitive laws leads a MaxSAT solver to output the correct results of the CSG problem, which consists of a set of negative value embedded rules.

We omit the proof of Theorem 5 since it is identical to that of Theorem 4.

Example 5. For $A = \{a_1, a_2, a_3, a_4\}$, assume there are four rules: $r_1 : a_1 \land a_2 \rightarrow 2, r_2 : a_3 \land \neg a_4 \rightarrow 1, r_3 : a_4 \land \neg a_1 \rightarrow 1, r_4 : a_1 \land a_2 \land \neg a_4 \land \neg a_3 \rightarrow -1$. A negative value rule $r_5 : a_4 \land \neg a_1 \rightarrow -1$ is encoded into a soft clause $(\neg u_3, 1)$ and a hard clause $C_{1,4} \lor u_3$. A negative value embedded rule $r_4 : a_1 \land a_2 \land \neg a_4 \land a_3 \land \neg a_3 \rightarrow -1$ is encoded into a soft clause $(\neg u_4, 1)$ and a hard clause $C_{1,2} \lor C_{1,4} \lor C_{3,4} \lor u_4$. To solve this problem, a total of 10 variables and 20 clauses are generated. These clauses include: 4 weighted soft clauses representing rules, 4 hard clauses encoded from conjunction formulas, and 12 hard clauses representing transitive laws.

5. Evaluation

This section evaluates the agent relation-based WPM approach (AWPM) and compare the performance of AWPM and the rule relation-based WPM (RWPM). The method of generating instances is summarized as follows [21]. First create a coalition with one random agent, then repeatedly add a new random agent with the probability of $\alpha$ until an agent is not added or the coalition includes all agents. Then,
we repeatedly add a new condition of each rule with probability $\beta$ until the condition is not added. The value of a rule is chosen between 0 and the number of agents in the rule, uniformly at random. In addition, for each coalition that contains more than one agent, we move an agent from positive to negative with the probability of $p$. Furthermore, we convert the value of a coalition from positive to negative with the probability of $q$. Throughout the experiment, we set $\alpha = 0.55, \beta = 0.15, p = 0.2, q = 0.2$, and $\#rules = \#agents$, ranging from 10 to 120. For each fixed $\#agents$, 100 problem instances are generated. All the tests were carried on a Core i5-2540 2.6GHz processor with 8GB RAM.

In order to select an appropriate MaxSAT solver, we evaluate several state-of-the-art MaxSAT solvers in our experiment, which can be classified into two categories. The one implements a branch and bound scheme, and the other one uses a state-of-the-art SAT solver as an inference engine, referred to as SAT-based solver. SAT-based solvers are further classified into satisfiability-based and unsatisfiability-based solvers. Among our tested solvers, Sat4j [27] and ShinMaxSat [28] are satisfiability-based solvers, while WPM1 [29] and Pwbo2.0 [30] are unsatisfiability-based solvers. Note that Pwbo2.0 is a parallel solver incorporating a satisfiability-based search into an unsatisfiability-based approach. Akmaxsat1.1 [31] and WmaxSatz [32] are branch and bound solvers. The evaluation results of these MaxSAT solvers are shown in Table 1. For each instance and solver, there is a time limit of 900 seconds. Number in bracket means the number of instances that were successfully solved within the time limit by the corresponding solver and is omitted in the table if the solver managed to solve all the 100 instances. At $\#agents = N$, if a solver fails to solve all instances within the time limit, then we terminate the solver and mark “/” for the corresponding solver with $\#agents > N$.

As can be seen from Table 1, Pwbo2.0, Sat4j and ShinMaxSat managed to solve all the problem instances with $\#agents$ ranging from 10 to 120. Among these three solvers, Pwbo2.0 had the most outstanding performance. By contrast, branch and bound-based and unsatisfiability-based solvers performed worst in our experiment, which could solve only a part of instances when $\#agents$ increases to 40. We briefly explain the reason why satisfiability-based solvers outperform the other two types of solvers as follows. In general, branch and bound-based solvers usually perform ineffectively as the size of problems increases. That is why Akmaxsat and WmaxSatz only managed to solve instances containing at most 40 agents. The SAT-based approach uses a SAT solver to solve a series of SAT instances obtained from a MaxSAT instance to be solved. Satisfiability-based solvers increase the lower bound of the MaxSAT solution while unsatisfiability-based solvers decrease the upper bound of the MaxSAT solution. Thus, satisfiability-based solvers prefer instances in which the number of soft clauses satisfied in the MaxSAT solution is relatively small. On the other hand, unsatisfiability-based solvers prefer instances in which the number is relatively large. In our experiment, the number is not so large, therefore, satisfiability-based solvers show good performance.

In what follows, we employed Pwbo2.0 as the solver to evaluate AWPM. In comparison, the solver chosen for RWPM is Sat4j, which performed best according to our previous experiment [24].

Figure 1 depicts the average computation time for solving problem instances by RWPM and AWPM. It is clear that AWPM is more time efficient than RWPM, and the superiority of AWPM becomes more remarkable as $\#agents$ goes up. Especially, when $\#agents$ reaches 120, the computation time for solving problem instances by AWPM approach is less than 4 seconds, while reaches around 9 seconds for RWPM.
We explain the reason why AWPM is more time efficient by investigating the number of Boolean variables and clauses in these two approaches, depicted in Figs. 2 and 3, respectively. Obviously, both the numbers of variables and clauses in RWPM are greater than that in AWPM. In general, the computation time goes up as the number of variables and clauses increases. That is why RWPM needs more computation time than AWPM in our experiment.

In the following, we discuss how the performances are influenced by \#rules and \#agents in the two WPM encodings. Given a CSG problem with \( m \) agents and \( n \) rules, the number of hard clauses for representing the transitive laws in AWPM and RWPM is \( m \cdot (m-1) \cdot (m-2)/2 \) and \( n \cdot (n-1) \cdot (n-2)/2 \) [24], respectively. In our experiments, \#rules = \#agents, which leads to the same number of clauses for representing the transitive laws. In this condition, the experiments demonstrate the superiority of AWPM. However, it is worth noting that, if \#agents is considerably larger than \#rules, the performance comparison of these two encodings may be inverse due to the overwhelming number of hard clauses for representing the transitive laws in AWPM.

6. Conclusion

In this paper, we presented an extension of WPM to deal with not only positive weights but also negative weights and applied the extended WPM (EWPM) to solve the CSG problem with externalities. Specifically, in order to solve EWPM instances with WPM solvers, we gave a transformation from EWPM to WPM and examined the relationship between their solutions. We theoretically proved that, for a WPM instance and its extension, the optimal solution of EWPM is always \(-W_{neg}\) larger than that of WPM, where \( W_{neg} \) is the sum of negative weights in the WPM instance. The development of EWPM paves a way for solving the MC-net-based CSG problem made up of a set of rules with both positive and negative weights. To be more specific, we exploited agent relations to encode the CSG problem into a set of Boolean propositional formulas, which could be formulated as an WPM instance with the EWPM-to-WPM transformation. Experimental results demonstrated the superiority of our agent relation-based encoding. To be more specific, problem instances with 120 agents (rules) could be solved within 4 seconds on average, while our previous rule relation-based WPM encoding, which excelled other state-of-the-art algorithms, required around 9 seconds to solve the same set of problem instances.

Acknowledgements

We wish to thank Prof. M. Yokoo and his students for introducing the MC-nets and embedded MC-nets to the CSG problem. This work is partially supported by JSPS KAKENHI Grant Number 21300054.

References

[1] X. Liao, M. Koshimura, H. Fujita, and R. Hasegawa, “Solving the coalition structure generation problem with MaxSAT,” Proc. ICTAI-12, pp.910–915, 2012.
[2] T. Rahwan, S. Ramchurn, N. Jennings, and A. Giovannucci, “An anytime algorithm for optimal coalition structure generation,” Journal of Artificial Intelligence Research, vol.34, pp.321–567, 2009.
[3] M. Tsvetovat and K. Sycara, “Customer coalitions in the electronic marketplace,” Proc. AGENTS, pp.263–264, 2000.
[4] V.D. Dang, R.K. Dash, A. Rogers, and N. Jennings, “Overlapping coalition formation for efficient data fusion in multi-sensor network,” Proc. AAAI, pp.635–640, 2006.
[5] T. Sandholm and V. Lesser, “Coalitions among computationally bounded agents,” Artificial Intelligence, vol.94, pp.99–137, 1997.
[6] M. Klusch and O. Shehory, “A polynomial kernel-oriented coalition formation algorithm for rational information agents,” Proc. IJCAI-96, pp.157–164, 1996.
[7] E. Catilina and R. Feinberg, “Market power and incentives to form research consortia,” Review of Industrial Organization, vol.28, pp.129–144, 2008.
[8] J.E. Plasmans, J. Engwerda, B. van Aarle, G. di Bartolomeo, and T. Michalak, Dynamic Modeling of Monetary and Fiscal Cooperation Among Nations, Springer, 2006.
[9] T. Rahwan, T. Michalak, M. Wooldridge, and N.R. Jennings, “Anytime coalition structure generation in multi-agent systems with positive or negative externalities,” Artificial Intelligence, vol.186, pp.95–122, 2012.
[10] T. Sandholm, K. Larson, M. Andersson, O. Shehory, and F. Tohmé, “Coalition structure generation with worst case guarantees,” Artificial Intelligence, vol.111, pp.209–238, 1999.
[11] M. Rothkopf, A. Pekec, and R. Harstad, “Computationally manageable combinatorial auctions,” Management Science, vol.44, pp.1131–1147, 1998.
[12] T. Rahwan and N. Jennings, “An improved dynamic program-
ming algorithm for coalition structure generation,” Proc. AAMAS, pp.1417–1420, 2008.

[13] T. Rahwan and N. Jennings, “Coalition structure generation: Dy-

namic programming meets anytime optimisation,” Proc. AAAI, pp.156–161, 2008.

[14] T. Rahwan, T. Michalak, and N. Jennings, “A hybrid algorithm for coalition structure generation,” Proc. AAAI-12, pp.1443–1449, 2012.

[15] T. Service and J. Adams, “Constant factor approximation algorithms for coalition structure generation,” Autonomous Agents and Multi-

Agent Systems, vol.23, pp.1–17, 2011.

[16] T. Michalak, A. Dowell, P. McBurney, and M. Wooldridge, “Opti-

mal coalition structure generation in partition function games,” Proc. European Conference on Artificial Intelligence, pp.388–392, 2008.

[17] B. Bikramjit and K. Landon, “Coalition structure generation in multi-agent systems with mixed externalities,” Proc. AAMAS, pp.175–182, 2010.

[18] S. Ieong and Y. Shoham, “Marginal contribution nets: A compact representation scheme for coalitional games,” Proc. EC, pp.193–202, 2005.

[19] T. Rahwan, T. Michalak, E. Elkind, P. Faliszewski, J. Sroka, M. Wooldridge, and N. Jennings, “Constrained coalition formation,” Proc. AAAI, pp.719–725, 2008.

[20] N. Ohta, V. Conitzer, R. Ichimura, Y. Sakurai, A. Iwasaki, and M. Yokoo, “Coalition structure generation utilizing compact character-

istic function representation,” Proc. CP, pp.623–638, 2009.

[21] S. Ueda, T. Hasegawa, N. Hashimoto, N. Ohta, A. Iwasaki, and M. Yokoo, “Handling negative value rules in MC-net-based coalition structure generation,” Proc. AAMAS, pp.795–804, 2012.

[22] S. Ueda, M. Kitaki, A. Iwasaki, and M. Yokoo, “Concise character-

istic function representations in coalitional games based on agent types,” Proc. IJCAI, pp.393–399, 2011.

[23] T. Michalak, D. Marciniak, M. Szamotulski, T. Rahwan, M. Wooldridge, P. McBurney, and N. Jennings, “A logic-based representation for coalitional games with externalities,” Proc. AAMAS, pp.125–132, 2010.

[24] X. Liao, M. Koshimura, H. Fujita, and R. Hasegawa, “MaxSAT encoding for MC-net-based coalition structure generation problem with externalities,” IEICE Trans. Inf. & Syst., vol.E97-D, no.7, pp.1781–1789, July 2014.

[25] C. Li, Z. Zhu, F. Manyà, and L. Simon, “Optimizing with mini-

mum satisfiability,” Artificial Intelligence, vol.190, pp.32–44, 2012.

[26] A. Biere, M. Heule, H. Maaren, and T. Walsh, Handbook of Satisfi-

ability, IOS Press, 2009.

[27] D. Le Berre and A. Parrain, “The Sat4j library, release 2.2,” Journal on Satisfiability, Boolean Modeling and Computation, vol.7, pp.59–64, 2010.

[28] K. Honjyo and T. Tanjo, “ShinMaxSat: A weighted parallel Max-SAT solver inspired by MiniSat+.” http://www.edu.kobe-u.ac.jp/
stc-tamlab/cspat/sms/

[29] C. Ansotegui, M. Bonet, and J. Levy, “Solving (weighted) partial MaxSAT through satisfiability testing,” Proc. SAT-09, pp.427–440, 2009.

[30] R. Martins, V. Manquinho, and I. Lynce, “Parallel search for maximum satisfiability,” AI Communications, vol.25, pp.75–95, 2012.

[31] A. Kugel, “Improved exact solver for the weighted Max-SAT prob-

lem,” Proc. POS-10, pp.15–27, 2010.

[32] C. Li, F. Manyà, N. Mohamedou, and J. Planes, “Exploiting cycle structures in Max-SAT,” Proc. SAT-09, pp.467–480, 2009.

Xiaojuan Liao received her B.E. from Uni-

versity of Electronic Science and Technology of China in 2009. She was supported by China Scholarship Council (CSC) from 2010 to 2013, pursuing the Ph.D. degree at Graduate School of Information Science and Electrical Engineering, Kyushu University, Fukuoka, Japan. Her research interests include maximum satisfiability and its applications.

Miyuki Koshimura received B.S. and M.S. degrees from University of Tsukuba in 1984 and 1986, respectively. He received D.E. degree from Kyushu University in 2002. He has been an Assistant Professor of Kyushu University since 2009. He was a research associate of Kyushu University from 1995 to 2009. His research interests include automated reasoning and its applications.

Hiroshi Fujita received B.S., M.E., and D.E. degrees from University of Tokyo in 1978, 1980, and 1991, respectively. He has been an Associate Professor of Kyushu University since 1996. His research interests include automated reasoning.

Ryuzo Hasegawa received B.E., M.E., and D.E. degrees from Kyushu University in 1972, 1974, and 1988, respectively. He has been a Professor of Kyushu University since 1995. His research interests include artificial intelligence, automated reasoning, and highly parallel computation mechanism.